
gunshotmatch-reports

Release 0.7.0b1

PDF Report Generation for GunShotMatch.

Dominic Davis-Foster

May 08, 2024

Contents

1	Installation	1
1.1	from PyPI	1
1.2	from GitHub	1
I	API Reference	3
2	gunshotmatch_reports.alignment	5
2.1	CSVRow	5
2.2	csv_two_projects	6
2.3	csv_two_projects_and_unknown	6
2.4	get_csv_data	6
3	gunshotmatch_reports.chromatogram	7
3.1	build_chromatogram_report	7
4	gunshotmatch_reports.peaks	9
4.1	CSVReports	9
4.2	PeakMetadataTable	10
4.3	PeakSummary	11
4.4	build_peak_report	11
5	gunshotmatch_reports.utils	13
5.1	extend_list	13
5.2	figure_to_drawing	13
5.3	save_pdf	13
5.4	save_svg	13
5.5	scale	14
II	Contributing	15
6	Contributing	17
6.1	Coding style	17
6.2	Automated tests	17
6.3	Type Annotations	17
6.4	Build documentation locally	18
7	Downloading source code	19
7.1	Building from source	20
8	License	21

Python Module Index	23
Index	25

Installation

1.1 from PyPI

```
$ python3 -m pip install gunshotmatch-reports --user
```

1.2 from GitHub

```
$ python3 -m pip install git+https://github.com/GunShotMatch/gunshotmatch-reports@master --user
```


Part I

API Reference

gunshotmatch_reports.alignment

CSV reports of alignment between reference projects and unknown samples.

New in version 0.5.0.

Classes:

<code>CSVRow([peak_no, name, rt, area, ...])</code>	Represents data for a peak in a CSV report.
---	---

Functions:

<code>csv_two_projects(p1, padded_p1_cp, p2, ...)</code>	Returns CSV report for the alignment between the given projects.
<code>csv_two_projects_and_unknown(p1, ..., ...)</code>	Returns CSV report for the alignment between the given projects and unknown.
<code>get_csv_data(project, cp, max_area)</code>	Return data for the CSV report for given peak in the given project.

namedtuple `CSVRow` (`peak_no=""`, `name=""`, `rt=""`, `area=""`, `area_percentage=""`, `match_factor=""`)

Bases: `NamedTuple`

Represents data for a peak in a CSV report.

Fields

- 0) **peak_no** (`str`) – Alias for field number 0
- 1) **name** (`str`) – Alias for field number 1
- 2) **rt** (`str`) – Alias for field number 2
- 3) **area** (`str`) – Alias for field number 3
- 4) **area_percentage** (`str`) – Alias for field number 4
- 5) **match_factor** (`str`) – Alias for field number 5

`__repr__()`

Return a nicely formatted representation string

classmethod `header()`

Returns the CSV column headers.

Return type `Tuple[str, str, str, str, str, str]`

csv_two_projects (*p1, padded_p1_cp, p2, padded_p2_cp*)

Returns CSV report for the alignment between the given projects.

Parameters

- **p1** (*Project*) – The first project.
- **padded_p1_cp** (*MutableSequence[Optional[ConsolidatedPeak]]*) – Padded consolidated peak list for the first project.
- **p2** (*Project*) – The second project.
- **padded_p2_cp** (*MutableSequence[Optional[ConsolidatedPeak]]*) – Padded consolidated peak list for the second project.

Return type *str*

csv_two_projects_and_unknown (*p1, padded_p1_cp, p2, padded_p2_cp, u, padded_u_cp, *, pair_only=False*)

Returns CSV report for the alignment between the given projects and unknown.

Parameters

- **p1** (*Project*) – The first project.
- **padded_p1_cp** (*MutableSequence[Optional[ConsolidatedPeak]]*) – Padded consolidated peak list for the first project.
- **p2** (*Project*) – The second project.
- **padded_p2_cp** (*MutableSequence[Optional[ConsolidatedPeak]]*) – Padded consolidated peak list for the second project.
- **u** (*Project*) – The unknown sample.
- **padded_u_cp** (*MutableSequence[Optional[ConsolidatedPeak]]*) – Padded consolidated peak list for the unknown sample.
- **pair_only** (*bool*) – Only show peaks in common between two or more of the projects/unknown. Default *False*.

Return type *str*

get_csv_data (*project, cp, max_area*)

Return data for the CSV report for given peak in the given project.

