# @prism project

Christophe BAL

9 Nov 2025 – Version 1.2.1

The `@prism` project [1] provides small size color palettes that can be used to create expressive color maps for graphics in different contexts.

---

## Last changes

1.2.1
2025-11-09

### Fix.

- Equal palettes: the floating point equality uses now a correct tolerance.

### Break.

- Palettes: the extra `Greys` has been removed (it is equal to `Grays`).

### New.

- Similar palettes: two PDF files show similar palettes in standard and black modes (semi-automated process used).

### Update.

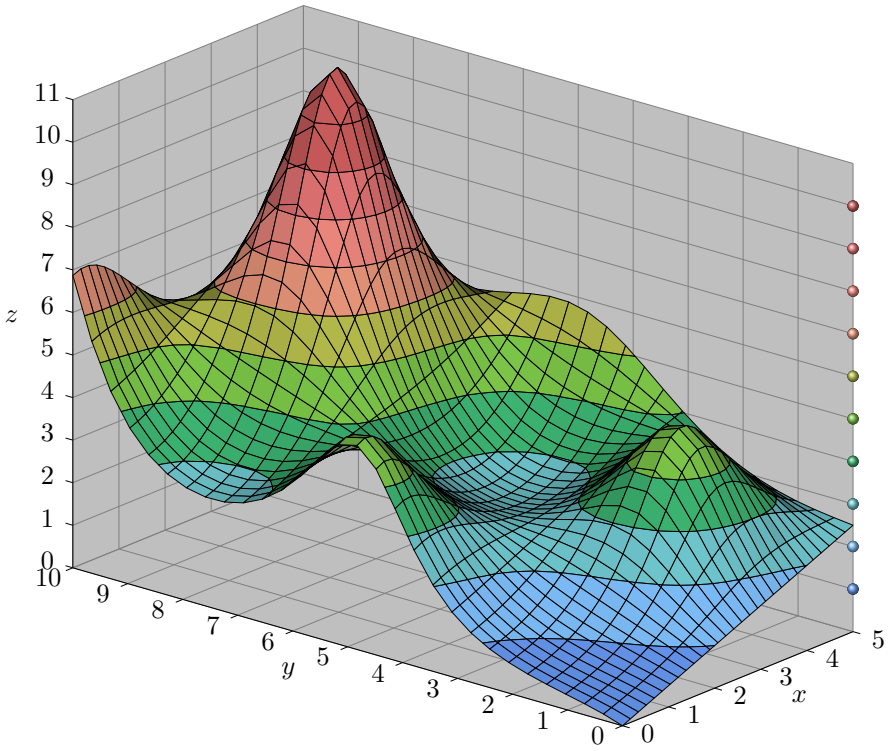
- `luadraw` product: the associative array `palNames` has been added for compatibility reasons with the `luadraw` package.
- `BlindFish` palette: the last color variation has been made smoother (`luadraw` process used).

[1] The name comes from "***@ · esthetic P · roducts*** *for* ***R · epresenting I · nformative S · cientific M · aps***". This name is a double play on words: [1] a prism splits light into an informative spectrum, symbolizing how data are decomposed into meaningful color, and [2] "@" read as "at" indicates where the light meets the prism to be broken down into an informative spectrum.

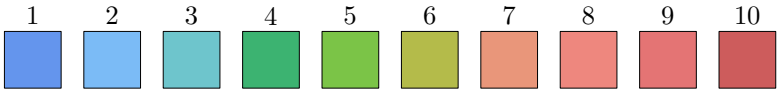
# Contents

# I. Motivations

Originally, this project was born out of a desire to enhance `luadraw` with a set of color palettes to easily produce something like the following 3D plot.



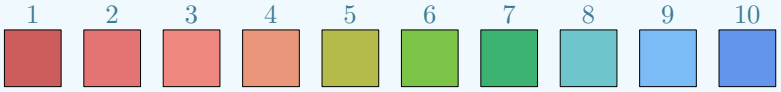

Technically, a finite list of colors is provided to `luadraw` which then uses linear interpolation to calculate the intermediate colors. In the previous case, the finite color palette used is defined as follows.



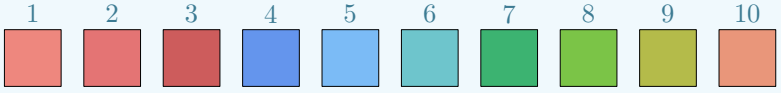

Using this palette, `luadraw` is able to produce the following spectrum, allowing us to create the graph above.

**Note.**

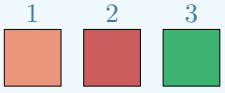
*Using the* `luadraw` *implementation of* `@prism`*, see the section V-2, we can create the palettes below made from the previous one named* `'GeoRainbow'`*. Each instruction used is given below each palette.*




```
getPal('GeoRainbow', {reverse = true})
```

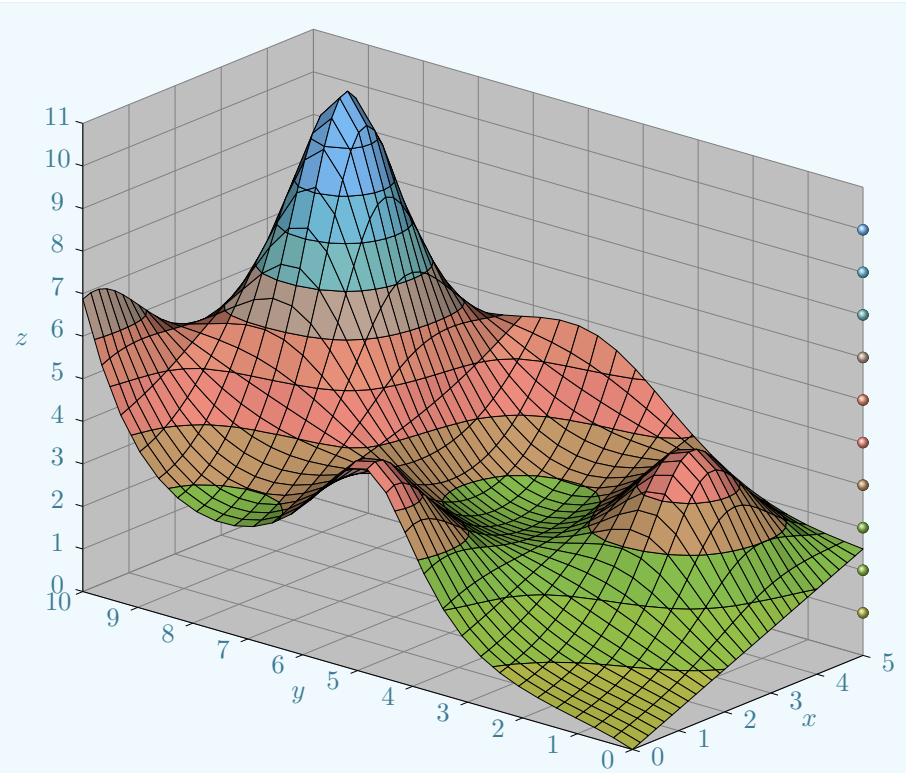



```
getPal('GeoRainbow', {shift = 3})
```




```
getPal('GeoRainbow', {extract = {7, 10, 4}})
```

*This features provide remarkable creative flexibility: with the same surface as before, but using the setting* `getPal('GeoRainbow', {extract = {2, 3, 7, 8, 5, 6}, reverse = true})` *instead of* `getPal('GeoRainbow')`*, we instantly change the visual tone, shifting from a seaside feel to a snow-covered world.*




## II. Where do the color palettes come from?

Most color palettes are obtained from `Matplotlib` and `Scientific Coulour Maps` by segmenting their color maps into 10 values.[2] We retain only palettes that comply with the following rules.

