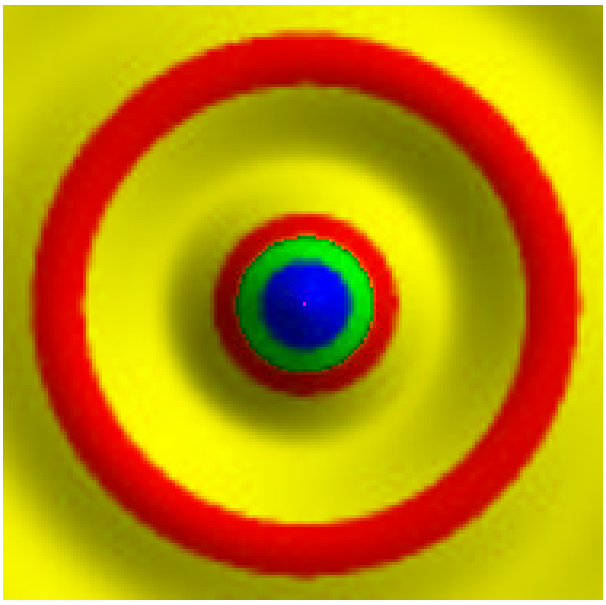
# Quick Reference of eps-Toolkit 1.9 for Octave and MATLAB(R)



www.epstk.de

Stefan Müller
stefan.mueller@fgan.de

FGAN,Wachtberg Werthhoven,January 15, 2002

# Contents

# Chapter 1

# Introduction

This EPS-Toolkit is a solution of my big problem that I had 1997. I needed graphical output functions for programs which ran with Octave (a freeware Matlab-clone on Unix(Linux)-sytems) and Matlab. But I could not find any tool in the internet. So, I wrote this tool myself.
The toolkit consists of some matlab functions, which generate graphical outputs with postscript commands. To view and print the generated postscript files you can use Ghostscript or Ghostview (freeware for all systems).

**Features**

- MATLAB(R)-Code(¡5.0) with graphical output functions runs in Matlab and Octave
- All functions are pure m-files
- Most 2D scientific graphics functions are written
- Modifications and extensions are no problem
- Generated EPS-files are very small and importable with no loss of quality
- WYSIWYG (with Ghostview)
- Freeware (GPL)

**Difference to version 1.8**

- import and export images (eppmread,eppmwrit)
- quiver, needle or other symbol plot (equiver)
- bar plot (ebar)
- mix shadow plot (eshadoix)
- import of MATLAB(R) eps-files (new version of einseps)
- many bugs removed in eisoline, ebitmap, etext, esubeps... ,
- new symbols and demos
- new special function (egradient,ecdcover)

**Requirements**

- Matlab $\geq 3.x$ or Octave $\geq 2.x$
- Ghostscript and Ghostview or other viewer for EPS-Files
- 700 kB space on disk

**Installation**

1. Include the epstk-directory in the octave-path or matlab-path
2. Edit file "einit.m" in epstk-directory to set the Parameter "ePath", "eGhostview" and "eGhostscript".

**Testing**

1. Start matlab or octave
2. Start demos edemo1, edemo2, ... edemo11

If all demos works, epstk should be ok.

**Thanks**

Thanks to Josef Worms, Jörg Heckenbach, Coletta Schumacher and Gerd Krämer.

# Chapter 2

# Global Parameters

## 2.1 The toolkit and the parameter

This toolkit needs global variables of MATLAB(R) or octave to get default values for the functions. All global parameter are defined in the file 'einit.m'. Which global parameter are used by a function you can see in the description of the functions after this chapter.

## 2.2 Handling of parameter

If you want to change any default values, edit the file 'einit.m'. This file is called by function 'eopen' everytime.

You can change a global parameter in a program temporarily, if you have called 'eglobpar' before. If you need the default values again after modifications of the global parameter, you should use the commands 'esavpar' and 'erespar'. 'esavpar' saves all global parameters and 'erespar' restores the last backup.

Please note: There are some parameter you can not change temporarily!
They are parameter like 'epsFileName' or 'pageOrientation', which initialize the plot-file, the page and the window with the function 'eopen'. In this case use the parameter of 'eopen'.

## 2.3 Default Values

Here are the default values of the toolkit (listing of 'einit.m'):

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

eGhostview='ghostview -magstep -1'; %ghostview for linux
%eGhostview='"c:/gstools/gsview/gsview32.exe"'; %ghostview for windows

eGhostscript='gs'; %ghostscript for linux
%eGhostscript='"c:/gstools/gs5.50/gswin32.exe"'; %ghostscript for windows

eFileName='epstkout.eps';                % default eps-outputfile
eUserUnit='mm';                         % or 'cm' or 'inch' or 'inch/72'
```

```
%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;
];


% 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
eXScaleFac=1; % 1=no resize
eXScaleFac=1; % 1=no resize
```

```
% window
eWinOrigin=[0 0]; % mm
eWinWidth=180; % mm
eWinHeight=250; % mm
eWinFrameVisible=0; % 1=on 0=off   draw frame around window
eWinGridVisible=0; % 1=on 0=off   draw grid of window
eWinFrameLineWidth=0.3; % mm
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;

%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;

% 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=1; % 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=2; % 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

% 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
eXAxisSouthValueFormat=0; %  n digits after decimal point,0=auto
eXAxisSouthValueVisible=1; %  0=off 1=on
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
eXAxisNorthValueFormat=0;  %  n digits after decimal point,0=auto
eXAxisNorthValueVisible=1;  %  0=off 1=on
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
eYAxisWestValueFormat=0;   %  n digits after decimal point,0=auto
eYAxisWestValueVisible=1;   %  0=off 1=on
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
eYAxisEastValueFormat=0;  %  n digits after decimal point,0=auto
eYAxisEastValueVisible=1;  %  0=off 1=on
eYAxisEastLabelDistance=6;  % mm  label distance from axis
eYAxisEastLabelText='';
eYAxisEastVisible=1; %  0=off 1=on

%polar radius axis
ePolarAxisRadScale=[0 0 0]; % [start step end]
ePolarAxisRadValueFormat=0;  %  n digits after decimal point,0=auto
ePolarAxisRadValueVisible=1;  %  0=off 1=on
```

```
ePolarAxisRadVisible=1;  %  0=off 1=on

%polar Angle axis
ePolarAxisAngScale=[0 0 0]; % [start step end]
ePolarAxisAngValueFormat=0;  %  n digits after decimal point,0=auto
ePolarAxisAngValueVisible=1;  %  0=off 1=on
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]
eImageLegendValueFormat=0; %  n digits after decimal point,0=auto
eImageLegendValueVisible=1;  %  0=off 1=on
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=1;  % font number   1=TimesRoman
eParamValue='';
eParamValueFont=9;  % 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

%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=0;  %  n digits after decimal point,0=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
```

# Chapter 3

# User Functions

## 3.1 Basic Functions

### 3.1.1 ebitmap.m

