

Package ‘gp3tools’

June 30, 2026

Type Package

Title Import, Inspect, Analyse, and Report Gazeport GP3 Exports

Version 1.0.2

Description Imports, inspects, cleans, summarises, models, and reports Gazeport GP3 and Gazeport Analysis CSV exports. Supports offline workflows for all-gaze, fixation, pupil, area-of-interest, transition, time-course, quality-audit, and manuscript-reporting analyses.

License MIT + file LICENSE

LazyData true

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

Imports dplyr, ggplot2, readr, rlang, stringr, tibble, tidyr, withr

Suggests DHARMA, emmeans, eyetools, glmmTMB, knitr, lme4, magick, mgcv, pracma, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

URL <https://github.com/stefanosbalaskas/gp3tools>

BugReports <https://github.com/stefanosbalaskas/gp3tools/issues>

NeedsCompilation no

Author Stefanos Balaskas [aut, cre]

Maintainer Stefanos Balaskas <s.balaskas@ac.upatras.gr>

Repository CRAN

Date/Publication 2026-06-30 09:20:02 UTC

Contents

as_gazepoint_master	4
audit_gazepoint_aoi_coding_matrix	6
audit_gazepoint_aoi_geometry	8
audit_gazepoint_aoi_margin_sensitivity	10
audit_gazepoint_aoi_overlap	12
audit_gazepoint_aoi_window_denominators	13
audit_gazepoint_condition_quality_imbalance	14
audit_gazepoint_design_balance	15
audit_gazepoint_event_sync	16
audit_gazepoint_exclusion_flow	17
audit_gazepoint_gaze_signal_quality	18
audit_gazepoint_master	20
audit_gazepoint_post_exclusion_balance	21
audit_gazepoint_pupil_baseline	22
audit_gazepoint_pupil_drift	24
audit_gazepoint_pupil_gaps	25
audit_gazepoint_pupil_imbalance	26
audit_gazepoint_pupil_overlap_risk	27
audit_gazepoint_pupil_reliability	29
audit_gazepoint_stimulus_luminance	30
baseline_correct_gazepoint_pupil	31
check_gazepoint_file_pairs	32
check_gazepoint_model_convergence	33
check_gazepoint_model_overdispersion	34
check_gazepoint_model_singularity	34
check_gazepoint_real_data_readiness	35
check_sampling_rate	37
classify_gazepoint_export	37
compare_gazepoint_nested_models	38
compute_gazepoint_aoi_transition_matrix	39
compute_gazepoint_time_varying_transition_matrix	40
compute_transition_matrix	41
create_gazepoint_analysis_decision_audit	42
create_gazepoint_markovchain_object	43
create_gazepoint_master	44
create_gazepoint_preprocessing_multiverse	45
create_gazepoint_preprocessing_registry	46
create_gazepoint_report	48
create_gazepoint_reporting_checklist	49
diagnose_gazepoint_gamm	50
diagnose_gazepoint_glmm	51
estimate_gazepoint_divergence_point	52
export_gazepoint_master_audit	54
export_gazepoint_model_tables	55
export_gazepoint_tables	56
fit_gazepoint_aoi_gamm	57

fit_gazepoint_aoi_model_sensitivity	59
fit_gazepoint_aoi_window_glmm	60
fit_gazepoint_gca	62
fit_gazepoint_pupil_gamm	63
fit_gazepoint_pupil_pfe_gamm	64
fit_gazepoint_pupil_window_lmm	66
fit_gazepoint_pupil_window_sensitivity	67
fit_gazepoint_transition_count_nb_sensitivity	69
flag_gazepoint_pupil	70
flag_gazepoint_pupil_artifacts	72
flag_gazepoint_pupil_hampel	74
flag_tracking_quality	75
gazepoint_example_aoi_geometry	76
gazepoint_example_aoi_windows	77
gazepoint_example_fixations	78
gazepoint_example_master	79
gazepoint_example_pupil_windows	80
inspect_gazepoint_columns	80
interpolate_gazepoint_pupil	81
interpolate_gazepoint_pupil_pchip	82
plot_gazepoint_aoi_gamm	83
plot_gazepoint_aoi_transition_matrix	85
plot_gazepoint_aoi_verification	86
plot_gazepoint_cluster_results	87
plot_gazepoint_gca	89
plot_gazepoint_model_predictions	90
plot_gazepoint_multiverse_results	92
plot_gazepoint_pupil_preprocessing	92
plot_gazepoint_pupil_status	94
plot_gazepoint_pupil_timecourse	96
plot_sampling_rate	97
plot_tracking_quality	97
plot_transition_heatmap	98
prepare_gazepoint_aoi_gamm_data	99
prepare_gazepoint_aoi_glmm_data	101
prepare_gazepoint_aoi_sequences	102
prepare_gazepoint_cluster_data	103
prepare_gazepoint_eyetools_data	105
prepare_gazepoint_eyetrackingr_data	106
prepare_gazepoint_fixation_aligned_data	108
prepare_gazepoint_gazer_data	109
prepare_gazepoint_gca_data	111
prepare_gazepoint_hmm_data	112
prepare_gazepoint_pupillometryr_data	113
prepare_gazepoint_pupil_gamm_data	115
prepare_gazepoint_pupil_window_model_data	116
prepare_gazepoint_semimarkov_data	117
read_gazepoint	119

read_gazepoint_folder	119
read_gazepoint_summary	120
recalibrate_gazepoint_gaze	121
recommend_gazepoint_exclusions	122
run_gazepoint_aoi_multiverse	124
run_gazepoint_cluster_permutation	125
run_gazepoint_eyetools_fixation_detection	126
run_gazepoint_gazer_crosscheck	128
run_gazepoint_model_leave_one_out	130
run_gazepoint_pupil_multiverse	131
run_gazepoint_workflow	132
save_gazepoint_plots	134
smooth_gazepoint_pupil	135
standardise_gazepoint_names	136
summarise_aoi_samples	137
summarise_fixations	137
summarise_gazepoint_aoi	138
summarise_gazepoint_aoi_entries	138
summarise_gazepoint_aoi_transitions	139
summarise_gazepoint_aoi_trial_features	141
summarise_gazepoint_aoi_windows	142
summarise_gazepoint_clusters	143
summarise_gazepoint_emmeans	144
summarise_gazepoint_fixation_trials	145
summarise_gazepoint_fixed_effects	147
summarise_gazepoint_multiverse_results	148
summarise_gazepoint_pupil	148
summarise_gazepoint_pupil_trial_features	149
summarise_gazepoint_pupil_windows	151
summarise_gazepoint_workflow	152
summarise_tracking_quality	153
tidy_gazepoint_model_summary	153
transform_gazepoint_aoi_empirical_logit	154
validate_gazepoint_master	156
write_gazepoint_outputs	157

Index**159**

as_gazepoint_master	<i>Convert Gazepoint all-gaze data to a master sample table</i>
---------------------	---

Description

Converts a Gazepoint all-gaze export into a standard sample-level table with one row per gaze sample. The returned table keeps Gazepoint identifiers but also adds analysis-friendly columns such as subject, media_id, time_ms, x, y, left_pupil, right_pupil, mean_pupil, valid_sample, missing_gaze, missing_pupil, trackloss, blink, aoi_current, message, and event_type.

Usage

```
as_gazepoint_master(
  data,
  screen_width_px = NULL,
  screen_height_px = NULL,
  source_col = "USER_FILE",
  media_col = "MEDIA_ID",
  media_name_col = "MEDIA_NAME",
  time_col = "TIME",
  coordinate_unit = c("auto", "normalised", "pixels"),
  event_latency_offset_ms = 0
)
```

Arguments

<code>data</code>	A Gazepoint all-gaze data frame, usually <code>gazepoint_example_master</code> .
<code>screen_width_px</code>	Optional screen width in pixels. If supplied and gaze coordinates are detected as normalised 0-1 coordinates, x coordinates are converted to pixels.
<code>screen_height_px</code>	Optional screen height in pixels. If supplied and gaze coordinates are detected as normalised 0-1 coordinates, y coordinates are converted to pixels.
<code>source_col</code>	Column identifying the source/user file.
<code>media_col</code>	Column identifying the Gazepoint media/stimulus.
<code>media_name_col</code>	Column identifying the Gazepoint media/stimulus name.
<code>time_col</code>	Gazepoint time column, usually <code>TIME</code> .
<code>coordinate_unit</code>	One of "auto", "normalised", or "pixels". "auto" detects normalised coordinates when coordinate values are mostly between 0 and 1.
<code>event_latency_offset_ms</code>	Optional timing correction in milliseconds. Positive values shift event/sample time forward.

Details

This function is intended as a bridge between raw Gazepoint exports and more advanced eye-tracking workflows. It does not require an external trial log. Later, experiment-level information such as condition, trial ID, response, accuracy, or reaction time can be joined to the returned table.

Value

A tibble with one row per sample and standardised sample-level eye-tracking columns.

Examples

```
results <- run_gazepoint_workflow(  
  export_dir = system.file("extdata", "gazepoint_realistic_demo_exports", package = "gp3tools"),  
  output_dir = file.path(tempdir(), "gp3_outputs")  
)  
  
master <- as_gazepoint_master(  
  gazepoint_example_master,  
  screen_width_px = 1920,  
  screen_height_px = 1080  
)  
  
dplyr::glimpse(master)
```

audit_gazepoint_aoi_coding_matrix

Audit AOI coding against geometry

Description

Validate observed sample-level AOI labels against AOI labels derived from gaze coordinates and AOI geometry.

Usage

```
audit_gazepoint_aoi_coding_matrix(  
  gaze_data,  
  aoi_geometry,  
  observed_aoi_col = NULL,  
  gaze_x_col = NULL,  
  gaze_y_col = NULL,  
  gaze_stimulus_col = NULL,  
  sample_id_cols = NULL,  
  geometry_aoi_col = NULL,  
  geometry_stimulus_col = NULL,  
  x_min_col = NULL,  
  y_min_col = NULL,  
  x_max_col = NULL,  
  y_max_col = NULL,  
  x_col = NULL,  
  y_col = NULL,  
  width_col = NULL,  
  height_col = NULL,  
  screen_x_range = c(0, 1),  
  screen_y_range = c(0, 1),  
  tie_method = c("ambiguous", "first"),
```

```

    outside_label = "outside",
    ambiguous_label = "ambiguous",
    missing_label = "missing_coordinate",
    observed_outside_values = c("outside", "none", "no_aoi", "non_aoi", "background",
    "off_aoi"),
    max_mismatch_prop = 0.05,
    max_ambiguous_prop = 0.05,
    max_missing_coordinate_prop = 0.2,
    ignore_invalid_geometry = TRUE
)

```

Arguments

gaze_data A data frame containing gaze samples and observed AOI labels.

aoi_geometry A data frame containing AOI geometry definitions.

observed_aoi_col
Observed AOI label column in `gaze_data`.

gaze_x_col Gaze x-coordinate column. If NULL, common aliases are detected automatically.

gaze_y_col Gaze y-coordinate column. If NULL, common aliases are detected automatically.

gaze_stimulus_col
Optional gaze stimulus/media column.

sample_id_cols Optional columns to carry into the sample-level coding audit table.

geometry_aoi_col
AOI label/name column in `aoi_geometry`.

geometry_stimulus_col
Optional stimulus/media column in `aoi_geometry`.

x_min_col Optional AOI left/x-min column.

y_min_col Optional AOI top/y-min column.

x_max_col Optional AOI right/x-max column.

y_max_col Optional AOI bottom/y-max column.

x_col Optional AOI left/x column used with `width_col`.

y_col Optional AOI top/y column used with `height_col`.

width_col Optional AOI width column.

height_col Optional AOI height column.

screen_x_range Numeric length-2 vector defining the screen x range.

screen_y_range Numeric length-2 vector defining the screen y range.

tie_method How to handle samples falling in multiple AOIs. Use "ambiguous" to label them as ambiguous or "first" to use the first matching AOI.

outside_label Label used for samples outside all AOIs.

ambiguous_label
Label used for samples inside multiple AOIs when `tie_method = "ambiguous"`.

missing_label Label used for samples with missing/non-finite gaze coordinates.

observed_outside_values	Character values in observed_aoi_col treated as outside/non-AOI labels.
max_mismatch_prop	Maximum acceptable proportion of comparable samples where observed and geometry-derived AOI labels differ.
max_ambiguous_prop	Maximum acceptable proportion of samples with ambiguous geometry-derived AOI assignment.
max_missing_coordinate_prop	Maximum acceptable proportion of samples with missing/non-finite gaze coordinates.
ignore_invalid_geometry	Logical. If TRUE, AOIs with invalid coordinates or dimensions are excluded before coding validation.

Value

A list with class `gp3_aoi_coding_matrix_audit` containing overview, geometry_summary, sample_coding, coding_matrix, observed_summary, derived_summary, flagged_samples, and settings tables.

audit_gazepoint_aoi_geometry
Audit AOI geometry definitions

Description

Create a publication-level audit of AOI geometry definitions, including AOI size, area, coordinate validity, screen-bound checks, and duplicate geometry.

Usage

```
audit_gazepoint_aoi_geometry(  
  data,  
  aoi_col = NULL,  
  stimulus_col = NULL,  
  x_min_col = NULL,  
  y_min_col = NULL,  
  x_max_col = NULL,  
  y_max_col = NULL,  
  x_col = NULL,  
  y_col = NULL,  
  width_col = NULL,  
  height_col = NULL,  
  screen_x_range = c(0, 1),  
  screen_y_range = c(0, 1),  
  min_width = 0,
```

```

    min_height = 0,
    min_area = 0,
    max_area_prop = 1,
    require_within_screen = TRUE
)

```

Arguments

data	A data frame containing AOI geometry definitions.
aoi_col	AOI label/name column. If NULL, common AOI-name aliases are detected automatically.
stimulus_col	Optional stimulus/media column.
x_min_col	Optional AOI left/x-min column.
y_min_col	Optional AOI top/y-min column.
x_max_col	Optional AOI right/x-max column.
y_max_col	Optional AOI bottom/y-max column.
x_col	Optional AOI left/x column used with width_col.
y_col	Optional AOI top/y column used with height_col.
width_col	Optional AOI width column.
height_col	Optional AOI height column.
screen_x_range	Numeric length-2 vector defining the screen x range.
screen_y_range	Numeric length-2 vector defining the screen y range.
min_width	Minimum acceptable AOI width.
min_height	Minimum acceptable AOI height.
min_area	Minimum acceptable AOI area.
max_area_prop	Maximum acceptable AOI area as a proportion of screen area.
require_within_screen	Logical. If TRUE, AOIs extending outside the screen range are flagged.

Value

A list with class `gp3_aoi_geometry_audit` containing `overview`, `geometry_summary`, `size_summary`, `flagged_aois`, `duplicate_geometry`, and `settings` tables.

audit_gazepoint_aoi_margin_sensitivity
Audit AOI margin sensitivity

Description

Audit whether sample-level AOI assignments are sensitive to small expansions or shrinkages of AOI boundaries.