- **No repetition.** Some `Matplotlib` palettes are duplicated,[3] in which case we keep the first one in lexicographical order.
- **No reversed versions.** Unlike `Matplotlib`,[4] `@prism` never includes reversed palettes as fixed data.

In addition to `Matplotlib` and `Scientific Coulour Maps` palettes, `@prism` includes some original creations.

We list below the palettes ignored due to duplication.[5] The symbol $\boxed{=}$ indicates equality, $\boxed{\rightleftharpoons}$ indicates reversal, and the rightmost palette is the one retained in `@prism`.

- `Matplotlib`

| | | |
|---|---|---|
| `GistGray` | $\rightleftharpoons$ | `Binary` |
| `GistGrey` | $\rightleftharpoons$ | `Binary` |
| `GistYarg` | $=$ | `Binary` |
| `GistYerg` | $=$ | `Binary` |
| `Gray` | $\rightleftharpoons$ | `Binary` |
| `Grey` | $\rightleftharpoons$ | `Binary` |
| `Greys` | $=$ | `Grays` |

**Note.**

*Adding new palettes to `@prism` is straightforward (no coding skills required). See section VI-2 to get started.*

[2] `Asymptote` is also used, but currently offers nothing beyond `Matplotlib`, despite different implementations.
[3] Likely for historical reasons.
[4] Most `Matplotlib` color maps have a reversed version named with the `_r` suffix, possibly for performance reasons.
[5] Recall that `Matplotlib` reversed color maps (with the `_r` suffix) are systematically excluded and therefore not shown here.

# III. Reuse from...

## 1. Matplotlib

Here are the key points to remember when using palettes similar to those offered by Matplotlib.

1. `@prism` uses standardized `CamelCase` notation. Therefore, Matplotlib palette names such as `berlin` and `gist_heat` become `Berlin` and `GistHeat` respectively.
2. Palettes with a name ending with the `_r` suffix (reversed color order) are not included in `@prism`. However, the `@prism` implementations provide methods to easily obtain reversed palettes, sub-palettes, and color-shifted palettes.

## 2. Asymptote and Scientific Coulour Maps

Simply apply the naming standardization explained in the previous section (see point 1).

# IV. How to choose a palette?

Two methods are available to find the ideal palette.

1. The documents `showcase-en-std.pdf` (light theme) and `showcase-en-dark.pdf` (dark theme) present use cases for each palette.
2. Appendix 1 page 10 presents all palettes organized by theme with a visualization of their color spectrum.

**Note.**

*Appendix 2 page 19 groups visually similar palettes together.*

# V. Supported implementations

The implementations are inside the folder `products`.

## 1. JSON, the versatile default format

By default, a file `palettes.json` is provided to allow unsupported coding languages to also integrate `@prism` palettes. Here are the first line of this file.

```
{
  "Accent": [
    [0.498039, 0.788235, 0.498039],
    [0.690196, 0.705881, 0.757298],
    [0.882352, 0.721568, 0.661437],
    [0.99477, 0.835294, 0.550326],
    [0.913289, 0.935947, 0.610021],
    [0.306317, 0.487581, 0.680174],
    [0.700653, 0.146404, 0.562091],
    [0.855772, 0.162962, 0.316775],
    [0.671459, 0.366448, 0.159041],
    [0.4, 0.4, 0.4]
  ],
  ...
}
```

## 2. luadraw palettes

### a. Description

You can use `@prism` palettes with `luadraw` which is a package that greatly facilitates the creation of high-quality 2D and 3D plots via LuaLaTeX and TikZ.

**Note.**

*Initially, the `@prism` project was created to provide ready-to-use palettes for `luadraw`.*

### b. Use a luadraw palette

The Lua palette names all use the prefix `pal` followed by the name available in the file `palettes.json`. You can access a palette by three ways.

- `palGistHeat` is a Lua variable.
- `getPal('GistHeat')` and `getPal('palGistHeat')` are equal to `palGistHeat`.
- `palNames['palGistHeat']` is equal to `palGistHeat`.

**Note.**

*The Lua palette variables are arrays of arrays of three floats. Here is the definition of `palGistHeat`.*

```
palGistHeat = {
    {0.0, 0.0, 0.0},
    {0.105882, 0.0, 0.0},
    {0.211764, 0.0, 0.0},
    {0.317647, 0.0, 0.0},
    {0.429411, 0.0, 0.0},
    {0.535294, 0.0, 0.0},
    {0.641176, 0.0, 0.0},
    {0.752941, 0.003921, 0.0},
    {0.858823, 0.145098, 0.0},
    {0.964705, 0.286274, 0.0},
    {1.0, 0.42745, 0.0},
    {1.0, 0.57647, 0.152941},
    {1.0, 0.717647, 0.435294},
    {1.0, 0.858823, 0.717647},
    {1.0, 1.0, 1.0}
}
```

The `getPal` function has some options. To explain how this works, let's consider the following use case.

```
mypal = getPal(
    'GistHeat',
    {
        extract = {2, 5, 8, 9},
        shift   = 1,
        reverse = true
    }
)
```

To simplify the explanations, we will refer to the colors in the standard palette `'GistHeat'` as `coul_1`, `coul_2`, etc. The options are then **processed in the following order**.

1. `{coul_2, coul_5, coul_8, coul_9}` is the result of the extraction.
2. `{coul_9, coul_2, coul_5, coul_8}` comes from the shifting applied to the extracted palette (colors move to the right if `shift` is positive).
3. `{coul_8, coul_5, coul_2, coul_9}` is the reversed version of the shifted palette.

**Note.**

*The reversed version of any palette can be obtained using `getPal(palname, {reverse = true})`.*

# VI. Contribute via `Git`

**Caution.**

*Never use the* `main` *branch, which is for freezing the latest stable versions of all the projects in the mono repository* `https://github.com/projetmbc/for-writing`.