Parameters

- **project** (*Project*)
- **cp** (*Optional[ConsolidatedPeak]*)
- **max_area** (*float*) – The maximum peak area in the project.

Return type *CSVRow*

gunshotmatch_reports.chromatogram

PDF Chromatogram Generator.

Functions:

<code>build_chromatogram_report(project[, ...])</code>	Construct a chromatogram report for the given project and write to the chosen file.
--	---

build_chromatogram_report (*project*, *pdf_filename=None*)

Construct a chromatogram report for the given project and write to the chosen file.

Parameters

- **project** (`Project`)
- **pdf_filename** (`Union[str, Path, PathLike, None]`) – Optional output filename. Defaults to `project_name_chromatogram.pdf`.

Return type `str`

gunshotmatch_reports. peaks

Peak Report Generator.

Classes:

<i>CSVReports</i> (project)	Class for producing CSV peak reports.
<i>PeakMetadataTable</i> (project)	Helper class for peak metadata for insertion into a PDF or CSV report.
<i>PeakSummary</i> (peak_no, name, rt, area, ...)	Summary data for a <i>ConsolidatedPeak</i> .

Functions:

<i>build_peak_report</i> (project[, pdf_filename, ...])	Construct a peak report for the given project and write to the chosen file.
---	---

class CSVReports (*project*)

Bases: *object*

Class for producing CSV peak reports.

Parameters **project** (*Project*) – A GunShotMatch project.

New in version 0.4.0.

Methods:

<i>overview_csv</i> ()	Produce an overview report of the peaks, giving the name, retention time and peak area of the peaks.
<i>summary_csv</i> ()	Produce a summary report of the peaks, giving the individual retention times and hits.

overview_csv ()

Produce an overview report of the peaks, giving the name, retention time and peak area of the peaks.

The output columns are as follows:

- Peak No.
- Name – The name of the top hit
- Rt – Mean retention time
- Area – Mean peak area
- Area % – Mean peak area as a percentage of the largest peak
- MF – Mean match factor for the top hit

- Rejected – Whether the peak has been rejected (e.g. with PeakViewer)

The peaks are sorted from largest to smallest.

Return type `str`

summary_csv()

Produce a summary report of the peaks, giving the individual retention times and hits.

The table for each peak mirrors the table in the PDF peak report (`build_peak_report()`). The peaks are sorted from largest to smallest.

Return type `str`

class PeakMetadataTable (*project*)

Bases: `object`

Helper class for peak metadata for insertion into a PDF or CSV report.

Parameters `project` (`Project`) – A GunShotMatch project.

New in version 0.3.0.

Attributes:

<code>area_percentages</code>	Peak areas as percentage of largest peak
<code>max_peak_number</code>	The total number of peaks in the project.
<code>num_rows</code>	Maximum number of rows for the metadata table.
<code>project</code>	A GunShotMatch project.

Methods:

<code>get_summary_for_peak(peak, peak_number)</code>	Return a formatted summary of the peak.
<code>get_table_for_peak(peak, peak_number)</code>	Return tabulated data on the peak, with individual peak areas, retention times, and the top 5 hits.

area_percentages

Peak areas as percentage of largest peak

get_summary_for_peak (*peak, peak_number*)

Return a formatted summary of the peak.

Parameters

- `peak` (`ConsolidatedPeak`)
- `peak_number` (`int`) – The peak number, 1-indexed.

Return type `PeakSummary`

get_table_for_peak (*peak, peak_number*)

Return tabulated data on the peak, with individual peak areas, retention times, and the top 5 hits.

Parameters

- `peak` (`ConsolidatedPeak`)
- `peak_number` (`int`) – The peak number, 1-indexed.

Return type `List[Tuple[str, str, str, str, str, str, str, str, str]]`

max_peak_number

The total number of peaks in the project.

num_rows

Maximum number of rows for the metadata table.

project

A GunShotMatch project.

namedtuple PeakSummary (*peak_no, name, rt, area, area_percentage, mf, rejected*)

Bases: `NamedTuple`

Summary data for a `ConsolidatedPeak`.

Formatted for insertion into a PDF or CSV peak report.

New in version 0.3.0.

