

Package: PhysioPipe (via r-universe)

October 3, 2026

Type Package

Title Reproducible 'targets' Pipelines for Physiological Signal Analysis

Version 0.1.1

Description A collection of reusable 'targets' target factories for the Physio ecosystem: drop-in pipelines for the standard single-modality analyses (ECG heart-rate variability, EEG band power, EMG amplitude/fatigue, EDA skin conductance) plus multi-record cohort branching and Quarto reporting. Each factory returns a list of targets so a whole reproducible '_targets.R' is a couple of lines, with incremental re-runs and byte-level provenance.

Depends R (>= 4.2)

Imports targets, grDevices, graphics, stats

Suggests PhysioCore, PhysioIO, PhysioECG, PhysioAnalysis, PhysioEMG, PhysioEDA, PhysioPreprocess, SummarizedExperiment, S4Vectors, arrow, ragg, tarchetypes, quarto, testthat (>= 3.2.0), knitr, rmarkdown

VignetteBuilder knitr

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

URL <https://github.com/x-biosignal/PhysioPipe>,
<https://x-biosignal.github.io/PhysioPipe/>

BugReports <https://github.com/x-biosignal/PhysioExperiment/issues>

Config/testthat/edition 3

Config/pak/sysreqs libglpk-dev libxml2-dev

Repository <https://x-biosignal.r-universe.dev>

Date/Publication 2026-10-03 13:11:48 UTC

RemoteUrl <https://github.com/x-biosignal/PhysioPipe>
RemoteRef HEAD
RemoteSha a36920226b9b0c1e70177033de6ccd89c91f2554

Contents

pp_bind_cohort	2
pp_ecg_demo_files	3
pp_ecg_read	3
pp_hrv_combine	4
pp_plot_tachogram	4
pp_simulate	5
pp_wfdb_files	5
pp_write_parquet	6
tar_ecg_hrv	7
tar_eda_scr	8
tar_eeg_bandpower	8
tar_emg_features	9
tar_physio_report	10

Index	11
--------------	-----------

pp_bind_cohort	<i>Stack per-branch data frames from a cohort run into one table</i>
----------------	--

Description

Stack per-branch data frames from a cohort run into one table

Usage

pp_bind_cohort(branches, id = NULL)

Arguments

- branches A list of per-record data frames (dynamic-branch output).
- id Optional vector of record ids, added as a record column.

Value

A single row-bound data frame.

Examples

```
branches <- list(
  data.frame(sdn = 50),
  data.frame(sdn = 61)
)
pp_bind_cohort(branches, id = c("rec100", "rec101"))
```

pp_ecg_demo_files	<i>File paths of the bundled demo ECG record (MIT-BIH 100 excerpt)</i>
-------------------	--

Description

File paths of the bundled demo ECG record (MIT-BIH 100 excerpt)

Usage

```
pp_ecg_demo_files()
```

Value

Character vector of the record's companion files (.dat, .hea).

pp_ecg_read	<i>Read a WFDB record (by companion files or base) into a PhysioExperiment</i>
-------------	--

Description

Read a WFDB record (by companion files or base) into a PhysioExperiment

Usage

```
pp_ecg_read(record)
```

Arguments

record	Either the companion files (c(".dat", ".hea")) or a single record base path (no extension).
--------	---

Value

A PhysioExperiment.

pp_hrv_combine	<i>Join time- and frequency-domain HRV into one row per channel</i>
----------------	---

Description

Join time- and frequency-domain HRV into one row per channel

Usage

```
pp_hrv_combine(hrv_time, hrv_freq)
```

Arguments

hrv_time	Output of <code>PhysioECG::ecgHRVtime()</code> .
hrv_freq	Output of <code>PhysioECG::ecgHRVfreq()</code> .

Value

A data frame merged on channel.