```
NAME
  ebitmap  -  transform  the current eps-file to bitmap-file

SYNOPSIS
  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
                 default: 200
  mapFileName    name of bitmap-file
                 default: 'eFileName.typeSuffix'
  epsFileName    name of eps-file
                 default: 'eFileName'
GLOBAL PARAMETER(S)
  eFileName
  eGhostscript
```

### 3.1.2 eclose.m

```
NAME
  eclose - finish plot(s) and close EPS-file

SYNOPSIS
  eclose ([endOfPage[,message]])

PARAMETER(S)
  endOfPage   switch for the postscript command 'showpage'
```

```
              if endOfPage=1 then append 'showpage' (default)
              else 'showpage' will not append
  message     switch for file written message
              if message=1 then write message (default)
              else message

GLOBAL PARAMETER(S)
  eWinFrameVisible
  eWinTimeStampVisible
```

### 3.1.3 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]]])
        [,origin[,xScaleFac,yScaleFac[,pageWidth,pageHeight]]]]]])

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)
  origin            vector of origin, [xOffset yOffset]
  xScaleFac         scale factor 1= no resize
  yScaleFac         scale factor 1= no resize
  pageWidth         width of page
  pageHeight        height of page

GLOBAL PARAMETER(S)
  eFileName
  ePageWidth
  ePageHeight
  ePageScaleFac
  ePageOrientation
  eUserUnit
  eWinWidth
  eWinHeight
  eWinOrigin
  eFonts
```

### 3.1.4 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
```

## 3.2 Axes Functions

### 3.2.1 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)
  eXAxis(South|West|East|North)ValueFormat
  eXAxis(South|West|East|North)ValueVisible
  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
  ePlotAreaXValueStart
  ePlotAreaXValueEnd
  ePlotAreaYValueStart
  ePlotAreaYValueEnd
```

```
  ePlotAreaPos
  ePlotAreaWidth
  ePlotAreaHeight
```

### 3.2.2 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)
  ePolarAxisRadScale
  ePolarAxisAngScale
  ePolarAxisRadVisible
  ePolarPlotAreaCenterPos
  ePolarPlotAreaRadMin
  ePolarPlotAreaRadMax
  ePolarPlotAreaAngStart
  ePolarPlotAreaAngEnd
  ePolarPlotAreaValStart
  ePolarPlotAreaValEnd
  ePolarAxisRadVisible
  ePolarAxisRadValueFormat
  ePolarAxisRadValueVisible
  ePolarAxisAngVisible
  ePolarAxisAngValueFormat
  ePolarAxisAngValueVisible
  eAxesValueFontSize
  eAxesColor
  eAxesLineWidth
  eAxesTicShortLength
  eAxesTicLongLength
  eAxesTicLongMaxN
  eAxesValueSpace
```

### 3.2.3 eaxis.m

```
NAME
  eaxis - draw scaled axis

SYNOPSIS
  eaxis(xPos,yPos,length,type,scale[,angle[,color]])
```

```
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

GLOBAL PARAMETER(S)
  eXAxisSouthValueFormat
  eYAxisWestValueFormat
  eXAxisNorthValueFormat
  eYAxisEastValueFormat
  eXAxisSouthValueVisible
  eYAxisWestValueVisible
  eXAxisNorthValueVisible
  eYAxisEastValueVisible
  eAxesColor
  eAxesValueFontSize
  eAxesValueSpace
  eAxesLineWidth
  eAxesTicShortLength
  eAxesTicLongLength
  eAxesTicLongMaxN
```

## 3.3 Grid Functions

### 3.3.1 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)
  eXAxisSouthScale
  eXAxisSouthScaleType
```

```
  eYAxisWestScale
  eYAxisWestScaleType
  eXAxisNorthScale
  eYAxisEastScale
  ePlotAreaPos
  ePlotAreaWidth
  ePlotAreaHeight
  eXGridVisible
  eYGridVisible
  eAxesTicLongMaxN
  eXGridLineWidth
  eXGridColor
  eXGridDash
```

### 3.3.2 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)
  ePolarAxisRadScale
  ePolarAxisAngScale
  ePolarPlotAreaValStart
  ePolarPlotAreaValEnd
  ePolarPlotAreaCenterPos
  ePolarPlotAreaRadMin
  ePolarPlotAreaRadMax
  ePolarPlotAreaAngStart
  ePolarPlotAreaAngEnd
  ePolarRadiusGridVisible
  ePolarRadiusGridLineWidth
  ePolarRadiusGridColor
  ePolarRadiusGridDash
  ePolarAngleGridVisible
  ePolarAngleGridLineWidth
  ePolarAngleGridColor
  ePolarAngleGridDash
  eAxesTicLongMaxN
```

## 3.4 Plot Functions

### 3.4.1 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)
  eContourLineColor
  eContourLineDash
  eContourScale
  eContourLevelsMaxN
  eContourValueFormat
  ePlotAreaWidth
  ePlotAreaHeight
  ePlotAreaPos
  eContourLineWidth
  eContourValueVisible
  eContourValueDistance
  eContourValueFont
  eContourValueFontSize
  eYAxisWestScale
  eXAxisSouthScale
        valueForm=sprintf('1.%df',vForm);
```

### 3.4.2 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           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
```

```
 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
```

### 3.4.3 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;
```

### 3.4.4 epolari.m

```
NAME
  epolari  - draw polar image of a matrix

SYNOPSIS
  epolari(image[,colorMap])

PARAMETER(S)
  matrix     matrix for image
             each value of the matrix is a row index of the colormap
  colorMap   define own colormap

GLOBAL PARAMETER(S)
  eImageDefaultColorMap
  ePolarPlotAreaCenterPos
  ePolarPlotAreaRadMax
  ePolarPlotAreaAngEnd
  ePolarPlotAreaAngStart
  ePolarPlotAreaRadMin
  ePolarPlotAreaRadMax
```

## 3.5 Images Functions

### 3.5.1 eimage.m

```
NAME
  eimage  - draw image of a matrix

SYNOPSIS
  eimage(matrix[,colorMap])

PARAMETER(S)
  matrix     matrix for image
             each value of the matrix is a row index of the colormap
  colorMap   define own colormap

GLOBAL PARAMETER(S)
  eImageDefaultColorMap
  ePlotAreaPos
  ePlotAreaWidth
  ePlotAreaHeight
  eImageFrameVisible
  eAxesLineWidth
```

### 3.5.2 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
  eYAxisWestScale
  eXAxisSouthScale
  eImageLegendScale
  eImageLegendVisible
  eImageLegendPos
  eImageLegendHeight
  eImageLegendWidth
  eImageLegendLabelDistance
  eImageLegendValueFormat
  eImageLegendLabelText
  eAxesColor
  eAxesTicLongLength
  eAxesTicShortLength
  eAxesTicLongMaxN
  eAxesValueSpace
  eAxesValueFontSize
  eAxesLabelFontSize
  eAxesLabelTextFont
  eAxesLineWidth
```

### 3.5.3 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
  eImageLegendLabelDistance
  eImageLegendValueFormat
  eImageLegendLabelText
  eImageLegendPos
  eImageLegendVisible
  eImageLegendWidth
  eImageLegendHeight
  ePolarAxisRadScale
  ePolarAxisAngScale
  ePolarPlotAreaCenterPos
  ePolarPlotAreaAngStart
  ePolarPlotAreaAngEnd
  ePolarPlotAreaRadMax
  eAxesColor
  eAxesTicLongLength
  eAxesTicShortLength
  eAxesTicLongMaxN
  eAxesValueSpace
  eAxesValueFontSize
  eAxesLabelFontSize
  eAxesLabelTextFont
  eAxesLineWidth
```