## 1. Complete the translations

**Important.**

*Although we're going to explain how to translate the documentation, it doesn't seem relevant to do so, as English should suffice these days.*

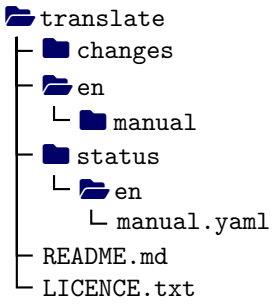



Figure 1: Simplified view of the translation folder

The translations are roughly organized as in figure 1 where just the important folders for the translations have been "*opened*".[6] **A little further down, the section VI-1-e explains how to add new translations**.

### a. The en folder

This folder, managed by the author of `@prism`, contains files easy to translate even if you're not a coder.

### b. The changes folder

This folder is a communication tool where important changes are indicated without dwelling on minor modifications specific to one or more translations.

### c. The status folder

This folder is used to keep track of translations from the project's point of view. Everything is done via well-commented `YAML` files, readable by a non-coder.

### d. The README.md and LICENCE.txt files

The `LICENCE.txt` file is aptly named, while the `README.md` file takes up in English the important points of what is said in this section about new translations.

### e. New translations

**Note.**

*The folder* `manual` *is reserved for documentation. It contains* `TEX` *files that can be compiled directly for real-time validation of translations.*

**Warning.**

*Only start from the* `en` *folder, as it's the responsibility of the* `@prism` *author.*

[6] This was the organization on October 26, 2025.

***Let's say you want to add support for Italian.***[7] To do this, you must use Git as follows.

1. Via https://github.com/projetmbc/for-writing/tree/aprism/@prism, recover the entire project folder. Do not use the `main` branch, which is used to freeze the latest stable versions of all the projects in the mono repository https://github.com/projetmbc/for-writing.
2. In the `@prism/contrib/translate` folder, create an `it` copy of the `en` folder, where `it` is the short name of the language documented in the page "*IIETF language tag*" from `Wikipedia`.
3. Once the translation is complete in the `it` folder, share it via https://github.com/projetmbc/for-writing/tree/aprism/@prism using a classic `git push`.

### 2. Improving the source code

Participation as a coder is made via the repository https://github.com/projetmbc/for-writing/tree/aprism/@prism corresponding to the `@prism` development branch. Here is what you can do, details can be found in the file https://github.com/projetmbc/for-writing/blob/aprism/@prism/contrib/products/README.md.

1. Create new palettes within an existing implementation. No coding skills required.
2. Propose a new implementation in your favorite programming language.
3. Combine both approaches.

## VII. History

1.2.1
2025-11-09

**Fix.**

- Equal palettes: the floating point equality uses now a correct tolerance.

**Break.**

- Palettes: the extra `Greys` has been removed (it is equal to `Grays`).

**New.**

- Similar palettes: two PDF files show similar palettes in standard and black modes (semi-automated process used).

**Update.**

- `luadraw` product: the associative array `palNames` has been added for compatibility reasons with the `luadraw` package.
- `BlindFish` palette: the last color variation has been made smoother (`luadraw` process used).

---

1.2.0
2025-10-29

**Break.**

- Palettes: all final palettes now consist of 10 colors.
- `luadraw` products: the `getPal` dictionary array has been converted into a function accepting string palette names (with or without `pal` prefix). See below.

**New.**

- Palettes.
  - Added `Lemon` and `ShiftRainbow` palettes (`luadraw` creation process used).
  - Added 37 palettes from the `Scientific Coulour Maps` project.
- `luadraw` product: accessing a palette and creating new ones can be made using the `getPal` function which has an optional argument `options` (dict-like array) with the following keys and their values.
  - `extract`: a list of non-zero integers used to extract specific colors from the palette (the order is preserved).
  - `reverse`: a boolean value indicating whether to reverse the palette color order (`false` by default).
  - `shift`: an integer value for applying a circular color shift to the palette.
- Documentations
  - Added English PDF manual.
  - Showcase: two PDF files demonstrate the use of each palette (white and dark modes).

---

[7] As mentioned above, there is no real need for the `doc` folder.

1.1.0
2025-10-14

### Break.

- Duplicate palettes and those that are reverse of others are ignored (strict equalities only).

### New.

- New palettes added: `BurningGrass`, `GeoRainbow` and `PastelRainbow` (`luadraw` creation process used).
- The `luadraw` palette product has a new dictionary like variable `getPal` to access a palette using its name (as a string variable).

### Update.

- Palette contributions: in the mandatory `extend.py` file, the `build_code` function must work with the dictionary of all the palettes, and manage a credit to the `@prism` project.

---

1.0.0
2025-10-11

First public version of the project.

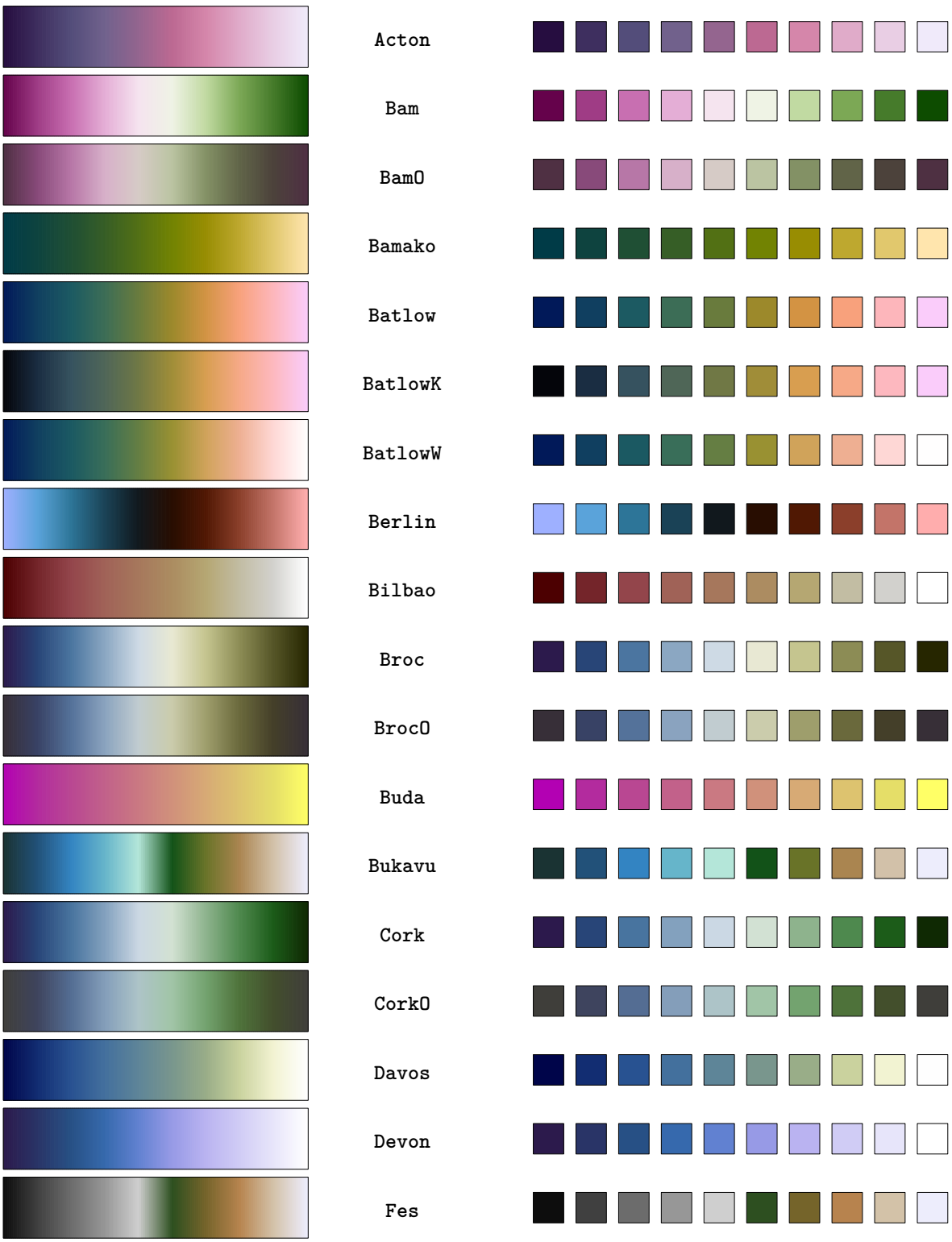
# Appendix 1 – The 127 palettes at a glance