Fields

- 0) **peak_no** (`str`) – Peak number, 1-indexed
- 1) **name** (`str`) – The name of the top hit
- 2) **rt** (`str`) – Mean retention time
- 3) **area** (`str`) – Mean peak area
- 4) **area_percentage** (`str`) – Mean peak area as a percentage of the largest peak
- 5) **mf** (`str`) – Mean match factor for the top hit
- 6) **rejected** (`str`) – Whether the peak has been rejected (e.g. with PeakViewer)

__repr__ ()

Return a nicely formatted representation string

build_peak_report (*project, pdf_filename=None, *, title_every_page=False*)

Construct a peak report for the given project and write to the chosen file.

Parameters

- **project** (`Project`)
- **pdf_filename** (`Union[str, Path, PathLike, None]`) – Optional output filename. Defaults to `project_name_peak_report.pdf`.

Return type `str`

Returns The output filename.

gunshotmatch_reports.utils

Utility functions.

Functions:

<code>extend_list(l, fillvalue, length)</code>	Extend list to the given length with fillvalue.
<code>figure_to_drawing(figure)</code>	Convert a matplotlib figure to a reportlab drawing.
<code>save_pdf(fig, filename)</code>	Save a PDF without a creation date.
<code>save_svg(fig, filename)</code>	Save an SVG with fixed hashsalt and without a creation date.
<code>scale(drawing, scale)</code>	Scale reportlab.graphics.shapes.Drawing() object while maintaining aspect ratio.

extend_list (*l*, *fillvalue*, *length*)

Extend list to the given length with fillvalue.

Parameters

- **l** (`List[~T]`)
- **fillvalue** (`~T`)
- **length** (`int`)

Return type `List[~T]`

figure_to_drawing (*figure*)

Convert a matplotlib figure to a reportlab drawing.

Parameters **figure** (`Figure`)

Return type `Drawing`

save_pdf (*fig*, *filename*)

Save a PDF without a creation date.

Parameters

- **fig** (`Figure`)
- **filename** (`Union[str, Path, PathLike]`)

New in version 0.6.0.

save_svg (*fig*, *filename*)

Save an SVG with fixed hashsalt and without a creation date.

Parameters

- **fig** (`Figure`)
- **filename** (`Union[str, Path, PathLike]`)

New in version 0.6.0.

scale (*drawing, scale*)

Scale `reportlab.graphics.shapes.Drawing()` object while maintaining aspect ratio.

Parameters

- **drawing** (`Drawing`)
- **scale** (`float`)

Return type `Drawing`

Part II

Contributing

Contributing

`gunshotmatch-reports` uses `tox` to automate testing and packaging, and `pre-commit` to maintain code quality.

Install `pre-commit` with `pip` and install the git hook:

```
$ python -m pip install pre-commit
$ pre-commit install
```

6.1 Coding style

`formate` is used for code formatting.

It can be run manually via `pre-commit`:

```
$ pre-commit run formate -a
```

Or, to run the complete autoformatting suite:

```
$ pre-commit run -a
```

6.2 Automated tests

Tests are run with `tox` and `pytest`. To run tests for a specific Python version, such as Python 3.6:

```
$ tox -e py36
```

To run tests for all Python versions, simply run:

```
$ tox
```

6.3 Type Annotations

Type annotations are checked using `mypy`. Run `mypy` using `tox`:

```
$ tox -e mypy
```

6.4 Build documentation locally

The documentation is powered by Sphinx. A local copy of the documentation can be built with `tox`:

```
$ tox -e docs
```

Downloading source code

The `gunshotmatch-reports` source code is available on GitHub, and can be accessed from the following URL:
<https://github.com/GunShotMatch/gunshotmatch-reports>

If you have `git` installed, you can clone the repository with the following command:

```
$ git clone https://github.com/GunShotMatch/gunshotmatch-reports
```

```
Cloning into 'gunshotmatch-reports'...
remote: Enumerating objects: 47, done.
remote: Counting objects: 100% (47/47), done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 173 (delta 16), reused 17 (delta 6), pack-reused 126
Receiving objects: 100% (173/173), 126.56 KiB | 678.00 KiB/s, done.
Resolving deltas: 100% (66/66), done.
```

Alternatively, the code can be downloaded in a ‘zip’ file by clicking:

