MooViE - User Manual

Anton Stratmann and Martin Beyss

July 31st, 2018

Contents

Chapter 1

Introduction

MooViE is a fast, lightweight program to visualize the results from multi-criterial optimization (pareto optima). It can be easily downloaded, installed and used. Currently, MooViE is only supported for Linux.
MooViE can be easily described mathmatically as a relation R that is defined as:

$$R \subset (\mathcal{R}^m \times \mathcal{R}^n)$$

where m is the number of inputs and n is the number of outputs.
Figure 1.1 shows a simple MooViE scene. It is build around the center of the file and consists of several structural elements that are introduced in the following.

1.1 DomainAxis

A InputAxis represents a part of the domain of the displayed relation. According to the mathmatical model there are m DomainAxis in a scene.

1.1.1 Scale

The InputAxis scale displays an interval of $\mathcal{R}$ that contains every value this InputAxis will assume.

1.1.2 Histogram

The domain interval is divided into equidistant subintervals whose relative frequencies are displayed as histogram.

1.2 OutputGrid

The OutputGrid represents the codomain of the displayed relation. It has n levels of which one is a ring with colored segments and $n-1$ are circle segments

in a curved grid. The scale of both ring and circle segments is an interval of $\mathcal{R}$ that contains every value the associated output will assume.

1.2.1 Segmented Ring

The segmented ring should represent the most significant output. Every output curve displayed by the OutputGrid is colored according to the segment it meets.

1.2.2 Grid

Every further output is displayed on this grid. The outputs are connected by curves whereas the spiral segements are called connectors.

1.3 IOVector

A IOVector represents an element of R. It consists of m curves from each input value to the first output value and a curve that goes through every output value.

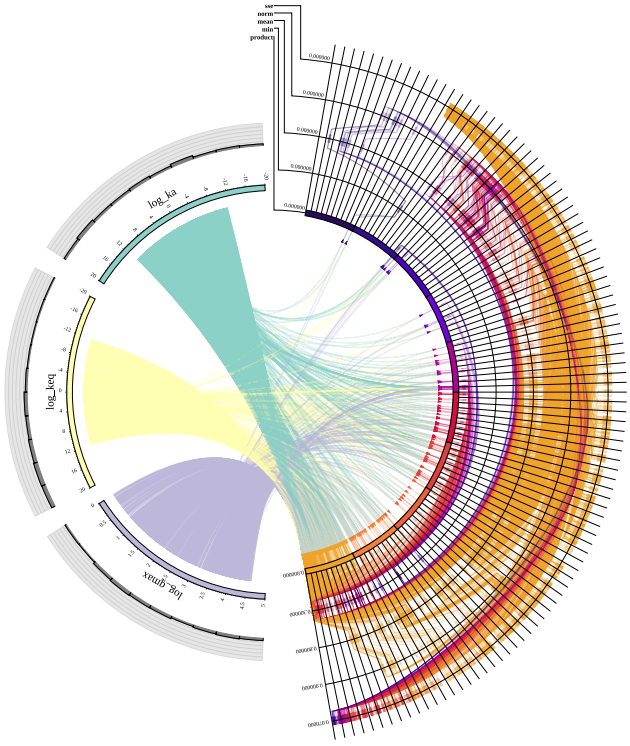

Figure 1.1: A simple MooViE scene

Chapter 2

Install

It follows an installation guide for MooViE.

2.1 Requirements

For building MooViE a compiler for **C++11 or later** and the libraries **cairo** and **cairomm-1.0** have to be installed. Note that cairo and cairomm require the **freetype2** and the **sigc++-2.0** library.
The hardware requirements for MooViE depend on the size of the data sets you want to load and display.

2.2 Getting started

To meet the requirements of MooViE you need to

1. Install a C++ compiler and make
2. Install cairo
3. Install cairomm-1.0
4. Install and set up CMake
5. Install git

If you are set up you can clone our repository from our GitHub repository[1].