The palette names used in this appendix are standard. Depending on the chosen implementation, they may be prefixed.

**Important.**

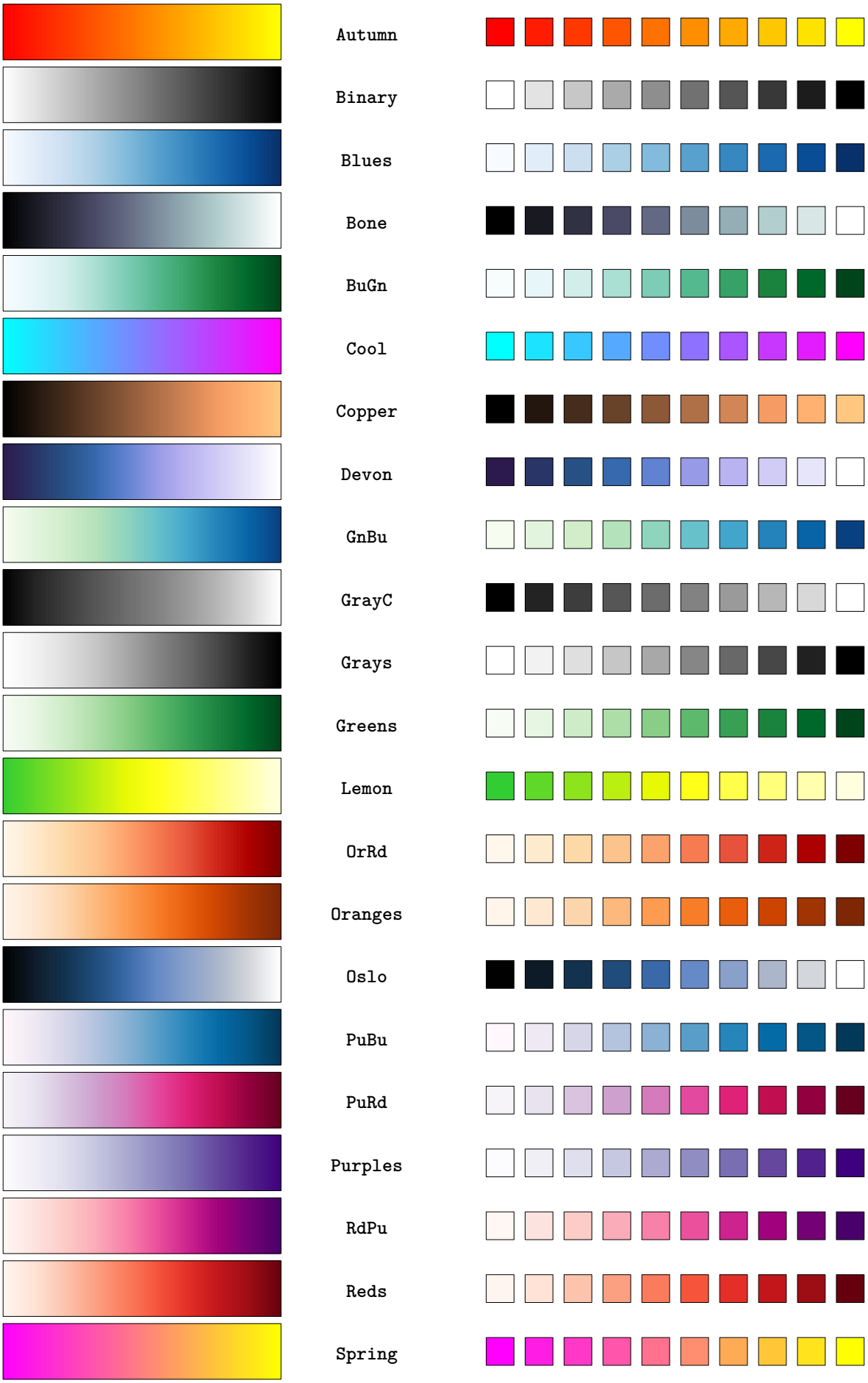
*Categories were generated semi-automatically using a program, followed by manual selection to obtain relevant choices. If you identify any errrors, please contact the author of `@prism`.*

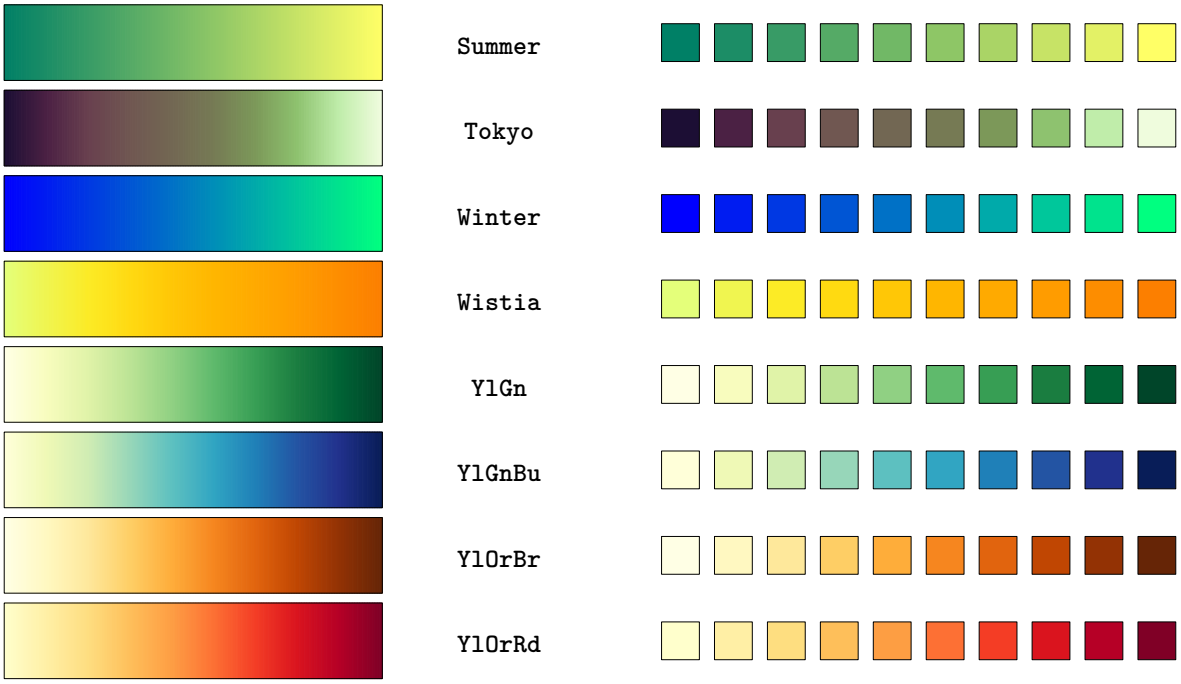
## Colorblind-friendly palettes (coming from Scientific Coulour Maps)



