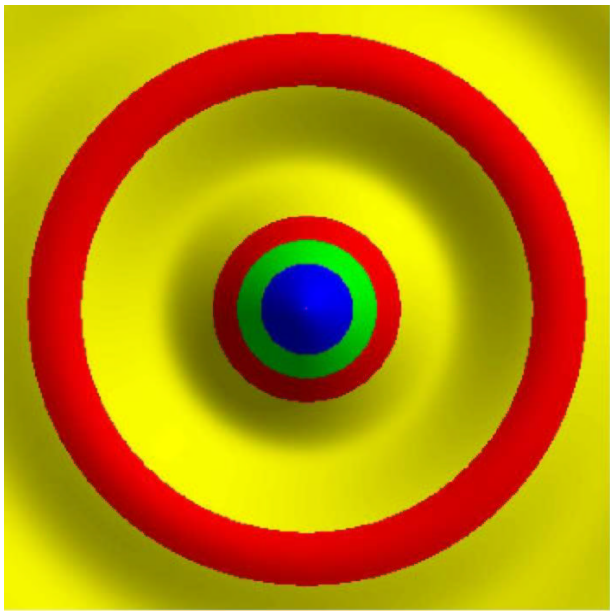
Listing of eps-Toolkit 2.0



Stefan Müller
stefan.mueller@fgan.de

FGAN,Wachtberg-Werthhoven,January 13, 2003

Contents

Chapter 1

Sources

1.1 Global Parameter

1.1.1 einit.m

```
ePath='./';%default directory of epstk-mfiles
%ePath='/usr/share/octave/site/m/epstk/';%default directory of epstk-mfiles

%eGhostscript=''; %no ghostscript
eGhostscript='gs'; %ghostscript for linux
%eGhostscript='"c:/gs/gs7.04/bin/gswin32.exe"'; %ghostscript for windows

%eGhostview=''; %no ghostview
%eGhostview='gv -scale -2'; %gv for linux
eGhostview='ghostview -magstep -2'; %ghostview for linux
%eGhostview='"c:/gs/gsview/gsview/gsview32.exe"'; %ghostview for windows

eFileName='epstkout.eps';                % default eps-outputfile
eFile=0; %fileId of eFileName
eUserUnit='mm';                          % or 'cm' or 'inch' or 'inch/72'
if strcmp(eUserUnit,'mm'),eFac=2.834646;
elseif strcmp(eUserUnit,'cm'),eFac=28.34646;
elseif strcmp(eUserUnit,'inch'),eFac=72;
else eFac=1;
end


%fonts  (standard fonts of postscript)
eFonts=[
'Times-Roman             ';      % font number 1
'Times-Italic            ';      % font number 2
'Times-Bold              ';      % font number 3
'Times-BoldItalic        ';      % font number 4
'Helvetica               ';      % font number 5
'Helvetica-Oblique       ';      % font number 6
'Helvetica-Bold          ';      % font number 7
'Helvetica-BoldOblique   ';      % font number 8
'Courier                 ';      % font number 9
'Courier-Oblique         ';      % font number 10
'Courier-Bold            ';      % font number 11
```

```
'Courier-BoldOblique     ';      % font number 12
'Symbol                  '];     % font number 13

%colormaps
eColorMaps=[...
  %0 black->white                                get it with ecolors(0)
  0 0.0 0.0 0.0;0 1.0 1.0 1.0;

  %1 red->yellow                                 get it with ecolors(1)
  1 0.4 0.0 0.0;1 1.0 0.0 0.0;1 1.0 1.0 0.0;

  %2 violet->blue->yellow->red                   get it with ecolors(2)
  2 0.4 0.0 0.4;2 0.0 0.0 1.0;2 0.0 1.0 1.0;2 1.0 1.0 0.0;2 1.0 0.0 0.0;

  %3 blue->yellow->red                           get it with ecolors(3)
  3 0.0 0.0 0.4;3 0.0 0.0 1.0;3 0.0 1.0 1.0;3 1.0 1.0 0.0;3 1.0 0.0 0.0;

  %4 black->violet->blue->yellow->red            get it with ecolors(4)
  4 0.1 0.0 0.1;4 0.4 0.0 0.4;4 0.0 0.0 1.0;4 0.0 1.0 1.0;
  4 1.0 1.0 0.0;4 1.0 0.0 0.0;

  %5 green->yellow->red->violet                  get it with ecolors(5)
  5 0.0 0.4 0.0;5 0.0 1.0 0.0;5 1.0 1.0 0.0;5 1.0 1.0 0.0;
  5 1.0 0.0 0.0;5 0.5 0.0 0.2;

  %6 white->black->violet->blue->yellow->red   get it with ecolors(6)
  6 1.0 1.0 1.0;6 0.0 0.0 0.0;6 0.4 0.0 0.4;6 0.0 0.0 1.0;
  6 0.0 1.0 1.0;6 0.0 1.0 0.0;6 1.0 1.0 0.0;6 1.0 0.0 0.0;

  %7 grey->yellow->red                           get it with ecolors(7)
  7 1.0 1.0 0.9;7 1.0 1.0 0.0;7 1.0 0.0 0.0;

  %8 white->blue->grey->red->white               get it with ecolors(8)
  8 1.0 1.0 1.0;8 0.2 0.2 1.0;8 0.5 0.5 0.5;8 1.0 0.2 0.2;
  8 1.0 1.0 1.0;
];


% page
ePageWidth=210; % mm A3=297 A4=210 A5=148
ePageHeight=297;% mm A3=420 A4=297 A5=210
ePageOrientation=0; % 0=Portrait 1=Landscape 2=Upside-down 3=Seaside
ePageReflection=0; %  1=on 0=off  reflect page
eXScaleFac=1; % 1=no resize  0.5=50% reduce  2=200% enlarge
eYScaleFac=1; % 1=no resize  0.5=50% reduce  2=200% enlarge

% window
eWinWidth=180; % mm
eWinHeight=250; % mm
eWinFrameVisible=0; % 1=on 0=off   draw frame around window
eWinFrameLineWidth=0.3; % mm
eWinGridVisible=0; % 1=on 0=off   draw grid of window
eWinTimeStampVisible=0; % 1=on 0=off  print time stamp outside of frame
eWinTimeStampFont=1; % font number 1=TimesRoman select font of time stamp
```

```
eWinTimeStampFontSize=1.5; % mm

%plot area
ePlotAreaPos=[40 100]; % x y position of left bottom corner of plot area
ePlotAreaWidth=100; % mm
ePlotAreaHeight=100; % mm
ePlotAreaXValueStart=0; % value range of x-axis
ePlotAreaXValueEnd=100;
ePlotAreaYValueStart=0; % value range of y-axis
ePlotAreaYValueEnd= 100;
ePlotLineNo=0;

%polar plot area
ePolarPlotAreaCenterPos=[90 160]; % x y position of Center of polar plot area
ePolarPlotAreaRadMin=10; % mm
ePolarPlotAreaRadMax=50; % mm
ePolarPlotAreaAngStart=0; % deg, 0=east 90=north 180=west 270=south
ePolarPlotAreaAngEnd=360; % deg, 0=east 90=north 180=west 270=south
ePolarPlotAreaValStart=0; % value range of radius-axis
ePolarPlotAreaValEnd=100;
ePolarPlotLineNo=0;
ePieSliceNo=0;

% title obove plots
ePlotTitleDistance=20; % mm
ePlotTitleFontSize=6; % mm
ePlotTitleText=''; % text string
ePlotTitleTextFont=1; % font number   1=TimesRoman

% grid
eXGridLineWidth=0.1; % mm
eXGridColor=[0 0 0]; % [r g b]   [0 0 0]=black  [1 1 1]=white
eXGridDash=0.5; % mm    0=solid line >0=dash length
eXGridVisible=0; %
eYGridLineWidth=0.1; % mm
eYGridColor=[0 0 0]; % [r g b]   [0 0 0]=black  [1 1 1]=white
eYGridDash=0.5; % mm    0=solid line >0=dash length
eYGridVisible=0; % 0=off 1=on

% polar grid
ePolarRadiusGridLineWidth=0.1; % mm
ePolarRadiusGridColor=[0 0 0]; %  [r g b]   [0 0 0]=black  [1 1 1]=white
ePolarRadiusGridDash=1; % mm   0=solid line >0=dash length
ePolarRadiusGridVisible=1; %  0=off 1=on
ePolarAngleGridLineWidth=0.1; % mm
ePolarAngleGridColor=[0 0 0]; %   [r g b]   [0 0 0]=black  [1 1 1]=white
ePolarAngleGridDash=2; % mm   0=solid line >0=dash length
ePolarAngleGridVisible=1; %  0=off 1=on

% axes
eAxesColor=[0 0 0]; %   [r g b]   [0 0 0]=black  [1 1 1]=white
eAxesLineWidth=0.3; % mm
eAxesTicShortLength=1.5; % mm
eAxesTicLongLength=3; % mm
```

```
eAxesTicLongMaxN=9; % max. number of long Tics
eAxesValueSpace=1; % mm
eAxesValueFontSize=4; % mm
eAxesLabelFontSize=4; % mm
eAxesLabelTextFont=5; % font number   5=Helvetica
eAxesCrossOrigin=0; % 0=off 1=on 2=on and with arrows

% scale vectors:if start=0 and end=0 then autorange,if step=0 then autoscale
% south axis
eXAxisSouthScale=[0 0 0]; % [start step end]
eXAxisSouthScaleType=0; % 0=linear 1=classes 2=log10
eXAxisSouthValueFormat=-1; %  n digits after decimal point,-1=auto
eXAxisSouthValueVisible=1; %  0=off 1=on
eXAxisSouthValuePos=[0 0]; %  value positions after drawing of axis
eXAxisSouthLabelDistance=2; % mm  label distance from axis
eXAxisSouthLabelText='';
eXAxisSouthVisible=1; %  0=off 1=on

% north axis
eXAxisNorthScale=[0 0 0]; % [start step end]
eXAxisNorthScaleType=0; % 0=linear 1=classes 2=log10
eXAxisNorthValueFormat=-1;  %  n digits after decimal point,-1=auto
eXAxisNorthValueVisible=1;  %  0=off 1=on
eXAxisNorthValuePos=[0 0]; %  value positions after drawing of axis
eXAxisNorthLabelDistance=2;  % mm  label distance from axis
eXAxisNorthLabelText='';
eXAxisNorthVisible=1; %  0=off 1=on

% west axis
eYAxisWestScale=[0 0 0]; % [start step end]
eYAxisWestScaleType=0; % 0=linear 1=classes 2=log10
eYAxisWestValueFormat=-1;   %  n digits after decimal point,-1=auto
eYAxisWestValueVisible=1;   %  0=off 1=on
eYAxisWestValuePos=[0 0]; %  value positions after drawing of axis
eYAxisWestLabelDistance=6;   % mm  label distance from axis
eYAxisWestLabelText='';
eYAxisWestVisible=1; %  0=off 1=on

% east axis
eYAxisEastScale=[0 0 0]; % [start step end]
eYAxisEastScaleType=0; % 0=linear 1=classes 2=log10
eYAxisEastValueFormat=-1;  %  n digits after decimal point,-1=auto
eYAxisEastValueVisible=1;  %  0=off 1=on
eYAxisEastValuePos=[0 0]; %  value positions after drawing of axis
eYAxisEastLabelDistance=6;  % mm  label distance from axis
eYAxisEastLabelText='';
eYAxisEastVisible=1; %  0=off 1=on

%polar radius axis
ePolarAxisRadScale=[0 0 0]; % [start step end]
ePolarAxisRadScaleType=0; % 0=linear 1=classes 2=log10
ePolarAxisRadValueFormat=-1;  %  n digits after decimal point,-1=auto
ePolarAxisRadValueVisible=3;  %  0=off,1=RadStart on,2=RadEnd on,3=Start+End on
ePolarAxisRadValuePos=[0 0]; %  value positions after drawing of axis
```

```
ePolarAxisRadVisible=1; %  0=off,1=RadStart on,2=RadEnd on,3=Start+End on

%polar angle axis
ePolarAxisAngScale=[0 0 0]; % [start step end]
ePolarAxisAngValueFormat=-1;  %  n digits after decimal point,-1=auto
ePolarAxisAngValueVisible=1;  %  0=off 1=on
ePolarAxisAngValueAngle=0;  %  angle positions of values after drawing of axis
ePolarAxisAngVisible=1;  %  0=off 1=on

%plot line
ePlotLineColor=[0 0 0]; %   [r g b]   [0 0 0]=black  [1 1 1]=white
ePlotLineDash=0;  % mm 0=solid line,>0=dash length,<0=fill line,'abc'=symbol abc
ePlotLineWidth=0.3; % mm
ePlotLineInterpolation=0; % 0=off 1=on

%plot legend
ePlotLegendPos=[-15 -20];% position relativ to left bottom corner of plot area
ePlotLegendFontSize=4; % mm
ePlotLegendDistance=100; % in percent, depend on ePlotLegendFontSize
ePlotLegendTextFont=1;  % font number   1=TimesRoman

%image
eImageDefaultColorMap=0; % number of default map of eColorMaps
eImageFrameVisible=0; % 0=off 1=on

%image legend
eImageLegendPos=[0 -25]; % position relativ to left bottom corner of plot area
eImageLegendWidth=0; % mm 0=ePlotAreaWidth
eImageLegendHeight=5; % mm
eImageLegendScale=[0 0 0]; % [start step end]
eImageLegendScale=[0 0 0]; % [start step end]
eImageLegendScaleType=0; % 0=linear 1=classes 2=log10
eImageLegendValueFormat=-1; %  n digits after decimal point,-1=auto
eImageLegendValueVisible=1;  %  0=off 1=on
eImageLegendValuePos=[0 0];  %  value positions after drawing of axis
eImageLegendLabelDistance=2; % mm
eImageLegendLabelText='';
eImageLegendVisible=1; %  0=off 1=on

%parameter
eParamPos=[30 65];  % absolut position of window
eParamFontSize=4; % mm
eParamLineDistance=100; % in percent, depend on eParamFontSize
eParamTextValueDistance=100; % in percent, depend on eParamFontSize
eParamText='';
eParamTextFont=3;  % font number   1=TimesRoman
eParamValue='';
eParamValueFont=11;  % font number   9=Courier

%line
eLineWidth=0.3; % mm
eLineColor=[0 0 0]; %   [r g b]   [0 0 0]=black  [1 1 1]=white
eLineDash=0;  % mm   0=solid line   >0=dash length
```

```
%text
eTextFont=1;  % font number   1=TimesRoman
eTextFontSize=4; % mm
eTextPos=[30 eWinHeight-eTextFontSize]; % inital position is left top of window
eTextColor=[0 0 0]; %   [r g b]   [0 0 0]=black  [1 1 1]=white
eTextAlignment=1; % 1=right 0=center -1=left
eTextRotation=0; % in deg
eTextLimitWord=' '; % character to limit words
eTextLimitPara=setstr(10); % character  to limit paragraphs, setstr(10)=linefeed

%text box
eTextBoxFeedLine=0;  % mm 0=auto else fix linefeed
eTextBoxFeedPara=0;  % mm space between paragraphs
eTextBoxSpaceNorth=0;  % mm  space between text and the north border of box
eTextBoxSpaceSouth=0;  % mm  space between text and the south border of box
eTextBoxSpaceWest=0;  % mm  space between text and the north border of box
eTextBoxSpaceEast=0;  % mm  space between text and the south border of box

%contour
eContourLineWidth=0.2; % mm
eContourLineColor=[0 0 0]; %   [r g b]   [0 0 0]=black  [1 1 1]=white
eContourLineDash=0; % mm    0=solid line   >0=dash length
eContourScale=[0 0 0]; % [start step end]
eContourValueVisible=0;  %  0=off 1=on
eContourValueFormat=-1;  %  n digits after decimal point,-1=auto
eContourValueFont=5; % font number   5=Helvetica
eContourValueFontSize=2; % mm
eContourValueDistance=2+eContourLineWidth/2; % mm
eContourLevelsMaxN=10; % max. number of isolevels if autoscaling on

%table
eTabBackgroundColor=[-1 0 0]; %   [r g b] if r<0 then transparent
eTabFrameVisible=1;  %  0=off 1=on
eTabFrameLineWidth=eLineWidth; % mm
eTabFrameColor=[0 0 0]; %   [r g b]   [0 0 0]=black  [1 1 1]=white
eTabFrameDash=0; % mm    0=solid line   >0=dash length
eTabXLineVisible=1;  %  0=off 1=on
eTabXLineWidth=eLineWidth; % mm
eTabXLineColor=[0 0 0]; %   [r g b]   [0 0 0]=black  [1 1 1]=white
eTabXLineDash=0; % mm    0=solid line   >0=dash length
eTabYLineVisible=1;  %  0=off 1=on
eTabYLineWidth=eLineWidth; % mm
eTabYLineColor=[0 0 0]; %   [r g b]   [0 0 0]=black  [1 1 1]=white
eTabYLineDash=0; % mm    0=solid line   >0=dash length
```

1.2 User Functions

1.2.1 eaxes.m

```
%%NAME
%%  eaxes - draw scaled axes around plot area
%%
```

```
%%SYNOPSIS
%%  eaxes ([xAxisSouthScale,yAxisWestScale[,xAxisNorthScale,yAxisEastScale]])
%%
%%PARAMETER(S)
%%  xAxisSouthScale   scale vector of south axis [start step end]
%%  yAxisWestScale    scale vector of west axis  [start step end]
%%  xAxisNorthScale   scale vector of north axis [start step end]
%%  yAxisEastScale    scale vector of east axis  [start step end]
%%
%%    special cases of scale vectors are:
%%      if start=0 and end=0 then autorange=on
%%      if step=0 then autoscale=on
%%    (default scale vector=[0 0 0])
%%
%%GLOBAL PARAMETER(S)
%%  ePlotAreaXValueStart
%%  ePlotAreaXValueEnd
%%  ePlotAreaYValueStart
%%  ePlotAreaYValueEnd
%%  ePlotAreaPos
%%  ePlotAreaWidth
%%  ePlotAreaHeight
%%  eAxesValueFontSize
%%  eAxesValueSpace
%%  eAxesColor
%%  eAxesLineWidth
%%  eAxesTicShortLength
%%  eAxesTicLongLength
%%  eAxesTicLongMaxN
%%  eAxesCrossOrigin
%%  eAxesValueSpace
%%  eAxesLabelFontSize
%%  eAxesLabelTextFont
%%  eXAxis(South|West|East|North)Scale
%%  eXAxis(South|West|East|North)ScaleType
%%  eXAxis(South|West|East|North)Visible
%%  eXAxis(South|West|East|North)ValueFormat
%%  eXAxis(South|West|East|North)ValueVisible
%%  eXAxis(South|West|East|North)LabelText
%%  eXAxis(South|West|East|North)LabelDistance
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eaxes(xAxisSouthScale,yAxisWestScale,xAxisNorthScale,yAxisEastScale)
  if nargin~=0 & nargin~=2 & nargin~=4
    eusage('eaxes([xAxisSouthScale,yAxisWestScale[,xAxisNorthScale,yAxisEastScale]])');
  end
  eglobpar;
  if nargin>0
    eXAxisSouthScale=xAxisSouthScale;
    if eXAxisSouthScaleType==2
      if (eXAxisSouthScale(1)>0)&(eXAxisSouthScale(3)>0)
        eXAxisSouthScale(1)=log10(eXAxisSouthScale(1));
        eXAxisSouthScale(3)=log10(eXAxisSouthScale(3));
      else
```

```
      error('xValues<=0 for log scale');
    end
  end
  if eXAxisSouthScale(1)==eXAxisSouthScale(3)
    eXAxisSouthScale(3)=eXAxisSouthScale(1)+1;
    eXAxisSouthScale(2)=0;
  end
  ePlotAreaXFac=ePlotAreaWidth*eFac/...
    (eXAxisSouthScale(3)-eXAxisSouthScale(1));

  eYAxisWestScale=yAxisWestScale;
  if eYAxisWestScaleType==2
    if (eYAxisWestScale(1)>0)&(eYAxisWestScale(3)>0)
      eYAxisWestScale(1)=log10(eYAxisWestScale(1));
      eYAxisWestScale(3)=log10(eYAxisWestScale(3));
    else
      error('yValues<=0 for log scale');
    end
  end
  if eYAxisWestScale(1)==eYAxisWestScale(3)
    eYAxisWestScale(3)=eYAxisWestScale(1)+1;
    eYAxisWestScale(2)=0;
  end
  ePlotAreaYValueEnd=eYAxisWestScale(3);
  ePlotAreaYFac=ePlotAreaHeight*eFac/...
    (eYAxisWestScale(3)-eYAxisWestScale(1));
end
if nargin>2
  eXAxisNorthScale=xAxisNorthScale;
  eYAxisEastScale=yAxisEastScale;
end
if eXAxisSouthScale(1)~=eXAxisSouthScale(3)
  ePlotAreaXValueStart=eXAxisSouthScale(1);
  ePlotAreaXValueEnd=eXAxisSouthScale(3);
end
if eYAxisWestScale(1)~=eYAxisWestScale(3)
  ePlotAreaYValueStart=eYAxisWestScale(1);
  ePlotAreaYValueEnd=eYAxisWestScale(3);
end
if eXAxisNorthScale(1)~=eXAxisNorthScale(3)
  if eXAxisNorthScaleType==2
    if (eXAxisNorthScale(1)>0)&(eXAxisNorthScale(3)>0)
      eXAxisNorthScale(1)=log10(eXAxisNorthScale(1));
      eXAxisNorthScale(3)=log10(eXAxisNorthScale(3));
    else
      error('xValues<=0 for log scale');
    end
  end
  eXAxisNorthValueStart=eXAxisNorthScale(1);
  eXAxisNorthValueEnd=eXAxisNorthScale(3);
else
  eXAxisNorthValueStart=ePlotAreaXValueStart;
  eXAxisNorthValueEnd=ePlotAreaXValueEnd;
  eXAxisNorthScale=eXAxisSouthScale;
```

```
  eXAxisNorthScaleType=eXAxisSouthScaleType;
end
if eYAxisEastScale(1)~=eYAxisEastScale(3)
  if eYAxisEastScaleType==2
    if (eYAxisEastScale(1)>0)&(eYAxisEastScale(3)>0)
      eYAxisEastScale(1)=log10(eYAxisEastScale(1));
      eYAxisEastScale(3)=log10(eYAxisEastScale(3));
    else
      error('yValues<=0 for log scale');
    end
  end
  eYAxisEastValueStart=eYAxisEastScale(1);
  eYAxisEastValueEnd=eYAxisEastScale(3);
else
  eYAxisEastValueStart=ePlotAreaYValueStart;
  eYAxisEastValueEnd=ePlotAreaYValueEnd;
  eYAxisEastScale=eYAxisWestScale;
  eYAxisEastScaleType=eYAxisWestScaleType;
end


if sign(ePlotAreaXValueStart*ePlotAreaXValueEnd)<0 &...
   sign(ePlotAreaYValueStart*ePlotAreaYValueEnd)<0 & eAxesCrossOrigin
  eXAxisSouthVisible=1;
  eYAxisWestVisible=1;
  eXAxisNorthVisible=0;
  eYAxisEastVisible=0;
  xAxisOffset=-ePlotAreaYValueStart*ePlotAreaYFac;
  yAxisOffset=-ePlotAreaXValueStart*ePlotAreaXFac;
  if eAxesCrossOrigin==2
    edsymbol('arrow','spire.psd',eAxesLineWidth,eAxesLineWidth,0,0,0,eAxesColor);
    esymbol(ePlotAreaPos(1)+ePlotAreaWidth,...
        ePlotAreaPos(2)+xAxisOffset/eFac,'arrow');
    esymbol(ePlotAreaPos(1)+yAxisOffset/eFac,...
        ePlotAreaPos(2)+ePlotAreaHeight,'arrow',1,1,90);
  end
else
  xAxisOffset=0;
  yAxisOffset=0;
end
% scale and draw axes
if eXAxisSouthVisible
  eaxis(ePlotAreaPos(1),ePlotAreaPos(2),...
        ePlotAreaWidth,'s',...
        [ePlotAreaXValueStart eXAxisSouthScale(2)  ePlotAreaXValueEnd],...
        0,eAxesColor,-xAxisOffset);
end
if eXAxisNorthVisible
  eaxis(ePlotAreaPos(1),ePlotAreaPos(2)+ePlotAreaHeight,...
        ePlotAreaWidth,'n',...
        [eXAxisNorthValueStart eXAxisNorthScale(2)  eXAxisNorthValueEnd],...
        0,eAxesColor,0);
end
if eYAxisWestVisible
```

```
    eaxis(ePlotAreaPos(1),ePlotAreaPos(2),...
          ePlotAreaHeight,'w',...
          [ePlotAreaYValueStart eYAxisWestScale(2)  ePlotAreaYValueEnd],...
          0,eAxesColor,-yAxisOffset);
  end
  if eYAxisEastVisible
    eaxis(ePlotAreaPos(1)+ePlotAreaWidth,ePlotAreaPos(2),...
          ePlotAreaHeight,'e',...
          [eYAxisEastValueStart eYAxisEastScale(2)  eYAxisEastValueEnd],...
          0,eAxesColor,0);
  end

  %print labels
  scaleSpace=eAxesTicLongLength+eAxesValueSpace+eAxesValueFontSize;
  if strcmp(eXAxisSouthLabelText,'')~=1
    etext(eXAxisSouthLabelText,...
          ePlotAreaPos(1)+ePlotAreaWidth/2,...
          ePlotAreaPos(2)-scaleSpace-...
          eAxesLabelFontSize*0.72-eXAxisSouthLabelDistance,...
          eAxesLabelFontSize,0,eAxesLabelTextFont,0,eAxesColor);
  end
  if strcmp(eXAxisNorthLabelText,'')~=1
    etext(eXAxisNorthLabelText,...
          ePlotAreaPos(1)+ePlotAreaWidth/2,...
          ePlotAreaPos(2)+ePlotAreaHeight+...
          scaleSpace+eXAxisNorthLabelDistance,...
          eAxesLabelFontSize,0,eAxesLabelTextFont,0,eAxesColor);
  end
  if strcmp(eYAxisWestLabelText,'')~=1
    etext(eYAxisWestLabelText,...
          ePlotAreaPos(1)-scaleSpace-eYAxisWestLabelDistance,...
          ePlotAreaPos(2)+ePlotAreaHeight/2,...
          eAxesLabelFontSize,0,eAxesLabelTextFont,90,eAxesColor);
  end
  if strcmp(eYAxisEastLabelText,'')~=1
    etext(eYAxisEastLabelText,...
          ePlotAreaPos(1)+ePlotAreaWidth+scaleSpace+...
          eAxesLabelFontSize/2+eYAxisEastLabelDistance,...
          ePlotAreaPos(2)+ePlotAreaHeight/2,...
          eAxesLabelFontSize,0,eAxesLabelTextFont,90,eAxesColor);
  end
```

1.2.2 eaxespol.m

```
%%NAME
%%  eaxespol - draw scaled axes and arc around polar plot area
%%
%%SYNOPSIS
%%  eaxespol([axisRadiusScale,axisAngleScale])
%%
%%PARAMETER(S)
%%  axisRadiusScale   scale vector of radius axis  [start step end]
%%  axisAngleScale    scale vector of angle circle [start step end]
```

```
%%
%%    special cases of scale vectors are:
%%      if start=0 and end=0 then autorange=on
%%      if step=0 then autoscale=on
%%
%%GLOBAL PARAMETER(S)
%%  ePolarPlotAreaCenterPos
%%  ePolarPlotAreaRadMin
%%  ePolarPlotAreaRadMax
%%  ePolarPlotAreaAngStart
%%  ePolarPlotAreaAngEnd
%%  ePolarPlotAreaValStart
%%  ePolarPlotAreaValEnd
%%  ePolarAxisRadScale
%%  ePolarAxisRadVisible
%%  ePolarAxisRadValueFormat
%%  ePolarAxisRadValueVisible
%%  ePolarAxisRadPos
%%  ePolarAxisAngScale
%%  ePolarAxisAngVisible
%%  ePolarAxisAngValueFormat
%%  ePolarAxisAngValueVisible
%%  ePolarAxisAngValueAngle
%%  eAxesValueFontSize
%%  eAxesColor
%%  eAxesLineWidth
%%  eAxesTicShortLength
%%  eAxesTicLongLength
%%  eAxesTicLongMaxN
%%  eAxesValueSpace
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eaxespol(axisRadiusScale,axisAngleScale)
  if nargin~=0 & nargin~=2
    eusage('egridpol([axisRadiusScale,axisAngleScale])');
  end
  eglobpar;
  if nargin==2
    ePolarAxisRadScale=axisRadiusScale;
    ePolarPlotAreaValStart=ePolarAxisRadScale(1);
    ePolarPlotAreaValEnd=ePolarAxisRadScale(3);
    ePolarAxisAngScale=axisAngleScale;
    yRange=ePolarPlotAreaValEnd-ePolarPlotAreaValStart;
    if yRange==0
      yRange=1;
    end
    ePolarPlotAreaFac=(ePolarPlotAreaRadMax-ePolarPlotAreaRadMin)*...
                      eFac/yRange;
  end
  if ePolarAxisRadScale(1)~=ePolarAxisRadScale(3)
    ePolarPlotAreaValStart=ePolarAxisRadScale(1);
    ePolarPlotAreaValEnd=ePolarAxisRadScale(3);
  end
  % scale and draw axes
```

```
  saveTypeE=eYAxisEastScaleType;
  saveVisibleE=eYAxisEastValueVisible;
  saveValuePosE=eYAxisEastValuePos;
  eYAxisEastScaleType=ePolarAxisRadScaleType;
  saveTypeW=eYAxisWestScaleType;
  saveVisibleW=eYAxisWestValueVisible;
  saveValuePosW=eYAxisWestValuePos;
  eYAxisWestScaleType=ePolarAxisRadScaleType;
  ePolarAxisRadValuePos=[0 0];
  if rem(ePolarAxisRadVisible,2)
    angle=rem(ePolarPlotAreaAngStart,360);
    if (angle>=0) & (angle<=180)
      eYAxisEastValueVisible=rem(ePolarAxisRadValueVisible,2);
      scale=[ePolarPlotAreaValStart...
             ePolarAxisRadScale(2)...
             ePolarPlotAreaValEnd];
      rr=ePolarPlotAreaRadMin;
      aa=-90;
      eaxis(ePolarPlotAreaCenterPos(1)+cos(pi*angle/180)*rr,...
          ePolarPlotAreaCenterPos(2)+sin(pi*angle/180)*rr,...
          ePolarPlotAreaRadMax-ePolarPlotAreaRadMin,'e',...
          scale,angle+aa,eAxesColor,0);
      ePolarAxisRadValuePos=eYAxisEastValuePos;
    else
      eYAxisWestValueVisible=rem(ePolarAxisRadValueVisible,2);
      scale=[ePolarPlotAreaValEnd...
             -ePolarAxisRadScale(2)...
             ePolarPlotAreaValStart];
      rr=ePolarPlotAreaRadMax;
      aa=90;
      eaxis(ePolarPlotAreaCenterPos(1)+cos(pi*angle/180)*rr,...
          ePolarPlotAreaCenterPos(2)+sin(pi*angle/180)*rr,...
          ePolarPlotAreaRadMax-ePolarPlotAreaRadMin,'w',...
          scale,angle+aa,eAxesColor,0);
      ePolarAxisRadValuePos=eYAxisWestValuePos;
    end
  end

  if fix(ePolarAxisRadVisible/2)
    nPos=size(ePolarAxisRadValuePos,1);
    angle=rem(ePolarPlotAreaAngEnd,360);
    if (angle>=0) & (angle<=180)
      eYAxisWestValueVisible=fix(ePolarAxisRadValueVisible/2);
      scale=[ePolarPlotAreaValStart...
             ePolarAxisRadScale(2)...
             ePolarPlotAreaValEnd];
      rr=ePolarPlotAreaRadMin;
      aa=-90;
      eaxis(ePolarPlotAreaCenterPos(1)+cos(pi*angle/180)*rr,...
          ePolarPlotAreaCenterPos(2)+sin(pi*angle/180)*rr,...
          ePolarPlotAreaRadMax-ePolarPlotAreaRadMin,'w',...
          scale,angle+aa,eAxesColor,0);
      if nPos==size(eYAxisWestValuePos,1)
        ePolarAxisRadValuePos=[ePolarAxisRadValuePos eYAxisWestValuePos];
```

```
    else
      ePolarAxisRadValuePos=eYAxisWestValuePos;
    end
  else
    eYAxisEastValueVisible=fix(ePolarAxisRadValueVisible/2);
    scale=[ePolarPlotAreaValEnd...
           -ePolarAxisRadScale(2)...
           ePolarPlotAreaValStart];
    rr=ePolarPlotAreaRadMax;
    aa=90;
    eaxis(ePolarPlotAreaCenterPos(1)+cos(pi*angle/180)*rr,...
        ePolarPlotAreaCenterPos(2)+sin(pi*angle/180)*rr,...
        ePolarPlotAreaRadMax-ePolarPlotAreaRadMin,'e',...
        scale,angle+aa,eAxesColor,0);
    if nPos==size(eYAxisEastValuePos,1)
      ePolarAxisRadValuePos=[ePolarAxisRadValuePos eYAxisEastValuePos];
    else
      ePolarAxisRadValuePos=eYAxisEastValuePos;
    end
  end
end

eYAxisEastValueVisibleE=saveVisibleE;
eYAxisEastValuePos=saveValuePosE;
eYAxisEastScaleType=saveTypeE;
eYAxisWestValueVisibleW=saveVisibleW;
eYAxisWestValuePos=saveValuePosW;
eYAxisWestScaleType=saveTypeW;

if ePolarAxisAngVisible
  maxValues=fix(20*(ePolarPlotAreaAngEnd-ePolarPlotAreaAngStart)/360);
  ePolarAxisAngValueAngle=escalepa(eFile,...
          ePolarPlotAreaCenterPos(1)*eFac,...
          ePolarPlotAreaCenterPos(2)*eFac,...
          ePolarPlotAreaRadMin*eFac,...
          ePolarPlotAreaRadMax*eFac,...
          ePolarPlotAreaAngStart,...
          ePolarPlotAreaAngEnd,...
          ePolarAxisAngScale(1),...
          ePolarAxisAngScale(2),...
          ePolarAxisAngScale(3),...
          ePolarAxisAngValueFormat,...
          ePolarAxisAngValueVisible,...
          eAxesValueFontSize*eFac,...
          eAxesLineWidth*eFac,...
          eAxesTicShortLength*eFac,...
          eAxesTicLongLength*eFac,...
          maxValues,...
          eAxesValueSpace*eFac,...
          eAxesColor);
end
```

1.2.3 eaxis.m

```
%%NAME
%%  eaxis - draw scaled axis
%%
%%SYNOPSIS
%%  eaxis(xPos,yPos,length,type,scale[,angle[,color[,offset]]])
%%
%%PARAMETER(S)
%%  xPos         x-value of start position of axis
%%  yPos         y-value of start position of axis
%%  length       length  of axis
%%  type       orientation of scaling  'w'=west, 'e'=east, 's'=south, 'n'=north
%%  scale      vector of scaling, [startValue stepValue endValue]
%%  angle      angle to rotate axis
%%  color      color of axis
%%  offset     offset of position
%%
%%GLOBAL PARAMETER(S)
%%  eAxesColor
%%  eAxesValueFontSize
%%  eAxesValueSpace
%%  eAxesLineWidth
%%  eAxesTicShortLength
%%  eAxesTicLongLength
%%  eAxesTicLongMaxN
%%  eXAxisSouthValueFormat
%%  eYAxisWestValueFormat
%%  eXAxisNorthValueFormat
%%  eYAxisEastValueFormat
%%  eXAxisSouthValueVisible
%%  eYAxisWestValueVisible
%%  eXAxisNorthValueVisible
%%  eYAxisEastValueVisible
%%  eXAxisSouthValuePos
%%  eYAxisWestValuePos
%%  eXAxisNorthValuePos
%%  eYAxisEastValuePos
%%  eXAxisSouthScaleType
%%  eYAxisWestScaleType
%%  eXAxisNorthScaleType
%%  eYAxisEastScaleType
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eaxis(xPos,yPos,length,type,scale,angle,color,offset)
  eglobpar;
  if nargin>8 | nargin<5
    eusage('eaxis(xPos,yPos,length,type,scale[,angle[,color[,offset]]])');
  end
  if nargin<8
    offset=0;
  end
  if nargin<7
```

```
    color=eAxesColor;
  end
  if nargin<6
    angle=0;
  end
  if type=='s'
   valueFormat=eXAxisSouthValueFormat;
   valueVisible=eXAxisSouthValueVisible;
   scaleType=eXAxisSouthScaleType;
  elseif type=='w'
   valueFormat=eYAxisWestValueFormat;
   valueVisible=eYAxisWestValueVisible;
   scaleType=eYAxisWestScaleType;
  elseif type=='n'
   valueFormat=eXAxisNorthValueFormat;
   valueVisible=eXAxisNorthValueVisible;
   scaleType=eXAxisNorthScaleType;
  else
   valueFormat=eYAxisEastValueFormat;
   valueVisible=eYAxisEastValueVisible;
   scaleType=eYAxisEastScaleType;
  end

  if scaleType==0
    valueTextPos=escalexy(eFile,type,xPos*eFac,yPos*eFac,...
           offset,angle,length*eFac,scale(1),scale(2),scale(3),...
           valueFormat,valueVisible,...
           eAxesValueFontSize*eFac,...
           eAxesLineWidth*eFac,...
           eAxesTicShortLength*eFac,...
           eAxesTicLongLength*eFac,...
           eAxesTicLongMaxN,...
           eAxesValueSpace*eFac,...
           color);
  elseif scaleType==1
    valueTextPos=escalecl(eFile,type,xPos*eFac,yPos*eFac,...
           offset,angle,length*eFac,scale(1),scale(2),scale(3),...
           valueFormat,valueVisible,...
           eAxesValueFontSize*eFac,...
           eAxesLineWidth*eFac,...
           eAxesTicLongLength*eFac,...
           eAxesTicLongMaxN,...
           eAxesValueSpace*eFac,...
           color);
  elseif scaleType==2
    valueTextPos=escalelog(eFile,type,xPos*eFac,yPos*eFac,...
           offset,angle,length*eFac,scale(1),scale(2),scale(3),...
           valueFormat,valueVisible,...
           eAxesValueFontSize*eFac,...
           eAxesLineWidth*eFac,...
           eAxesTicShortLength*eFac,...
           eAxesTicLongLength*eFac,...
           eAxesTicLongMaxN,...
           eAxesValueSpace*eFac,...
```

```
          color);
  end
  if type=='s'
    eXAxisSouthValuePos=valueTextPos/eFac;
  elseif type=='w'
    eYAxisWestValuePos=valueTextPos/eFac;
  elseif type=='n'
    eXAxisNorthValuePos=valueTextPos/eFac;
  else
    eYAxisEastValuePos=valueTextPos/eFac;
  end
```

1.2.4 ebar.m

```
%%NAME
%%  ebar - get coordinates for bar-plotting
%%
%%SYNOPSIS
%%  [xb yb]=ebar(y[,barWidth[,barNumber,clusterSize,[,x]])
%%
%%PARAMETER(S)
%%
%%  y            vector of y-data
%%  barWidth     x-size of bars
%%               if barWidth=0 then autosize
%%               default: barWidth=0
%%  number       number whithin the cluster
%%  clusterSize  total number of bars in one cluster
%%  x            vector of x-data
%%
%%  xb           vector of x-coodinates
%%  yb           vector of y-coodinates
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function [xb,yb]=ebar(y,barWidth,barNumber,clusterSize,x)
  eglobpar
  if nargin>5 | nargin<1 | nargin==3
    eusage('[xb yb]=ebar(y[,barWidth[,barNumber,clusterSize[,x]])');
  end
  n=length(y);
  m=1:n;
  if nargin<5
    x=m;
  end
  if nargin<4
    barNumber=1;
    clusterSize=1;
  end
  xDiff=x(2:n)-x(1:n-1);
  minDeltaX=min(xDiff);
  if nargin<2 | barWidth==0
    barWidth=0.7*minDeltaX/clusterSize;
  end
```

```
  barWidth=barWidth/2;

  i=1:4:4*n;
  yb=zeros(4*n,1);
  yb(i(m)+1)=y(m);
  yb(i(m)+2)=y(m);

  xb=zeros(4*n,1);
  xb(i(m))=x(m)-barWidth;
  xb(i(m)+1)=x(m)-barWidth;
  xb(i(m)+2)=x(m)+barWidth;
  xb(i(m)+3)=x(m)+barWidth;

  xb=xb+(barNumber*2-1-clusterSize)*barWidth;
  if eXAxisSouthScaleType
    xb=xb-minDeltaX/2;
  end
```