Clone or download → Download Zip

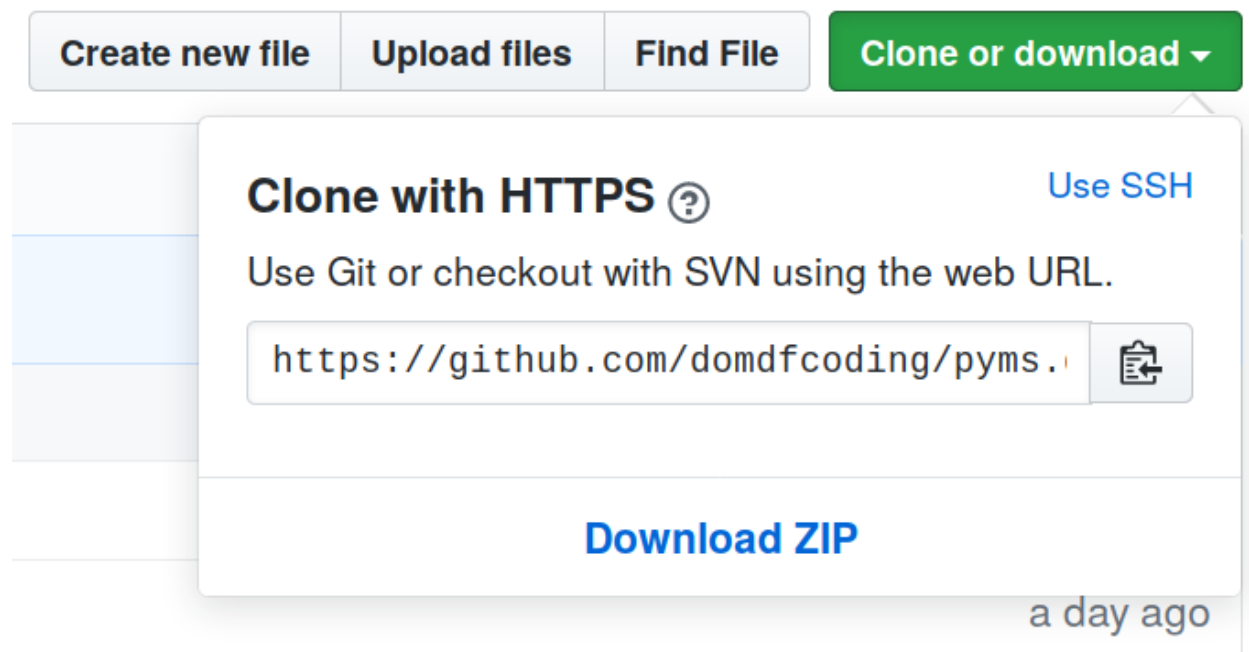


Fig. 1: Downloading a ‘zip’ file of the source code

7.1 Building from source

The recommended way to build `gunshotmatch-reports` is to use `tox`:

```
$ tox -e build
```

The source and wheel distributions will be in the directory `dist`.

If you wish, you may also use `pep517.build` or another **PEP 517**-compatible build tool.

License

`gunshotmatch-reports` is licensed under the [MIT License](#)

A short and simple permissive license with conditions only requiring preservation of copyright and license notices. Licensed works, modifications, and larger works may be distributed under different terms and without source code.

Permissions

- Commercial use – The licensed material and derivatives may be used for commercial purposes.
- Modification – The licensed material may be modified.
- Distribution – The licensed material may be distributed.
- Private use – The licensed material may be used and modified in private.

Conditions

- License and copyright notice – A copy of the license and copyright notice must be included with the licensed material.

Limitations

- Liability – This license includes a limitation of liability.
- Warranty – This license explicitly states that it does NOT provide any warranty.

[See more information on choosealicense.com](#) ⇒

```
Copyright (c) 2024 Dominic Davis-Foster
```

```
Permission is hereby granted, free of charge, to any person obtaining a copy  
of this software and associated documentation files (the "Software"), to deal  
in the Software without restriction, including without limitation the rights  
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
copies of the Software, and to permit persons to whom the Software is  
furnished to do so, subject to the following conditions:
```

```
The above copyright notice and this permission notice shall be included in all  
copies or substantial portions of the Software.
```

```
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,  
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF  
MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT.  
IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM,  
DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR  
OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE  
OR OTHER DEALINGS IN THE SOFTWARE.
```


Python Module Index

g

`gunshotmatch_reports.alignment`, [5](#)
`gunshotmatch_reports.chromatogram`, [7](#)
`gunshotmatch_reports.peak`s, [9](#)
`gunshotmatch_reports.utils`, [13](#)

