fc tools Octave package, User's Guide*

version 0.1.0

François Cuvelier†

March 28, 2025

Abstract

The fc tools Octave package contains some basic tools used in my other packages.

*LaTeX manual, revision 0.1.0, compiled with Octave 9.4.0, and package `fc-tools`[0.1.0].

†LAGA, UMR 7539, CNRS, Université Sorbonne Paris Nord - Institut Galilée, 99 Avenue J-B Clément, F-93430 Villetaneuse, France, cuvelier@math.univ-paris13.fr

0 Contents

1 graphics module

Here is a resume of the functions of the `fc_tools.graphics` namespace:

- `CMRcolormap` returns a black and white map to be used with colormap function
- `ColumnLegend` just an interface to `fc_tools.graphics.columnlegend.columnlegend` function.
- `PlaneCoefs` compute coefficients of a plan in dimension 3.
- `SaveAllFigsAsFiles` Save all figures in files.
- `colorbarIso` draw a specific colorbar used with isovalues.
- `colormap_default` returns name of the default colormap.
- `display_rgb` display colors of a n-by-3 RGB colors array with their names if available.
- `fontsize` change fontsize on all figures.
- `is2Dview` test if actual view of the current figure is a 2D-view.
- `isCurrentFigureVisible` returns true if the current figure is visible, false otherwise.
- `markers` returns list of avaible markers as a cell array of char.
- `plot_RGBcolors` display colors of a n-by-3 RGB colors array with their names if available.
- `selectColors` returns N colors that are maximally perceptually distinct.

See `fc_tools.help('namespace','graphics')` for this resume.
One can used `help fc_tools.graphics.<funname>` to obtain more help where `<funname>` is the name of the function.

The avaible sub-namespaces are :

- `fc_tools.graphics.columnlegend`
- `fc_tools.graphics.crop`
- `fc_tools.graphics.gptoolbox`
- `fc_tools.graphics.monitors`
- `fc_tools.graphics.vfield3`
- `fc_tools.graphics.xcolor`

1.1 main functions

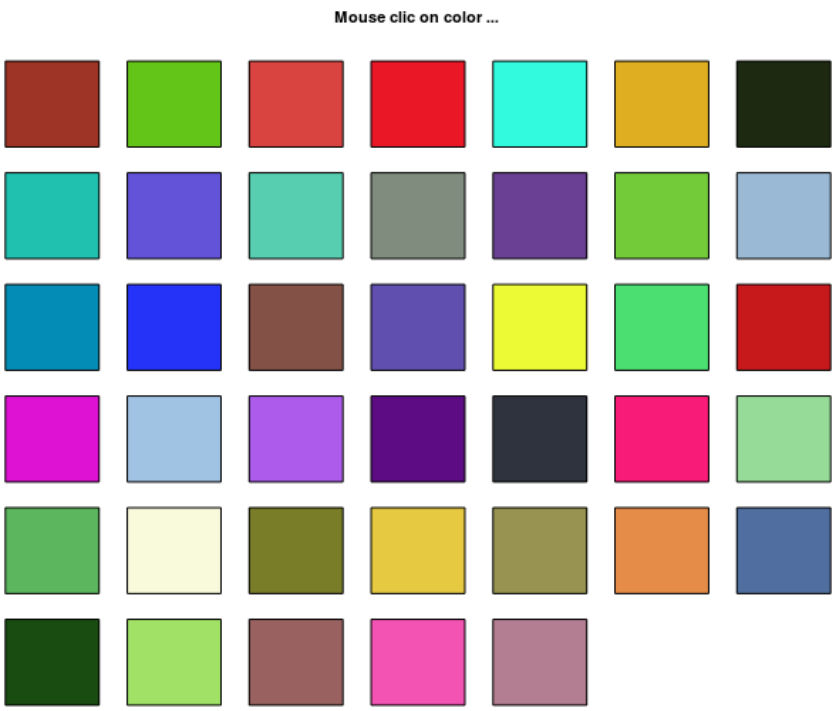
1.1.1 `fc_tools.graphics.display_rgb` function

The `fc_tools.graphics.display_rgb` displays colors of a n-by-3 RGB colors array.