### 3.5.4 eppmread.m

```
NAME
  eppmread  - read PPM-file

SYNOPSIS
  [image,colormap]=eppmread(filename)

PARAMETER(S)
  filename    name of PPM-file
  image       image matrix
  colormap    color table
```

### 3.5.5 eppmwrit.m

```
NAME
  eppmwrit  - save image as PPM-file

SYNOPSIS
  eppmwrit(filename,image,colormap[,binary])

PARAMETER(S)
  filename    name of PPM-file
  image       image matrix
  colormap    color table
  binary      default: binary=1 for binary PPM-file
              binary=0 for ascii PPM-file
```

### 3.5.6 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
```

### 3.5.7 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)
  eImageLegendScale
  eImageDefaultColorMap
  eImageLegendVisible
  eImageLegendWidth
  eImageLegendHeight
  ePlotAreaPos
  ePlotAreaWidth
  ePlotAreaHeight
  eAxesTicLongLength
  eAxesValueSpace
  eAxesValueFontSize
  eAxesLabelFontSize
  eAxesLabelTextFont
  eAxesColor
  eAxesLineWidth
  eAxesTicShortLength
  eAxesTicLongLength
  eAxesTicLongMaxN
  eAxesValueSpace
  eImageLegendPos
  eImageLegendLabelDistance
  eImageLegendValueFormat
  eImageLegendVisible,...
  eImageLegendLabelText
  eYAxisWestScale
  eXAxisSouthScale
```

### 3.5.8 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
```

## 3.6 Line Functions

### 3.6.1 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 polyline
  dash        if dash=0 then draw solid lines
              else value of dash is the distance of dashes
  color       vector of polyline color ([r g b])

GLOBAL PARAMETER(S)
  eLineColor
  eLineDash
  eLineWidth
```

### 3.6.2 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
  dash        if dash=0 then draw solid lines
              if dash<0 then fill polyline
              else value of dash is the distance of dashes
  color       vector of polyline color ([r g b])

GLOBAL PARAMETER(S)
  eLineColor
  eLineDash
  eLineWidth
```

## 3.7 Text Functions

### 3.7.1 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
```

### 3.7.2 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      size of current font
  alignment     1=left, 0=center, -1=right
  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
```

## 3.8 Symbol Functions

### 3.8.1 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
```

### 3.8.2 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
```

### 3.8.3 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)
```

## 3.9 Table Functions

### 3.9.1 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
```

### 3.9.2 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)
```

### 3.9.3 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=left,0=center, -1=right
  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
```

## 3.10 Special Functions

### 3.10.1 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
```

### 3.10.2 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(ppm-file)  of frontImage , default=''
  background    filename(ppm-file) of background, default=''
  logo          filename(ppm-file)  of logo, default=''
  content       filename(asci-file, tabs with '#')  of table of contents
```

### 3.10.3 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
```

### 3.10.4 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
```

### 3.10.5 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
```

### 3.10.6 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
```

### 3.10.7 eplo2win.m

```
NAME
  eplo2win - transform coordinates , plotarea to window

SYNOPSIS
  [winX winY]=eplo2win(x,y)

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)
```

### 3.10.8 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
```

### 3.10.9 esubeps.m

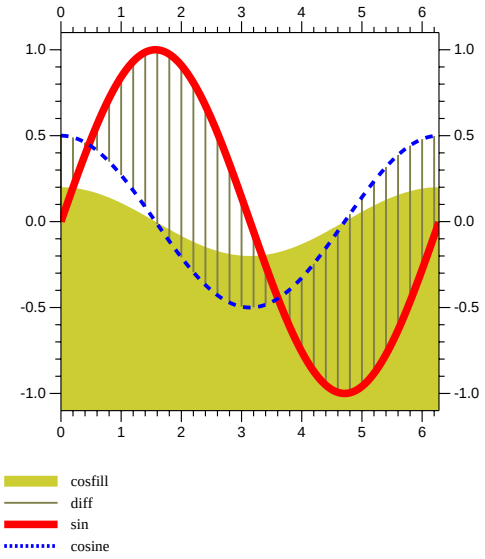
```
NAME
  esubeps  -  insert eps-file in a subarea of the window

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
```

# Appendix A

## A.1 Examples

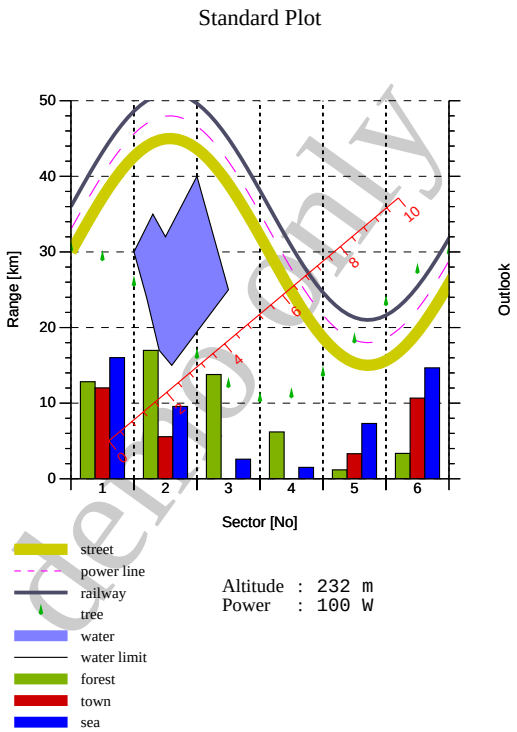
### A.1.1 edemo1.m




```
   x1=[0:0.01:2*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,[0.5 0.5 0.3],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)

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

### A.1.2 edemo2.m

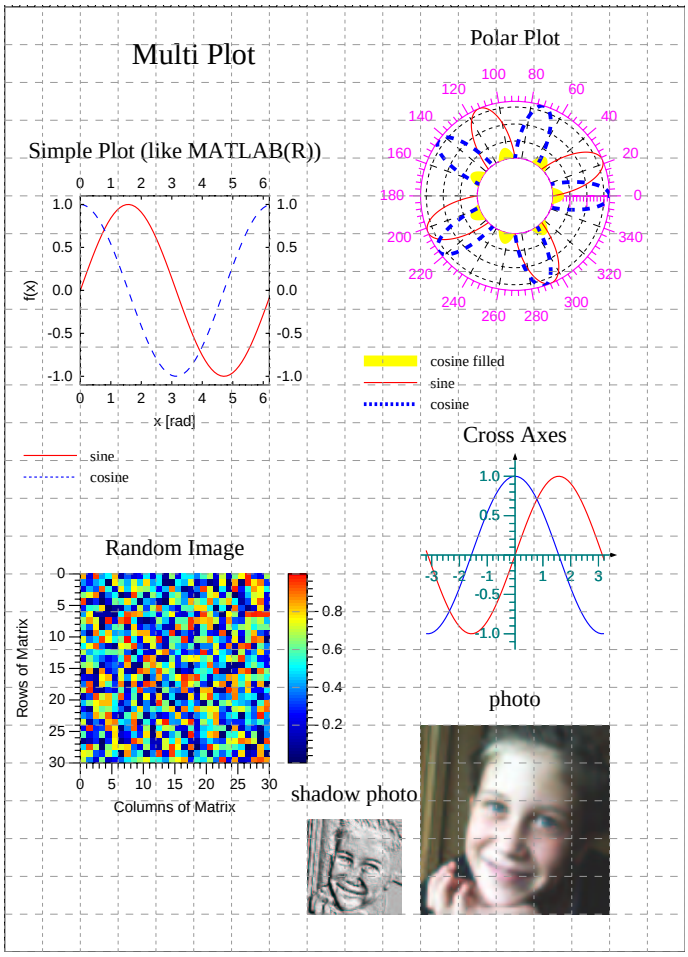