Usage

```
audit_gazepoint_aoi_margin_sensitivity(  
  gaze_data,  
  aoi_geometry,  
  gaze_x_col = NULL,  
  gaze_y_col = NULL,  
  gaze_stimulus_col = NULL,  
  sample_id_cols = NULL,  
  geometry_aoi_col = NULL,  
  geometry_stimulus_col = NULL,  
  x_min_col = NULL,  
  y_min_col = NULL,  
  x_max_col = NULL,  
  y_max_col = NULL,  
  x_col = NULL,  
  y_col = NULL,  
  width_col = NULL,  
  height_col = NULL,  
  margins = c(-0.02, 0, 0.02),  
  screen_x_range = c(0, 1),  
  screen_y_range = c(0, 1),  
  tie_method = c("ambiguous", "first"),  
  outside_label = "outside",  
  ambiguous_label = "ambiguous",  
  missing_label = "missing_coordinate",  
  max_margin_change_prop = 0.1,  
  max_ambiguous_prop = 0.05,  
  ignore_invalid_geometry = TRUE  
)
```

Arguments

gaze_data	A data frame containing gaze samples.
aoi_geometry	A data frame containing AOI geometry definitions.
gaze_x_col	Gaze x-coordinate column. If NULL, common aliases are detected automatically.
gaze_y_col	Gaze y-coordinate column. If NULL, common aliases are detected automatically.

gaze_stimulus_col	Optional gaze stimulus/media column.
sample_id_cols	Optional columns to carry into the sample-level audit table.
geometry_aoi_col	AOI label/name column in aoi_geometry.
geometry_stimulus_col	Optional stimulus/media column in aoi_geometry.
x_min_col	Optional AOI left/x-min column.
y_min_col	Optional AOI top/y-min column.
x_max_col	Optional AOI right/x-max column.
y_max_col	Optional AOI bottom/y-max column.
x_col	Optional AOI left/x column used with width_col.
y_col	Optional AOI top/y column used with height_col.
width_col	Optional AOI width column.
height_col	Optional AOI height column.
margins	Numeric vector of AOI boundary margins. Positive values expand AOIs; negative values shrink AOIs. A zero-margin baseline is always added.
screen_x_range	Numeric length-2 vector defining the screen x range.
screen_y_range	Numeric length-2 vector defining the screen y range.
tie_method	How to handle samples falling in multiple AOIs. Use "ambiguous" to label them as ambiguous or "first" to use the first matching AOI.
outside_label	Label for samples outside all AOIs.
ambiguous_label	Label for samples inside multiple AOIs when tie_method = "ambiguous".
missing_label	Label for samples with missing/non-finite gaze coordinates.
max_margin_change_prop	Maximum acceptable proportion of samples whose AOI assignment changes from the zero-margin baseline.
max_ambiguous_prop	Maximum acceptable proportion of ambiguous samples.
ignore_invalid_geometry	Logical. If TRUE, AOIs with invalid coordinates or dimensions are excluded before margin coding.

Value

A list with class `gp3_aoi_margin_sensitivity_audit` containing `overview`, `geometry_summary`, `sample_sensitivity`, `margin_summary`, `aoi_margin_summary`, `flagged_samples`, and `settings` tables.

```
audit_gazepoint_aoi_overlap
      Audit AOI overlap
```

Description

Create a publication-level audit of pairwise AOI overlap within each stimulus.

Usage

```
audit_gazepoint_aoi_overlap(
  data,
  aoi_col = NULL,
  stimulus_col = NULL,
  x_min_col = NULL,
  y_min_col = NULL,
  x_max_col = NULL,
  y_max_col = NULL,
  x_col = NULL,
  y_col = NULL,
  width_col = NULL,
  height_col = NULL,
  screen_x_range = c(0, 1),
  screen_y_range = c(0, 1),
  min_overlap_area = 0,
  min_overlap_prop = 0,
  ignore_invalid_geometry = TRUE
)
```

Arguments

<code>data</code>	A data frame containing AOI geometry definitions.
<code>aoi_col</code>	AOI label/name column. If NULL, common AOI-name aliases are detected automatically by <code>audit_gazepoint_aoi_geometry()</code> .
<code>stimulus_col</code>	Optional stimulus/media column.
<code>x_min_col</code>	Optional AOI left/x-min column.
<code>y_min_col</code>	Optional AOI top/y-min column.
<code>x_max_col</code>	Optional AOI right/x-max column.
<code>y_max_col</code>	Optional AOI bottom/y-max column.
<code>x_col</code>	Optional AOI left/x column used with <code>width_col</code> .
<code>y_col</code>	Optional AOI top/y column used with <code>height_col</code> .
<code>width_col</code>	Optional AOI width column.
<code>height_col</code>	Optional AOI height column.
<code>screen_x_range</code>	Numeric length-2 vector defining the screen x range.

screen_y_range Numeric length-2 vector defining the screen y range.
 min_overlap_area Minimum overlap area above which an AOI pair is flagged.
 min_overlap_prop Minimum overlap proportion above which an AOI pair is flagged. This is computed relative to the smaller AOI in the pair.
 ignore_invalid_geometry Logical. If TRUE, AOIs with invalid geometry are excluded from pairwise overlap calculations.

Value

A list with class `gp3_aoi_overlap_audit` containing `overview`, `geometry_summary`, `pairwise_overlap`, `overlap_summary`, `flagged_overlaps`, and `settings` tables.

audit_gazepoint_aoi_window_denominators
Audit AOI window denominators before binomial modelling

Description

Audit AOI-window sample denominators before confirmatory binomial or logistic mixed-effects modelling. The function is designed for output from `summarise_gazepoint_aoi_windows()`.

Usage

```

audit_gazepoint_aoi_window_denominators(
  data,
  window_col = "window_label",
  window_start_col = "window_start_ms",
  window_end_col = "window_end_ms",
  denominator_col = "n_valid_denominator_samples",
  total_col = "n_window_samples",
  target_col = "n_target_samples",
  condition_col = "condition",
  group_cols = NULL,
  min_denominator_samples = 5,
  min_valid_denominator_prop = 0.7,
  max_denominator_cv = 0.25,
  max_condition_ratio = 2
)

```

Arguments

data AOI-window summary data.
 window_col Name of the AOI-window label column.

window_start_col	Optional window-start column.
window_end_col	Optional window-end column.
denominator_col	Name of the denominator column to audit.
total_col	Name of the total window-sample column.
target_col	Name of the target-success count column.
condition_col	Optional condition column.
group_cols	Optional grouping columns for row-level audit context.
min_denominator_samples	Minimum acceptable denominator count.
min_valid_denominator_prop	Minimum acceptable valid-denominator proportion relative to total window samples.
max_denominator_cv	Maximum acceptable denominator coefficient of variation within each window.
max_condition_ratio	Maximum acceptable ratio between the largest and smallest mean denominator across conditions within a window.

Value

A named list containing overview, row audit, window summary, condition-window summary, denominator-imbalance summary, flagged rows, and settings.

```
audit_gazepoint_condition_quality_imbalance
      Audit condition-level quality imbalance
```

Description

Create a publication-level audit of whether gaze, pupil, retention, or other quality metrics differ across experimental conditions.

Usage

```
audit_gazepoint_condition_quality_imbalance(
  data,
  condition_col = "condition",
  quality_cols = NULL,
  subject_col = NULL,
  min_units_per_condition = 1L,
  max_mean_difference = 0.1,
  max_condition_ratio = 2,
  lower_is_better = c("missing_gaze_prop", "offscreen_prop", "excluded_prop",
    "failure_prop", "artifact_prop")
)
```

Arguments

<code>data</code>	A data frame containing condition-level, unit-level, or subject-condition-level quality metrics.
<code>condition_col</code>	Condition column.
<code>quality_cols</code>	Numeric quality-metric columns. If NULL, common quality columns are detected automatically.
<code>subject_col</code>	Optional subject column.
<code>min_units_per_condition</code>	Minimum number of rows/units expected per condition.
<code>max_mean_difference</code>	Maximum acceptable absolute difference between condition means for each quality metric.
<code>max_condition_ratio</code>	Maximum acceptable ratio between the largest and smallest non-zero condition mean for each quality metric.
<code>lower_is_better</code>	Optional character vector naming metrics where lower values indicate better quality, such as missing-gaze or exclusion metrics.

Value

A list with class `gp3_condition_quality_imbalance_audit` containing overview, `condition_summary`, `metric_summary`, `flagged_metrics`, and settings tables.

`audit_gazepoint_design_balance`

Audit Gazepoint experimental design balance

Description

Create a publication-level audit of observed design balance across subjects, conditions, and optional stimulus/trial identifiers.

Usage

```
audit_gazepoint_design_balance(
  data,
  subject_col = "subject",
  condition_col = "condition",
  unit_cols = c("media_id", "trial_global"),
  expected_conditions = NULL,
  min_units_per_condition = 1L,
  max_condition_ratio = 2,
  require_all_conditions_per_subject = TRUE
)
```

Arguments

data	A data frame containing trial-level, window-level, or sample-level Gazepoint-derived data.
subject_col	Subject/participant identifier column.
condition_col	Experimental condition column.
unit_cols	Optional columns defining the repeated unit to count within each subject and condition, such as media, trial, block, or window.
expected_conditions	Optional character vector of expected condition labels.
min_units_per_condition	Minimum number of observed units expected per subject-condition cell.
max_condition_ratio	Maximum allowed ratio between a subject's largest and smallest non-zero condition counts.
require_all_conditions_per_subject	Logical. If TRUE, flag subjects who do not have all expected or observed conditions.

Value

A list with class `gp3_design_balance_audit` containing overview, subject_summary, condition_summary, cell_summary, imbalance_summary, flagged_cells, and settings tables.

audit_gazepoint_event_sync

Audit Gazepoint event and timing synchronisation

Description

Create a publication-level audit of event timing, trial timing, and event availability in a Gazepoint master table or sample-level export.

Usage

```
audit_gazepoint_event_sync(
  data,
  time_col = "time",
  event_col = NULL,
  group_cols = c("subject", "media_id", "trial_global"),
  condition_col = NULL,
  expected_event_labels = NULL,
  onset_event_label = NULL,
  response_event_label = NULL,
  min_samples_per_unit = 1L,
  max_time_gap_ms = NULL
)
```

Arguments

data	A data frame containing sample-level Gazepoint data.
time_col	Name of the time column.
event_col	Optional event-label column. If NULL, the function tries to detect a common event column.
group_cols	Columns defining a trial or recording unit.
condition_col	Optional condition column.
expected_event_labels	Optional character vector of expected event labels.
onset_event_label	Optional event label identifying trial/stimulus onset.
response_event_label	Optional event label identifying response events.
min_samples_per_unit	Minimum number of samples expected per unit.
max_time_gap_ms	Optional maximum allowed within-unit time gap in milliseconds.

Value

A list with class `gp3_event_sync_audit` containing `overview`, `unit_summary`, `event_summary`, `expected_event_summary`, `flagged_units`, and `settings` tables.

audit_gazepoint_exclusion_flow

Audit Gazepoint exclusion and retention flow

Description

Create a publication-level audit of retained and excluded analysis units.

Usage

```
audit_gazepoint_exclusion_flow(
  data,
  subject_col = "subject",
  condition_col = NULL,
  unit_cols = c("media_id", "trial_global"),
  include_col = NULL,
  exclude_col = NULL,
  status_col = NULL,
  reason_col = NULL,
  included_values = c("included", "include", "kept", "keep", "retained", "ok", "ready",
    "complete", "completed"),
```

```

excluded_values = c("excluded", "exclude", "drop", "dropped", "removed", "fail",
  "failed", "not_ready", "review", "invalid"),
min_retained_prop = 0.7,
max_condition_exclusion_ratio = 2
)

```

Arguments

data	A data frame containing row-, trial-, window-, or unit-level data.
subject_col	Subject/participant identifier column.
condition_col	Optional condition column.
unit_cols	Optional columns defining the analysis unit, such as media, trial, block, or window.
include_col	Optional logical/numeric/character column indicating rows or units retained for analysis.
exclude_col	Optional logical/numeric/character column indicating rows or units excluded from analysis.
status_col	Optional status column used to infer inclusion or exclusion.
reason_col	Optional exclusion-reason column.
included_values	Character values in status_col treated as retained.
excluded_values	Character values in status_col treated as excluded.
min_retained_prop	Minimum acceptable retained-unit proportion.
max_condition_exclusion_ratio	Maximum allowed ratio between condition exclusion proportions.

Value

A list with class `gp3_exclusion_flow_audit` containing `overview`, `unit_flow`, `reason_summary`, `condition_summary`, `subject_summary`, `flagged_units`, and `settings` tables.

audit_gazepoint_gaze_signal_quality
Audit Gazepoint gaze-signal quality

Description

Create a publication-level audit of gaze coordinate availability, validity, off-screen samples, and optional pupil availability.

Usage

```
audit_gazepoint_gaze_signal_quality(
  data,
  subject_col = "subject",
  condition_col = NULL,
  group_cols = c("subject", "media_id", "trial_global"),
  x_col = NULL,
  y_col = NULL,
  validity_cols = NULL,
  pupil_col = NULL,
  screen_x_range = c(0, 1),
  screen_y_range = c(0, 1),
  min_gaze_valid_prop = 0.7,
  max_missing_gaze_prop = 0.3,
  max_offscreen_prop = 0.3,
  min_pupil_valid_prop = 0.7
)
```

Arguments

<code>data</code>	A sample-level Gazepoint data frame.
<code>subject_col</code>	Subject/participant identifier column.
<code>condition_col</code>	Optional condition column.
<code>group_cols</code>	Columns defining a recording, stimulus, trial, or analysis unit.
<code>x_col</code>	Optional gaze-x coordinate column. If NULL, common Gazepoint aliases are detected.
<code>y_col</code>	Optional gaze-y coordinate column. If NULL, common Gazepoint aliases are detected.
<code>validity_cols</code>	Optional gaze-validity columns. If NULL, common Gazepoint validity columns are detected.
<code>pupil_col</code>	Optional pupil column. If NULL, common pupil aliases are detected.
<code>screen_x_range</code>	Numeric length-2 vector defining plausible on-screen x range.
<code>screen_y_range</code>	Numeric length-2 vector defining plausible on-screen y range.
<code>min_gaze_valid_prop</code>	Minimum acceptable gaze-validity proportion.
<code>max_missing_gaze_prop</code>	Maximum acceptable missing-gaze proportion.
<code>max_offscreen_prop</code>	Maximum acceptable off-screen proportion.
<code>min_pupil_valid_prop</code>	Minimum acceptable valid-pupil proportion when a pupil column is available.

Value

A list with class `gp3_gaze_signal_quality_audit` containing overview, `unit_summary`, `subject_summary`, `condition_summary`, `signal_issue_summary`, `flagged_units`, and settings tables.

audit_gazepoint_master

Audit a Gazepoint master sample table

Description

Creates compact quality-audit tables from a master sample-level table created by `as_gazepoint_master()` or `create_gazepoint_master()`. The audit summarises missing gaze, missing pupil, off-screen gaze, AOI states, pupil availability, coordinate ranges, and subject/media-level quality.

Usage

```
audit_gazepoint_master(master)
```

Arguments

`master` A master sample-level table created by `as_gazepoint_master()` or `create_gazepoint_master()`.

Value

A named list of tibbles:

overview One-row overview of the master table.

by_subject Quality summary by participant/source.

by_media Quality summary by media/stimulus.

by_subject_media Quality summary by participant/source and media/stimulus.

aoi_states Counts and percentages of AOI states.

pupil_summary Pupil summary by participant/source and media/stimulus.

coordinate_summary Coordinate range and off-screen summary.

Examples

```
master <- gazepoint_example_master
results <- run_gazepoint_workflow(
  export_dir = system.file("extdata", "gazepoint_realistic_demo_exports", package = "gp3tools"),
  output_dir = file.path(tempdir(), "gp3_outputs")
)
```

```
master <- create_gazepoint_master(
  gaze_data = gazepoint_example_master,
  screen_width_px = 1920,
  screen_height_px = 1080
)
```

```
audit <- audit_gazepoint_master(master)
```

```
audit$overview
```

```
audit$by_subject
audit$aoi_states
```

```
audit_gazepoint_post_exclusion_balance
      Audit post-exclusion condition balance
```

Description

Create a publication-level audit of whether the retained analysis sample remains balanced across subjects and experimental conditions after exclusions.