Syntaxe

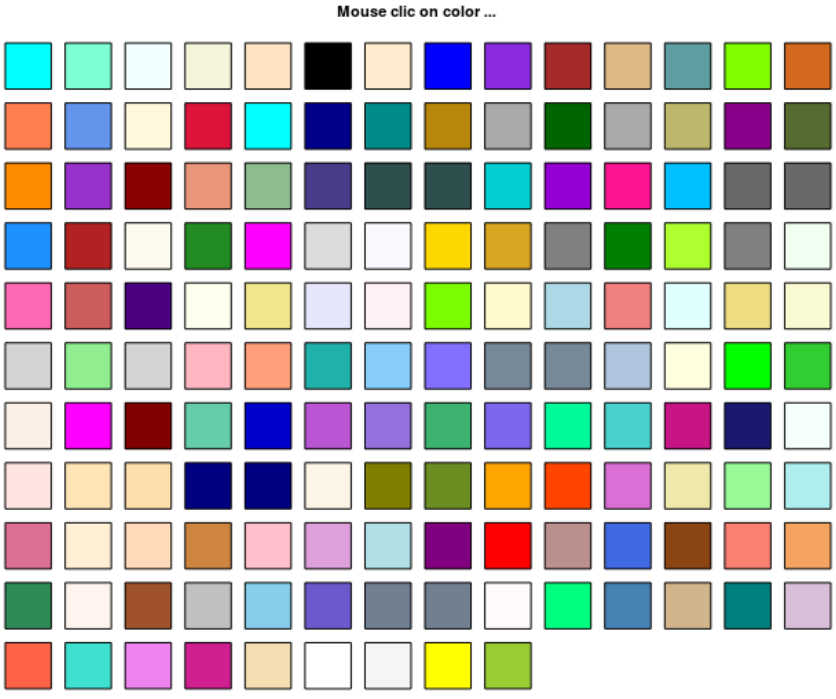
```
fc_tools.graphics.display_rgb(rgb)
fc_tools.graphics.display_rgb(rgb, names)
```

When mouse click on a specific color, its RGB color is printed with its index in the RGB colors array `rgb` and with its name if the second optional input argument `names` is given (1-by-n cell array of text).




```
fc_tools.graphics.display_rgb(rand(40,3))
```

Listing 1: `fc_tools.graphics.display_rgb` function




```
[names,colors]=fc_tools.graphics.xcolor.get_from_theme('svg');
fc_tools.graphics.display_rgb(colors,names)
```

Listing 2: `fc_tools.graphics.display_rgb_svg` function

1.1.2 `fc_tools.graphics.selectColors` function

The `fc_tools.graphics.selectColors` function returns colors that are maximally perceptually distinct without using the Image Processing Toolbox.

This function is inspired by the function `select_colors` (or `distinguishable_colors`) of *Timothy E. Holy* which uses the Image Processing Toolbox of Matlab.

Syntaxe

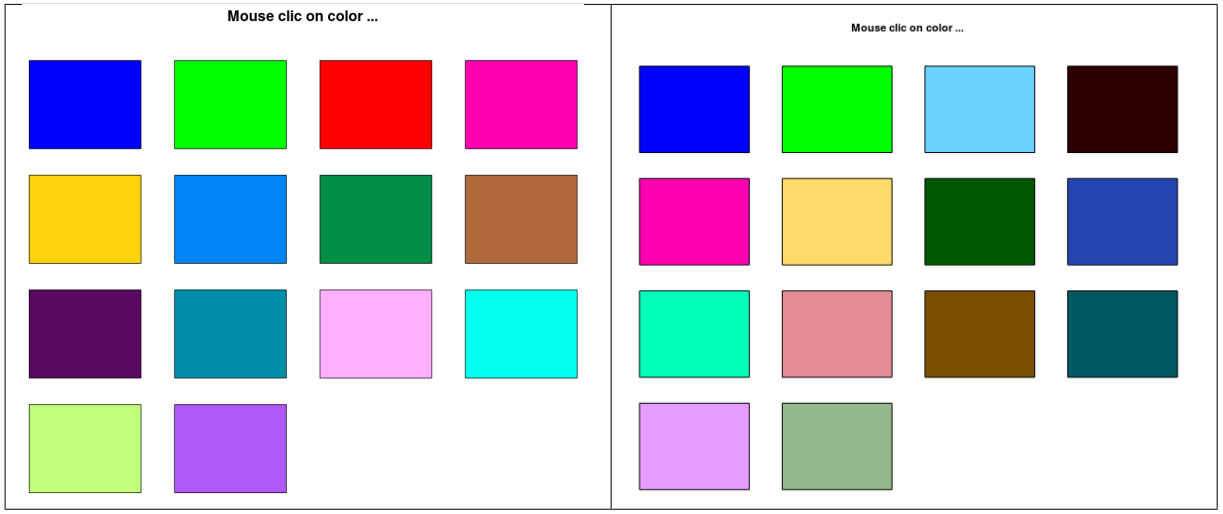
```
colors=fc_tools.graphics.selectColors(N)
colors=fc_tools.graphics.selectColors(N, key, value)
```

Description

`colors=fc_tools.graphics.selectColors(N)` Returns `N` colors that are maximally perceptually distinct as a `N`-by-3 RGB colors array.