```
   % standard plot
eopen('demo2.eps')                      % open eps-file and write eps-head
eglobpar                                % get access to global parameters
ePlotTitleText='Standard Plot';         % set title
```

```
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
etext('demo only',eWinWidth/2,eWinHeight/2,40,0,1,...   %print text
     atan(eWinHeight/eWinWidth)*180/pi,[0.8 0.8 0.8])
eXAxisSouthScale=[1999 1 2005];
eaxes([0 0 6],[0 10 50])               % scale axes

% 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])

% 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 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')

% axis
eaxis(50,110,100,'s',[0 0 10],40,[1 0 0]);

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

### A.1.3 edemo3.m



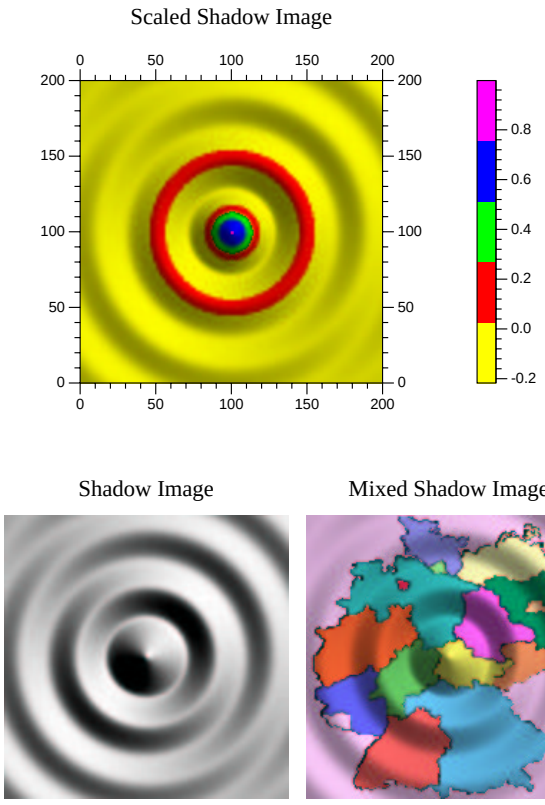

```
   eopen('demo3.eps');
eglobpar;                       % get access to global parameters
eWinGridVisible=1;
esavpar;    % save default parameter

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

% set tics like Matlab
eAxesValueSpace=3;
eAxesTicLongLength=-1;
eAxesTicShortLength=0;

% simple plot
ePlotAreaPos=[20 150];
ePlotAreaHeight=50;
ePlotAreaWidth=50;
eYAxisWestLabelText='f(x)';
eXAxisSouthLabelText='x [rad]';
ePlotTitleDistance=10;
ePlotTitleText='Simple Plot (like MATLAB(R))';
```

```
xData=0:0.1:2*pi;
eplot(xData,sin(xData),'sine',0,[1 0 0]);
eplot(xData,cos(xData),'cosine',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;

%  cross axes plot
erespar; % set default parameter
eAxesColor=[0 0.5 0.5];
ePlotAreaPos=[110 80];
ePlotAreaHeight=50;
ePlotAreaWidth=50;
ePlotTitleDistance=5;
ePlotTitleText='Cross Axes';
eAxesCrossOrigin=1;
edsymbol('arrow','spire.psd',0.2,0.3);
esymbol(ePlotAreaPos(1)+ePlotAreaWidth,...
        ePlotAreaPos(2)+ePlotAreaHeight/2,'arrow');
esymbol(ePlotAreaPos(1)+ePlotAreaWidth/2,...
        ePlotAreaPos(2)+ePlotAreaHeight,'arrow',1,1,90);
eaxes([-3.4 0 3.4],[-1.2 0 1.2]);
xData=-3.2:0.1:3.2;
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='Rows of Matrix';
eXAxisSouthLabelText='Columns of Matrix';
eImageLegendPos=[0 -5];
ePlotTitleDistance=5;
ePlotTitleText='Random Image';
eXAxisNorthVisible=0;
eYAxisEastVisible=0;
matrix=rand(30,30);
```

```
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;
%stdImage=eimage;  % print image
photo=eimage;  % print default image

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

eclose;
eview;
```

### A.1.4 edemo4.m

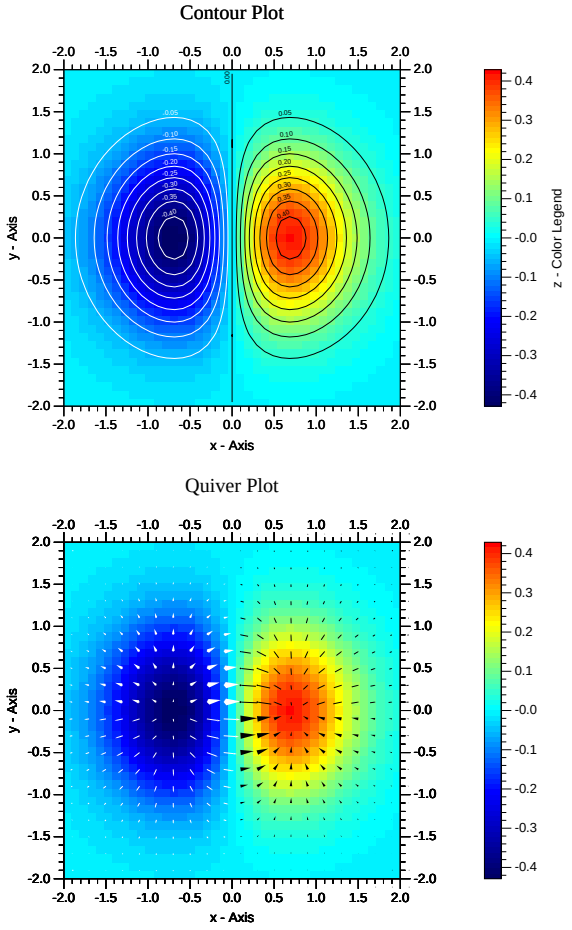
```
eopen('demo4.eps');
eglobpar;

[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=sin(R)./R;

% scaled shadow image
ePlotAreaPos=[30 140];
ePlotAreaHeight=80;
ePlotAreaWidth=80;
ePlotTitleDistance=15;
ePlotTitleText='Scaled Shadow Image';
colorMap=[1 1 0;1 0 0;0 1 0;0 0 1;1 0 1];
eshadois(matrix,colorMap,'e');

% 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;
```