1.2.5 ebitmap.m

```
%%NAME
%%  ebitmap  -  transform  the current eps-file to bitmap-file
%%
%%SYNOPSIS
%%  mapFileName=ebitmap([bitmapType[,resolution[,mapFileName[,epsFileName]]]])
%%
%%PARAMETER(S)
%%  bitmapType     bitmap-type
%%                 default: 0
%%                 0 = PNG -format
%%                 1 = JPEG-format
%%                 2 = TIFF-format
%%                 3 = PPM-format
%%                 4 = PCX-format
%%  resolution     in dpi,resolution of bitmap-file
%%                 if scalar then resolution x and y direction are equal
%%                 default: 200  (dpi)
%%                 if [x y] vector then  resolution of x and y direction
%%                 if [x y q] vector then resolution and qualitiy of JPEG
%%                     q=100 for no lost of qualitiy
%%                     q=75  standard compression
%%  mapFileName    name of bitmap-file
%%                 default: 'eFileName.typeSuffix'
%%  epsFileName    name of eps-file
%%                 default: 'eFileName'
%%GLOBAL PARAMETER(S)
%%  eFileName
%%  eGhostscript
% written by stefan.mueller stefan.mueller@fgan.de (C) 2003

function mapFileName=ebitmap(bitmapType,resolution,mapFileName,epsFileName)
  if nargin>4
    eusage('mapFileName=ebitmap([bitmapType[,resolution[,mapFileName[,epsFileName]]]])');
```

```
end
eglobpar;
if exist('eFac')
  if isempty(eFac)
    einit;
  end
else
  einit;
end
if nargin<1
  bitmapType=0;
end
if nargout>0
  message=0;
else
  message=1;
end
if bitmapType<0
 error
end
if bitmapType==0
  outputFormat='png256';
  suffix='png';
elseif bitmapType==1
  outputFormat='jpeg';
  suffix='jpg';
elseif bitmapType==2
  outputFormat='tiff24nc';
  suffix='tif';
elseif bitmapType==3
  outputFormat='ppmraw';
  suffix='ppm';
elseif bitmapType==4
  outputFormat='pcx24b';
  suffix='pcx';
end
if nargin<2
  resolution=200;
end
if nargin<3
  pos=findstr(eFileName,'.eps');
  if length(pos)>0
    mapFileName=[eFileName(1:pos(1)) suffix];
  else
    mapFileName=[eFileName '.' suffix];
  end
end
if nargin<4
  epsFileName=eFileName;
end

headsize=500;
epsFile=fopen(epsFileName,'r');
data=fread(epsFile,headsize,'uchar');
```

```
fclose(epsFile);

head=setstr(data');
pos=findstr(head,'epsTk');
if length(pos)<1
  pos=findstr(head,'BoundingBox:')+12;
  win=sscanf(head(pos(1):pos(1)+40),'%f',4);
  win=win/eFac;
  xPos=win(1);
  yPos=win(2);
  width=win(3)-win(1);
  height=win(4)-win(2);

  eFileNameSave=eFileName;
  eopen(mapFileName,0,width,height,[0 0]);
  einseps(0,0,epsFileName);
  eclose(1,0);
  epsFileName=mapFileName;
  eFileName=eFileNameSave;
end

epsFile=fopen(epsFileName,'r');
data=fread(epsFile,inf,'uchar');
fclose(epsFile);
headsize=500;
head=setstr(data(1:headsize)');
pos=findstr(head,'BoundingBox:')+12;
win=sscanf(head(pos(1):headsize),'%f',4);
match=sscanf(head(pos(1):headsize),'%s',1);
pos=findstr(head,match);
if length(pos)>1
  pos2=findstr(head,'translate')+8;
  data(pos(2):pos2(1))=ones(1,(pos2(1)-pos(2)+1))*32;
end
mapFile=fopen(mapFileName,'w');
fwrite(mapFile,data,'uchar');
fclose(mapFile);

if size(resolution,2)>2
  quality=round(resolution(1,3));
else
  quality=75;
end
options=sprintf('-q -dNOPAUSE -dSAFER -dBATCH -dJPEGQ=%d',quality);
xResolution=resolution(1,1);
yResolution=resolution(1,1);
if size(resolution,2)>1
  yResolution=resolution(1,2);
else
  yResolution=resolution(1,1);
end
xPixel=round((win(3)-win(1))/72*xResolution);
yPixel=round((win(4)-win(2))/72*yResolution);
pixelSize=sprintf('-g%dx%d',fix(xPixel),fix(yPixel));
```

```
  resDpi=sprintf('-r%d',resolution);
  device=sprintf('-sDEVICE=%s',outputFormat);
  outFileName=sprintf('-sOutputFile=%s',mapFileName);
  ghostscript=sprintf('%s %s %s %s %s %s %s',...
  eGhostscript,options,pixelSize,resDpi,device,outFileName,mapFileName);
  if isempty(eGhostscript)
    disp('ebitmap: sorry, no ghostscript installed');
  else
    if exist('matlabpath')~=5
      system(ghostscript);
    else
      unix(ghostscript);
    end
    if message
      message=sprintf('%s is written',mapFileName);
      disp(message);
    end
  end
```

1.2.6 ebright.m

```
%%NAME
%%  ebright  - change brightness of colormap
%%
%%SYNOPSIS
%%  newColormap=ebright(colormap,brightness[,colorChannel])
%%
%%PARAMETER(S)
%%  colormap      color table
%%  brightness    +/- brigthness in per cent
%%  newColormap   changed color table
%%  colorChannel  vector of Channel; 1=red, 2=green, 3=blue
%%                e.g. [1 3] = Channel red and blue
%%                default=[1 2 3]
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
function newColormap= ebright (colormap,brightness,colorChannel)
  if nargin <2 | nargin >3
    eusage('newColormap = ebright(colormap,brightness[,colorChannel])');
  end
  if nargin<3
    colorChannel=[1 2 3];
  end

  newColormap=colormap(:,colorChannel);
  [rows cols]=size(newColormap);
  newColormap=reshape(newColormap,rows*cols,1);
  if brightness<0
    if brightness<-100
      brightness=-100;
    end
    a=1+brightness/100;
    newColormap=newColormap*a;
```

```
  else
    if brightness>100
      brightness=100;
    end
    a=brightness/100;
    b=1-a;
    newColormap=newColormap*b+a;
  end
  newColormap=reshape(newColormap,rows,cols);
  colormap(:,colorChannel)=newColormap;
  newColormap=colormap;
```

1.2.7 ecdcover.m

```
%%NAME
%%  ecdcover  -  write a cdcover
%%
%%SYNOPSIS
%%  ecdcover(title[,description[,author[,version[,date[,textColor
%%           [,frontImage[,background[,logo[,content]]]]]]]]])
%%
%%PARAMETER(S)
%%  title         string of title of cd
%%  description   string of description, default=''
%%  author        string of author, default=''
%%  version       string of version, default=''
%%  date          string of time or date, default=''
%%  textColor     color vector [r g b]  [0 0 0]=black [1 1 1]=white,
%%                default=[0 0 0]
%%  frontImage    filename of frontImage , default=''
%%  background    filename of background, default=''
%%  logo          filename  of logo, default=''
%%  content       filename(asci-file, tabs with '#')  of table of contents
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function ecdcover(title,description,author,version,date,textColor, ...
                  frontImage,background,logo,content)
  if nargin<1 | nargin>10
    eusage('ecdcover(title[,description[,author[,version[,date[,textColor[,frontImage[,background[,l
  end
  eglobpar;
  if nargin<10
    content='';
  end
  if nargin<9
    logo='';
  end
  if nargin<8
    background='';
  end
  if nargin<7
```

```
    frontImage='';
  end
  if nargin<6
    textColor=[0 0 0];
  end
  if nargin<5
    date='';
  end
  if nargin<4
    version='';
  end
  if nargin<3
    author='';
  end
  if nargin<2
    description='';
  end

  esavpar;
  fCoverH=119.5;
  fCoverW=119.5;
  bCoverH=117.5;
  bCoverW=150;
  bCoverPos=[(eWinWidth-bCoverW)/2 (eWinHeight-fCoverH-bCoverH)/2];
  fCoverPos=[bCoverPos(1) bCoverPos(2)+0.25+bCoverH];
  contentW=120;
  contentH=60;
  logoH=10;
  fImageH=60;
  if isempty(version)
    fImageH=fImageH-10
  end

%frontside
  %background
  if ~isempty(background)
    im=eimgread(background);
    eframe(fCoverPos(1),fCoverPos(2),fCoverW,fCoverH,0,im,-1);
  else
    eframe(fCoverPos(1),fCoverPos(2),fCoverW,fCoverH,1);
  end

  %frontImage
  if ~isempty(frontImage)
    im=eimgread(frontImage);
    [ih iw]=size(im);
    whFac=iw/ih;
    fImageW=fImageH*whFac;
    fImagePos=[fCoverPos(1)+(fCoverW-fImageW)/2 ...
               fCoverPos(2)+(fCoverH-fImageH)/3];
    eframe(fImagePos(1),fImagePos(2),fImageW,fImageH,0,im,-1);
    eframe(fImagePos(1),fImagePos(2),fImageW,fImageH,0.2);
  end
```

```
  %title
  etext(title,fCoverPos(1)+fCoverW/2,fCoverPos(2)+100,10,0,1,0,textColor);

  %description
  etext(description,fCoverPos(1)+fCoverW/2,fCoverPos(2)+90,6,0,1,0,textColor);

  %version
  etext(version,fCoverPos(1)+fCoverW/2,fCoverPos(2)+10,6,0,1,0,textColor);

%backside
  %background
  %background
  if ~isempty(background)
    im=eimgread(background);
    eframe(bCoverPos(1),bCoverPos(2),bCoverW,bCoverH,0,im,-1);
  else
    eframe(bCoverPos(1),bCoverPos(2),bCoverW,bCoverH,1);
  end

  %logo
  if ~isempty(logo)
    im=eimgread(logo);
    [ih iw]=size(im);
    whFac=iw/ih;
    logoW=logoH*whFac;
    logoPos=[bCoverPos(1)+(bCoverW-logoW)/2 ...
             bCoverPos(2)+10+5/2-logoH/2];
    eframe(logoPos(1),logoPos(2),logoW,logoH,0,im,-1);
  end

  %title
  etext(title,bCoverPos(1)+bCoverW/2,bCoverPos(2)+105,8,0,1,0,textColor);
  etext(title,bCoverPos(1)+4,bCoverPos(2)+bCoverH*5/6,4,0,1,90,textColor);
  etext(title,bCoverPos(1)+bCoverW-4,bCoverPos(2)+bCoverH*5/6,4,0,1,270,textColor);

  %description
  etext(description,bCoverPos(1)+bCoverW/2,bCoverPos(2)+97,5,0,1,0,textColor);
  etext(description,bCoverPos(1)+4,bCoverPos(2)+bCoverH*3/12,4,0,1,90,textColor);
  etext(description,bCoverPos(1)+bCoverW-4,bCoverPos(2)+bCoverH*3/12,4,0,1,270,textColor);

  %date
  etext(date,bCoverPos(1)+bCoverW*1/4,bCoverPos(2)+10,5,0,1,0,textColor);

  %version
  etext(version,bCoverPos(1)+4,bCoverPos(2)+bCoverH*7/12,4,0,1,90,textColor);
  etext(version,bCoverPos(1)+bCoverW-4,bCoverPos(2)+bCoverH*7/12,4,0,1,270,textColor);

  %author
  etext(author,bCoverPos(1)+bCoverW*3/4,bCoverPos(2)+10,5,0,1,0,textColor);

  % table of contents
  if ~isempty(content)
    [textString,tl]=etxtread(content);
    cr=setstr(13);
```

```
    lf=setstr(10);
    tab='#';
    pos=findstr(textString,cr);
    textString(pos)=' ';
    pos=findstr(textString,lf);
    nLines=length(pos);
    fontSpace=contentH/nLines;
    if fontSpace>5;
      fontSize=5;
    else
      fontSize=fontSpace;
    end
    lineStart=1;
    for i=1:nLines
      tabPos=findstr(textString(lineStart:pos(i)-1),tab);
      if length(tabPos)>0
        tabPos=[tabPos+lineStart-1 pos(i)];
      else
        tabPos=pos(i);
      end
      nRows=length(tabPos);
      tabSpace=contentW/nRows;
      textStart=lineStart;
      for k=1:nRows
        if tabPos(k)>textStart
          text=textString(textStart:(tabPos(k)-1));
          etext(text,bCoverPos(1)+(bCoverW-contentW)/2+(k-1)*tabSpace,...
                bCoverPos(2)+90-i*fontSpace,...
                fontSize,1,1,0,textColor);
        end
        textStart=tabPos(k)+1;

      end
      lineStart=pos(i)+1;
    end
  end

  erespar;
```

1.2.8 echrcode.m

```
eopen('chrcode.eps');
eglobpar;

%titel
etext('Character Codes',eWinWidth/2,230,10,0);

%text character
eTextFont=1;
eTextFontSize=4;
xValue=5;
yValue=210;
lineStep=-4.5;
```

```
etext('Character codes of font 1-12 call by  octal value',xValue,yValue);
yValue=yValue+lineStep;
eTextFontSize=3;
for i=0:254
  c=sprintf('  \\134%o=',i);
  s=sprintf('\\%o',i);
  if rem(i,15)==0
    yValue=yValue+lineStep;
    xValue=5;
  end
  etext(c,xValue,yValue,eTextFontSize,1);
  etext(s,0,0,eTextFontSize,1);
  xValue=xValue+11;
end
yValue=yValue+lineStep;
yValue=yValue+lineStep;
yValue=yValue+lineStep;
eTextFontSize=4;
xValue=5;
etext('Character Codes of font 13 call by  octal value',xValue,yValue);
yValue=yValue+lineStep;
eTextFontSize=3;
for i=0:254
  c=sprintf('  \\134%o=',i);
  s=sprintf('\\%o',i);
  if rem(i,15)==0
    yValue=yValue+lineStep;
    xValue=5;
  end
  etext(c,xValue,yValue,eTextFontSize,1);
  etext(s,0,0,eTextFontSize,1,13);
  xValue=xValue+11;
end

eclose;
eview;
```

1.2.9 eclose.m

```
%%NAME
%%  eclose - finish plot(s) and close EPS-file
%%
%%SYNOPSIS
%%  eclose ([nCopies[,message]])
%%
%%PARAMETER(S)
%%  nCopies     number of hardcopies, for printing 1 or more copies
%%              default: nCopies=1, print one copy of current page
%%              if nCopies=0 then 'showpage' will not append
%%  message     switch for 'file written' message
%%              if message=1 then write message (default)
%%              else no message
%%
```

```
%%GLOBAL PARAMETER(S)
%%  eWinFrameVisible
%%  eWinTimeStampVisible
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eclose(nCopies,message)
  eglobpar;
  if nargin>2
    eusage('eclose([nCopies[,message]])');
  end
  if nargin<2
    message=1;
  end
  if nargin<1
    nCopies=1;
  end
  if ePlotLineNo>0
    eplot
    ePlotLineNo=0;
  end
  if ePolarPlotLineNo>0
    epolar
    ePolarPlotLineNo=0;
  end
  if ePieSliceNo>0
    epie
    ePieSliceNo=0;
  end
  if eWinTimeStampVisible
    timeStamp=clock;
    min1=fix(timeStamp(5)/10);
    min2=rem(timeStamp(5),10);
    timeStampText=sprintf('epsTk 2.0 %d.%d.%d %d:%d%d:%d File:%s',...
                          timeStamp(3),timeStamp(2),timeStamp(1),...
                          timeStamp(4),min1,min2,timeStamp(6),...
                          eFileName);
    etext(timeStampText,-1,0,eWinTimeStampFontSize,1,...
         eWinTimeStampFont,90);
  end
  if eWinFrameVisible
    erect(eFile,0,0,eWinWidth*eFac,eWinHeight*eFac,...
          eWinFrameLineWidth,[0 0 0],0,0);
  end
  if eWinGridVisible
    esavpar;
    ePlotAreaPos=[0 0];
    ePlotAreaHeight=eWinHeight;
    ePlotAreaWidth=eWinWidth;
    eXGridVisible=1;
    eYGridVisible=1;
    eXGridColor=[0.6 0.6 0.6];
    eYGridColor=[0.6 0.6 0.6];
    eaxes([0 10 eWinWidth],[0 10 eWinHeight]);
    egrid;
```

```
    erespar;
  end
  if nCopies>0
    fprintf(eFile,'/#copies %d def\n',nCopies);
    fprintf(eFile,'showpage\n');
  end
  fclose(eFile);
  eFile=-1;
  if message==1
    if nCopies>1
      message=sprintf('%s (hardcopies:%d) is written',eFileName,nCopies);
    else
      message=sprintf('%s is written',eFileName);
    end
    disp(message);
  end
  if exist('close')
    close
  end
```

1.2.10 ecolors.m

```
%%NAME
%%  ecolors - get a colormap defined in einit.m
%%
%%SYNOPSIS
%%  colorMap=ecolors([mapNo [,nColors]])
%%
%%PARAMETER(S)
%%  colorMap  matrix of nColors x 3 Values between 0 and 1
%%  mapNo     map number in the definition matrix eColorMaps (default=0)
%%  nColors   number of colors (default=64)
%%
%%GLOBAL PARAMETER(S)
%%  eColorMaps
% written by Joerg Heckenbach and Stefan Mueller

function colorMap=ecolors(mapNo,nColors)
  if nargin > 2
    eusage('colorMap=ecolors([mapNo [,nColors]])');
  end
  eglobpar;
  if nargin < 2
    nColors=64;
  end
  if nargin < 1
    mapNo=0;
  end
  mapDef=eColorMaps(find(eColorMaps(:,1)==mapNo),2:4);
  if size(mapDef,1)==0
    gray=0:1/(nColors-1):1;
    colorMap=[gray' gray' gray'];
  else
```

```
    nColors=min(max(nColors, size(mapDef,1)),256);
    colorMap=zeros(nColors,3);
    N=(size(colorMap,1)-1)/(size(mapDef,1)-1);
    colorMap(1,:)=mapDef(1,:);
    for n=2:size(colorMap,1)
       nN=fix((n-1)/N)+1;      %Nummer des aktuellen N-Blocks
       nn=fix(rem(n-1,N));  %Position innerhalb eines N-Blocks
       if nn==0
          colorMap(n,:)=mapDef(nN,:);
       else
          colorMap(n,:)=(mapDef(nN+1,:)-mapDef(nN,:))/N*nn+mapDef(nN,:);
       end
    end
  end
```

1.2.11 econtour.m

```
%%NAME
%%  econtour - draw a contour plot of matrix
%%
%%SYNOPSIS
%%  econtour(matrix[,scale[,dash[,colorMap]]])
%%
%%PARAMETER(S)
%%  matrix     matrix for contour plot
%%  scale      vector of scaling [start step end]
%%  dash       if dash=0 then draw solid lines
%%             else value of dash is the distance of dashs
%%  colorMap   colors for different iso-lines
%%
%%GLOBAL PARAMETER(S)
%%  ePlotAreaWidth
%%  ePlotAreaHeight
%%  ePlotAreaPos
%%  eContourLineColor
%%  eContourLineDash
%%  eContourScale
%%  eContourLevelsMaxN
%%  eContourValueFormat
%%  eContourLineWidth
%%  eContourValueVisible
%%  eContourValueDistance
%%  eContourValueFont
%%  eContourValueFontSize
%%  eYAxisWestScale
%%  eXAxisSouthScale
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function econtour(matrix,scale,dash,colorMap)
  if nargin>4
    eusage('econtour(matrix[,scale[,dash[,colorMap]]])');
  end
```

```
eglobpar;
if nargin<4
  colorMap=eContourLineColor;
end
if nargin<3
  dash=eContourLineDash;
end
if nargin<2
  scale=eContourScale;
end
if nargin<1
  x=-2:.1:2;
  y=-2:.1:2;
  [X,Y]=meshgrid(x,y);
  matrix=X.*exp(-X.^2-Y.^2);
end

%scaling
[ rows cols ] = size (matrix);
maxval = max (max (matrix));
minval = min (min (matrix));
if scale(1)==scale(3)
  scale(1)=minval;
  scale(3)=maxval;
end
startEndDiff=scale(3)-scale(1);
if scale(2)==0
  %autoscale
  signOfDelta=sign(startEndDiff);
  delta=eticdis(signOfDelta*startEndDiff,eContourLevelsMaxN);
  offset=rem(scale(1),delta)*signOfDelta;
  if offset<0
    offset=delta+offset;
  end
  firstValue=scale(1)+delta*signOfDelta-offset;
else
  %fixscale
  signOfDelta=sign(scale(2));
  delta=signOfDelta*scale(2);
  firstValue=scale(1);
end

%value format
vForm=eContourValueFormat;
if vForm==0
  expo=-log10(delta);
  if rem(expo,1)>0
    expo=expo+1;
  end
  autoForm=fix(expo);
  if autoForm>0
    vForm=autoForm;
  end
end
```

```
if vForm<0
  valueForm='%g';
else
  valueForm=sprintf('%%1.%df',vForm);
end


%isolines loop
eclip(eFile,ePlotAreaPos(1)*eFac,ePlotAreaPos(2)*eFac,...
      ePlotAreaWidth*eFac,ePlotAreaHeight*eFac);
isoValues=firstValue:delta*signOfDelta:scale(3);
ncolor=size(colorMap,1);
colorFac=(ncolor-1)/(maxval-minval);
for i=1:length(isoValues)
  xy=eisoline(matrix,isoValues(i));
  if size(xy,1)>1
    ePlotAreaXFac=ePlotAreaWidth*eFac/cols;
    ePlotAreaYFac=ePlotAreaHeight*eFac/rows;
    xy(:,1)=xy(:,1)*ePlotAreaXFac;
    xy(:,2)=xy(:,2)*ePlotAreaYFac;
    color=colorMap(1+fix((isoValues(i)-minval)*colorFac),:);
    %draw isolines
    exyline(eFile,...
          ePlotAreaPos(1)*eFac,...
          ePlotAreaPos(2)*eFac,...
          xy(:,1),xy(:,2),color,...
          dash*eFac,eContourLineWidth*eFac);
    % write values
    if eContourValueVisible==1
      [maxv maxi]=max(xy(:,2));
      if rem(maxi,2)==1
        txy1=xy(maxi,:);
        txy2=xy(maxi+1,:);
      else
        txy1=xy(maxi-1,:);
        txy2=xy(maxi,:);
      end
      xdiff=txy2(1)-txy1(1);
      ydiff=txy2(2)-txy1(2);
      if xdiff>eps
        angle=atan(ydiff/xdiff)/pi*180;
      else
        angle=90;
      end
      if abs(isoValues(i))<1e-14
        isoValues(i)=0;
      end
      valueStr=sprintf(valueForm,isoValues(i));
      distance=1*eFac;
      textx=ePlotAreaPos(1)*eFac+txy1(1)+xdiff/2-...
            sin(angle/180*pi)*eContourValueDistance;
      texty=ePlotAreaPos(2)*eFac+txy1(2)+ydiff/2+...
            cos(angle/180*pi)*eContourValueDistance;
      etextxy(eFile,textx,texty,angle,0,valueStr,...
```

```
                eFonts(eContourValueFont,:),eContourValueFontSize*eFac,color);
      end
    end
  end
  eclip(eFile,0,0,0,0);

  if eYAxisWestScale(1)==eYAxisWestScale(3)
    eYAxisWestScale=[0 0 rows];
  end
  if eXAxisSouthScale(1)==eXAxisSouthScale(3)
    eXAxisSouthScale=[0 0 cols];
  end
  etitle;
  egrid;
  eaxes;
```

1.2.12 econtra.m

```
%%NAME
%%  econtra  - change contra of colormap
%%
%%SYNOPSIS
%%  newColormap=econtra(colormap,contrast[,colorChannel])
%%
%%PARAMETER(S)
%%  colormap      color table
%%  contrast      -200 to 200 per cent
%%  newColormap   changed color table
%%  colorChannel  vector of Channel; 1=red, 2=green, 3=blue
%%                e.g. [1 3] = Channel red and blue
%%                default=[1 2 3]
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
function newColormap= econtra (colormap,contrast,colorChannel)
  if nargin <2 | nargin >3
    eusage('newColormap = econtra(colormap,contrast[,colorChannel])');
  end
  if nargin<3
    colorChannel=[1 2 3];
  end

  newColormap=colormap(:,colorChannel);
  [rows cols]=size(newColormap);
  newColormap=reshape(newColormap,rows*cols,1);
  if contrast<0
    if contrast<-200
      contrast=-200;
    end
    a=-0.5*contrast/100;
    b=(1-2*a);
    newColormap=newColormap*b+a;
  else
    if contrast>200
```

```
      contrast=200;
    end
    a=0.5*contrast/100;
    b=(1-2*a);
    if b*b<eps
      b=eps;
    end
    newColormap=(newColormap-a)/b;
    newColormap(find(newColormap>1))=1;
    newColormap(find(newColormap<0))=0;
  end
  newColormap=reshape(newColormap,rows,cols);
  colormap(:,colorChannel)=newColormap;
  newColormap=colormap;
```

1.2.13 edsymbol.m

```
%%NAME
%%  edsymbol - define symbols for plotting
%%
%%SYNOPSIS
%%  edsymbol(name,symbolFileName
%%           [,scaleX[,scaleY[,moveX[,moveY[,rotation[,color]]]]]])
%%
%%PARAMETER(S)
%%  name             definition name for new symbol
%%  symbolFileName   filename of postscript symbol file (*.psd)
%%  scaleX           scale factor in X-direction
%%  scaleY           scale factor in Y-direction
%%  moveX            offset in X-direction
%%  moveY            offset in X-direction
%%  rotation         rotate symbol (deg)
%%  color            color vector [r g b], color of symbol
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function edsymbol(name,symbolFileName,scaleX,scaleY,moveX,moveY,...
                  rotation,color)
  if nargin<2 |nargin>8
    eusage('edsymbol(name,symbolFileName[,scaleX[,scaleY[,moveX[,moveY[,rotation[,color]]]]]])');
  end
  if nargin<8
    color=[-1 0 0];
  end
  if nargin<7
    rotation=0;
  end
  if nargin<6
    moveY=0;
  end
  if nargin<5
    moveX=0;
```

```
  end
  if nargin<4
    scaleY=1;
    if nargin==3
      scaleY=scaleX;
    end
  end
  if nargin<3
    scaleX=1;
  end
  eglobpar;
  fprintf(eFile,'/%s  gsave %1.2f %1.2f translate \n',name,moveX,moveY);
  fprintf(eFile,'%1.2f rotate\n',rotation);
  fprintf(eFile,'%1.2f %1.2f scale\n',scaleX,scaleY);
  if color(1)>=0
    fprintf(eFile,'%1.2f %1.2f %1.2f setrgbcolor\n',...
          color(1),color(2),color(3));
  end

  psdFileName=[ePath symbolFileName(find(symbolFileName~=' '))];
  epsFile=fopen(psdFileName,'r');

  if epsFile>1
    % get file length
    fseek(epsFile,0,1);
    epsFileLength=ftell(epsFile);
    fclose(epsFile);
    bufferSize=100000;
    epsFile=fopen(psdFileName,'r');
    nBuffer=fix(epsFileLength/bufferSize);
    tail=rem(epsFileLength,bufferSize);
    for i=1:nBuffer
      buffer=fread(epsFile,bufferSize,'char');
      fwrite(eFile,buffer,'char');
    end
    if tail>0
      buffer=fread(epsFile,tail,'char');
      fwrite(eFile,buffer,'char');
    end
    fclose(epsFile);
  else
    eval(symbolFileName);
  end
  fprintf(eFile,' grestore def\n');
```

1.2.14 eellipse.m

```
%%NAME
%%  eellipse  - draw ellipse
%%
%%SYNOPSIS
%%  eellipse(xPos,yPos,width,height[,lineWidth[,dash[,color
%%         [,rotation]]]])
```

```
%%
%%PARAMETER(S)
%%  xPos          x-Position of center of ellipse
%%  yPos          y-Position of center of ellipse
%%  width         width of ellipse
%%  height        height of ellipse
%%  lineWidth     linewidth of ellipse
%%                default: lineWidth=eLineWidth
%%  dash          if a scalar
%%                  =0  solid ellipse,
%%                  >0  dash length
%%                  <0  fill ellipse with color
%%                default: dash=eLineDash
%%                if a matrix and color=-1
%%                  dash is the image of ellipse
%%                  and filled with RGB values
%%                  value=R*2^16+G*2^8+B) and R,G,B are integer of 0:255
%%                if a matrix and color is a colormap
%%                  dash is the image of ellipse
%%                  and filled with indices of colormap
%%                if a string dash is filename of a JPEG-file
%%  color         if dash>=0 vector of ellipse color ([r g b])
%%                if dash<0  vector of background color
%%                if dash a matrix then colormap of image or -1
%%                default: dash=eLineColor
%%  rotation      rotation of ellipse (in deg)
%%
%%GLOBAL PARAMETER(S)
%%  eLineWidth
%%  eLineDash
%%  eLineColor
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eellipse (xPos,yPos,width,height,lineWidth,dash,color,rotation,cornerRadius)
  if nargin<4 | nargin>8
    eusage('eellipse(xPos,yPos,width,height[,lineWidth[,dash[,color[,rotation[,cornerRadius]]]]])');
  end
  eglobpar;
  if nargin<8
    rotation=0;
  end
  if nargin<7
    color=eLineColor;
  end
  if nargin<6
    dash=eLineDash;
  end
  if nargin<5
    lineWidth=eLineWidth;
  end
  eellipxy(eFile,xPos*eFac,yPos*eFac,width*eFac,height*eFac,...
            lineWidth*eFac,color,dash,rotation);
```

1.2.15 eerrbar.m

```
%%NAME
%%  eerrbar - get coordinates-matrix for errorbar-plotting
%%
%%SYNOPSIS
%%  [xeb yeb]=eerrbar(x,y,error[,barWidth])
%%
%%PARAMETER(S)
%%
%%  x             vector of x-data
%%  y             vector of y-data
%%  barWidth      x-size of bars
%%                default: autosize
%%
%%  xeb           matrix of x errorbar-coodinates
%%  yeb           matrix of y errorbar-coodinates
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function [xeb,yeb]=eerrbar(x,y,error,barWidth)
  eglobpar
  if nargin>4 | nargin<3
    eusage('[xeb yeb]=eerrbar(x,y,error,barWidth)');
  end
  [r c]=size(x);
  if r>1
    x=x';
    c=r;
  end
  if nargin<4
    barWidth=min(x(2:c)-x(1:c-1))/3;
  end

  xeb=[x x-barWidth/2 x-barWidth/2;x x+barWidth/2 x+barWidth/2];

  [r c]=size(y);
  if r>1
    y=y';
  end

  [r c]=size(error);
  if r>1
    error=error';
  end
  ype=y+error;
  yme=y-error;
  yeb=[ype ype yme;yme ype yme];
```

1.2.16 efillmat.m

```
%%NAME
%%  efillmat  - fill matrix with interpolated values by given xyz-samples
%%
```

```
%%SYNOPSIS
%%  matrix=efillmat(xData,yData,zData,dX,dY)
%%
%%PARAMETER(S)
%%  xData       vector of x-coordinates
%%  yData       vector of y-coordinates
%%  zData       vector of z-values
%%  dx          pixel distance of x-direction
%%  dy          pixel distance of y-direction
%%  matrix      interpolated matrix
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
function matrix= efillmat (xData,yData,zData,dx,dy)
  if (nargin ~= 5)
    eusage('matrix = efillmat(xData,yData,zData,dx,dy)');
  end
  minX=min(xData);
  maxX=max(xData);
  minY=min(yData);
  maxY=max(yData);
  cols=round((maxX-minX)/dx);
  rows=round((maxY-minY)/dx);
  dx=(maxX-minX)/cols;
  dy=(maxY-minY)/rows;
  cols=cols+1;
  rows=rows+1;
  x=round((xData-minX)/dx);
  y=round((yData-minY)/dy);
  matrix=ones(rows*cols,1)*NaN;
  matrix(x*rows+cols-y)=zData;
  matrix=reshape(matrix,rows,cols);
  smat=~isnan(matrix);
  xr=sum(smat)/cols;
  yr=sum(smat')/rows;
  [sv sp]=sort([xr yr]);
  for i=fliplr(sp)
    if i>cols
      j=i-cols;
      svec=~isnan(matrix(j,:));
      x=find(svec);
      n=length(x);
      if n<cols
        xi=find(~svec);
        x0=1;z0=matrix(j,x(1));
        xn=cols;zn=matrix(j,x(n));
        matrix(j,xi)=elineip([x0 x xn],[z0 matrix(j,x) zn],xi);
      end
    else
      svec=~isnan(matrix(:,i));
      x=find(svec);
      n=length(x);
      if n<rows
        xi=find(~svec);
        x0=0;z0=matrix(x(1),i);
```

```
        xn=rows+1;zn=matrix(x(n),i);
        matrix(xi,i)=elineip([x0 x' xn],[z0 matrix(x,i)' zn],xi')';
      end
    end
  end
```

1.2.17 eframe.m

```
%%NAME
%%  eframe  - draw frame
%%
%%SYNOPSIS
%%  eframe(xPos,yPos,width,height[,lineWidth[,dash[,color
%%         [,rotation[,cornerRadius]]]]])
%%
%%PARAMETER(S)
%%  xPos           x-Position of sw-corner of frame
%%  yPos           y-Position of sw-corner of frame
%%  width          width of frame
%%  height         height of frame
%%  lineWidth      linewidth of frame
%%                 default: lineWidth=eLineWidth
%%  dash           if a scalar
%%                   =0  solid frame,
%%                   >0  dash length
%%                   <0  fill frame with color
%%                 default: dash=eLineDash
%%                 if a matrix and color=-1
%%                   dash is the image of frame
%%                   and filled with RGB values
%%                   value=R*2^16+G*2^8+B) and R,G,B are integer of 0:255
%%                 if a matrix and color is a colormap
%%                   dash is the image of frame
%%                   and filled with indices of colormap
%%                 if a string then dash is filename of a JPEG-file
%%  color          if dash>=0 vector of frame color ([r g b])
%%                 if dash<0  vector of background color
%%                 if dash a matrix then colormap of image or -1
%%                 default: dash=eLineColor
%%  rotation       rotation of frame (in deg)
%%  cornerRadius   radius of rounded corner
%%                 default: 0=no rounded corner
%%
%%GLOBAL PARAMETER(S)
%%  eLineWidth
%%  eLineDash
%%  eLineColor
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eframe (xPos,yPos,width,height,lineWidth,dash,color,rotation,cornerRadius)
  if nargin<4 | nargin>9
    eusage('eframe(xPos,yPos,width,height[,lineWidth[,dash[,color[,rotation[,cornerRadius]]]]])');
  end
```

```
  eglobpar;
  if nargin<9
    cornerRadius=0;
  end
  if nargin<8
    rotation=0;
  end
  if nargin<7
    color=eLineColor;
  end
  if nargin<6
    dash=eLineDash;
  end
  if nargin<5
    lineWidth=eLineWidth;
  end
  if cornerRadius>0
    erectrc(eFile,xPos*eFac,yPos*eFac,width*eFac,height*eFac,...
            lineWidth*eFac,color,dash,rotation,cornerRadius*eFac);
  else
    erect(eFile,xPos*eFac,yPos*eFac,width*eFac,height*eFac,...
            lineWidth*eFac,color,dash,rotation);
  end
```

1.2.18 egradient.m

```
%%NAME
%%  egradient - get numerical partial derivatives of matrix
%%
%%SYNOPSIS
%%  [px py]=egradient(z[,dx[,dy]])
%%
%%PARAMETER(S)
%%
%%  px          px=dz/dx
%%  py          py=dz/dy
%%  z           z-matrix
%%  dx          delta x
%%  dy          delta y
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function [px,py] = egradient(z,dx,dy)
  if nargin > 3
    eusage('[px py]=egradient(z[,dx[,dy]])');
  end
  if nargin < 3
    dy = dx;
  end
  if nargin < 2
    dx = 1;
    dy = 1;
  end
```

```
[m,n] = size(z);
if length(dx) == 1
  dx = dx .* (0:n-1);
end
if length(dy) == 1
  dy = dy .* (0:m-1);
end

if size(dx,1)>1
  dx = dx';
end
x = zeros(m, n);
j = 1:m;
if n > 1
  % left
  d = dx(2) - dx(1);
  x(j, 1) = (z(j, 2) - z(j, 1))./d;
  % right
  d = dx(n) - dx(n-1);
  x(j, n) = (z(j, n) - z(j, n-1))./d;
end
if n > 2
  % middle
  k = 1:n-2;
  d = ones(m, 1) * (dx(k+2) - dx(k));
  x(j, k+1) = (z(j, k+2) - z(j, k)) ./d;
end

if size(dy,2)>1
  dy = dy';
end;
y = zeros(m, n);
j = 1:n;
if n > 1
  % left
  d = dy(2) - dy(1);
  y(1,j) = (z(2,j) - z(1,j))./d;
  % right
  d = dy(m) - dy(m-1);
  y(m,j) = (z(m,j) - z(m-1,j))./d;
end
if n > 2
  % middle
  k = 1:m-2;
  d = (dy(k+2) - dy(k)) * ones(1,n);
  y(k+1,j) = (z(k+2,j) - z(k,j))./d;
end

px = (x + sqrt(-1) .* y);
if nargout == 2
  py = imag(px);
  px = real(px);
end
```

1.2.19 egrid.m

```
%%NAME
%%  egrid - draw grid
%%
%%SYNOPSIS
%%  egrid([xAxisSouthScale,yAxisWestScale[,xAxisNorthScale,yAxisEastScale]])
%%
%%PARAMETER(S)
%%  xAxisSouthScale   scale vector of south axis [start step end]
%%  yAxisWestScale    scale vector of west axis  [start step end]
%%  xAxisNorthScale   scale vector of north axis [start step end]
%%  yAxisEastScale    scale vector of east axis  [start step end]
%%
%%    special cases of scale vectors are:
%%      if start=0 and end=0 then autorange=on
%%      if step=0 then autoscale=on
%%    (default scale vector=[0 0 0])
%%
%%GLOBAL PARAMETER(S)
%%  ePlotAreaPos
%%  ePlotAreaWidth
%%  ePlotAreaHeight
%%  eAxesTicLongMaxN
%%  eXAxisSouthScale
%%  eXAxisSouthScaleType
%%  eYAxisWestScale
%%  eYAxisWestScaleType
%%  eXAxisNorthScale
%%  eYAxisEastScale
%%  eXGridVisible
%%  eYGridVisible
%%  eXGridLineWidth
%%  eXGridColor
%%  eXGridDash
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function egrid(xAxisSouthScale,yAxisWestScale,xAxisNorthScale,yAxisEastScale)
  if rem(nargin,2) | nargin>4
    eusage('egrid([xAxisSouthScale,yAxisWestScale[,xAxisNorthScale,yAxisEastScale]])');
  end
  eglobpar;
  if nargin>0
    eXAxisSouthScale=xAxisSouthScale;
    ePlotAreaXValueStart=eXAxisSouthScale(1);
    ePlotAreaXValueEnd=eXAxisSouthScale(3);
    eYAxisWestScale=yAxisWestScale;
    ePlotAreaYValueStart=eYAxisWestScale(1);
    ePlotAreaYValueEnd=eYAxisWestScale(3);
    ePlotAreaXFac=ePlotAreaWidth*eFac/...
      (ePlotAreaXValueEnd-ePlotAreaXValueStart);
```

```
    ePlotAreaYFac=ePlotAreaHeight*eFac/...
      (ePlotAreaYValueEnd-ePlotAreaYValueStart);
  end
  if nargin>2
    eXAxisNorthScale=xAxisNorthScale;
    eYAxisEastScale=yAxisEastScale;
  end
  if nargin==0
    if eXAxisSouthScale(1)~=eXAxisSouthScale(3)
      ePlotAreaXValueStart=eXAxisSouthScale(1);
      ePlotAreaXValueEnd=eXAxisSouthScale(3);
    end
    if eYAxisWestScale(1)~=eYAxisWestScale(3)
      ePlotAreaYValueStart=eYAxisWestScale(1);
      ePlotAreaYValueEnd=eYAxisWestScale(3);
    end

    % scale and draw axes
    if eXGridVisible
      if eXAxisSouthScaleType==0
        egridxy(eFile,'s',...
              ePlotAreaPos(1)*eFac,...
              ePlotAreaPos(2)*eFac,...
              ePlotAreaWidth*eFac,...
              ePlotAreaXValueStart,...
              eXAxisSouthScale(2),...
              ePlotAreaXValueEnd,...
              eAxesTicLongMaxN,...
              ePlotAreaHeight*eFac,...
              eXGridLineWidth*eFac,...
              eXGridColor,...
              eXGridDash*eFac);
      elseif eXAxisSouthScaleType==1
        egridcl(eFile,'s',...
              ePlotAreaPos(1)*eFac,...
              ePlotAreaPos(2)*eFac,...
              ePlotAreaWidth*eFac,...
              ePlotAreaXValueStart,...
              eXAxisSouthScale(2),...
              ePlotAreaXValueEnd,...
              eAxesTicLongMaxN,...
              ePlotAreaHeight*eFac,...
              eXGridLineWidth*eFac,...
              eXGridColor,...
              eXGridDash*eFac);
      elseif eXAxisSouthScaleType==2
        egridlog(eFile,'s',...
              ePlotAreaPos(1)*eFac,...
              ePlotAreaPos(2)*eFac,...
              ePlotAreaWidth*eFac,...
              ePlotAreaXValueStart,...
              eXAxisSouthScale(2),...
              ePlotAreaXValueEnd,...
              eAxesTicLongMaxN,...
```

```
            ePlotAreaHeight*eFac,...
            eXGridLineWidth*eFac,...
            eXGridColor,...
            eXGridDash*eFac);
    end
  end
  if eYGridVisible
    if eYAxisWestScaleType==0
      egridxy(eFile,'w',...
            ePlotAreaPos(1)*eFac,...
            ePlotAreaPos(2)*eFac,...
            ePlotAreaHeight*eFac,...
            ePlotAreaYValueStart,...
            eYAxisWestScale(2),...
            ePlotAreaYValueEnd,...
            eAxesTicLongMaxN,...
            ePlotAreaWidth*eFac,...
            eYGridLineWidth*eFac,...
            eYGridColor,...
            eYGridDash*eFac);
    elseif eYAxisWestScaleType==1
      egridcl(eFile,'w',...
            ePlotAreaPos(1)*eFac,...
            ePlotAreaPos(2)*eFac,...
            ePlotAreaHeight*eFac,...
            ePlotAreaYValueStart,...
            eYAxisWestScale(2),...
            ePlotAreaYValueEnd,...
            eAxesTicLongMaxN,...
            ePlotAreaWidth*eFac,...
            eYGridLineWidth*eFac,...
            eYGridColor,...
            eYGridDash*eFac);
    elseif eYAxisWestScaleType==2
      egridlog(eFile,'w',...
            ePlotAreaPos(1)*eFac,...
            ePlotAreaPos(2)*eFac,...
            ePlotAreaHeight*eFac,...
            ePlotAreaYValueStart,...
            eYAxisWestScale(2),...
            ePlotAreaYValueEnd,...
            eAxesTicLongMaxN,...
            ePlotAreaWidth*eFac,...
            eYGridLineWidth*eFac,...
            eYGridColor,...
            eYGridDash*eFac);
    end
  end
end
```