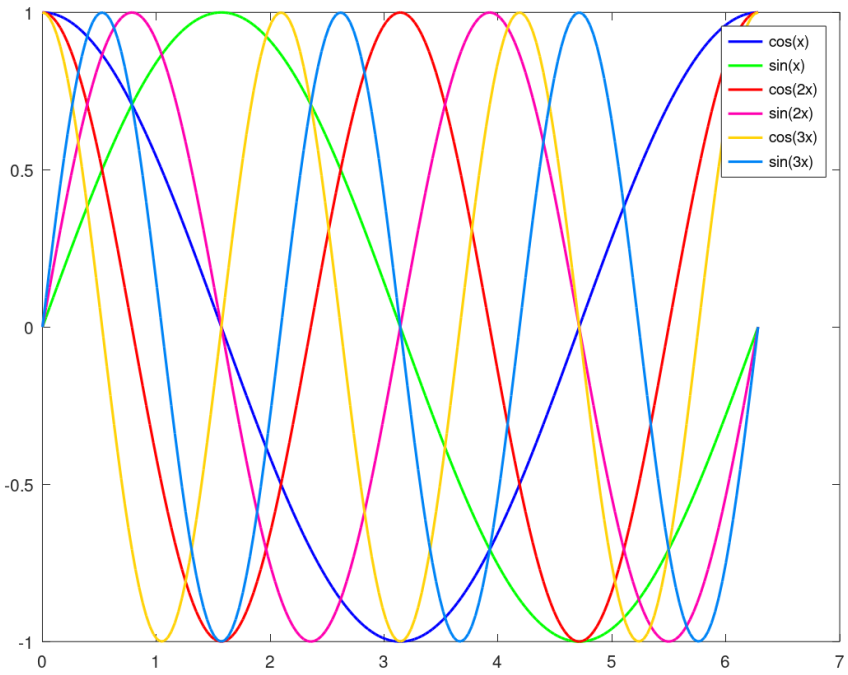
`colors=fc_tools.graphics.selectColors(N, Name, Value)` specifies function options using one or more `Name`,`Value` pair arguments. Options are

- `'background'` : the `N` colors selected will be as far as possible from the colors specified by this options as a n-by-3 RGB colors array. Default is `[1 1 1; 0 0 0]`
- `'func'` : To specify an other function for converting RGB colors to LAB colors. Default is local `RGB2LAB` function.




```
figure(1)
colors1=fc_tools.graphics.selectColors(14);
fc_tools.graphics.display_rgb(colors1)
figure(2)
colors2=fc_tools.graphics.selectColors(14,'background',[1 0 0]);
fc_tools.graphics.display_rgb(colors2)
```

Listing 3: `fc_tools.graphics.xcolor.selectColors` function




```
colors=fc_tools.graphics.selectColors(6);
x=0:pi/100:2*pi;
figure(1)
plot(x,cos(x),'color',colors(1,:),'Linewidth',1.5)
hold on
plot(x,sin(x),'color',colors(2,:),'Linewidth',1.5)
plot(x,cos(2*x),'color',colors(3,:),'Linewidth',1.5)
plot(x,sin(2*x),'color',colors(4,:),'Linewidth',1.5)
plot(x,cos(3*x),'color',colors(5,:),'Linewidth',1.5)
plot(x,sin(3*x),'color',colors(6,:),'Linewidth',1.5)
legend('cos(x)','sin(x)','cos(2x)','sin(2x)','cos(3x)','sin(3x)')
```

Listing 4: `fc_tools.graphics.xcolor.selectColors` function

1.1.3 `fc_tools.graphics.SaveAllFigsAsFiles` function

The `fc_tools.graphics.SaveAllFigsAsFiles` saves all figures as files.

Syntaxe

```
fc_tools.graphics.SaveAllFigsAsFiles(basename)
fc_tools.graphics.SaveAllFigsAsFiles(file, key, value, ...)
```

Description

`fc_tools.graphics.SaveAllFigsAsFiles(basename)` save each figure in the file

`[basename,'_fig',fignumber]`

of the current directory where `fignumber` is the number of the figure to be saved.