### A.1.5 edemo5.m

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

% contour plot
```



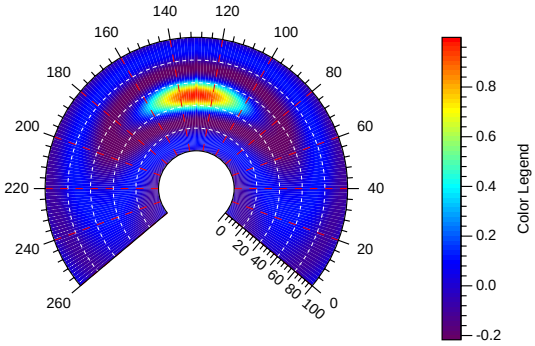

```
eopen('demo5a.eps',0,180,140);
eglobpar;
ePlotTitleText='Contour Plot';
ePlotTitleDistance=15;
ePlotAreaPos=[20 20];
eXAxisSouthLabelText='x - Axis';
eYAxisWestLabelText='y - Axis';
eImageLegendLabelText='z - Color Legend';
eContourValueVisible=1;
eaxes([-2 0 2],[-2 0 2]);
eimagesc(matrix,ecolors(3),'e');
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';
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
```

### A.1.6 edemo6.m

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

Scaled Polar Image

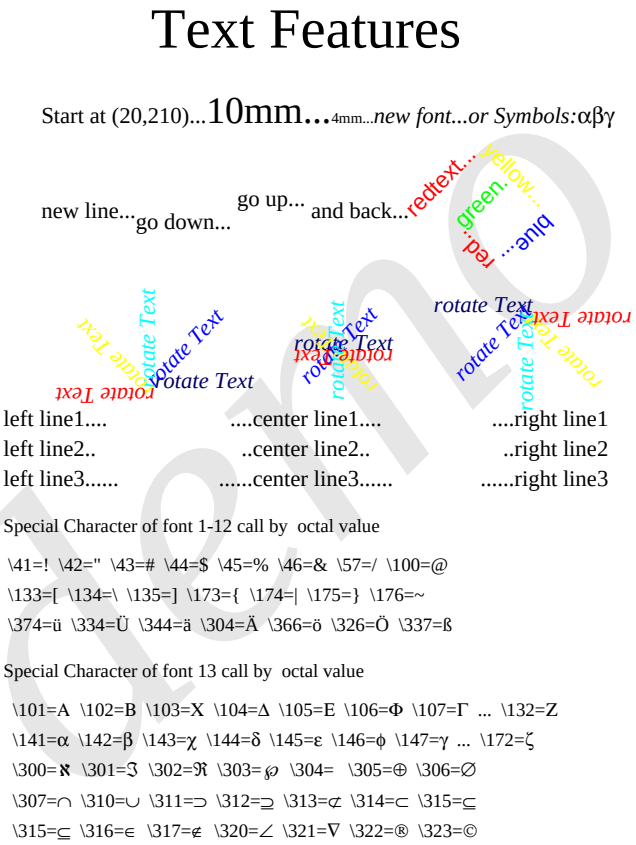



Polar Image

```
% 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;
ePlotTitleText='Scaled Polar Image';
ePolarPlotAreaCenterPos=[70,170];
ePolarPlotAreaRadMax=40;
ePolarPlotAreaAngStart=-40;
ePolarPlotAreaAngEnd=220;
ePolarRadiusGridColor=[1 1 1];
ePolarAngleGridColor=[1 0 0];
eImageLegendLabelText='Color Legend';
matrix=epolaris(matrix,ecolors(2),'e');

[matrix cm]=eppmread([ePath 'defMap.ppm']);
% polar image plot
ePlotTitleText='Polar Image';
ePlotTitleDistance=-25;
ePolarPlotAreaRadMin=0;
ePolarPlotAreaRadMax=40;
```

```
ePolarPlotAreaCenterPos=[70,80];
ePolarPlotAreaAngStart=180;
ePolarPlotAreaAngEnd=290;
epolari(matrix,cm);
[matrix cm]=eppmread([ePath 'default.ppm']);
ePolarPlotAreaCenterPos=[70,80];
ePolarPlotAreaAngStart=290;
ePolarPlotAreaAngEnd=360;
epolari(matrix,cm);

eclose;
eview;
```

### A.1.7 edemo7.m




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

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

```
%titel
etext('Text Features',eWinWidth/2,230,15,0);

%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',100,150,6,0,2,45,colorMap(2,:));
etext('rotate Text',100,150,6,0,2,90,colorMap(3,:));
etext('rotate Text',100,150,6,0,2,135,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;
```

### A.1.8 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'];
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];

%define symbols
nSym=12;
for i=1:nSym
  edsymbol(sprintf('s%d',i),sFiles(i,:),... % define symbol
```

Symbols

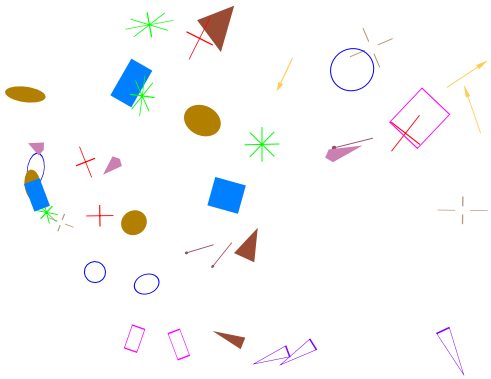


Table of Symbols

| No | Filename | Symbol |
|---|---|---|
| 1. | oplus.psd | |
| 2. | plus.psd | |
| 3. | star.psd | |
| 4. | ring.psd | |
| 5. | fring.psd | |
| 6. | rect.psd | |
| 7. | frect.psd | |
| 8. | tria.psd | |
| 9. | ftria.psd | |
| 10. | spire.psd | |
| 11. | farrow.psd | |
| 12. | needle.psd | |

```
            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
```

### A.1.9 edemo9.m


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

%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];
eppmwrit('title.ppm',titleImg,titleCM); % save image

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

%make logo image
[logoImg logoCM]=eshadois; % get default shadow image
eppmwrit('logo.ppm',logoImg,logoCM); % save image