1.2.20 eidx2rgb.m

```
%%NAME
```

```
%%  eidx2rgb  - covert  index-matrix to RGB-matrix
%%
%%SYNOPSIS
%%  matrix=eidx2rgb(image,colormap)
%%
%%PARAMETER(S)
%%  image       index-matrix
%%  colormap    color table
%%  matrix      RBG-matrix
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
function matrix= eidx2rgb (image,colormap)
  if (nargin ~= 2)
    eusage('matrix = eidx2rgb(image,colormap)');
  end

  [rows cols]= size(image)
  colormap=fix(colormap*255);
  colormap=colormap*[65536;256;1];
  image=reshape(image,rows*cols,1);
  image=colormap(image);
  matrix=reshape(image,rows,cols);
```

1.2.21 eimage.m

```
%%NAME
%%  eimage  - draw image of a matrix
%%
%%SYNOPSIS
%%  eimage(matrix[,colorMap])
%%
%%PARAMETER(S)
%%  matrix     rows x cols matrix for image
%%               if colorMap=-1 then
%%                  image is filled with RGB values
%%                  value=R*2^16+G*2^8+B) and R,G,B are integer of 0:255
%%                else
%%                  matrix is filled with indices of colormap
%%             or a string of filename of a JPEG-file
%%  colorMap   define own colormap
%%             default:colorMap=ecolors(eImageDefaultColorMap)
%%
%%GLOBAL PARAMETER(S)
%%  eImageDefaultColorMap
%%  ePlotAreaPos
%%  ePlotAreaWidth
%%  ePlotAreaHeight
%%  eImageFrameVisible
%%  eAxesLineWidth
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eimage(matrix,colorMap)
  if  nargin >2
```

```
    eusage('eimage(matrix[,colorMap])');
  end
  eglobpar;
  if nargin<2
    colorMap=ecolors(eImageDefaultColorMap);
  end
  if nargin<1
    matrix=eppmread;
    colorMap=-1;
  end
  % write title
  etitle;
  eimagexy(eFile,matrix,colorMap,...
           ePlotAreaPos(1)*eFac,ePlotAreaPos(2)*eFac,...
           ePlotAreaWidth*eFac,ePlotAreaHeight*eFac);
  if eImageFrameVisible
    erect(eFile,ePlotAreaPos(1)*eFac,ePlotAreaPos(2)*eFac,...
          ePlotAreaWidth*eFac,ePlotAreaHeight*eFac,...
          eAxesLineWidth*eFac,[0 0 0],0,0);
  end
```

1.2.22 eimagesc.m

```
%%NAME
%%  eimagesc  - draw scaled image of a matrix
%%
%%SYNOPSIS
%%  x=eimagesc(matrix[,colorMap[,legendOrientation[,legendScale]]])
%%
%%PARAMETER(S)
%%  x                   transformed output matrix with values of color numbers
%%  matrix              matrix for image
%%  colorMap            define own colormap
%%  legendOrientation   side of the image where the legend appears
%%                      character 's'(south),'n'(north),'w'(west) or 'e'(east)
%%                      (default orientation is south)
%%  legendScale         scale vector of legend [start step end]
%%                      special cases of scale vector are:
%%                      if start=0 and end=0 then autorange=on
%%                      if step=0 then autoscale=on
%%                      (default scale vector=[0 0 0])
%%
%%GLOBAL PARAMETER(S)
%%  ePlotAreaPos
%%  ePlotAreaWidth
%%  ePlotAreaHeight
%%  eImageDefaultColorMap
%%  eImageLegendScale
%%  eYAxisWestScale
%%  eXAxisSouthScale
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function x=eimagesc(matrix,colorMap,legendOrientation,legendScale)
```

```
if (nargin>4)
  eusage('x=eimagesc(matrix[,colorMap[,legendOrientation[,legendScale]]])');
end
eglobpar;
if nargin<4
  legendScale=eImageLegendScale;
end
if nargin<3
  legendOrientation='s';
end
if nargin<2
  colorMap=ecolors(eImageDefaultColorMap);
end
if nargin<1
  [matrix colorMap]=eppmread([ePath 'default.ppm']);
  ePlotTitleText='Photo';
end

% scale image
if legendScale(1)==legendScale(3)
  minval=min (min (matrix));
  maxval=max (max (matrix));
else
  minval=legendScale(1);
  maxval=legendScale(3);
end
if eImageLegendScaleType==2
  minval=log10(minval);
  maxval=log10(maxval);
  matrix=log10(matrix);
end
legendScale(1)=minval;
legendScale(3)=maxval;
if (maxval - minval)<=1000*eps
  maxval=minval+1000*eps;
end
x = fix ((matrix - minval) / (maxval - minval) *...
      (length (colorMap) - 1)) + 1;
eimage(x,colorMap)

% image axes
if eYAxisWestScale(1)==eYAxisWestScale(3)
  eYAxisWestScale=[size(matrix,1) 0 0];
end
if eXAxisSouthScale(1)==eXAxisSouthScale(3)
  eXAxisSouthScale=[0 0 size(matrix,2)];
end
egrid;
eaxes;

% color legend
eimgleg(ePlotAreaPos(1),ePlotAreaPos(2),ePlotAreaWidth,ePlotAreaHeight,...
        colorMap,legendScale,legendOrientation);
```

1.2.23 eimgread.m

```
%%NAME
%%  eimgread  - read image-file
%%
%%SYNOPSIS
%%  [image,colormap]=eimgread(imageFileName)
%%
%%PARAMETER(S)
%%  imageFileName  name of JPEG- or PPM-file
%%  image          image matrix
%%                 if colormap used then
%%                   image is filled with indices of colormap
%%                 else
%%                   image is filled with RGB values
%%                   value=R*2^16+G*2^8+B) and R,G,B are integer of 0:255
%%                   that's a very fast way
%%  colormap       color table
%%
% written by stefan.mueller stefan.mueller@fgan.de (C) 2003
function [image,colormap]=eimgread(imageFileName)
  if nargin>1
    eusage('[image,colormap]=eimgread(imageFileName)');
  end
  eglobpar;
  if exist('ePath')
    if isempty(ePath)
      einit;
    end
  else
      einit;
  end
  if nargin<1
    imageFileName=[ePath 'default.ppm'];
  end

  jpgpos=findstr(imageFileName,'.jpg');
  ppmpos=findstr(imageFileName,'.ppm');
  if length(jpgpos)
    esavpar;
    tempFileName='imgread.ppm';
    dpi=ejpg2eps(imageFileName,'imgread.eps');
    tempFileName=ebitmap(3,dpi,tempFileName,'imgread.eps');
    erespar;
  elseif length(ppmpos)
    tempFileName=imageFileName;
  end
  if nargout==2
    [image colormap]=eppmread(tempFileName);
  else
    image=eppmread(tempFileName);
  end
```

1.2.24 eimgrot.m

```
%%NAME
%%  eimgrot  - rotate image
%%
%%SYNOPSIS
%%  matrix=eimgrot(image,rotation)
%%
%%PARAMETER(S)
%%  image       index-matrix
%%  rotation    rotation in deg, 90 180 or 270
%%  matrix      RBG-matrix
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
function matrix= eimgrot (image,rotation)
  if (nargin ~= 2)
    eusage('matrix = eimgrot(image,rotation)');
  end
  rotation=rem(rotation,360);
  if rotation>=270
    matrix=flipud(image)';
  elseif rotation>=180
    matrix=flipud(fliplr(image));
  elseif rotation>=90
    matrix=flipud(image');
  else
    matrix=image;
  end
```

1.2.25 eimgview.m

```
%%NAME
%%  eimgview  - create and view eps-file of an image
%%
%%SYNOPSIS
%%  eimgview(matrix[,colorMap[,epsFileName]])
%%
%%PARAMETER(S)
%%  matrix      matrix for image
%%              if colorMap=-1 then
%%                matrix is filled with RGB values
%%                value=R*2^16+G*2^8+B) and R,G,B are integer of 0:255
%%              else
%%                matrix is filled with indices of colorMap
%%              or a string of filename of a JPEG-file
%%  colorMap    own colormap
%%              default:colorMap=ecolors(eImageDefaultColorMap)
%%  epsFileName default=eFileName
%%
% written by stefan.mueller stefan.mueller@fgan.de (C) 2003
function eimgview(matrix,colorMap,epsFileName)
  if nargin>3
    eusage('eimgview(matrix[,colorMap[,epsFileName]])');
```

```
  end
  eglobpar;
  if exist('eFac')
    if isempty(eFac)
      einit;
    end
  else
    einit;
  end
  if nargin<3
    epsFileName=eFileName;
  end
  if nargin<2
    colorMap=ecolors(eImageDefaultColorMap);
  end
  if nargin<1
    matrix=[ePath 'default.jpg'];
  end
  eglobpar;
  if exist('ePath')
    if isempty(ePath)
      einit;
    end
  else
      einit;
  end
  if isstr(matrix)
    ejpg2eps(matrix,epsFileName);
  else
    if colorMap(1,1)<0
      [matrix colorMap]=ergb2idx(matrix);
    end
    [imgH imgW]=size(matrix);
    imgFac=imgH/imgW;
    winFac=eWinHeight/eWinWidth;
    if winFac<imgFac
      eWinWidth=eWinHeight/imgFac;
    else
      eWinHeight=eWinWidth*imgFac;
    end
    offsetX=eWinWidth*eFac/imgW/2;
    offsetY=eWinHeight*eFac/imgH/2;
    eopen(epsFileName,0,eWinWidth,eWinHeight)
    fprintf(eFile,'%1.2f %1.2f translate\n',offsetX,offsetY);
    eframe(0,0,eWinWidth,eWinHeight,0,matrix,colorMap);
    eclose(1,0);
  end
  eview;
```

1.2.26 eimgwrit.m

```
%%NAME
%%  eimgwrit  - write image-file
```

```
%%
%%SYNOPSIS
%%  eimgwrit(imageFileName,image,colormap,quality)
%%
%%PARAMETER(S)
%%  imageFileName  name of image-file
%%                 possible are: png,jpg,tif,ppm,pcx
%%  image          matrix for image
%%                 if colormap=-1 then
%%                   image is filled with RGB values
%%                   value=R*2^16+G*2^8+B) and R,G,B are integer of 0:255
%%                 else
%%                   matrix is filled with indices of colormap
%%  colormap       color table
%%  quality        quality of JPEG-files, default=75  (%)
%%
% written by stefan.mueller stefan.mueller@fgan.de (C) 2003
function eimgwrit(imageFileName,image,colormap,quality)
  if nargin>4
    eusage('eimgwrit(imageFileName,image,colormap[,quality])');
  end
  eglobpar;
  if nargin<4
    quality=75;
  end
  if exist('ePath')
    if isempty(ePath)
      einit;
    end
  else
      einit;
  end
  pos0=findstr(imageFileName,'.png');
  pos1=findstr(imageFileName,'.jpg');
  pos2=findstr(imageFileName,'.tif');
  pos3=findstr(imageFileName,'.ppm');
  pos4=findstr(imageFileName,'.pcx');
  epsFileName=imageFileName;
  if length(pos0);
    type=0;
    pos=pos0;
  elseif length(pos1)
    type=1;
    pos=pos1;
  elseif length(pos2)
    type=2;
    pos=pos2;
  elseif length(pos3)
    type=3;
    pos=pos3;
  elseif length(pos4)
    type=4;
    pos=pos4;
  end
```

```
  epsFileName(pos(1):pos(1)+2);
  eglobpar;
  if exist('eFac')
    if isempty(eFac)
      einit;
    end
  else
    einit;
  end
  if type==3
    eppmwrit(imageFileName,image,colormap);
  else
    [imgH imgW]=size(image);
    imgFac=imgH/imgW;
    winFac=eWinHeight/eWinWidth;
    if winFac<imgFac
      eWinWidth=eWinHeight/imgFac;
      dpi=imgH*72/eWinHeight/eFac;
    else
      eWinHeight=eWinWidth*imgFac;
      dpi=imgW*72/eWinWidth/eFac;
    end
    offsetX=1.2*eWinWidth/imgW;
    offsetY=1.2*eWinHeight/imgH;
    eopen(epsFileName,0,eWinWidth,eWinHeight)
    eframe(0,0,eWinWidth+offsetX,eWinHeight+offsetY,0,image,colormap);
    eclose(1,0);
    ebitmap(type,[dpi dpi quality],imageFileName);
  end
```

1.2.27 einseps.m

```
%%NAME
%%  einseps  -  insert eps-file
%%
%%SYNOPSIS
%%  einseps(xPos,yPos,epsFileName,[,scaleX[,scaleY[,rotation]]])
%%
%%PARAMETER(S)
%%  xPos          x position
%%  yPos          y position
%%  epsFileName   name of eps-file
%%  scaleX        scale factor in x-direction
%%  scaleY        scale factor in y-direction
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function einseps(xPos,yPos,epsFileName,scaleX,scaleY,rotation)
  if nargin<3 | nargin>6
    eusage('einseps(xPos,yPos,epsFileName,[,scaleX[,scaleY[,rotation]]])');
  end
  if nargin<6
    rotation=0;
  end
```

```
  if nargin<5
    scaleY=1;
  end
  if nargin<4
    scaleX=1;
  end
  eglobpar;

  % read eps file
  epsFile=fopen(epsFileName,'r');
  if epsFile>0
    [data dl]=fread(epsFile,inf,'uchar');
    fclose(epsFile);
    headsize=500;
    head=setstr(data(1:headsize)');

    % read box
    pos=findstr(head,'BoundingBox:')+12;
    win=sscanf(head(pos(1):pos(1)+40),'%f',4);

    % delete showpage
    pos=dl-headsize;
    tail=setstr(data(pos:dl)');
    pos2=findstr(tail,'showpage');
    data(pos+pos2(1)-1:pos+pos2(1)+6)=32;

    % write head
    fprintf(eFile,'gsave %1.2f %1.2f translate\n',xPos*eFac,yPos*eFac);
    fprintf(eFile,'%1.2f rotate\n',rotation);
    fprintf(eFile,'%1.2f %1.2f translate\n',-win(1)*scaleX,-win(2)*scaleY);
    fprintf(eFile,'%1.2f %1.2f scale\n',scaleX,scaleY);

    % insert eps file
    fwrite(eFile,data,'uchar');
    fprintf(eFile,'grestore\n');
  end
```

1.2.28 eisoline.m

```
%%NAME
%%  eisoline  - get isolines of a matrix
%%
%%SYNOPSIS
%%  lines=eisoline(matrix,isoValue)
%%
%%PARAMETER(S)
%%  lines      empty matrix or 2n x 2 matrix,
%%             n=number of lines x [x1 y1;x2 y2]
%%  matrix     matrix of values
%%  isoValue   value of isoline
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
```

```
function lines=eisoline(matrix,isoValue)
  if nargin~=2
    eusage('eisoline(matrix,isoValue)');
  end
  if exist('matlabpath')~=5
    save_do_fortran_indexing = do_fortran_indexing;
    save_empty_list_elements_ok = empty_list_elements_ok;
    do_fortran_indexing = 1;
    empty_list_elements_ok = 1;
  end
  [rows colums]=size(matrix);
  diff=matrix(:,2:colums)-matrix(:,1:(colums-1));
  findzero=find(diff==0);
  diff(findzero)=diff(findzero)+eps;
  wex=(isoValue-matrix(:,1:(colums-1)))./diff;
  diff=matrix(2:rows,:)-matrix(1:(rows-1),:);
  findzero=find(diff==0);
  diff(findzero)=diff(findzero)+eps;
  nsx=(isoValue-matrix(1:(rows-1),:))./diff;
  wexT=wex';
  crossns=find(wexT>=0 & wexT<=1);
  crosswe=find(nsx>=0 & nsx<=1);
  limit=(rows-1)*(colums-1);
  kn=crossns(find(crossns<=limit));
  kn=rem(kn-1,colums-1)*(rows-1)+fix((kn-1)/(colums-1))+1;
  kw=crosswe(find(crosswe<=limit));
  limit=(colums-1);
  ks=crossns(find(crossns>limit))-limit;
  ks=rem(ks-1,colums-1)*(rows-1)+fix((ks-1)/(colums-1))+1;
  limit=(rows-1);
  ke=crosswe(find(crosswe>limit))-limit;
  lines=[kn   ones(length(kn),1);kw 2*ones(length(kw),1);
             ks 3*ones(length(ks),1);ke 4*ones(length(ke),1)];
  ncrosspoints=size(lines,1);
  if ncrosspoints>0
    [slist index]=sort(lines(:,1));
    lines=lines(index,:);
    si=1:ncrosspoints;
    sv=[lines(:,1);lines(ncrosspoints,1)+1];
    c1=find(sv(si)==sv(si+1));
    c2=c1+1;
    c3=[c1';c2'];
    c3=reshape(c3,1,2*length(c2));
    lines=lines(c3,:);
    row=rem(lines(:,1)-1,rows-1)+1;
    col=fix((lines(:,1)-1)/(rows-1))+1;
    north=find(lines(:,2)==1);
    west=find(lines(:,2)==2);
    south=find(lines(:,2)==3);
    east=find(lines(:,2)==4);
    if length(north)>0
      lines(north,1)=rem(lines(north,1)-1,rows-1)*...
                      (colums-1)+fix((lines(north,1)-1)/(rows-1))+1;
      lines(north,1)=col(north)-0.5+wexT(lines(north,1));
```

```
      lines(north,2)=rows-row(north)+0.5;
    end
    if length(west)>0
      lines(west,2)=rows-row(west)+0.5-nsx(lines(west,1));
      lines(west,1)=col(west)-0.5;
    end
    if length(south)>0
      lines(south,1)=rem(lines(south,1)-1,rows-1)*...
                      (colums-1)+fix((lines(south,1)-1)/(rows-1))+1;
      lines(south,1)=col(south)-0.5+wexT(lines(south,1)+colums-1);
      lines(south,2)=rows-row(south)-0.5;
    end
    if length(east)>0
      lines(east,2)=rows-row(east)+0.5-nsx(lines(east,1)+rows-1);
      lines(east,1)=col(east)+0.5;
    end
  end
  if exist('matlabpath')~=5
    do_fortran_indexing = save_do_fortran_indexing;
    empty_list_elements_ok = save_empty_list_elements_ok;
  end
```

1.2.29 ejpglist.m

```
%%NAME
%%  ejpglist  - generate photoprints of a JPEG-filelist
%%
%%SYNOPSIS
%%  ejpglist([listFileName[,maxPhotoSize[,fitPhoto[,outputFileName]]]])
%%
%%PARAMETER(S)
%%  listFileName         textfile of JPEG-filenames
%%                         one name per line
%%                       default=current directory
%%  maxPhotoSize         vector [width heigth] of photos
%%                       default=[90 120] (90mmx120mm)
%%  fitPhoto             switch, 0=off 1=fit photos to maxPhotoSize,default=0
%%  outputFileName       Praefix of eps-outputfile
%%                       default='photos' ->photos01.jpg,photos02.jpg, ...
%%
% written by stefan.mueller stefan.mueller@fgan.de (C) 2003
function ejpglist(listFileName,maxPhotoSize,fitPhoto,outputFileName)
  if nargin>4
    eusage('ejpglist(listFileName[,maxPhotoSize[,fitPhoto[,outputFileName]]])');
  end
  if nargin<4
    outputFileName='jpglist';
  end
  if nargin<3
    fitPhoto=0;
  end
  if nargin<2
    maxPhotoSize=[90 120];
```

```
end
if nargin<1
  listFileName='.';
end
eglobpar;
einit;
textDis=2;
textSize=4;
rows1=fix(eWinHeight/maxPhotoSize(1));
cols1=fix(eWinWidth/maxPhotoSize(2));
rows2=fix(eWinHeight/maxPhotoSize(2));
cols2=fix(eWinWidth/maxPhotoSize(1));
if rows1*cols1>rows2*cols2
 rows=rows1;
 cols=cols1;
 imgH=maxPhotoSize(1);
 imgW=maxPhotoSize(2);
else
 rows=rows2;
 cols=cols2;
 imgH=maxPhotoSize(2);
 imgW=maxPhotoSize(1);
end
imgFac=imgH/imgW;
nPerPage=rows*cols;
offsetX=(eWinWidth-cols*imgW)/2;
offsetY=(eWinHeight-rows*imgH);
[tabX tabY]=etabdef(rows,cols,offsetX,offsetY,cols*imgW,rows*imgH);
ttabH=(ePageHeight-eWinHeight)/2+offsetY;
if ttabH/cols>5
  ttabH=5*cols;
end
[ttabX ttabY]=etabdef(rows,cols,offsetX,offsetY-ttabH,cols*imgW,ttabH);

% list of images
list=etxtread(listFileName);
[lpos n]=etxtlpos(list);
lpos=flipud(lpos);
nPages=ceil(n/nPerPage);
outtext=sprintf('Writing %d eps-file(s) with %d images ...',...
                nPages,n);
disp(outtext);
page=0;
while n>0
  page=page+1;
  cRows=ceil(n/cols);
  if cRows<rows
    rows=cRows;
    offsetX=(eWinWidth-cols*imgW)/2;
    offsetY=(eWinHeight-rows*imgH);
    [tabX tabY]=etabdef(rows,cols,offsetX,offsetY,cols*imgW,rows*imgH);
    ttabH=(ePageHeight-eWinHeight)/2+offsetY;
    if ttabH/cRows>5
      ttabH=5*cRows;
```

```
  end
  [ttabX ttabY]=etabdef(rows,cols,offsetX,offsetY-ttabH,cols*imgW,ttabH);
end
if nPages>1
  epsFileName=sprintf('%s%d%d.eps',outputFileName,...
    floor(page/10),rem(page,10));
else
  epsFileName=sprintf('%s.eps',outputFileName);
end
eopen(epsFileName);
eglobpar;
for i=1:rows
  for j=1:cols
    if n>0
      % outputfile
      imgFileName=list(lpos(n,1):lpos(n,2));
      [image head]=ejpgread(imgFileName);
      jpgH=head(2);jpgW=head(3);
      outtext=sprintf('%s with %dx%d Pixel loaded',...
        imgFileName,jpgH,jpgW);
      disp(outtext);
      % rotation
      if abs(jpgH/jpgW-imgFac)>abs(jpgW/jpgH-imgFac)
        rotationAngle=90;
        imgR=jpgW/jpgH;
      else
        rotationAngle=0;
        imgR=jpgH/jpgW;
      end
      if fitPhoto
        photoW=imgW;
        photoH=imgH;
      else
        if imgH/imgW<imgR
          photoH=imgH;
          photoW=imgH/imgR;
        else
          photoW=imgW;
          photoH=imgW*imgR;
        end
      end
      if rotationAngle>0
        x=tabX(j,1)+photoW;
        eframe(x,tabY(i,1),photoH,photoW,0,image,head,rotationAngle);
      else
        x=tabX(j,1);
        eframe(x,tabY(i,1),photoW,photoH,0,image,head);
      end
      slPos=findstr(imgFileName,'/');
      slPosL=length(slPos);
      if slPosL>0
        iFileName=imgFileName(slPos(slPosL)+1:length(imgFileName));
      else
        iFileName=imgFileName;
```

```
          end
          etabtext(ttabX,ttabY,i,j,iFileName,0);
          n=n-1;
        end
      end
    end
    eclose
  end
```

1.2.30 ejpgread.m

```
%%NAME
%%  ejpgread  - read JPEG-file
%%
%%SYNOPSIS
%%  [image,head]=ejpgread(jpgFileName)
%%
%%PARAMETER(S)
%%  imageFileName  name of JPEG-file e.g. 'photo.jpg'
%%  image       whole JPEG-file in a vector of uchar
%%  head        1 x 4 vector, [sizeOfJpegFile rowsOfImage colsOfImage rgb]
%%              rgb=1 if color image, rgb=0 if black and white image
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
function [image,head]=ejpgread(filename)
  if (nargin >1)
    eusage('[image,head]=ejpgread(filename)');
  end
  jpgFile=fopen(filename ,'r');
  [image n]=fread(jpgFile,inf,'uchar');
  ipos=3;
  while image(ipos)==255
    mark=image(ipos+1);
    nextMark=image(ipos+2)*256+image(ipos+3);
    if mark==192
      nrows=image(ipos+5)*256+image(ipos+6);
      ncols=image(ipos+7)*256+image(ipos+8);
      if image(ipos+9)==3
        rgb=1;
      else
        rgb=0;
      end
    end
    ipos=ipos+nextMark+2;
  end
  head=[n nrows ncols rgb];
```

1.2.31 elineip.m

```
%%NAME
%%  elineip  - linear interpolation of a vector
%%
```

```
%%SYNOPSIS
%%  yi=elineip(x,y,xi)
%%
%%PARAMETER(S)
%%  x             sample x vector
%%  y             sample y vector
%%  xi            x vector for interpolation
%%  yi            interpolated y vector
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
function yi=elineip(x,y,xi)
   if (nargin ~= 3)
    eusage('yi= elineip(x,y,xi)');
  end

  n=length(x);
  [sv sp]=sort([x xi]);
  idx(sp)=cumsum(sp<n);
  idx=idx(n+1:n+length(xi));
  in=1:n-1;
  inp=in+1;
  dy=y(inp)-y(in);
  dx=x(inp)-x(in);
  nz=find(dx);
  m=dx-dx;
  m(nz)=dy(nz)./dx(nz);
  yi=(xi-x(idx)).*m(idx)+y(idx);
```

1.2.32 elines.m

```
%%NAME
%%  elines  - draw lines
%%
%%SYNOPSIS
%%  elines(xData,yData[,lineWidth[,dash[,color]]])
%%
%%PARAMETER(S)
%%  xData       matrix(2xn) of x0,x1-data of lines
%%  yData       matrix(2xn) of y0,y1-data of lines
%%  lineWidth   width of lines
%%              default: lineWidth=eLineWidth
%%  dash        if dash=0 then draw solid lines
%%              else value of dash is the distance of dashes
%%              default: dash=eLineDash
%%  color       vector of line color ([r g b])
%%              default: color=eLineColor
%%
%%GLOBAL PARAMETER(S)
%%  eLineWidth
%%  eLineDash
%%  eLineColor
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
```

```
function elines (xData,yData,lineWidth,dash,color)
  if nargin<2 | nargin>5
    eusage('elines(xData,yData,[,lineWidth[,dash[,color]]])');
  end
  eglobpar;
  if nargin<5
    color=eLineColor;
  end
  if nargin<4
    dash=eLineDash;
  end
  if nargin<3
    lineWidth=eLineWidth;
  end
  [xr xc]=size(xData);
  if xr>2
    xData=xData';
    xc=xr;
  end
  xData=reshape(xData,1,2*xc);
  [yr yc]=size(yData);
  if yr>2
    yData=yData';
    yc=yr;
  end
  yData=reshape(yData,1,2*yc);
    exyline(eFile,0,0,xData*eFac,yData*eFac,...
            color,dash*eFac,lineWidth*eFac);
```

1.2.33 eopen.m

```
%%NAME
%%  eopen  - open EPS-file, define size of page, size of window and
%%           call 'einit' to initialize the global parameter
%%
%%SYNOPSIS
%%  eopen([ epsFileName[,pageOrientation[,winWidth,winHeight
%%        [,winShift[,xScaleFac,yScaleFac[,pageWidth,pageHeight
%%        [,pageReflection]]]]]])
%%
%%PARAMETER(S)
%%  epsFileName       name of eps-file (default name is defined as eFileName)
%%  pageOrientation   page orientation,
%%                    0=portrait 1=landscape 2=upside-down 3=seaside
%%  winWidth          width of window(=eps bounding-box)
%%  winHeight         height of window(=eps bounding-box)
%%  winShift          shift-vector of window, [xOffset yOffset]
%%                    ,shift of window center on page,
%%                    default vector is [0 0]=middle of page
%%  xScaleFac         scale factor 1= no resize
%%  yScaleFac         scale factor 1= no resize
%%  pageWidth         width of page
%%  pageHeight        height of page
```

```
%%  pageReflection    refection 1=on 0=off
%%
%%GLOBAL PARAMETER(S)
%%  eFileName
%%  ePageWidth
%%  ePageHeight
%%  ePageScaleFac
%%  ePageOrientation
%%  ePageReflection
%%  eUserUnit
%%  eWinWidth
%%  eWinHeight
%%  eFonts
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eopen(epsFileName,pageOrientation,winWidth,winHeight,winShift,xScaleFac,yScaleFac,pageWidth
  if nargin>10 | nargin==3 | nargin==6 | nargin==8
    eusage('eopen([epsFileName[,pageOrientation,[winWidth,winHeight[,winShift[,xScaleFac,yScaleFac[,
  end
  eglobpar;
  einit;
  ePlotLineNo=0;
  ePolarPlotLineNo=0;
  ePieSliceNo=0;
  if nargin>0
    eFileName=epsFileName;
  end
  if nargin>1
    ePageOrientation=pageOrientation;
  end
  if nargin==2 & rem(ePageOrientation,2)
    winW=eWinWidth;
    eWinWidth=eWinHeight;
    eWinHeight=winW;
  end
  if nargin>3
    eWinWidth=winWidth;
    eWinHeight=winHeight;
  end
  if nargin<5
    winShift=[0 0];
  end
  if nargin>5
    eXScaleFac=xScaleFac;
    eYScaleFac=yScaleFac;
  end
  if nargin>7
    ePageWidth=pageWidth;
    ePageHeight=pageHeight;
  end
  if nargin>9
    ePageReflection=pageReflection;
  end
```

```
  % open eps file
  eFile=fopen(eFileName,'w');
  if eFile<0
    errortext=sprintf('error in eopen: can not open %s',eFileName);
    disp(errortext);
  else
    % write eps head
    ehead(eFile,eWinWidth*eFac,eWinHeight*eFac,...
          ePageWidth*eFac,ePageHeight*eFac,...
          ePageOrientation,eXScaleFac,eYScaleFac,...
          winShift(1)*eFac,winShift(2)*eFac,ePageReflection);

    % reencode fonts
    newFonts=erencode(eFile,eFonts(1,:));
    for i=2:size(eFonts,1)
      newFonts=[newFonts;erencode(eFile,eFonts(i,:))];
    end
    eFonts=newFonts;
  end
```

1.2.34 eparam.m

```
%%NAME
%%  eparam  -  print parameter text in two columns under plots
%%
%%SYNOPSIS
%%  eparam(text1,text2[,x,y])
%%
%%PARAMETER(S)
%%  text1        text of the left column
%%  text2        text of the right column
%%  x            x-coordinate of start position
%%  y            y-coordinate of start position
%%
%%GLOBAL PARAMETER(S)
%%  eParamPos
%%  eParamFontSize
%%  eParamTextValueDistance
%%  eParamTextFont
%%  eParamValueFont
%%  eParamLineDistance
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eparam(text1,text2,x,y)
  if nargin~=2 & nargin~=4
    eusage('eparam(text1,text2[,x,y])');
  end
  eglobpar;
  if nargin<4
    x=eParamPos(1);
    y=eParamPos(2);
  else
    eParamPos(1)=x;
```

```
    eParamPos(2)=y;
  end
  lineDistance=eParamLineDistance/100*eParamFontSize;
  valueDistance=eParamTextValueDistance/20*eParamFontSize;
  valueOffset=eParamFontSize;
  [xw yh]=etabdef(1,3,x,y,2*valueDistance,eParamFontSize,[10 1 10]);
  etabtext(xw,yh,1,1,text1,-1,eParamTextFont);
  etabtext(xw,yh,1,2,':',0,eParamTextFont);
  etabtext(xw,yh,1,3,text2,1,eParamValueFont);
  eParamPos(2)=eParamPos(2)-lineDistance;
```

1.2.35 epie.m

```
%%NAME
%%  epie - draw a pie chart
%%
%%SYNOPSIS
%%  angles=epie([value[,valueText[,legendText[,dash[,offset[,color]]]]]])
%%
%%PARAMETER(S)
%%  value          value of pie slide
%%  valueText      text of value , if empty  string then no text at pie slice
%%  legendText     text of legend, if empty  string then no legend
%%  dash           border type,0=solid line,>0=dash length,
%%                 <0=fill slice with color
%%  offset         radial offset of pieslice, default=0
%%  color          color of pie, vector [r g b]
%%  angles         n x 2 matrix of pie slice angles, if epie without parameter
%%                 [pieSlice1StartAngle pieSlice1SizeAngle;
%%                  pieSlice2StartAngle ...
%%
%%GLOBAL PARAMETER(S)
%%  ePolarPlotAreaCenterPos
%%  ePolarPlotAreaRadMax
%%  ePolarPlotAreaValStart
%%  ePolarPlotAreaValEnd
%%  ePolarPlotAreaAngStart;
%%  ePolarPlotAreaAngEnd
%%  ePolarPlotAreaRadMax
%%  ePolarPlotAreaRadMin
%%  ePlotLegendTextFont
%%  ePlotLegendFontSize
%%  ePlotLineDash;
%%  ePlotLineDash;
%%  eAxesTicLongLength
%%  eAxesValueSpace
%%  eAxesValueFontSize
%%  eAxesLineWidth
%%  eAxesColor
%%  ePolarAxisRadScale
%%  ePolarAxisAngScale
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
```

```
function angles=epie(value,valueText,legendText,dash,offset,color)
  if nargin>6
    eusage('angles=epie([value[,valueText[,legendText[,dash[,offset[,color]]]]]])');
  end
  eglobpar;
  if (nargin==0)
    %finish plotting

    % write title
    eptitle;
    angles=zeros(ePieSliceNo,2);
    legendPos=ePlotLegendPos;
    for i=1:ePieSliceNo
      parameter=sprintf('global ePieSliceW%d;',i);
      eval(parameter);
      parameter=sprintf('width=ePieSliceW%d;',i);
      eval(parameter);
      parameter=sprintf('global ePieSliceC%d;',i);
      eval(parameter);
      parameter=sprintf('color=ePieSliceC%d;',i);
      eval(parameter);
      parameter=sprintf('global ePieSliceDash%d;',i);
      eval(parameter);
      parameter=sprintf('dash=ePieSliceDash%d;',i);
      eval(parameter);
      parameter=sprintf('global ePieSliceLegText%d;',i);
      eval(parameter);
      parameter=sprintf('legendText=ePieSliceLegText%d;',i);
      eval(parameter);
      parameter=sprintf('global ePieSliceValue%d;',i);
      eval(parameter);
      parameter=sprintf('value=ePieSliceValue%d;',i);
      eval(parameter);
      parameter=sprintf('global ePieSliceValueText%d;',i);
      eval(parameter);
      parameter=sprintf('valueText=ePieSliceValueText%d;',i);
      eval(parameter);
      parameter=sprintf('global ePieSliceOffset%d;',i);
      eval(parameter);
      parameter=sprintf('offset=ePieSliceOffset%d;',i);
      eval(parameter);
      pieSize=ePolarPlotAreaAngEnd-ePolarPlotAreaAngStart;
      pieSliceSize=pieSize*value/ePolarPlotAreaValStart;
      if i==1
        angle=ePolarPlotAreaAngStart;
      end
      angles(i,:)=[angle pieSliceSize];
      if max(size(dash))==1
        dash=dash*eFac;
      end
      epiesxy(eFile,...
              ePolarPlotAreaCenterPos(1)*eFac,...
              ePolarPlotAreaCenterPos(2)*eFac,...
```

```
        ePolarPlotAreaRadMin*eFac,...
        ePolarPlotAreaRadMax*eFac,...
        angle,...
        pieSliceSize,...
        color,...
        dash,...
        width*eFac,...
        offset*eFac);

  if strcmp(valueText,'')~=1
    degAngle=rem(angle+pieSliceSize/2+360,360);
    vTextAngle=degAngle*pi/180;
    cossin=[cos(vTextAngle) sin(vTextAngle)];
    vTextR1=(ePolarPlotAreaRadMax+offset)*cossin;
    vTextR2=vTextR1+eAxesTicLongLength*cossin;
    vTextR3=vTextR2+eAxesValueSpace*cossin;
    pos1=ePolarPlotAreaCenterPos+vTextR1;
    pos2=ePolarPlotAreaCenterPos+vTextR2;
    pos3=ePolarPlotAreaCenterPos+vTextR3;
    elines([pos1(1);pos2(1)],[pos1(2);pos2(2)],eAxesLineWidth,0,eAxesColor);
    if ((degAngle<=60) & (degAngle>=0)) | ...
       ((degAngle<360) & (degAngle>300))
      yShift=-eAxesValueFontSize/4;
      al=1;
    end
    if (degAngle<=120) & (degAngle>60)
      al=0;
      yShift=0;
    end
    if (degAngle<=240) & (degAngle>120)
      yShift=-eAxesValueFontSize/4;
      al=-1;
    end
    if (degAngle<=300) & (degAngle>240)
      yShift=-eAxesValueFontSize;
      al=0;
    end
    etext(valueText,pos3(1),pos3(2)+yShift,...
          eAxesValueFontSize,al,5,0,eAxesColor);
  end
  if strcmp(legendText,'')~=1
    eplotlg(eFile,...
      (ePolarPlotAreaCenterPos(1)-ePolarPlotAreaRadMax+legendPos(1))*...
      eFac,...
      (ePolarPlotAreaCenterPos(2)-ePolarPlotAreaRadMax+legendPos(2))*...
      eFac,...
      color,...
      dash,...
      width*eFac,...
      legendText,...
      eFonts(ePlotLegendTextFont,:),...
      ePlotLegendFontSize*eFac,eAxesColor);
    legendPos(2)=legendPos(2)-ePlotLegendDistance/70*ePlotLegendFontSize;
  end
```

```
      angle=angle+pieSliceSize;
    end %for
    ePieSliceNo=0;
  else
    % add plot line
    ePieSliceNo=ePieSliceNo+1;

    %lineWidth
    width=eAxesLineWidth;
    parameter=sprintf('global ePieSliceW%d;',ePieSliceNo);
    eval(parameter);
    parameter=sprintf('ePieSliceW%d=width;',ePieSliceNo);
    eval(parameter);

    %dash
    if (nargin<4)
      dash=ePlotLineDash;
    end
    parameter=sprintf('global ePieSliceDash%d;',ePieSliceNo);
    eval(parameter);
    parameter=sprintf('ePieSliceDash%d=dash;',ePieSliceNo);
    eval(parameter);

    %color
    if (nargin<6)
      if dash>=0
        color=eAxesColor;
      else
        color=0.5+...
          [0.5*sin(ePieSliceNo*3) 0.4*cos(ePieSliceNo*2) -0.3*cos(ePieSliceNo)];
      end
    end
    parameter=sprintf('global ePieSliceC%d;',ePieSliceNo);
    eval(parameter);
    parameter=sprintf('ePieSliceC%d=color;',ePieSliceNo);
    eval(parameter);

    % legend text
    if (nargin<3)
      legendText='';
    end
    parameter=sprintf('global ePieSliceLegText%d;',ePieSliceNo);
    eval(parameter);
    parameter=sprintf('ePieSliceLegText%d=legendText;',ePieSliceNo);
    eval(parameter);

    % value
    parameter=sprintf('global ePieSliceValue%d;',ePieSliceNo);
    eval(parameter);
    parameter=sprintf('ePieSliceValue%d=value;',ePieSliceNo);
    eval(parameter);

    % valueText
    if (nargin<2)
```

```
    valueText='';
  end
  parameter=sprintf('global ePieSliceValueText%d;',ePieSliceNo);
  eval(parameter);
  parameter=sprintf('ePieSliceValueText%d=valueText;',ePieSliceNo);
  eval(parameter);

  % offset
  if (nargin<5)
    offset=0;
  end
  parameter=sprintf('global ePieSliceOffset%d;',ePieSliceNo);
  eval(parameter);
  parameter=sprintf('ePieSliceOffset%d=offset;',ePieSliceNo);
  eval(parameter);

  if ePieSliceNo==1
    ePolarPlotAreaValStart=value;
  else
    ePolarPlotAreaValStart=ePolarPlotAreaValStart+value;
  end
end
```

1.2.36 epline.m

```
%%NAME
%%  epline  - draw polyline
%%
%%SYNOPSIS
%%  epline(xData,yData[,lineWidth[,dash[,color]]])
%%
%%PARAMETER(S)
%%  xData       vector of x-values of polyline
%%  yData       vector of y-values of polyline
%%  lineWidth   width of polyline
%%              default: lineWidth=eLineWidth
%%  dash          if a scalar
%%                  =0  solid lines,
%%                  >0  dash length
%%                  <0  fill polylines with color
%%                default: dash=eLineDash
%%                if a matrix and color=-1
%%                  dash is the image of polyline
%%                  and filled with RGB values
%%                  value=R*2^16+G*2^8+B) and R,G,B are integer of 0:255
%%                if a matrix and color is a colormap
%%                  dash is the image of polyline
%%                  and filled with indices of colormap
%%  color         if dash>=0 vector of frame color ([r g b])
%%                if dash<0  vector of background color
%%                if dash a matrix then colormap of image or -1
%%                default: dash=eLineColor
%%
```

```
%%GLOBAL PARAMETER(S)
%%  eLineColor
%%  eLineDash
%%  eLineWidth
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function epline (xData,yData,lineWidth,dash,color)
  if nargin<2 | nargin>5
    eusage('epline(xData,yData,[,lineWidth[,dash[,color]]])');
  end
  eglobpar;
  if nargin<5
    color=eLineColor;
  end
  if nargin<4
    dash=eLineDash;
  end
  if nargin<3
    lineWidth=eLineWidth;
  end
  xData=xData*eFac;
  yData=yData*eFac;
  if max(size(dash))==1
    if dash<0
      exyplotf(eFile,0,0,xData,yData,color);
    else
      exyplot( eFile,0,0,xData,yData,color,dash*eFac,...
             lineWidth*eFac);
    end
  else
    exyploti(eFile,0,0,xData,yData,dash,color);
  end
```

