fc tools Octave package, User's Guide*

version 0.0.34

François Cuvelier†

January 5, 2023

Abstract

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

*LaTeX manual, revision 0.0.34.a, compiled with Octave 7.3.0, and package `fc-tools`[0.0.34].

†LAGA, UMR 7539, CNRS, Université Paris 13 - Sorbonne Paris Cité, Université Paris 8, 99 Avenue J-B Clément, F-93430 Villetaneuse, France, cuvelier@math.univ-paris13.fr

This work was partially supported by the ANR project DEDALES under grant ANR-14-CE23-0005.

0 Contents

1 graphics module

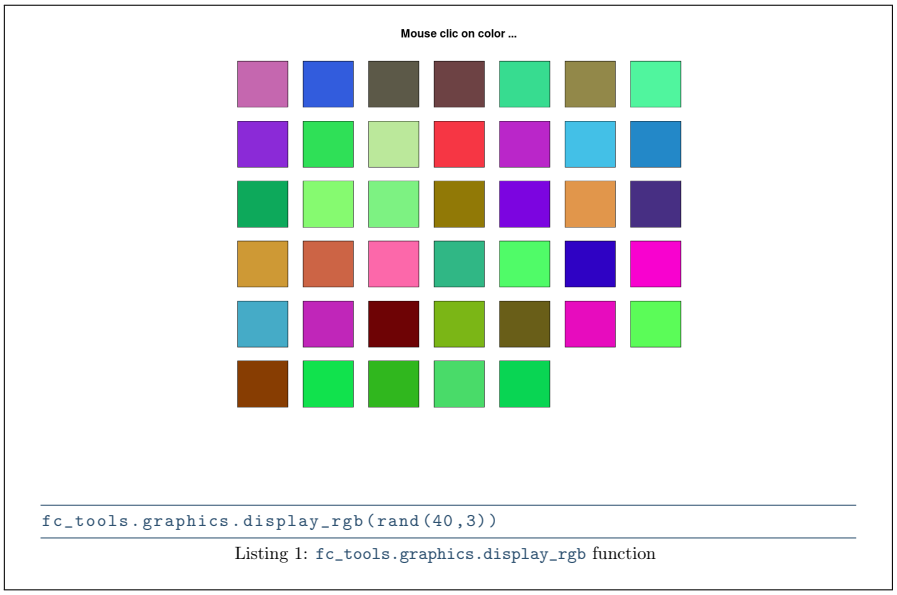
1.1 main functions

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 with their names if available.

Syntaxe

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




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

Listing 1: `fc_tools.graphics.display_rgb` 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

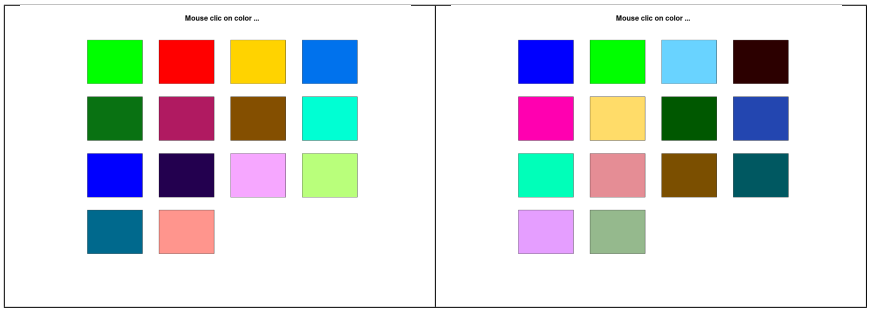
```
colors=fc_tools.graphics.selectColors(N)
colors=fc_tools.graphics.selectColors(N, ...
   key, value)
```

Description

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

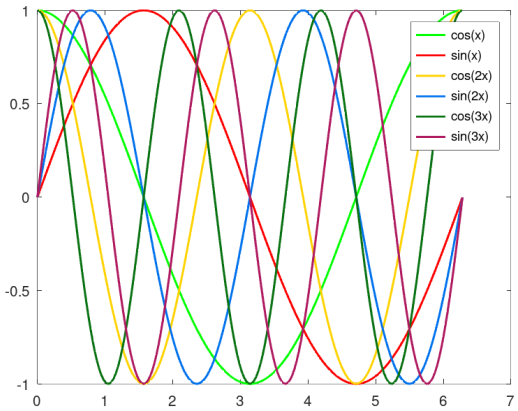
`colors=fc_tools.graphics.selectColors(N, Name, Value)` specifies function options using one or more `Name`,`Value` pair arguments. Options are

- `'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; 0.8 0.8 0.8;1,0,1]`
- `'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 2: `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 3: `fc_tools.graphics.xcolor.selectColors` function

1.1.3 `fc_tools.graphics.DisplayFigures` function

The `fc_tools.graphics.DisplayFigures` function regularly distributes the figures on the screen.

Syntaxe

```
fc_tools.graphics.DisplayFigures()
fc_tools.graphics.DisplayFigures(n)
fc_tools.graphics.DisplayFigures('nfig',n)
```

Without argument, all figures are regularly distributed on the screen. Otherwise, empty figures with numbers `1` to `n` are created and regularly distributed on the screen.