Usage

```
audit_gazepoint_post_exclusion_balance(
  data,
  subject_col = "subject",
  condition_col = "condition",
  unit_cols = c("media_id", "trial_global"),
  retained_col = NULL,
  include_col = NULL,
  exclude_col = NULL,
  status_col = NULL,
  expected_conditions = NULL,
  included_values = c("included", "include", "kept", "keep", "retained", "ok", "ready",
    "complete", "completed"),
  excluded_values = c("excluded", "exclude", "drop", "dropped", "removed", "fail",
    "failed", "not_ready", "review", "invalid"),
  min_retained_units_per_condition = 1L,
  min_retained_units_per_subject_condition = 1L,
  max_condition_count_ratio = 2,
  max_subject_condition_ratio = 2,
  require_all_conditions_per_subject = TRUE
)
```

Arguments

<code>data</code>	A data frame containing row-, sample-, trial-, or unit-level data.
<code>subject_col</code>	Subject/participant identifier column.
<code>condition_col</code>	Experimental condition column.
<code>unit_cols</code>	Optional columns defining the analysis unit, such as media, trial, block, or window.
<code>retained_col</code>	Optional logical/numeric/character column indicating retained units.
<code>include_col</code>	Optional logical/numeric/character inclusion column.

exclude_col	Optional logical/numeric/character exclusion column.
status_col	Optional status column used to infer retained/excluded units.
expected_conditions	Optional character vector of expected conditions.
included_values	Character values in status_col treated as retained.
excluded_values	Character values in status_col treated as excluded.
min_retained_units_per_condition	Minimum retained units required per condition.
min_retained_units_per_subject_condition	Minimum retained units required per subject-condition cell.
max_condition_count_ratio	Maximum allowed ratio between condition-level retained counts.
max_subject_condition_ratio	Maximum allowed within-subject retained condition-count ratio.
require_all_conditions_per_subject	Logical. If TRUE, flag subjects missing retained units in one or more expected conditions.

Value

A list with class `gp3_post_exclusion_balance_audit` containing `overview`, `unit_flow`, `cell_summary`, `condition_summary`, `subject_summary`, `flagged_cells`, `flagged_subjects`, and settings tables.

audit_gazepoint_pupil_baseline
Audit Gazepoint pupil baseline quality

Description

Summarise baseline quality after `baseline_correct_gazepoint_pupil()`.

Usage

```
audit_gazepoint_pupil_baseline(
  data,
  group_cols = c("subject", "media_id"),
  time_col = "time",
  pupil_col = "pupil_interpolated",
  baseline_n_col = "pupil_baseline_n",
  baseline_status_col = "pupil_baseline_status",
  baseline_available_col = "pupil_baseline_available",
  baseline_used_col = "pupil_baseline_used",
  baseline_window_start_col = "pupil_baseline_window_start",
  baseline_window_end_col = "pupil_baseline_window_end",
```

```

baseline_flag_col = NULL,
interpolated_col = "pupil_was_interpolated",
artifact_col = NULL,
artifact_reason_col = NULL,
min_baseline_samples = 1,
max_missing_pct = 50,
max_interpolated_pct = 50,
max_artifact_pct = 50
)

```

Arguments

<code>data</code>	A data frame returned by <code>baseline_correct_gazepoint_pupil()</code> or a later pupil-preprocessing step.
<code>group_cols</code>	Character vector of grouping columns. Use <code>character(0)</code> for an overall audit.
<code>time_col</code>	Name of the time column.
<code>pupil_col</code>	Name of the pupil column used to evaluate missingness.
<code>baseline_n_col</code>	Name of the baseline valid-sample count column.
<code>baseline_status_col</code>	Name of the baseline-status column.
<code>baseline_available_col</code>	Name of the baseline-availability column.
<code>baseline_used_col</code>	Name of the logical column indicating whether a row used a baseline value.
<code>baseline_window_start_col</code>	Name of the baseline-window start column.
<code>baseline_window_end_col</code>	Name of the baseline-window end column.
<code>baseline_flag_col</code>	Optional logical column identifying baseline rows. If <code>NULL</code> , baseline rows are detected from the time column and baseline window start/end columns.
<code>interpolated_col</code>	Name of the logical interpolation flag column.
<code>artifact_col</code>	Optional artifact flag column. If <code>NULL</code> , the function tries to detect <code>pupil_artifact_flag</code> , <code>pupil_flag_invalid</code> , or <code>artifact_flag</code> .
<code>artifact_reason_col</code>	Optional artifact-reason column. If <code>NULL</code> , the function tries to detect <code>pupil_artifact_reason</code> , <code>pupil_flag_reason</code> , or <code>artifact_reason</code> .
<code>min_baseline_samples</code>	Minimum acceptable number of valid baseline samples before a group is flagged as low quality.
<code>max_missing_pct</code>	Maximum acceptable percentage of missing baseline samples.
<code>max_interpolated_pct</code>	Maximum acceptable percentage of interpolated baseline samples.
<code>max_artifact_pct</code>	Maximum acceptable percentage of artifact-flagged baseline samples.

Details

The function reports baseline-row counts, valid/missing baseline samples, interpolated baseline samples, artifact-flagged baseline samples, no-baseline cases, and low-quality baseline flags by subject, media, trial, condition, or any selected grouping variables.

Value

A tibble with one row per group.

```
audit_gazepoint_pupil_drift
```

Audit Gazepoint pupil drift

Description

Summarise tonic pupil/time-on-task drift in processed Gazepoint pupil data.

Usage

```
audit_gazepoint_pupil_drift(
  data,
  group_cols = "subject",
  pupil_col = NULL,
  time_col = "time",
  order_col = "trial",
  condition_col = "condition",
  exclude_col = "excluded_trial",
  include_excluded = FALSE,
  min_valid_samples = 3,
  max_abs_slope_per_min = 1,
  max_condition_time_mean_diff_ms = 1000,
  max_condition_order_mean_diff = 1
)
```

Arguments

<code>data</code>	A data frame from a Gazepoint pupil preprocessing pipeline.
<code>group_cols</code>	Character vector of grouping columns for the main drift audit. The default is "subject".
<code>pupil_col</code>	Name of the pupil column to analyse. If NULL, the function automatically tries <code>pupil_smoothed</code> , <code>pupil_baseline_corrected</code> , <code>pupil_interpolated</code> , <code>pupil_clean</code> , and <code>pupil</code> .
<code>time_col</code>	Name of the within-trial or sample-time column.
<code>order_col</code>	Optional trial/order column used to assess time-on-task imbalance. If NULL, order-based summaries are skipped.

condition_col	Optional condition column used to summarise condition drift and time-on-task imbalance. If NULL, condition summaries are skipped.
exclude_col	Optional logical exclusion column. If present and include_excluded = FALSE, excluded rows are removed before analysis.
include_excluded	Logical. If FALSE, rows marked by exclude_col are excluded when that column exists.
min_valid_samples	Minimum valid pupil samples required to estimate a drift slope.
max_abs_slope_per_min	Maximum acceptable absolute pupil slope per minute before a drift warning is raised.
max_condition_time_mean_diff_ms	Maximum acceptable difference in mean sample time across conditions.
max_condition_order_mean_diff	Maximum acceptable difference in mean trial/order value across conditions.

Details

The function estimates simple linear pupil trends over time within selected grouping variables, usually subjects. It also reports subject-level drift, condition-level drift, and possible condition imbalance in time-on-task.

Value

A named list containing by_group, by_subject, by_condition, condition_balance, and summary tibbles.

```
audit_gazepoint_pupil_gaps
```

Audit Gazepoint pupil interpolation gaps

Description

Summarise observed, interpolated, and remaining missing pupil samples, together with gap-level counts and gap duration/sample-size summaries.

Usage

```
audit_gazepoint_pupil_gaps(
  data,
  group_cols = c("subject", "media_id"),
  status_col = "pupil_interpolation_status",
  gap_id_col = "pupil_gap_id",
  gap_n_samples_col = "pupil_gap_n_samples",
  gap_duration_col = "pupil_gap_duration_ms",
  interpolated_col = "pupil_was_interpolated",
  pupil_col = "pupil_interpolated"
)
```

Arguments

<code>data</code>	A data frame containing pupil interpolation status columns.
<code>group_cols</code>	Character vector of grouping columns. Use <code>character(0)</code> for an overall audit.
<code>status_col</code>	Name of the interpolation status column.
<code>gap_id_col</code>	Name of the gap identifier column.
<code>gap_n_samples_col</code>	Name of the column containing gap size in samples.
<code>gap_duration_col</code>	Name of the column containing gap duration in ms.
<code>interpolated_col</code>	Name of the logical column indicating whether a sample was interpolated.
<code>pupil_col</code>	Name of the pupil column after interpolation.

Details

This function is intended for data returned by `interpolate_gazepoint_pupil()`, but it can also be used with any table that contains compatible interpolation-status and gap columns.

Value

A tibble with one row per group, or one row overall when `group_cols = character(0)`.

`audit_gazepoint_pupil_imbalance`
Audit Gazepoint pupil preprocessing imbalance

Description

Check whether pupil preprocessing loss differs across experimental conditions or other grouping variables.

Usage

```
audit_gazepoint_pupil_imbalance(
  data,
  group_cols = "condition",
  pupil_col = "pupil_interpolated",
  interpolated_col = "pupil_was_interpolated",
  interpolation_status_col = "pupil_interpolation_status",
  artifact_col = NULL,
  artifact_reason_col = NULL,
  min_group_n = 1,
  max_valid_pct_diff = 10,
  max_artifact_pct_diff = 10,
  max_missing_pct_diff = 10,
  max_interpolated_pct_diff = 10
)
```

Arguments

data	A data frame from a pupil preprocessing pipeline.
group_cols	Character vector of grouping columns. By default, summaries are produced by condition.
pupil_col	Name of the post-preprocessing pupil column used to define remaining valid and missing samples.
interpolated_col	Name of the logical interpolation flag column.
interpolation_status_col	Name of the interpolation-status column.
artifact_col	Optional artifact flag column. If NULL, the function tries to detect pupil_artifact_flag, pupil_flag_invalid, or artifact_flag.
artifact_reason_col	Optional artifact-reason column. If NULL, the function tries to detect pupil_artifact_reason, pupil_flag_reason, or artifact_reason.
min_group_n	Minimum group size below which a group is flagged.
max_valid_pct_diff	Maximum acceptable range in valid-sample percentage across groups.
max_artifact_pct_diff	Maximum acceptable range in artifact percentage across groups.
max_missing_pct_diff	Maximum acceptable range in remaining-missing percentage across groups.
max_interpolated_pct_diff	Maximum acceptable range in interpolated percentage across groups.

Details

The function summarises valid pupil samples, interpolated samples, artifact-flagged samples, and remaining missing samples. It also adds simple imbalance flags based on differences between groups.

Value

A tibble with one row per group and imbalance-warning columns.

audit_gazepoint_pupil_overlap_risk

Audit Gazepoint pupil-response overlap risk

Description

Check whether event-related pupil-response windows may overlap.

Usage

```
audit_gazepoint_pupil_overlap_risk(
  data,
  group_cols = "subject",
  trial_col = "trial_global",
  time_col = "time",
  event_time_cols = c("stimulus_onset_time", "target_onset_time", "response_time"),
  window_start_ms = 0,
  window_end_ms = 2000,
  min_event_gap_ms = 1000,
  exclude_col = "excluded_trial",
  include_excluded = FALSE
)
```

Arguments

<code>data</code>	A Gazeport sample-level data frame.
<code>group_cols</code>	Character vector of grouping columns, usually "subject".
<code>trial_col</code>	Name of the trial identifier column.
<code>time_col</code>	Name of the within-trial time column.
<code>event_time_cols</code>	Character vector of event-time columns, in ms.
<code>window_start_ms</code>	Response-window start relative to each event, in ms.
<code>window_end_ms</code>	Response-window end relative to each event, in ms.
<code>min_event_gap_ms</code>	Minimum acceptable event-to-event gap in ms.
<code>exclude_col</code>	Optional logical exclusion column.
<code>include_excluded</code>	Logical. If FALSE, rows marked by <code>exclude_col</code> are removed before the audit when that column exists.

Details

This function is designed as a deconvolution/readiness gate. It checks whether events within the same trial are too close together and whether their response windows overlap. If no usable event-time values are found, the function returns a clean audit with `overlap_assessment_status = "no_usable_event_times"`.

Value

A named list containing events, event_gaps, by_trial, and summary tibbles.

 audit_gazepoint_pupil_reliability

Audit split-half reliability for Gazepoint pupil outcomes

Description

Create a split-half reliability audit for trial-level or window-level pupil outcomes. The helper is intended for publication-readiness checks when pupil features are interpreted as stable participant-level outcomes or individual difference measures.

Usage

```
audit_gazepoint_pupil_reliability(
  data,
  outcome_cols = NULL,
  participant_col = NULL,
  trial_col = NULL,
  split_col = NULL,
  by_cols = NULL,
  split_method = c("odd_even", "first_second"),
  aggregate_function = c("mean", "median"),
  correlation_method = c("pearson", "spearman"),
  min_trials_per_split = 2,
  name = "gazepoint_pupil_reliability"
)
```

Arguments

data	A data frame containing trial-level or window-level pupil outcomes.
outcome_cols	Character vector of pupil outcome columns. If NULL, common pupil outcome columns are detected automatically.
participant_col	Participant/subject column. If NULL, common participant columns are detected automatically.
trial_col	Trial/order column. If NULL, common trial columns are detected automatically when available. If no trial column is available, row order within participant is used.
split_col	Optional pre-existing split column. If supplied, it must have exactly two non-missing levels.
by_cols	Optional grouping columns for separate reliability audits, such as "condition" or "window".
split_method	Split method used when split_col = NULL. Options are "odd_even" and "first_second".
aggregate_function	Function used to aggregate trial-level values within participant and split. Options are "mean" and "median".

correlation_method	Correlation method for split-half association. Options are "pearson" and "spearman".
min_trials_per_split	Minimum number of non-missing outcome values required in each split for a participant to contribute to the reliability estimate.
name	Character label stored in the audit object.

Value

A list with class gp3_pupil_reliability_audit.

audit_gazepoint_stimulus_luminance
Audit stimulus luminance and brightness for Gazepoint studies

Description

Compute stimulus-level luminance, brightness, and contrast summaries for image stimuli used in Gazepoint eye-tracking and pupillometry studies. This helper is intended as a publication-readiness audit because pupil size is strongly affected by stimulus brightness.

Usage

```
audit_gazepoint_stimulus_luminance(
  data,
  stimulus_file_col = NULL,
  stimulus_id_col = NULL,
  condition_col = NULL,
  image_dir = NULL,
  recursive = TRUE,
  name = "gazepoint_stimulus_luminance"
)
```

Arguments

data	A data frame containing at least a stimulus image/file column.
stimulus_file_col	Name of the stimulus image/file column. If NULL, common file-column names are detected automatically.
stimulus_id_col	Optional stimulus identifier column. If NULL, common stimulus/media/item identifier columns are detected automatically.
condition_col	Optional experimental condition column. If NULL, common condition columns are detected automatically. If no condition column exists, all rows are assigned to "all_data".
image_dir	Optional directory prepended to relative stimulus paths.
recursive	If TRUE, unresolved relative file names are searched for recursively under image_dir.
name	Character label stored in the audit object.

Value

A list with class `gp3_stimulus_luminance_audit`.

baseline_correct_gazepoint_pupil

Baseline-correct Gazepoint pupil data

Description

Computes baseline-corrected pupil columns from a Gazepoint pupil time series, typically after [flag_gazepoint_pupil\(\)](#) and [interpolate_gazepoint_pupil\(\)](#). Baselines can be defined either by a time window, such as `c(-200, 0)`, or by a user-supplied logical baseline/pre-stimulus flag column.