Examples

```
hrv_time <- data.frame(channel = "MLII", sdn = 50, rmssd = 42)
hrv_freq <- data.frame(channel = "MLII", lf = 400, hf = 330)
pp_hrv_combine(hrv_time, hrv_freq)
```

pp_plot_tachogram	<i>Write a simple RR tachogram PNG (headless) and return its path</i>
-------------------	---

Description

Write a simple RR tachogram PNG (headless) and return its path

Usage

```
pp_plot_tachogram(rr, path = "figures/tachogram.png")
```

Arguments

rr	Output of <code>PhysioECG::ecgRRintervals()</code> .
path	Output PNG path.

Value

path (so the target can be `format = "file"`).

pp_simulate

*Simulate a synthetic PhysioExperiment for demos and tests***Description**

Deterministic (seeded) synthetic signals per modality. Clearly synthetic — for exercising a pipeline’s mechanics, not for making claims.

Usage

```
pp_simulate(
  modality = c("eeg", "emg", "eda"),
  seed = 1L,
  n_channels = NULL,
  duration = NULL,
  sampling_rate = NULL
)
```

Arguments

modality One of "eeg", "emg", "eda".
 seed Integer RNG seed (fixed so pipelines stay reproducible).
 n_channels, duration, sampling_rate
 Optional geometry overrides.

Value

A PhysioExperiment (time x channels) with a raw assay.

Examples

```
if (requireNamespace("PhysioCore", quietly = TRUE)) {
  pe <- pp_simulate("eeg")
}
```

pp_wfdb_files

*Companion files of a WFDB record base***Description**

Companion files of a WFDB record base

Usage

```
pp_wfdb_files(base)
```

Arguments

base Record base path (no extension), e.g. "data/100".

Value

Character vector of existing base.* signal/header files.

Examples

```
# Create a stand-in record base and list its WFDB companion files.
base <- file.path(tempdir(), "100")
file.create(paste0(base, c(".dat", ".hea", ".atr")))
basename(pp_wfdb_files(base))
```

pp_write_parquet	<i>Write a data frame to Parquet (language-neutral interchange)</i>
------------------	---

Description

Intermediate/terminal tables written as Parquet can be read by Python (pandas/pyarrow), DuckDB, and Nextflow processes — unlike the default .rds target store. This is the seam that keeps pipelines orchestrator-agnostic.

Usage

```
pp_write_parquet(df, path)
```

Arguments

df A data frame.

path Output .parquet path.

Value

path (so the target can be format = "file").

Examples

```
if (requireNamespace("arrow", quietly = TRUE)) {
  path <- pp_write_parquet(data.frame(a = 1:3, b = 4:6),
                           file.path(tempdir(), "demo.parquet"))
  file.exists(path)
}
```

tar_ecg_hrv	<i>ECG heart-rate-variability pipeline (target factory)</i>
-------------	---

Description

Builds a standard resting-ECG pipeline as a list of targets: read -> lead -> R-peaks -> RR -> {HRV time, HRV freq} -> plus an RR-tachogram figure. With more than one record it fans out via dynamic branching (one branch per record) and adds a combined cohort table.

Usage

```
tar_ecg_hrv(
  name = "ecg",
  records = NULL,
  channel = 1L,
  detector = "pan_tompkins"
)
```

Arguments

name	Target-name prefix (a string; targets are "<name>_pe" etc.).
records	NULL to use the bundled demo record (MIT-BIH 100 excerpt), or a character vector of WFDB record bases (paths without extension).
channel	Lead index to analyse (MIT-BIH 100: 1 = MLII, 2 = V5).
detector	ecgDetectRpeaks() method (e.g. "pan_tompkins", "hamilton").

Value

A list of target objects to splice into a `_targets.R` pipeline.