2.3 Build MooViE

Having MooViE cloned from GitHub you are ready to start the CMake-Build-Sequence. You only need to execute the following steps: Execute

[1] https://github.com/modsim/MooViE.git

```
1 $ cmake <path to MooViE directory>
```

and CMake will check all requirements and produce a Makefile. You can now decide if you want to

- Only build the MooViE library and the MooViE executable, but not install any of them; then go into your MooViE directory and type

  ```
  $ make
  ```

- Build both library and executable and install both them; the go into your MooViE directory and type

  ```
  $ sudo make install
  ```

Hereafter, the MooViE directory should contain the files **libmoovie.so** (the library) and **moovie** (the executable). In case you installed MooViE, the command **moovie** should be available from the command line and the MooViE library can be included into your own program.

Chapter 3

Quickstart

Now that MooViE is installed on your computer you can now start using it. The following shows a simple instruction how to use MooViE.

3.1 The input file

Every MooViE scene needs an input file. It provides the data to display and a header with meta information. The input data can be imagined as a table where the columns are the inputs and outputs and the rows are connection of specific input and output values. A table header provides the type, name and unit of the column.
The format of the input file is CSV. Blank must be used as column separator and **\n** as row separator. Comment lines must begin with a #. The table header syntax is

```
i#<input name>[<unit>]
```

for inputs and

```
o#<output name>[<unit>]
```

for outputs where square brackets and units are optional for both. Therefore, the input file should look like

i#$input_1$ [$unit_1$]	$\cdots$	i#$input_m$ [$unit_m$]	o#$output_1$ [$unit_1$]	$\cdots$	o#$output_n$ [$unit_n$]
$ival_1$	$\cdots$	$ival_m$	$oval_1$	$\cdots$	$oval_n$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

3.2 Calling MooViE

Once installed MooViE can be called using the command line:

```
$ moovie <input file path>
```

With a valid input file path and input file, this command will render a MooViE scene and store it under the name `image.svg` in the current directory. Any existing file with the same name will be overwritten.
For further information on how to use command line parameters and a configuration file see chapter 4.

3.3 Including the MooViE library

When installing MooViE, a library with the name **libmoovie.so** is installed under **/usr/lib** and MooViE's header file are are installed under **/usr/include/moovie**. If you want to use the MooViE library in your project (e.g. for directly drawing MooViE scenes from your source code), you need to add the above mentioned library and include path to your project.
Because MooViE depends on **cairo** and **cairomm-1.0** which depend on **sigc++-2.0** and **freetype2**, all four library's include and library paths need to be added to your project.
A GNU g++ compiler call using the MooViE library would look like:

```
$ g++ -I<cairo include path> -I<cairomm-1.0 include path>
      -I<cairomm-1.0 config path> -I<sigc++-2.0 include dir>
      -I<sigc++ config path> -I<freetype2 include dir>
4     -I/usr/include/moovie
      -L<cairo/cairomm-1.0 library path>
      -o <your binary path> <your src paths>
      -lmoovie -lcairo -lcairomm-1.0
```

Hint: MooViE is licensed under GPL3. If you use MooViE in your project it must also be licensed under GPL3.

Chapter 4

Modifications

MooViE can be both modified using its command line syntax or a configuration file. Using the command line is recommended for quick changes in generell parameters like width, height or output file path. It provides far less options than the configuration file.
Therefore it is recommended to use a configuration file for enhanced costumization of MooViE. The user can change sizes, fonts and a lot more.

4.1 The command line syntax

A call to MooViE looks like:

```
$ moovie [OPTION <arg>] <input file path>
```

The only necessary option is the `input file path`, although the user will probably reach a more satisfying result when using more. It follows a listing of the parameters:

input file path A valid system path to the input file. The file must be of type csv and use `blank` as column and `newline` as row seperator. For running successfully it is also required to not have empty cells and follow the column header format as described in.

w/width A positive integer that will be the width of the result SVG image. Any value from $z \in \mathcal{Z}, z \leq 0$ will cause the program abort.

h/height A positive integer that will be the height of the result SVG image. Any value from $z \in \mathcal{Z}, z \leq 0$ will cause the program abort.

o/output-file A valid system path to the output file where the result SVG image will be stored. If a file with the same path exists, it will be overwritten.

c/configuration-file A valid system path to a MooViE configuration file. The file must be a text file and follow the configuration file format as described in.

