

---

# **pydamage**

*Release 0.7*

**Maxime Borry**

**Mar 08, 2022**



## CONTENTS:

<b>1</b>	<b>PyDamage</b>	<b>3</b>
1.1	Installation . . . . .	3
1.2	Quick start . . . . .	4
1.3	CLI help . . . . .	4
1.4	Documentation . . . . .	4
1.5	Cite . . . . .	4
<b>2</b>	<b>API</b>	<b>5</b>
<b>3</b>	<b>CLI</b>	<b>7</b>
3.1	pydamage . . . . .	7
<b>4</b>	<b>Output</b>	<b>11</b>
4.1	pydamage_results.csv . . . . .	11
4.2	pydamage_filtered_results.csv . . . . .	12
4.3	Plots . . . . .	12
4.4	Example . . . . .	12
<b>5</b>	<b>Troubleshooting</b>	<b>15</b>
5.1	My alignment files don't have a MD tag . . . . .	15
<b>6</b>	<b>Indices and tables</b>	<b>17</b>
	<b>Python Module Index</b>	<b>19</b>
	<b>Index</b>	<b>21</b>



Homepage: [github.com/maxibor/pydamage](https://github.com/maxibor/pydamage)





## PYDAMAGE

Pydamage, is a Python software to automate the process of contig damage identification and estimation. After modelling the ancient DNA damage using the C to T transitions, Pydamage uses a likelihood ratio test to discriminate between truly ancient, and modern contigs originating from sample contamination.

### 1.1 Installation

#### 1.1.1 With conda (recommended)

```
conda install -c bioconda pydamage
```

#### 1.1.2 With pip

```
pip install pydamage
```

#### 1.1.3 Install from source to use the development version

Using pip

```
pip install git+ssh://git@github.com/maxibor/pydamage.git@dev
```

By cloning in a dedicated conda environment

```
git clone git@github.com:maxibor/pydamage.git
cd pydamage
git checkout dev
conda env create -f environment.yml
conda activate pydamage
pip install -e .
```

## 1.2 Quick start

```
pydamage --outdir result_directory analyze aligned.bam
```

Note that if you specify `--outdir`, it has to be before the PyDamage subcommand, example: `pydamage --outdir test filter pydamage_results.csv`

## 1.3 CLI help

Command line interface help message

```
pydamage --help
```

## 1.4 Documentation

[pydamage.readthedocs.io](http://pydamage.readthedocs.io)

## 1.5 Cite

PyDamage has been published in PeerJ: [10.7717/peerj.11845](https://doi.org/10.7717/peerj.11845)

```
@article{borry_pydamage_2021,  
  author = {Borry, Maxime and Hübner, Alexander and Rohrlach, Adam B. and Warinner,   
↪Christina},  
  doi = {10.7717/peerj.11845},  
  issn = {2167-8359},  
  journal = {PeerJ},  
  language = {en},  
  month = {July},  
  note = {Publisher: PeerJ Inc.},  
  pages = {e11845},  
  shorttitle = {PyDamage},  
  title = {PyDamage: automated ancient damage identification and estimation for   
↪contigs in ancient DNA de novo assembly},  
  url = {https://peerj.com/articles/11845},  
  urldate = {2021-07-27},  
  volume = {9},  
  year = {2021}  
}
```



`pydamage.main.pydamage_analyze(bam, wlen=30, show_al=False, process=1, outdir="", plot=False, verbose=False, force=False, group=False)`

Runs the pydamage analysis for each reference separately

**Parameters**

- **bam** (*str*) – Path to alignment (sam/bam/cram) file
- **wlen** (*int*) – window length
- **show\_al** (*bool*) – print alignments representations
- **process** (*int*) – Number of processes for parellel computing
- **outdir** (*str*) – Path to output directory
- **verbose** (*bool*) – verbose mode
- **force** (*bool*) – force overwriting of results directory

**Returns** pandas DataFrame containg pydamage results

**Return type** pd.DataFrame



To access the help menu:

```
$ pydamage --help
```

The list of arguments of options is detailed below

## 3.1 pydamage

PyDamage: Damage parameter estimation for ancient DNA

Author: Maxime Borry

Contact: <borry[at]shh.mpg.de>

Homepage & Documentation: [github.com/maxibor/pydamage](https://github.com/maxibor/pydamage)

```
pydamage [OPTIONS] COMMAND [ARGS] ...
```

### Options

**--version**

Show the version and exit.