`fc_tools.graphics.SaveAllFigsAsFiles(file, key, value, ...)` specifies function options using one or more `Name`,`Value` pair arguments. Options are

- `'format'` : to specify the file format. `Value` could be `'epsc'` (default), `'pdf'`, `'png'` or `'pdflatex'`.
- `'dir'` : to specify the directory (default `'.'`). the directory is created if it does not exist.
- `'verbose'` : if `true`, prints file names. Default is `false`.
- `'tag'` : if `true` each figure is saved in the file:

  `[basename,'_fig',fignumber,'_',software,version]`

  where `software` is Octave and `version` is its release. Default is `false`.
- `'size'` : to specify size of the image. Default is `[800,600]`.

1.2 xcolor submodule

Here is a resume of the functions of the `fc_tools.graphics.xcolor` namespace:

- `X11` returns X11 colors as a cell array of names and a RGB array
- `display_colors` displays all colors in all themes
- `display_rgb_names` displays colors of a n-by-3 RGB colors array with their names.
- `display_theme` displays colors of a color theme
- `find_nearest_colors` finds n nearest colors in a theme color from an RGB color
- `fullX11` returns X11 (full) colors as a cell array of names and a RGB array
- `get_from_theme` returns colors as a cell array of names and a RGB array from a theme color.
- `get_themes` returns all theme colors in a cell array.
- `matlab` returns matlab colors as a cell array of names and a RGB array
- `svg` returns svg colors as a cell array of names and a RGB array
- `val2rgb` transforms name of usual color names (X11, svg, wiki, matlab) as a RGB 1-by-3 array (values in [0,1]).
- `wiki` returns wiki colors as a cell array of names and a RGB array

See `fc_tools.help('namespace','graphics.xcolor')` for this resume.
One can used `help fc_tools.graphics.xcolor.<funname>` to obtain more help where `<funname>` is the name of the function.

1.2.1 `fc_tools.graphics.xcolor.get_themes` function

The `fc_tools.graphics.xcolor.get themes` function returns available color themes.

Listing 5: : example using `fc_tools.graphics.xcolor.get_themes` functions

```
themes=fc_tools.graphics.xcolor.get_themes()
```

Output

```
themes =
{
  [1,1] = matlab
  [1,2] = svg
  [1,3] = X11
  [1,4] = fullX11
  [1,5] = wiki
}
```

1.2.2 `fc_tools.graphics.xcolor.val2rgb` function

The `fc_tools.graphics.xcolor.val2rgb` transform name(s) of usual colors (X11, svg, wiki, matlab) color as RGB triplet(s) (values in [0,1]).

Syntaxe

```
color=fc_tools.graphics.xcolor.val2rgb(val)
color=fc_tools.graphics.xcolor.val2rgb(val,'theme',text)
[color,theme]=fc_tools.graphics.xcolor.val2rgb(val)
```

Listing 6: : example using fc_tools.graphics.xcolor.val2rgb functions

```
color=fc_tools.graphics.xcolor.val2rgb('Olive')
[colors,themes]=fc_tools.graphics.xcolor.val2rgb({'Orange','LightGrey','Brown'})
```

Output

```
color =

   0.5000   0.5000        0

colors =

   1.0000   0.6480        0
   0.8280   0.8280   0.8280
   0.6480   0.1650   0.1650

themes =
{
  [1,1] = svg
  [1,2] = svg
  [1,3] = svg
}
```

1.2.3 fc_tools.graphics.xcolor.get_from_theme function

The fc_tools.graphics.xcolor.get_from_theme function returns names and RGB values of an available theme colors.

Listing 7: : example using fc_tools.graphics.xcolor.get_from_theme functions

```
[name,rgb]=fc_tools.graphics.xcolor.get_from_theme('X11');
whos name rgb
k=15;
fprintf('color␣name␣:␣''%s'',␣rgb=[%.6f,%.6f,%.6f]\n',name{k},rgb(k,1),rgb(k,2),rgb(k,3))
```

Output

```
Variables visible from the current scope:

variables in scope: top scope

  Attr  Name       Size                Bytes  Class
  ====  ====       ====                =====  =====
        name       1x317                2982  cell
        rgb      317x3                  7608  double

Total is 1268 elements using 10590 bytes

color name : 'Bisque3', rgb=[0.804000,0.716000,0.620000]
```

1.2.4 fc_tools.graphics.xcolor.svg function

The fc_tools.graphics.xcolor.svg function returns names and RGB values of the 149 SVG colors.

