The ionio-illustrate package

Version 0.3.0

1. Introduction

This package implements a Cetz chart-like object for displaying mass spectrometric data in Typst documents. It allows for individually styled mass peaks, callouts, titles, and mass callipers.

2. Usage

This is the minimal starting point:

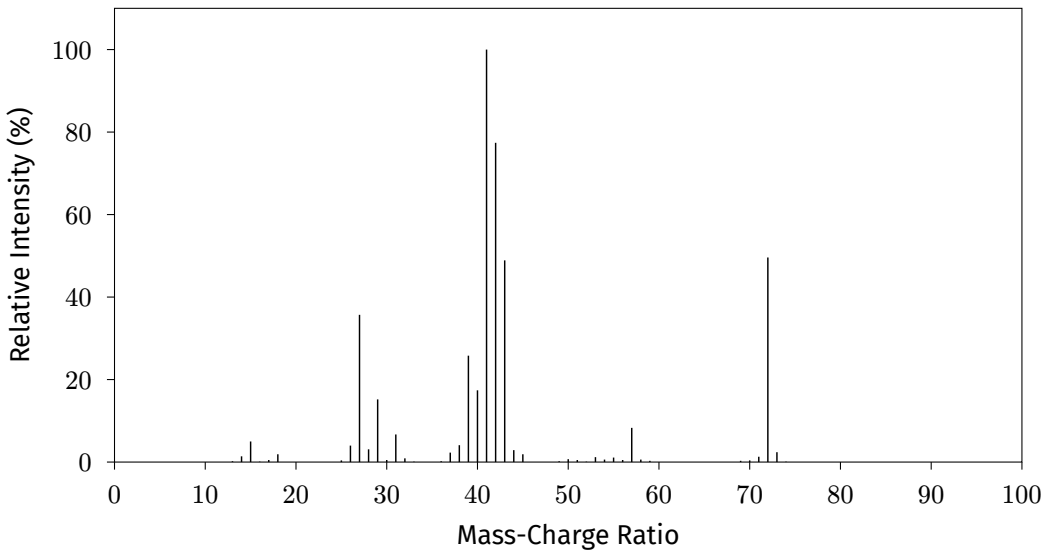
Example

```
#import "@preview/ionio-illustrate:0.3.0": *
#let data = csv("isobutelene_epoxide.csv")

#let ms = mass-spectrum(massspec, args: (
  size: (12,6),
  range: (0,100),
))

#figure((ms.display)())
```

The above code produces the following content:



It is important to note at this point that the syntax for interacting with mass spectrum objects will certainly change with the introduction of a native type system. This document will be updated to reflect this upon implementation of those changes.

Contents

3. Documentation

This documentation is generated automatically for each package release, and is guaranteed to be an acurate representation of the API in the strictest of terms, but may lack the additional explanations and examples that make for a good documentation. For a more approachable documentation (at the cost of potentially incorrect descriptions due to oversight), please see the hand-written documentation in Section 4.

3.1.1. mass-spectrum

Returns an object representing mass spectrum content.

3.1.1.1. Parameters

```
mass-spectrum(
  args: dictionary,
  data1: array,
  data2: array
) -> dictionary none
```

args `dictionary`

Override default behaviour of the mass spectrum by overriding methods, or setting fields.

Default: `(:)`

data1 `array`

The mass spectrum in the format of a 2D array, or an array of dictionarys. By default, the mass-charges ratios are in the first column, and the relative intensities are in the second column.

data2 `array`

similar format as `data1`, but to contain a second mass spectrum.

Default: `none`

4. Humanist Documentation

This documentation is hand-written, and therefore may sometimes be incorrect if it hasn't been updated to a recent API change (though hopefully those are few). If you see an issue in this documentation, please put in an issue or a pull request on the GitHub repository. That being said, a best effort is made to ensure that this section is useful.

4.1. `mass-spectrum()`

The `mass-spectrum()` function takes two positional arguments:

`data1` (`array` or `dictionary`) This is a 2-dimensional array relating mass-charge ratios to their intensities. By default, the first column is the mass-charge ratio and the second column is the intensity. Data for a second mass spectrum is stored within the `args` parameter below.

`data2` (`array` or `dictionary`) An optional second mass spectrum to display. Data is in the same format as in `data1`.

`args` (`dictionary`) This contains suplemental data that can be used to change the style of the mass spectrum, or to add additional content using provided functions (see Section 4.1.7).