Usage

```
baseline_correct_gazepoint_pupil(
  data,
  pupil_col = NULL,
  time_col = NULL,
  baseline_time_col = NULL,
  baseline_window = c(-200, 0),
  baseline_flag_col = NULL,
  group_cols = c("subject", "media_id"),
  baseline_method = c("mean", "median"),
  min_baseline_samples = 1
)
```

Arguments

<code>data</code>	A Gazepoint master table, preferably after interpolate_gazepoint_pupil() .
<code>pupil_col</code>	Optional name of the pupil column to baseline-correct. If NULL, the function detects one of <code>pupil_interpolated</code> , <code>pupil_for_preprocessing</code> , <code>mean_pupil</code> , <code>pupil</code> , <code>pupil_raw</code> , <code>left_pupil</code> , or <code>right_pupil</code> .
<code>time_col</code>	Optional name of the main time column. If NULL, the function detects one of <code>time_ms</code> , <code>time</code> , <code>time_orig</code> , or <code>time_orig_ms</code> .
<code>baseline_time_col</code>	Optional name of the time column used for selecting baseline samples. If NULL, the function detects relative-time columns first, then falls back to <code>time_col</code> .
<code>baseline_window</code>	Numeric vector of length two giving the baseline window in milliseconds. Defaults to <code>c(-200, 0)</code> . This can also be set to post-onset or early-window values such as <code>c(0, 200)</code> when no pre-stimulus period is available.
<code>baseline_flag_col</code>	Optional logical column identifying baseline samples. If supplied, this takes priority over <code>baseline_window</code> .

`group_cols` Character vector of grouping columns used to compute one baseline per independent time series. Defaults to `c("subject", "media_id")`. Columns such as `"trial"` or `"trial_global"` can be added when available. Use `character(0)` for one global baseline.

`baseline_method` Baseline statistic. One of `"mean"` or `"median"`. Defaults to `"mean"`.

`min_baseline_samples` Minimum number of valid baseline samples required to compute a baseline. Defaults to 1.

Value

A tibble containing the original data plus baseline-correction columns.

Examples

```

master <- gazepoint_example_master
flagged <- flag_gazepoint_pupil(master)
interpolated <- interpolate_gazepoint_pupil(flagged)

corrected <- baseline_correct_gazepoint_pupil(
  interpolated,
  baseline_window = c(-200, 0)
)

corrected <- baseline_correct_gazepoint_pupil(
  interpolated,
  baseline_window = c(0, 200)
)

```

`check_gazepoint_file_pairs`

Check Gazepoint all-gaze and fixation file pairs

Description

Checks whether each Gazepoint participant/source has both an all-gaze CSV file and a fixation CSV file before running the full workflow.

Usage

```

check_gazepoint_file_pairs(
  folder,
  all_gaze_pattern = "_all_gaze\\.csv$",
  fixation_pattern = "_fixations\\.csv$",
  recursive = FALSE
)

```

Arguments

folder	Folder containing Gazepoint CSV export files.
all_gaze_pattern	Regular expression for selecting all-gaze files.
fixation_pattern	Regular expression for selecting fixation files.
recursive	Logical. If TRUE, search subfolders recursively.

Value

A tibble with one row per detected participant/source and file-pair status.

```
check_gazepoint_model_convergence
  Check model convergence
```

Description

Check convergence status for fitted models used in gp3tools workflows.

Usage

```
check_gazepoint_model_convergence(model, model_name = NULL)
```

Arguments

model	A fitted model object, or a gp3tools fit object containing a \$model element.
model_name	Optional model label used in the returned table.

Details

This helper supports lme4 mixed models, mgcv GAM/GAMM objects, glm objects, and lm objects where convergence is meaningful. It returns a compact diagnostic table instead of printing model-specific messages.

Value

A tibble with model class, convergence status, diagnostic status, and message.

```
check_gazepoint_model_overdispersion
      Check model overdispersion
```

Description

Compute a Pearson-residual overdispersion diagnostic for models where this is meaningful, especially binomial and count models.

Usage

```
check_gazepoint_model_overdispersion(
  model,
  ratio_threshold = 1.2,
  model_name = NULL
)
```

Arguments

`model` A fitted model object, or a `gp3tools` fit object containing a `$model` element.

`ratio_threshold` Numeric threshold above which the model is flagged as overdispersed.

`model_name` Optional model label used in the returned table.

Details

This helper supports `glm`, `lme4` GLMMs, and `mgcv` GAM/GAMM objects when their family is binomial, quasibinomial, Poisson, quasipoisson, or negative-binomial-like. Gaussian `lm`, `lmer`, and Gaussian GAM models return a structured `not_applicable` diagnostic row.

Value

A tibble with Pearson chi-square, residual degrees of freedom, dispersion ratio, overdispersion flag, diagnostic status, and message.

```
check_gazepoint_model_singularity
      Check model singularity
```

Description

Check whether a fitted mixed model has a singular random-effects structure.

Usage

```
check_gazepoint_model_singularity(model, tolerance = 1e-04, model_name = NULL)
```

Arguments

model	A fitted model object, or a gp3tools fit object containing a \$model element.
tolerance	Numeric tolerance passed to lme4::isSingular().
model_name	Optional model label used in the returned table.

Details

This helper is primarily intended for lme4 mixed models. For model classes where singularity is not meaningful, such as lm, glm, and mgcv GAM objects, it returns a structured not_applicable diagnostic row.

Value

A tibble with model class, singular-fit status, diagnostic status, and message.

check_gazepoint_real_data_readiness

Check real-data readiness before Gazepoint analysis

Description

Create an explicit final readiness gate for real Gazepoint or gp3tools master data before confirmatory analysis. The gate checks required identifiers, trial/time structure, analysis-specific columns, missingness, basic design balance, and optional upstream audit objects.

Usage

```
check_gazepoint_real_data_readiness(
  data,
  analysis_type = c("general", "pupil", "aoi", "combined"),
  participant_col = NULL,
  trial_col = NULL,
  time_col = NULL,
  condition_col = NULL,
  stimulus_col = NULL,
  aoi_col = NULL,
  pupil_col = NULL,
  gaze_x_col = NULL,
  gaze_y_col = NULL,
  tracking_valid_col = NULL,
  required_cols = NULL,
  audit_objects = NULL,
  min_rows = 1L,
  min_participants = 1L,
  min_trials = 1L,
  max_missing_pupil_prop = 0.4,
```

```

max_missing_gaze_prop = 0.4,
max_condition_imbalance_ratio = 3,
name = "gazepoint_real_data_readiness_gate"
)

```

Arguments

<code>data</code>	A Gazepoint or gp3tools data frame.
<code>analysis_type</code>	Analysis target. Options are "general", "pupil", "aoi", and "combined".
<code>participant_col</code>	Optional participant column. If NULL, common names are detected.
<code>trial_col</code>	Optional trial column. If NULL, common names are detected.
<code>time_col</code>	Optional time column. If NULL, common names are detected.
<code>condition_col</code>	Optional condition/group column. If NULL, common names are detected when present.
<code>stimulus_col</code>	Optional stimulus/media column. If NULL, common names are detected when present.
<code>aoi_col</code>	Optional AOI/state column. If NULL, common names are detected when present.
<code>pupil_col</code>	Optional pupil column. If NULL, common names are detected when present.
<code>gaze_x_col</code>	Optional horizontal gaze coordinate column.
<code>gaze_y_col</code>	Optional vertical gaze coordinate column.
<code>tracking_valid_col</code>	Optional tracking-validity column.
<code>required_cols</code>	Additional required columns.
<code>audit_objects</code>	Optional list of upstream audit/validation objects.
<code>min_rows</code>	Minimum acceptable number of rows.
<code>min_participants</code>	Minimum acceptable number of participants.
<code>min_trials</code>	Minimum acceptable number of participant-trial units.
<code>max_missing_pupil_prop</code>	Maximum acceptable pupil missingness proportion when pupil data are required.
<code>max_missing_gaze_prop</code>	Maximum acceptable gaze-coordinate missingness proportion when gaze coordinate columns are present.
<code>max_condition_imbalance_ratio</code>	Warning threshold for condition imbalance.
<code>name</code>	Character label stored in the returned object.

Details

This helper is a final decision wrapper. It complements, but does not replace, `validate_gazepoint_master()` or `create_gazepoint_analysis_decision_audit()`.

Value

A list with class `gp3_real_data_readiness_gate`.

check_sampling_rate *Check sampling rate by group*

Description

Check sampling rate by group

Usage

```
check_sampling_rate(data, group_cols = "MEDIA_ID", time_col = "TIME")
```

Arguments

data	A Gazepoint all-gaze data frame.
group_cols	Character vector of grouping columns, usually MEDIA_ID or c("participant_id", "MEDIA_ID").
time_col	Name of elapsed-time column.

Value

A tibble with sample interval and estimated Hz.

classify_gazepoint_export
Classify a Gazepoint export

Description

Uses the filename and header structure to classify a file as all_gaze, fixations, summary, or unknown.

Usage

```
classify_gazepoint_export(path)
```

Arguments

path	File path.
------	------------

Value

A single character string.

`compare_gazepoint_nested_models`*Compare nested Gazepoint models*

Description

Compare a sequence of nested models, such as null, main-effect, time, condition, and interaction models. The helper returns model-level fit indices, likelihood-ratio comparisons, ranking information, and extraction statuses.

Usage

```
compare_gazepoint_nested_models(  
  models,  
  model_names = NULL,  
  comparison = c("sequential", "against_first"),  
  name = "gazepoint_nested_model_comparison"  
)
```

Arguments

<code>models</code>	A list of fitted model objects.
<code>model_names</code>	Optional character vector of model names. If NULL, names are taken from <code>models</code> or generated as <code>model_1</code> , <code>model_2</code> , etc.
<code>comparison</code>	Comparison strategy. "sequential" compares each model with the previous model. "against_first" compares each model with the first model.
<code>name</code>	Character label stored in the returned object.

Details

This helper is useful for GCA, confirmatory LMM/GLMM workflows, AOI GLMMs, pupil LMMs, and other fitted model workflows where reviewers expect explicit model-comparison evidence.

Value

A list with class `gp3_nested_model_comparison`.

```
compute_gazepoint_aoi_transition_matrix
      Compute Gazepoint AOI transition matrices
```

Description

Compute AOI transition count and probability matrices from sample-level Gazepoint AOI data, AOI-entry tables, or AOI-sequence tables. The function returns both matrix and long-table forms.

Usage

```
compute_gazepoint_aoi_transition_matrix(
  data,
  aoi_col = NULL,
  time_col = "time",
  group_cols = c("subject", "MEDIA_ID", "trial_global"),
  by_cols = NULL,
  include_non_aoi = TRUE,
  include_self_transitions = TRUE,
  states = NULL,
  time_window = NULL,
  non_aoi_values = c("non_aoi", "none", "background", "outside", "outside_aoi",
    "missing", "missing_aoi"),
  missing_aoi_label = "missing_aoi"
)
```

Arguments

<code>data</code>	A Gazepoint sample-level data frame, AOI-entry table, or AOI-sequence table.
<code>aoi_col</code>	Name of the AOI-state column. Used only when data is sample-level data. If NULL, the function tries <code>aoi_current</code> , <code>AOI</code> , and <code>aoi_state</code> .
<code>time_col</code>	Name of the time column, in milliseconds. Used only when data is sample-level data.
<code>group_cols</code>	Character vector of columns defining independent AOI sequences, usually subject/media/trial.
<code>by_cols</code>	Optional character vector of columns used to compute separate matrices, for example condition or MEDIA_ID.
<code>include_non_aoi</code>	Logical. If TRUE, non-AOI/background states are included in the transition matrix.
<code>include_self_transitions</code>	Logical. If TRUE, same-state transitions are retained. These can occur after non-AOI states are removed.
<code>states</code>	Optional character vector giving the desired row/column order for the transition matrices.

<code>time_window</code>	Optional numeric vector of length 2 giving an entry-start time window in milliseconds.
<code>non_aoi_values</code>	Character vector of AOI labels treated as background or non-AOI states.
<code>missing_aoi_label</code>	Label used when the AOI value is missing.

Value

A list containing count matrices, probability matrices, and long-form transition counts/probabilities.

```
compute_gazepoint_time_varying_transition_matrix
```

Compute time-varying Gazepoint transition matrices

Description

Compute transition-count and transition-probability matrices across time windows. This helper is a convenience wrapper for studies where AOI/state transitions are expected to vary over the course of a stimulus, trial, or analysis window.

Usage

```
compute_gazepoint_time_varying_transition_matrix(
  data,
  from_col = NULL,
  to_col = NULL,
  time_col = NULL,
  window_col = NULL,
  window_size_ms = NULL,
  by_cols = NULL,
  count_col = NULL,
  states = NULL,
  complete_states = TRUE,
  drop_self_transitions = FALSE,
  normalise = c("row", "global", "none"),
  name = "gazepoint_time_varying_transition_matrix"
)
```

Arguments

<code>data</code>	A data frame containing transition-level rows.
<code>from_col</code>	Transition origin column. If NULL, common origin columns are detected automatically.
<code>to_col</code>	Transition destination column. If NULL, common destination columns are detected automatically.

time_col	Optional numeric time column used to construct windows when window_col = NULL.
window_col	Optional existing time-window column.
window_size_ms	Numeric window size used when window_col = NULL.
by_cols	Optional grouping columns, such as subject, condition, trial, or stimulus.
count_col	Optional count/weight column. If NULL, each row contributes one transition.
states	Optional character vector of allowed states/AOIs. If NULL, states are detected from from_col and to_col.
complete_states	If TRUE, complete all state-pair combinations within each time window and group.
drop_self_transitions	If TRUE, remove transitions where origin and destination are the same.
normalise	Probability normalisation. Options are "row", "global", and "none".
name	Character label stored in the returned object.

Value

A list with class `gp3_time_varying_transition_matrix`.

`compute_transition_matrix`
Compute an AOI transition matrix

Description

Compute an AOI transition matrix

Usage

```
compute_transition_matrix(  
  data,  
  group_cols = "MEDIA_ID",  
  aoi_col = "AOI",  
  time_col = "TIME",  
  collapse_repeats = TRUE  
)
```

Arguments

data	A Gazepoint data frame with AOI labels.
group_cols	Columns defining independent sequences.
aoi_col	AOI column name.
time_col	Time column name.
collapse_repeats	If TRUE, consecutive identical AOI labels are reduced to one visit before transitions are counted.

Value

A tibble with from, to, n, and prob.

```
create_gazepoint_analysis_decision_audit
```

Create a final Gazepoint analysis-decision audit

Description

Create a final audit table for a Gazepoint analysis workflow. The function records which analysis branches were run, classifies them as confirmatory, sensitivity, exploratory, diagnostic, preprocessing, or reporting branches, summarises available diagnostics, flags interpretation cautions, and creates a final analysis-readiness table.

Usage

```
create_gazepoint_analysis_decision_audit(
  ...,
  results = NULL,
  branch_roles = NULL,
  required_confirmatory = character(),
  diagnostics_required = TRUE,
  require_clean_diagnostics = FALSE
)
```

Arguments

...	Named analysis result objects. Each named object is treated as one analysis branch.
results	Optional named list of analysis result objects. This can be used instead of, or together with, ...
branch_roles	Optional data frame describing branch roles. It must contain branch_name and decision_type. Optional columns include analysis_family, interpretation_scope, and notes.
required_confirmatory	Character vector of confirmatory branches that must be present for the analysis to be considered complete.
diagnostics_required	Logical. If TRUE, confirmatory model branches without extractable diagnostics are flagged with a caution.
require_clean_diagnostics	Logical. If TRUE, diagnostic warnings in required confirmatory branches make the final readiness status not_ready.