1.1.4 `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

1.2.1 `fc_tools.graphics.xcolor.get_themes` function

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

Listing 4: : 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.get_from_theme function

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

Listing 5: : 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]
```

1.2.3 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.

1.2.4 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.

1.2.5 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** ...

1.3.1 `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

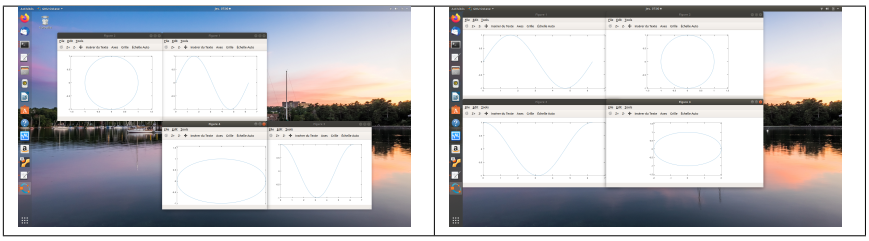
```
fc_tools.graphics.monitors.onGrid(n,m)
fc_tools.graphics.monitors.onGrid(n,m, ...
    key,value, ...)
```

Description

`fc_tools.graphics.monitors.onGrid(n,m)`
A virtual `m`-by-`n` grid is created on the first 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 arguments. Options 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`.



```
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  fc_tools.graphics.monitors.onGrid(2,2,'figures',1:4,'covers',4/5, ...
       'location','NorthWest')
```

Listing 6: 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).

1.3.2 `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()`

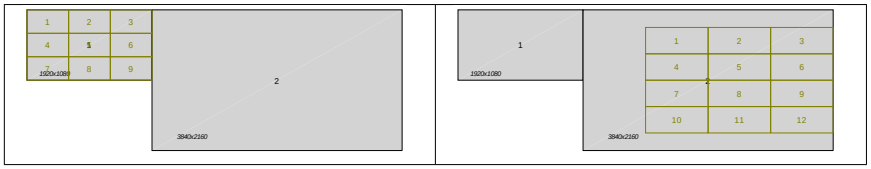
Displays monitors with their resolution, position and number on a figure.

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

Add the `m`-by-`n` 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.

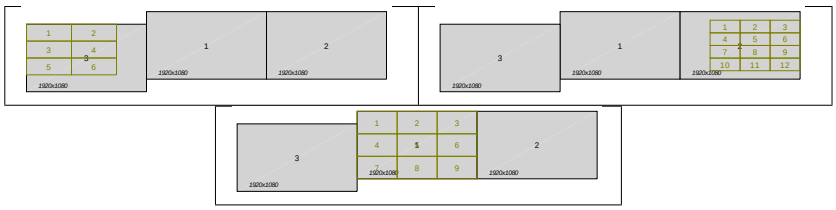
`fc_tools.graphics.monitors.show(n,m, key,value)`

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

Listing 7: 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 8: Using `fc_tools.graphics.monitors.show` function, part of `fc_tools.graphics.monitors.demos.demo03` function on a computer with three monitors.

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

- `fc_tools.utils.deleteCellOptions` deletes specified key/value pairs in a cell array of key/value pairs.
- `fc_tools.utils.isfunHandle` test if the input argument is a function handle.
- `fc_tools.utils.funHandleName` test if the input argument is a function handle.
- `fc_tools.utils.fun2str` ...

3 sys module

- `fc_tools.sys.getComputerName()` returns the name of the computer as a string.
- `fc_tools.sys.getUserName()` returns the name (login) of the current user as a string.
- `fc_tools.sys.getRAM()` returns available memory (RAM) in GB of the computer.
- `fc_tools.sys.getCPUinfo()` returns CPU(s) informations as a structure.
- `fc_tools.sys.getOSinfo()` returns OS informations as a structure.
- `fc_tools.sys.isfolder()` return true if a folder exists.
- `fc_tools.sys.isfileexists()` return true if a file exists.

In Listing 9, some examples are provided.

Listing 9: : 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 : 13.60 GB
CPU =

  scalar structure containing the fields:

    name = AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx
    nthreadspercore = 2
    ncoreperproc = 4
    nprocs = 1

OS =

  scalar structure containing the fields:

    distributor = Ubuntu
    description = Ubuntu 22.04.1 LTS
    release = 22.04
    codename = jammy
    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 : 0.0.34
commit : 625f31d5dd97131d3f3a96ef415f810c008c19c6
  date : 2023-01-04
  time : 12-59-08
status : 0
------------------------------------------------
```

git informations on the LaTeX package used to build this manual

```
------------------------------------------------
  name : fctools
   tag :
commit : c9a33ce7b4dacf90f66e5e49856e69afa1dac0a3
  date : 2022-12-17
  time : 07:57:20
status : 1
------------------------------------------------
```

Using the remote configuration repository:

url	ssh://lagagit/MCS/Cuvelier/Matlab/fc-config
commit	268d29786e42ee37b3531aed5a97635dfc1d9ada