See Also

[tar_eeg_bandpower\(\)](#), [tar_emg_features\(\)](#), [tar_eda_scr\(\)](#)

Examples

```
# The factory returns target objects to splice into a `_targets.R` list();
# building them does not run the pipeline.
targs <- tar_ecg_hrv()          # bundled demo record (single-record pipeline)
length(targs)
class(targs[[1]])
# Cohort fan-out (dynamic branching) when given several record bases:
length(tar_ecg_hrv(records = c("data/100", "data/101")))
```

tar_eda_scr	<i>EDA skin-conductance pipeline (target factory)</i>
-------------	---

Description

signal -> tonic/phasic decomposition -> SCR features.

Usage

```
tar_eda_scr(name = "eda", source = NULL)
```

Arguments

name	Target-name prefix.
source	Expression producing a PhysioExperiment, or NULL for a synthetic EDA demo (drifting SCL + discrete SCRs).

Value

A list of targets.

Examples

```
# Returns target objects (synthetic EDA demo: drifting SCL + discrete SCRs).
targs <- tar_eda_scr()
length(targs)
```

tar_eeg_bandpower	<i>EEG band-power pipeline (target factory)</i>
-------------------	---

Description

signal -> band power per channel. Band power is computed from the Welch PSD integrated over each band, so it is inherently frequency-selective (no pre-filter needed).

Usage

```
tar_eeg_bandpower(
  name = "eeg",
  source = NULL,
  bands = list(alpha = c(8, 13), beta = c(13, 30)),
  method = "welch",
  relative = FALSE
)
```

Arguments

name	Target-name prefix.
source	Expression producing a <code>PhysioExperiment</code> , or <code>NULL</code> for a synthetic EEG demo (10 Hz alpha + noise).
bands	Named list of <code>c(low, high)</code> Hz bands.
method	Welch or wavelet PSD.
relative	Report band power relative to total.

Value

A list of targets.

Examples

```
# Returns target objects (synthetic EEG demo); splice into a `_targets.R`.
targs <- tar_eeg_bandpower()
length(targs)
# Point it at a real file instead of the synthetic demo:
# tar_eeg_bandpower(source = quote(PhysioIO::readEDF("sub01.edf")))
```

tar_emg_features	<i>EMG amplitude / fatigue pipeline (target factory)</i>
------------------	--

Description

signal -> linear envelope and signal -> spectral fatigue (median/mean frequency + RMS over windows).

Usage

```
tar_emg_features(name = "emg", source = NULL)
```

Arguments

name	Target-name prefix.
source	Expression producing a <code>PhysioExperiment</code> , or <code>NULL</code> for a synthetic bursty EMG demo.

Value

A list of targets.

Examples

```
# Returns target objects (synthetic bursty-EMG demo).
targs <- tar_emg_features()
length(targs)
```

tar_physio_report	<i>Quarto report pipeline (target factory)</i>
-------------------	--

Description

Wraps `tarchetypes::tar_quarto_raw()` so a rendered report becomes a tracked target that re-renders when any embedded result changes. The `.qmd` reads results with `targets::tar_read()`; `targets` discovers those dependencies automatically.

Usage

```
tar_physio_report(name = "report", path = NULL)
```

Arguments

name	Target name (string).
path	Path to the <code>.qmd</code> . Defaults to the bundled template.

Value

A single target object (requires the `tarchetypes` and `quarto` packages plus the Quarto CLI at build time).

Examples

```
# list(tar_ecg_hrv(), tar_physio_report())
```

Index

pp_bind_cohort, [2](#)
pp_ecg_demo_files, [3](#)
pp_ecg_read, [3](#)
pp_hrv_combine, [4](#)
pp_plot_tachogram, [4](#)
pp_simulate, [5](#)
pp_wfdb_files, [5](#)
pp_write_parquet, [6](#)

tar_ecg_hrv, [7](#)
tar_eda_scr, [8](#)
tar_eda_scr(), [7](#)
tar_eeg_bandpower, [8](#)
tar_eeg_bandpower(), [7](#)
tar_emg_features, [9](#)
tar_emg_features(), [7](#)
tar_physio_report, [10](#)