1.2.37 eplo2win.m

```
%%NAME
%%  eplo2win - transform coordinates , plotarea to window
%%
%%SYNOPSIS
%%  [winX winY]=eplo2win(ploX,plotY)
%%
%%PARAMETER(S)
%%  ploX     x-vector of coordinates of plotarea
%%  ploY     y-vector of coordinates of plotarea
%%  winX     x-vector of coordinates of window
%%  winY     y-vector of coordinates of window
%%
%%GLOBAL PARAMETER(S)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function [winX,winY]=eplo2win(ploX,ploY)
  if nargin~=2
    eusage('[winX winY]=eplo2win(ploX,ploY)');
```

```
  end
  eglobpar;
  if  eXAxisSouthScaleType==2
    ploX=log10(ploX);
  end
  if  eYAxisWestScaleType==2
    ploY=log10(ploY);
  end
  xFac=ePlotAreaWidth/...
      (ePlotAreaXValueEnd-ePlotAreaXValueStart);
  yFac=ePlotAreaHeight/...
      (ePlotAreaYValueEnd-ePlotAreaYValueStart);
  winX=ePlotAreaPos(1)+(ploX-ePlotAreaXValueStart)*xFac;
  winY=ePlotAreaPos(2)+(ploY-ePlotAreaYValueStart)*yFac;
```

1.2.38 eplot.m

```
%%NAME
%%  eplot  - make linear plot
%%
%%SYNOPSIS
%%  eplot ([xData,[yData,[legendText,[dash,[color[,width]]]]]])
%%
%%PARAMETER(S)
%%  xData           vector of x-data
%%                  or matrix(2xn) of x0,x1-data to plot lines
%%  yData           vector of y-data
%%                  or matrix(2xn) of y0,y1-data to plot lines
%%  legendText      text of legend, if empty  string then no legend
%%  dash           if a scalar
%%                   =0  solid plot line,
%%                   >0  dash length
%%                   <0  fill plot line with color
%%                 default: dash=eLineDash
%%                 if a string then dash is a name of symbol
%%                 if a matrix and color=-1
%%                   dash is the image of plot
%%                   and filled with RGB values
%%                   value=R*2^16+G*2^8+B) and R,G,B are integer of 0:255
%%                 if a matrix and color is a colormap
%%                   dash is the image of plot
%%                   and filled with indices of colormap
%%                 if a string dash is filename of a JPEG-file
%%  color          if dash>=0 vector of plot color ([r g b])
%%                 if dash<0  vector of background color
%%                 if dash a matrix then colormap of image or -1
%%                 default: dash=eLineColor
%%  width          width of plot line
%%
%% Important: eplot without parameters closes the current plot explicit.
%%            it's useful for several plot on one page
%%
%%GLOBAL PARAMETER(S)
```

```
%%  ePlotAreaPos
%%  ePlotAreaWidth
%%  ePlotAreaHeight
%%  eXAxisSouthScale
%%  eYAxisWestScale
%%  ePlotAreaXValueStart
%%  ePlotAreaXValueEnd
%%  ePlotAreaYValueStart
%%  ePlotAreaYValueEnd
%%  ePlotLineInterpolation
%%  ePlotLineWidth
%%  ePlotLineColor;
%%  ePlotLineDash;
%%  ePlotLegendPos;
%%  ePlotLegendTextFont
%%  ePlotLegendFontSize
%%  ePlotLegendDistance
%%  eAxesColor
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eplot(xData,yData,legendText,dash,color,width)
  if nargin>6
    eusage('eplot([xData,[yData,[legendText,[dash,[color[,width]]]]]])');
  end
  eglobpar;
  if (nargin==0)
    %finish plotting

    % write title
    etitle;

    %value range
    if eXAxisSouthScale(1)~=eXAxisSouthScale(3)
      %fix scale
      ePlotAreaXValueStart=eXAxisSouthScale(1);
      ePlotAreaXValueEnd=eXAxisSouthScale(3);
    end
    if eYAxisWestScale(1)~=eYAxisWestScale(3)
      %fix scale
      ePlotAreaYValueStart=eYAxisWestScale(1);
      ePlotAreaYValueEnd=eYAxisWestScale(3);
    end
    if eXAxisSouthScaleType==2
      if (ePlotAreaXValueStart>0)&(ePlotAreaXValueEnd>0)
        ePlotAreaXValueStart=log10(ePlotAreaXValueStart);
        ePlotAreaXValueEnd=log10(ePlotAreaXValueEnd);
      else
        error('xValues<=0 for log scale');
      end
    end
    if eYAxisWestScaleType==2
      if (ePlotAreaYValueStart>0)&(ePlotAreaYValueEnd>0)
        ePlotAreaYValueStart=log10(ePlotAreaYValueStart);
        ePlotAreaYValueEnd=log10(ePlotAreaYValueEnd);
```

```
  else
    error('yValues<=0 for log scale');
  end
end
if (eXAxisSouthScale(1)==eXAxisSouthScale(3))&eAxesCrossOrigin
  xRange=ePlotAreaXValueEnd-ePlotAreaXValueStart;
  ePlotAreaXValueStart=ePlotAreaXValueStart-0.05*xRange;
  ePlotAreaXValueEnd=ePlotAreaXValueEnd+0.05*xRange;
end
if eYAxisWestScale(1)==eYAxisWestScale(3)
  yRange=ePlotAreaYValueEnd-ePlotAreaYValueStart;
  ePlotAreaYValueStart=ePlotAreaYValueStart-0.05*yRange;
  ePlotAreaYValueEnd=ePlotAreaYValueEnd+0.05*yRange;
end
egrid;

% plot line and write legend
ePlotAreaXFac=ePlotAreaWidth*eFac/...
  (ePlotAreaXValueEnd-ePlotAreaXValueStart);
ePlotAreaYFac=ePlotAreaHeight*eFac/...
  (ePlotAreaYValueEnd-ePlotAreaYValueStart);
legendPos=ePlotLegendPos;
for i=1:ePlotLineNo
  parameter=sprintf('global ePlotLineWidth%d;',i);
  eval(parameter);
  parameter=sprintf('width=ePlotLineWidth%d;',i);
  eval(parameter);
  parameter=sprintf('global ePlotLineColor%d;',i);
  eval(parameter);
  parameter=sprintf('color=ePlotLineColor%d;',i);
  eval(parameter);
  parameter=sprintf('global ePlotLineDash%d;',i);
  eval(parameter);
  parameter=sprintf('dash=ePlotLineDash%d;',i);
  eval(parameter);
  parameter=sprintf('global ePlotLegendText%d;',i);
  eval(parameter);
  parameter=sprintf('legendText=ePlotLegendText%d;',i);
  eval(parameter);
  parameter=sprintf('global ePlotXData%d;',i);
  eval(parameter);
  parameter=sprintf('xData=ePlotXData%d;',i);
  eval(parameter);
  parameter=sprintf('global ePlotYData%d;',i);
  eval(parameter);
  parameter=sprintf('yData=ePlotYData%d;',i);
  eval(parameter);
  if eXAxisSouthScaleType==2
    xData=log10(xData);
  end
  xData=(xData-ePlotAreaXValueStart)*ePlotAreaXFac;
  [xr xc]=size(xData);
  if eYAxisWestScaleType==2
    yData=log10(yData);
```

```
end
yData=(yData-ePlotAreaYValueStart)*ePlotAreaYFac;
[yr yc]=size(yData);

eclip(eFile,ePlotAreaPos(1)*eFac,ePlotAreaPos(2)*eFac,...
      ePlotAreaWidth*eFac,ePlotAreaHeight*eFac);
if isstr(dash)
  n=size(xData,2);
  exyplots(eFile,...
    ePlotAreaPos(1)*eFac,...
    ePlotAreaPos(2)*eFac,...
    xData(1,:),...
    yData(1,:),...
    ones(1,n),...
    ones(1,n),...
    zeros(1,n),...
    dash,...
    color);

elseif max(size(dash))>1
  exyploti(eFile,...
    ePlotAreaPos(1)*eFac,...
    ePlotAreaPos(2)*eFac,...
    xData(1,:),...
    yData(1,:),...
    dash,...
    color);
elseif dash<0;
  exyplotf(eFile,...
    ePlotAreaPos(1)*eFac,...
    ePlotAreaPos(2)*eFac,...
    xData(1,:),...
    yData(1,:),...
    color)
else
  if ePlotLineInterpolation
    exyplotc(eFile,...
      ePlotAreaPos(1)*eFac,...
      ePlotAreaPos(2)*eFac,...
      xData(1,:),...
      yData(1,:),...
      color,...
      dash*eFac,...
      width*eFac);
  elseif xr==1
    exyplot(eFile,...
      ePlotAreaPos(1)*eFac,...
      ePlotAreaPos(2)*eFac,...
      xData(1,:),...
      yData(1,:),...
      color,...
      dash*eFac,...
      width*eFac);
  else
```

```
        xData=reshape(xData,1,2*xc);
        yData=reshape(yData,1,2*yc);
        exyline(eFile,...
          ePlotAreaPos(1)*eFac,...
          ePlotAreaPos(2)*eFac,...
          xData,...
          yData,...
          color,...
          dash*eFac,...
          width*eFac);
      end
    end
    eclip(eFile,0,0,0,0);
    if strcmp(legendText,'')~=1
      eplotlg(eFile,...
        (ePlotAreaPos(1)+legendPos(1))*eFac,...
        (ePlotAreaPos(2)+legendPos(2))*eFac,...
        color,...
        dash,...
        width*eFac,...
        legendText,...
        eFonts(ePlotLegendTextFont,:),...
        ePlotLegendFontSize*eFac,eAxesColor);
      legendPos(2)=legendPos(2)-ePlotLegendDistance/70*ePlotLegendFontSize;
    end
  end
  eaxes;
  ePlotLineNo=0;
else
  % add plot line
  ePlotLineNo=ePlotLineNo+1;
  %width
  if (nargin<6)
    width=ePlotLineWidth;
  end
  parameter=sprintf('global ePlotLineWidth%d;',ePlotLineNo);
  eval(parameter);
  parameter=sprintf('ePlotLineWidth%d=width;',ePlotLineNo);
  eval(parameter);
  %color
  if (nargin<5)
    color=ePlotLineColor;
  end
  parameter=sprintf('global ePlotLineColor%d;',ePlotLineNo);
  eval(parameter);
  parameter=sprintf('ePlotLineColor%d=color;',ePlotLineNo);
  eval(parameter);

  %dash
  if (nargin<4)
    dash=ePlotLineDash;
  end
  parameter=sprintf('global ePlotLineDash%d;',ePlotLineNo);
  eval(parameter);
```

```
parameter=sprintf('ePlotLineDash%d=dash;',ePlotLineNo);
eval(parameter);

% legend text
if (nargin<3)
  legendText='';
end
parameter=sprintf('global ePlotLegendText%d;',ePlotLineNo);
eval(parameter);
parameter=sprintf('ePlotLegendText%d=legendText;',ePlotLineNo);
eval(parameter);

[xr xc]=size(xData);
if xr>2
  xData=xData';
end
if (nargin<2)
  yData=xData;
  xData=1:size(yData,2);
  if xr==2
    xData=[xData;xData];
  end
else
  [yr yc]=size(yData);
  if yr>2
   yData=yData';
  end
end
% data
parameter=sprintf('global ePlotXData%d;',ePlotLineNo);
eval(parameter);
parameter=sprintf('ePlotXData%d=xData;',ePlotLineNo);
eval(parameter);
parameter=sprintf('global ePlotYData%d;',ePlotLineNo);
eval(parameter);
parameter=sprintf('ePlotYData%d=yData;',ePlotLineNo);
eval(parameter);

%value range
xMin=min(min(xData));
xMax=max(max(xData));
if xMin<ePlotAreaXValueStart | ePlotLineNo==1
  ePlotAreaXValueStart=xMin;
end
if xMax>ePlotAreaXValueEnd | ePlotLineNo==1
  ePlotAreaXValueEnd=xMax;
end
yMin=min(min(yData));
yMax=max(max(yData));
if yMin<ePlotAreaYValueStart | ePlotLineNo==1
  ePlotAreaYValueStart=yMin;
end
if yMax>ePlotAreaYValueEnd | ePlotLineNo==1
  ePlotAreaYValueEnd=yMax;
```

```
    end
  end
```

1.2.39 epolar.m

```
%%NAME
%%  epolar - make polar plot
%%
%%SYNOPSIS
%%  epolar ([xData,[yData,[legendText,[dash,[color[,width]]]]]])
%%
%%PARAMETER(S)
%%  xData          vector of x-data
%%  yData          vector of y-data
%%  legendText     text of legend, if empty  string then no legend
%%  dash           0=solid line,>0=dash length,
%%                 <0=fill line,string=name of symbol
%%  color          color of plot, vetcor [r g b]
%%  width          width of plot
%%
%%GLOBAL PARAMETER(S)
%%  ePolarAxisRadScale
%%  ePolarAxisAngScale
%%  ePolarPlotAreaCenterPos
%%  ePolarPlotAreaRadMax
%%  ePolarPlotAreaValStart
%%  ePolarPlotAreaValEnd
%%  ePolarPlotAreaAngStart;
%%  ePolarPlotAreaAngEnd
%%  ePolarPlotAreaRadMax
%%  ePolarPlotAreaRadMin
%%  ePlotLegendPos
%%  ePlotLegendTextFont
%%  ePlotLegendFontSize
%%  ePlotLegendDistance;
%%  ePlotLineWidth
%%  ePlotLineColor;
%%  ePlotLineDash;
%%  eAxesColor;
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function epolar(xData,yData,legendText,dash,color,width)
  if nargin>6
    eusage('epolar([xData,[yData,[legendText,[dash,[color[,width]]]]]])');
  end
  eglobpar;
  if (nargin==0)
    %finish plotting

    % write title
    eptitle;

    %value range
```

```
if ePolarAxisRadScale(1)~=ePolarAxisRadScale(3)
  %fix scale
  ePolarPlotAreaValStart=ePolarAxisRadScale(1);
  ePolarPlotAreaValEnd=ePolarAxisRadScale(3);
end
if ePolarAxisRadScaleType==2
  if(ePolarPlotAreaValStart>0)&(ePolarPlotAreaValEnd>0)
     ePolarPlotAreaValStart=log10(ePolarPlotAreaValStart);
     ePolarPlotAreaValEnd=log10(ePolarPlotAreaValEnd);
  else
    error('yValues<=0');
  end
end
yRange=ePolarPlotAreaValEnd-ePolarPlotAreaValStart;
if yRange==0
  yRange=1;
end
if ePolarAxisAngScale(1)==ePolarAxisAngScale(3)
  ePolarAxisAngScale(1)=0;
  ePolarAxisAngScale(3)=ePolarPlotAreaAngEnd-ePolarPlotAreaAngStart;
end
egridpol;
eaxespol;

% plot line and write legend
ePolarPlotAreaFac=(ePolarPlotAreaRadMax-ePolarPlotAreaRadMin)*...
                    eFac/yRange;
legendPos=ePlotLegendPos;
for i=1:ePolarPlotLineNo
  parameter=sprintf('global ePolarPlotLineW%d;',i);
  eval(parameter);
  parameter=sprintf('width=ePolarPlotLineW%d;',i);
  eval(parameter);
  parameter=sprintf('global ePolarPlotLineC%d;',i);
  eval(parameter);
  parameter=sprintf('color=ePolarPlotLineC%d;',i);
  eval(parameter);
  parameter=sprintf('global ePolarPlotLineDash%d;',i);
  eval(parameter);
  parameter=sprintf('dash=ePolarPlotLineDash%d;',i);
  eval(parameter);
  parameter=sprintf('global ePolarPlotLegText%d;',i);
  eval(parameter);
  parameter=sprintf('legendText=ePolarPlotLegText%d;',i);
  eval(parameter);
  parameter=sprintf('global ePolarPlotXData%d;',i);
  eval(parameter);
  parameter=sprintf('xData=ePolarPlotXData%d;',i);
  eval(parameter);
  parameter=sprintf('global ePolarPlotYData%d;',i);
  eval(parameter);
  parameter=sprintf('yData=ePolarPlotYData%d;',i);
  eval(parameter);
  xData=xData+ePolarPlotAreaAngStart;
```

```
  if ePolarAxisRadScaleType==2
    yData=log10(yData);
  end
  yData=(yData-ePolarPlotAreaValStart)*ePolarPlotAreaFac+...
         ePolarPlotAreaRadMin*eFac;
  eclippol(eFile,...
           ePolarPlotAreaCenterPos(1)*eFac,...
           ePolarPlotAreaCenterPos(2)*eFac,...
           ePolarPlotAreaRadMin*eFac,...
           ePolarPlotAreaRadMax*eFac,...
           ePolarPlotAreaAngStart,...
           ePolarPlotAreaAngEnd);
  if isstr(dash)
    epolplos(eFile,...
      ePolarPlotAreaCenterPos(1)*eFac,...
      ePolarPlotAreaCenterPos(2)*eFac,...
      xData,...
      yData,...
      dash,...
      color)
  elseif dash<0;
    epolplof(eFile,...
      ePolarPlotAreaCenterPos(1)*eFac,...
      ePolarPlotAreaCenterPos(2)*eFac,...
      xData,...
      yData,...
      color)
  else
    epolplot(eFile,...
      ePolarPlotAreaCenterPos(1)*eFac,...
      ePolarPlotAreaCenterPos(2)*eFac,...
      xData,...
      yData,...
      color,...
      dash*eFac,...
      width*eFac);
  end
  eclippol(eFile);

  if strcmp(legendText,'')~=1
    eplotlg(eFile,...
      (ePolarPlotAreaCenterPos(1)-ePolarPlotAreaRadMax+legendPos(1))*...
      eFac,...
      (ePolarPlotAreaCenterPos(2)-ePolarPlotAreaRadMax+legendPos(2))*...
      eFac,...
      color,...
      dash,...
      width*eFac,...
      legendText,...
      eFonts(ePlotLegendTextFont,:),...
      ePlotLegendFontSize*eFac,eAxesColor);
    legendPos(2)=legendPos(2)-ePlotLegendDistance/70*ePlotLegendFontSize;
  end
end
```

```
    ePolarPlotLineNo=0;
  else
    % add plot line
    ePolarPlotLineNo=ePolarPlotLineNo+1;
    %width
    if (nargin<6)
      width=ePlotLineWidth;
    end
    parameter=sprintf('global ePolarPlotLineW%d;',ePolarPlotLineNo);
    eval(parameter);
    parameter=sprintf('ePolarPlotLineW%d=width;',ePolarPlotLineNo);
    eval(parameter);
    %color
    if (nargin<5)
      color=ePlotLineColor;
    end
    parameter=sprintf('global ePolarPlotLineC%d;',ePolarPlotLineNo);
    eval(parameter);
    parameter=sprintf('ePolarPlotLineC%d=color;',ePolarPlotLineNo);
    eval(parameter);

    %dash
    if (nargin<4)
      dash=ePlotLineDash;
    end
    parameter=sprintf('global ePolarPlotLineDash%d;',ePolarPlotLineNo);
    eval(parameter);
    parameter=sprintf('ePolarPlotLineDash%d=dash;',ePolarPlotLineNo);
    eval(parameter);

    % legend text
    if (nargin<3)
      legendText='';
    end
    parameter=sprintf('global ePolarPlotLegText%d;',ePolarPlotLineNo);
    eval(parameter);
    parameter=sprintf('ePolarPlotLegText%d=legendText;',ePolarPlotLineNo);
    eval(parameter);

    if (nargin==1)
      yData=xData;
      xStep=length(yData)/(ePolarPlotAreaAngEnd-ePolarPlotAreaAngStart)
      xData=1:length(yData);
    else
      rad2deg=180/pi;
      xData=xData*rad2deg;
    end
    % data
    parameter=sprintf('global ePolarPlotXData%d;',ePolarPlotLineNo);
    eval(parameter);
    parameter=sprintf('ePolarPlotXData%d=xData;',ePolarPlotLineNo);
    eval(parameter);
    parameter=sprintf('global ePolarPlotYData%d;',ePolarPlotLineNo);
    eval(parameter);
```

```
    parameter=sprintf('ePolarPlotYData%d=yData;',ePolarPlotLineNo);
    eval(parameter);

    %value range
    yMin=min(yData);
    yMay=max(yData);
    if yMin<ePolarPlotAreaValStart | ePolarPlotLineNo==1
      ePolarPlotAreaValStart=yMin;
    end
    if yMay>ePolarPlotAreaValEnd | ePolarPlotLineNo==1
      ePolarPlotAreaValEnd=yMay;
    end
  end
```

1.2.40 epolari.m

```
%%NAME
%%  epolari  - draw polar image of a matrix
%%
%%SYNOPSIS
%%  epolari(matrix[,colorMap])
%%
%%PARAMETER(S)
%%  matrix     matrix for image
%%             if colorMap=-1 then
%%                image is filled with RGB values
%%                value=R*2^16+G*2^8+B) and R,G,B are integer of 0:255
%%              else
%%                matrix is filled with indices of colormap
%%  colorMap   define own colormap
%%             default:colorMap=ecolors(eImageDefaultColorMap)
%%
%%GLOBAL PARAMETER(S)
%%  eImageDefaultColorMap
%%  ePolarPlotAreaCenterPos
%%  ePolarPlotAreaRadMax
%%  ePolarPlotAreaAngEnd
%%  ePolarPlotAreaAngStart
%%  ePolarPlotAreaRadMin
%%  ePolarPlotAreaRadMax
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function epolari(matrix,colorMap)
  if nargin>5
    eusage('epolari(matrix[,colorMap])');
  end
  eglobpar;
  if nargin<2
    colorMap=ecolors(eImageDefaultColorMap);
  end
  if nargin<1
    [matrix colorMap]=eppmread([ePath 'default.ppm']);
    ePlotTitleText='Photo';
```

```
  end

  % write title
  eptitle;

  nSteps=size(matrix,2);
  dPhi=(ePolarPlotAreaAngEnd-ePolarPlotAreaAngStart)/nSteps;
  radiusMin=ePolarPlotAreaRadMin*eFac;
  radiusMax=ePolarPlotAreaRadMax*eFac;
  dRadius=radiusMax-radiusMin;
  sectorH=dPhi/180*pi*radiusMax;
  fprintf(eFile,'gsave %1.2f %1.2f translate\n',...
          ePolarPlotAreaCenterPos(1)*eFac,...
          ePolarPlotAreaCenterPos(2)*eFac);
  fprintf(eFile,'%1.2f rotate\n',ePolarPlotAreaAngStart+dPhi/2);
  for i=1:nSteps
    fprintf(eFile,'%1.2f rotate\n',dPhi*(i-1));
    eclippol(eFile,0,0,radiusMin,radiusMax,-dPhi/2,dPhi/2);
    eimagexy(eFile,matrix(:,i)',colorMap,...
             radiusMin,-sectorH/2,dRadius,sectorH);
    eclippol(eFile);
    fprintf(eFile,'%1.2f rotate\n',-dPhi*(i-1));
  end
  fprintf(eFile,'grestore\n');
```

1.2.41 epolaris.m

```
%%NAME
%%  epolaris  - draw scaled polar image of a matrix
%%
%%SYNOPSIS
%%  x=epolaris(matrix[,colorMap[,legendOrientation[,legendScale]]])
%%
%%PARAMETER(S)
%%  x                   transformed output matrix with values of color numbers
%%  matrix              matrix for image
%%  colorMap            define own colormap
%%  legendOrientation   side of the image where the legend appears
%%                      character 's'(south),'n'(north),'w'(west) or 'e'(east)
%%                      (default orientation is south)
%%  legendScale         scale vector of legend [start step end]
%%                      special cases of scale vector are:
%%                      if start=0 and end=0 then autorange=on
%%                      if step=0 then autoscale=on
%%                      (default scale vector=[0 0 0])
%%
%%GLOBAL PARAMETER(S)
%%  eImageDefaultColorMap
%%  eImageLegendScale
%%  ePolarAxisRadScale
%%  ePolarAxisAngScale
%%  ePolarPlotAreaCenterPos
%%  ePolarPlotAreaAngStart
```

```
%%  ePolarPlotAreaAngEnd
%%  ePolarPlotAreaRadMax
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function x=epolaris (matrix,colorMap,legendOrientation,legendScale)
  if (nargin>4)
    eusage('x=epolaris(matrix[,colorMap[,legendOrientation]])');
  end
  eglobpar;
  if nargin<4
    legendScale=eImageLegendScale;
  end
  if nargin<3
    legendOrientation='s';
  end
  if nargin<2
    colorMap=ecolors(eImageDefaultColorMap);
  end
  if nargin<1
    [matrix colorMap]=eppmread([ePath 'default.ppm']);
    ePlotTitleText='Photo';
  end

  % scale image
  if legendScale(1)==legendScale(3)
    minval=min (min (matrix));
    maxval=max (max (matrix));
  else
    minval=legendScale(1);
    maxval=legendScale(3);
  end
  if eImageLegendScaleType==2
    minval=log10(minval);
    maxval=log10(maxval);
    matrix=log10(matrix);
  end
  legendScale(1)=minval;
  legendScale(3)=maxval;
  if (maxval - minval)<=1000*eps
    maxval=minval+1000*eps;
  end
  x = fix ((matrix - minval) / (maxval - minval) *...
        (length (colorMap) - 1)) + 1;
  epolari(x,colorMap);

  %fix scale
  if ePolarAxisRadScale(1)==ePolarAxisRadScale(3)
    ePolarAxisRadScale(1)=0;
    ePolarAxisRadScale(3)=100;
  end
  if ePolarAxisAngScale(1)==ePolarAxisAngScale(3)
    ePolarAxisAngScale(1)=0;
    ePolarAxisAngScale(3)=ePolarPlotAreaAngEnd-ePolarPlotAreaAngStart;
  end
```

```
  egridpol;
  eaxespol;

  % color legend
  wh=2*ePolarPlotAreaRadMax;
  eimgleg(ePolarPlotAreaCenterPos(1)-ePolarPlotAreaRadMax,...
          ePolarPlotAreaCenterPos(2)-ePolarPlotAreaRadMax,...
          wh,wh,colorMap,legendScale,legendOrientation);
```

1.2.42 eppmread.m

```
%%NAME
%%  eppmread  - read PPM-file
%%
%%SYNOPSIS
%%  [image,colormap]=eppmread(filename)
%%
%%PARAMETER(S)
%%  filename    name of PPM-file
%%  image       image matrix
%%              if colormap used then
%%                image is filled with indices of colormap
%%              else
%%                image is filled with RGB values
%%                value=R*2^16+G*2^8+B) and R,G,B are integer of 0:255
%%                that's a very fast way
%%  colormap    color table
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
function [image,colormap]= eppmread (filename)
  if (nargin >1)
    eusage('[image,colormap] = eppmread(filename)');
  end
  eglobpar;
  if exist('ePath')
    if isempty(ePath)
      einit;
    end
  else
      einit;
  end
  if nargin<1
    filename=[ePath 'default.ppm'];
  end

  % read img-file
  imgFile=fopen(filename ,'r');
  line= fgetl(imgFile);
  if      strcmp(line,'P3'); bindata=0;
  elseif  strcmp(line,'P6'); bindata=1;
  else
   bindata=3;
  end
```

```
  if  bindata<3
    line= fgetl(imgFile);
    skip=1;
    while skip
      if length(line)==0;line= fgetl(imgFile);
      elseif line(1)=='#';line= fgetl(imgFile);
      else skip=0;
      end
    end
    imgSize= sscanf( line,'%d %d');
    maxValue= sscanf( fgetl(imgFile),'%d');
    if bindata; [data n]= fread(imgFile,imgSize(1)*imgSize(2)*3,'uchar');
    else        [data n]= fscanf(imgFile,'%d',imgSize(1)*imgSize(2)*3);
    end
    fclose(imgFile);

    if nargout==2
      % generate colormap
      data=reshape(data,3,size(data,1)/3)';
      id=data*[65536;256;1];
      [cmap index]=sort(id);
      change=diff(cmap);
      dIndex=[1;find(change)+1];
      colorId=cmap(dIndex);
      colormap=data(index(dIndex),:)/maxValue;

      % generate image
      dIndex=[dIndex;size(cmap,1)+1];
      for i=1:size(colorId,1)
        data(index(dIndex(i):dIndex(i+1)-1),1)=i;
      end
      image=reshape(data(:,1),imgSize(1),imgSize(2))';
    else
      % generate image without colormap
      data=reshape(data,3,size(data,1)/3)*255/maxValue;
      data=data(1,:)*65536+data(2,:)*256+data(3,:);
      image=reshape(data,imgSize(1),imgSize(2))';
    end
  else
    disp(['error in eppmread:' filename ' does not appear to be a ppm-file']);
  end
```

1.2.43 eppmwrit.m

```
%%NAME
%%  eppmwrit  - save image as PPM-file
%%
%%SYNOPSIS
%%  eppmwrit(filename,image,colormap[,binary])
%%
%%PARAMETER(S)
%%  filename    name of PPM-file
%%  image       matrix for image
```

```
%%              if colormap=-1 then
%%                image is filled with RGB values
%%                value=R*2^16+G*2^8+B) and R,G,B are integer of 0:255
%%              else
%%                matrix is filled with indices of colormap
%%  colormap    color table
%%  binary      default: binary=1 for binary PPM-file
%%              binary=0 for ascii PPM-file
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
function eppmwrit (filename,image,colormap,binary)
  if (nargin<3) | (nargin>4)
    eusage('eppmwrit(filename,image,colormap[,binary])');
  end
% image matrix and colormap -> rgb-map
  if nargin<4
    binary=1;
  end
  maxValue=255;
  imgSize=size(image);
  if colormap(1,1)<0
     image=reshape(image',1,imgSize(1)*imgSize(2));
     rImg=fix(image/65536);
     image=image-rImg*65536;
     gImg=fix(image/256);
     bImg=image-gImg*256;
     data=[rImg;gImg;bImg];
     data=reshape(data,imgSize(1)*imgSize(2)*3,1);
  else
    colormap=colormap'*maxValue;
    image=reshape(image',1,imgSize(1)*imgSize(2));
    data=colormap(:,image);
    data=reshape(data,1,imgSize(1)*imgSize(2)*3);
  end

% write ppm-file
  ppmFile=fopen(filename ,'w');
  if binary
    fprintf(ppmFile,'P6\n');
  else
    fprintf(ppmFile,'P3\n');
  end
  fprintf(ppmFile,...
    '# Image generated %s by epsTk 2.0 stefan.mueller@fgan.de\n',date);
  fprintf(ppmFile,'%d %d\n',imgSize(2),imgSize(1));
  fprintf(ppmFile,'%d\n',maxValue);

  if binary
    n=fwrite(ppmFile,data,'uchar');
  else
    n=fprintf(ppmFile,'%d ',data);
  end
  fclose(ppmFile);
```

1.2.44 equiver.m

```
%%NAME
%%  equiver - draw a quiver plot of matrix
%%
%%SYNOPSIS
%%  equiver(xData,yData,dx,dy[,color[,symbolName]])
%%
%%PARAMETER(S)
%%  xData           vector or matrix of x-positions of the symbols
%%  yData           vector or matrix of y-positions of the symbols
%%  dx              vector or matrix of x-values to determine
%%                  the direction and relative magnitude of the symbols
%%  dy              vector or matrix of y-values to determine
%%                  the direction and relative magnitude of the symbols
%%  color           color of symbols, vector [r g b]
%%  symbolName      symbol name of edsymbol() function
%%                  default symbol is an arrow
%%
%%GLOBAL PARAMETER(S)
%%  ePlotAreaPos
%%  ePlotAreaWidth
%%  ePlotAreaHeight
%%  eXAxisSouthScale
%%  eYAxisWestScale
%%  ePlotAreaXValueStart
%%  ePlotAreaXValueEnd
%%  ePlotAreaYValueStart
%%  ePlotAreaYValueEnd
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function equiver(xData,yData,dx,dy,color,symbolName)
  if nargin>6 | nargin<4
    eusage('equiver(xData,yData,dx,dy[,color[,symbolName]])');
  end
  eglobpar;
  if nargin<6
    symbolName='vector';
    edsymbol(symbolName,'farrow.psd');
  end
  if nargin<5
    color=[0 0 0];
  end
  [rows cols]=size(xData);
  if cols>1
    xData=reshape(xData,1,rows*cols);
    yData=reshape(yData,1,rows*cols);
    dx=reshape(dx,1,rows*cols);
    dy=reshape(dy,1,rows*cols);
  end

  dx2=dx.*dx;
  dy2=dy.*dy;
  angle=dx;
```

```
  vDiff=dy2-dx2;
  xgy=find(vDiff<0);
  angle(xgy)=atan(dy(xgy)./dx(xgy))*180/pi;
  sCase=find(dx(xgy)<0);
  angle(xgy(sCase))=angle(xgy(sCase))+180;
  ygx=find(vDiff>=0);
  angle(ygx)=90-atan(dx(ygx)./dy(ygx))*180/pi;
  sCase=find(dy(ygx)<0);
  angle(ygx(sCase))=angle(ygx(sCase))+180;
  vectorLength=sqrt(dx2+dy2);
  maxL=max(vectorLength);
  n=length(xData);
  deltaX=xData(1:n-1)-xData(2:n);
  deltaX=deltaX(find(deltaX));
  deltaX=deltaX.*deltaX;
  deltaX=sqrt(min(deltaX));
  deltaY=yData(1:n-1)-yData(2:n);
  deltaY=deltaY(find(deltaY));
  deltaY=deltaY.*deltaY;
  deltaY=sqrt(min(deltaY));
  deltaMin=min([deltaX deltaY]);
  if maxL>deltaMin
    vectorLength=vectorLength/maxL*deltaMin;
    maxL=deltaMin;
  end


 %value range
  if eXAxisSouthScale(1)==eXAxisSouthScale(3)
    ePlotAreaXValueStart=min(xData);
    ePlotAreaXValueEnd=max(xData);
    ePlotAreaXValueStart=ePlotAreaXValueStart-maxL;
    ePlotAreaXValueEnd=ePlotAreaXValueEnd+maxL;
  else
    %fix scale
    ePlotAreaXValueStart=eXAxisSouthScale(1);
    ePlotAreaXValueEnd=eXAxisSouthScale(3);
  end
  if eYAxisWestScale(1)==eYAxisWestScale(3)
    ePlotAreaYValueStart=min(yData);
    ePlotAreaYValueEnd=max(yData);
    ePlotAreaYValueStart=ePlotAreaYValueStart-maxL;
    ePlotAreaYValueEnd=ePlotAreaYValueEnd+maxL;
  else
    %fix scale
    ePlotAreaYValueStart=eYAxisWestScale(1);
    ePlotAreaYValueEnd=eYAxisWestScale(3);
  end
  ePlotAreaXFac=ePlotAreaWidth*eFac/...
    (ePlotAreaXValueEnd-ePlotAreaXValueStart);
  ePlotAreaYFac=ePlotAreaHeight*eFac/...
    (ePlotAreaYValueEnd-ePlotAreaYValueStart);
  xData=(xData-ePlotAreaXValueStart)*ePlotAreaXFac;
  yData=(yData-ePlotAreaYValueStart)*ePlotAreaYFac;
```

```
  exyplots(eFile,...
          ePlotAreaPos(1)*eFac,...
          ePlotAreaPos(2)*eFac,...
          xData(1,:),...
          yData(1,:),...
          vectorLength*ePlotAreaXFac/28.35,...
          vectorLength*ePlotAreaYFac/28.35,...
          angle,...
          symbolName,...
          color);

  egrid;
  eaxes;
```

1.2.45 ergb2idx.m

```
%%NAME
%%  ergb2idx  - covert RGB-matrix to index-matrix
%%
%%SYNOPSIS
%%  [image,colormap]=ergb2idx(matrix)
%%
%%PARAMETER(S)
%%  matrix      RBG-matrix
%%  image       index-matrix
%%  colormap    color table
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
function [image,colormap]= ergb2idx (matrix)
  if (nargin ~= 1)
    eusage('[image,colormap] = ergb2idx(matrix)');
  end

  [rows cols]= size(matrix);
  matrix=reshape(matrix,rows*cols,1);

  % generate colormap
  [cmap index]=sort(matrix);
  change=diff(cmap);
  dIndex=[1;find(change)+1];
  colorId=cmap(dIndex);
  colormap=matrix(index(dIndex));
  rColor=fix(colormap/65536);
  colormap=colormap-rColor*65536;
  gColor=fix(colormap/256);
  bColor=colormap-gColor*256;
  colormap=[rColor gColor bColor]/255;

  % generate image
  dIndex=[dIndex;size(cmap,1)+1];
  for i=1:size(colorId,1)
    matrix(index(dIndex(i):dIndex(i+1)-1))=i;
```

```
  end
  image=reshape(matrix,rows,cols);
```

1.2.46 eshadoi.m

```
%%NAME
%%  eshadoi  - draw shadow image of a matrix
%%
%%SYNOPSIS
%%  [x,colorMapNew]=eshadoi(matrix[,colorMap])
%%
%%PARAMETER(S)
%%  matrix             matrix for image
%%                     each value of the matrix is a row index of the colormap
%%  colorMap           define own colormap
%%
%%  if the next return parameters are used then no output
%%  x                  shadow image matrix
%%  colorMapNew        colormap of x
%%
%%GLOBAL PARAMETER(S)
%%  eImageDefaultColorMap
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function [x,colorMapNew]=eshadoi(matrix,colorMap)
  if nargin >2
    eusage('[x,colorMapNew]=eshadoi(matrix[,coloMap])');
  end
  eglobpar;
  if nargin==0
    x=-3*pi:0.1:3*pi;
    [a b]=meshgrid(x,x);
    R=sqrt(a.^2+b.^2) + eps;
    matrix=sin(R)./R;
  end
  if nargin<2
    colorMap=ecolors(eImageDefaultColorMap);
  end
  if sum(2*colorMap(:,1)-colorMap(:,2)-colorMap(:,3))
    nColors=size(colorMap,1);
    nBrL=64;
    minFac=0.4;
    colorMapNew=colorMap*(minFac+(1-minFac)*1/nBrL);
    for i=2:nBrL
      colorMapNew=[colorMapNew;colorMap*(minFac+(1-minFac)*i/nBrL)];
    end
  else
    nColors=1;
    colorMapNew=colorMap;
  end
  x=eshadow(matrix,nColors,colorMapNew,[1 1 1]);
  if nargout==0
    eimage(x,colorMapNew);
```

```
  end
```

1.2.47 eshadois.m

```
%%NAME
%%  eshadois - draw scaled shadow image of a matrix
%%
%%SYNOPSIS
%%  [x colorMapNew]=eshadois(matrix[,colorMap[,legendOrientation[,legendScale]]])
%%
%%PARAMETER(S)
%%  matrix             matrix for image
%%  colorMap           define own colormap
%%  legendOrientation  side of the image where the legend appears
%%                     character 's'(south),'n'(north),'w'(west) or 'e'(east)
%%                     (default orientation is east)
%%  legendScale        scale vector of legend [start step end]
%%                     special cases of scale vector are:
%%                     if start=0 and end=0 then autorange=on
%%                     if step=0 then autoscale=on
%%                     (default scale vector=[0 0 0])
%%
%%  if the next return parameters are used then no output
%%  x                  scaled shadow image matrix
%%  colorMapNew        colormap of x
%%
%%GLOBAL PARAMETER(S)
%%  ePlotAreaPos
%%  ePlotAreaWidth
%%  ePlotAreaHeight
%%  eImageDefaultColorMap
%%  eImageLegendScale
%%  eYAxisWestScale
%%  eXAxisSouthScale
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function [x,colorMapNew]=eshadois(matrix,colorMap,legendOrientation,legendScale)
  if nargin >4
    eusage('[x colorMapNew]=eshadois(matrix[,coloMap[,legendOrientation[,legendScale]]])');
  end
  eglobpar;
  if nargin<4
    legendScale=eImageLegendScale;
  end
  if nargin<3
    legendOrientation='e';
  end
  if nargin<2
    colorMap=ecolors(eImageDefaultColorMap);
  end
  if nargin==0
    x=-3*pi:0.1:3*pi;
    [a b]=meshgrid(x,x);
```

```
    R=sqrt(a.^2+b.^2) + eps;
    matrix=sin(R)./R;
    colorMap=[1 1 0;1 0 0;0 1 0;0 0 1;1 0 1];
  end

  if legendScale(1)==legendScale(3)
    minval=min (min (matrix));
    maxval=max (max (matrix));
  else
    minval=legendScale(1);
    maxval=legendScale(3);
  end
  if eImageLegendScaleType==2
    minval=log10(minval);
    maxval=log10(maxval);
    matrix=log10(matrix);
  end

  legendScale(1)=minval;
  legendScale(3)=maxval;
  [x colorMapNew]=eshadoi(matrix,colorMap);

  if nargout==0 & sum(2*colorMap(:,1)-colorMap(:,2)-colorMap(:,3))
    %image
    eimage(x,colorMapNew);

    % image axes
    if eYAxisWestScale(1)==eYAxisWestScale(3)
      eYAxisWestScale=[0 0 size(matrix,1)];
    end
    if eXAxisSouthScale(1)==eXAxisSouthScale(3)
      eXAxisSouthScale=[0 0 size(matrix,2)];
    end
    egrid;
    eaxes;

    % color legend
    eimgleg(ePlotAreaPos(1),ePlotAreaPos(2),ePlotAreaWidth,ePlotAreaHeight,...
      colorMap,legendScale,legendOrientation);
  end
```

1.2.48 eshadoix.m

```
%%NAME
%%  eshadoix  - mix a shadow image with a cover image
%%
%%SYNOPSIS
%%  [x colorMapNew]=eshadoix(matrix,coverImg,colorMap)
%%
%%PARAMETER(S)
%%  matrix             matrix to calculate the shadow image
%%  coverImg           matrix for cover image
%%                     each value of this matrix is a row index of the colormap
```

```
%%  colorMap            colormap of coverImg
%%
%%  if the next return parameters are used then no output
%%  x                   mix shadow image matrix
%%  colorMapNew         colormap of x
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function [x,colorMapNew]=eshadoix(matrix,coverImg,colorMap)
  if nargin ~= 3
    eusage('[x colorMapNew]=eshadoix(matrix,coverImg,coloMap)');
  end
  eglobpar;
  if sum(2*colorMap(:,1)-colorMap(:,2)-colorMap(:,3))
    nColors=size(colorMap,1);
    nBrL=64;
    minFac=0.4;
    colorMapNew=colorMap*(minFac+(1-minFac)*1/nBrL);
    for i=2:nBrL
      colorMapNew=[colorMapNew;colorMap*(minFac+(1-minFac)*i/nBrL)];
    end
  else
    nColors=1;
    colorMapNew=colorMap;
  end
  x=eshadow(matrix,nColors,colorMapNew,[1 1 1],coverImg);
  if nargout==0
    eimage(x,colorMapNew);
  end
```

