

# Package ‘pawscore’

May 9, 2026

**Date** 2023-09-18

**Version** 1.0.3

**Title** Pain Assessment at Withdrawal Speeds (PAWS)

**Description** Automated pain scoring from paw withdrawal tracking data. Based on Jones et al. (2020) ``A machine-vision approach for automated pain measurement at millisecond timescales" <[doi:10.7554/eLife.57258](https://doi.org/10.7554/eLife.57258)>.

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 3.6), brglm2 (>= 0.6)

**Imports** signal (>= 0.7)

**RoxygenNote** 7.2.0

**NeedsCompilation** no

**Author** Colin Twomey [aut, cre],  
Jessica Jones [aut],  
William Foster [aut],  
Joshua Plotkin [aut],  
Ishmail Abdus-Saboor [aut]

**Maintainer** Colin Twomey <[crtwomey@gmail.com](mailto:crtwomey@gmail.com)>

**Repository** CRAN

**Date/Publication** 2023-09-18 15:10:02 UTC

## Contents

create_pain_model . . . . .	2
create_strain_standard . . . . .	3
default_parameters . . . . .	3
default_standards . . . . .	4
extract_features . . . . .	4
jones2020.tracks . . . . .	5
pain_class . . . . .	5

pain_score . . . . .	6
pawsscore . . . . .	7
set_parameters . . . . .	7

<b>Index</b>	<b>8</b>
--------------	----------

---

create_pain_model	<i>Create a new pain model</i>
-------------------	--------------------------------

---

## Description

Create a new pain model based on a set of paw features, corresponding pain stimuli, and corresponding strain identity.

## Usage

```
create_pain_model(
  paw.features,
  strains = NULL,
  pain.stimulus = NULL,
  strain.standards = jones2020.standards,
  feature.set = c("post.peak", "pre.peak")
)
```

## Arguments

paw.features	the paw features returned by <a href="#">extract_features</a>
strains	a vector containing strain information for each mouse
pain.stimulus	a vector containing stimulus information for each mouse
strain.standards	z-scores (centering and scaling) information by strain
feature.set	use either pre-peak or post-peak features

## Value

pain model

---

`create_strain_standard`*Create new strain standards*

---

**Description**

Creates a new reference centering and scaling for the paw features of the given strain or strains.

**Usage**

```
create_strain_standard(paw.features, strain)
```

**Arguments**

<code>paw.features</code>	list of extracted paw features
<code>strain</code>	character string or a vector of strain names, each of which is a character string, matching in length and order the list of paw features.

**Value**

list of pre-peak and post-peak strain standards, indexed by strain

---

`default_parameters`*Default parameters for extracting paw features*

---

**Description**

Default parameters for extracting paw features

**Usage**

```
default_parameters()
```

**Value**

parameters used for Jones et al. (2020)

---

default_standards	<i>Default strain-based standards for paw features</i>
-------------------	--

---

**Description**

Default strain-based standards for paw features

**Usage**

```
default_standards()
```

**Value**

standards used for Jones et al. (2020)

---

extract_features	<i>Extract features for paw time series</i>
------------------	---

---

**Description**

Extract features for paw time series

**Usage**

```
extract_features(
  x,
  y = NULL,
  parameters = default_parameters(),
  diagnostics = FALSE
)
```

**Arguments**

x	time series of horizontal paw movement. Alternatively, a two column matrix of x and y time series, respectively.
y	time series of vertical paw movement, or NULL if x is a two column matrix.
parameters	contains information about frames per second, filtering, windowing, and thresholds, for paw features (see <a href="#">default_parameters</a> , or use <a href="#">set_parameters</a> to modify the defaults).
diagnostics	set to TRUE will record intermediate values used when computing paw features. This information can be helpful for debugging parameter choices. The default, FALSE, is to not record these values.

**Value**

pre-peak and post-peak paw features (plus diagnostics, if enabled)

## Examples

```
# example usage with a track from Jones et al. (2020)
track <- jones2020.tracks[[1]]
features <- extract_features(track$time.series)
```

---

jones2020.tracks	<i>Jones et al. (2020) paw trajectory data</i>
------------------	--

---

## Description

Paw trajectory time series, strain, and stimulus information for the cohort 1 data used in Jones et al.

## Usage

```
jones2020.tracks
```

## Format

A list of paw trajectories, each containing:

`id` A unique id for each mouse  
`strain` The corresponding mouse strain  
`stimulus` The stimulus used  
`time.series` The paw trajectory when stimulus was applied

## Source

Jones et al. (2020) A machine-vision approach for automated pain measurement at millisecond timescales. eLife 9:e57258 doi:[10.7554/eLife.57258](https://doi.org/10.7554/eLife.57258)

---

pain_class	<i>Identify pain class based on pain score</i>
------------	--

---

## Description

Convenience function to convert pain scores to binary or trinary pain classifications, e.g. pain / non-pain or pain / low-pain / high-pain. Note that trinary classifications are only valid for pain models fit with trinary classes (e.g. Jones et al. 2020). Otherwise the scale of the score is arbitrary, and the boundary between low and high pain is not scaled to be 1.

## Usage

```
pain_class(score, type = c("binary", "trinary"))
```

**Arguments**

score            pain scores generated by [pain\\_score](#)  
 type            binary or trinary (ternary) pain classifications

**Value**

one or more pain classes

---

pain_score	<i>Scoring pain from paw features</i>
------------	---------------------------------------

---

**Description**

Returns a pain score based on Jones et al. 2020 or the given pain.model. Pain scores are standardized so that increasingly positive (negative) values correspond to increasingly likely to be painful (unlikely to be painful) experiences. Pain scores based on pain models derived from trinary pain stimuli will additionally be scaled such that the transition between low and high pain is occurs at 1. Scores are not comparable across pain models.

**Usage**

```

pain_score(
  paw.features,
  strains = NULL,
  strain.standards = jones2020.standards,
  feature.set = c("post.peak", "pre.peak"),
  pain.model = NULL
)

```

**Arguments**

paw.features    the paw features returned by [extract\\_features](#)  
 strains        a vector containing strain information for each mouse  
 strain.standards    z-scores (centering and scaling) information by strain  
 feature.set    use either pre-peak or post-peak features  
 pain.model    a pain model returned by [create\\_pain\\_model](#) or NULL, in which case the model from Jones et al. 2020 is used.

**Value**

one or more pain scores as a vector

---

pawsscore	<i>paws: Pain Assessment at Withdrawal Speeds</i>
-----------	---

---

**Description**

Automated pain scoring from paw withdrawal tracking data based on Jones et al. (2020) A machine-vision approach for automated pain measurement at millisecond timescales. This R package takes paw trajectory data in response to a stimulus and provides an automated scoring of pain.

---

set_parameters	<i>Convenience function for changing parameters</i>
----------------	---

---

**Description**

Convenience function for changing parameters

**Usage**

```
set_parameters(..., based.on = default_parameters())
```

**Arguments**

...	name and value of parameters to set (e.g. fps=1500)
based.on	an existing complete set of parameters to modify (see <a href="#">default_parameters</a> ).

**Value**

the modified list of parameters

# Index

## \* datasets

jones2020.tracks, 5

create\_pain\_model, 2, 6

create\_strain\_standard, 3

default\_parameters, 3, 4, 7

default\_standards, 4

extract\_features, 2, 4, 6

jones2020.tracks, 5

pain\_class, 5

pain\_score, 6, 6

pawsscore, 7

set\_parameters, 4, 7