Value

A list with class `gp3_analysis_decision_audit` containing overview, branch audit, diagnostics summary, interpretation cautions, readiness, and settings tables.

```
create_gazepoint_markovchain_object
```

Create a Gazepoint AOI Markov-chain object

Description

Create a dependency-free Markov-chain-style object from AOI/state sequences. The function computes transition counts, transition probabilities, and matrix representations from ordered gaze/AOI states. It does not require the external `markovchain` package; instead, it returns a lightweight `gp3tools` object that can be inspected, exported, or converted later.

Usage

```
create_gazepoint_markovchain_object(
  data,
  state_col = NULL,
  participant_col = NULL,
  trial_col = NULL,
  time_col = NULL,
  sequence_id_cols = NULL,
  state_order = NULL,
  exclude_states = c("missing", "missing_aoi", "missing_coordinate", "trackloss",
    "track_loss"),
  missing_state_label = NULL,
  include_self_transitions = TRUE,
  laplace = 0,
  empty_state_handling = c("self", "zero", "NA"),
  name = "gazepoint_markovchain"
)
```

Arguments

<code>data</code>	A data frame containing ordered AOI/state observations.
<code>state_col</code>	AOI/state column. If <code>NULL</code> , common AOI/state column names are detected automatically.
<code>participant_col</code>	Optional participant/subject column.
<code>trial_col</code>	Optional trial/sequence column.
<code>time_col</code>	Optional time/order column.
<code>sequence_id_cols</code>	Optional character vector of columns defining separate sequences. If <code>NULL</code> , participant and trial columns are used when available.

state_order	Optional character vector giving the preferred state order in the output matrices.
exclude_states	Character vector of states to exclude before transition calculation.
missing_state_label	Optional label used to retain missing states. If NULL, missing/blank states are removed.
include_self_transitions	Logical. If FALSE, transitions from a state to the same state are removed.
laplace	Numeric smoothing value added to all transition cells when computing probabilities.
empty_state_handling	How to handle states with no outgoing transitions: "self" creates an absorbing self-transition, "zero" leaves a zero row, and "NA" returns NA probabilities for that row.
name	Character label stored in the returned object.

Value

A list with class `gp3_markovchain_object`.

`create_gazepoint_master`

Create a master long-format dataset from Gazepoint all-gaze data

Description

Converts Gazepoint all-gaze exports imported with `read_gazepoint()` or `read_gazepoint_folder()` into a master sample-level structure suitable for quality checks, AOI summaries, pupil preprocessing, time-course analyses, and publication reporting.

Usage

```
create_gazepoint_master(
  gaze_data,
  screen_width_px = NULL,
  screen_height_px = NULL,
  screen_width_cm = NULL,
  screen_height_cm = NULL,
  viewing_distance_cm = NULL,
  time_unit = c("seconds", "milliseconds"),
  user_col = "USER_FILE",
  media_col = "MEDIA_ID",
  media_name_col = "MEDIA_NAME",
  tracker_model = "Gazepoint",
  tracker_sampling_rate = 60,
  event_latency_offset_ms = 0,
  baseline_window = NULL,
  analysis_window = NULL
)
```

Arguments

gaze_data	Gazepoint all-gaze data frame.
screen_width_px	Optional screen width in pixels. If provided and gaze coordinates are normalised, x-coordinates are converted to pixels.
screen_height_px	Optional screen height in pixels. If provided and gaze coordinates are normalised, y-coordinates are converted to pixels.
screen_width_cm	Optional physical screen width in centimetres.
screen_height_cm	Optional physical screen height in centimetres.
viewing_distance_cm	Optional viewing distance in centimetres.
time_unit	Unit of the Gazepoint TIME column. Usually "seconds".
user_col	Column identifying the source/user file.
media_col	Column identifying the stimulus/media.
media_name_col	Column identifying the media/stimulus name.
tracker_model	Tracker label stored in the master data.
tracker_sampling_rate	Expected tracker sampling rate.
event_latency_offset_ms	Optional event-latency correction in milliseconds.
baseline_window	Optional numeric vector of length 2 giving baseline start and end in milliseconds.
analysis_window	Optional numeric vector of length 2 giving analysis start and end in milliseconds.

Value

A tibble with one row per Gazepoint sample and publication-oriented master columns.

```
create_gazepoint_preprocessing_multiverse
  Create a Gazepoint preprocessing multiverse
```

Description

Create a preprocessing multiverse specification for Gazepoint pupil and AOI workflows. The returned object defines alternative preprocessing decisions that can later be passed to pupil or AOI multiverse runners.

Usage

```

create_gazepoint_preprocessing_multiverse(
  pupil_max_gap_ms = c(75, 150, 250),
  pupil_smoothing_window_samples = c(3L, 5L, 7L),
  pupil_baseline_windows = list(c(0, 200), c(-200, 0)),
  pupil_artifact_padding_ms = c(0, 50),
  aoi_denominators = c("valid", "all", "aoi_only"),
  aoi_min_denominator_samples = c(1L, 5L, 10L),
  include_pupil = TRUE,
  include_aoi = TRUE,
  label_prefix = "mv"
)

```

Arguments

`pupil_max_gap_ms` Numeric vector of maximum pupil-interpolation gap durations in milliseconds.

`pupil_smoothing_window_samples` Integer vector of pupil smoothing window sizes in samples.

`pupil_baseline_windows` List of numeric vectors of length 2 defining pupil baseline windows in milliseconds.

`pupil_artifact_padding_ms` Numeric vector of artifact-padding values in milliseconds.

`aoi_denominators` Character vector of AOI denominator definitions. Typical values are "valid", "all", and "aoi_only".

`aoi_min_denominator_samples` Integer vector of minimum denominator sample thresholds for AOI-window modelling.

`include_pupil` Logical. If TRUE, create pupil preprocessing branches.

`include_aoi` Logical. If TRUE, create AOI preprocessing branches.

`label_prefix` Character prefix used for branch identifiers.

Value

A list with class `gp3_preprocessing_multiverse` containing overview, pupil grid, AOI grid, combined grid, and settings tables.

create_gazepoint_preprocessing_registry

Create a Gazepoint pupil-preprocessing registry

Description

Creates a compact registry of commonly used preprocessing parameters for Gazepoint pupil and gaze analyses. The registry is designed to make preprocessing choices explicit, auditable, and easy to report.

Usage

```
create_gazepoint_preprocessing_registry(  
    blink_padding_pre_ms = 100,  
    blink_padding_post_ms = 100,  
    max_interpolation_gap_ms = 150,  
    smoothing_window_ms = 50,  
    baseline_start_ms = -200,  
    baseline_end_ms = 0,  
    pupil_physiological_min = 1,  
    pupil_physiological_max = 9,  
    pupil_speed_mad_k = 6,  
    binocular_mad_k = 6,  
    baseline_missing_prop_threshold = 0.3,  
    baseline_interpolated_prop_threshold = 0.3,  
    baseline_artifact_prop_threshold = 0.3,  
    overlap_trial_duration_ms = 3000,  
    overlap_event_gap_ms = 1000  
)
```

Arguments

blink_padding_pre_ms
Padding before bad pupil samples, blinks, or tracking artifacts, in milliseconds. Defaults to 100.

blink_padding_post_ms
Padding after bad pupil samples, blinks, or tracking artifacts, in milliseconds. Defaults to 100.

max_interpolation_gap_ms
Maximum missing-pupil gap duration to interpolate, in milliseconds. Defaults to 150.

smoothing_window_ms
Rolling smoothing window, in milliseconds. Defaults to 50.

baseline_start_ms
Baseline-window start, in milliseconds. Defaults to -200.

baseline_end_ms
Baseline-window end, in milliseconds. Defaults to 0.

pupil_physiological_min
Minimum plausible pupil value when the pupil unit is known to be millimetres. Defaults to 1.

pupil_physiological_max
Maximum plausible pupil value when the pupil unit is known to be millimetres. Defaults to 9.

pupil_speed_mad_k	MAD multiplier for pupil-speed outlier detection. Defaults to 6.
binocular_mad_k	MAD multiplier for left-right pupil disagreement. Defaults to 6.
baseline_missing_prop_threshold	Baseline missingness threshold used for baseline-quality audits. Defaults to 0.30.
baseline_interpolated_prop_threshold	Baseline interpolation threshold used for baseline-quality audits. Defaults to 0.30.
baseline_artifact_prop_threshold	Baseline artifact threshold used for baseline-quality audits. Defaults to 0.30.
overlap_trial_duration_ms	Trial-duration threshold below which pupil overlap/deconvolution risk should be considered. Defaults to 3000.
overlap_event_gap_ms	Event-gap threshold below which pupil-response overlap should be considered. Defaults to 1000.

Value

A tibble with one row per preprocessing parameter.

Examples

```
registry <- create_gazepoint_preprocessing_registry()
registry
```

```
create_gazepoint_report
```

Create a Gazepoint HTML diagnostic report

Description

Creates a lightweight HTML report from a gp3tools workflow result object. The report includes dataset dimensions, sampling-rate checks, flagged recordings, AOI summaries, and standard diagnostic plots.

Usage

```
create_gazepoint_report(
  results,
  output_file,
  title = "Gazepoint diagnostic report",
  overwrite = TRUE,
  max_rows = 30,
```

```
    save_plots = TRUE,  
    plot_dir = NULL  
  )
```

Arguments

results	A named list returned by run_gazepoint_workflow().
output_file	Path to the HTML report file to create.
title	Report title.
overwrite	Logical. If FALSE, stop when the report file already exists.
max_rows	Maximum number of rows to show in preview tables.
save_plots	Logical. If TRUE, save and include diagnostic plots.
plot_dir	Optional folder where report plot files should be saved. If NULL, a folder next to the report is created.

Value

A tibble with the written report path and plot folder.

create_gazepoint_reporting_checklist

Create a Gazepoint reporting checklist

Description

Create an auto-generated reporting checklist for Gazepoint/gp3tools analyses. The checklist summarises whether key dataset, preprocessing, quality-control, AOI, pupil, modelling, sensitivity, and reproducibility elements are present or still need reporting.

Usage

```
create_gazepoint_reporting_checklist(  
  data = NULL,  
  objects = NULL,  
  analysis_type = c("general", "pupil", "aoi", "combined"),  
  study_title = NULL,  
  required_sections = NULL,  
  include_optional = TRUE,  
  name = "gazepoint_reporting_checklist"  
)
```

Arguments

data	Optional Gazepoint or gp3tools data frame.
objects	Optional list of gp3tools audit, model, workflow, readiness, or external-check objects.
analysis_type	Analysis target. Options are "general", "pupil", "aoi", and "combined".
study_title	Optional study title or short label.
required_sections	Optional character vector of checklist item IDs that should be treated as required. If NULL, a default set is used.
include_optional	Logical. If TRUE, include optional advanced-methods reporting items.
name	Character label stored in the returned object.

Details

This helper is intended as a reporting aid. It does not replace the underlying audit, preprocessing, modelling, or readiness-gate functions.

Value

A list with class `gp3_reporting_checklist`.

diagnose_gazepoint_gamm

Diagnose GAM and BAM models

Description

Run a compact diagnostics bundle for `mgcv` GAM/BAM models used in `gp3tools` workflows. The function combines convergence, basis-dimension checks, overdispersion checks, and optional DHARMA simulation-based residual diagnostics.

Usage

```
diagnose_gazepoint_gamm(
  model,
  model_name = NULL,
  check_convergence = TRUE,
  check_basis = TRUE,
  check_overdispersion = TRUE,
  use_dharma = FALSE,
  dharma_simulations = 250,
  seed = 123
)
```

Arguments

model	A fitted GAM/BAM object, a gp3tools fit object containing \$model, or a named list of fitted model objects.
model_name	Optional model label used in returned tables.
check_convergence	Logical. If TRUE, run convergence diagnostics.
check_basis	Logical. If TRUE, run mgcv::k.check() basis-dimension diagnostics when available.
check_overdispersion	Logical. If TRUE, run overdispersion diagnostics when meaningful for the model family.
use_dharma	Logical. If TRUE, try to run optional DHARMA diagnostics.
dharma_simulations	Number of DHARMA simulations.
seed	Random seed used before DHARMA simulation.

Details

The function accepts raw mgcv::gam() / mgcv::bam() model objects, gp3tools fit objects containing a \$model element, or a named list of fitted model objects.

Value

A list with overview, convergence, basis, overdispersion, DHARMA diagnostics, and settings.

diagnose_gazepoint_glm

Diagnose GLMM, LMM, and GLM models

Description

Run a compact diagnostics bundle for model objects used in gp3tools workflows. The function combines convergence, singularity, overdispersion, and optional DHARMA simulation-based residual diagnostics.

Usage

```
diagnose_gazepoint_glm(
  model,
  model_name = NULL,
  check_convergence = TRUE,
  check_singularity = TRUE,
  check_overdispersion = TRUE,
  use_dharma = TRUE,
  dharma_simulations = 250,
  seed = 123
)
```

Arguments

model	A fitted model object, a gp3tools fit object containing \$model, or a named list of fitted model objects.
model_name	Optional model label used in returned tables.
check_convergence	Logical. If TRUE, run convergence diagnostics.
check_singularity	Logical. If TRUE, run singularity diagnostics.
check_overdispersion	Logical. If TRUE, run overdispersion diagnostics.
use_dharma	Logical. If TRUE, try to run optional DHARMA diagnostics.
dharma_simulations	Number of DHARMA simulations.
seed	Random seed used before DHARMA simulation.

Details

The function accepts raw fitted models, gp3tools fit objects containing a \$model element, or a named list of fitted models. DHARMA diagnostics are optional and are skipped cleanly when DHARMA is not installed.

Value

A list with overview, convergence, singularity, overdispersion, DHARMA diagnostics, and settings.

estimate_gazepoint_divergence_point

Estimate a bootstrapped divergence point between two Gazepoint time courses

Description

Estimate the earliest time point at which two condition-level time courses reliably diverge. The helper computes observed condition curves, bootstraps the condition difference, identifies the first time point where the bootstrap confidence interval excludes the null value for a requested number of consecutive time points, and returns a bootstrap uncertainty interval for the divergence onset.

Usage

```
estimate_gazepoint_divergence_point(
  data,
  outcome_col,
  time_col,
  condition_col,
  participant_col = NULL,
```

```

trial_col = NULL,
comparison = NULL,
bootstrap_unit = c("participant", "trial", "row"),
summary_function = c("mean", "median"),
n_boot = 1000L,
ci = 0.95,
consecutive_points = 1L,
null_value = 0,
min_abs_difference = 0,
direction = c("two_sided", "positive", "negative"),
seed = NULL,
keep_bootstrap = TRUE,
name = "gazepoint_divergence_point"
)

```

Arguments

<code>data</code>	A data frame containing time-course observations.
<code>outcome_col</code>	Outcome column, for example pupil size, fixation probability, gaze proportion, or AOI time-course value.
<code>time_col</code>	Time column.
<code>condition_col</code>	Condition column. Exactly two conditions are compared unless <code>comparison</code> is supplied.
<code>participant_col</code>	Optional participant column used for participant-level bootstrap resampling.
<code>trial_col</code>	Optional trial column used for trial-level bootstrap resampling.
<code>comparison</code>	Optional character vector of two condition values. The estimated difference is <code>comparison[2] - comparison[1]</code> .
<code>bootstrap_unit</code>	Resampling unit. Options are "participant", "trial", and "row".
<code>summary_function</code>	Function used to summarise observations within condition-by-time cells. Options are "mean" and "median".
<code>n_boot</code>	Number of bootstrap resamples.
<code>ci</code>	Confidence level for bootstrap intervals.
<code>consecutive_points</code>	Number of consecutive time points required before declaring divergence.
<code>null_value</code>	Null difference value. Default is 0.
<code>min_abs_difference</code>	Optional minimum absolute observed difference required at a time point.
<code>direction</code>	Direction of divergence. "two_sided" checks whether the bootstrap interval excludes <code>null_value</code> in either direction. "positive" checks whether <code>comparison[2] > comparison[1]</code> . "negative" checks whether <code>comparison[2] < comparison[1]</code> .
<code>seed</code>	Optional random seed for reproducible bootstrap resampling.
<code>keep_bootstrap</code>	Logical. If TRUE, return bootstrap differences for each time point.
<code>name</code>	Character label stored in the returned object.