%content
lf=setstr(10); %linefeed
contenttext=[
  'New features:' lf ...
  '#import and export of images###'  lf ...
  '#quiverplots###' lf ...
  '#new symbols###' lf ...
  '#new demos###' lf  ...
  '#mix shadow plots###' lf ...
  '#import of matlab eps-files###' lf ...
  '#bugs removed in eisoline, esubeps, ebitmap...###' lf ...
  ];
fid=fopen('content.txt','w');
fprintf(fid,'%s',contenttext);
fclose(fid);

% make cover

ecdcover('The EpsTk',...
         'Graphic for Octave & MATLAB(R)',...
         'St.Mueller',...
         'Version 1.9',...
         '2001',...
         [1 1 0],...
         'title.ppm','backgr.ppm',...
         'logo.ppm','content.txt');


eclose
eview
```

### A.1.10 edemo10.m

```
%%  edemo10 - print calendar of a year
%%
%%
% written by Coletta Schumacher and Stefan Mueller
year=2002;
%              day   month  textIndex textColumn textColor    backgroundColor
```

| Januar | Februar | März | April | Mai | Juni |
|---|---|---|---|---|---|
| 1 Di Neujahr | 1 Fr | 1 Fr | 1 Mo Ostern | 1 Mi 1.Mai | 1 Sa |
| 2 Mi | 2 Sa | 2 Sa | 2 Di | 2 Do | 2 So vorgearb. |
| 3 Do | 3 So | 3 So | 3 Mi | 3 Fr | 3 Mo |
| 4 Fr | 4 Mo | 4 Mo | 4 Do | 4 Sa | 4 Di |
| 5 Sa | 5 Di | 5 Di | 5 Fr | 5 So | 5 Mi |
| 6 So | 6 Mi | 6 Mi | 6 Sa | 6 Mo | 6 Do |
| 7 Mo dkz | 7 Do | 7 Do | 7 So | 7 Di | 7 Fr |
| 8 Di | 8 Fr fed | 8 Fr | 8 Mo | 8 Mi | 8 Sa |
| 9 Mi | 9 Sa | 9 Sa | 9 Di fsf | 9 Do Chr.Himmelf | 9 So fsl |
| 10 Do | 10 So Fastnacht | 10 So | 10 Mi | 10 Fr | 10 Mo |
| 11 Fr | 11 Mo Fastnacht | 11 Mo | 11 Do | 11 Sa | 11 Di |
| 12 Sa | 12 Di | 12 Di | 12 Fr | 12 So | 12 Mi |
| 13 So | 13 Mi | 13 Mi ld | 13 Sa | 13 Mo | 13 Do |
| 14 Mo | 14 Do | 14 Do | 14 So lun | 14 Di | 14 Fr |
| 15 Di | 15 Fr | 15 Fr dom | 15 Mo | 15 Mi | 15 Sa |
| 16 Mi | 16 Sa | 16 Sa | 16 Di | 16 Do | 16 So |
| 17 Do | 17 So | 17 So | 17 Mi | 17 Fr | 17 Mo |
| 18 Fr | 18 Mo | 18 Mo | 18 Do | 18 Sa | 18 Di |
| 19 Sa | 19 Di | 19 Di | 19 Fr | 19 So Pfingsten | 19 Mi |
| 20 So | 20 Mi | 20 Mi | 20 Sa | 20 Mo Pfingsten | 20 Do |
| 21 Mo | 21 Do | 21 Do | 21 So | 21 Di | 21 Fr mwi |
| 22 Di | 22 Fr | 22 Fr | 22 Mo | 22 Mi | 22 Sa |
| 23 Mi | 23 Sa | 23 Sa | 23 Di | 23 Do | 23 So vorgearb. vdk |
| 24 Do sed | 24 So | 24 So | 24 Mi | 24 Fr | 24 Mo |
| 25 Fr | 25 Mo | 25 Mo | 25 Do | 25 Sa | 25 Di |
| 26 Sa | 26 Di | 26 Di | 26 Fr | 26 So | 26 Mi |
| 27 So | 27 Mi | 27 Mi | 27 Sa | 27 Mo wen | 27 Do |
| 28 Mo | 28 Do | 28 Do | 28 So | 28 Di | 28 Fr |
| 29 Di sdk | | 29 Fr Karfreitag sei | 29 Mo | 29 Mi | 29 Sa |
| 30 Mi | | 30 Sa | 30 Di | 30 Do Fronleich. | 30 So |
| 31 Do | | 31 So Ostern | | 31 Fr | |

## Jahr 2002

| Juli | August | September | Oktober | November | Dezember |
|---|---|---|---|---|---|
| 1 Mo | 1 Do | 1 So | 1 Di | 1 Fr Allerheiligen | 1 So |
| 2 Di | 2 Fr | 2 Mo | 2 Mi vorgearb. | 2 Sa | 2 Mo |
| 3 Mi | 3 Sa | 3 Di | 3 Do Dt. Einheit | 3 So | 3 Di |
| 4 Do | 4 So | 4 Mi | 4 Fr | 4 Mo | 4 Mi |
| 5 Fr | 5 Mo | 5 Do | 5 Sa | 5 Di | 5 Do |
| 6 Sa | 6 Di | 6 Fr | 6 So | 6 Mi | 6 Fr |
| 7 So | 7 Mi | 7 Sa | 7 Mo | 7 Do | 7 Sa |
| 8 Mo | 8 Do ouo | 8 So | 8 Di jos | 8 Fr | 8 So hra |
| 9 Di | 9 Fr ase | 9 Mo | 9 Mi | 9 Sa | 9 Mo |
| 10 Mi | 10 Sa | 10 Di | 10 Do | 10 So | 10 Di |
| 11 Do | 11 So | 11 Mi | 11 Fr | 11 Mo | 11 Mi ros |
| 12 Fr fsf | 12 Mo | 12 Do | 12 Sa | 12 Di | 12 Do |
| 13 Sa oei | 13 Di les | 13 Fr | 13 So | 13 Mi | 13 Fr |
| 14 So usk | 14 Mi | 14 Sa sdw | 14 Mo | 14 Do | 14 Sa |
| 15 Mo | 15 Do | 15 So wer | 15 Di | 15 Fr | 15 So |
| 16 Di | 16 Fr | 16 Mo | 16 Mi lhl | 16 Sa | 16 Mo |
| 17 Mi | 17 Sa | 17 Di | 17 Do | 17 So | 17 Di |
| 18 Do | 18 So | 18 Mi | 18 Fr | 18 Mo | 18 Mi |
| 19 Fr | 19 Mo | 19 Do | 19 Sa | 19 Di | 19 Do |
| 20 Sa | 20 Di | 20 Fr | 20 So | 20 Mi | 20 Fr |
| 21 So | 21 Mi | 21 Sa | 21 Mo | 21 Do | 21 Sa |
| 22 Mo | 22 Do | 22 So | 22 Di lei | 22 Fr | 22 So |
| 23 Di | 23 Fr | 23 Mo | 23 Mi | 23 Sa | 23 Mo |
| 24 Mi jkr | 24 Sa | 24 Di | 24 Do | 24 So | 24 Di |
| 25 Do | 25 So | 25 Mi | 25 Fr | 25 Mo | 25 Mi Weihnachten |
| 26 Fr | 26 Mo | 26 Do | 26 Sa qoa | 26 Di | 26 Do Weihnachten |
| 27 Sa gzi | 27 Di | 27 Fr | 27 So | 27 Mi | 27 Fr vorgearb. |
| 28 So | 28 Mi | 28 Sa | 28 Mo | 28 Do | 28 Sa vorgearb. |
| 29 Mo | 29 Do | 29 So | 29 Di | 29 Fr | 29 So vorgearb. |
| 30 Di | 30 Fr | 30 Mo | 30 Mi | 30 Sa | 30 Mo |
| 31 Mi | 31 Sa | | 31 Do | | 31 Di Silvester |