**-o, --outdir <outdir>**

Output directory

**Default** pydamage\_results

### 3.1.1 analyze

Run the PyDamage analysis

BAM: path to BAM/SAM/CRAM alignment file. MD tags need to be set.

```
pydamage analyze [OPTIONS] BAM
```

## Options

- w, --wlen** <wlen>  
Window length (in bp) for damage modeling  
**Default** 20
- p, --process** <process>  
Number of processes for parallel computing  
**Default** 2
- s, --show\_al**  
Display alignments representations
- pl, --plot**  
Write damage fitting plots to disk
- vv, --verbose**  
Verbose mode
- f, --force**  
Force overwriting of results directory
- g, --group**  
Use entire BAM file as single reference for analysis (ignore reference headers)

## Arguments

- BAM**  
Required argument

### 3.1.2 cite

Get pydamage citation in bibtex format

```
pydamage cite [OPTIONS]
```

### 3.1.3 filter

Filter PyDamage results on predicted accuracy and qvalue thresholds.

CSV: path to PyDamage result file

```
pydamage filter [OPTIONS] CSV
```

## Options

**-t, --threshold <threshold>**

Predicted accuracy filtering threshold. Set to 0 for finding threshold with kneed method

**Default** 0.5

## Arguments

**CSV**

Required argument



Pydamage generates both a tabular and a visual output.

The tabular outputs are comma-separated file (.csv) with the following columns, for each analysed reference:

### 4.1 pydamage\_results.csv

- **reference**: name of the reference genome/contig
- **predicted\_accuracy**: Predicted accuracy of Pydamage prediction, from the GLM modelling
- **null\_model\_p0**: parameter  $p_0$  of the null model
- **null\_model\_p0\_stdev**: standard error of the null model parameter  $p_0$
- **damage\_model\_p**: parameter  $p$  of the damage model
- **damage\_model\_p\_stdev**: standard error of the parameter  $p$  of the damage model
- **damage\_model\_pmin**: parameter  $p_{\min}$  of the damage model. *This is the modelled damage baseline*
- **damage\_model\_pmin\_stdev**: standard error of the parameter  $p_{\min}$  of the damage model
- **damage\_model\_pmax**: parameter  $p_{\max}$  of the damage model. *This is the modelled amount of damage on the 5' end.*
- **damage\_model\_pmax\_stdev**: standard error of the parameter  $p_{\max}$  of the damage model
- **pvalue**: p-value calculated from the likelihood-ratio test-statistic using a chi-squared distribution
- **qvalue**: p-value corrected for multiple testing using Benjamini-Hochberg procedure. *Only computed when multiple references are used*
- **RMSE**: residual mean standard error of the model fit of the damage model
- **nb\_reads\_aligned**: number of aligned reads
- **coverage**: average coverage along the reference genome
- **CtoT-N**: Proportion of CtoT substitutions observed at position N from 5' end
- **GtoA-N**: Proportion of GtoA substitutions observed at position N from 5'

## 4.2 pydamage\_filtered\_results.csv

Same file as above, but with contigs filtered with `qvalue <= 0.05` and `predicted_accuracy >= threshold` with a user defined filtering threshold (default = 0.5), or determined with the `kneedle` method.

## 4.3 Plots

The visual output are PNG files, one per reference contig. They show the frequency of observed C to T, and G to A transition at the 5' end of the sequencing data and overlay it with the fitted models for both the null and the damage model, including 95% confidence intervals. Furthermore, it provides a “residuals versus fitted” plot to help evaluate the fit of the pydamage damage model. Finally, the plot contains information on the average coverage along the reference and the p-value calculated from the likelihood-ratio test-statistic using a chi-squared distribution.