Details

This helper complements cluster-permutation analysis. Cluster permutation asks where a reliable time window exists; divergence-point analysis asks when the condition difference first emerges.

Value

A list with class `gp3_divergence_point_analysis`.

```
export_gazepoint_master_audit
```

Export a Gazepoint master table, audit tables, and validation tables

Description

Exports a Gazepoint master sample-level table together with the output of `audit_gazepoint_master()` and, optionally, `validate_gazepoint_master()`. This function is useful after creating a master table with `as_gazepoint_master()` or `create_gazepoint_master()`.

Usage

```
export_gazepoint_master_audit(
  master,
  audit = NULL,
  validation = NULL,
  output_dir = NULL,
  prefix = "gazepoint",
  export_master = TRUE,
  export_audit = TRUE,
  export_validation = TRUE,
  overwrite = TRUE,
  na = ""
)
```

Arguments

<code>master</code>	A Gazepoint master sample-level table.
<code>audit</code>	Optional audit list returned by <code>audit_gazepoint_master()</code> . If NULL, the audit is created automatically.
<code>validation</code>	Optional validation list returned by <code>validate_gazepoint_master()</code> . If NULL and <code>export_validation = TRUE</code> , validation is created automatically.
<code>output_dir</code>	Directory where CSV files should be written. Must be supplied explicitly.
<code>prefix</code>	File prefix used for all exported CSV files.
<code>export_master</code>	Logical. If TRUE, exports the full master table.
<code>export_audit</code>	Logical. If TRUE, exports audit tables.

export_validation	Logical. If TRUE, exports validation tables.
overwrite	Logical. If FALSE, the function aborts when any target file already exists.
na	String used for missing values in CSV output.

Value

A tibble listing the exported files, table names, and dimensions.

Examples

```

master <- tibble::tibble(
  subject = c("P1", "P1", "P2", "P2"),
  MEDIA_ID = c("M1", "M1", "M1", "M1"),
  time = c(0, 16, 0, 16),
  x = c(100, 120, 200, 220),
  y = c(100, 130, 200, 230),
  raw_x = c(0.10, 0.12, 0.20, 0.22),
  raw_y = c(0.20, 0.26, 0.30, 0.34),
  valid_sample = c(TRUE, TRUE, TRUE, TRUE),
  missing_gaze = c(FALSE, FALSE, FALSE, FALSE),
  missing_pupil = c(FALSE, FALSE, FALSE, FALSE),
  gaze_offscreen = c(FALSE, FALSE, FALSE, FALSE),
  mean_pupil = c(3.5, 3.6, 3.7, 3.8),
  aoi_current = c("AOI 1", "AOI 1", "AOI 2", "AOI 2"),
  aoi_count = c(1L, 1L, 1L, 1L),
  screen_width_px = rep(1000, 4),
  screen_height_px = rep(500, 4)
)

exported <- export_gazepoint_master_audit(
  master = master,
  output_dir = file.path(tempdir(), "gp3_master_audit"),
  prefix = "example"
)

exported

```

export_gazepoint_model_tables

Export manuscript-ready model tables

Description

Export model-summary tables and optional estimated marginal means tables to CSV files. The function is designed for objects returned by `tidy_gazepoint_model_summary()` and `summarise_gazepoint_emmeans()`.

Usage

```
export_gazepoint_model_tables(  
  model_summary = NULL,  
  emmeans_summary = NULL,  
  output_dir,  
  prefix = "gazepoint_model",  
  overwrite = TRUE,  
  include_diagnostics = TRUE  
)
```

Arguments

`model_summary` Optional object returned by `tidy_gazepoint_model_summary()`.

`emmeans_summary` Optional object returned by `summarise_gazepoint_emmeans()`.

`output_dir` Output directory.

`prefix` File-name prefix.

`overwrite` Logical. If FALSE, existing output files cause an error.

`include_diagnostics` Logical. If TRUE, export available diagnostic component tables from `model_summary`.

Value

A tibble indexing the written files.

export_gazepoint_tables

Export Gazepoint analysis tables to CSV files

Description

Writes a named list of analysis tables to CSV files in an output folder.

Usage

```
export_gazepoint_tables(  
  tables,  
  output_dir,  
  prefix = NULL,  
  overwrite = TRUE,  
  na = ""  
)
```

Arguments

tables	A named list of data frames or tibbles.
output_dir	Folder where CSV files should be written.
prefix	Optional filename prefix.
overwrite	Logical. If FALSE, the function stops when a target file already exists.
na	Value used for missing values in the exported CSV files.

Value

A tibble with table names and written file paths.

fit_gazepoint_aoi_gamm

Fit AOI time-course GAMMs

Description

Fit binomial GAMMs for AOI target-looking time courses prepared by `prepare_gazepoint_aoi_gamm_data()`. The model uses target-looking successes and failures over time and can include condition effects, condition-specific smooths, and subject random-effect smooths.

Usage

```
fit_gazepoint_aoi_gamm(  
  data,  
  include_condition = TRUE,  
  condition_smooths = TRUE,  
  random_subject = TRUE,  
  random_subject_time = FALSE,  
  time_k = 10,  
  subject_time_k = 5,  
  family = stats::binomial(),  
  method = "fREML",  
  discrete = FALSE,  
  select = FALSE,  
  drop_non_ok = TRUE,  
  min_rows = 10,  
  min_subjects = 2,  
  min_time_bins = 4,  
  ...  
)
```

Arguments

<code>data</code>	A data frame returned by <code>prepare_gazepoint_aoi_gamm_data()</code> .
<code>include_condition</code>	Logical. If TRUE, include condition as a parametric fixed effect when two or more conditions are available.
<code>condition_smooths</code>	Logical. If TRUE, fit condition-specific time smooths when two or more conditions are available.
<code>random_subject</code>	Logical. If TRUE, include a subject random-effect smooth.
<code>random_subject_time</code>	Logical. If TRUE, include subject-specific factor-smooth time deviations. This can be useful for repeated-measures time-course data but may be too heavy for very small datasets.
<code>time_k</code>	Basis dimension for the main time smooth.
<code>subject_time_k</code>	Basis dimension for subject-specific factor-smooth time deviations.
<code>family</code>	Model family. Defaults to <code>stats::binomial()</code> .
<code>method</code>	Smoothing-parameter estimation method passed to <code>mgcv::bam()</code> .
<code>discrete</code>	Logical passed to <code>mgcv::bam()</code> .
<code>select</code>	Logical passed to <code>mgcv::bam()</code> .
<code>drop_non_ok</code>	Logical. If TRUE, keep only rows with <code>.gp3_aoi_gamm_status == "ok"</code> before fitting.
<code>min_rows</code>	Minimum number of rows required for model fitting.
<code>min_subjects</code>	Minimum number of subjects required for model fitting.
<code>min_time_bins</code>	Minimum number of time bins required for model fitting.
<code>...</code>	Additional arguments passed to <code>mgcv::bam()</code> .

Details

This function is intended for AOI time-course modelling. It is separate from confirmatory AOI-window GLMMs and from cluster-based permutation tests.

Value

A list containing the fitted model, formula, model status, diagnostics, parametric table, smooth table, and settings.

 fit_gazepoint_aoi_model_sensitivity

Fit AOI-window model-family sensitivity checks

Description

Fit a compact set of sensitivity models for AOI-window outcomes. The main model is a binomial GLMM. Additional checks can include an empirical-logit LMM, a weighted proportion LMM, and a fixed-effects quasibinomial GLM.

Usage

```
fit_gazepoint_aoi_model_sensitivity(
  data,
  success_col = "aoi_glmm_success",
  failure_col = "aoi_glmm_failure",
  denominator_col = "aoi_glmm_denominator",
  proportion_col = "aoi_glmm_prop",
  subject_col = "aoi_glmm_subject",
  condition_col = "aoi_glmm_condition",
  window_col = "aoi_glmm_window",
  model_types = c("binomial_glmm", "empirical_logit_lmm", "proportion_lmm",
    "quasibinomial_glm"),
  include_condition = TRUE,
  include_window = TRUE,
  include_interaction = TRUE,
  random_intercept = TRUE,
  optimizer = "bobyqa",
  maxfun = 2e+05,
  nAGQ = 0,
  empirical_logit_correction = 0.5,
  drop_missing = TRUE
)
```

Arguments

data	AOI GLMM data returned by <code>prepare_gazepoint_aoi_glmm_data()</code> .
success_col	Success-count column.
failure_col	Failure-count column.
denominator_col	Denominator column.
proportion_col	Proportion column.
subject_col	Subject column.
condition_col	Condition column.
window_col	Window column.

model_types	Character vector of model types. Supported values are "binomial_glmm", "empirical_logit_lmm", "proportion_lmm", and "quasibinomial_glm".
include_condition	Logical. Include condition fixed effect when possible.
include_window	Logical. Include window fixed effect when possible.
include_interaction	Logical. Include condition-by-window interaction when both condition and window are included.
random_intercept	Logical. Include subject random intercept in mixed sensitivity models.
optimizer	Optimizer for lme4 mixed models.
maxfun	Maximum optimizer evaluations.
nAGQ	Number of adaptive Gauss-Hermite quadrature points for the binomial GLMM.
empirical_logit_correction	Small correction added to success and failure counts for empirical-logit models.
drop_missing	Logical. Drop rows with missing model variables.

Value

A list containing fitted models, formulas, comparison table, fixed effects table, settings, and status information.

```
fit_gazepoint_aoi_window_glmm
```

Fit an AOI-window binomial GLMM

Description

Fit a confirmatory AOI-window mixed-effects logistic regression from data prepared by prepare_gazepoint_aoi_glmm_data.

Usage

```
fit_gazepoint_aoi_window_glmm(
  data,
  success_col = "aoi_glmm_success",
  failure_col = "aoi_glmm_failure",
  subject_col = "aoi_glmm_subject",
  condition_col = "aoi_glmm_condition",
  window_col = "aoi_glmm_window",
  include_condition = TRUE,
  include_window = TRUE,
  include_interaction = TRUE,
  random_intercept = TRUE,
  random_window_slopes = FALSE,
  fallback_on_singular = TRUE,
```

```

optimizer = "bobyqa",
maxfun = 2e+05,
nAGQ = 0,
drop_missing = TRUE
)

```

Arguments

<code>data</code>	AOI GLMM data returned by <code>prepare_gazepoint_aoi_glmm_data()</code> .
<code>success_col</code>	Success-count column.
<code>failure_col</code>	Failure-count column.
<code>subject_col</code>	Subject factor/column.
<code>condition_col</code>	Condition factor/column.
<code>window_col</code>	AOI-window factor/column.
<code>include_condition</code>	Logical. Include condition fixed effects when at least two conditions are available.
<code>include_window</code>	Logical. Include window fixed effects when at least two windows are available.
<code>include_interaction</code>	Logical. Include condition-by-window interaction when both condition and window fixed effects are included.
<code>random_intercept</code>	Logical. Include subject random intercept.
<code>random_window_slopes</code>	Logical. Attempt subject-level random slopes for AOI window.
<code>fallback_on_singular</code>	Logical. If TRUE, fall back to a simpler random intercept model when a random-slope model is singular or fails.
<code>optimizer</code>	Optimizer passed to <code>lme4::glmerControl()</code> .
<code>maxfun</code>	Maximum optimizer evaluations.
<code>nAGQ</code>	Number of adaptive Gauss-Hermite quadrature points.
<code>drop_missing</code>	Logical. Drop rows with missing model variables before fitting.

Value

A list with fitted model, attempted model, formulas, comparison table, settings, status fields, and model data.

fit_gazepoint_gca *Fit a Gazepoint Growth Curve Analysis mixed model*

Description

Fit a Growth Curve Analysis (GCA) mixed model to prepared pupil time-course data. The function first attempts a random-intercept plus random-time-slopes model and, if the model fails or is singular, falls back to a random-intercept model.

Usage

```
fit_gazepoint_gca(
  data,
  outcome_col = "gca_pupil",
  subject_col = "subject",
  condition_col = "condition",
  time_terms = NULL,
  degree = NULL,
  weights_col = "gca_weight",
  use_weights = TRUE,
  random_slopes = TRUE,
  fallback_on_singular = TRUE,
  REML = FALSE,
  optimizer = "bobyqa",
  maxfun = 2e+05,
  drop_missing = TRUE
)
```

Arguments

data	A data frame created by <code>prepare_gazepoint_gca_data()</code> .
outcome_col	Name of the GCA outcome column.
subject_col	Name of the subject column.
condition_col	Name of the condition column.
time_terms	Optional character vector of polynomial time-term columns. If NULL, terms named <code>time_poly_1</code> , <code>time_poly_2</code> , ... are detected.
degree	Optional number of polynomial terms to use. If supplied and <code>time_terms = NULL</code> , the function uses <code>time_poly_1</code> through <code>time_poly_degree</code> .
weights_col	Optional weights column. Use NULL for unweighted models.
use_weights	Logical. If TRUE, uses <code>weights_col</code> when available.
random_slopes	Logical. If TRUE, first attempts random slopes for all polynomial time terms.
fallback_on_singular	Logical. If TRUE, falls back to a random-intercept model when the random-slope model is singular.

REML	Logical passed to <code>lme4::lmer()</code> .
optimizer	Optimizer passed to <code>lme4::lmerControl()</code> .
maxfun	Maximum optimizer function evaluations.
drop_missing	Logical. If TRUE, rows with missing model variables are removed before fitting.

Value

A list of class `gp3_gca_model` containing the fitted model, attempted and final formulas, model comparison information, settings, and status.

fit_gazepoint_pupil_gamm

Fit a Gazepoint pupil GAMM

Description

Fit a generalized additive mixed model for binned pupil time-course data using `mgcv::bam()`. The function is designed to work with data prepared by `prepare_gazepoint_pupil_gamm_data()`.

Usage

```
fit_gazepoint_pupil_gamm(
  data,
  pupil_col = "mean_pupil",
  time_col = "time_bin_center_ms",
  subject_col = "subject",
  condition_col = "condition",
  n_time_basis = 10,
  use_condition_smooths = TRUE,
  include_subject_random_effect = TRUE,
  family = c("gaussian", "scat"),
  method = "fREML",
  discrete = TRUE,
  rho = NULL,
  ar_start_col = "AR.start",
  weights_col = NULL,
  drop_missing = TRUE
)
```

Arguments

data	A binned pupil time-course data frame.
pupil_col	Name of the dependent pupil column.
time_col	Name of the time-bin centre column.
subject_col	Name of the subject column.

condition_col	Name of the condition column.
n_time_basis	Basis dimension for smooth time terms.
use_condition_smooths	Logical. If TRUE, condition-specific smooths are added when the condition column has more than one level.
include_subject_random_effect	Logical. If TRUE, adds a subject random-effect smooth.
family	Model family. Use "gaussian" for the default Gaussian model or "scat" for mgcv's scaled-t family.
method	Smoothing-parameter estimation method passed to mgcv::bam().
discrete	Logical passed to mgcv::bam().
rho	Optional AR(1) correlation parameter passed to mgcv::bam().
ar_start_col	Optional AR-start column. If present and rho is not NULL, it is passed to mgcv::bam() as AR.start.
weights_col	Optional weights column.
drop_missing	Logical. If TRUE, rows with missing model variables are removed before fitting.