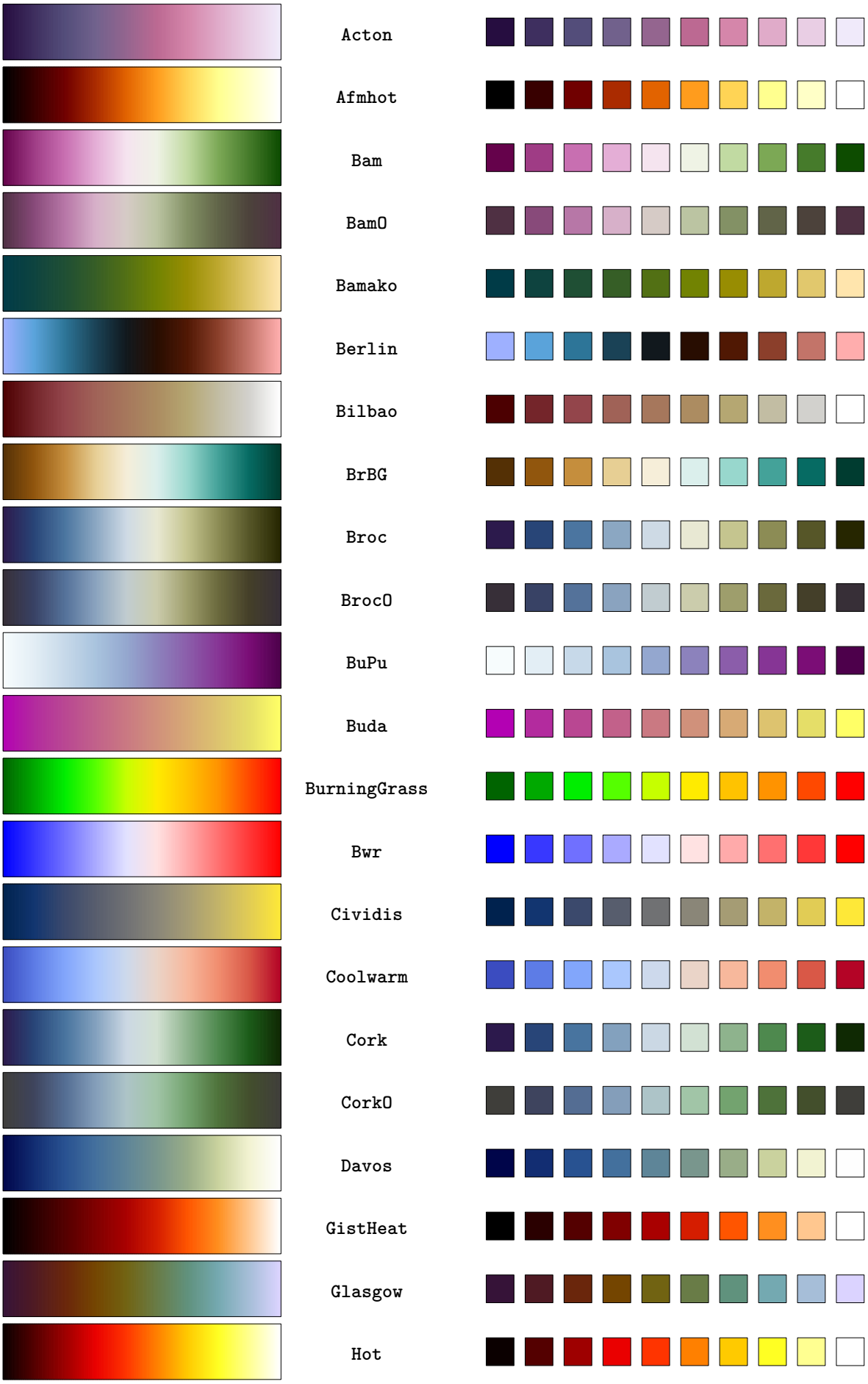

Glasgow
GrayC
Hawaii
Imola
Lajolla
Lapaz
Lipari
Lisbon
Managua
Navia
NaviaW
Nuuk
Oleron
Oslo
Roma
RomaO
Tofino
Tokyo
Turku
Vanimo
Vik
VikO