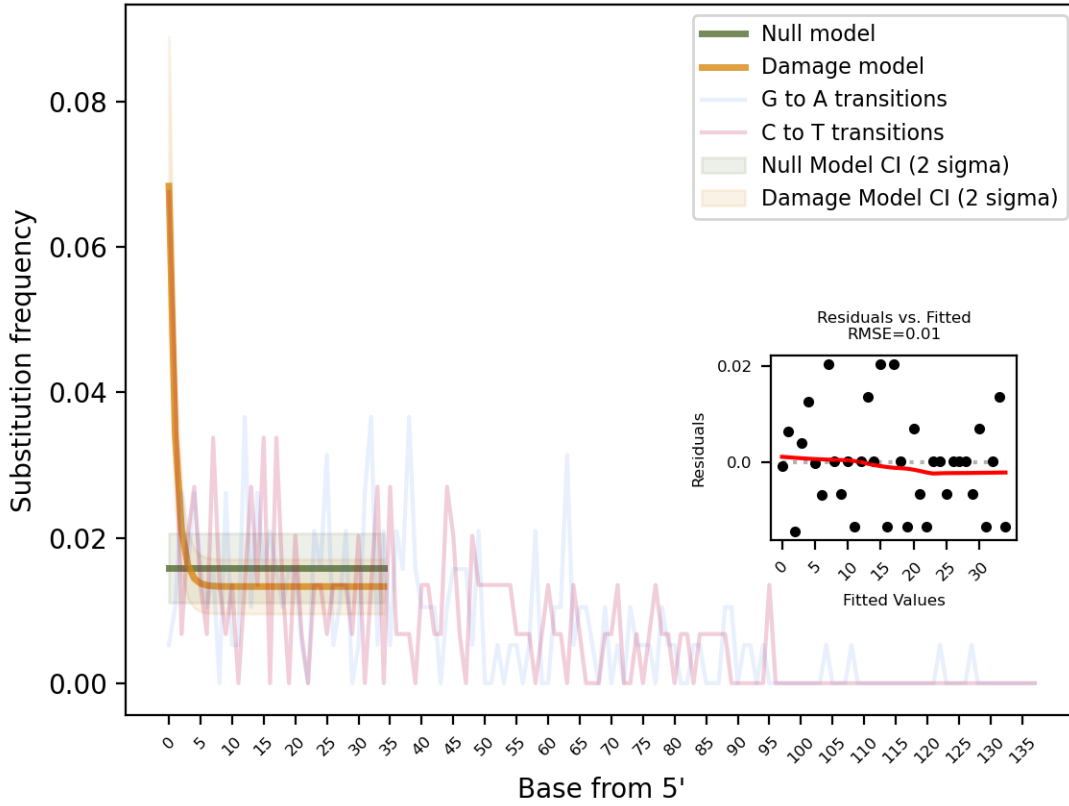
The visual output is only produced when using the `--plot` flag

## 4.4 Example

- **Tabular output**
  - `pydamage_results.csv`
  - `pydamage_filtered_results.csv`
- **Visual output**



NZ\_JHCB02000011.1  
 coverage: 0.2 - pvalue<0.001





## TROUBLESHOOTING

### 5.1 My alignment files don't have a MD tag

You can use `samtools calmd` to set the MD tag

Example:

```
samtools calmd -b alignment.bam reference.fasta > aln.bam
```



## INDICES AND TABLES

- [genindex](#)
- [modindex](#)
- [search](#)



## PYTHON MODULE INDEX

p

`pydamage.main`, 5





## Symbols

--force  
     pydamage-analyze command line option, 8  
 --group  
     pydamage-analyze command line option, 8  
 --outdir  
     pydamage command line option, 7  
 --plot  
     pydamage-analyze command line option, 8  
 --process  
     pydamage-analyze command line option, 8  
 --show\_al  
     pydamage-analyze command line option, 8  
 --threshold  
     pydamage-filter command line option, 9  
 --verbose  
     pydamage-analyze command line option, 8  
 --version  
     pydamage command line option, 7  
 --wlen  
     pydamage-analyze command line option, 8  
 -f  
     pydamage-analyze command line option, 8  
 -g  
     pydamage-analyze command line option, 8  
 -o  
     pydamage command line option, 7  
 -p  
     pydamage-analyze command line option, 8  
 -pl  
     pydamage-analyze command line option, 8  
 -s  
     pydamage-analyze command line option, 8  
 -t  
     pydamage-filter command line option, 9  
 -vv  
     pydamage-analyze command line option, 8  
 -w  
     pydamage-analyze command line option, 8

## B

BAM

pydamage-analyze command line option, 8

## C

CSV

pydamage-filter command line option, 9

## M

module

pydamage.main, 5

## P

pydamage command line option

--outdir, 7

--version, 7

-o, 7

pydamage.main

module, 5

pydamage\_analyze() (in module pydamage.main), 5

pydamage-analyze command line option

--force, 8

--group, 8

--plot, 8

--process, 8

--show\_al, 8

--verbose, 8

--wlen, 8

-f, 8

-g, 8

-p, 8

-pl, 8

-s, 8

-vv, 8

-w, 8

BAM, 8

pydamage-filter command line option

--threshold, 9

-t, 9

CSV, 9