Value

A list of class gp3_pupil_gamm containing the fitted model, formula, data, settings, and status information.

```
fit_gazepoint_pupil_pfe_gamm
```

Fit a gaze-position-adjusted pupil GAMM sensitivity model

Description

Fit and compare a main pupil GAMM with a gaze-position-adjusted sensitivity model. The adjusted model adds a two-dimensional tensor-product smooth over mean gaze x/y position using `te(mean_x, mean_y)`.

Usage

```
fit_gazepoint_pupil_pfe_gamm(
  data,
  pupil_col = "mean_pupil",
  time_col = "time_bin_center_ms",
  subject_col = "subject",
  condition_col = "condition",
  x_col = "mean_x",
  y_col = "mean_y",
  n_time_basis = 10,
  n_position_basis = 8,
```

```

    use_condition_smooths = TRUE,
    include_subject_random_effect = TRUE,
    family = c("gaussian", "scat"),
    method = "fREML",
    discrete = TRUE,
    rho = NULL,
    ar_start_col = "AR.start",
    weights_col = NULL,
    drop_missing = TRUE
  )

```

Arguments

<code>data</code>	A binned pupil time-course data frame, usually created by <code>prepare_gazepoint_pupil_gamm_data()</code> .
<code>pupil_col</code>	Name of the dependent pupil column.
<code>time_col</code>	Name of the time-bin centre column.
<code>subject_col</code>	Name of the subject column.
<code>condition_col</code>	Name of the condition column.
<code>x_col</code>	Name of the mean gaze x-position column.
<code>y_col</code>	Name of the mean gaze y-position column.
<code>n_time_basis</code>	Basis dimension for time smooths.
<code>n_position_basis</code>	Basis dimension for gaze-position smooths.
<code>use_condition_smooths</code>	Logical. If TRUE, condition-specific time smooths are used when multiple conditions are present.
<code>include_subject_random_effect</code>	Logical. If TRUE, adds a subject random-effect smooth.
<code>family</code>	Model family. Use "gaussian" or "scat".
<code>method</code>	Smoothing-parameter estimation method passed to <code>mgcv::bam()</code> .
<code>discrete</code>	Logical passed to <code>mgcv::bam()</code> .
<code>rho</code>	Optional AR(1) correlation parameter passed to <code>mgcv::bam()</code> .
<code>ar_start_col</code>	Optional AR-start column.
<code>weights_col</code>	Optional weights column.
<code>drop_missing</code>	Logical. If TRUE, rows with missing model variables are removed before fitting.

Value

A list of class `gp3_pupil_pfe_gamm` containing the main model, the gaze-position-adjusted model, formulas, comparison table, settings, and status information.

```
fit_gazepoint_pupil_window_lmm
```

Fit confirmatory pupil-window linear mixed models

Description

Fit the main confirmatory trial/window-level pupil model from data prepared with `prepare_gazepoint_pupil_window_model_data()`. The default model is a linear mixed model with pupil outcome as the continuous dependent variable, condition and/or window fixed effects when available, and a subject random intercept when feasible.

Usage

```
fit_gazepoint_pupil_window_lmm(
  data,
  formula = NULL,
  outcome_col = "pupil_model_outcome",
  subject_col = "pupil_model_subject",
  condition_col = "pupil_model_condition",
  window_col = "pupil_model_window",
  weights_col = "pupil_model_weight",
  use_weights = FALSE,
  include_condition = TRUE,
  include_window = TRUE,
  include_interaction = TRUE,
  random_intercept = TRUE,
  random_window_slopes = FALSE,
  fallback_on_singular = TRUE,
  REML = FALSE,
  optimizer = "bobyqa",
  maxfun = 2e+05,
  drop_missing = TRUE,
  ...
)
```

Arguments

<code>data</code>	Pupil-window model data, usually produced by <code>prepare_gazepoint_pupil_window_model_data()</code> .
<code>formula</code>	Optional model formula. If <code>NULL</code> , a formula is constructed automatically.
<code>outcome_col</code>	Outcome column.
<code>subject_col</code>	Subject column.
<code>condition_col</code>	Condition column.
<code>window_col</code>	Window column.
<code>weights_col</code>	Optional weights column.
<code>use_weights</code>	Logical. If <code>TRUE</code> , use <code>weights_col</code> as model weights.

include_condition	Logical. Include condition fixed effects when more than one condition level is available.
include_window	Logical. Include window fixed effects when more than one window level is available.
include_interaction	Logical. Include the condition-by-window interaction when both condition and window are used.
random_intercept	Logical. Include a subject random intercept when feasible.
random_window_slopes	Logical. Attempt subject-level random window slopes when feasible.
fallback_on_singular	Logical. If TRUE, fall back from a random-slope model to a random-intercept model when the attempted model is singular or fails.
REML	Logical. Passed to <code>lme4::lmer()</code> .
optimizer	Optimizer passed to <code>lme4::lmerControl()</code> .
maxfun	Maximum optimizer iterations passed to <code>lme4::lmerControl()</code> .
drop_missing	Logical. If TRUE, rows with missing or non-finite model inputs are removed before fitting.
...	Additional arguments passed to <code>lme4::lmer()</code> or <code>stats::lm()</code> .

Value

A list containing the fitted model, formula, attempted model, fallback information, fixed effects, comparison table, settings, and model diagnostics.

fit_gazepoint_pupil_window_sensitivity

Run sensitivity models for confirmatory pupil-window analyses

Description

Run a compact set of sensitivity models for confirmatory pupil-window analyses. Supported model families are the main linear mixed model, a weighted linear mixed model, a fixed-effects linear model, and a weighted fixed-effects linear model. Weighted models use the prepared valid-sample count column as weights by default.

Usage

```
fit_gazepoint_pupil_window_sensitivity(
  data,
  outcome_col = "pupil_model_outcome",
  subject_col = "pupil_model_subject",
  condition_col = "pupil_model_condition",
```

```

window_col = "pupil_model_window",
weights_col = "pupil_model_weight",
model_types = c("lmm", "weighted_lmm", "lm", "weighted_lm"),
include_condition = TRUE,
include_window = TRUE,
include_interaction = TRUE,
random_intercept = TRUE,
random_window_slopes = FALSE,
fallback_on_singular = TRUE,
REML = FALSE,
optimizer = "bobyqa",
maxfun = 2e+05,
drop_missing = TRUE,
...
)

```

Arguments

<code>data</code>	Pupil-window model data, usually produced by <code>prepare_gazepoint_pupil_window_model_data()</code> .
<code>outcome_col</code>	Outcome column.
<code>subject_col</code>	Subject column.
<code>condition_col</code>	Condition column.
<code>window_col</code>	Window column.
<code>weights_col</code>	Optional weights column.
<code>model_types</code>	Character vector of model types to fit. Supported values are "lmm", "weighted_lmm", "lm", and "weighted_lm".
<code>include_condition</code>	Logical. Include condition fixed effects when more than one condition level is available.
<code>include_window</code>	Logical. Include window fixed effects when more than one window level is available.
<code>include_interaction</code>	Logical. Include the condition-by-window interaction when both condition and window are used.
<code>random_intercept</code>	Logical. Include a subject random intercept for LMM model types when feasible.
<code>random_window_slopes</code>	Logical. Attempt subject-level random window slopes for LMM model types when feasible.
<code>fallback_on_singular</code>	Logical. If TRUE, LMM model types may fall back from random-window-slope models to random-intercept models when needed.
<code>REML</code>	Logical. Passed to <code>lme4::lmer()</code> .
<code>optimizer</code>	Optimizer passed to <code>lme4::lmerControl()</code> .

maxfun	Maximum optimizer iterations passed to <code>lme4::lmerControl()</code> .
drop_missing	Logical. If TRUE, rows with missing or non-finite model inputs are removed before fitting.
...	Additional arguments passed to <code>fit_gazepoint_pupil_window_lmm()</code> .

Value

A list containing fitted models, formulas, fixed effects, a comparison table, settings, and model-status information.

```
fit_gazepoint_transition_count_nb_sensitivity
```

Fit optional negative-binomial transition-count sensitivity models

Description

Fit an optional negative-binomial sensitivity model for AOI/state transition counts using `glmmTMB` when it is installed. This helper is intended as a publication sensitivity branch for overdispersed transition-count outcomes.

Usage

```
fit_gazepoint_transition_count_nb_sensitivity(
  data,
  count_col = NULL,
  from_col = NULL,
  to_col = NULL,
  condition_cols = NULL,
  random_effect_cols = NULL,
  exposure_col = NULL,
  offset_col = NULL,
  formula = NULL,
  family = c("nbinom2", "nbinom1"),
  zero_inflation = FALSE,
  ziformula = NULL,
  dispformula = NULL,
  control = NULL,
  name = "gazepoint_transition_count_nb_sensitivity"
)
```

Arguments

data	A data frame containing transition-count rows.
count_col	Transition-count outcome column. If NULL, common count columns are detected automatically.

from_col	Transition origin column. If NULL, common origin columns are detected automatically.
to_col	Transition destination column. If NULL, common destination columns are detected automatically.
condition_cols	Optional fixed-effect condition columns.
random_effect_cols	Optional random-intercept grouping columns.
exposure_col	Optional positive exposure column. If supplied, the model includes <code>offset(log(exposure_col))</code> .
offset_col	Optional numeric offset column. Use either <code>exposure_col</code> or <code>offset_col</code> , not both.
formula	Optional model formula. If NULL, a formula is constructed from transition origin, destination, condition columns, optional offset, and random intercepts.
family	Negative-binomial family. Options are "nbinom2" and "nbinom1".
zero_inflation	Logical. If TRUE, use <code>ziformula = ~1</code> unless <code>ziformula</code> is supplied.
ziformula	Optional zero-inflation formula passed to <code>glmmTMB</code> .
dispformula	Optional dispersion formula passed to <code>glmmTMB</code> .
control	Optional control object passed to <code>glmmTMB</code> .
name	Character label stored in the returned object.

Details

The helper keeps `glmmTMB` optional. If `glmmTMB` is not installed, it returns a structured skipped object rather than failing.

Value

A list with class `gp3_transition_count_nb_sensitivity`.

flag_gazepoint_pupil *Flag invalid, missing, implausible, and outlying Gazepoint pupil samples*

Description

Adds pupil-quality flags to a Gazepoint master sample-level table created by `as_gazepoint_master()` or `create_gazepoint_master()`. This function is intended as a preprocessing step before interpolation, filtering, baseline correction, or pupil-based modelling.

Usage

```
flag_gazepoint_pupil(
  master,
  pupil_col = NULL,
  time_col = NULL,
  missing_pupil_col = NULL,
  group_cols = c("subject", "media_id"),
  min_pupil = 0,
  max_pupil = Inf,
  outlier_k = 1.5,
  flag_iqr_outliers = TRUE
)
```

Arguments

master	A Gazepoint master sample-level table.
pupil_col	Optional name of the pupil column to flag. If NULL, the function detects one of mean_pupil, pupil, pupil_raw, left_pupil, or right_pupil.
time_col	Optional name of the time column. If NULL, the function detects one of time_ms, time, time_orig, or time_orig_ms.
missing_pupil_col	Optional name of the missing-pupil flag column. If NULL, the function uses missing_pupil when available.
group_cols	Character vector of grouping columns used for IQR-based outlier detection. Defaults to c("subject", "media_id") using internally standardised names. Use character(0) for global outlier detection.
min_pupil	Minimum plausible pupil value. Defaults to 0.
max_pupil	Maximum plausible pupil value. Defaults to Inf. Use narrower values, such as 1 and 9, only when the pupil column is known to be measured in millimetres.
outlier_k	Multiplier for IQR-based outlier detection. Defaults to 1.5.
flag_iqr_outliers	Logical. If TRUE, IQR-based outliers are flagged. Defaults to TRUE.

Value

A tibble containing the original master table plus pupil-flagging columns.

Examples

```
master <- gazepoint_example_master
flagged <- flag_gazepoint_pupil(master)

dplyr::count(flagged, pupil_flag_reason)
```

 flag_gazepoint_pupil_artifacts

Flag Gazepoint pupil artifacts before interpolation

Description

Adds pupil-specific artifact flags for blink/trackloss contamination, physiological implausibility when pupil units are millimetres, pupil-speed outliers, left-right binocular pupil disagreement, and temporal padding around bad samples. The function preserves raw pupil columns and creates pupil_clean, which can be used as input for interpolation.

Usage

```
flag_gazepoint_pupil_artifacts(
  data,
  pupil_col = NULL,
  left_pupil_col = NULL,
  right_pupil_col = NULL,
  time_col = NULL,
  blink_col = NULL,
  trackloss_col = NULL,
  missing_pupil_col = NULL,
  pupil_unit_col = NULL,
  group_cols = c("subject", "media_id"),
  registry = NULL,
  blink_padding_pre_ms = NULL,
  blink_padding_post_ms = NULL,
  pupil_min_mm = NULL,
  pupil_max_mm = NULL,
  pupil_speed_mad_k = NULL,
  binocular_mad_k = NULL,
  max_physio_outlier_prop = 0.8,
  flag_speed_outliers = TRUE,
  flag_binocular_disagreement = TRUE,
  flag_physiological_outliers = TRUE
)
```

Arguments

data	A Gazepoint master table or pupil-processing table.
pupil_col	Optional name of the pupil column to clean. If NULL, the function detects one of mean_pupil, pupil_raw, pupil, left_pupil, or right_pupil.
left_pupil_col	Optional left-pupil column. If NULL, left_pupil is used when available.
right_pupil_col	Optional right-pupil column. If NULL, right_pupil is used when available.

time_col	Optional time column. If NULL, the function detects one of time_ms, time, time_orig, or time_orig_ms.
blink_col	Optional blink column. If NULL, blink is used when available.
trackloss_col	Optional trackloss column. If NULL, one of trackloss or Trackloss is used when available.
missing_pupil_col	Optional missing-pupil column. If NULL, missing_pupil is used when available.
pupil_unit_col	Optional pupil-unit column. If NULL, pupil_unit is used when available.
group_cols	Character vector of grouping columns used for speed outlier detection and artifact-padding windows. Defaults to c("subject", "media_id"). Use character(0) for global processing.
registry	Optional preprocessing registry created by create_gazepoint_preprocessing_registry() .
blink_padding_pre_ms	Padding before bad samples, in milliseconds. If NULL, taken from registry or defaults to 100.
blink_padding_post_ms	Padding after bad samples, in milliseconds. If NULL, taken from registry or defaults to 100.
pupil_min_mm	Minimum plausible pupil value when units are millimetres. If NULL, taken from registry or defaults to 1.
pupil_max_mm	Maximum plausible pupil value when units are millimetres. If NULL, taken from registry or defaults to 9.
pupil_speed_mad_k	MAD multiplier for pupil-speed outlier detection. If NULL, taken from registry or defaults to 6.
binocular_mad_k	MAD multiplier for binocular-disagreement detection. If NULL, taken from registry or defaults to 6.
max_physio_outlier_prop	Maximum allowed proportion of non-missing millimetre-labelled pupil samples that may be rejected by the physiological rule before the rule is automatically suppressed. Defaults to 0.80. This prevents raw-unit Gazepoint exports from being silently erased when the unit label suggests millimetres but the numeric scale is not compatible with ordinary 1–9 mm thresholds.
flag_speed_outliers	Logical. If TRUE, pupil-speed outliers are flagged. Defaults to TRUE.
flag_binocular_disagreement	Logical. If TRUE, left-right pupil disagreement is flagged when both eyes are available. Defaults to TRUE.
flag_physiological_outliers	Logical. If TRUE, millimetre-based physiological thresholds are applied only when the pupil unit is identified as millimetres. Defaults to TRUE.