The defaults for the `args` dictionary are shown below:

```
data1: none,
data2: none,
keys: (
  mz: 0,
  intensity: 1
),
size: (14,5),
range: (40, 400),
style: mass-spectrum-default-style,
labels: (
  x: [Mass-Charge Ratio],
  y: [Relative Intensity (%)]
),
linestyle: (this, idx)=>{},
plot-extras: (this)=>{},
plot-extras-bottom: (this)=>{},
```

4.1.1. keys

The `keys` entry in the `args` positional argument is a `dictionary` that can be used to change which fields in the provided `data` `array` / `dictionary` are to be used to plot the mass spectrum. An example usage of this may be to store several mass spectra within a single datafile.

Info

Note that arrays are 0-index based.

Info

When two mass spectra are provided, both must use the same keys.

Example

```
#let ms = mass-spectrum(massspec, args: (
  keys: (
    mz: 0, // mass-charge is contained in the first column
    intensity: 1 // intensity is contained in the second column
  )
))
```

4.1.2. size

The keys entry in the args positional argument is a tuple specifying the size of the mass spectrum on the page, in Cetz units.

Example

```
#let ms = mass-spectrum(massspec, args: (
  size: (12,6)
))
```

4.1.3. range

The range entry in the args positional argument is a tuple specifying the min and the max of the mass-charge axis.

Example

```
#let ms = mass-spectrum(massspec, args: (
  range: (0,100) // Show mass spectrum between 0 m/z and 100 m/z
))
```

4.1.4. style

The style entry in the args positional argument is a cetz `style` dictionary. This dictionary accepts 5 entrys, each affecting a different part of the mass spectrum plot:

- **axes** This is a style dictionary that is passed to cetz.axes.scientific after expansion. Please refer to the Cetz documentation for the subentries that are available (at the time of writing, these include tick, frame, and label among other things)/
- **callouts** This is passed directly to cetz.draw.content after expansion. Please refer to the Cetz documentation for the subentries that are available (at the time of writing, these include stroke, fill, and frame, and padding)
- **peaks** This is the style passed to all peaks being drawn(overriden by the linestyle function). Internally, this is passed to cetz.draw.line. Please refer to the Cetz documentation for the subentries that are available (at the time of writing, this include stroke)
- **title** This is passed directly to cetz.draw.content after expansion. Please refer to the Cetz documentation for the subentries that are available (at the time of writing, these include stroke, fill, and frame, and padding)
- **callipers** This dictionary entry itself is a dictionary that takes line (which is passed directly to cetz.draw.line after expansion) which allows customisation of the calliper's lines, and content (which is passed directly to cetz.draw.content) which allows for customizing the content placed above the callipers.
- **peaks** TO DO
- **data1** TO DO
- **data2** TO DO
- **shift-amount** TO DO

4.1.5. labels

The labels entry in the args positional argument is a dictionary specifying the labels to be used on each axis.

Example

```
#let ms = mass-spectrum(massspec, args: (
  labels: (
    x: [Mass-Charge Ratio],
    y: [Relative Intensity \[%\]]
  )
))
```

Warning

Note that if you provide this entry, you must provide both child entries.

4.1.6. `linestyle`

The `linestyle` entry in the `args` positional argument is a function taking two parameters: `this` (refering to the `#ms` object), and `idx` which is an `integer` representing the mass-charge ratio of the peak being drawn. Returning a cetz style dictionary will change the appearence of the peaks. This may be used to draw the reader's attention to a particular mass spectrum peak by colouring it in red, for example.

Example

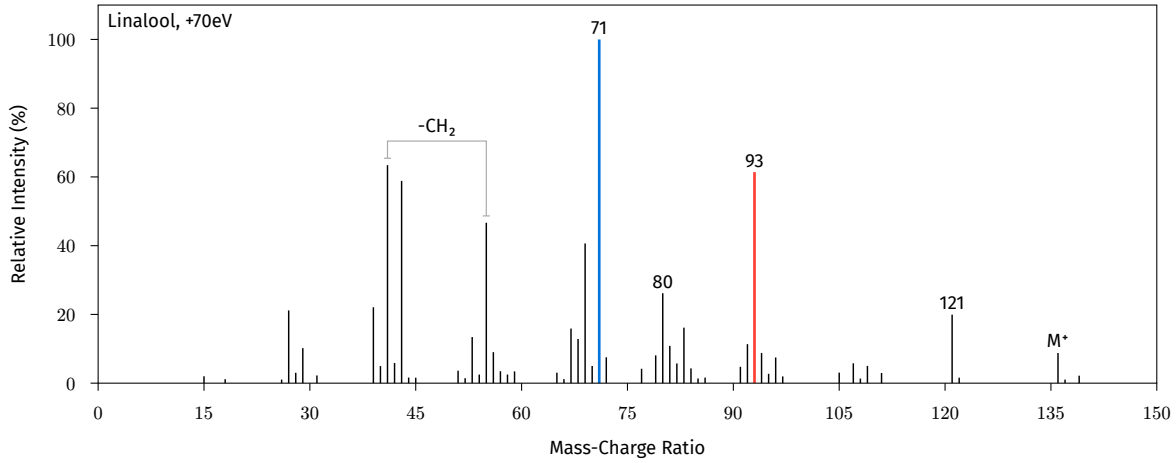
```
#let ms = mass-spectrum(massspec, args: (
  linestyle: (this, idx)=>{
      if idx in (41,) {return (stroke: red)}
    }
))
```