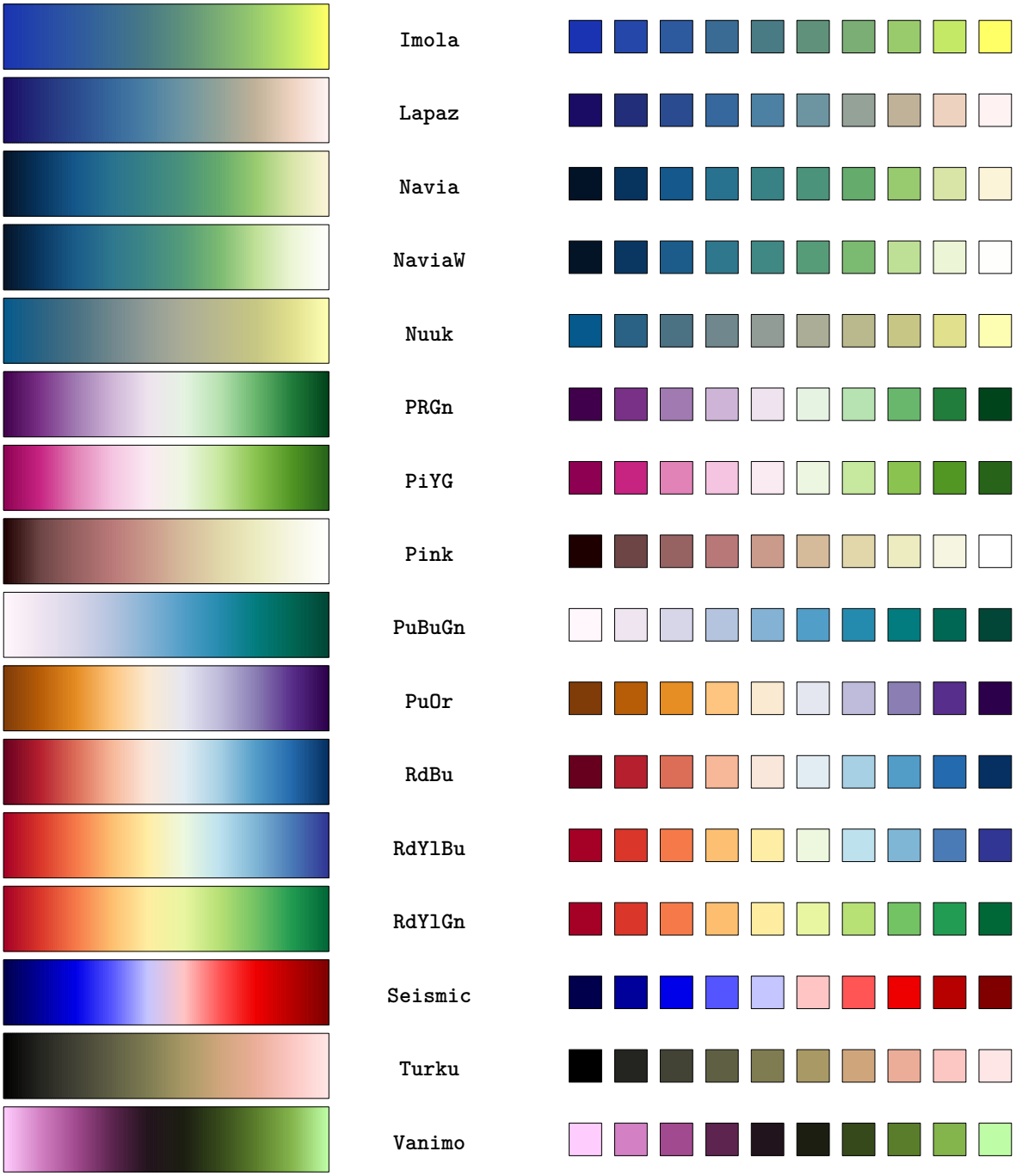
## Two-color palettes



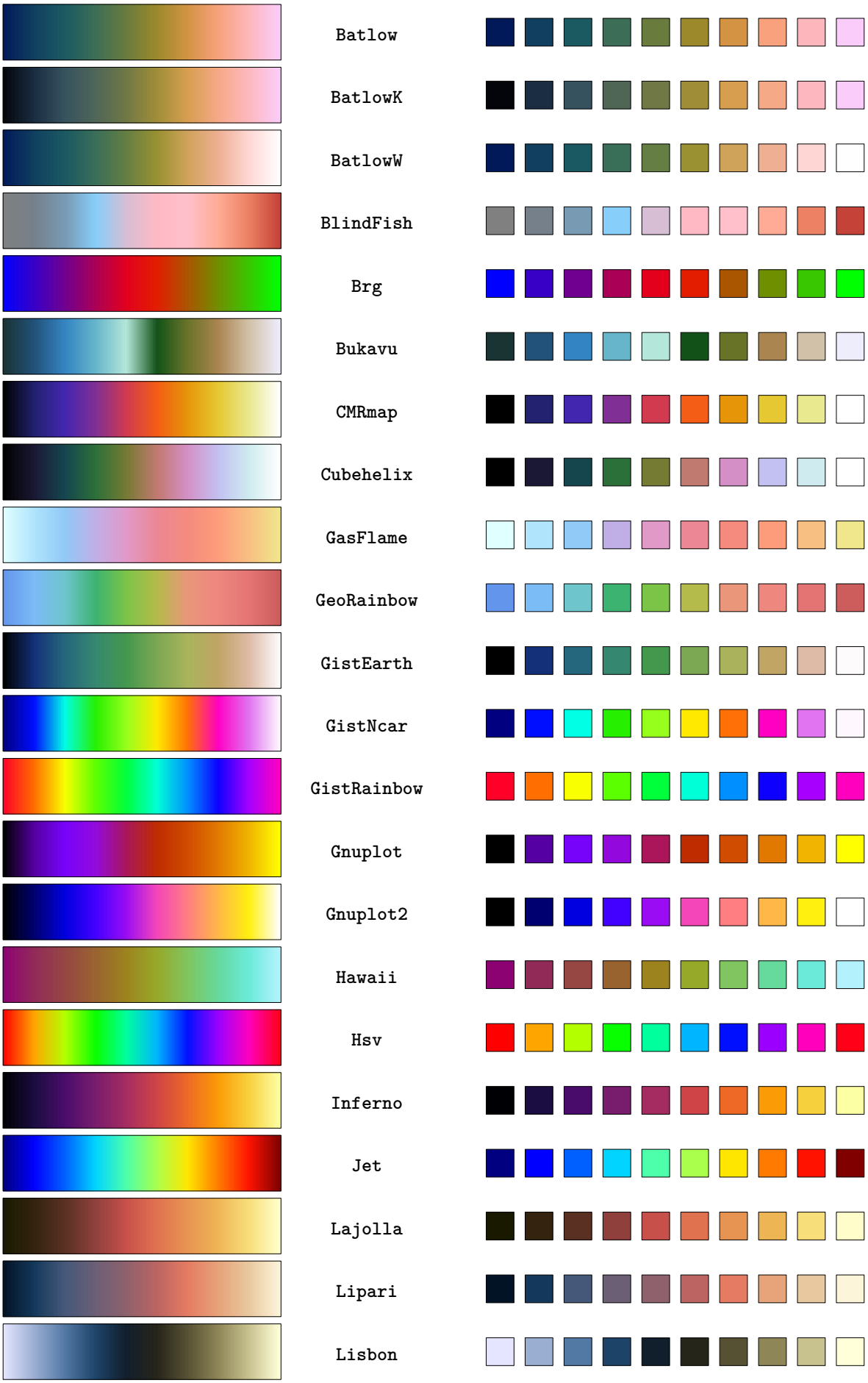

Summer
Tokyo
Winter
Wistia
YlGn
YlGnBu
YlOrBr
YlOrRd

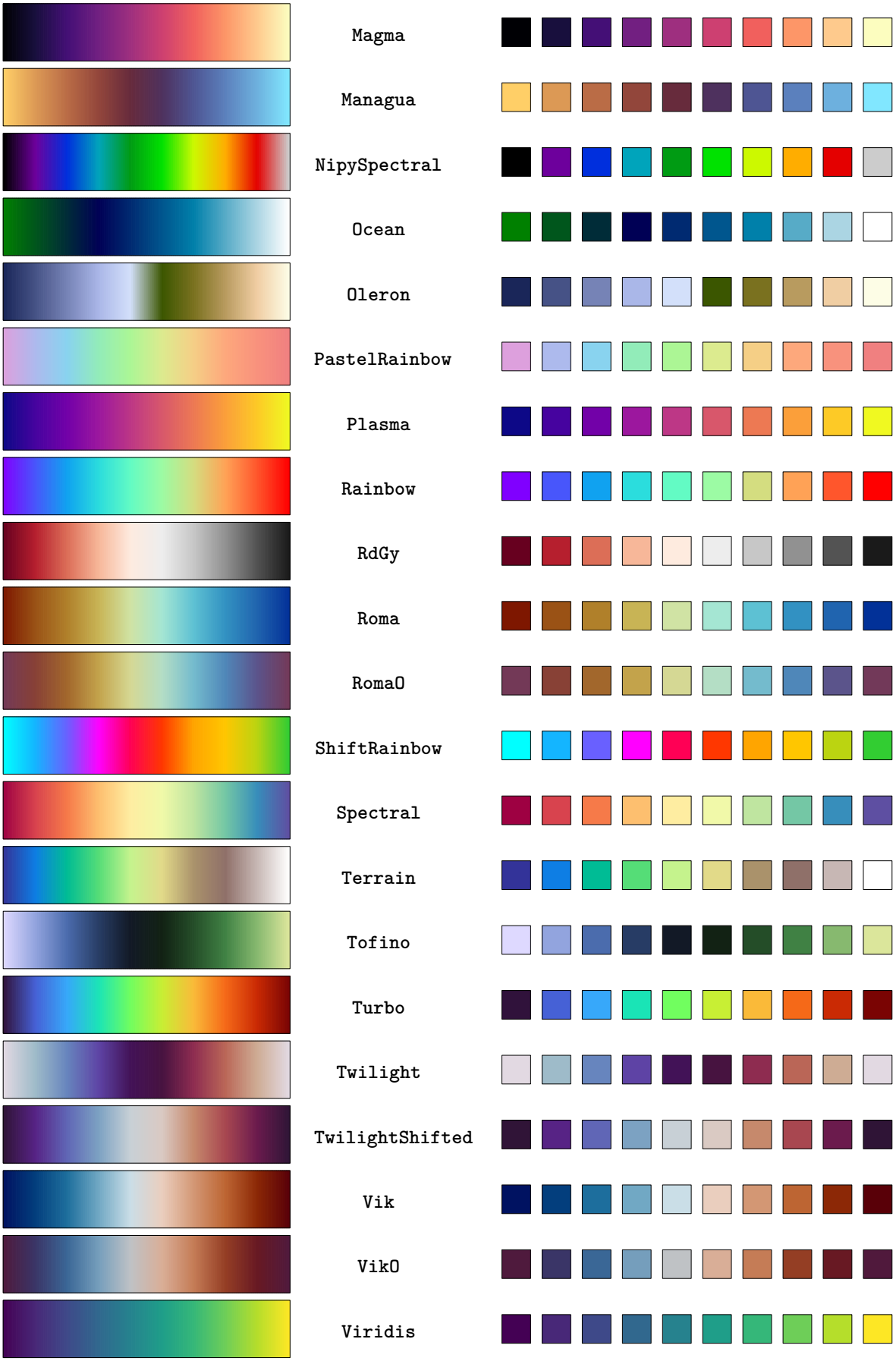