1.2.49 eshadow.m

```
%%NAME
%%  eshadow  - make shadow image matrix
%%
%%SYNOPSIS
%%  x=eshadow(matrix,nColors,colorMap,lumen,image)
%%
%%PARAMETER(S)
%%  matrix              matrix for image
%%  nColors             number of colors
%%  colorMap            (nColors*nBrightnessLevels) x 3 Matrix
%%  lumen               light direction, [x,y,z] vector
%%  image               cover image
%%  x                   shadow image matrix
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function x=eshadow(matrix,nColors,colorMap,lumen,image)
  if nargin == 5
    x=image;
  elseif nargin == 4
```

```
    x=matrix;
  else
    eusage('x=eshadow(matrix,nColors,colorMap,lumen[,image])');
  end
  nMapItems=size(colorMap,1);
  nBrightnessLevels=nMapItems/nColors;

  darkAngle=-0.5;
  brightnessConst=nBrightnessLevels/(1-darkAngle)*1.001;

  [rows colums]=size(matrix);
  neightbourE=[matrix(:,2:colums) matrix(:,colums)];
  neightbourS=[matrix(2:rows,:);matrix(rows,:)];
  neightbourE=neightbourE-matrix;
  neightbourS=matrix-neightbourS;
  square=neightbourE.*neightbourE+neightbourS.*neightbourS;
  sumAmount= sum(sum(sqrt(square)));
  averageAmount=sumAmount/rows/colums;
  neightbourE=neightbourE/averageAmount;
  neightbourS=neightbourS/averageAmount;
  square=neightbourE.*neightbourE+neightbourS.*neightbourS;
  lumen=lumen/norm(lumen)*sqrt(2);
  brightness=fix(brightnessConst*...
    ((-neightbourE*lumen(1)+(-neightbourS)*lumen(2)+lumen(3))./...
    sqrt(2*(square+1))-darkAngle));
  brightness=reshape(brightness,rows*colums,1);
  search=find(brightness<1);
  if length(search)>0
    brightness(search)=ones(length(search),1);
  end
  brightness=reshape(brightness,rows,colums);

  maxValue=max(max(x));
  minValue=min(min(x));
  if (maxValue==minValue)
    x=ones(size(x,1),size(x,2));
  else
    x=fix((x-minValue)/(maxValue-minValue)*(nColors-1))+1;
  end
  x=x+(brightness-1)*nColors;
  x=x(1:rows-1,1:colums-1);
```

1.2.50 esubeps.m

```
%%NAME
%%  esubeps  -  insert eps-file in a subarea of the window
%%
%%SYNOPSIS
%%  esubeps(nRows,nColumns,row,column,epsFileName)
%%
%%PARAMETER(S)
%%  nRows         number of rows of the window
%%  nColumns      number of columns of the windows
```

```
%%  row           index of row
%%  column        index of column
%%  epsFileName   name of eps-file
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function esubeps(nRows,nColumns,row,column,epsFileName)
  if nargin~=5
    eusage('esubeps(nRows,nColumns,row,column,epsFileName)');
  end
  eglobpar;
  tWidth=eWinWidth/nColumns;
  tHeight=eWinHeight/nRows;

  % read box
  headsize=500;
  epsFile=fopen(epsFileName,'r');
  if epsFile>0
    head=fread(epsFile,headsize,'uchar');
    fclose(epsFile);
    head=setstr(head');
    pos=findstr(head,'BoundingBox:')+12;
    win=sscanf(head(pos(1):pos(1)+40),'%f',4);

    % resize fac
    boxWidth=(win(3)-win(1))/eFac;
    boxHeight=(win(4)-win(2))/eFac;
    rFacW=tWidth/boxWidth;
    rFacH=tHeight/boxHeight;
    if rFacW<rFacH
     rFac=rFacW;
    else
     rFac=rFacH;
    end
    xPos=(column-1)*tWidth;
    yPos=(nRows-row)*tHeight;
    einseps(xPos,yPos,epsFileName,rFac,rFac);
  end
```

1.2.51 esymbol.m

```
%%NAME
%%  esymbol  - draw a defined symbol
%%
%%SYNOPSIS
%%  esymbol(xPos,yPos,symbolName,[,scaleX[,scaleY[,rotation]]])
%%
%%PARAMETER(S)
%%  xPos          x position
%%  yPos          y position
%%  symbolName    name of defined symbol
%%  scaleX        scale factor in x-direction
%%  scaleY        scale factor in y-direction
%%  rotation      rotate symbol (deg)
```

```
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function esymbol(xPos,yPos,symbolName,scaleX,scaleY,rotation)
  if nargin<3 |nargin>6
    eusage('esymbol(xPos,yPos,symbolName,[,scaleX[,scaleY[,rotation]]])');
  end
  if nargin<6
    rotation=0;
  end
  if nargin<5
    scaleY=1;
  end
  if nargin<4
    scaleX=1;
  end
  eglobpar;
  fprintf(eFile,'gsave %1.2f %1.2f translate\n',xPos*eFac,yPos*eFac);
  fprintf(eFile,'%1.2f rotate\n',rotation);
  fprintf(eFile,'%1.2f %1.2f scale\n',scaleX,scaleY);
  fprintf(eFile,'%s grestore\n',symbolName);
```

1.2.52 etabdef.m

```
%%NAME
%%  etabdef - defines a table
%%
%%SYNOPSIS
%%  [colsXW rowsYH]=etabdef(rows,cols[,x,y[,width,height
%%                           [,colsWidth[,rowsHeight]]])
%%
%%PARAMETER(S)
%%  rows       number of rows
%%  cols       number of columns
%%  x          x-position (sw-corner) of table
%%  y          y-position (sw-corner) of table
%%  width      width of table
%%  height     height of table
%%  colsWidth  vector of relative width of columns ([1 1 1 ... 1]==equal widths)
%%  rowsHeight vector of rel. height of columns ([1 1 1 ... 1]==equal heights)
%%  colsXW     matrix of x-positions and width of columns,
%%             size of matrix is (number of cols) X 2
%%             examples:  colsXW(5,1) = x-position of colum 5
%%                        colsXW(5,2) = width of colum 5
%%  rowsYH     matrix of y-positions and heigth of rows,
%%             size of matrix is (number of rows) X 2
%%             examples:  cellXW(5,1) = y-position of row 5
%%                        cellXW(5,2) = height of row 5
%%
%%GLOBAL PARAMETER(S)
%%  eTabBackgroundColor
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function [colsXW,rowsYH]=etabdef(rows,cols,x,y,width,height,colsWidth,rowsHeight)
```

```
  if nargin>8 | nargin<2 | nargin==3 | nargin==5
    eusage('[colsXW rowsYH]=etabdef(rows,cols[,x,y[width,height[,colsWidth[,rowsHeight]]]])');
  end
  eglobpar;
  if nargin<8
    rowsHeight=ones(1,rows);
  end
  if nargin<7
    colsWidth=ones(1,cols);
  end
  if nargin<6
    width=ePlotAreaWidth;
    height=ePlotAreaHeight;
  end
  if nargin<4
    x=ePlotAreaPos(1);
    y=ePlotAreaPos(2);
  end

% columns size
  colsXW=zeros(cols,2);
  totalSize=sum(colsWidth);
  colsXW(:,2)=width*colsWidth'/totalSize;
  for i=1:cols
    colsXW(i,1)=x+sum(colsXW(1:i-1,2));
  end

% rows size
  rowsYH=zeros(rows,2);
  totalSize=sum(rowsHeight);
  rowsYH(:,2)=height*rowsHeight'/totalSize;
  for i=1:rows
    rowsYH(i,1)=y+height-sum(rowsYH(1:i,2));
  end

% background
  if eTabBackgroundColor(1)>=0
    erect(eFile,x*eFac,...
                y*eFac,...
                width*eFac,...
                height*eFac,...
                0,eTabBackgroundColor,0,0);
  end
```

1.2.53 etabgrid.m

```
%%NAME
%%  etabgrid - draw lines of table
%%
%%SYNOPSIS
%%  etabgrid(colsXW,rowsYH)
%%
%%PARAMETER(S)
```

```
%%  colsXW     matrix of x-positions and width of columns,
%%             size of matrix is (number of cols) X 2
%%             examples:  cellXW(5,1) = x-position of colum 5
%%                        cellXW(5,2) = width of colum 5
%%  rowsYH     matrix of y-positions and heigth of rows,
%%             size of matrix is (number of rows) X 2
%%             examples:  cellXW(5,1) = y-position of row 5
%%                        cellXW(5,2) = height of row 5
%%
%%GLOBAL PARAMETER(S)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function etabgrid(colsXW,rowsYH)
  if nargin~=2
    eusage('etabgrid(colsXW,rowsYH)');
  end
  eglobpar;
  cols=size(colsXW,1);
  rows=size(rowsYH,1);
  width=sum(colsXW(:,2));
  height=sum(rowsYH(:,2));
  x=colsXW(1,1);
  y=rowsYH(rows,1);

% xLines
  if eTabXLineVisible
    xLineY=rowsYH(1:rows-1,1)';
    xLineY=[xLineY;xLineY];
    xLineY=reshape(xLineY,1,2*(rows-1));
    xLineX=[ones(1,rows-1)*colsXW(1,1);
            ones(1,rows-1)*(colsXW(cols,1)+colsXW(cols,2))];
    xLineX=reshape(xLineX,1,2*(rows-1));
    exyline(eFile,0,0,...
                  xLineX*eFac,...
                  xLineY*eFac,...
                  eTabXLineColor,...
                  eTabXLineDash*eFac,...
                  eTabXLineWidth*eFac);
  end
% yLines
  if eTabXLineVisible
    xLineX=colsXW(2:cols,1)';
    xLineX=[xLineX;xLineX];
    xLineX=reshape(xLineX,1,2*(cols-1));
    xLineY=[ones(1,cols-1)*(rowsYH(1,1)+rowsYH(1,2));
            ones(1,cols-1)*rowsYH(rows,1)];
    xLineY=reshape(xLineY,1,2*(cols-1));
    exyline(eFile,0,0,...
                  xLineX*eFac,...
                  xLineY*eFac,...
                  eTabXLineColor,...
                  eTabXLineDash*eFac,...
                  eTabXLineWidth*eFac);
  end
```

```
% frame
  if eTabFrameVisible
    erect(eFile,x*eFac,...
                y*eFac,...
                width*eFac,...
                height*eFac,...
                eTabFrameLineWidth*eFac,...
                eTabFrameColor,0,0);
  end
```

1.2.54 etabtext.m

```
%%NAME
%%  etabtext - fill cell of table with text
%%
%%SYNOPSIS
%%  etabtext(colsXW,rowsYH,row,col,text[,alignment
%%           [,font[,fontSize[,color[,bgColor]]]])
%%
%%PARAMETER(S)
%%  colsXW     matrix of x-positions and width of columns,
%%             size of matrix is (number of cols) X 2
%%             examples:  cellXW(5,1) = x-position of colum 5
%%                        cellXW(5,2) = width of colum 5
%%  rowsYH     matrix of y-positions and heigth of rows,
%%             size of matrix is (number of rows) X 2
%%             examples:  cellXW(5,1) = y-position of row 5
%%                        cellXW(5,2) = height of row 5
%%  row        number of row
%%  col        number of column
%%  text       text of cell
%%  alignment  1=right 0=center -1=left
%%  font       font number (definition in einit.m)
%%  fontSize   relative fontsize in percent (100=default)
%%  color      color of text
%%  bgColor    color of background, [r g b] vector, if r<0 then transparent
%%
%%GLOBAL PARAMETER(S)
%% eTextColor
%% eTextFont
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function etabtext(colsXW,rowsYH,row,col,text,alignment,font,fontSize,color,bgColor)
  if nargin<5 | nargin>10
    eusage('etabtext(colsXW,rowsYH,row,col,text[,alignment[,font[,fontSize[,color[,bgColor]]]]])');
  end
  eglobpar;
  if nargin<10
    bgColor=[-1 0 0];
  end
  if nargin<9
    color=eTextColor;
  end
```

```
  if nargin<8
    fontSize=100;
  end
  if nargin<7
    font=eTextFont;
  end
  if nargin<6
    alignment=0;
  end
  relTextDistance=0.25;

% background
  if bgColor(1)>=0
    erect(eFile,colsXW(col,1)*eFac,...
                rowsYH(row,1)*eFac,...
                colsXW(col,2)*eFac,...
                rowsYH(row,2)*eFac,...
                0,bgColor,-1,0);
  end
% fontsize
  minCellHeight=min(rowsYH(:,2));
  fontSize=minCellHeight*(1-relTextDistance)*fontSize/100;
  textDistance=minCellHeight*relTextDistance;
% x,y
  if alignment==1
    x=colsXW(col,1)+textDistance;
    y=rowsYH(row,1)+textDistance;
  elseif alignment==0
    x=colsXW(col,1)+colsXW(col,2)/2;
    y=rowsYH(row,1)+textDistance;
  else
    x=colsXW(col,1)+colsXW(col,2)-textDistance;
    y=rowsYH(row,1)+textDistance;
  end
  etext(text,x,y,fontSize,alignment,font,0,color);
```

1.2.55 etext.m

```
%%NAME
%%  etext  -  write text
%%
%%SYNOPSIS
%%  etext(text[,x[,y[,fontSize[,alignment[,font[,rotation[,color]]]]]]])
%%
%%PARAMETER(S)
%%  text          text string
%%  x             x of start position
%%                if x=0 then the text starts after
%%                the last text in the same line
%%  y             y of start position
%%                if x=0 then y is a relativ position to the current line
%%  fontSize      scalar size of current font
%%                or vector [xSize ySize obliqueAngle(in deg)] of current font
```

```
%%  alignment     1=right 0=center -1=left from x-positon, y = line position
%%                2=right 3=center 4=left from x-positon, y = height of text/2
%%  font          font number (definition in einit.m)
%%  rotation      rotation of text (in deg)
%%  color         color of text, [r g b] vector
%%
%%GLOBAL PARAMETER(S)
%%  eTextColor
%%  eTextRotation
%%  eTextFont
%%  eTextAlignment
%%  eTextFontSize
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function etext(text,x,y,fontSize,alignment,font,rotation,color)
  if nargin<1 | nargin>8
    eusage('etext(text[,x[,y[,fontSize[,alignment[,font[,rotation[,color]]]]]]])');
  end
  eglobpar;
  if nargin<8
    color=eTextColor;
  end
  if nargin<7
    rotation=eTextRotation;
  end
  if nargin<6
    font=eTextFont;
  end
  if nargin<5
    alignment=eTextAlignment;
  end
  if nargin<4
    fontSize=eTextFontSize;
  end
  if nargin<3
    x=0;
    y=0;
  end
  if length(fontSize)==3
    fontSize=[fontSize(1:2)*eFac fontSize(3)];
  else
    fontSize=fontSize*eFac;
  end
  etextxy(eFile,...
          x*eFac,y*eFac,rotation,alignment,text,eFonts(font,:),...
          fontSize,color);
```

1.2.56 etxtbox.m

```
%%NAME
%%  etxtbox  -  write text in a box
%%
%%SYNOPSIS
```

```
%%  etxtbox(text[,x[,y[,boxWidth[,boxHeight[,fontSize[,alignment
%%           [,font[,rotation[,color[,offset]]]]]]]]])
%%
%%PARAMETER(S)
%%  text          text string
%%  x             x of start position
%%  y             y of start position
%%  boxWidth      width of textbox
%%                default=eWinWidth
%%  boxHeight     height of textbox
%%  fontSize      scalar size of current font
%%                or vector [xSize ySize obliqueAngle(in deg)] of current font
%%  alignment     1=right 0=center -1=left 2=block
%%  font          font number (definition in einit.m)
%%  rotation      rotation of box (in deg)
%%  color         color of text, [r g b] vector
%%  offset        offset vector [x y] of text, default offset=[0 0]
%%
%%GLOBAL PARAMETER(S)
%%  eTextColor
%%  eTextFont
%%  eTextAlignment
%%  eTextFontSize
%%  eTextLimitWord
%%  eTextLimitPara
%%  eTextBoxFeedLine
%%  eTextBoxFeedPara
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function etxtbox(text,x,y,boxWidth,boxHeight,fontSize,alignment,font,rotation,color,offset)
  if nargin<1 | nargin>11
    eusage('etxtbox(text[,x[,y[,boxWidth[,boxHeight[,fontSize[,alignment[,font[,rotation[,color[,off
  end
  eglobpar;
  if nargin<11
    offset=[0 0];
  end
  if nargin<10
    color=eTextColor;
  end
  if nargin<9
    rotation =0;
  end
  if nargin<8
    font=eTextFont;
  end
  if nargin<7
    alignment=eTextAlignment;
  end
  if nargin<6
    fontSize=eTextFontSize;
  end
  if nargin<5
    boxHeight=eWinHeight;
```

```
  end
  if nargin<4
    boxWidth=eWinWidth;
  end
  if nargin<3
    x=0;
    y=0;
  end
  if length(fontSize)==3
    lineFeed=fontSize(2);
    fontSize=[fontSize(1:2)*eFac fontSize(3)];
  else
    lineFeed=fontSize;
    fontSize=fontSize*eFac;
  end
  if eTextBoxFeedLine~=0
    lineFeed=eTextBoxFeedLine;
  end
  fprintf(eFile,'gsave\n');
  fprintf(eFile,'%1.2f %1.2f translate\n',x*eFac,y*eFac);
  fprintf(eFile,'%1.2f rotate\n',rotation);
  etxt2box(eFile,text,0,0,boxWidth*eFac,boxHeight*eFac,...
    lineFeed*eFac,eTextBoxFeedPara*eFac,...
    eTextLimitWord,eTextLimitPara,...
    alignment,eFonts(font,:),fontSize,color,...
    eTextBoxSpaceNorth*eFac,eTextBoxSpaceWest*eFac,...
    eTextBoxSpaceEast*eFac,eTextBoxSpaceSouth*eFac,offset*eFac);
  fprintf(eFile,'grestore\n');
```

1.2.57 etxtlpos.m

```
%%NAME
%%  etxtlpos  - get text line positions
%%
%%SYNOPSIS
%%  [linePos,nLines]=etxtlpos(text)
%%
%%PARAMETER(S)
%%  text        sting of text
%%  linePos     nLines x 2 Matrix of start and end positions
%%              [line1StartPos line1EndPos; line2StartPos line2EndPos ...
%%  nLines      number of lines
%%
% written by stefan.mueller stefan.mueller@fgan.de (C) 2003
function [linePos,nLines]=etxtlpos(text)
  if nargin~=1
    eusage('[linePos,nLines]=etxtlpos(text)');
  end
  eglobpar;
  tl=length(text);
  pos=findstr(text,eTextLimitPara);
  nLines=length(pos);
  if nLines>0
```

```
    posStart=[1 pos(1:nLines-1)+1];
    posEnd=pos-1;
    if pos(1)>1
      if text(pos(1)-1)==setstr(13)
        posEnd=pos-2;
      end
    end
    if pos(nLines)<tl
      posStart=[posStart pos(nLines)+1];
      posEnd=[posEnd tl];
      nLines=nLines+1;
    end
    linePos=[posStart' posEnd'];
  else
    nLines=1;
    linePos=[1 length(text)];
  end
```

1.2.58 etxtread.m

```
%%NAME
%%  etxtread  - read text-file
%%
%%SYNOPSIS
%%  [text,textLength]=etxtread(textFileName)
%%
%%PARAMETER(S)
%%  textFileName  name of textfile
%%  text        sting of text
%%
% written by stefan.mueller stefan.mueller@fgan.de (C) 2003
function [text,textLength]=etxtread(textFileName)
  if nargin>1
    eusage('[text,textLength]=etxtread(textFileName)');
  end
  eglobpar;
  textFile=fopen(textFileName,'r');
  [text textLength]=fread(textFile,inf,'uchar');
  text=setstr(text');
```

1.2.59 etxtwrit.m

```
%%NAME
%%  etxtwrit  - write string to text-file
%%
%%SYNOPSIS
%%  etxtwrit(text,textFileName)
%%
%%PARAMETER(S)
%%  text        sting of text
%%  textFileName  name of textfile
%%
```

```
% written by stefan.mueller stefan.mueller@fgan.de (C) 2003
function etxtwrit(text,textFileName)
  if nargin~=2
    eusage('etxtwrit(text,textFileName)');
  end
  textFile=fopen(textFileName,'w');
  fprintf(textFile,'%s',text);
  fclose(textFile);
```

1.2.60 eview.m

```
%%NAME
%%  eview  -  start ghostview to show eps-file
%%
%%SYNOPSIS
%%  eview([epsFileName])
%%
%%PARAMETER(S)
%%  epsFileName    name of eps-file
%%                 default: string of global parameter 'eFileName'
%%GLOBAL PARAMETER(S)
%%  eFileName
% written by stefan.mueller stefan.mueller@fgan.de (C) 2003
function eview(epsFileName)
  if nargin>1
    eusage('eview([epsFileName])');
  end
  eglobpar;
  if nargin<1
    epsFileName=eFileName;
  end
  if exist('eGhostview')~=1
    einit;
    eglobpar;
  end
  if isempty(eGhostview)
    disp('error in eview: no Postscript viewer installed');
  else
    gsview=sprintf('%s %s &',eGhostview,epsFileName);
    if exist('matlabpath')~=5
      system(gsview);
    else
      unix(gsview);
    end
  end
```

1.2.61 ewinsize.m

```
%%NAME
%%  ewinsize  -  get size of Bounding Box of eps-file
%%
%%SYNOPSIS
```

```
%%  [width,height]=ewinsize([epsFileName])
%%
%%PARAMETER(S)
%%  epsFileName   name of eps-file
%%                default: current eFileName
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function [width,height]=ewinsize(epsFileName)
  if nargin>1
    eusage('[width height]=ewinsize([epsFileName])');
  end
  eglobpar;
  if exist('eFac')
    if isempty(eFac)
      einit;
    end
  else
    einit;
  end
  if nargin<1
    epsFileName=eFileName;
  end

  % read eps file
  epsFile=fopen(epsFileName,'r');
  if epsFile<0
    errortext=sprintf('error in ewinsize: can not open %s',epsFileName);
    disp(errortext);
  else
    [data dl]=fread(epsFile,inf,'uchar');
    fclose(epsFile);
    headsize=500;
    head=setstr(data(1:headsize)');

    % read box
    pos=findstr(head,'BoundingBox:')+12;
    win=sscanf(head(pos(1):pos(1)+40),'%f',4);
    width=(win(3)-win(1))/eFac;
    height=(win(4)-win(2))/eFac;
  end
```

1.3 Low Level Functions

1.3.1 eclip.m

```
%eclip (epsFile,x,y,width,height)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eclip (epsFile,x,y,width,height)
  if nargin~=5
    eusage('eclip(epsFile,x,y,width,height)');
  end
```

```
  if width~=0
    fprintf(epsFile,'gsave newpath %1.2f %1.2f moveto\n',x,y);
    fprintf(epsFile,'%1.2f %1.2f lineto\n',x+width,y);
    fprintf(epsFile,'%1.2f %1.2f lineto\n',x+width,y+height);
    fprintf(epsFile,'%1.2f %1.2f lineto closepath clip\n',...
            x,y+height);
  else
    fprintf(epsFile,'grestore\n');
  end
```

1.3.2 eclippol.m

```
%eclippol (epsFile,x,y,minRadius,maxRadius,angleStart,angleEnd)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eclippol(epsFile,x,y,minRadius,maxRadius,angleStart,angleEnd)
  if nargin~=7 & nargin~=1
    eusage('eclippol(epsFile,x,y,minRadius,maxRadius,angleStart,angleEnd)');
  end
  if nargin~=1
    moveForm='%1.2f %1.2f 2 copy cos mul 3 1 roll sin mul moveto\n';
    arcForm='0 0 %1.2f %1.2f %1.2f arc\n';
    arcnForm='0 0 %1.2f %1.2f %1.2f arcn\n';
    fprintf(epsFile,'gsave %1.2f %1.2f 2 copy translate newpath\n',x,y);
    fprintf(epsFile,moveForm,minRadius,angleStart);
    fprintf(epsFile,arcForm,maxRadius,angleStart,angleEnd);
    fprintf(epsFile,arcnForm,minRadius,angleEnd,angleStart);
    fprintf(epsFile,'closepath clip neg exch neg exch translate\n');
  else
    fprintf(epsFile,'grestore\n');
  end
```

1.3.3 eellipxy.m

```
%eellipxy ( epsFile , x , y , width , height , lineWidth, color, dash, rotation)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eellipxy(epsFile,x,y,width,height,lineWidth,color,dash,rotation)
  if (nargin~=9)
    eusage('eellipxy(epsFile,x,y,width,height,lineWidth,color,dash,rotation)');
  end
  if max(size(dash))==1
    width=width-lineWidth;
    height=height-lineWidth;
    fprintf(epsFile,'gsave\n');
    if dash>0
      fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
    end
    fprintf(epsFile,'%1.2f %1.2f %1.2f setrgbcolor\n',...
                    color(1),color(2),color(3));
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    fprintf(epsFile,'%1.2f rotate\n',rotation);
```

```
    fprintf(epsFile,'1 %1.2f %1.2f div scale\n',height,width);
    fprintf(epsFile,'newpath 0 0 %1.2f 0 360 arc closepath\n',width/2);
    fprintf(epsFile,'1 %1.2f %1.2f div scale\n',width,height);
    fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
    if dash<0
      fprintf(epsFile,'fill\n');
    else
      fprintf(epsFile,'stroke\n');
    end
    fprintf(epsFile,'grestore\n');
  else
    width=width-2*lineWidth;
    height=height-2*lineWidth;
    fprintf(epsFile,'gsave\n');
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    fprintf(epsFile,'%1.2f rotate\n',rotation);
    fprintf(epsFile,'1 %1.2f %1.2f div scale\n',height,width);
    fprintf(epsFile,'newpath 0 0 %1.2f 0 360 arc closepath clip\n',width/2);
    fprintf(epsFile,'1 %1.2f %1.2f div scale\n',width,height);
    eimagexy(epsFile,dash,color,-width/2,-height/2,...
      width,height);
    fprintf(epsFile,'grestore\n');
  end
```

1.3.4 egridcl.m

```
%egridcl(epsFile,side,x,y,length,valueStart,valueStep,valueEnd,nMax,
%        lineLength,lineWidth,color,dash)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function egridcl(epsFile,side,x,y,length,valueStart,valueStep,valueEnd,...
                 nMax,lineLength,lineWidth,color,dash)
  if (nargin~=13)
    eusage(...
   'egridcl(epsFile,side,x,y,length,valueStart,valueStep,valueEnd,nMax,lineLength,lineWidth,color,da
  end

  startEndDiff=valueEnd-valueStart;
  signOfDiff=sign(startEndDiff);
  if valueStep==0
    valueStep=signOfDiff*eticdis(signOfDiff*startEndDiff,nMax);
  end
  valueStart=valueStart+valueStep;
  startEndDiff=valueEnd-valueStart;
  classValue=valueStart:valueStep:valueEnd;
  classStart=classValue-valueStep/2;
  classStart=(classStart-classStart(1))/...
    (startEndDiff+valueStep);
  classEnd=classStart+classStart(2);

  if side=='s'
    startPos=x;
    xLength=length;
```

```
    yLength=0;
    moveForm=sprintf('%%1.2f %1.2f moveto\n',y);
    ticLineForm='0 %1.2f rlineto\n';
  elseif side=='n'
    startPos=x;
    xLength=length;
    yLength=0;
    moveForm=sprintf('%%1.2f %1.2f moveto\n',y);
    ticLineForm='0 -%1.2f rlineto\n';
  elseif side=='w'
    startPos=y;
    xLength=0;
    yLength=length;
    moveForm=sprintf('%1.2f %%1.2f moveto\n',x);
    ticLineForm='%1.2f 0 rlineto\n';
  elseif side=='e'
    startPos=y;
    xLength=0;
    yLength=length;
    moveForm=sprintf('%1.2f %%1.2f moveto\n',x);
    ticLineForm='-%1.2f 0 rlineto\n';
  end

  % start draw
  if dash >0
    fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
  end
  fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
          color(1),color(2),color(3));
  fprintf(epsFile,'newpath\n');

  %lines
  nClasses=size(classStart,2);
  for i=1:nClasses
    % grid line
    currentPos=startPos+length*classStart(i);
    fprintf(epsFile,moveForm,currentPos);
    fprintf(epsFile,ticLineForm,lineLength);
    currentPos=startPos+length*classEnd(i);
    fprintf(epsFile,moveForm,currentPos);
    fprintf(epsFile,ticLineForm,lineLength);
  end

  fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
  fprintf(epsFile,'stroke\n');
  fprintf(epsFile,'setrgbcolor [] 0 setdash\n');
```

1.3.5 egridlog.m

```
%egridlog(epsFile,side,x,y,length,startValue,step,endValue,
%        maxValues,lineLength,lineWidth,color,dash)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
```

```
function egridlog(epsFile,side,x,y,length,startValue,step,endValue,...
                  maxValues,lineLength,lineWidth,color,dash)
  if (nargin~=13)
    eusage(...
   'egridlog(epsFile,side,x,y,length,startValue,step,endValue,maxValues,lineLength,lineWidth,color,d
  end

  if side=='s'
    startPos=x;
    xLength=length;
    yLength=0;
    moveForm=sprintf('%%1.2f %1.2f moveto\n',y);
    ticLineForm='0 %1.2f rlineto\n';
  elseif side=='w'
    startPos=y;
    xLength=0;
    yLength=length;
    moveForm=sprintf('%1.2f %%1.2f moveto\n',x);
    ticLineForm='%1.2f 0 rlineto\n';
  end
  startEndDiff=endValue-startValue;
  signOfDelta=sign(startEndDiff);
  if (step==0)
    deltaLabel=eticdis(signOfDelta*startEndDiff,maxValues);
  else
    deltaLabel=abs(step);
  end
  if deltaLabel>1
    startValue=endValue;
  end
  if deltaLabel<1
    deltaLabel=1;
  end


  %start ticNo and offset
  nShortTics=rem(startValue,deltaLabel)/deltaLabel*9*signOfDelta;
  if nShortTics<0
    nShortTics=9+nShortTics;
  end
  if rem(nShortTics,1)>0
      i=fix(nShortTics)+1;
      ticOffset=(i-nShortTics)*signOfDelta*deltaLabel/9;
  else
    i=nShortTics;
    ticOffset=0;
  end
  firstTicValue=startValue+ticOffset;
  currentValue=firstTicValue;
  deltaTic=signOfDelta*deltaLabel/9;
  axisFac=length/startEndDiff;
  shortTicD=log10([2 3 4 5 6 7 8 9]);
  shortTicD=[0 shortTicD];
```

```
  % start draw
  if dash >0
    fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
  end
  fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
          color(1),color(2),color(3));
  fprintf(epsFile,'newpath\n');

  %lines
  currentTic=0;
  currentValue=firstTicValue;
  while signOfDelta*currentValue<=signOfDelta*endValue
    rest=rem(i,9);
    interValue=currentValue+deltaTic*(9*shortTicD(rest+1)-rest);
    % grid line
    if signOfDelta*interValue<=signOfDelta*endValue
      currentPos=startPos+axisFac*(interValue-startValue);
      fprintf(epsFile,moveForm,currentPos);
      fprintf(epsFile,ticLineForm,lineLength);
    end
    i=i+1;
    currentTic=currentTic+1;
    currentValue=firstTicValue+currentTic*deltaTic;
  end

  fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
  fprintf(epsFile,'stroke\n');
  fprintf(epsFile,'setrgbcolor [] 0 setdash\n');
```

1.3.6 egridpa.m

```
%egridpa(epsFile,x,y,minRadius,maxRadius,angleStart,angleEnd,startValue,step,
%         endValue,maxValues,lineWidth,color,dash)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function egridpa(epsFile,x,y,minRadius,maxRadius,angleStart,angleEnd,...
                 startValue,step,endValue,maxValues,lineWidth,color,dash)
  if (nargin~=14)
    eusage(...
   'egridpd(epsFile,x,y,minRadius,maxRadius,angleStart,angleEnd,startValue,step,endValue,maxValues,l
  end

  startEndDiff=endValue-startValue;
  if step==0
    %autoscale
    signOfDelta=sign(startEndDiff);
    deltaLabel=eticdis(signOfDelta*startEndDiff,maxValues);
  else
    %fixscale
    signOfDelta=sign(step);
    deltaLabel=signOfDelta*step;
  end
  if rem(deltaLabel,3)==0
```

```
    nTics=3;
  else
    nTics=5;
  end
  nShortTics=rem(startValue,deltaLabel)*nTics/deltaLabel;
  i=fix(nShortTics);
  ticOffset=(nShortTics-i)*deltaLabel/nTics;
  ticOffset=sign(ticOffset)*ticOffset;
  firstTicValue=startValue+signOfDelta*ticOffset;
  currentValue=firstTicValue;
  deltaTic=signOfDelta*deltaLabel/nTics;
  axisFac=(angleEnd-angleStart)/startEndDiff;

  startPos=angleStart;
  moveForm=sprintf('%1.2f %%1.2f 2 copy cos mul 3 1 roll sin mul moveto\n',...
                   minRadius);
  ticLineForm=sprintf(...
         '%1.2f %%1.2f 2 copy cos mul 3 1 roll sin mul rlineto\n',...
         maxRadius-minRadius);

  % start draw
  fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
  if dash>0
    fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
  end
  fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
          color(1),color(2),color(3));
  fprintf(epsFile,'newpath\n');

  %lines
  currentTic=0;
  currentValue=firstTicValue;
  while signOfDelta*currentValue<=signOfDelta*endValue
    currentPos=startPos+axisFac*(currentValue-startValue);
    if rem(i,nTics)==0
      fprintf(epsFile,moveForm,currentPos);
      fprintf(epsFile,ticLineForm,currentPos);
      % value
    end
    i=i+1;
    currentTic=currentTic+1;
    currentValue=firstTicValue+currentTic*deltaTic;
  end
  fprintf(epsFile,'stroke\n');
  fprintf(epsFile,'setrgbcolor [] 0 setdash\n');
  fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
```

1.3.7 egridpol.m

```
%%NAME
%%  egridpol - draw polar grid
%%
%%SYNOPSIS
```

```
%%  egridpol ([axisRadiusScale,axisAngleScale])
%%
%%PARAMETER(S)
%%  axisRadiusScale   scale vector of radius axis  [start step end]
%%  axisAngleScale    scale vector of angle circle [start step end]
%%
%%    special cases of scale vectors are:
%%      if start=0 and end=0 then autorange=on
%%      if step=0 then autoscale=on
%%
%%GLOBAL PARAMETER(S)
%%  ePolarPlotAreaValStart
%%  ePolarPlotAreaValEnd
%%  ePolarPlotAreaCenterPos
%%  ePolarPlotAreaRadMin
%%  ePolarPlotAreaRadMax
%%  ePolarPlotAreaAngStart
%%  ePolarPlotAreaAngEnd
%%  ePolarAxisRadScale
%%  ePolarAxisAngScale
%%  ePolarRadiusGridVisible
%%  ePolarRadiusGridLineWidth
%%  ePolarRadiusGridColor
%%  ePolarRadiusGridDash
%%  ePolarAngleGridVisible
%%  ePolarAngleGridLineWidth
%%  ePolarAngleGridColor
%%  ePolarAngleGridDash
%%  eAxesTicLongMaxN
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function egridpol(axisRadiusScale,axisAngleScale)
  if nargin~=0 & nargin~=2
    eusage('egridpol([axisRadiusScale,axisAngleScale])');
  end
  eglobpar;
  if nargin==2
    ePolarAxisRadScale=axisRadiusScale;
    ePolarPlotAreaValStart=ePolarAxisRadScale(1);
    ePolarPlotAreaValEnd=ePolarAxisRadScale(3);
    ePolarAxisAngScale=axisAngleScale;
    yRange=ePolarPlotAreaValEnd-ePolarPlotAreaValStart;
    if yRange==0
      yRange=1;
    end
    ePolarPlotAreaFac=(ePolarPlotAreaRadMax-ePolarPlotAreaRadMin)*...
                      eFac/yRange;
  end
  if nargin==0
    if ePolarAxisRadScale(1)~=ePolarAxisRadScale(3)
      ePolarPlotAreaValStart=ePolarAxisRadScale(1);
      ePolarPlotAreaValEnd=ePolarAxisRadScale(3);
    end
```

```
    % scale and draw grids
    if ePolarRadiusGridVisible
      egridpr(eFile,...
              ePolarPlotAreaCenterPos(1)*eFac,...
              ePolarPlotAreaCenterPos(2)*eFac,...
              ePolarPlotAreaRadMin*eFac,...
              ePolarPlotAreaRadMax*eFac,...
              ePolarPlotAreaAngStart,...
              ePolarPlotAreaAngEnd,...
              ePolarPlotAreaValStart,...
              ePolarAxisRadScale(2),...
              ePolarPlotAreaValEnd,...
              eAxesTicLongMaxN,...
              ePolarRadiusGridLineWidth*eFac,...
              ePolarRadiusGridColor,...
              ePolarRadiusGridDash*eFac);
    end
    if ePolarAngleGridVisible
      maxValues=fix(20*(ePolarPlotAreaAngEnd-ePolarPlotAreaAngStart)/360);
      egridpa(eFile,...
              ePolarPlotAreaCenterPos(1)*eFac,...
              ePolarPlotAreaCenterPos(2)*eFac,...
              ePolarPlotAreaRadMin*eFac,...
              ePolarPlotAreaRadMax*eFac,...
              ePolarPlotAreaAngStart,...
              ePolarPlotAreaAngEnd,...
              ePolarAxisAngScale(1),...
              ePolarAxisAngScale(2),...
              ePolarAxisAngScale(3),...
              maxValues,...
              ePolarAngleGridLineWidth*eFac,...
              ePolarAngleGridColor,...
              ePolarAngleGridDash*eFac);
    end
  end
```

1.3.8 egridpr.m

```
%egridpr(epsFile,x,y,minRadius,maxRadius,angleStart,angleEnd,...
%        startValue,step,endValue,maxValues,lineWidth,color,dash)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function egridpr(epsFile,x,y,minRadius,maxRadius,angleStart,angleEnd,...
                 startValue,step,endValue,maxValues,lineWidth,color,dash)
  if (nargin~=14)
    eusage(...
   'egridpr(epsFile,x,y,minRadius,maxRadius,angleStart,angleEnd,startValue,step,endValue,maxValues,l
  end

  %scale
  length=maxRadius-minRadius;
  startEndDiff=endValue-startValue;
  if step==0
```

```
    %autoscale
    signOfDelta=sign(startEndDiff);
    deltaLabel=eticdis(signOfDelta*startEndDiff,maxValues);
  else
    %fixscale
    signOfDelta=sign(step);
    deltaLabel=signOfDelta*step;
  end
  nShortTics=rem(startValue,deltaLabel)*5.0/deltaLabel;
  i=fix(nShortTics);
  ticOffset=(nShortTics-i)*deltaLabel/5;
  ticOffset=sign(ticOffset)*ticOffset;
  firstTicValue=startValue+signOfDelta*ticOffset;
  currentValue=firstTicValue;
  deltaTic=signOfDelta*deltaLabel/5;
  axisFac=length/startEndDiff;

  % start draw
  fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
  fprintf(epsFile,'%1.2f rotate\n',angleStart);
  if dash >0
    fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
  end
  fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
          color(1),color(2),color(3));
  fprintf(epsFile,'newpath\n');

  %lines
  startPos=minRadius;
  moveForm='0 %1.2f moveto\n';
  currentTic=0;
  currentValue=firstTicValue;
  while signOfDelta*currentValue<=signOfDelta*endValue
    currentPos=startPos+axisFac*(currentValue-startValue);
    if rem(i,5)==0
      % grid line
      fprintf(epsFile,'%1.2f 0 moveto\n',currentPos);
      fprintf(epsFile,'0 0 %1.2f 0 %1.2f arc\n',currentPos,angleEnd-angleStart);
    end
    i=i+1;
    currentTic=currentTic+1;
    currentValue=firstTicValue+currentTic*deltaTic;
  end

  fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
  fprintf(epsFile,'stroke\n');
  fprintf(epsFile,'setrgbcolor [] 0 setdash\n');
  fprintf(epsFile,'%1.2f rotate\n',-angleStart);
  fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
```