Syntaxe

```
[name,rgb]=fc_tools.graphics.xcolor.svg()
```

name is cell array of string (color names) and rgb is 149-by-3 array (rgb values) such that the color name{i} has rgb(i,:) for rgb values.

1.2.5 fc_tools.graphics.xcolor.X11 function

The fc_tools.graphics.xcolor.X11 function returns names and RGB values of the 317 X11 colors.

Syntaxe

```
[name,rgb]=fc_tools.graphics.xcolor.X11()
```

name is cell array of string (color names) and rgb is 317-by-3 array (rgb values) such that the color name{i} has rgb(i,:) for rgb values.

1.2.6 fc_tools.graphics.xcolor.fullX11 function

The fc_tools.graphics.xcolor.fullX11 function returns names and RGB values of the 738 X11 colors.

Syntaxe

```
[name,rgb]=fc_tools.graphics.xcolor.fullX11()
```

`name` is cell array of string (color names) and `rgb` is 738-by-3 array (rgb values) such that the color `name{i}` has `rgb(i,:)` for rgb values.

1.3 monitors submodule

This module is **experimental** ...

Here is a resume of the functions of the `fc_tools.graphics.monitors` namespace:

- `autoGrid` : distributes all figures on a grid with automatically defined rows and columns.
- `autoGridSize` : returns numbers of rows and columns of a grid for a given number of figures.
- `get_locations_list` : returns a list of possible locations.
- `number` : returns the number of monitors.
- `onGrid` : distributes figures on a grid.
- `setGrid` : returns positions for a m-by-n grid of figures.
- `set_defaultGrid` : sets an saves monitor, covers and location of the default grid of figures.
- `show` : displays the position of the monitor(s) with location, size and number.

See `fc_tools.help('namespace','graphics.monitors')` for this resume.

1.3.1 `fc_tools.graphics.monitors.setGrid` function

The `fc_tools.graphics.monitors.setGrid` function returns positions for a virtual `m`-by-`n` grid of figures (as `subplot` command with axes) positioned on a selected monitor.

Syntaxe

```
G=fc_tools.graphics.monitors.setGrid(n,m)
G=fc_tools.graphics.monitors.setGrid(n,m, key,value, ...)
```

Description

`G=fc_tools.graphics.monitors.setGrid(n,m)`

Creates a 4-by-`m`-by-`n` array of positions associated with a virtual `m`-by-`n` grid with default parameters `'monitor'`, `'covers'`, and `'location'`. These default values can be set and save by using `fc_tools.graphics.monitor` function.

The 4-by-`m`-by-`n` `G` array : 4-by-m-by-n array. The (`i`,`j`) position (x, y, w, h) on the grid is `G(:,i,j)` where

- `x=G(1,i,j)` (x-position in pixel),
- `y=G(2,i,j)` (y-position in pixel),
- `w=G(3,i,j)` (width-position in pixel),
- `h=G(4,i,j)` (height-position in pixel).

`G=fc_tools.graphics.monitors.setGrid(n,m, key,value)` specifies function options using one or more `key`,`value` pair of arguments. The possible keys are

- `'monitor'` : to specify the number of the monitor on which the grid is determined. Uses `fc_tools.graphics.monit` to display monitors with their number.
- `'covers'` : value in $]0, 1]$, to specify the percentage of screen coverage of the grid.
- `'location'` : used to specify location of the grid on the selected monitor. Value must be `'center'`, `'left'`, `'west'`, `'right'`, `'east'`, `'bottomright'`, `'southeast'`, `'bottom'`, `'south'`, `'bottomleft'`, `'southwest'`, `'topleft'`, `'northwest'`, `'top'`, `'north'`, `'topright'` or `'northeast'`.

1.3.2 `fc_tools.graphics.monitors.onGrid` function

The `fc_tools.graphics.monitors.onGrid` displays figures on a virtual `m`-by-`n` grid (as `subplot` command with axes) positioned on a selected monitor.