```
myHolidays=   [
                 7     1     20        2          1.0 1.0 1.0  0.9 0.2 0.2;
                24     1     21        2          1.0 1.0 1.0  0.9 0.2 0.2;
                29     1     22        2          1.0 1.0 1.0  0.9 0.2 0.2;
                 8     2     23        2          1.0 1.0 1.0  0.9 0.2 0.2;
                 9     4     24        2          1.0 1.0 1.0  0.9 0.2 0.2;
                 9     6     25        2          1.0 1.0 1.0  0.9 0.2 0.2;
                23     6     26        2          1.0 1.0 1.0  0.9 0.2 0.2;
                12     7     27        2          1.0 1.0 1.0  0.9 0.2 0.2;
                13     7     28        2          1.0 1.0 1.0  0.9 0.2 0.2;
                24     7     29        2          1.0 1.0 1.0  0.9 0.2 0.2;
                 8     8     30        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;
                13     8     34        2          1.0 1.0 1.0  0.9 0.2 0.2;
                14     9     33        2          1.0 1.0 1.0  0.9 0.2 0.2;
                15     9     32        2          1.0 1.0 1.0  0.9 0.2 0.2;
                16    10     35        2          1.0 1.0 1.0  0.9 0.2 0.2;
                22    10     36        2          1.0 1.0 1.0  0.9 0.2 0.2;
                26    10     37        2          1.0 1.0 1.0  0.9 0.2 0.2;
                13     3     40        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;
                15     3     42        2          1.0 1.0 1.0  0.9 0.2 0.2;
```

```
                27    7     43        2          1.0 1.0 1.0  0.9 0.2 0.2;
                14    4     44        2          1.0 1.0 1.0  0.9 0.2 0.2;
                11   12     45        2          1.0 1.0 1.0  0.9 0.2 0.2;
                 8   10     46        2          1.0 1.0 1.0  0.9 0.2 0.2;
                29    3     47        2          1.0 1.0 1.0  0.9 0.2 0.2;
                14    7     48        2          1.0 1.0 1.0  0.9 0.2 0.2;
                27    5     49        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;
                 2    6     80        1          0.0 0.0 0.0  0.9 0.8 0.5;
                23    6     80        1          0.0 0.0 0.0  0.9 0.8 0.5;
                 2   10     80        1          0.0 0.0 0.0  0.9 0.8 0.5;
                27   12     80        1          0.0 0.0 0.0  0.9 0.8 0.5;
                28   12     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;
               ];
myHoliText=    [
                '020dke          ';
                '021sed          ';
                '022sdk          ';
                '023led          ';
                '024fsf          ';
                '025fsl          ';
                '026vdk          ';
                '027fsf          ';
                '028oei          ';
                '029jke          ';
                '030ouo          ';
                '031ase          ';
                '032wer          ';
                '033xdw          ';
                '034les          ';
                '035lhl          ';
                '036lei          ';
                '037qoa          ';
                '040ltl          ';
                '041bra          ';
                '042dom          ';
                '043gai          ';
                '044lun          ';
                '045ros          ';
                '046jos          ';
                '047sei          ';
                '048usk          ';
                '049wen          ';
                '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;
```

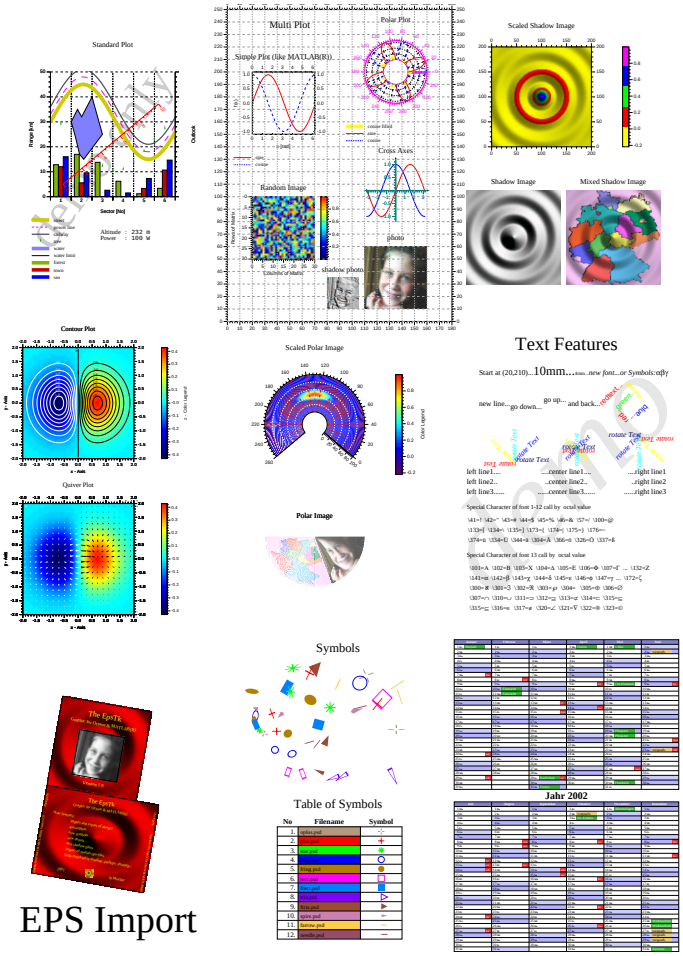
```
                50    0      5        1          1.0 1.0 1.0  0.2 0.7 0.2;
                88    0      6        1          1.0 1.0 1.0  0.2 0.7 0.2;
                98    0      7        1          1.0 1.0 1.0  0.2 0.7 0.2;
                99    0      7        1          1.0 1.0 1.0  0.2 0.7 0.2;
               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('demo10.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;
```

### A.1.11 edemo11.m