1.3.9 egridxy.m

```
%egridxy(epsFile,side,x,y,length,startValue,step,endValue,
```

```
%        maxValues,lineLength,lineWidth,color,dash)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function egridxy(epsFile,side,x,y,length,startValue,step,endValue,...
                 maxValues,lineLength,lineWidth,color,dash)
  if (nargin~=13)
    eusage(...
   'egridxy(epsFile,side,x,y,length,startValue,step,endValue,maxValues,lineLength,lineWidth,color,da
  end

  if side=='s'
    startPos=x;
    xLength=length;
    yLength=0;
    moveForm=sprintf('%%1.2f %1.2f moveto\n',y);
    ticLineForm='0 %1.2f rlineto\n';
  elseif side=='w'
    startPos=y;
    xLength=0;
    yLength=length;
    moveForm=sprintf('%1.2f %%1.2f moveto\n',x);
    ticLineForm='%1.2f 0 rlineto\n';
  end
  startEndDiff=endValue-startValue;
  if step==0
    %autoscale
    signOfDelta=sign(startEndDiff);
    deltaLabel=eticdis(signOfDelta*startEndDiff,maxValues);
  else
    %fixscale
    signOfDelta=sign(step);
    deltaLabel=signOfDelta*step;
  end
  %start ticNo and offset
  nShortTics=rem(startValue,deltaLabel)/deltaLabel*5*signOfDelta;
  if nShortTics<0
    nShortTics=5+nShortTics;
  end
  if rem(nShortTics,1)>0
      i=fix(nShortTics)+1;
      ticOffset=(i-nShortTics)*signOfDelta*deltaLabel/5;
  else
    i=nShortTics;
    ticOffset=0;
  end
  firstTicValue=startValue+ticOffset;
  currentValue=firstTicValue;
  deltaTic=signOfDelta*deltaLabel/5;
  axisFac=length/startEndDiff;

  % start draw
  if dash >0
    fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
  end
```

```
  fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
          color(1),color(2),color(3));
  fprintf(epsFile,'newpath\n');

  %lines
  currentTic=0;
  currentValue=firstTicValue;
  while signOfDelta*currentValue<=signOfDelta*endValue
    currentPos=startPos+axisFac*(currentValue-startValue);
    if rem(i,5)==0
      % grid line
      fprintf(epsFile,moveForm,currentPos);
      fprintf(epsFile,ticLineForm,lineLength);
    end
    i=i+1;
    currentTic=currentTic+1;
    currentValue=firstTicValue+currentTic*deltaTic;
  end

  fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
  fprintf(epsFile,'stroke\n');
  fprintf(epsFile,'setrgbcolor [] 0 setdash\n');
```

1.3.10 ehead.m

```
%ehead(epsFile,winWidth,winHeight,pageWidth,pageHeight,pageOrientation,xScaleFac,yScaleFac,xOffset,y
% this function write postscript commands in  epsFile to initialize
% the eps-output
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function ehead(epsFile,winWidth,winHeight,pageWidth,pageHeight,pageOrientation,xScaleFac,yScaleFac,x
  if (nargin~=11)
    eusage('ehead(epsFile,winWidth,winHeight,pageWidth,pageHeight,pageOrientation,xScaleFac,yScaleFa
  end

  % win size
  if pageOrientation==1 | pageOrientation==3
    winW=winWidth;
    winWidth=winHeight;
    winHeight=winW;
  end
  winWidth=winWidth*xScaleFac;
  winHeight=winHeight*yScaleFac;

  % max plot area
  printerFrame=17;
  pageHeight=pageHeight-2*printerFrame;
  pageWidth=pageWidth-2*printerFrame;

  % scale factor
  if winWidth/pageWidth>winHeight/pageHeight
    maxFac=pageWidth/winWidth;
  else
```

```
  maxFac=pageHeight/winHeight;
end
if maxFac<1
  disp('Graphic reduced !');
  winHeight=winHeight*maxFac;
  winWidth=winWidth*maxFac;
else
  maxFac=1;
end
xScaleFac=xScaleFac*maxFac;
yScaleFac=yScaleFac*maxFac;

% win offset
winX0=(pageWidth-winWidth)/2+printerFrame+xOffset;
winY0=(pageHeight-winHeight)/2+printerFrame+yOffset;

% new origin
if pageOrientation==0
  originX=winX0;
  originY=winY0;
  reflectShift=winWidth;
elseif pageOrientation==1
  originX=winY0;
  originY=-(winX0+winWidth);
  reflectShift=winHeight;
elseif pageOrientation==2
  originX=-(winX0+winWidth);
  originY=-(winY0+winHeight);
  reflectShift=winWidth;
else
  originX=-(winY0+winHeight);
  originY=winX0;
  reflectShift=winHeight;
end

% write eps head
fprintf(epsFile,'%%!PS-Adobe-2.0 EPSF-2.0\n');
fprintf(epsFile,'%%%%Creator: epsTk 2.0 stefan.mueller@fgan.de 2003\n');
timeStamp=clock;
min1=fix(timeStamp(5)/10);
min2=rem(timeStamp(5),10);
fprintf(epsFile,'%%%%Time: %d.%d.%d %d:%d%d:%d\n',...
               timeStamp(3),timeStamp(2),timeStamp(1),...
               timeStamp(4),min1,min2,timeStamp(6));
fprintf(epsFile,'%%%%BoundingBox: %d %d %d %d\n',...
        fix(winX0),fix(winY0),fix(winX0+winWidth),fix(winY0+winHeight));
fprintf(epsFile,'%%%%EndComments\n');
fprintf(epsFile,'%1.2f rotate\n',pageOrientation*90);
fprintf(epsFile,'%d %d translate\n',fix(originX),fix(originY));
if pageReflection==1
  fprintf(epsFile,'%1.2f 0 translate\n',reflectShift);
  fprintf(epsFile,'-1 1 scale\n');
end
fprintf(epsFile,'%1.2f %1.2f scale\n',xScaleFac,yScaleFac);
```

```
  fprintf(epsFile,'/GermanExtension[\n');
  fprintf(epsFile,'8#374 /udieresis\n');
  fprintf(epsFile,'8#334 /Udieresis\n');
  fprintf(epsFile,'8#344 /adieresis\n');
  fprintf(epsFile,'8#304 /Adieresis\n');
  fprintf(epsFile,'8#366 /odieresis\n');
  fprintf(epsFile,'8#326 /Odieresis\n');
  fprintf(epsFile,'8#337 /germandbls\n');
  fprintf(epsFile,']def\n');
  fprintf(epsFile,'/ReEncode \n');
  fprintf(epsFile,'/newFontName exch def\n');
  fprintf(epsFile,'/oldFontName exch def\n');
  fprintf(epsFile,'/basefontdict oldFontName findfont def\n');
  fprintf(epsFile,'/nFont basefontdict maxlength dict def\n');
  fprintf(epsFile,'basefontdictexch dup /FID ne dup /Encoding eq\n');
  fprintf(epsFile,'exch dup length array copy nFont 3 1 roll put\n');
  fprintf(epsFile,'exch nFont 3 1 roll putifelsepop pop ifelseforall\n');
  fprintf(epsFile,'nFont /FontName newFontName put\n');
  fprintf(epsFile,'GermanExtension aload pop GermanExtension length 2 idiv\n');
  fprintf(epsFile,'nFont /Encoding get 3 1 roll put repeat\n');
  fprintf(epsFile,'newFontName nFont definefont pop def\n');
```

1.3.11 eimagexy.m

```
% eimagexy(epsFile,image,colorMap,x,y,width,height)
% write postscript commands in  epsFile to create an image
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eimagexy(epsFile,image,colorMap,x,y,width,height)
  if (nargin~=7)
    eusage('eimagexy(epsFile,image,colorMap,x,y,width,height)');
  end
  if isstr(image)
    [image,head]=ejpgread(image);
    n=head(1);rows=head(2);cols=head(3);rgb=head(4);
    if rgb
      decodeText='/Decode [0 1 0 1 0 1]\n/DataSource currentfile\n';
    else
      decodeText='/Decode [0 1]\n/DataSource currentfile\n';
    end
    filterText='/ASCII85Decode filter\n/DCTDecode filter\n>>\nimage\n';
  elseif colorMap(1,1)>1
    n=colorMap(1);rows=colorMap(2);cols=colorMap(3);rgb=colorMap(4);
    if rgb
      decodeText='/Decode [0 1 0 1 0 1]\n/DataSource currentfile\n';
    else
      decodeText='/Decode [0 1]\n/DataSource currentfile\n';
    end
    filterText='/ASCII85Decode filter\n/DCTDecode filter\n>>\nimage\n';
  elseif colorMap(1,1)<=1
    [rows cols]=size(image);
    if colorMap(1,1)<0
      image=reshape(image',rows*cols,1);
```

```
      bImg=rem(image,256);
      image=fix(image/256);
      gImg=rem(image,256);
      rImg=fix(image/256);
      image=[rImg';gImg';bImg'];
    else
      colorMap=fix(colorMap*255);
      image=reshape(image',rows*cols,1);
      image=[colorMap(image,1)';colorMap(image,2)';colorMap(image,3)'];
    end
    n=rows*cols*3;
    image=reshape(image,n,1);
    decodeText='/Decode [0 1 0 1 0 1]\n/DataSource currentfile\n';
    filterText='/ASCII85Decode filter\n>>\nimage\n';
  end
  nTuples=ceil(n/4);
  nTail=nTuples*4-n;
  if nTail>0
    image=[image;zeros(nTail,1)];
  end
  image=[reshape(image,4,nTuples);zeros(1,nTuples)];
  tupSum=image(4,:)+image(3,:)*256+image(2,:)*65536+image(1,:)*16777216;
  image(5,:)=rem(tupSum,85);
  tupSum=fix(tupSum/85);
  image(4,:)=rem(tupSum,85);
  tupSum=fix(tupSum/85);
  image(3,:)=rem(tupSum,85);
  tupSum=fix(tupSum/85);
  image(2,:)=rem(tupSum,85);
  image(1,:)=fix(tupSum/85);
  image=image+33;
  image=reshape(image,5*nTuples,1);
  n=nTuples*5-nTail;

  fprintf(epsFile,'gsave\n');
  fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
  fprintf(epsFile,'%1.2f %1.2f scale\n',width,height);
  fprintf(epsFile,'/DeviceRGB setcolorspace\n<<\n');
  fprintf(epsFile,'/ImageType 1\n/Width %d /Height %d\n',cols,rows);
  fprintf(epsFile,'/BitsPerComponent 8\n/ImageMatrix ');
  fprintf(epsFile,'[%d 0 0 -%d 0 %d]\n',cols,rows,rows);
  decodefilter=[decodeText filterText];
  fprintf(epsFile,decodefilter);
  fwrite(epsFile,image(1:n),'uchar');
  fprintf(epsFile,'~>\ngrestore\n');
```

1.3.12 eimgleg.m

```
%%NAME
%%  eimgleg  - draw image legend of area
%%
%%SYNOPSIS
%%  eimgleg(x,y,width,height,map,scale,orientation)
```

```
%%
%%PARAMETER(S)
%%  x                   sw-x position of area
%%  y                   sw-y position of area
%%  width               width of area
%%  height              height of area
%%  map                 map of legend
%%  scale               scale vector of legend [start step end]
%%                      special cases of scale vector are:
%%                      if start=0 and end=0 then autorange=on
%%                      if step=0 then autoscale=on
%%  orientation         side of the area where the legend appears
%%                      character 's'(south),'n'(north),'w'(west) or 'e'(east)
%%
%%GLOBAL PARAMETER(S)
%%  eImageLegendScaleType
%%  eImageLegendVisible
%%  eImageLegendPos
%%  eImageLegendHeight
%%  eImageLegendWidth
%%  eImageLegendLabelDistance
%%  eImageLegendValueFormat
%%  eImageLegendLabelText
%%  eAxesColor
%%  eAxesTicLongLength
%%  eAxesTicShortLength
%%  eAxesTicLongMaxN
%%  eAxesValueSpace
%%  eAxesValueFontSize
%%  eAxesLabelFontSize
%%  eAxesLabelTextFont
%%  eAxesLineWidth
%%
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eimgleg(x,y,width,height,map,scale,orientation)
  if (nargin~=7)
    eusage('eimgleg(x,y,width,height,map,scale,orientation');
  end
  eglobpar;

  if eImageLegendVisible
    if eImageLegendWidth==0
      if orientation=='s' | ...
         orientation=='n'
        xl=width;
        yl=eImageLegendHeight;
        scaleLength=xl;
      else
        yl=height;
        xl=eImageLegendHeight;
        scaleLength=yl;
      end
    else
```

```
  if orientation=='s' | ...
     orientation=='n'
    xl=eImageLegendWidth;
    yl=eImageLegendHeight;
    scaleLength=xl;
  else
    xl=eImageLegendHeight;
    yl=eImageLegendWidth;
    scaleLength=yl;
  end
end
scaleSpace=eAxesTicLongLength+eAxesValueSpace+eAxesValueFontSize;
if orientation=='s'
 colorImage=(1:length(map));
 legendPosX=x+eImageLegendPos(1);
 legendPosY=y+eImageLegendPos(2);
 scalePosX=legendPosX;
 scalePosY=legendPosY;
 textPosX=scalePosX+scaleLength/2;
 textPosY=scalePosY-scaleSpace-eAxesLabelFontSize*0.72-...
          eImageLegendLabelDistance;
 textAngle=0;
elseif orientation=='n'
 colorImage=(1:length(map));
 legendPosX=x+eImageLegendPos(1);
 legendPosY=y+height-eImageLegendPos(2);
 scalePosX=legendPosX;
 scalePosY=legendPosY+yl;
 textPosX=scalePosX+scaleLength/2;
 textPosY=scalePosY+scaleSpace+yl+eAxesLabelFontSize*0.72+...
          eImageLegendLabelDistance;
 textAngle=0;
elseif orientation=='e'
 colorImage=(length(map):-1:1)';
 legendPosX=x+width-eImageLegendPos(2);
 legendPosY=y+eImageLegendPos(1);
 scalePosX=legendPosX+xl;
 scalePosY=legendPosY;
 textPosX=scalePosX+scaleSpace+xl+eAxesLabelFontSize*0.72+...
          eImageLegendLabelDistance;
 textPosY=scalePosY+scaleLength/2;
 textAngle=90;
elseif orientation=='w'
 colorImage=(length(map):-1:1)';
 legendPosX=x+eImageLegendPos(2)-xl;
 legendPosY=y-eImageLegendPos(1);
 scalePosX=legendPosX;
 scalePosY=legendPosY;
 textPosX=scalePosX-scaleSpace-eAxesLabelFontSize*0.72-...
          eImageLegendLabelDistance;
 textPosY=scalePosY+scaleLength/2;
 textAngle=90;
end
eimagexy(eFile,colorImage,map,...
```

```
          legendPosX*eFac,legendPosY*eFac,xl*eFac,yl*eFac);
  erect(eFile,legendPosX*eFac,legendPosY*eFac,...
        xl*eFac,yl*eFac,eAxesLineWidth*eFac,[0 0 0],0,0);
  if eImageLegendScaleType==0
    eImageLegendValuePos=escalexy(eFile,orientation,...
             scalePosX*eFac,scalePosY*eFac,0,0,...
             scaleLength*eFac,...
             scale(1),...
             scale(2),...
             scale(3),...
             eImageLegendValueFormat,...
             eImageLegendValueVisible,...
             eAxesValueFontSize*eFac,...
             eAxesLineWidth*eFac,...
             eAxesTicShortLength*eFac,...
             eAxesTicLongLength*eFac,...
             eAxesTicLongMaxN,...
             eAxesValueSpace*eFac,...
             eAxesColor);
    eImageLegendValuePos=eImageLegendValuePos/eFac;
  elseif eImageLegendScaleType==1
    eImageLegendValuePos=escalecl(eFile,orientation,...
             scalePosX*eFac,scalePosY*eFac,0,0,...
             scaleLength*eFac,...
             scale(1),...
             scale(2),...
             scale(3),...
             eImageLegendValueFormat,...
             eImageLegendValueVisible,...
             eAxesValueFontSize*eFac,...
             eAxesLineWidth*eFac,...
             eAxesTicLongLength*eFac,...
             eAxesTicLongMaxN,...
             eAxesValueSpace*eFac,...
             eAxesColor);
    eImageLegendValuePos=eImageLegendValuePos/eFac;
  elseif eImageLegendScaleType==2
    eImageLegendValuePos=escalelog(eFile,orientation,...
             scalePosX*eFac,scalePosY*eFac,0,0,...
             scaleLength*eFac,...
             scale(1),...
             scale(2),...
             scale(3),...
             eImageLegendValueFormat,...
             eImageLegendValueVisible,...
             eAxesValueFontSize*eFac,...
             eAxesLineWidth*eFac,...
             eAxesTicShortLength*eFac,...
             eAxesTicLongLength*eFac,...
             eAxesTicLongMaxN,...
             eAxesValueSpace*eFac,...
             eAxesColor);
    eImageLegendValuePos=eImageLegendValuePos/eFac;
  end
```

```
    if strcmp(eImageLegendLabelText,'')~=1
      etext(eImageLegendLabelText,...
            textPosX,...
            textPosY,...
            eAxesLabelFontSize,0,eAxesLabelTextFont,textAngle,eAxesColor);
    end
  end
```

1.3.13 ejpg2eps.m

```
%%NAME
%%  ejpg2eps  - convert JPEG-file to EPS-file
%%
%%SYNOPSIS
%%  dpi=ejpg2eps(jpgFileName[,epsFileName])
%%
%%PARAMETER(S)
%%  jpgFileName          JPEG-filenames
%%  epsFileName          EPS-filenames
%%
% written by stefan.mueller stefan.mueller@fgan.de (C) 2003
function dpi=ejpg2eps(jpgFileName,epsFileName)
  if nargin>2
    eusage('dpi=ejpg2eps(jpgFileName[,epsFileName])');
  end
  eglobpar;
  if exist('eFac')
    if isempty(eFac)
      einit;
    end
  else
    einit;
  end
  if nargin<1
    jpgFileName=[ePath 'default.jpg'];
  end
  if nargin<2
    epsFileName=jpgFileName;
    suffixPos=findstr(epsFileName,'.jpg');
    epsFileName(suffixPos(1):suffixPos(1)+3)='.eps';
  end
 [image head]=ejpgread(jpgFileName);
 jpgH=head(2);
 jpgW=head(3);
 winFac=eWinHeight/eWinWidth;
 imgFac=jpgH/jpgW;
 if winFac<imgFac
   eWinWidth=eWinHeight/imgFac;
   dpi=jpgH*72/eWinHeight/eFac;
 else
   eWinHeight=eWinWidth*imgFac;
   dpi=jpgW*72/eWinWidth/eFac;
 end
```

```
offsetX=1.2*eWinWidth/jpgW;
offsetY=1.2*eWinHeight/jpgH;
eopen(epsFileName,0,eWinWidth,eWinHeight)
eframe(0,0,eWinWidth+offsetX,eWinHeight+offsetY,0,image,head);
eclose(1,0);
```

1.3.14 epiesxy.m

```
%epiesxy (epsFile,x,y,minRadius,maxRadius,angleStart,sliceSize,color,dash,lineWidth,offset)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function epiesxy(epsFile,x,y,minRadius,maxRadius,angleStart,sliceSize,color,dash,lineWidth,offset)
  if nargin~=11
    eusage('epiesxy(epsFile,x,y,minRadius,maxRadius,angleStart,sliceSize,color,dash,lineWidth)');
  end
  angleEnd=angleStart+sliceSize;
  fprintf(epsFile,'/W2 %1.2f 2 div def\n',lineWidth);
  fprintf(epsFile,'/Rmin %1.2f W2 add def\n',minRadius);
  fprintf(epsFile,'/Rmax %1.2f W2 sub def\n',maxRadius);
  fprintf(epsFile,'Rmin Rmax gt /Rmax Rmin defif\n');
  fprintf(epsFile,'gsave %1.2f %1.2f translate\n',x,y);
  if offset>0
    fprintf(epsFile,...
      '%1.2f %1.2f 2 copy cos mul 3 1 roll sin mul translate\n',...
      offset,angleStart+sliceSize/2);
  end
  if max(size(dash))==1
    fprintf(epsFile,'%1.2f %1.2f %1.2f setrgbcolor\n',...
            color(1),color(2),color(3));
    if dash>0
      fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
    end
    fprintf(epsFile,'newpath\n');
    fprintf(epsFile,...
      'Rmin %1.2f 2 copy cos mul 3 1 roll sin mul moveto\n',angleStart);
    fprintf(epsFile,'0 0 Rmax %1.2f %1.2f arc\n',angleStart,angleEnd);
    fprintf(epsFile,'0 0 Rmin %1.2f %1.2f arcn\n',angleEnd,angleStart);
    fprintf(epsFile,'closepath\n');
    fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
    if dash<0
      fprintf(epsFile,'fill\n');
    else
      fprintf(epsFile,'stroke\n');
    end
    fprintf(epsFile,'grestore\n');
  else
    fprintf(epsFile,'newpath\n');
    fprintf(epsFile,...
      'Rmin %1.2f 2 copy cos mul 3 1 roll sin mul moveto\n',angleStart);
    fprintf(epsFile,'0 0 Rmax %1.2f %1.2f arc\n',angleStart,angleEnd);
    fprintf(epsFile,'0 0 Rmin %1.2f %1.2f arcn\n',angleEnd,angleStart);
    fprintf(epsFile,'closepath clip\n');
    rot=(angleStart+angleEnd)/2-90;
```

```
    alpha=pi*(angleEnd-angleStart)/360;
    if alpha>pi/2
      width=2*maxRadius;
      y=maxRadius*cos(alpha);
      height=maxRadius-y-minRadius*cos(alpha);
    else
      width=2*maxRadius*sin(alpha);
      height=maxRadius-minRadius*cos(alpha);
      y=0;
    end
    x=-width/2;
    fprintf(epsFile,'%1.2f rotate\n',rot);
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    eimagexy(epsFile,dash,color,0,0,width,height);
    fprintf(epsFile,'grestore\n');
  end
```

1.3.15 eplotlg.m

```
%eplotlg ( epsFile,x,y,color,dash,lineWidth,textFont,textSize,textColor)
% this function write postscript commands in  epsFile
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eplotlg ( epsFile,x,y,color,dash,lineWidth,text,textFont,textSize,textColor)
  if (nargin~=10)
    eusage('eplotlg(epsFile,x,y,color,dash,lineWidth,text,textFont,textSize,textColor)');
  end
  textH=textSize*0.75;
  lineLength=3*textH;

  % print text
  etextxy(epsFile,x+lineLength+textH,y,0,1,text,textFont,textSize,textColor);

  % draw line
  if isstr(dash)
    fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
            color(1),color(2),color(3));
    fprintf(epsFile,'%1.2f %1.2f 2 copy translate 0 0 moveto\n',...
            x+lineLength/2,y+textH/3);
    fprintf(epsFile,'%s neg exch neg exch translate\n',dash);
    fprintf(epsFile,'setrgbcolor\n');
  elseif max(size(dash))==1
    if dash>=0
      y=y+textSize/4;
      exyline(epsFile,0,0,[x;x+lineLength],[y;y],color,dash,lineWidth);
    else
      erect(epsFile,x,y,lineLength,textH,0,color,dash,0)
      erect(epsFile,x,y,lineLength,textH,0,textColor,0,0)
    end
  else
    erect(epsFile,x,y,lineLength,textH,0,color,dash,0)
    erect(epsFile,x,y,lineLength,textH,0,textColor,0,0)
  end
```

1.3.16 epolplof.m

```
%epolplof (epsFile,x,y,alpha,radia,color)
% this function write postscript commands in  epsFile to plot data
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function epolplof(epsFile,x,y,alpha,radia,color)
  if (nargin~=6)
    eusage('epolplof(epsFile,x,y,alpha,radia,color)');
  end
  [rows colums]=size(alpha);
  if rows==1
    nData=colums;
    xradia=[alpha; radia];
  else
    nData=rows;
    xradia=[alpha'; radia'];
  end
  if alpha(1)==alpha(nData) & radia(1)==radia(nData)
    figOpen=0;
  else
    figOpen=1;
  end
  nData=2*nData;
  xradia=reshape(xradia,1,nData);
  dataStart=1;
  while dataStart<nData
    dataEnd=dataStart+49999;
    if dataEnd>nData
      dataEnd=nData;
    end
    array=sprintf('%1.2f ',xradia(dataStart:dataEnd));
    fprintf(epsFile,'/plotdata[%s] def\n',array);
    fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
            color(1),color(2),color(3));
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    fprintf(epsFile,'newpath\n');
    if figOpen
      fprintf(epsFile,'0 plotdata 0 get\n');
      fprintf(epsFile,'2 copy cos mul 3 1 roll sin mul moveto\n');
      fprintf(epsFile,'0 2 plotdata length 2 sub\n');
    else
      fprintf(epsFile,'plotdata 1 get plotdata 0 get\n');
      fprintf(epsFile,'2 copy cos mul 3 1 roll sin mul moveto\n');
      fprintf(epsFile,'2 2 plotdata length 2 sub\n');
    end
    fprintf(epsFile,' dup 1 add plotdata exch get\n');
    fprintf(epsFile,'exch plotdata exch get\n');
    fprintf(epsFile,'2 copy cos mul 3 1 roll sin mul lineto  for\n');
    if figOpen
      fprintf(epsFile,'0 plotdata plotdata length 1 sub get\n');
      fprintf(epsFile,'2 copy cos mul 3 1 roll sin mul lineto\n');
```

```
    end
    fprintf(epsFile,'closepath fill stroke\n');
    fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
    fprintf(epsFile,'setrgbcolor\n');
    dataStart=dataEnd+1;
  end
```

1.3.17 epolplos.m

```
%epolplos ( epsFile,x,y,alpha,radia,dash,color)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function epolplos ( epsFile,x,y,alpha,radia,dash,color)
  if (nargin~=7)
    eusage('epolplos(epsFile,x,y,alpha,radia,dash,color)');
  end
 [rows colums]=size(alpha);
  if rows==1
    nData=2*colums;
    xradia=[alpha; radia];
  else
    nData=2*rows;
    xradia=[alpha'; radia'];
  end
  xradia=reshape(xradia,1,nData);
  dataStart=1;
  while dataStart<nData
    dataEnd=dataStart+49999;
    if dataEnd>nData
      dataEnd=nData;
    end
    array=sprintf('%1.2f ',xradia(dataStart:dataEnd));
    fprintf(epsFile,'/plotdata[%s] def\n',array);
    fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
            color(1),color(2),color(3));
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    fprintf(epsFile,'0 2 plotdata length 2 sub\n');
    fprintf(epsFile,' dup 1 add plotdata exch get\n');
    fprintf(epsFile,'exch plotdata exch get\n');
    fprintf(epsFile,'dup rotate exch dup 0 translate %s\n',dash);
    fprintf(epsFile,'neg 0 translate neg rotate for\n');
    fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
    fprintf(epsFile,'setrgbcolor\n');
    dataStart=dataEnd+1;
  end
```

1.3.18 epolplot.m

```
%epolplot ( epsFile,x,y,alpha,radia,color,dash,lineWidth)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function epolplot ( epsFile,x,y,alpha,radia,color,dash,lineWidth)
```

```
  if (nargin~=8)
    eusage('epolplot(epsFile,x,y,alpha,radia,color,dash,lineWidth)');
  end
  [rows colums]=size(alpha);
  if rows==1
    nData=2*colums;
    xradia=[alpha; radia];
  else
    nData=2*rows;
    xradia=[alpha'; radia'];
  end
  xradia=reshape(xradia,1,nData);
  dataStart=1;
  while dataStart+2<nData
    dataEnd=dataStart+49999;
    if dataEnd>nData
      dataEnd=nData;
    end
    array=sprintf('%1.2f ',xradia(dataStart:dataEnd));
    fprintf(epsFile,'/plotdata[%s] def\n',array);
    if dash>0
      fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
    end
    fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
            color(1),color(2),color(3));
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    fprintf(epsFile,'newpath\n');
    fprintf(epsFile,'plotdata 1 get plotdata 0 get\n');
    fprintf(epsFile,'2 copy cos mul 3 1 roll sin mul moveto\n');
    fprintf(epsFile,'2 2 plotdata length 2 sub\n');
    fprintf(epsFile,' dup 1 add plotdata exch get\n');
    fprintf(epsFile,'exch plotdata exch get\n');
    fprintf(epsFile,'2 copy cos mul 3 1 roll sin mul lineto  for\n');
    fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
    fprintf(epsFile,'stroke\n');
    fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
    fprintf(epsFile,'setrgbcolor\n');
    fprintf(epsFile,'[] 0 setdash\n');
    dataStart=dataEnd-1;
  end
```

1.3.19 eptitle.m

```
%%NAME
%%  eptitle - print title of polar plot
%%
%%SYNOPSIS
%%  eptitle ([text[,distance[,fontSize]]])
%%
%%PARAMETER(S)
%%  text     title text
%%  distance distance from plot area
%%  fontSize fontsize of text
```

```
%%
%%GLOBAL PARAMETER(S)
%%  ePlotTitleText
%%  ePlotTitleDistance
%%  ePlotTitleFontSize
%%  ePlotTitleTextFont
%%  ePolarPlotAreaCenterPos
%%  ePolarPlotAreaRadMax
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function eptitle(text,distance,fontSize)
  if  nargin >3
    eusage('eptitle([text[,distance[,fontSize]]])');
  end
  eglobpar;
  if nargin<3
    fontSize=ePlotTitleFontSize;
  end
  if nargin<2
    distance=ePlotTitleDistance;
  end
  if nargin<1
    text=ePlotTitleText;
  end
  if strcmp(ePlotTitleText,'')~=1
    etext(ePlotTitleText,ePolarPlotAreaCenterPos(1),...
         ePolarPlotAreaCenterPos(2)+ePolarPlotAreaRadMax+...
         ePlotTitleDistance,ePlotTitleFontSize,0,ePlotTitleTextFont,0);
  end
```

1.3.20 erect.m

```
%erect ( epsFile , x , y , width , height , lineWidth, color, dash, rotation)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function erect(epsFile,x,y,width,height,lineWidth,color,dash,rotation)
  if (nargin~=9)
    eusage('erect(epsFile,x,y,width,height,lineWidth,color,dash,rotation)');
  end
  if max(size(dash))==1
    width=width-lineWidth;
    height=height-lineWidth;
    fprintf(epsFile,'gsave\n');
    if dash>0
      fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
    end
    fprintf(epsFile,'%1.2f %1.2f %1.2f setrgbcolor\n',...
                    color(1),color(2),color(3));
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    fprintf(epsFile,'%1.2f rotate\n',rotation);
    fprintf(epsFile,'newpath %1.2f dup moveto\n',lineWidth/2);
    fprintf(epsFile,'%1.2f 0 rlineto\n',width);
    fprintf(epsFile,'0 %1.2f rlineto\n',height);
```

```
    fprintf(epsFile,'%1.2f 0 rlineto\n',-width);
    fprintf(epsFile,'0 %1.2f rlineto\n',-height);
    fprintf(epsFile,'closepath\n');
    fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
    if dash<0
      fprintf(epsFile,'fill\n');
    else
      fprintf(epsFile,'stroke\n');
    end
    fprintf(epsFile,'grestore\n');
  else
    width=width-2*lineWidth;
    height=height-2*lineWidth;
    fprintf(epsFile,'gsave\n');
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    fprintf(epsFile,'%1.2f rotate\n',rotation);
    eimagexy(epsFile,dash,color,lineWidth,lineWidth,...
      width,height);
    fprintf(epsFile,'grestore\n');
  end
```

1.3.21 erectrc.m

```
%erect ( epsFile,x,y,width,height,lineWidth,color,dash,rotation,cornerRadius)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function erectrc(epsFile,x,y,width,height,lineWidth,color,dash,rotation,cornerRadius)
  if (nargin~=10)
    eusage('erectrc(epsFile,x,y,width,height,lineWidth,color,dash,rotation,cornerRadius)');
  end
  if max(size(dash))==1
    cornerRadius=cornerRadius-lineWidth/2;
    width=width-lineWidth;
    height=height-lineWidth;
    fprintf(epsFile,'gsave\n');
    if dash>0
      fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
    end
    fprintf(epsFile,'%1.2f %1.2f %1.2f setrgbcolor\n',...
                    color(1),color(2),color(3));
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    fprintf(epsFile,'%1.2f rotate\n',rotation);
    fprintf(epsFile,'%1.2f dup translate\n',lineWidth/2);
    fprintf(epsFile,'newpath\n');
    fprintf(epsFile,'%1.2f %1.2f moveto\n',...
      width-cornerRadius+lineWidth/2,0);
    fprintf(epsFile,'%1.2f 0 %1.2f %1.2f %1.2f arcto\n',...
      width,width,height-cornerRadius,cornerRadius);
    fprintf(epsFile,'%1.2f %1.2f %1.2f %1.2f %1.2f arcto\n',...
      width,height,cornerRadius,height,cornerRadius);
    fprintf(epsFile,'0 %1.2f 0 %1.2f %1.2f arcto\n',...
      height,cornerRadius,cornerRadius);
    fprintf(epsFile,'0 0 %1.2f 0 %1.2f arcto\n',...
```

```
      cornerRadius,cornerRadius);
    fprintf(epsFile,'closepath\n');
    fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
    if dash<0
      fprintf(epsFile,'fill\n');
    else
      fprintf(epsFile,'stroke\n');
    end
    fprintf(epsFile,'grestore\n');
  else
    cornerRadius=cornerRadius-lineWidth;
    width=width-2*lineWidth;
    height=height-2*lineWidth;
    fprintf(epsFile,'gsave\n');
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    fprintf(epsFile,'%1.2f rotate\n',rotation);
    fprintf(epsFile,'%1.2f dup translate\n',lineWidth);
    fprintf(epsFile,'newpath\n');
    fprintf(epsFile,'%1.2f dup %1.2f add exch moveto\n',...
      lineWidth,width-cornerRadius);
    fprintf(epsFile,'%1.2f 0 moveto\n',width-cornerRadius);
    fprintf(epsFile,'%1.2f 0 %1.2f %1.2f %1.2f arcto\n',...
      width,width,height-cornerRadius,cornerRadius);
    fprintf(epsFile,'%1.2f %1.2f %1.2f %1.2f %1.2f arcto\n',...
      width,height,cornerRadius,height,cornerRadius);
    fprintf(epsFile,'0 %1.2f 0 %1.2f %1.2f arcto\n',...
      height,cornerRadius,cornerRadius);
    fprintf(epsFile,'0 0 %1.2f 0 %1.2f arcto\n',...
      cornerRadius,cornerRadius);
    fprintf(epsFile,'closepath clip\n');
    eimagexy(epsFile,dash,color,0,0,width,height);
    fprintf(epsFile,'grestore\n');
  end
```

1.3.22 erencode.m

```
% newFont=erencode( epsFile ,oldFont)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function newFont=erencode(epsFile,oldFont)
  if (nargin~=2)
    eusage('newFont=erencode(epsFile,oldFont)');
  end
  blaStr='                                                  ';
  oldSize=length(oldFont);
  oldFont=oldFont(find(oldFont~=' '));
  newFont=sprintf('%sG',oldFont);
  fprintf(epsFile,'(%s) (%s) ReEncode\n',oldFont,newFont);
  newFont=[newFont blaStr(1:oldSize+1-length(oldFont))];
```

1.3.23 escalecl.m

```
%valuePos=escalecl(epsFile,side,x,y,offset,angle,length,valueStart,valueStep,
% valueEnd,vForm,vVisible,fontSize,lineWidth,longTicLength,nMax,space,color)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function valuePos=escalecl(epsFile,side,x,y,offset,angle,length,valueStart,...
                  valueStep,valueEnd,vForm,vVisible,fontSize,...
                  lineWidth,longTicLength,nMax,space,color)
  if (nargin~=18)
    eusage(...
   'valuePos=escalecl(epsFile,side,x,y,offset,angle,length,valueStart,valueStep,valueEnd,vForm,vVisi
  end
  startEndDiff=valueEnd-valueStart;
  signOfDiff=sign(startEndDiff);
  if valueStep==0
    valueStep=signOfDiff*eticdis(signOfDiff*startEndDiff,nMax);
  end
  if vForm==0
    expo=-log10(valueStep*signOfDiff);
    if rem(expo,1)>0
      expo=expo+1;
    end
    autoForm=fix(expo);
    if autoForm>0
      vForm=autoForm;
    end
  end
  if vForm<0
    valueForm='%g';
  else
    valueForm=sprintf('%%1.%df',vForm);
  end

  valueStart=valueStart+valueStep;
  startEndDiff=valueEnd-valueStart;
  classValue=valueStart:valueStep:valueEnd;
  classStart=classValue-valueStep/2;
  classStart=(classStart-classStart(1))/...
    (startEndDiff+valueStep);
  if size(classStart,2) <2
    classEnd=classStart+1;
  else
    classEnd=classStart+classStart(2);
  end

  if side=='s'
    startPos=0;
    xLength=length;
    yLength=0;
    valueOffset=-(offset+space);
    moveForm=sprintf('%%1.2f %1.2f moveto\n',-offset);
    ticLineForm='0 -%1.2f rlineto\n';
    moveValueForm=sprintf('0 -%1.2f rmoveto\n',space+fontSize*0.72);
```

```
  showForm='(%s) dup stringwidth pop dup 2 div sub neg 0 rmoveto show\n';
elseif side=='n'
  startPos=0;
  xLength=length;
  yLength=0;
  valueOffset=offset+space;
  moveForm=sprintf('%%1.2f %1.2f moveto\n',offset);
  ticLineForm='0 %1.2f rlineto\n';
  moveValueForm=sprintf('0 %1.2f rmoveto\n',space);
  showForm='(%s) dup stringwidth pop dup 2 div sub neg 0 rmoveto show\n';
elseif side=='w'
  startPos=0;
  xLength=0;
  yLength=length;
  valueOffset=-(offset+space);
  moveForm=sprintf('%1.2f %%1.2f moveto\n',-offset);
  ticLineForm='-%1.2f 0 rlineto\n';
  moveValueForm=sprintf('-%1.2f -%1.2f 0.28 mul rmoveto\n',space,fontSize);
  showForm='(%s) dup stringwidth pop neg 0 rmoveto show\n';
elseif side=='e'
  startPos=0;
  xLength=0;
  yLength=length;
  valueOffset=offset+space;
  moveForm=sprintf('%1.2f %%1.2f moveto\n',offset);
  ticLineForm='%1.2f 0 rlineto\n';
  moveValueForm=sprintf('%1.2f -%1.2f 0.28 mul rmoveto\n',space,fontSize);
  showForm='(%s) show\n';
end



% start draw
fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
fprintf(epsFile,'%1.2f rotate\n',angle);
fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
        color(1),color(2),color(3));
fprintf(epsFile,'newpath\n');
fprintf(epsFile,'/Helvetica findfont %d scalefont setfont\n',fontSize);
nClasses=size(classStart,2);
valuePos=zeros(nClasses,1);
for i=1:nClasses
  % long tic of start of class
  currentPos=startPos+length*classStart(i);
  fprintf(epsFile,moveForm,currentPos);
  fprintf(epsFile,ticLineForm,longTicLength);
  % long tic of end of class
  currentPos=startPos+length*classEnd(i);
  fprintf(epsFile,moveForm,currentPos);
  fprintf(epsFile,ticLineForm,longTicLength);
  % class value
  currentPos=startPos+length*(classStart(i)+classEnd(i))/2;
  valuePos(i)=currentPos;
  fprintf(epsFile,moveForm,currentPos);
```

```
    if vVisible
      fprintf(epsFile,moveValueForm);
      if abs(classValue(i))<1e-14
          classValue(i)=0;
      end
      valueStr=sprintf(valueForm,classValue(i));
      fprintf(epsFile,showForm,valueStr);
    end
  end
  valueOffset=ones(nClasses,1)*valueOffset;
  sinAngle=sin(angle*pi/180);
  cosAngle=cos(angle*pi/180);
  if xLength==0
    valueDeltaX=valueOffset*cosAngle-valuePos*sinAngle;
    valueDeltaY=valueOffset*sinAngle+valuePos*cosAngle;
  else
    valueDeltaX=valuePos*cosAngle-valueOffset*sinAngle;
    valueDeltaY=valuePos*sinAngle+valueOffset*cosAngle;
  end
  valuePos=[x+valueDeltaX y+valueDeltaY];


  %axis
  fprintf(epsFile,moveForm,startPos);
  fprintf(epsFile,'%1.2f %1.2f rlineto\n',xLength,yLength);
  fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
  fprintf(epsFile,'stroke\n');
  fprintf(epsFile,'setrgbcolor\n');
  fprintf(epsFile,'%1.2f rotate\n',-angle);
  fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
```