Syntaxe

```
fc_tools.graphics.monitors.onGrid(n,m)
fc_tools.graphics.monitors.onGrid(n,m, key,value, ...)
```

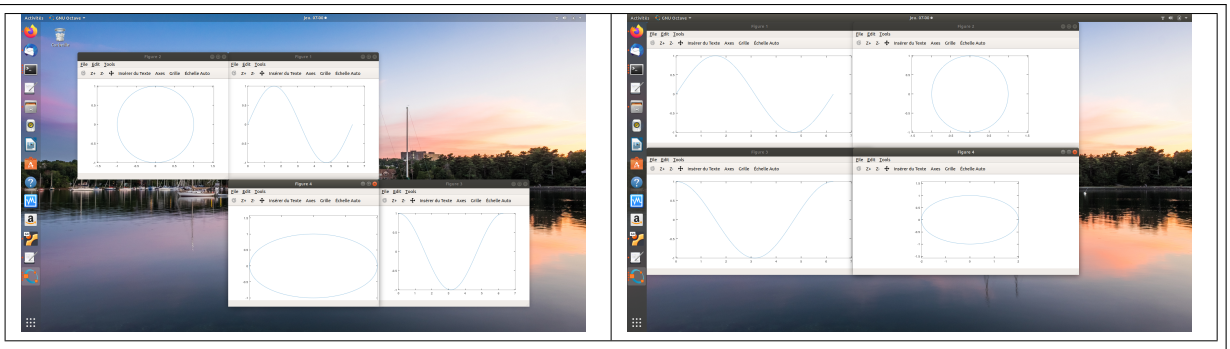
Description

`fc_tools.graphics.monitors.onGrid(n,m)`
A virtual `m`-by-`n` grid is created on the default monitor and the figures numbered from 1 to `m*n` are moved or created (if it doesn't exist) respectively to the position given by an index (default is the figure number). This index runs row-wise; all columns of the first row are numbered from left to right and so on with rows 2 to `m`.

`fc_tools.graphics.monitors.onGrid(n,m, key,value)` specifies function options using one or more `key`,`value` pair of arguments. The possible keys are

- `'figures'` : specifies the numbers of the figures to be used. Default is `1:m*n`.
- `'positions'` : specifies the index on the grid corresponding to the `'figures'` option. Default is `1:m*n`.

Other `key`,`value` pair of arguments are those of the `fc_tools.graphics.monitors.setGrid` function and its keys are `'monitor'`, `'covers'` and `'location'`.



```
1   x=0:pi/100:2*pi;
2   close all
3   fc_tools.graphics.monitors.onGrid(2,3,'figures',[1,3], 'positions',[2,6],'covers',4/5)
4   figure(1)
5   plot(x,sin(x))
6   figure(3)
7   plot(x,cos(x))
8   fc_tools.graphics.monitors.onGrid(2,3,'figures',[2,4], 'positions',[1,5],'covers',4/5)
9   figure(2)
10  plot(sin(x),cos(x))
11  axis equal
12  figure(4)
13  plot(2*sin(x),cos(x))
14  axis equal
15  fprintf('waiting␣2s␣...\n');pause(2)
16  fprintf('Moving␣the␣4␣figures␣on␣topleft/northwest␣and␣covers␣2/3␣of␣the␣screen\n')
17  fc_tools.graphics.monitors.onGrid(2,2,'figures',1:4,'covers',4/5, 'location','NorthWest')
```

Listing 8: Using `fc_tools.graphics.monitors.onGrid` function, part of `fc_tools.graphics.monitors.demos.demo02` function. Figures are screenshot taken at line 15 (left) and after last line (right).

1.3.3 `fc_tools.graphics.monitors.show` function

The `fc_tools.graphics.monitors.show` displays monitors with their resolution, position and number on a figure. When arguments are provided, they are those of the `fc_tools.graphics.monitors.onGrid` function and then the grid is also drawn with the indices of the positions of the grid elements.

Syntaxe

```
fc_tools.graphics.monitors.show()
fc_tools.graphics.monitors.show(n,m)
fc_tools.graphics.monitors.show(n,m, key,value, ...)
```

Description

`fc_tools.graphics.monitors.show()`

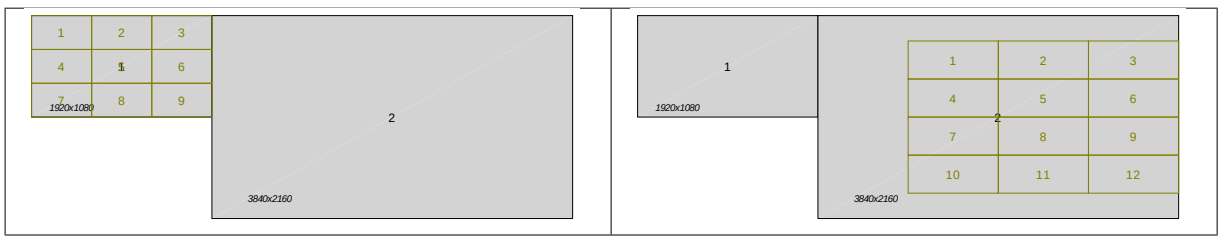
Displays monitors with their resolution, position and number on a figure.