Symbols

`__repr__()` (*CSVRow* method), 5
`__repr__()` (*PeakSummary* method), 11

A

`area` (*namedtuple* field)
 CSVRow (*namedtuple* in
 gunshotmatch_reports.alignment), 5
 PeakSummary (*namedtuple* in
 gunshotmatch_reports.peaks), 11
`area_percentage` (*namedtuple* field)
 CSVRow (*namedtuple* in
 gunshotmatch_reports.alignment), 5
 PeakSummary (*namedtuple* in
 gunshotmatch_reports.peaks), 11
`area_percentages` (*PeakMetadataTable* attribute),
 10

B

`build_chromatogram_report()` (*in module*
 gunshotmatch_reports.chromatogram), 7
`build_peak_report()` (*in module*
 gunshotmatch_reports.peaks), 11

C

`csv_two_projects()` (*in module*
 gunshotmatch_reports.alignment), 6
`csv_two_projects_and_unknown()` (*in module*
 gunshotmatch_reports.alignment), 6
CSVReports (*class* in *gunshotmatch_reports.peaks*), 9
CSVRow (*namedtuple* in
 gunshotmatch_reports.alignment), 5
 `area` (*namedtuple* field), 5
 `area_percentage` (*namedtuple* field), 5
 `match_factor` (*namedtuple* field), 5
 `name` (*namedtuple* field), 5
 `peak_no` (*namedtuple* field), 5
 `rt` (*namedtuple* field), 5

E

`extend_list()` (*in module*
 gunshotmatch_reports.utils), 13

F

`figure_to_drawing()` (*in module*
 gunshotmatch_reports.utils), 13

G

`get_csv_data()` (*in module*
 gunshotmatch_reports.alignment), 6
`get_summary_for_peak()` (*PeakMetadataTable*
 method), 10
`get_table_for_peak()` (*PeakMetadataTable*
 method), 10
gunshotmatch_reports.alignment
 module, 5
gunshotmatch_reports.chromatogram
 module, 7
gunshotmatch_reports.peaks
 module, 9
gunshotmatch_reports.utils
 module, 13

H

`header()` (*CSVRow* class method), 5

M

`match_factor` (*namedtuple* field)
 CSVRow (*namedtuple* in
 gunshotmatch_reports.alignment), 5
`max_peak_number` (*PeakMetadataTable* attribute), 11
`mf` (*namedtuple* field)
 PeakSummary (*namedtuple* in
 gunshotmatch_reports.peaks), 11
MIT License, 21
module
 gunshotmatch_reports.alignment, 5
 gunshotmatch_reports.chromatogram, 7
 gunshotmatch_reports.peaks, 9
 gunshotmatch_reports.utils, 13

N

`name` (*namedtuple* field)
 CSVRow (*namedtuple* in
 gunshotmatch_reports.alignment), 5
 PeakSummary (*namedtuple* in
 gunshotmatch_reports.peaks), 11

`num_rows` (*PeakMetadataTable* attribute), 11

O

`overview_csv()` (*CSVReports* method), 9

P

`peak_no` (*namedtuple* field)
 `CSVRow` (*namedtuple* in
 gunshotmatch_reports.alignment), 5
 `PeakSummary` (*namedtuple* in
 gunshotmatch_reports.peaks), 11
`PeakMetadataTable` (class in
 gunshotmatch_reports.peaks), 10
`PeakSummary` (*namedtuple* in
 gunshotmatch_reports.peaks), 11
 `area` (*namedtuple* field), 11
 `area_percentage` (*namedtuple* field), 11
 `mf` (*namedtuple* field), 11
 `name` (*namedtuple* field), 11
 `peak_no` (*namedtuple* field), 11
 `rejected` (*namedtuple* field), 11
 `rt` (*namedtuple* field), 11
`project` (*PeakMetadataTable* attribute), 11
Python Enhancement Proposals
 PEP 517, 20

R

`rejected` (*namedtuple* field)
 `PeakSummary` (*namedtuple* in
 gunshotmatch_reports.peaks), 11
`rt` (*namedtuple* field)
 `CSVRow` (*namedtuple* in
 gunshotmatch_reports.alignment), 5
 `PeakSummary` (*namedtuple* in
 gunshotmatch_reports.peaks), 11

S

`save_pdf()` (in module *gunshotmatch_reports.utils*),
 13
`save_svg()` (in module *gunshotmatch_reports.utils*),
 13
`scale()` (in module *gunshotmatch_reports.utils*), 14
`summary_csv()` (*CSVReports* method), 10