1.3.24 escalelog.m

```
%longTicPos=escalelog(epsFile,side,x,y,offset,angle,length,startValue,step,
%         endValue,vForm,vVisible,fontSize,lineWidth,shortTicLength,
%         longTicLength,maxValues,space,color)
% log scaled axis
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function longTicPos=escalelog(epsFile,side,x,y,offset,angle,length,...
          startValue,step,endValue,vForm,vVisible,fontSize,lineWidth,...
          shortTicLength,longTicLength,maxValues,space,color)
  if (nargin~=19)
    usage(...
   'longTicPos=escalelog(epsFile,side,x,y,offset,angle,length,startValue,step,endValue,vForm,vVisibl
  end

  startPos=0;
  if side=='s'
    xLength=length;
    yLength=0;
    longTicOffset=-(offset+longTicLength+space);
```

```
    moveForm=sprintf('%%1.2f %1.2f moveto\n',-offset);
    ticLineForm='0 %1.2f neg rlineto\n';
    moveValueForm=sprintf('0 -%1.2f rmoveto\n',space+fontSize*0.72);
    showForm='(%s) dup stringwidth pop dup 2 div sub neg 0 rmoveto show\n';
    showFormE='dup 2 div sub neg 0 rmoveto (%s) show\n';
  elseif side=='n'
    xLength=length;
    yLength=0;
    longTicOffset=offset+longTicLength+space;
    moveForm=sprintf('%%1.2f %1.2f moveto\n',offset);
    ticLineForm='0 %1.2f rlineto\n';
    moveValueForm=sprintf('0 %1.2f rmoveto\n',space);
    showForm='(%s) dup stringwidth pop dup 2 div sub neg 0 rmoveto show\n';
    showFormE='dup 2 div sub neg 0 rmoveto (%s) show\n';
  elseif side=='w'
    xLength=0;
    yLength=length;
    longTicOffset=-(offset+longTicLength+space);
    moveForm=sprintf('%1.2f %%1.2f moveto\n',-offset);
    ticLineForm='%1.2f neg 0 rlineto\n';
    moveValueForm=sprintf('-%1.2f -%1.2f 0.28 mul rmoveto\n',space,fontSize);
    showForm='(%s) dup stringwidth pop neg 0 rmoveto show\n';
    showFormE='neg 0 rmoveto (%s) show\n';
  elseif side=='e'
    xLength=0;
    yLength=length;
    longTicOffset=offset+longTicLength+space;
    moveForm=sprintf('%1.2f %%1.2f moveto\n',offset);
    ticLineForm='%1.2f 0 rlineto\n';
    moveValueForm=sprintf('%1.2f -%1.2f 0.28 mul rmoveto\n',space,fontSize);
    showForm='(%s) show\n';
    showFormE='(%s) show\n';
  end
  startEndDiff=endValue-startValue;

  signOfDelta=sign(startEndDiff);
  if (step==0)
    deltaLabel=eticdis(signOfDelta*startEndDiff,maxValues);
  else
    deltaLabel=abs(step);
  end
  if deltaLabel>1
    shortTicLength=0;
  end
  if deltaLabel<1
    deltaLabel=1;
  end

  %start ticNo and offset
  nShortTics=rem(startValue,deltaLabel)/deltaLabel*9*signOfDelta;
  if nShortTics<0
    nShortTics=9+nShortTics;
  end
  if rem(nShortTics,1)>0
```

```
  iTic=fix(nShortTics)+1;
  ticOffset=(iTic-nShortTics)*signOfDelta*deltaLabel/9;
else
  iTic=nShortTics;
  ticOffset=0;
end
firstTicValue=startValue+ticOffset;
currentValue=firstTicValue;
deltaTic=signOfDelta*deltaLabel/9;
axisFac=length/startEndDiff;
shortTicD=log10([2 3 4 5 6 7 8 9]);

% start draw
fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
fprintf(epsFile,'%1.2f rotate\n',angle);
fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
        color(1),color(2),color(3));
fprintf(epsFile,'newpath\n');
if vForm>=0
  digitForm=sprintf('%%0.%df',vForm);
end

%tics
currentTic=0;
currentValue=firstTicValue;
longTicPos=0;
nPos=0;
while signOfDelta*currentValue<=signOfDelta*endValue
  rest=rem(iTic,9);
  if rest~=0
    % short tics
    interValue=currentValue+deltaTic*(9*shortTicD(rest)-rest);
    if signOfDelta*interValue<=signOfDelta*endValue
      currentPos=startPos+axisFac*(interValue-startValue);
      fprintf(epsFile,moveForm,currentPos);
      fprintf(epsFile,ticLineForm,shortTicLength);
    end
  else
    % long tics
    currentPos=startPos+axisFac*(currentValue-startValue);
    longTicPos=[longTicPos;currentPos];
    nPos=nPos+1;
    fprintf(epsFile,moveForm,currentPos);
    fprintf(epsFile,ticLineForm,longTicLength);
    % value
    if vVisible & (offset>=0 | abs(currentValue)>1e-14)
      fprintf(epsFile,moveValueForm);
      if abs(currentValue)>vForm
        valueForm=sprintf('%1.0f',currentValue);
        fprintf(epsFile,'/Helvetica findfont %d scalefont setfont\n',...
                fontSize*0.7);
        fprintf(epsFile,'(%s) stringwidth pop\n',valueForm);
        fprintf(epsFile,'/Helvetica findfont %d scalefont setfont\n',...
                fontSize);
```

```
        fprintf(epsFile,'(10) stringwidth pop add\n');
        fprintf(epsFile,showFormE,'10');
        fprintf(epsFile,'/Helvetica findfont %d scalefont setfont\n',...
                fontSize*0.7);
        fprintf(epsFile,'0 %1.2f rmoveto\n',fontSize*0.6);
        fprintf(epsFile,'(%s) show\n',valueForm);
      else
        valueForm=sprintf(digitForm,10^currentValue);
        pos=find(valueForm~='0');
        lpos=size(pos,2);
        pos=pos(lpos);
        if pos<size(valueForm,2)
          if  valueForm(pos)=='.'
            pos=pos-1;
          end
          valueForm=valueForm(1:pos);
        end
        fprintf(epsFile,'/Helvetica findfont %d scalefont setfont\n',...
                fontSize);
        fprintf(epsFile,'(%s) stringwidth pop\n',valueForm);
        fprintf(epsFile,showForm,valueForm);
      end
    end
  end
  iTic=iTic+1;
  currentTic=currentTic+1;
  currentValue=firstTicValue+currentTic*deltaTic;
end
longTicPos=longTicPos(2:nPos+1);
longTicOffset=ones(nPos,1)*longTicOffset;
sinAngle=sin(angle*pi/180);
cosAngle=cos(angle*pi/180);
if xLength==0
  longTicDeltaX=longTicOffset*cosAngle-longTicPos*sinAngle;
  longTicDeltaY=longTicOffset*sinAngle+longTicPos*cosAngle;
else
  longTicDeltaX=longTicPos*cosAngle-longTicOffset*sinAngle;
  longTicDeltaY=longTicPos*sinAngle+longTicOffset*cosAngle;
end
longTicPos=[x+longTicDeltaX y+longTicDeltaY];

%axis
fprintf(epsFile,moveForm,startPos);
fprintf(epsFile,'%1.2f %1.2f rlineto\n',xLength,yLength);
fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
fprintf(epsFile,'stroke\n');
fprintf(epsFile,'setrgbcolor\n');
fprintf(epsFile,'%1.2f rotate\n',-angle);
fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
```

1.3.25 escalepa.m

```
%longTicAngle=escalepa(epsFile,x,y,minRadius,maxRadius,angle1,angle2,...
```

```
%         startValue,step,endValue,vForm,vVisible,fontSize,lineWidth,...
%         shortTicLength,longTicLength,maxValues,space,color)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function longTicAngle=escalepa(epsFile,x,y,minRadius,maxRadius,angle1,angle2,...
  startValue,step,endValue,vForm,vVisible,fontSize,lineWidth,...
          shortTicLength,longTicLength,maxValues,space,color)
  if (nargin~=19)
    eusage(...
   'longTicAngle=escalepa(epsFile,x,y,minRadius,maxRadius,angle1,angle2,startValue,step,endValue,vFo
  end
  startEndDiff=endValue-startValue;
  if step==0
    %autoscale
    signOfDelta=sign(startEndDiff);
    deltaLabel=eticdis(signOfDelta*startEndDiff,maxValues);
  else
    %fixscale
    signOfDelta=sign(step);
    deltaLabel=signOfDelta*step;
  end
  if vForm==0
    expo=-log10(deltaLabel);
    if rem(expo,1)>0
      expo=expo+1;
    end
    autoForm=fix(expo);
    if autoForm>0
      vForm=autoForm;
    end
  end
  if vForm<0
    valueForm='%g';
  else
    valueForm=sprintf('%%1.%df',vForm);
  end
  if rem(deltaLabel,3)==0
    nTics=3;
  else
    nTics=5;
  end
  %start ticNo and offset
  nShortTics=rem(startValue,deltaLabel)/deltaLabel*nTics*signOfDelta;
  if nShortTics<0
    nShortTics=5+nShortTics;
  end
  if rem(nShortTics,1)>0
      i=fix(nShortTics)+1;
      ticOffset=(i-nShortTics)*signOfDelta*deltaLabel/nTics;
  else
    i=nShortTics;
    ticOffset=0;
  end
  firstTicValue=startValue+ticOffset;
```

```
currentValue=firstTicValue;
deltaTic=signOfDelta*deltaLabel/nTics;
angleDiff=angle2-angle1;
axisFac=angleDiff/startEndDiff;
startPos=angle1;
if angleDiff==360
 endValue=endValue-deltaTic;
end

moveForm=sprintf('%1.2f %%1.2f 2 copy cos mul 3 1 roll sin mul moveto\n',...
                maxRadius);
ticLineForm='%1.2f %1.2f 2 copy cos mul 3 1 roll sin mul rlineto\n';
moveValueForm=sprintf(...
                '%1.2f %%1.2f 2 copy cos mul 3 1 roll sin mul rmoveto\n',...
                space+10);
moveFontSizeForm=sprintf('0 %1.2f neg rmoveto\n',fontSize*0.28);
showForm='(%s) dup stringwidth pop dup 2 div sub neg 0 rmoveto show\n';

% start draw
fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
        color(1),color(2),color(3));
fprintf(epsFile,'newpath\n');
fprintf(epsFile,'/Helvetica findfont %d scalefont setfont\n',fontSize);

%tics
currentTic=0;
currentValue=firstTicValue;
longTicAngle=0;
nPos=0;
while signOfDelta*currentValue<=signOfDelta*endValue
  currentPos=startPos+axisFac*(currentValue-startValue);
  fprintf(epsFile,moveForm,currentPos);
  if rem(i,nTics)~=0
    % short tics
    fprintf(epsFile,ticLineForm,shortTicLength,currentPos);
  else
    % long tics
    longTicAngle=[longTicAngle;currentPos];
    nPos=nPos+1;
    fprintf(epsFile,ticLineForm,longTicLength,currentPos);
    % value
    if vVisible
      fprintf(epsFile,moveValueForm,currentPos);
      fprintf(epsFile,moveFontSizeForm);
      if abs(currentValue)<1e-14
        currentValue=0;
      end
      valueStr=sprintf(valueForm,currentValue);
      fprintf(epsFile,showForm,valueStr);
    end
  end
  i=i+1;
  currentTic=currentTic+1;
```

```
    currentValue=firstTicValue+currentTic*deltaTic;
  end
  longTicAngle=longTicAngle(2:nPos+1);

  %axis
  fprintf(epsFile,'%1.2f %1.2f 2 copy cos mul 3 1 roll sin mul moveto\n',...
                  maxRadius,startPos);
  fprintf(epsFile,'0 0 %1.2f %1.2f %1.2f arc\n',maxRadius,angle1,angle2);
  fprintf(epsFile,'%1.2f %1.2f 2 copy cos mul 3 1 roll sin mul moveto\n',...
                  minRadius,startPos);
  fprintf(epsFile,'0 0 %1.2f %1.2f %1.2f arc\n',minRadius,angle1,angle2);
  fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
  fprintf(epsFile,'stroke\n');
  fprintf(epsFile,'setrgbcolor\n');
  fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
```

1.3.26 escalexy.m

```
%longTicPos=escalexy(epsFile,side,x,y,offset,angle,length,startValue,step,
%         endValue,vForm,vVisible,fontSize,lineWidth,shortTicLength,
%         longTicLength,maxValues,space,color)
% linear scaled axis
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function longTicPos=escalexy(epsFile,side,x,y,offset,angle,length,startValue,...
              step,endValue,vForm,vVisible,fontSize,lineWidth,shortTicLength,...
              longTicLength,maxValues,space,color)
  if (nargin~=19)
    eusage(...
   'longTicPos=escalexy(epsFile,side,x,y,offset,angle,length,startValue,step,endValue,vForm,vVisible
  end
  if side=='s'
    startPos=0;
    xLength=length;
    yLength=0;
    longTicOffset=-(offset+longTicLength+space);
    moveForm=sprintf('%%1.2f %1.2f moveto\n',-offset);
    ticLineForm='0 %1.2f neg rlineto\n';
    moveValueForm=sprintf('0 -%1.2f rmoveto\n',space+fontSize*0.72);
    showForm='(%s) dup stringwidth pop dup 2 div sub neg 0 rmoveto show\n';
  elseif side=='n'
    startPos=0;
    xLength=length;
    yLength=0;
    longTicOffset=offset+longTicLength+space;
    moveForm=sprintf('%%1.2f %1.2f moveto\n',offset);
    ticLineForm='0 %1.2f rlineto\n';
    moveValueForm=sprintf('0 %1.2f rmoveto\n',space);
    showForm='(%s) dup stringwidth pop dup 2 div sub neg 0 rmoveto show\n';
  elseif side=='w'
    startPos=0;
    xLength=0;
    yLength=length;
```

```
  longTicOffset=-(offset+longTicLength+space);
  moveForm=sprintf('%1.2f %%1.2f moveto\n',-offset);
  ticLineForm='%1.2f neg 0 rlineto\n';
  moveValueForm=sprintf('-%1.2f -%1.2f 0.28 mul rmoveto\n',space,fontSize);
  showForm='(%s) dup stringwidth pop neg 0 rmoveto show\n';
elseif side=='e'
  startPos=0;
  xLength=0;
  yLength=length;
  longTicOffset=offset+longTicLength+space;
  moveForm=sprintf('%1.2f %%1.2f moveto\n',offset);
  ticLineForm='%1.2f 0 rlineto\n';
  moveValueForm=sprintf('%1.2f -%1.2f 0.28 mul rmoveto\n',space,fontSize);
  showForm='(%s) show\n';
end
startEndDiff=endValue-startValue;
if step==0
  %autoscale
  signOfDelta=sign(startEndDiff);
  deltaLabel=eticdis(signOfDelta*startEndDiff,maxValues);
else
  %fixscale
  signOfDelta=sign(step);
  deltaLabel=signOfDelta*step;
end
if vForm==0
  expo=-log10(deltaLabel);
  if rem(expo,1)>0
    expo=expo+1;
  end
  autoForm=fix(expo);
  if autoForm>0
    vForm=autoForm;
  end
end
if vForm<0
  valueForm='%g';
else
  valueForm=sprintf('%%1.%df',vForm);
end


%start ticNo and offset
nShortTics=rem(startValue,deltaLabel)/deltaLabel*5*signOfDelta;
if nShortTics<0
  nShortTics=5+nShortTics;
end
if rem(nShortTics,1)>0
  iTic=fix(nShortTics)+1;
  ticOffset=(iTic-nShortTics)*signOfDelta*deltaLabel/5;
else
  iTic=nShortTics;
  ticOffset=0;
end
```

```
firstTicValue=startValue+ticOffset;
currentValue=firstTicValue;
deltaTic=signOfDelta*deltaLabel/5;
axisFac=length/startEndDiff;

% start draw
fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
fprintf(epsFile,'%1.2f rotate\n',angle);
fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
        color(1),color(2),color(3));
fprintf(epsFile,'newpath\n');
fprintf(epsFile,'/Helvetica findfont %d scalefont setfont\n',fontSize);

%tics
currentTic=0;
currentValue=firstTicValue;
longTicPos=0;
nPos=0;
while signOfDelta*currentValue<=signOfDelta*endValue
  currentPos=startPos+axisFac*(currentValue-startValue);
  fprintf(epsFile,moveForm,currentPos);
  if rem(iTic,5)~=0
    % short tics
    fprintf(epsFile,ticLineForm,shortTicLength);
  else
    % long tics
    longTicPos=[longTicPos;currentPos];
    nPos=nPos+1;
    fprintf(epsFile,ticLineForm,longTicLength);
    % value
    if vVisible & (offset>=0 | abs(currentValue)>1e-14)
      if abs(currentValue)<1e-14
        currentValue=0;
      end
      fprintf(epsFile,moveValueForm);
      valueStr=sprintf(valueForm,currentValue);
      fprintf(epsFile,showForm,valueStr);
    end
  end
  iTic=iTic+1;
  currentTic=currentTic+1;
  currentValue=firstTicValue+currentTic*deltaTic;
end
longTicPos=longTicPos(2:nPos+1);
longTicOffset=ones(nPos,1)*longTicOffset;
sinAngle=sin(angle*pi/180);
cosAngle=cos(angle*pi/180);
if xLength==0
  longTicDeltaX=longTicOffset*cosAngle-longTicPos*sinAngle;
  longTicDeltaY=longTicOffset*sinAngle+longTicPos*cosAngle;
else
  longTicDeltaX=longTicPos*cosAngle-longTicOffset*sinAngle;
  longTicDeltaY=longTicPos*sinAngle+longTicOffset*cosAngle;
end
```

```
  longTicPos=[x+longTicDeltaX y+longTicDeltaY];

  %axis
  fprintf(epsFile,moveForm,startPos);
  fprintf(epsFile,'%1.2f %1.2f rlineto\n',xLength,yLength);
  fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
  fprintf(epsFile,'stroke\n');
  fprintf(epsFile,'setrgbcolor\n');
  fprintf(epsFile,'%1.2f rotate\n',-angle);
  fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
```

1.3.27 etextxy.m

```
%etextxy(epsFile,x,y,rotation,alignment,text,font,size,color)
% this function write postscript commands in  epsFile to draw a West scale
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function etextxy(epsFile,x,y,rotation,alignment,text,font,fontSize,color)
  if (nargin~=9)
    eusage('etextxy(epsFile,x,y,rotation,alignment,text,font,fontSize,color)');
  end
  fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
          color(1),color(2),color(3));
  if x~=0
    fprintf(epsFile,'%1.2f %1.2f moveto\n',x,y);
  else
    fprintf(epsFile,'0 %1.2f rmoveto\n',y);
  end
  if length(fontSize)>2
    fprintf(epsFile,...
      '/%s findfont [%1.2f 0 %1.2f %1.2f 0 0] makefont setfont\n',...
      font,fontSize(1),fontSize(2)*tan(fontSize(3)*pi/180),fontSize(2));
  else
    fprintf(epsFile,'/%s findfont %1.2f scalefont setfont\n',font,fontSize(1));
  end
  if rotation~=0
    if (alignment==0) | (alignment==3)
      fprintf(epsFile,'(%s) stringwidth pop 2 div %1.2f 2 copy\n',text,rotation);
      fprintf(epsFile,'cos mul 3 1 roll sin mul\n');
    elseif (alignment==1) | (alignment==4)
      fprintf(epsFile,'(%s) stringwidth pop %1.2f 2 copy\n',text,rotation);
      fprintf(epsFile,'cos mul 3 1 roll sin mul\n');
    else
      fprintf(epsFile,'0 0\n');
    end
    fprintf(epsFile,'gsave %1.2f rotate\n',rotation);
  end
  if alignment==0
    fprintf(epsFile,...
      '(%s) dup stringwidth pop dup 2 div sub neg 0 rmoveto show\n',text);
  elseif alignment==3
    fprintf(epsFile,...
      '(%s) dup stringwidth pop dup 2 div sub neg %1.2f rmoveto show\n',...
```

```
      text,-fontSize(1)/4);
  elseif alignment==1
    fprintf(epsFile,...
      '(%s) show\n',text);
  elseif alignment==4
    fprintf(epsFile,...
      '(%s) 0 %1.2f rmoveto show\n',text,-fontSize(1)/4);
  elseif alignment==2
    fprintf(epsFile,...
      '(%s) dup stringwidth pop neg %1.2f rmoveto show\n',text,-fontSize(1)/4);
  else
    fprintf(epsFile,...
      '(%s) dup stringwidth pop neg 0 rmoveto show\n',text);
  end
  if rotation~=0
    fprintf(epsFile,'grestore rmoveto\n');
  end
```

1.3.28 eticdis.m

```
%ticDistance=eticdis ( axesLength,maxTics)
% compute tic distance for axes
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function ticDistance=eticdis(axesLength,maxTics)

  % test parameter
  if (nargin~=2)
    eusage('ticDistance=eticdis(axesLength,maxTics)');
  end
  if maxTics<2
    maxTics=2;
  end
  if axesLength<0
    axesLength=axesLength*(-1.0);
  end

  % get exponent
  factor=[0.5 2 1];
  exponent=1;
  while axesLength>=1.0
    axesLength=axesLength/10;
    exponent=exponent*10;
  end

  % get distance factor
  ticDistance=factor(3);
  i=1;
  currentTics=axesLength/factor(i);
  while currentTics<=maxTics
    ticDistance=factor(i);
    i=i+1;
    if i>3
```

```
      i=1;
    end
    factor(i)=factor(i)/10;
    currentTics=axesLength/factor(i);
  end

  ticDistance=ticDistance*exponent;
```

1.3.29 etitle.m

```
%%NAME
%%  etitle - print title of plot
%%
%%SYNOPSIS
%%  etitle ([text[,distance[,fontSize[,color]]]])
%%
%%PARAMETER(S)
%%  text     title text
%%  distance distance from plot area
%%  fontSize fontsize of text
%%  color of text
%%
%%GLOBAL PARAMETER(S)
%%  ePlotTitleText
%%  ePlotTitleDistance
%%  ePlotTitleFontSize
%%  ePlotTitleTextFont
%%  ePlotAreaPos
%%  ePlotAreaWidth
%%  ePlotAreaHeight
%%  eTextColor
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function etitle(text,distance,fontSize,color)
  if  nargin >4
    eusage('etitle([text[,distance[,fontSize[,color]]]])');
  end
  eglobpar;
  if nargin<4
    color=eTextColor;
  end
  if nargin<3
    fontSize=ePlotTitleFontSize;
  end
  if nargin<2
    distance=ePlotTitleDistance;
  end
  if nargin<1
    text=ePlotTitleText;
  end
  if strcmp(text,'')~=1
    etext(text,ePlotAreaPos(1)+ePlotAreaWidth/2,...
       ePlotAreaPos(2)+ePlotAreaHeight+distance,...
```

```
      fontSize,0,ePlotTitleTextFont,0,color);
  end
```

1.3.30 etxt2box.m

```
%etxt2box(epsFile,text,x,y,width,height,lFeed,pFeed,wBreak,pBreak,alignment,font,fontSize,color,spac
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function etxt2box(epsFile,text,x,y,width,height,lFeed,pFeed,wBreak,pBreak,alignment,font,fontSize,co
  if (nargin~=19)
    eusage('etxt2box(epsFile,text,x,y,width,height,lFeed,pFeed,wBreak,pBreak,alignment,font,fontSize
  end
  fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
          color(1),color(2),color(3));
  if length(fontSize)>2
    fprintf(epsFile,...
      '/%s findfont [%1.2f 0 %1.2f %1.2f 0 0] makefont setfont\n',...
     font,fontSize(1),fontSize(2)*tan(fontSize(3)*pi/180),fontSize(2));
  else
    fprintf(epsFile,'/%s findfont %1.2f scalefont setfont\n',font,fontSize(1));
  end

  fprintf(epsFile,'/lf %1.2f def\n',lFeed);
  fprintf(epsFile,'/pf %1.2f def\n',pFeed);
  fprintf(epsFile,'/bl ( ) def /blw bl stringwidth pop def\n');
  fprintf(epsFile,'/wb (%s) def\n',wBreak);
  fprintf(epsFile,'/pb (%s) def /pbl pb length def\n',pBreak);
  if width>0
    fprintf(epsFile,'/lw %1.2f def\n',width-spaceW-spaceE);
    fprintf(epsFile,'/lineX %1.2f def\n',x+spaceW);
  else
    fprintf(epsFile,'/lw %1.2f def\n',-width-spaceW-spaceE);
    fprintf(epsFile,'/lineX %1.2f def\n',x+width+spaceW);
  end
  if height>0
    fprintf(epsFile,'/th %1.2f def\n',height-spaceN-spaceS);
    fprintf(epsFile,'/lineY %1.2f def\n',y+height-spaceN);
  else
    fprintf(epsFile,'/th %1.2f def\n',-height-spaceN-spaceS);
    fprintf(epsFile,'/lineY %1.2f def\n',y-spaceN);
  end
  if alignment==0
    fprintf(epsFile,...
      '/pline lineX  lw 2 div add clw 2 div sub lineY moveto show\n');
    fprintf(epsFile,'/lineY lineY lf sub def  def\n');
  elseif alignment==2
    fprintf(epsFile,...
      '/plastline  lineX lineY moveto show /lineY lineY lf sub def def\n');
    fprintf(epsFile,'/pline /lineText exch def\n');
    fprintf(epsFile,'/space lw clw sub def\n');
    fprintf(epsFile,'/ns 0 def /sText lineText def\n');
    fprintf(epsFile,'sText bl search\n');
    fprintf(epsFile,'pop pop  /sText exch def /ns ns 1 add def\n');
```

```
  fprintf(epsFile,'pop exit ifelseloop\n');
  fprintf(epsFile,'lineX lineY moveto\n');
  fprintf(epsFile,'ns 0 gt\n');
  fprintf(epsFile,'/space space ns 1 sub div def\n');
  fprintf(epsFile,'/lX lineX def\n');
  fprintf(epsFile,'/sText lineText def\n');
  fprintf(epsFile,'sText bl search\n');
  fprintf(epsFile,'/nword exch def pop /sText exch def nword show\n');
  fprintf(epsFile,'/lX lX blw add space add nword stringwidth pop add def\n');
  fprintf(epsFile,'lX lineY moveto');
  fprintf(epsFile,'pop exit ifelseloop\n');
  fprintf(epsFile,'sText show\n');
  fprintf(epsFile,'/lineY lineY lf sub def \n');
  fprintf(epsFile,'lineText show /lineY lineY lf sub def ifelse def\n');
elseif alignment==-1
  fprintf(epsFile,...
    '/pline lineX lw add clw sub lineY moveto show\n');
  fprintf(epsFile,'/lineY lineY lf sub def  def\n');
else
  fprintf(epsFile,...
    '/pline  lineX lineY moveto show /lineY lineY lf sub def def\n');
end
fprintf(epsFile,'newpath lineX lineY moveto lw 0 rlineto 0 th neg rlineto\n');
fprintf(epsFile,'lw neg 0 rlineto closepath clip\n');
fprintf(epsFile,'/lineX lineX %1.2f add def\n',offset(1));
fprintf(epsFile,'/lineY lineY lf 0.75 mul sub %1.2f add def\n',offset(2));
fprintf(epsFile,'/text (%s) def\n',text);
fprintf(epsFile,'/ePos text length def\n');
fprintf(epsFile,'/rText ePos 1 add string def\n');
fprintf(epsFile,'rText 0 text putinterval\n');
fprintf(epsFile,'rText ePos pb putinterval\n');
fprintf(epsFile,'rText pb search\n');
fprintf(epsFile,'/pText exch def pop /rText exch def\n');
fprintf(epsFile,'/ePos pText length def\n');
fprintf(epsFile,'/lText ePos 1 add string def\n');
fprintf(epsFile,'lText 0 pText putinterval /pText lText def\n');
fprintf(epsFile,'pText ePos wb putinterval /prText pText def\n');
fprintf(epsFile,'/clw blw neg def /sPos 0 def /cPos 0 def\n');
fprintf(epsFile,'prText wb search\n/nw exch def pop /prText exch def ');
fprintf(epsFile,'/ww nw stringwidth pop def\nclw blw add ww add lw gt\n');
fprintf(epsFile,'pText sPos cPos sPos sub getinterval pline ');
fprintf(epsFile,'/sPos cPos def /clw ww def \n');
fprintf(epsFile,'/clw clw blw add ww add def ifelse\n');
fprintf(epsFile,'/cPos cPos nw length add 1 add def\n');
fprintf(epsFile,'pText cPos 1 sub bl putinterval\n');
fprintf(epsFile,'pop exit ifelse  loop\n');
fprintf(epsFile,'pText sPos ePos sPos sub getinterval\n');
if alignment==2
  fprintf(epsFile,' plastline\n');
else
  fprintf(epsFile,' pline\n');
end
fprintf(epsFile,'/lineY lineY pf sub def\n');
fprintf(epsFile,'pop exit ifelse  loop\n');
```

1.3.31 eusage.m

```
% eusage(text)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003
function eusage(text)
text=sprintf('syntax error of epsTk function\nusage: %s',text);
disp('-------------------------------------------------------------------------------');
disp(text);
disp('-------------------------------------------------------------------------------');
error(' ');
```

1.3.32 exyline.m

```
%exyline ( epsFile,x,y,xData,yData,color,dash,lineWidth)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function exyline ( epsFile,x,y,xData,yData,color,dash,lineWidth)
  if (nargin~=8)
    eusage('exyline(epsFile,x,y,xData,yData,color,dash,lineWidth)');
  end
  [rows colums]=size(xData);
  if rows==1
    nData=2*colums;
    xyData=[xData; yData];
  else
    nData=2*rows;
    xyData=[xData'; yData'];
  end
  xyData=reshape(xyData,1,nData);
  dataStart=1;
  while dataStart<nData
    dataEnd=dataStart+49999;
    if dataEnd>nData
      dataEnd=nData;
    end
    array=sprintf('%1.2f ',xyData(dataStart:dataEnd));
    fprintf(epsFile,'/plotdata[%s] def\n',array);
    if dash>0
      fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
    end
    fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
            color(1),color(2),color(3));
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    fprintf(epsFile,'newpath\n');
    fprintf(epsFile,'0 4 plotdata length 4 sub\n');
    fprintf(epsFile,' dup dup dup plotdata exch get\n');
    fprintf(epsFile,'exch 1 add plotdata exch get moveto\n');
    fprintf(epsFile,'2 add plotdata exch get\n');
    fprintf(epsFile,'exch 3 add plotdata exch get lineto  for\n');
    fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
    fprintf(epsFile,'stroke\n');
```

```
    fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
    fprintf(epsFile,'setrgbcolor\n');
    fprintf(epsFile,'[] 0 setdash\n');
    dataStart=dataEnd+1;
  end
```

1.3.33 exyplot.m

```
%exyplot ( epsFile,x,y,xData,yData,color,dash,lineWidth)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function exyplot ( epsFile,x,y,xData,yData,color,dash,lineWidth)
  if (nargin~=8)
    eusage('exyplot(epsFile,x,y,xData,yData,color,dash,lineWidth)');
  end
  [rows colums]=size(xData);
  if rows==1
    nData=2*colums;
    xyData=[xData; yData];
  else
    nData=2*rows;
    xyData=[xData'; yData'];
  end
  xyData=reshape(xyData,1,nData);
  dataStart=1;
  while dataStart+2<nData
    dataEnd=dataStart+49999;
    if dataEnd>nData
      dataEnd=nData;
    end
    array=sprintf('%1.2f ',xyData(dataStart:dataEnd));
    fprintf(epsFile,'/plotdata[%s] def\n',array);
    if dash>0
      fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
    end
    fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
            color(1),color(2),color(3));
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    fprintf(epsFile,'newpath\n');
    fprintf(epsFile,'plotdata 0 get plotdata 1 get moveto\n');
    fprintf(epsFile,'2 2 plotdata length 2 sub\n');
    fprintf(epsFile,' dup plotdata exch get\n');
    fprintf(epsFile,'exch 1 add plotdata exch get\n');
    fprintf(epsFile,'lineto  for\n');
    fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
    fprintf(epsFile,'stroke\n');
    fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
    fprintf(epsFile,'setrgbcolor\n');
    if dash>0
      fprintf(epsFile,'[] 0 setdash\n');
    end
    dataStart=dataEnd-1;
  end
```

1.3.34 exyplotc.m

```
%exyplotc ( epsFile,x,y,xData,yData,color,dash,lineWidth)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function exyplotc(epsFile,x,y,xData,yData,color,dash,lineWidth)
  if (nargin~=8)
    eusage('exyplotc(epsFile,x,y,xData,yData,color,dash,lineWidth)');
  end
  [rows colums]=size(xData);
  if rows==1
    nData=colums;
    xyData=[xData; yData];
  else
    nData=rows;
    xyData=[xData'; yData'];
  end
  xyData=reshape(xyData,1,2*nData);
  nCurveData=nData-rem(nData-1,3);
  nData=2*nCurveData;
  dataStart=1;
  while dataStart+2<nData
    dataEnd=dataStart+49999;
    if dataEnd>nData
      dataEnd=nData;
    end
    array=sprintf('%1.2f ',xyData(dataStart:dataEnd));
    fprintf(epsFile,'/plotdata[%s] def\n',array);
    if dash>0
      fprintf(epsFile,'[%1.2f %1.2f] 0 setdash\n',dash,dash);
    end
    fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
            color(1),color(2),color(3));
    fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
    fprintf(epsFile,'newpath\n');
    fprintf(epsFile,'plotdata 0 get plotdata 1 get moveto\n');
    fprintf(epsFile,'2 6 plotdata length 2 sub\n');
    fprintf(epsFile,' dup plotdata exch get\n');
    fprintf(epsFile,'exch 1 add dup plotdata exch get\n');
    fprintf(epsFile,'exch 1 add dup plotdata exch get\n');
    fprintf(epsFile,'exch 1 add dup plotdata exch get\n');
    fprintf(epsFile,'exch 1 add dup plotdata exch get\n');
    fprintf(epsFile,'exch 1 add plotdata exch get\n');
    fprintf(epsFile,'curveto  for\n');
    fprintf(epsFile,'%1.2f setlinewidth\n',lineWidth);
    if dash<0
      fprintf(epsFile,'closepath fill\n');
    else
      fprintf(epsFile,'stroke\n');
    end
    fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
    fprintf(epsFile,'setrgbcolor\n');
```

```
    fprintf(epsFile,'[] 0 setdash\n');
    dataStart=dataEnd-1;
  end
```

1.3.35 exyplotf.m

```
%exyplotf ( epsFile,x,y,xData,yData,color)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function exyplotf ( epsFile,x,y,xData,yData,color)
  if (nargin~=6)
    eusage('exyplotf(epsFile,x,y,xData,yData,color)');
  end
  [rows colums]=size(xData);
  if rows==1
    nData=colums;
    xyData=[xData; yData];
  else
    nData=rows;
    xyData=[xData'; yData'];
  end
  if xData(1)==xData(nData) & yData(1)==yData(nData)
    figOpen=0;
  else
    figOpen=1;
  end
  nData=2*nData;
  xyData=reshape(xyData,1,nData);
  array=sprintf('%1.2f ',xyData);
  fprintf(epsFile,'/plotdata[%s] def\n',array);
  fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
          color(1),color(2),color(3));
  fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
  fprintf(epsFile,'newpath\n');
  if figOpen
    fprintf(epsFile,'plotdata 0 get 0 moveto\n');
    fprintf(epsFile,'0 2 plotdata length 2 sub\n');
  else
    fprintf(epsFile,'plotdata 0 get plotdata 1 get moveto\n');
    fprintf(epsFile,'2 2 plotdata length 2 sub\n');
  end
  fprintf(epsFile,' dup plotdata exch get\n');
  fprintf(epsFile,'exch 1 add plotdata exch get\n');
  fprintf(epsFile,'lineto  for\n');
  if figOpen
    fprintf(epsFile,'plotdata plotdata length 2 sub get 0 lineto\n');
  end
  fprintf(epsFile,'closepath\n');
  fprintf(epsFile,'fill\n');
  fprintf(epsFile,'stroke\n');
  fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
  fprintf(epsFile,'setrgbcolor\n');
```

1.3.36 exyploti.m

```
%exyploti( epsFile,x,y,xData,yData,image,colormap)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function exyploti( epsFile,x,y,xData,yData,image,colormap)
  if (nargin~=7)
    eusage('exyploti(epsFile,x,y,xData,yData,image,colormap)');
  end
  [rows colums]=size(xData);
  xmin=min(xData);
  xmax=max(xData);
  ymin=min(yData);
  ymax=max(yData);
  if rows==1
    nData=colums;
    xyData=[xData; yData];
  else
    nData=rows;
    xyData=[xData'; yData'];
  end
  xyData=reshape(xyData,1,2*nData);
  array=sprintf('%1.2f ',xyData);
  fprintf(epsFile,'/plotdata[%s] def\n',array);
  fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
  fprintf(epsFile,'gsave newpath\n');
  fprintf(epsFile,'plotdata 0 get plotdata 1 get moveto\n');
  fprintf(epsFile,'2 2 plotdata length 2 sub\n');
  fprintf(epsFile,' dup plotdata exch get\n');
  fprintf(epsFile,'exch 1 add plotdata exch get\n');
  fprintf(epsFile,'lineto  for\n');
  fprintf(epsFile,'closepath clip\n');
  eimagexy(epsFile,image,colormap,xmin,ymin,xmax-xmin,ymax-ymin);
  fprintf(epsFile,'grestore %1.2f %1.2f translate\n',-x,-y);
```

1.3.37 exyplots.m

```
%exyplots( epsFile,x,y,xData,yData,dash,color)
% written by Stefan Mueller stefan.mueller@fgan.de (C) 2003

function exyplots( epsFile,x,y,xData,yData,xScale,yScale,angle,dash,color)
  if (nargin~=10)
    eusage('exyplots(epsFile,x,y,xData,yData,xScale,yScale,angle,dash,color)');
  end
  [rows colums]=size(xData);
  if rows==1
    nData=colums;
    xyData=[xData; yData; xScale; yScale; angle];
  else
    nData=rows;
    xyData=[xData'; yData'; xScale'; yScale'; angle'];
  end
  xyData=reshape(xyData,1,5*nData);
```

```
array=sprintf('%1.2f ',xyData);
fprintf(epsFile,'/pdata[%s] def\n',array);
fprintf(epsFile,'currentrgbcolor %1.2f %1.2f %1.2f setrgbcolor\n',...
        color(1),color(2),color(3));
fprintf(epsFile,'%1.2f %1.2f translate\n',x,y);
fprintf(epsFile,'0 5 pdata length 5 sub\n');
fprintf(epsFile,' gsave dup dup dup dup pdata exch get\n');
fprintf(epsFile,'exch 1 add pdata exch get translate\n');
fprintf(epsFile,'4 add pdata exch get rotate\n');
fprintf(epsFile,'2 add pdata exch get exch 3 add pdata exch get scale\n');
fprintf(epsFile,' %s grestore for\n',dash);
fprintf(epsFile,'%1.2f %1.2f translate\n',-x,-y);
fprintf(epsFile,'setrgbcolor\n');
```

Chapter 2

Examples

2.1 Demo Files

2.1.1 edemo1.m

```
x1=[-0.5:0.01:2*pi];
x2=[0:0.2:2*pi];


eopen('demo1.eps')                      % open eps-file and write eps-head

% fill area
eplot(x1,cos(x1)*0.2,'cosfill',-1,[0.8 0.8 0.2])

% plot single lines
eplot([x2;x2],[cos(x2)*0.5;sin(x2)],'diff',0,[0.7 0 0.7],0.5)

% plot red solid line
eplot(x1,sin(x1),'sin',0,[1 0 0],2)

% plot blue dash line
eplot(x1,cos(x1)*0.5,'cosine',2,[0 0 1],1)

% plot errorbar
error=rand(size(x2))/3;
[xm ym]=eerrbar(x2,sin(x2),error);
eplot(xm,ym,'error',0,[0 0.4 0],0.5);

eclose                                  % close eps-file
eview                                   % start ghostview with eps-file
```

2.1.2 edemo10.m

```
eopen('demo10.eps');
eglobpar
[im icm]=eshadois;
data=[115 124 123 50 149 189];
n=length(data);
cm=ecolors(3,n);
```

```
ePolarPlotAreaRadMin=0;
ePolarPlotAreaAngEnd=360;
%data=data/sum(data)
shadowColor=[0.2 0.2 0.2];
ePlotLegendPos=[ePolarPlotAreaRadMax*2.2 30];
legend=['epstk ';'tool 2';'tool 3';'tool 4';'tool 5';'tool 6'];

offset=10;

%shadow pie
epie(data(1),'','',-1,offset,shadowColor);
for i=2:n
  epie(data(i),'','',-1,0,shadowColor);
end
epie;

%color pie
ePolarPlotAreaCenterPos=ePolarPlotAreaCenterPos-[1 1];
eAxesValueFontSize=5;
epie(data(1),'',legend(1,:),im,offset,icm);
for i=2:n
  epie(data(i),sprintf('%d files',data(i)),legend(i,:),-1);
end
epie;

%frame pie
ePlotTitleText='Pie Plot';
epie(data(1),'','',0,offset);
for i=2:n
  epie(data(i));
end
angles=epie;

%label
labelColor=[0 0 0.5];
labelAngle=(angles(1,1)+angles(1,2)/2)*pi/180;
p1=ePolarPlotAreaCenterPos+[cos(labelAngle) sin(labelAngle)]*...
  (offset+ePolarPlotAreaRadMax*3/4);
p2=ePolarPlotAreaCenterPos+[cos(labelAngle) sin(labelAngle)]*...
  (offset+ePolarPlotAreaRadMax+10);
p3=[ePolarPlotAreaCenterPos(1)+ePolarPlotAreaRadMax+offset+5 p2(2)];
pline=[p1;p2;p3];
epline(pline(:,1),pline(:,2),eLineWidth,0,labelColor);
edsymbol('dot','frect.psd',0.3,0.3,0,0,0,labelColor);
esymbol(p1(1),p1(2),'dot');
etext('my files',p3(1),p3(2),6,4,3);

ePlotAreaHeight=50;
ePlotAreaWidth=50;
eXAxisNorthVisible=0;
eYAxisEastVisible=0;
[xb yb]=ebar(data,0);
```

```
ePlotTitleText='Bar Plot 1';
ePlotAreaPos=[20 20];
eYAxisWestScale=[0 0 200];
eXAxisSouthScale=[0.5 0 6.5];
eYAxisWestLabelText='files';
eXAxisSouthLabelText='tool';

eplot(xb+0.05,yb+1,'',-1,[0 0 0])
eplot(xb,yb,'',-1,[0.9 0.9 0])
eplot(xb,yb,'',0,[0 0 0])
eplot

ePlotTitleText='Bar Plot 2';
ePlotAreaPos=[110 20];
eXAxisSouthScale=[0 0 200];
eYAxisWestScale=[0.5 0 6.5];
eXAxisSouthLabelText='files';
eYAxisWestLabelText='tool';
[xb yb]=ebar(data,0);
eplot(yb+1,xb+0.05,'',-1,[0 0 0])
eplot(yb,xb,'',-1,[0.9 0.0 0])
eplot(yb,xb,'',0,[0 0 0])
eplot


eclose
eview
```