```
eopen('demo11.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,'demo2.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,'demo7.eps');
einseps(5,20,'demo9.eps',0.2,0.2,-10);
esubeps(3,3,3,2,'demo8.eps');
esubeps(3,3,3,3,'demo10.eps');
eclose                                   % close eps-file
eview                                    % start ghostview with eps-file
```




## A.2 Character Code

### Character Codes

Character codes of font 1-12 call by octal value

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| \0= | \1= | \2= | \3= | \4= | \5= | \6= | \7= | \10= | \11= | \12= | \13= | \14= | \15= | \16= |
| \17= | \20= | \21= | \22= | \23= | \24= | \25= | \26= | \27= | \30= | \31= | \32= | \33= | \34= | \35= |
| \36= | \37= | \40= | \41=! | \42=" | \43=# | \44=$ | \45=% | \46=& | \47=' | \50=( | \51=) | \52=* | \53=+ | \54=, |
| \55=- | \56=. | \57=/ | \60=0 | \61=1 | \62=2 | \63=3 | \64=4 | \65=5 | \66=6 | \67=7 | \70=8 | \71=9 | \72=: | \73=; |
| \74=< | \75== | \76=> | \77=? | \100=@ | \101=A | \102=B | \103=C | \104=D | \105=E | \106=F | \107=G | \110=H | \111=I | \112=J |
| \113=K | \114=L | \115=M | \116=N | \117=O | \120=P | \121=Q | \122=R | \123=S | \124=T | \125=U | \126=V | \127=W | \130=X | \131=Y |
| \132=Z | \133=[ | \134=\ | \135=] | \136=^ | \137=_ | \140=‘ | \141=a | \142=b | \143=c | \144=d | \145=e | \146=f | \147=g | \150=h |
| \151=i | \152=j | \153=k | \154=l | \155=m | \156=n | \157=o | \160=p | \161=q | \162=r | \163=s | \164=t | \165=u | \166=v | \167=w |
| \170=x | \171=y | \172=z | \173={ | \174=\| | \175=} | \176=~ | \177= | \200= | \201= | \202= | \203= | \204= | \205= | \206= |
| \207= | \210= | \211= | \212= | \213= | \214= | \215= | \216= | \217= | \220= | \221= | \222= | \223= | \224= | \225= |
| \226= | \227= | \230= | \231= | \232= | \233= | \234= | \235= | \236= | \237= | \240= | \241=¡ | \242=¢ | \243=£ | \244=⁄ |
| \245=¥ | \246=ƒ | \247=§ | \250=¤ | \251=' | \252=“ | \253=« | \254=‹ | \255=› | \256=fi | \257=fl | \260=° | \261=– | \262=† | \263=‡ |
| \264=· | \265=µ | \266=¶ | \267=• | \270=‚ | \271=„ | \272=” | \273=» | \274=… | \275=‰ | \276=¾ | \277=¿ | \300=À | \301=´ | \302=ˆ |
| \303=˜ | \304=Ā | \305=¯ | \306=˘ | \307=˙ | \310=¨ | \311=É | \312=˚ | \313=¸ | \314=Ì | \315=˝ | \316=˛ | \317=ˇ | \320=— | \321=Ñ |
| \322=Ò | \323=Ó | \324=Ô | \325=Õ | \326=Ö | \327=× | \330=Ø | \331=Ù | \332=Ú | \333=Û | \334=Ü | \335=Ý | \336=Þ | \337=ß | \340=à |
| \341=Æ | \342=â | \343=ª | \344=ä | \345=å | \346=æ | \347=ç | \350=Ł | \351=Ø | \352=Œ | \353=º | \354=ì | \355=í | \356=î | \357=ï |
| \360=ð | \361=æ | \362=ò | \363=ó | \364=ô | \365=ı | \366=ö | \367=÷ | \370=l | \371=ø | \372=œ | \373=ß | \374=ü | \375=ý | \376=þ |

Character Codes of font 13 call by octal value

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| \0= | \1= | \2= | \3= | \4= | \5= | \6= | \7= | \10= | \11= | \12= | \13= | \14= | \15= | \16= |
| \17= | \20= | \21= | \22= | \23= | \24= | \25= | \26= | \27= | \30= | \31= | \32= | \33= | \34= | \35= |
| \36= | \37= | \40= | \41=! | \42=∀ | \43=# | \44=∃ | \45=% | \46=& | \47=∋ | \50=( | \51=) | \52=* | \53=+ | \54=, |
| \55=− | \56=. | \57=/ | \60=0 | \61=1 | \62=2 | \63=3 | \64=4 | \65=5 | \66=6 | \67=7 | \70=8 | \71=9 | \72=: | \73=; |
| \74=< | \75== | \76=> | \77=? | \100=≅ | \101=A | \102=B | \103=X | \104=Δ | \105=E | \106=Φ | \107=Γ | \110=H | \111=I | \112=ϑ |
| \113=K | \114=Λ | \115=M | \116=N | \117=O | \120=Π | \121=Θ | \122=P | \123=Σ | \124=T | \125=Y | \126=ς | \127=Ω | \130=Ξ | \131=Ψ |
| \132=Z | \133=[ | \134=∴ | \135=] | \136=⊥ | \137=_ | \140= ¯ | \141=α | \142=β | \143=χ | \144=δ | \145=ε | \146=ϕ | \147=γ | \150=η |
| \151=ι | \152=φ | \153=κ | \154=λ | \155=μ | \156=ν | \157=ο | \160=π | \161=θ | \162=ρ | \163=σ | \164=τ | \165=υ | \166=ϖ | \167=ω |
| \170=ξ | \171=ψ | \172=ζ | \173={ | \174=\| | \175=} | \176=~ | \177= | \200= | \201= | \202= | \203= | \204= | \205= | \206= |
| \207= | \210= | \211= | \212= | \213= | \214= | \215= | \216= | \217= | \220= | \221= | \222= | \223= | \224= | \225= |
| \226= | \227= | \230= | \231= | \232= | \233= | \234= | \235= | \236= | \237= | \240=€ | \241=ϒ | \242=′ | \243=≤ | \244=⁄ |
| \245=∞ | \246=ƒ | \247=♣ | \250=♦ | \251=♥ | \252=♠ | \253=↔ | \254=← | \255=↑ | \256=→ | \257=↓ | \260=° | \261=± | \262=″ | \263=≥ |
| \264=× | \265=∝ | \266=∂ | \267=• | \270=÷ | \271=≠ | \272=≡ | \273=≈ | \274=… | \275=\| | \276=— | \277=↵ | \300=ℵ | \301=ℑ | \302=ℜ |
| \303=℘ | \304= | \305=⊕ | \306=∅ | \307=∩ | \310=∪ | \311=⊃ | \312=⊇ | \313=⊄ | \314=⊂ | \315=⊆ | \316=∈ | \317=∉ | \320=∠ | \321=∇ |
| \322=® | \323=© | \324=™ | \325=∏ | \326= | \327=· | \330=¬ | \331=∧ | \332=∨ | \333=⇔ | \334= | \335=⇑ | \336=⇒ | \337= | \340=◊ |
| \341=⟨ | \342=® | \343=© | \344= | \345=∑ | \346=⎧ | \347=⎨ | \350=⎩ | \351=⎧ | \352=⎨ | \353=⎩ | \354=⎡ | \355=⎨ | \356=⎢ | \357=⎣ |
| \360= | \361=⎫ | \362=⎬ | \363=⎭ | \364=⎢ | \365=⎭ | \366= | \367=⎢ | \370=⎭ | \371=⎤ | \372=⎢ | \373=⎦ | \374= | \375=⎬ | \376=⎦ |