## Three-color palettes



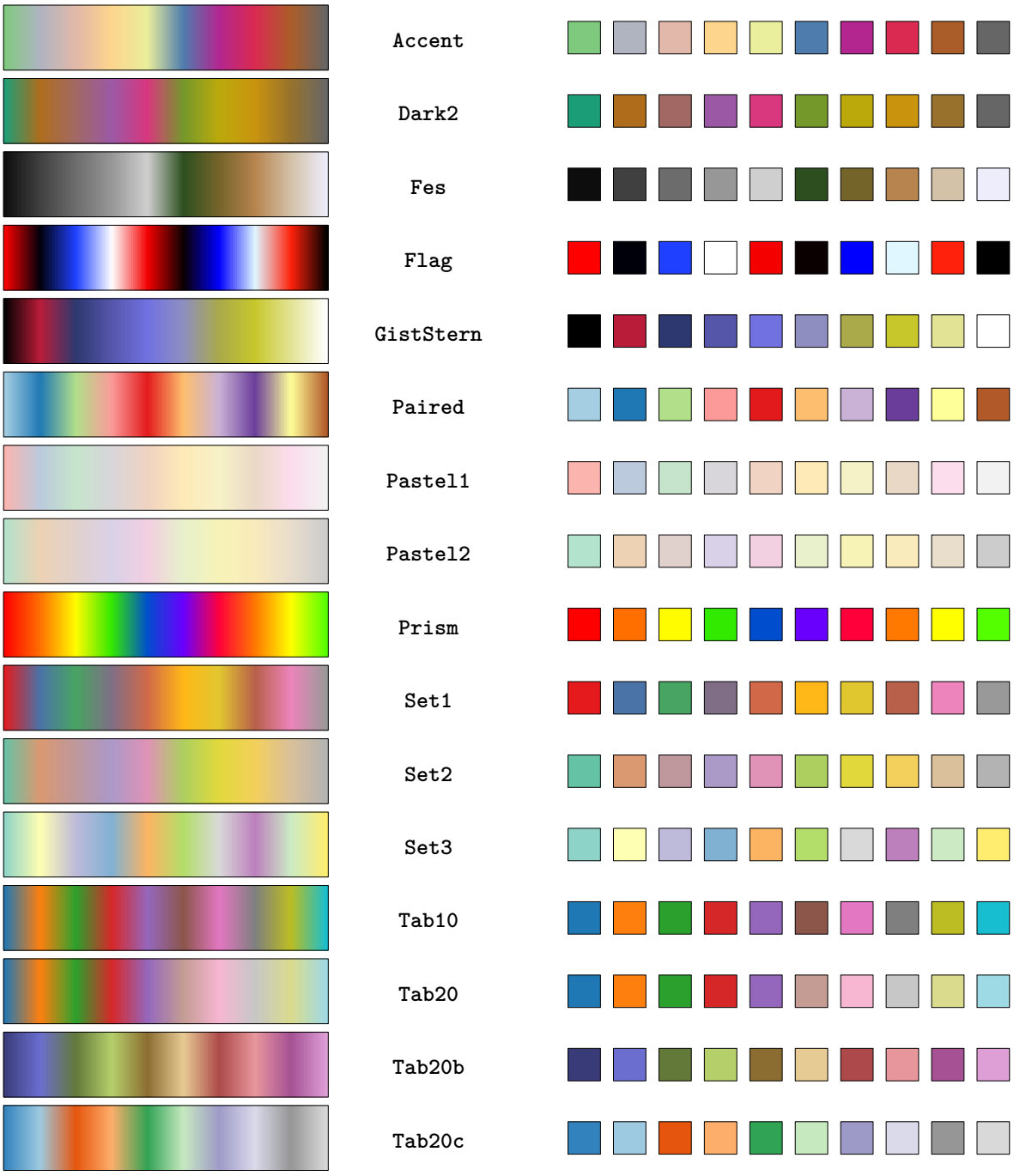

Imola
Lapaz
Navia
NaviaW
Nuuk
PRGn
PiYG
Pink
PuBuGn
PuOr
RdBu
RdYlBu
RdYlGn
Seismic
Turku
Vanimo