2.1.3 edemo11.m

```
bgcolor=[1 1 0.8];
eopen('demo11.eps');
eglobpar

% head
%eWinGridVisible=1;
w=eWinWidth;
fs=20;
x=0;
h=fs/2;y=eWinHeight-h;
text='epsTk News';
eframe(x,y,w,h,0,-1,[0.0 0.5 0.0]); %background
etxtbox(text,x,y,w,h,[fs fs 10],0,1,0,[0 0 0],[1 1]); %shadow text
etxtbox(text,x,y,w,h,[fs],0,1,0,bgcolor); %main text
etext('January 2003',160,y+h/2,4,3,1,0,[1 1 0]); %text
h=h*0.8;y=y-h;
eframe(x,y,w,h,0,-1,bgcolor); %background
etxtbox(text,x,y,w,h,[fs fs 10],0,1,0,[0 0 0],[1 1+fs/2]); %shadow text
etxtbox(text,x,y,w,h,fs,0,1,0,[1 0 0],[0 fs/2]); % main text
yHead=y;
eframe(0,0,eWinWidth,yHead,0,-1,bgcolor); %background of ellipse
eellipse(x+20,y+h,40,10,0,-1,[1 0 0],15); %fill ellipse
eellipse(x+20,y+h,40,10,0,0,[0 0 0],15);  %border of ellipse
```

```
etext('version 2.0',x+20,y+h,4,3,1,15,[1 1 0]); %text

% text
eTextBoxSpaceWest=1.5;
eTextBoxSpaceEast=1.5;
eTextBoxSpaceNorth=1.5;
w=eWinWidth/3;
h=17;y=y-h;
fs=6;
text='Econometrics with Octave';
etxtbox(text,x,y,w,h,[fs*1.5 fs 10],0,3,0,[0 0 0]); %title

rText=etxtread([ePath 'octave.asc']); %get text from file
h=y;y=y-h;
fs=3;
endKey='Version 1.0';
pos=findstr(rText,endKey);
text=rText(1:pos(1)-1);
rText=rText(pos(1):length(rText));
etxtbox(text,x,y,w,h,fs,2,1,0,[0 0 0]); %1. column of text

x=x+w;
h=143;y=yHead-h;
endKey='octave-epstk';
pos=findstr(rText,endKey);
text=rText(1:pos(1)-1);
rText=rText(pos(1):length(rText));
etxtbox(text,x,y,w,h,fs,2,1,0,[0 0 0]); %2. column of text

[im cm]=eshadois;
h=w;y=y-h;
eframe(x,y,w,w,0.5,im,cm); % insert epstk logo
eframe(x,y,w,w,0.5,0,[0 0 0]); %frame

h=50;y=y-h;
text=rText;
etxtbox(text,x,y,w,h,fs,2,3,0,[1 0 0]); % last part of text

% filelist
fs=6;
x=x+w;
h=eTextBoxSpaceNorth+fs+eTextBoxSpaceSouth;y=yHead-h;
text='M-Files of epsTk';
etxtbox(text,x,y,w,h,fs,0,3,0,[0 0 0]); %title
text=etxtread([ePath 'mFileList']);
[tPos n]=etxtlpos(text);
nRows=ceil(n/3);
fs=3;
h=eTextBoxSpaceNorth+nRows*fs+eTextBoxSpaceSouth;y=y-h;
cx=x;
cText=text(tPos(1,1):tPos(nRows,2));
etxtbox(cText,cx,y,w/3,h,fs,2,1,0,[0 0 1]); %1. column
cx=x+w/3;
cText=text(tPos(nRows+1,1):tPos(2*nRows,2));
```

```
etxtbox(cText,cx,y,w/3,h,fs,2,1,0,[0 0 1]); %2. column
cx=x+2*w/3;
cText=text(tPos(2*nRows+1,1):tPos(n,2));
etxtbox(cText,cx,y,w/3,h,fs,2,1,0,[0 0 1]); %3. column

% feature list
y=y-fs;
fs=6;
h=eTextBoxSpaceNorth+fs+eTextBoxSpaceSouth;y=y-h;
text='New feature of 2.0';
etxtbox(text,x,y,w,h,fs,0,3,0,[0 0 0]); %title

fs=5;
font=1;
dotSize=fs/2;
iDotX=x+eTextBoxSpaceWest+dotSize/2;
iDotShift=eTextBoxSpaceNorth+fs/2;
iTxtX=iDotX+dotSize/2;
iw=w-iTxtX+x;
edsymbol('dot','fring.psd',dotSize/10,dotSize/10,0,0,0,[0 0.3 0]);

rows=1;h=eTextBoxSpaceNorth+rows*fs+eTextBoxSpaceSouth;y=y-h;
esymbol(iDotX,y+h-iDotShift,'dot');
text='log scaled axes';
etxtbox(text,iTxtX,y,iw,h,fs,1,font,0,[0 0 0]); %feature list item

rows=1;h=eTextBoxSpaceNorth+rows*fs+eTextBoxSpaceSouth;y=y-h;
esymbol(iDotX,y+h-iDotShift,'dot');
text='return long tic positions of axes';
etxtbox(text,iTxtX,y,iw,h,fs,1,font,0,[0 0 0]); %feature list item

rows=2;h=eTextBoxSpaceNorth+rows*fs+eTextBoxSpaceSouth;y=y-h;
esymbol(iDotX,y+h-iDotShift,'dot');
text='new image functions e.g. ebright, econtra, ...';
etxtbox(text,iTxtX,y,iw,h,fs,1,font,0,[0 0 0]); %feature list item

rows=2;h=eTextBoxSpaceNorth+rows*fs+eTextBoxSpaceSouth;y=y-h;
esymbol(iDotX,y+h-iDotShift,'dot');
text='text functions etxtbox, etxtread, etxtlpos';
etxtbox(text,iTxtX,y,iw,h,fs,1,font,0,[0 0 0]); %feature list item

rows=1;h=eTextBoxSpaceNorth+rows*fs+eTextBoxSpaceSouth;y=y-h;
esymbol(iDotX,y+h-iDotShift,'dot');
text='pie plot and error bars';
etxtbox(text,iTxtX,y,iw,h,fs,1,font,0,[0 0 0]); %feature list item

rows=2;h=eTextBoxSpaceNorth+rows*fs+eTextBoxSpaceSouth;y=y-h;
esymbol(iDotX,y+h-iDotShift,'dot');
text='linear interpolation with elineip and efillmat';
etxtbox(text,iTxtX,y,iw,h,fs,1,font,0,[0 0 0]); %feature list item

rows=1;h=eTextBoxSpaceNorth+rows*fs+eTextBoxSpaceSouth;y=y-h;
esymbol(iDotX,y+h-iDotShift,'dot');
text='frames, circles, ellipses';
```

```
etxtbox(text,iTxtX,y,iw,h,fs,1,font,0,[0 0 0]); %feature list item

rows=2;h=eTextBoxSpaceNorth+rows*fs+eTextBoxSpaceSouth;y=y-h;
esymbol(iDotX,y+h-iDotShift,'dot');
text='ASCII85 bitmap code, generates small EPS-files';
etxtbox(text,iTxtX,y,iw,h,fs,1,font,0,[0 0 0]); %feature list item

rows=1;h=eTextBoxSpaceNorth+rows*fs+eTextBoxSpaceSouth;y=y-h;
esymbol(iDotX,y+h-iDotShift,'dot');
text='JPEG-file integration';
etxtbox(text,iTxtX,y,iw,h,fs,1,font,0,[0 0 0]); %feature list item

rows=1;h=eTextBoxSpaceNorth+rows*fs+eTextBoxSpaceSouth;y=y-h;
esymbol(iDotX,y+h-iDotShift,'dot');
text='a lot of bugs removed';
etxtbox(text,iTxtX,y,iw,h,fs,1,font,0,[0 0 0]); %feature list item
eclose
eview
```

2.1.4 edemo12.m

```
eopen('demo12.eps')                      % open eps-file and write eps-head

eglobpar                                 % get access to global parameters
etext('EPS Import',5,5,10,1)

esubeps(3,3,1,1,'demo10.eps');
esubeps(3,3,1,2,'demo3.eps');
esubeps(3,3,1,3,'demo4.eps');
esubeps(3,3,2,1,'demo5.eps');
esubeps(3,3,2,2,'demo6.eps');
esubeps(3,3,2,3,'demo11.eps');
einseps(0,20,'demo9.eps',0.2,0.2,-10);
einseps(60,20,[ePath 'epstkbc'],0.6,0.6,90); % file generated by GNU barcode
esubeps(3,3,3,2,'demo2.eps');
esubeps(3,3,3,3,'demo8.eps');
eclose                                   % close eps-file
eview                                    % start ghostview with eps-file
```

2.1.5 edemo13.m

```
% create a image list with ejpglist()
eglobpar
einit

% read  eps file list
list=etxtread([ePath 'epsFileList']);
jList=' ';
[lPos n]=etxtlpos(list);

% make jpeg-files and a list of jpeg-files
```

```
for i=1:n
  eFileName=list(lPos(i,1):lPos(i,2));
  if exist(eFileName)
    jFileName=ebitmap(1,100); % eps-file to jpeg-file
    jList=[jList jFileName eTextLimitPara];
  end
end
jList=jList(2:size(jList,2));
etxtwrit(jList,'demo_jFileList');

% create image list by ejpglist()
ejpglist('demo_jFileList',[50 40],0,'demo13');
eview
```

2.1.6 edemo14.m

```
% print calendar of a year
% written by Coletta Schumacher and Stefan Mueller
year=2003;
%              day   month  textIndex textColumn textColor    backgroundColor
myHolidays=   [
                24    1     21        2          1.0 1.0 1.0  0.9 0.2 0.2;
                 9    8     31        2          1.0 1.0 1.0  0.9 0.2 0.2;
                 8   12     41        2          1.0 1.0 1.0  0.9 0.2 0.2;
                21    6     50        2          1.0 1.0 1.0  0.9 0.2 0.2;
                30    5     80        1          0.0 0.0 0.0  0.9 0.8 0.5;
                20    6     80        1          0.0 0.0 0.0  0.9 0.8 0.5;
                29   12     80        1          0.0 0.0 0.0  0.9 0.8 0.5;
                30   12     80        1          0.0 0.0 0.0  0.9 0.8 0.5;
              ];
myHoliText=   [
               '021gau          ';
               '031ase          ';
               '041bra          ';
               '050mwi          ';
               '080vorgearb.    ';
              ];
%variable holydays
%              day   month  textIndex textColumn textColor    backgroundColor
varHolidays=  [
                 0    0      1        1          1.0 1.0 1.0  0.2 0.7 0.2;
                 1    0      1        1          1.0 1.0 1.0  0.2 0.7 0.2;
                47    0      3        1          1.0 1.0 1.0  0.2 0.7 0.2;
                49    0      4        1          1.0 1.0 1.0  0.2 0.7 0.2;
                50    0      5        1          1.0 1.0 1.0  0.2 0.7 0.2;
                88    0      6        1          1.0 1.0 1.0  0.2 0.7 0.2;
                98    0      7        1          1.0 1.0 1.0  0.2 0.7 0.2;
                99    0      7        1          1.0 1.0 1.0  0.2 0.7 0.2;
               109    0      9        1          1.0 1.0 1.0  0.2 0.7 0.2;
              ];
varHoliText=  [
               '001Fastnacht    ';
               '003Karfreitag   ';
```

```
               '004Ostern       ';
               '005Ostern       ';
               '006Chr.Himmelf .';
               '007Pfingsten    ';
               '009Fronleich.   ';
              ];
% fixed holidays
%               day   month  textIndex textColumn textColor    backgroundColor
fixHolidays=[
                 1    1     10        1          1.0 1.0 1.0  0.2 0.7 0.2;
                 1    5     11        1          1.0 1.0 1.0  0.2 0.7 0.2;
                 3   10     12        1          1.0 1.0 1.0  0.2 0.7 0.2;
                 1   11     13        1          1.0 1.0 1.0  0.2 0.7 0.2;
                25   12     14        1          1.0 1.0 1.0  0.2 0.7 0.2;
                26   12     14        1          1.0 1.0 1.0  0.2 0.7 0.2;
                31   12     16        1          1.0 1.0 1.0  0.2 0.7 0.2;
              ];
fixHoliText=[
               '010Neujahr      ';
               '0111.Mai        ';
               '012Dt. Einheit  ';
               '013Allerheiligen';
               '014Weihnachten  ';
               '016Silvester    ';
              ];
weekday= ['Mo'; 'Di'; 'Mi'; 'Do'; 'Fr'; 'Sa'; 'So'];
saBgColor=[0.8 0.8 1.0];
suBgColor=[0.7 0.7 1.0];
monthT=['Januar   '; 'Februar  '; 'M\\344rz '; 'April    ';
        'Mai      '; 'Juni     '; 'Juli     '; 'August   ';
        'September'; 'Oktober  '; 'November '; 'Dezember '];
nDaysOfM = [31 28 31 30 31 30 31 31 30 31 30 31];
if ~rem (year,4),nDaysOfM(2)=29;end
nDaysOfY=sum(nDaysOfM);

% sundays of carneval until 2019
cSundays=[ 5 3;25 2;10 2; 2 3;22 2; 6 2;26 2;18 2; 3 2;22 2;...
          14 2; 6 3;19 2;10 2; 2 3;15 2; 7 2;26 2;11 2; 3 3];
cSunday=year-2000+1;
cDay=cSundays(cSunday,1);
cMonth=cSundays(cSunday,2);
dayOfY=rem(cDay+sum(nDaysOfM(1:cMonth-1)),7);
if dayOfY
  firstDayOfY=7-dayOfY+1;
else
  firstDayOfY=1;
end

% variable holidays   day of the
for k=1:size(varHolidays,1)
  varDay=cDay+varHolidays(k,1);
  for i=0:4
    if varDay>nDaysOfM(cMonth+i)
      varDay=varDay-nDaysOfM(cMonth+i);
```

```
    else
      break
    end
  end
  varHolidays(k,1)=varDay;
  varHolidays(k,2)=cMonth+i;
end

holidays=[myHolidays;varHolidays;fixHolidays];
holitext=[myHoliText;varHoliText;fixHoliText];
[nTextRows nTextCols]=size(holitext);
holiIndex=holidays(:,3);
for i=1:nTextRows
  index=str2num(holitext(i,1:3));
  fresult=find(holiIndex==index);
  holidays(fresult,3)=i*ones(size(fresult,1),1);
end
holitext=holitext(:,4:nTextCols);

% draw table
eopen('demo14.eps');
eglobpar
eWinGridVisible=0;
dayOfY=0;
[calX calY]=etabdef(32,6,0,130,180,120);
for month=1:12
  if month==7
    etabgrid(calX,calY);
    [calX calY]=etabdef(32,6,0,0,180,120);
  end
  tabCol=rem(month-1,6)+1;
  etabtext(calX,calY,1,tabCol,monthT(month,:),0,3,100,[1 1 1],[0.5 0.5 0.8]);
  offset=3.8;
  [dayX dayY]=etabdef(32,1,calX(tabCol,1)+offset,calY(32,1),1,120);
  [wdX wdY]=etabdef(32,1,calX(tabCol,1)+0.9*offset,calY(32,1),1,120);
  for dayOfM=1:nDaysOfM(month)
    dayOfW=rem(firstDayOfY-1+dayOfY,7)+1;
    dayOfY=dayOfY+1;
    if dayOfW==6
      etabtext(calX,calY,dayOfM+1,tabCol,'',1,1,100,[1 1 1],saBgColor);
    elseif dayOfW==7
      etabtext(calX,calY,dayOfM+1,tabCol,'',1,1,100,[1 1 1],suBgColor);
    end
    etabtext(dayX,dayY,dayOfM+1,1,sprintf('%d',dayOfM),-1);
    etabtext(wdX,wdY,dayOfM+1,1,sprintf('%s',weekday(dayOfW,:)),1,3,70);
  end
  offset=8;
  [nX nY]=etabdef(32,2,calX(tabCol,1)+offset,calY(32,1),...
                  calX(tabCol,2)-offset,120,[3 1]);
  for notes=find(holidays(:,2)==month)'
    if holidays(notes,4)==1
      etabtext(nX,nY,holidays(notes,1)+1,1,...
               sprintf('%s',holitext(holidays(notes,3),:)),...
               1,1,100,holidays(notes,5:7),holidays(notes,8:10));
```

```
    elseif holidays(notes,4)==2
      etabtext(nX,nY,holidays(notes,1)+1,2,...
               sprintf('%s',holitext(holidays(notes,3),:)),...
               1,1,80,holidays(notes,5:7),holidays(notes,8:10));
    end
  end
end
etabgrid(calX,calY);
etext(sprintf('Jahr %d',year),90,122,8,0,3);
eclose;
eview;
```

2.1.7 edemo2.m

```
% standard plot
eopen('demo2.eps')                      % open eps-file and write eps-head
eglobpar                                % get access to global parameters
eXAxisSouthLabelText='Sector [No]';     % set South Label of XAxis
eXAxisSouthScaleType=1;                 % set classes scaling
eYAxisWestLabelText='Range [km]';       % set West Label of YAxis
eYAxisEastLabelText='Outlook';          % set East Label of YAxis
eXAxisNorthVisible=0;                   % switch North-XAxis off
eYAxisEastValueVisible=0;               % switch East-YAxis Values off
eXGridVisible=1;                        % switch x-Grid on
eYGridVisible=1;                        % switch y-Grid on

% background
[im cm]=eimgread; % read standard image
cm=econtra(cm,200,[1 2]);  % change contrast of image
cm=ebright(cm,60);  % change brightness of image
im=eimgrot(im,90);  % rotate image
eframe(0,0,eWinWidth,eWinHeight,0,im,cm,0,10); % print background image

% frame
eframe(0,0,eWinWidth,eWinHeight,0.5,0,[0 0 0],0,10); % draw frame around window
etext('demo only',eWinWidth/2,eWinHeight/2,40,0,1,...   %print demo text
     atan(eWinHeight/eWinWidth)*180/pi,[0.8 0.8 0.8])
eXAxisSouthScale=[0 0 6];  %set scale x-axis
eYAxisWestScale=[0 10 60]; %set scale y-axis

etitle('Standard Plot',25,9,[0 0.3 0])         % print title

% 1. plot lines
x=0:0.1:2*pi;
eplot(x,sin(x)*15+30,'street',0,[0.8 0.8 0],3) %solid line
eplot(x,sin(x)*15+33,'power line',5,[1 0 1]) % dash plot
eplot(x,sin(x)*15+36,'railway',0,[0.3 0.3 0.4],1) % solid line

% 2. plot symbols
x=0:0.5:2*pi;
edsymbol('spire','spire.psd',0.3,0.3,0,0,90) % define symbol with name 'spire'
eplot(x,cos(x)*10+20,'tree','spire',[0 0.7 0]) % plot trees
```

```
% 3. plot area
lake=[1 30;1.3 35;1.5 32;2 40;2.5 25;1.6 15;1.4 17;1.2 24;1 30];
eplot(lake(:,1),lake(:,2),'water',-1,[0.5 0.5 1]); % filled area,dash<0
eplot(lake(:,1),lake(:,2),'water limit',0,[0 0 0]); % solid line around the area

% 4. plot image
x1=0:pi/3:pi;
x1=x1';
x2=flipud(x1);
girl=[x1+2.5 sin(x1)*10+50;x2+2.5 -sin(x2)*10+50];
im=eimgread;
eplot(girl(:,1),girl(:,2),'girl',im,-1); % fill area with image
eplot(girl(:,1),girl(:,2),'',0,[0 0.4 0]); % solid line around the image

% 5. plot bars
x=0.5:1:5.5;
[xb yb]=ebar(sin(x)*8+9,0,1,3);     % 1. bars
eplot(xb,yb,'forest',-1,[0.5 0.7 0])
eplot(xb,yb,'',0,[0 0 0])
[xb yb]=ebar(cos(x)*8+5,0,2,3);     % 2. bars
eplot(xb,yb,'town',-1,[0.8 0 0])
eplot(xb,yb,'',0,[0 0 0])
[xb yb]=ebar(cos(x)*8+9,0,3,3);     % 3. bars
eplot(xb,yb,'sea',-1,[0 0 1])
eplot(xb,yb,'',0,[0 0 0])
eplot

% write parameters
eParamPos=[80,70];
eParamFontSize=5;
eparam('Altitude','232 m')
eparam('Power','100 W')
valuePos=eYAxisEastValuePos;
posNo=1;
etext('Start of EastAxis',valuePos(posNo,1),valuePos(posNo,2),4,4);
posNo=2;
etext('2. Value',valuePos(posNo,1),valuePos(posNo,2),4,4);
posNo=7;
etext('End of EastAxis',valuePos(posNo,1),valuePos(posNo,2),4,4);

% axis
eYAxisEastScaleType=0;                  % set linear scaling
eYAxisEastValueVisible=0;               % no values visible
angle=-55;
eaxis(80,30,100,'e',[0 1 10],angle,[1 0 0]);
valuePos=eYAxisEastValuePos;
posNo=1;
etext('Start',valuePos(posNo,1),valuePos(posNo,2),4,1,1,angle,[1 0 0]);
posNo=4;
etext('3 cm',valuePos(posNo,1),valuePos(posNo,2),4,4,1,angle,[0 0.5 0]);
etext(' or 30 mm',0,0,4,4,1,angle,[0 0 1]);
posNo=6;
etext('5 cm',valuePos(posNo,1),valuePos(posNo,2),4,4,1,angle,[0.3 0 1]);
posNo=11;
```

```
etext('10 cm',valuePos(posNo,1),valuePos(posNo,2),4,4,1,angle,[0 0 1]);

eclose                                  % close ps output
eview                                   % start ghostview with eps-file
```

2.1.8 edemo3.m

```
eopen('demo3.eps');
eglobpar;                       % get access to global parameters
eWinGridVisible=1;
esavpar;    % save default parameter

%title
etext('Multi Plot',50,eWinHeight-15,8,0,1)

% set tics like Matlab
eAxesValueSpace=3;
eAxesTicLongLength=-1;
eAxesTicShortLength=-0.5;

% simple plot
eAxesColor=[0 0.4 0];
ePlotAreaPos=[20 150];
ePlotAreaHeight=50;
ePlotAreaWidth=50;
eYAxisWestLabelText='f(x)';
eXAxisSouthLabelText='x [rad]';
etitle('Simple Plot (like MATLAB(R))',10,6,[0 0 0.5]);
eYAxisWestScaleType=2;
eYGridVisible=1;
xData=0:0.1:2*pi;
eYAxisEastValueFormat=2;
eplot(xData,exp(sin(xData).*xData),'exp(sin(x)*x)',0,[1 0 0]);
eplot(xData,exp(cos(xData).*xData),'exp(cos(x)*x)',2,[0 0 1]);
eplot;

% polar plot
erespar; % set default parameter
eAxesColor=[1 0 1];
ePolarPlotAreaCenterPos=[135 200];
ePolarPlotAreaRadMin=10;
ePolarPlotAreaRadMax=25;
ePolarAxisRadValueVisible=0;
ePlotTitleDistance=15;
ePlotTitleText='Polar Plot';
xData=0:0.01:2*pi;
ePolarAxisRadScale=[0 0.3 1];
epolar(xData,cos(xData*7)*0.2,'cosine filled',-1,[1 1 0]);
epolar(xData,sin(xData*4),'sine',0,[1 0 0]);
epolar(xData,cos(xData*5),'cosine',2,[0 0 1],1);
epolar;
angles=ePolarAxisAngValueAngle*pi/180;
dis=11;
```

```
lPos=[cos(angles) sin(angles)]*(ePolarPlotAreaRadMax+dis);
lPos=[lPos(:,1)+ePolarPlotAreaCenterPos(1)...
      lPos(:,2)+ePolarPlotAreaCenterPos(2)];
i=1;
etext('Start',lPos(i,1),lPos(i,2),4,4,1,0,[0 0.8 0]);
i=2;
etext('second',lPos(i,1),lPos(i,2),4,4,1,ePolarAxisAngValueAngle(i),[0 1 0]);
i=12;
etext('12.value',lPos(i,1),lPos(i,2),4,2,1,0,[1 0.2 0]);
i=18;
ellipseW=2*(dis-eAxesTicLongLength);
eellipse(lPos(i,1),lPos(i,2),ellipseW,ellipseW,0,-1,[1 1 1]); % cover value
etext('End',lPos(i,1),lPos(i,2),4,3,1,0,[0.8 0 0]);

%  cross axes plot
erespar; % set default parameter
eAxesColor=[0 0.4 0];
ePlotAreaPos=[110 80];
ePlotAreaHeight=50;
ePlotAreaWidth=50;
ePlotTitleDistance=5;
ePlotTitleText='Cross Axes';
eAxesCrossOrigin=2;  % cross axes with arrows
xData=-3.2:0.1:6.2;
eplot(xData,sin(xData),'',0,[1 0 0]);
eplot(xData,cos(xData),'',0,[0 0 1]);
eplot

% random image
erespar; % set default parameter
ePlotAreaPos=[20 50];
ePlotAreaHeight=50;
ePlotAreaWidth=50;
eYAxisWestLabelText='Y-Values';
eXAxisSouthLabelText='X-Values';
eImageLegendPos=[0 -5];
ePlotTitleDistance=5;
ePlotTitleText='Linear interpolation';
eXAxisNorthVisible=0;
eYAxisEastVisible=0;
%matrix=rand(30,30);
matrix=efillmat([1 -1 -1 1],[1 1 -1 -1],[100 1 100 1],0.02,0.02);
eImageLegendScaleType=2;
eXAxisSouthScale=[-1 0 1];  %set scale x-axis
eYAxisWestScale=[-1 0 1]; %set scale y-axis
eimagesc(matrix,ecolors(3),'e'); % print scaled image

% photo
erespar; % set default parameter
ePlotAreaPos=[110 10];
ePlotAreaHeight=50;
ePlotAreaWidth=50;
ePlotTitleDistance=5;
ePlotTitleText='photo';
```

```
eImageLegendVisible=0;
eimage([ePath 'default.jpg']);  % print a JPEG-file

% shadow photo
erespar; % set default parameter
ePlotAreaPos=[80 10];
ePlotAreaHeight=25;
ePlotAreaWidth=25;
ePlotTitleDistance=5;
ePlotTitleText='shadow photo';
[photo colormap]=eppmread; % read default image
eshadoi(photo); % print shadow image

eclose;
eview;
```

2.1.9 edemo4.m

```
eopen('demo4.eps');
eglobpar;

[img cm]=eppmread([ePath 'defMap.ppm']);
n=size(img,1);
delta=10*pi/(n-1);
x=-5*pi:delta:5*pi;
[a b]=meshgrid(x,x);
R=sqrt(a.^2+b.^2) + eps;
matrix=100*(sin(R)./R+0.3);

% scaled shadow image
ePlotAreaPos=[30 140];
ePlotAreaHeight=80;
ePlotAreaWidth=80;
ePlotTitleDistance=15;
eImageLegendScaleType=2;
ePlotTitleText='Scaled Shadow Image';
colorMap=[1 1 0;1 0 0;0 1 0;0 0 1;1 0 1];
eshadois(matrix,colorMap,'e');
etext('        (log scale)',eImageLegendValuePos(1,1),...
  eImageLegendValuePos(1,2),4,4,1);


% shadow image
ePlotAreaPos=[10 30];
ePlotAreaHeight=75;
ePlotAreaWidth=75;
ePlotTitleDistance=5;
ePlotTitleText='Shadow Image';
eshadoi(matrix);

% mixed shadow image
ePlotAreaPos=[90 30];
ePlotAreaHeight=75;
```

```
ePlotAreaWidth=75;
ePlotTitleDistance=5;
ePlotTitleText='Mixed Shadow Image';
eshadoix(matrix,img,cm);

eclose;
eview;
```

2.1.10 edemo5.m

```
x=-2:.1:2;
y=-2:.1:2;
[X,Y]=meshgrid(x,y);
matrix=X.*exp(-X.^2-Y.^2);

% contour plot
eopen('demo5a.eps',0,180,140);
eglobpar;
ePlotTitleText='Contour Plot';
ePlotTitleDistance=15;
ePlotAreaPos=[20 20];
eXAxisSouthLabelText='x - Axis';
eYAxisWestLabelText='y - Axis';
eImageLegendLabelText='z - Color Legend';
eContourValueVisible=1;
eaxes([-2 0 2],[-2 0 2]);
eimagesc(matrix,ecolors(3),'e');
etext('        (1. value)',eImageLegendValuePos(1,1),...
  eImageLegendValuePos(1,2),4,4,2);
econtour(matrix,[-0.5 0.05 0.5],0,[1 1 1;0 0 0;0 0 0]);
eclose(1,0);


% quiver plot
eopen('demo5b.eps',0,180,140);
eglobpar;
ePlotTitleText='Quiver Plot';
ePlotTitleDistance=15;
ePlotAreaPos=[20 20];
eXAxisSouthLabelText='x - Axis';
eYAxisWestLabelText='y - Axis';
eImageLegendLabelText='z - Color Legend';
eaxes([-2 0 2],[-2 0 2]);
eimagesc(matrix,ecolors(3),'e');

% sw
x=-2.1:.2:-0.1;
y=-2.1:.2:-0.1;
[X,Y]=meshgrid(x,y);
qmatrix=X.*exp(-X.^2-Y.^2);
[dx dy]=egradient(qmatrix,.2,.2);
equiver(X,Y,dx,dy,[1 1 1]);
```

```
% nw
x=-2.1:.2:-0.1;
y=0.1:.2:2.1;
[X,Y]=meshgrid(x,y);
qmatrix=X.*exp(-X.^2-Y.^2);
[dx dy]=egradient(qmatrix,.2,.2);
edsymbol('spire','spire.psd');
equiver(X,Y,dx,dy,[1 1 1],'spire');

% ne
x=0.1:.2:2.1;
y=0.1:.2:2.1;
[X,Y]=meshgrid(x,y);
qmatrix=X.*exp(-X.^2-Y.^2);
[dx dy]=egradient(qmatrix,.2,.2);
edsymbol('needle','needle.psd');
equiver(X,Y,dx,dy,[0 0 0],'needle');

% se
x=0.1:.2:2.1;
y=-2.1:.2:-0.1;
[X,Y]=meshgrid(x,y);
qmatrix=X.*exp(-X.^2-Y.^2);
[dx dy]=egradient(qmatrix,.2,.2);
edsymbol('ftria','ftria.psd',1,0.4);
equiver(X,Y,dx,dy,[0 0 0],'ftria');

eclose(1,0);

% quiver and contour
eopen('demo5.eps')
esubeps(2,1,1,1,'demo5a.eps');
esubeps(2,1,2,1,'demo5b.eps');
eclose
eview
```

2.1.11 edemo6.m

```
eopen('demo6.eps');
eglobpar;

% scaled polar image plot
x=-3*pi:0.1:3*pi;
[a b]=meshgrid(x,x);
R=sqrt(a.^2+b.^2) + eps;
matrix=sin(R)./R+10;
ePlotTitleText='Scaled Polar Image';
ePlotTitleDistance=15;
ePolarPlotAreaCenterPos=[70,170];
ePolarPlotAreaRadMax=40;
ePolarPlotAreaAngStart=-40;
ePolarPlotAreaAngEnd=220;
ePolarAxisRadVisible=3;
```

```
ePolarAxisRadValueVisible=1;
ePolarRadiusGridColor=[1 1 1];
ePolarAngleGridColor=[1 0 0];
eImageLegendLabelText='Color Legend';
matrix=epolaris(matrix,ecolors(2),'e');
etext('Max Rad',ePolarAxisRadValuePos(1,3),ePolarAxisRadValuePos(1,4),...
  4,4,1,ePolarPlotAreaAngEnd+90);
etext('3. Value',ePolarAxisRadValuePos(3,3),ePolarAxisRadValuePos(3,4),...
  4,4,1,ePolarPlotAreaAngEnd+90);
etext('Min Rad',ePolarAxisRadValuePos(6,3),ePolarAxisRadValuePos(6,4),...
  4,4,1,ePolarPlotAreaAngEnd+90);
etext('      (1. Value)',eImageLegendValuePos(1,1),eImageLegendValuePos(1,2),...
  4,4,1);

% polar image plot
ePlotTitleText='Polar Image';
ePlotTitleDistance=5;
ePolarPlotAreaCenterPos=[70,80];
ePolarPlotAreaRadMin=0;
ePolarPlotAreaRadMax=40;

ePolarPlotAreaAngStart=0;
ePolarPlotAreaAngEnd=180;
epolari(matrix,ecolors(2));

[matrix cm]=eppmread([ePath 'defMap.ppm']);
ePolarPlotAreaAngStart=180;
ePolarPlotAreaAngEnd=290;
epolari(matrix,cm);

[matrix cm]=eppmread([ePath 'default.ppm']);
ePolarPlotAreaAngStart=290;
ePolarPlotAreaAngEnd=360;
epolari(matrix,cm);

eclose;
eview;
```

2.1.12 edemo7.m

```
eopen('demo7.eps');
eglobpar;

%a few global parameter
eTextFont=1;
eTextFontSize=6;

%titel
etext('Text',15,230,[15*2 15 -45]);
etext('Features',0,0,[15*2 15 45]);

%append text
etext('demo',30,30,100,1,2,45,[0.9 0.9 0.9]);
```

```
etext('Start at (20,210)...',20,210);
etext('10mm...',0,0,10);
etext('4mm...',0,0,4);
etext('new font...or Symbols:',0,0,6,1,2);
etext('\\141\\142\\147',0,0,6,1,13);
etext('new line...',20,185);
etext('go down... ',0,-3);
etext('go up... ',0,6);
etext('and back... ',0,-3);
etext('redtext... ',0,0,6,1,5,45,[1 0 0]);
etext('yellow... ',0,0,6,1,5,-45,[1 1 0]);
etext('blue... ',0,0,6,1,5,-135,[0 0 1]);
etext('red..',0,0,6,1,5,-225,[1 0 0]);
etext('green. ',0,0,6,1,5,45,[0 1 0]);
colorMap=ecolors(3,5);

% rotation
etext('rotate Text',50,140,6,1,2,0,colorMap(1,:));
etext('rotate Text',50,140,6,1,2,45,colorMap(2,:));
etext('rotate Text',50,140,6,1,2,90,colorMap(3,:));
etext('rotate Text',50,140,6,1,2,135,colorMap(4,:));
etext('rotate Text',50,140,6,1,2,180,colorMap(5,:));

etext('rotate Text',100,150,6,0,2,0,colorMap(1,:));
etext('rotate Text',100,150,6,3,2,45,colorMap(2,:));
etext('rotate Text',100,150,6,3,2,90,colorMap(3,:));
etext('rotate Text',100,150,6,3,2,135,colorMap(4,:));
etext('rotate Text',100,150,6,0,2,180,colorMap(5,:));

etext('rotate Text',150,160,6,-1,2,0,colorMap(1,:));
etext('rotate Text',150,160,6,-1,2,45,colorMap(2,:));
etext('rotate Text',150,160,6,-1,2,90,colorMap(3,:));
etext('rotate Text',150,160,6,-1,2,135,colorMap(4,:));
etext('rotate Text',150,160,6,-1,2,180,colorMap(5,:));

% left  center right
lineStep=-8;
yValue=130;
etext('left line1....',10,yValue,6,1);
etext('....center line1....',90,yValue,6,0);
etext('....right line1',170,yValue,6,-1);
yValue=yValue+lineStep;
etext('left line2..',10,yValue,6,1);
etext('..center line2..',90,yValue,6,0);
etext('..right line2',170,yValue,6,-1);
yValue=yValue+lineStep;
etext('left line3......',10,yValue,6,1);
etext('......center line3......',90,yValue,6,0);
etext('......right line3',170,yValue,6,-1);

%special character
eTextFont=1;
eTextFontSize=5;
xValue=10;
```

```
yValue=yValue+1.5*lineStep;
etext('Special Character of font 1-12 call by  octal value',xValue,yValue);
s=' ';
for i=[33,34,35,36,37,38,47,64]
  c=sprintf('\\134%o=\\%o  ',i,i);
  s=[s,c];
end
yValue=yValue+1.3*lineStep;
etext(s,xValue,yValue);
s=' ';
for i=[91,92,93,123,124,125,126]
  c=sprintf('\\134%o=\\%o  ',i,i);
  s=[s,c];
end
yValue=yValue+lineStep;
etext(s,xValue,yValue);
s=' ';
for i=[252,220,228,196,246,214,223]
  c=sprintf('\\134%o=\\%o  ',i,i);
  s=[s,c];
end
yValue=yValue+lineStep;
etext(s,xValue,yValue);

yValue=yValue+1.5*lineStep;
etext('Special Character of font 13 call by  octal value',xValue,yValue);
yValue=yValue+1.3*lineStep;
etext('',xValue,yValue);
for i=65:71
  c=sprintf('  \\134%o=',i);
  s=sprintf('\\%o',i);
  etext(c,0,0,eTextFontSize,1);
  etext(s,0,0,eTextFontSize,1,13);
end
etext('  ...',0,0,eTextFontSize,1);
i=90;
c=sprintf('  \\134%o=',i);
s=sprintf('\\%o',i);
etext(c,0,0,eTextFontSize,1);
etext(s,0,0,eTextFontSize,1,13);
yValue=yValue+lineStep;
etext('',xValue,yValue);
for i=97:103
  c=sprintf('  \\134%o=',i);
  s=sprintf('\\%o',i);
  etext(c,0,0,eTextFontSize,1);
  etext(s,0,0,eTextFontSize,1,13);
end
etext('  ...',0,0,eTextFontSize,1);
i=122;
c=sprintf('  \\134%o=',i);
s=sprintf('\\%o',i);
etext(c,0,0,eTextFontSize,1);
etext(s,0,0,eTextFontSize,1,13);
```

```
yValue=yValue+lineStep;
etext('',xValue,yValue);
for i=192:198
  c=sprintf('  \\134%o=',i);
  s=sprintf('\\%o',i);
  etext(c,0,0,eTextFontSize,1);
  etext(s,0,0,eTextFontSize,1,13);
end
yValue=yValue+lineStep;
etext('',xValue,yValue);
for i=199:205
  c=sprintf('  \\134%o=',i);
  s=sprintf('\\%o',i);
  etext(c,0,0,eTextFontSize,1);
  etext(s,0,0,eTextFontSize,1,13);
end
yValue=yValue+lineStep;
etext('',xValue,yValue);
for i=205:211
  c=sprintf('  \\134%o=',i);
  s=sprintf('\\%o',i);
  etext(c,0,0,eTextFontSize,1);
  etext(s,0,0,eTextFontSize,1,13);
end

eclose;
eview;
```

2.1.13 edemo8.m

```
% standard plot
eopen('demo8.eps')                      % open eps-file and write eps-head

eglobpar                                % get access to global parameters
%eWinGridVisible=1;
etext('Symbols ',eWinWidth/2,eWinHeight-10,10,0)
sFiles=['oplus.psd ';'plus.psd  ';'star.psd  ';'ring.psd  ';...
        'fring.psd ';'rect.psd  ';'frect.psd ';'tria.psd  ';...
        'ftria.psd ';'spire.psd ';'farrow.psd';'needle.psd';...
        'euro.psd  ';'feuro.psd '];
sColors=[0.7 0.6 0.5;1 0 0;0 1 0;0 0 1;0.7 0.5 0;1 0 1;...
         0 0.5 1;0.5 0 1;0.6 0.3 0.2;0.8 0.5 0.7;...
         1 0.8 0.3;0.6 0.3 0.4;0.6 0.4 0.8;0.8 0.9 0.3];

%define symbols
nSym=14;
for i=1:nSym
  edsymbol(sprintf('s%d',i),sFiles(i,:),... % define symbol
           1,1,0,0,0,sColors(i,:));
end

%draw symbols
nPos=40;
```

```
randPos=rand(nPos,2);
xPos(:,1)=randPos(:,1)*eWinWidth*0.7+eWinWidth*0.1;
yPos(:,2)=randPos(:,2)*eWinHeight*0.35+eWinHeight*0.56;
for i=1:nPos
  symbol=sprintf('s%d',rem(i,nSym)+1);
  esymbol(xPos(i,1),yPos(i,2),symbol,randPos(i,1)+0.4,...
  randPos(i,2)+0.3,(randPos(i,1)-randPos(i,2))*360);% draw symbol
end

etext('Table of Symbols ',eWinWidth/2,115,10,0);

%body of table
[tabx,taby]=etabdef(nSym,3,40,10,100,90,[1 3 2]);
for i=1:nSym
  etabtext(tabx,taby,i,1,sprintf('%d.',i),-1);
  etabtext(tabx,taby,i,2,sFiles(i,:),1,1,80,[0 0 0],sColors(i,:));
  esymbol(tabx(3,1)+tabx(3,2)/2,...
          taby(i,1)+taby(i,2)/2,...
          sprintf('s%d',i),0.5,0.5);

end
etabgrid(tabx,taby);

%head of table
[htabx htaby]=etabdef(1,3,40,100,100,8,[1 3 2],1);
etabtext(htabx,htaby,1,1,'No',0,3);
etabtext(htabx,htaby,1,2,'Filename',0,3);
etabtext(htabx,htaby,1,3,'Symbol',0,3);
eclose                                  % close eps-file
eview                                   % start ghostview with eps-file
```

2.1.14 edemo9.m

```
eopen('demo9.eps')                      % open eps-file and write eps-head
eglobpar;

titleFile='demo_title.ppm';
backgrFile='demo_backgr.ppm';
logoFile='demo_logo.ppm';
contentFile='demo_content.txt';

%make title image
[titleImg titleCM]=eppmread([ePath 'default.ppm']); % read image
swCM=titleCM(:,1)+titleCM(:,2)+titleCM(:,3);
swCM=swCM/max(swCM);
titleCM=[swCM swCM swCM]; % color -> gray
eppmwrit(titleFile,titleImg,titleCM); % save image

%make background image
[backImg backCM]=eshadoi; % get default shadow image
backCM(:,2:3)=0; % red colormap
eppmwrit(backgrFile,backImg,backCM); % save image
```

```
%make logo image
[logoImg logoCM]=eshadois; % get default shadow image
eppmwrit(logoFile,logoImg,logoCM); % save image

%content
lf=setstr(10); %linefeed
contenttext=[
  'New features:' lf ...
  '#log scaled axes###'  lf ...
  '#return long tic positions of axes###'  lf ...
  '#new image functions e.g. ebright,econtra,eimgrot###' lf ...
  '#text functions etxtbox, etxtread, etxtlpos###' lf ...
  '#pie plot and error bars###' lf ...
  '#linear interpolation with elineip and efillmat###' lf  ...
  '#frames, circles, ellipses###' lf ...
  '#ASCII85 bitmap code, generates small EPS-files###' lf ...
  '#JPEG-file integration###' lf ...
  '#bugs removed in eopen, ehead, egridcl ...###' lf ...
  ];
etxtwrit(contenttext,contentFile);

% make cover

ecdcover('The EpsTk',...
         'Graphic for Octave & MATLAB(R)',...
         'St.Mueller',...
         'Version 2.0',...
         '2003',...
         [1 1 0],...
         titleFile,backgrFile,...
         logoFile,contentFile);


eclose
eview
```