4.1.7. `plot-extras` and `plot-extras-bottom`

The `plot-extras` entry in the `args` positional argument is a function taking one parameter, `this`, which refers to the `#ms` object. It can be used to add additional content to a mass spectrum using provided functions

Example

```
#let ms = mass-spectrum(massspec, args: (
  range: (0,150),
  plot-extras: (this) => {
    (this.callout-above)(136, content: MolecularIon())
    (this.callout-above)(121)
    (this.callout-above)(93)
    (this.callout-above)(80)
    (this.callout-above)(71)
    (this.callipers)(41, 55, content: [-CH#sub[2]])
    (this.title)([Linalool, +70eV])
  },
  linestyle: (this, mz)=>{
    if mz in (93,) { return (stroke: red) }
    if mz in (71,) { return (stroke: blue) }
  }
))
#(ms.display)()
```



4.2. Method functions

This section briefly outlines method functions and where/why they might be used.

4.2.1. Display function(s)

These are the functions that will render the mass spectrum. For the moment there is only one, though as there are several desirable ways to render a mass spectrum, I envision adding more functions to this.

> ⚠ **Warning**
>
> Display functions **must not** be called within the context of a `plot-extras(this)` function.

4.2.1.1. `#ms.display(mode)`

The `#ms.display` method is used to place a single mass spectrum within a document. It can be called several times.

`mode` can be any of

- **`single`** (default) Displays a single mass spectrum from the first dataset.
- **`dual-reflection`** displays both given mass spectra, with one reflected about the mass axis.
- **`dual-shift`** Displays both mass spectra on the same axis, offset from one another.

4.2.2. `plot-extras` Functions

> ⚠ **Warning**
>
> The behaviour of `plot-extra` functions is **undefined** when called outside of the context of a `plot-extras(this)` function.

4.2.2.1. `#ms.title(content)`

The `#ms.title` method allows the addition of a title to a mass spectrum.

It takes one positional argument, content (`content` or `string`).

4.2.2.2. `#ms.callout-above(mz, content: [], inset)`

The `#ms.callout-above` method places a callout slightly above the intensity peak for a given mass-charge ratio.

It takes one positional argument `mz` (`integer`, `float`, or `string`) and two named arguments, `content` (`content`, `string`, or `none`) to be displayed above the mass peak, and `y-offset` (`length`), which is the distance above the mass peak at which the content is displayed.

- If `content` is `none`, the default value is that which is provided as `mz`.

- If inset is none, the default value is 0.3em.
- If mz is outside of the mass spectrums rang x-axis range, it will not be shown

4.2.2.3. #ms.callout-aside(mz, position, height, content, anchor, inset)

Behaves similarly to #ms.calloutbove, however, content is instead rendered at a specified position, with a faint line connecting the content to the mass peak at the specified height.

If height is not specified, it is auto. If it is auto, it is set to 100%. If height is a ratio, the height is set to that ratio of the mass-peak intensity at mz.

If content is not provided, it defaults to mz.

4.2.2.4. #ms.callipers(mz1, mz2, content: none, height: none, arrow-width: 1, inset: 0.5em)

The #ms.callipers method places a mass callipers between two mass spectrum peaks, along with any desired content centered above the callipers.

It takes two positional arguments mz1 and mz2 (either of which are integer, float, or string) which represent the start and end of the callipers respectively, and two named arguments, content (content, string, or none) which is displayed centered above the callipers, and and height (length), which is the distance at which the content floats above the mass peak.

- If content is none, it is set automatically to represent the loss of mass between the specified peaks.
- If height is none, the default value is 0.3em.

> ⚠ **Warning**
>
> The behaviour is **undefined** when either mz1 or mz2 are outside the x-axis range.