# *berlin*

*a perceptually uniform colour-map* - www.fabiocrameri.ch/berlin

DOI 10.5281/zenodo.1243862

## Authors & Contributors

- **Fabio Crameri** - *Colour map* - contact
- **Grace Shephard** - *Conversion to .cpt files and other contributions* - contact
- **Clint Conrad** - *Wider compatibility of .cpt files* - contact
- **Casper Pranger** - *Mathematica compatibility* - contact
- **Alexis Plunder** - *Wider compatibility of .xml files* - contact
- **Krister Stræte Karlsen** - *User instruction for use with python* - contact
- **Emilia** - *Plotly versions* - contact
- **Thomas Lin Pedersen** - *The 'scico' package for use with R* - contact

## Acknowledgement

Please acknowledge the free use of the colour map.

> e.g., "The perceptually uniform colour-map ***berlin*** is used in this study to prevent visual distortion of the data (Crameri 2018a,b)."
>
> *Crameri, F. (2018a), Scientific colour-maps. Zenodo. http://doi.org/10.5281/zenodo.1243862*
>
> *Crameri, F. (2018b), Geodynamic diagnostics, scientific visualisation and StagLab 3.0, Geosci. Model Dev. Discuss., doi:10.5194/gmd-2017-328*

## Instructions

### Using the .mat Format (MatLab)

Load the colour map into MatLab, either by adding the .mat file to the MatLab search path and using the command:

```
load('berlin.mat');
```

or by specifying the full file path to the .mat file:

```
load('~/work/Colormaps/berlin.mat');
```

Then use it, for example, with:

```
figure(1)
colormap(berlin)
colorbar
```

## Using the .cpt Format (GMT)

The file berlin.cpt can be resampled for a given z-value range with the Generic Mapping Tools (GMT; http://gmt.soest.hawaii.edu/) command "makecpt".

For example to resample for an array from -2000 to 2000 in 100 increments you could generate a new file with:

```
$makecpt -Cberlin.cpt -T-2000/2000/100 > berlin_resampled.cpt
```

## Using the .ct Format (VisIt)

The file berlin.ct can imported to VisIt by placing the .ct file in the .visit directory, which can be found on macOS under e.g.,:

```
/Applications/VisIt.app/Contents/Resources/ ...
... 2.12.3/darwin-x86_64/resources/colortables
```

The colour map should appear in the built-in list after VisIt has been restarted.

## Using the .mat Format (Mathematica)

```
ColorMapSuitePath = "/Path/To/ColourMapSuite/";

ColorMapSuite[name_String] := ColorMapSuite[name, -1]
ColorMapSuite[name_String, el_] := With[{
   list =
    Transpose@{Subdivide[0, 1, 255],
      RGBColor @@@
       First@Import[
         ColorMapSuitePath <> "/" <> name <> "/" <> name <> ".mat"]}
   },
  Blend[list, {##}[[el]]] &
  ]
```

The function call `ColorMapSuite["name", i = -1]` returns a lambda function whose ith argument

is used to define color (see the Manual for `ColorFunction` for details). `"name"` should be replaced with the name (in quotes) of the color scheme, e.g. `"berlin"`. Be sure to set the variable `ColorMapSuitePath` to the path where your ColorMapSuite is installed.

General rules are:

- 1D plots of 1D functions/data: no (default) argument *i* suffices
- 2D plots of 2D functions/data: no (default) argument *i* suffices
- 3D plots of 2D functions/data: use *i* = 3
- 3D plots of 3D functions/data: use *i* = 4 (results might be worse than default Mathematica color functions, possibly due to lack of surface normal mapping)

```
ContourPlot[Sin[x] Sin[y], {x, 0, 2 Pi},
{y, 0, 2 Pi}, ColorFunction → ColorMapSuite["berlin"]]
```

## Using the .txt Format (Python)

#### Step 1: Load colour-map data

Load the colour-map data into Python using `numpy.loadtxt()`:

```
import numpy as np
cm_data = np.loadtxt("berlin.txt")
```

#### Step 2: Set up colour map

Use `matplotlib.colors.LinearSegmentedColormap()` to create a colour map that can be used with matplotlib.

```
from matplotlib.colors import LinearSegmentedColormap
berlin_map = LinearSegmentedColormap.from_list('berlin', cm_data)
```

#### Complete example:

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import LinearSegmentedColormap

cm_data = np.loadtxt("berlin_RGB(0-1).txt")
berlin_map = LinearSegmentedColormap.from_list('berlin', cm_data)

x = np.linspace(0, 100, 100)[None, :]
plt.imshow(x, aspect='auto',cmap=berlin_map)
plt.axis('off')
plt.show()
```

### Using the .py Format (plotly)

Plotly versions of the scientific colour-maps are provided by Emilia are available at https://github.com/empet/scientific-colorscales.

The plotly scientific colour-maps (see the file `scicolorscales.py`) were created by converting the provided .py file of each colour map.

Direct applications and some scientific tests are illustrated in this Jupyter Notebook: http://nbviewer.jupyter.org/github/empet/scientific-colorscales/blob/master/Tests-for-scientific-colorscales.ipynb.

### Using the scico package (R)

`scico` (https://travis-ci.org/thomasp85/scico; pronounced as "psycho") is a small package developed by Thomas Lin Pedersen that provides access to the scientific colour-maps within R. It provides scales for `ggplot2` without requiring `ggplot2` to be installed.

`scico` can be installed from CRAN with `install.packages('scico')`. If you want the development version then install directly from GitHub:

```
# install.packages("devtools")
devtools::install_github("thomasp85/scico")
```

For further details and user instructions are included in a README file within `scico`.

## References

Included colour-map diagnostics are based on:

- *Kovesi (2015), Good Colour Maps: How to Design Them, CoRR, abs/1509.03700, http://arxiv.org/abs/1509.03700* and related MatLab functions available at https://www.peterkovesi.com/matlabfns/index.html#colour.

For further details see:

- *Crameri, F. (2018), Geodynamic diagnostics, scientific visualisation and StagLab 3.0, Geosci. Model Dev. Discuss., doi:10.5194/gmd-2017-328*

## License

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.