`fc_tools.graphics.monitors.show(n,m)`

Also displays the `m`-by-`n` virtual grid on the figure created by the `fc_tools.graphics.monitors.show()` command to to preview positions of figures created or moved by the `fc_tools.graphics.monitors.onGrid(n,m)` command.

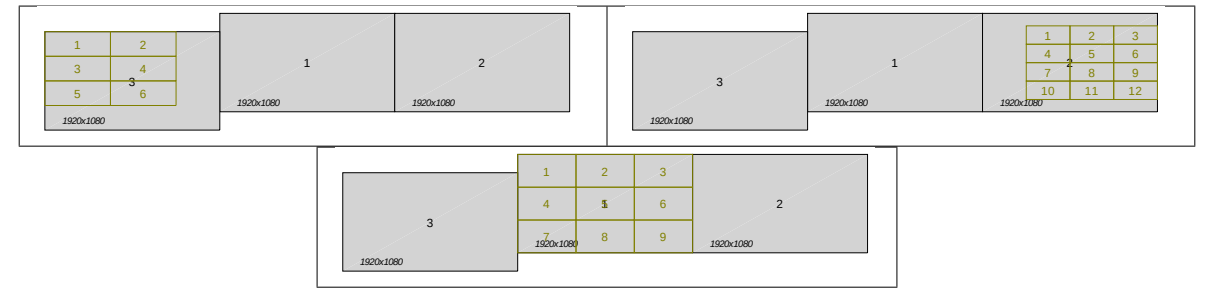
`fc_tools.graphics.monitors.show(n,m, key,value)`

Specifies function options using one or more `key`,`value` pair arguments.
Options are those of the `fc_tools.graphics.monitors.setGrid` function.




```
close all
nM=fc_tools.graphics.monitors.number();
figure(1)
fc_tools.graphics.monitors.show(3,3,'monitor',1,'covers',1)
if nM>=2
  figure(2)
  fc_tools.graphics.monitors.show(4,3,'monitor',2, 'location','east','covers',3/4)
end
if nM>=3
  figure(3)
  fc_tools.graphics.monitors.show(3,2,'monitor',3, 'location','topleft','covers',3/4)
end
fc_tools.graphics.monitors.onGrid(2,2,'figures',1:min([nM,3]))
```

Listing 9: Using `fc_tools.graphics.monitors.show` function, part of `fc_tools.graphics.monitors.demos.demo03` function on a computer with two monitors.




```
close all
nM=fc_tools.graphics.monitors.number();
figure(1)
fc_tools.graphics.monitors.show(3,3,'monitor',1,'covers',1)
if nM>=2
  figure(2)
  fc_tools.graphics.monitors.show(4,3,'monitor',2, 'location','east','covers',3/4)
end
if nM>=3
  figure(3)
  fc_tools.graphics.monitors.show(3,2,'monitor',3, 'location','topleft','covers',3/4)
end
```

Listing 10: Using `fc_tools.graphics.monitors.show` function, part of `fc_tools.graphics.monitors.demos.demo03` function on a computer with three monitors.

1.3.4 `fc_tools.graphics.monitors.autoGrid` function

The `fc_tools.graphics.monitors.autoGrid` automatically sets the numbers of rows and columns of the grid in function of the number of figures obtain with `fc_tools.graphics.getNbfigs` and distributes the figures on the grid with default parameters: `'monitor'`, `'covers'`, and `'location'`.

Syntaxe

```
fc_tools.graphics.monitors.autoGrid()
fc_tools.graphics.monitors.autoGrid(key,value, ...)
```

Optional `key`/`value` pairs can be used to modify default parameters and these as the one of the `fc_tools.graphics.monitors` function.

1.4 gptoolbox submodule

This submodule contains some files of the **gptoolbox** from *Alec Jacobson* (see `https://github.com/alecjacobson/gptoolbox`)

1.5 crop submodule

This submodule contains the function `crop` from *Andrew Bliss*.