## Rainbow-style palettes




Magma
Managua
NipySpectral
Ocean
Oleron
PastelRainbow
Plasma
Rainbow
RdGy
Roma
RomaO
ShiftRainbow
Spectral
Terrain
Tofino
Turbo
Twilight
TwilightShifted
Vik
VikO
Viridis

## High-contrast palettes



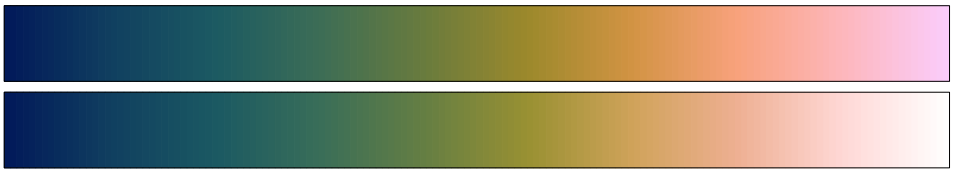
## Appendix 2 – Similar palettes

This appendix includes visually similar color palettes retained by `@prism` to respect individual preferences, and also because there are subtle differences between them, even when minimal.

**Important.**

*Clusters were generated semi-automatically using a program that suggests similar palettes, followed by manual curation to retain only relevant groupings. This approach may occasionally miss some similarities. If you identify any omissions, please contact the author of `@prism`.*

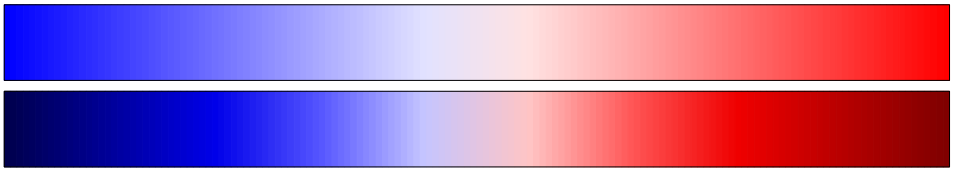
### Cluster #1



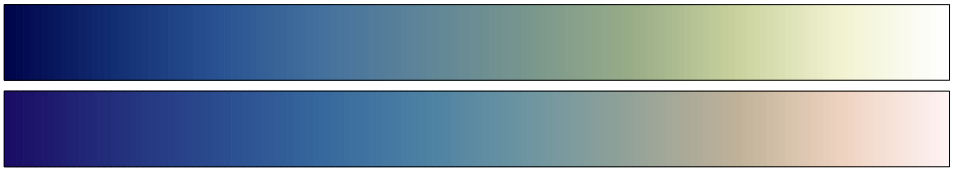

### Cluster #2


### Cluster #3


### Cluster #4




### Cluster #5

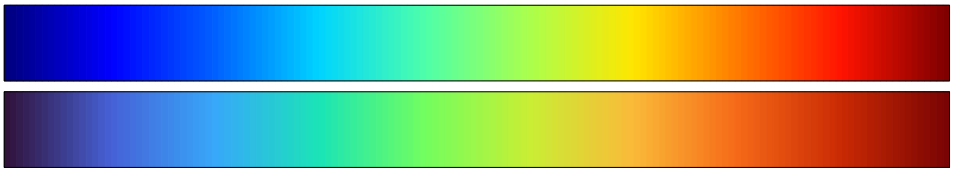


### Cluster #6

Inferno

Magma

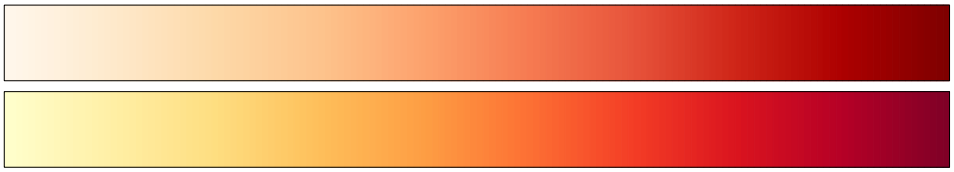
### Cluster #7



Jet

Turbo

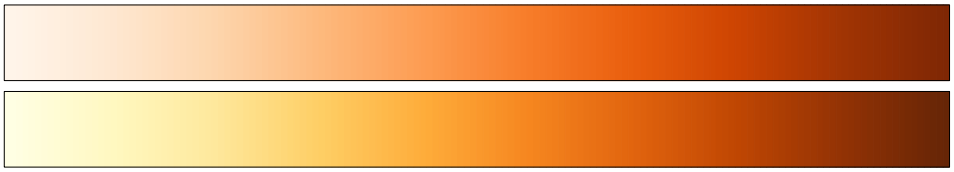
### Cluster #8



OrRd

YlOrRd

### Cluster #9



Oranges

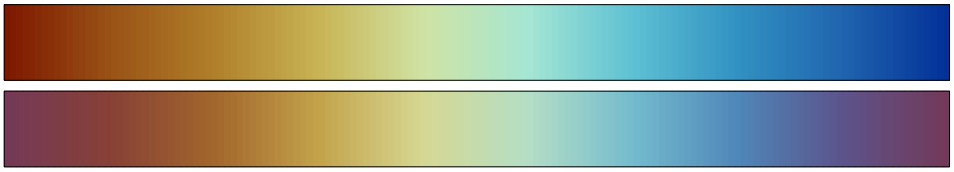
YlOrBr

### Cluster #10

RdBu

RdYlBu

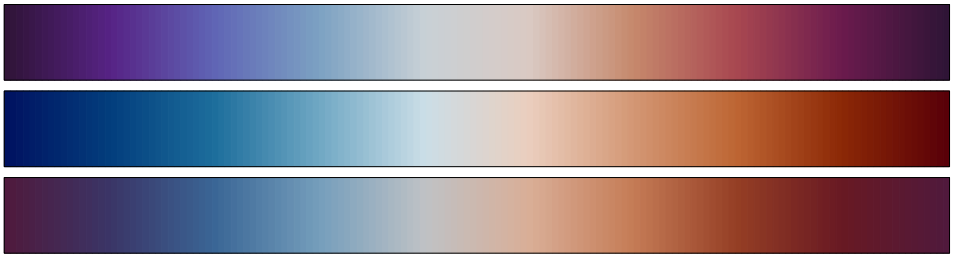
### Cluster #11



Roma

RomaO

### Cluster #12



TwilightShifted

Vik

VikO