4.2 The MooViE configuration file

The MooViE configuration supports three line types:

- MooViE configuration statements with the syntax:
  moovie.<key> = <value>
- Comment lines starting with #
- Blank lines

Every other line will be ignored but produce a warning to the standard output. If a value does not meet its restictions, the program aborts. A valid statement would be for instance

```
# Setting the width of the MooViE scene
moovie.width = 160
```

It follows a list with the possible keys, their meaning and value ranges:

width A positive integer that will be the width of the result SVG image. Any value from $z \in \mathcal{Z}, z \leq 0$ will cause the program abort.
Default: 750

height A positive integer that will be the height of the result SVG image. Any value from $z \in \mathcal{Z}, z \leq 0$ will cause the program abort.
Default: 750

output_angle_span The CodomainGrid will be centered on the right half of the scene circle. The output angle span determines the angle between its begin and its end. It must be in $(0, 175]$.
Default: 160

output_inner_radius The radius measured from the center where the CodomainGrid will begin. It can be in $(0, \infty)$.
Default: 160

output_thickness The height of the segmented ring the CodomainGrid starts with. The ring should represented the most important output, as it colors the outputs depending on their values. It must be in $(0, \infty)$.
Default: 5

grid_size The height of the output actual grid of the CodomainGrid. The grid starts after the segmented ring for the first output and ends after its maximal radius is $output\_inner\_radius + output\_thickness + grid\_size$. Grid size must be in $(0, \infty)$.
Default: 150

num_major_sections_grid The CodomainGrids grid is divided into sections by a number of thick lines. Allowed is a positive number of lines.
Default: 10

um_minor_sections_grid The CodomainGrids grid is divided into sections that are themself divided into minor sections by thin lines. Allowed is a positive number of lines.
Default: 10

input_inner_radius Each DomainAxis starts at the same radius. The radius is again measured from the center of the MooViE scene. It must be in $(0, \infty)$.
Default: 180

input_thickness The height of each DomainAxis colored scale. It must be in $(0, \infty)$.
Default: 5

input_separation_angle The DomainAxis will be put on the left half of the MooViE scene circle. Between two DomainAxis there is one separation angle. It must be be in $(0, 180)$.
Default: 5

um_major_sections_axis The scale of each DomainAxis is divided into thick sections. Allowed is a positive number of sections.
Default: 10

um_minor_sections_axis The scale of each DomainAxis is divided into sections that are themself divided into minor sections. Allowed is a positive number of sections.
Default: 10

histograms_enabled Toggle value that determines whether histograms should be drawn to each DomainAxis. It must be either `true` or `false`.
Default: true

num_histogram_classes The domain of every input is separated into equidistant sections and the quantities of values from each section is calculated. The number of this sections must be a positive number.
Default: 10

histogram_height The height of the histogram. It must be in $(0, \infty)$.
Default: 20

histogram_background Color of the histograms background entered as a color-separated triple of numbers. Each value must either be from $[0, 1]$ or an integer from 0 to 255.
Default: 0, 0, 0

histogram_fill Color of the histograms bins entered as a color-separated triple of numbers. Each value must either be from $[0, 1]$ or an integer from 0 to 255.
Default: 0.5, 0.5, 0.5

connector_arc_ratio A connector is drawn in a specific radius range between two output points. The ratio of this range to the total radius range of these outputs is set by `connector_arc_ratio`. It must be in $(0, 1)$.
Default: 0.6

thick_line_width The thick line property is used to draw boundries and major sections. Its width must be in $(0, \infty)$.
Default: 0.5

thin_line_width The thin line property is used to draw links, connectors, minor sections and output levels. Its width must be in $(0, \infty)$.
Default: 0.1

scale_label_font The font family that is used to draw scale labels. CSS2 generic font family names should be used.
Default: Liberation Serif

scale_label_font_size The size of the font that is used to draw scale labels. It must be in $(0, \infty)$.
Default: 5

axis_label_font The font family that is used to draw DomainAxis descriptions. CSS2 generic font family names should be used.
Default: Liberation Serif

axis_label_font_size The size of the font that is used to draw DomainAxis descriptions. It must be in $(0, \infty)$.
Default: 10