1.6 vfield3 submodule

This submodule contains the function `vfield3` from *M MA* (see `https://www.mathworks.com/matlabcentral/fileexchange/8653-vfield3`)

2 utils module

Here is a resume of the functions of the `fc_tools.utils` namespace:

- `fc_tools.utils.deleteCellOptions` deletes key/value pairs in a cell array
- `fc_tools.utils.disp_object` displays properties of an object or field of a structure.
- `fc_tools.utils.eval_m_file` evaluates a .m file
- `fc_tools.utils.funHandleName` returns the name of a function handle as a char array
- `fc_tools.utils.getNumVersion` returns Matlab or Octave release/version as a scalar
- `fc_tools.utils.help_methods` obtains short help or full help on methods of an object
- `fc_tools.utils.help_namespace` obtains help on namespace by displaying short help on functions and scripts
- `fc_tools.utils.inputParserUnmatched2Cell` transforms Unmatched, for inputParser object, in a cell array
- `fc_tools.utils.isOctave` returns true if running under Octave
- `fc_tools.utils.is_fcPackage` checks if a `fc_<shortname>` package/toolbox is present
- `fc_tools.utils.is_function` checks if a char array is the name of a function
- `fc_tools.utils.is_script` checks if a name is a script
- `fc_tools.utils.isfunhandle` checks if the input is a function handle
- `fc_tools.utils.line_text_delimiter` returns a text line delimiter as a char array
- `fc_tools.utils.namespace2filename` returns filename associated to a function or script in a namespace
- `fc_tools.utils.rep_by_user_dir` replaces ' /...' directory name by corresponding full path

- `fc_tools.utils.strversion2num` converts a string version in a number

See `fc_tools.help('namespace','utils')` for this resume.
One can used `help fc_tools.utils.<funname>` to obtain more help where `<funname>` is the name of the function.

3 sys module

Here is a resume of the functions of the `fc_tools.sys` namespace:

- `fc_tools.sys.create_directory(dirname)` creates a directory.
- `fc_tools.sys.getCPUinfo()` returns CPU(s) informations as a structure.
- `fc_tools.sys.getComputerName()` returns the name of the computer as a string.
- `fc_tools.sys.getOSinfo()` returns OS informations as a structure.
- `fc_tools.sys.getRAM()` returns available memory (RAM) in GB of the computer.
- `fc_tools.sys.getRelease()` returns Matlab or Octave release/version as a string.
- `fc_tools.sys.getSoftware()` returns used software (Matlab or Octave) and its release.
- `fc_tools.sys.getRelease()` returns Matlab or Octave release/version as a string.
- `fc_tools.sys.getUserDir()` returns the user home directory.
- `fc_tools.sys.info()` returns informations on current computer as a string.
- `fc_tools.sys.isfileexists()` returns true if a file exists.
- `fc_tools.sys.isfolder()` returns true if a folder exists.

See `fc_tools.help('namespace','sys')` for this resume.
One can used `help fc_tools.sys.<funname>` to obtain more help where `<funname>` is the name of the function.
In Listing 11, some examples are provided.

Listing 11: : example using `fc_tools.sys` functions

```
fprintf('RAM : %.2f GB\n',fc_tools.sys.getRAM())
CPU= fc_tools.sys.getCPUinfo()
OS=fc_tools.sys.getOSinfo()
```

Output

```
RAM : 42.85 GB
CPU =

  scalar structure containing the fields:

    name = AMD Ryzen 9 6900HX with Radeon Graphics
    nthreadspercore = 2
    ncoreperproc = 8
    nprocs = 1

OS =

  scalar structure containing the fields:

    distributor = Ubuntu
    description = Ubuntu 24.04.2 LTS
    release = 24.04
    codename = noble
    arch = x86_64
    shortname = Ubuntu
```

Informations for git maintainers of the fctools Octave package

git informations on the packages used to build this manual

```
------------------------------------------------
  name : fc-tools
   tag : 0.1.0
commit : 9ba3fed3b9336d8cd07285a85acb5a52afc3c03f
  date : 2025-03-28
  time : 11-02-27
status : 0
------------------------------------------------
```

git informations on the LaTeX package used to build this manual

```
------------------------------------------------
  name : fctools
   tag :
commit : 03d38737a795cdbf4e1a8754470e963cdfe83316
  date : 2025-01-24
  time : 09:58:52
status : 1
------------------------------------------------
```

Using the remote configuration repository:

url	ssh://lagagit/MCS/Cuvelier/Matlab/fc-config
commit	7eecec0b026c8ccd1eb2f88fdfea2c0679bbd268