Value

A tibble containing the original data plus pupil-artifact columns.

Examples

```
master <- gazepoint_example_master
registry <- create_gazepoint_preprocessing_registry()

artifact_pupil <- flag_gazepoint_pupil_artifacts(
  master,
  registry = registry
)

dplyr::count(artifact_pupil, pupil_artifact_reason)
```

```
flag_gazepoint_pupil_hampel
```

Flag pupil artifacts with a Hampel filter

Description

Apply a rolling Hampel filter to a Gazepoint pupil column. The helper computes a rolling median and median absolute deviation (MAD) within a centred sample window, then flags pupil samples whose absolute deviation from the local median exceeds $k * MAD$.

Usage

```
flag_gazepoint_pupil_hampel(
  data,
  pupil_col,
  time_col = NULL,
  grouping_cols = NULL,
  window_size_samples = 7L,
  k = 3,
  min_valid_samples = 3L,
  scale_mad = 1.4826,
  flag_col = "pupil_hampel_outlier",
  median_col = "pupil_hampel_median",
  mad_col = "pupil_hampel_mad",
  threshold_col = "pupil_hampel_threshold",
  corrected_col = NULL,
  status_col = "pupil_hampel_status",
  overwrite = FALSE,
  name = "gazepoint_pupil_hampel"
)
```

Arguments

data A data frame containing pupil observations.

pupil_col	Pupil column to screen.
time_col	Optional time column used to order samples within groups.
grouping_cols	Optional grouping columns, for example participant and trial.
window_size_samples	Odd integer rolling-window size in samples.
k	Hampel threshold multiplier.
min_valid_samples	Minimum finite pupil samples required inside a rolling window.
scale_mad	Scaling factor applied to MAD. The default 1.4826 makes MAD comparable to the standard deviation under normality.
flag_col	Name of the logical Hampel-flag output column.
median_col	Name of the rolling median output column.
mad_col	Name of the rolling MAD output column.
threshold_col	Name of the rolling threshold output column.
corrected_col	Optional name of a corrected pupil column. If supplied, flagged samples are replaced with the local rolling median.
status_col	Name of the row-level Hampel status column.
overwrite	Logical. If FALSE, the function errors when output columns already exist.
name	Character label stored in object attributes.

Details

This helper is intended as an optional sensitivity/artifact-flagging branch. It complements existing pupil artifact checks and should not automatically replace confirmatory preprocessing decisions.

Value

A tibble with Hampel-filter columns added. The object has class `gp3_pupil_hampel_flags`.

flag_tracking_quality *Flag low-quality Gazeport recordings*

Description

Combines tracking-quality and sampling-rate summaries and flags rows with low gaze validity, low pupil validity, abnormal sampling rate, or short duration.

Usage

```
flag_tracking_quality(
  quality,
  sampling,
  by = c("USER_FILE", "MEDIA_ID"),
  min_gaze_valid_pct = 70,
  min_pupil_valid_pct = 70,
  expected_hz = 60,
  hz_tolerance = 5,
  min_duration_sec = NULL
)
```

Arguments

quality	Tracking-quality table from summarise_tracking_quality().
sampling	Sampling-rate table from check_sampling_rate().
by	Columns used to join quality and sampling.
min_gaze_valid_pct	Minimum acceptable FPOGV validity percentage.
min_pupil_valid_pct	Minimum acceptable pupil validity percentage.
expected_hz	Expected sampling rate.
hz_tolerance	Allowed deviation from the expected sampling rate.
min_duration_sec	Minimum acceptable recording duration in seconds.

Value

A tibble with quality, sampling, flag columns, and an overall review flag.

gazepoint_example_aoi_geometry
Example AOI geometry table

Description

A lightweight synthetic AOI geometry table for AOI-verification examples. Coordinates are normalised to a 0–1 screen coordinate system.

Usage

```
gazepoint_example_aoi_geometry
```

Format

A tibble with one row per stimulus and AOI, including:

media_id Synthetic stimulus identifier.

aoi Synthetic AOI label.

x_min, y_min, x_max, y_max Normalised rectangular AOI boundaries.

Examples

```
data(gazeptoint_example_aoi_geometry)
gazeptoint_example_aoi_geometry
```

gazeptoint_example_aoi_windows

Example AOI-window summary table

Description

A lightweight synthetic AOI-window summary table created from gazeptoint_example_master. It can be used in examples for AOI-window denominator checks, GLMM preparation, and AOI-window modelling.

Usage

```
gazeptoint_example_aoi_windows
```

Format

A tibble with one row per participant, stimulus/trial, and AOI time window.

Examples

```
data(gazeptoint_example_aoi_windows)
head(gazeptoint_example_aoi_windows)
```

gazeport_example_fixations

Example Gazeport fixation table

Description

A lightweight synthetic Gazeport-style fixation table for examples, tests, README workflows, and vignettes. The data are artificial and are not from a real participant study.

Usage

```
gazeport_example_fixations
```

Format

A tibble with fixation-level rows and columns including:

USER_FILE Synthetic participant/file identifier.

subject Synthetic participant identifier.

MEDIA_ID Synthetic stimulus identifier.

trial_global Synthetic trial identifier.

condition Synthetic experimental condition.

FPOGID Synthetic fixation identifier.

FPOGS Synthetic fixation start time.

FPOGD Synthetic fixation duration.

FPOGX, FPOGY Synthetic fixation coordinates.

FPOGV Synthetic fixation validity flag.

AOI Synthetic AOI label.

Examples

```
data(gazeport_example_fixations)
head(gazeport_example_fixations)
```

`gazeport_example_master`*Example Gazeport master table*

Description

A lightweight synthetic Gazeport-style sample-level master table for examples, tests, README workflows, and vignettes. The data are artificial and are not from a real participant study.

Usage

```
gazeport_example_master
```

Format

A tibble with sample-level rows and columns including:

subject Synthetic participant identifier.

USER_FILE Gazeport-style participant/file identifier.

MEDIA_ID Synthetic stimulus identifier.

trial_global Synthetic trial identifier.

condition Synthetic experimental condition.

time Sample time in milliseconds.

x, y Normalised gaze coordinates.

pupil Synthetic pupil value.

valid Logical gaze/pupil validity flag.

artifact Logical synthetic pupil-artifact flag.

aoi_current Synthetic AOI state.

is_fixation, is_saccade Synthetic fixation and saccade indicators.

event_label Synthetic event marker.

Examples

```
data(gazeport_example_master)
head(gazeport_example_master)
```

```
gazepoint_example_pupil_windows
```

Example pupil-window summary table

Description

A lightweight synthetic pupil-window summary table created from `gazepoint_example_master`. It can be used in examples for pupil-window model-data preparation and confirmatory pupil-window modelling.

Usage

```
gazepoint_example_pupil_windows
```

Format

A tibble with one row per participant, stimulus/trial, condition, and pupil time window.

Examples

```
data(gazepoint_example_pupil_windows)
head(gazepoint_example_pupil_windows)
```

```
inspect_gazepoint_columns
```

Inspect Gazepoint columns

Description

Inspect Gazepoint columns

Usage

```
inspect_gazepoint_columns(x)
```

Arguments

`x` A data frame or path to a Gazepoint CSV export.

Value

A tibble describing column names, semantic groups, and missingness.

interpolate_gazepoint_pupil

Interpolate short missing gaps in Gazepoint pupil data

Description

Performs linear interpolation over short internal gaps in Gazepoint pupil data. This function is intended to be used after `flag_gazepoint_pupil()` or `flag_gazepoint_pupil_artifacts()`. When available, `pupil_clean` is used as the preferred default input column, followed by `pupil_for_preprocessing`. Leading gaps, trailing gaps, long gaps, non-finite time values, and groups with too few valid pupil samples are not interpolated.

Usage

```
interpolate_gazepoint_pupil(
  data,
  pupil_col = NULL,
  time_col = NULL,
  group_cols = c("subject", "media_id"),
  max_gap_ms = 150,
  max_gap_samples = Inf,
  min_valid_points = 2
)
```

Arguments

<code>data</code>	A Gazepoint master table, preferably after <code>flag_gazepoint_pupil()</code> or <code>flag_gazepoint_pupil_artifacts()</code> .
<code>pupil_col</code>	Optional name of the pupil column to interpolate. If NULL, the function detects one of <code>pupil_clean</code> , <code>pupil_for_preprocessing</code> , <code>mean_pupil</code> , <code>pupil</code> , <code>pupil_raw</code> , <code>left_pupil</code> , or <code>right_pupil</code> .
<code>time_col</code>	Optional name of the time column. If NULL, the function detects one of <code>time_ms</code> , <code>time</code> , <code>time_orig</code> , or <code>time_orig_ms</code> .
<code>group_cols</code>	Character vector of grouping columns used to keep interpolation within independent time series. Defaults to <code>c("subject", "media_id")</code> using internally standardised names. Use <code>character(0)</code> for global interpolation.
<code>max_gap_ms</code>	Maximum duration, in milliseconds, of a gap that may be interpolated. The duration is measured between the valid samples immediately before and after the gap. Defaults to 150.
<code>max_gap_samples</code>	Maximum number of consecutive missing samples that may be interpolated. Defaults to Inf.
<code>min_valid_points</code>	Minimum number of valid samples required within a group before interpolation is attempted. Defaults to 2.

Value

A tibble containing the original data plus interpolation columns.

Examples

```
master <- gazepoint_example_master
flagged <- flag_gazepoint_pupil(master)

interpolated <- interpolate_gazepoint_pupil(flagged)

dplyr::count(interpolated, pupil_interpolation_status)

artifact_flagged <- flag_gazepoint_pupil_artifacts(master)

artifact_interpolated <- interpolate_gazepoint_pupil(artifact_flagged)

dplyr::count(artifact_interpolated, pupil_interpolation_status)
```

interpolate_gazepoint_pupil_pchip

Interpolate Gazepoint pupil data using PCHIP

Description

Apply shape-preserving piecewise cubic Hermite interpolation to short internal gaps in Gazepoint pupil time series. This helper is intended as a sensitivity branch alongside the default linear interpolation workflow. It does not overwrite the original pupil column.

Usage

```
interpolate_gazepoint_pupil_pchip(
  data,
  pupil_col = NULL,
  time_col = NULL,
  grouping_cols = NULL,
  max_gap_ms = 500,
  max_gap_samples = NULL,
  min_valid_points = 3,
  output_col = "pupil_interpolated_pchip",
  flag_col = "interpolated_pupil_pchip",
  status_col = "pchip_interpolation_status"
)
```

Arguments

data	A data frame containing pupil time-series data.
pupil_col	Name of the pupil column to interpolate. If NULL, common processed pupil columns are detected automatically.
time_col	Name of the time column. If NULL, common time columns are detected automatically.
grouping_cols	Optional character vector of grouping columns. If NULL, common participant/trial/media grouping columns are detected automatically. Use character(0) to interpolate the full data as one sequence.
max_gap_ms	Maximum internal gap duration, in milliseconds, eligible for interpolation. If both max_gap_ms and max_gap_samples are supplied, both criteria must be satisfied.
max_gap_samples	Maximum number of consecutive missing samples eligible for interpolation.
min_valid_points	Minimum number of valid non-missing points required within a group before PCHIP interpolation is attempted.
output_col	Name of the interpolated pupil output column.
flag_col	Name of the logical flag column indicating samples filled by PCHIP interpolation.
status_col	Name of the interpolation-status column.

Value

A tibble with PCHIP interpolation columns added.

plot_gazepoint_aoi_gamm

Plot AOI time-course GAMM results

Description

Plot observed AOI target-looking proportions and fitted GAMM trajectories from a model returned by `fit_gazepoint_aoi_gamm()`.

Usage

```
plot_gazepoint_aoi_gamm(
  fit,
  n_time_points = 100,
  include_observed = TRUE,
  include_fitted = TRUE,
  show_ci = TRUE,
  ci_level = 0.95,
```

```

exclude_random_effects = TRUE,
observed_summary = c("pooled"),
point_size = 1.8,
point_alpha = 0.65,
line_width = 0.8,
ribbon_alpha = 0.15,
title = NULL,
subtitle = NULL,
x_label = "Time (ms)",
y_label = "Target AOI looking probability",
y_limits = c(0, 1)
)

```

Arguments

<code>fit</code>	A result object returned by <code>fit_gazepoint_aoi_gamm()</code> .
<code>n_time_points</code>	Number of time points used for the fitted prediction grid. If <code>NULL</code> , the observed time bins are used.
<code>include_observed</code>	Logical. If <code>TRUE</code> , plot observed binned proportions.
<code>include_fitted</code>	Logical. If <code>TRUE</code> , plot fitted GAMM trajectories.
<code>show_ci</code>	Logical. If <code>TRUE</code> , plot fitted confidence intervals.
<code>ci_level</code>	Confidence level for fitted intervals.
<code>exclude_random_effects</code>	Logical. If <code>TRUE</code> , exclude subject random-effect smooths from fitted predictions.
<code>observed_summary</code>	Character. Currently "pooled" pools successes and denominators by condition and time.
<code>point_size</code>	Size of observed points.
<code>point_alpha</code>	Transparency for observed points.
<code>line_width</code>	Width of fitted trajectory lines.
<code>ribbon_alpha</code>	Transparency for fitted confidence ribbons.
<code>title</code>	Optional plot title.
<code>subtitle</code>	Optional plot subtitle.
<code>x_label</code>	X-axis label.
<code>y_label</code>	Y-axis label.
<code>y_limits</code>	Optional numeric vector of length 2 for y-axis limits.

Details

The plot supports single-condition fallback models and multi-condition AOI time-course GAMMs. By default, fitted trajectories are population-level predictions with subject random-effect smooths excluded.

Value

A `ggplot` object with prediction and observed data stored as attributes.

plot_gazepoint_aoi_transition_matrix

Plot Gazepoint AOI transition matrix

Description

Plot a heatmap of AOI transition counts or probabilities from the output of `compute_gazepoint_aoi_transition_matrix()` or from a compatible long-form transition table.

Usage

```
plot_gazepoint_aoi_transition_matrix(
  transitions,
  value = c("prob", "n"),
  state_order = NULL,
  by_cols = NULL,
  include_zero = TRUE,
  show_labels = TRUE,
  label_digits = 2,
  label_size = 3,
  facet = TRUE,
  title = NULL
)
```

Arguments

<code>transitions</code>	A <code>gp3_aoi_transition_matrix</code> object, a long-form transition table with <code>from</code> , <code>to</code> , <code>n</code> , and/or <code>prob</code> columns, or a numeric matrix with AOI states as row and column names.
<code>value</code>	Which value to plot: "prob" for transition probabilities or "n" for transition counts.
<code>state_order</code>	Optional character vector defining the AOI order on the heatmap axes.
<code>by_cols</code>	Optional character vector of grouping columns to facet by. If <code>NULL</code> , the function uses grouping columns stored in a <code>gp3_aoi_transition_matrix</code> object, when available.
<code>include_zero</code>	Logical. If <code>TRUE</code> , all possible state-to-state cells are shown, with missing transitions displayed as zero.
<code>show_labels</code>	Logical. If <code>TRUE</code> , cell values are printed inside tiles.
<code>label_digits</code>	Number of digits used when labelling probabilities.
<code>label_size</code>	Text size for cell labels.
<code>facet</code>	Logical. If <code>TRUE</code> , grouped transition tables are faceted.
<code>title</code>	Optional plot title.

Value

A `ggplot2` plot object.

`plot_gazepoint_aoi_verification`*Plot AOI geometry for visual verification*

Description

Create a visual verification plot of AOI rectangles, with optional gaze samples overlaid.

Usage

```
plot_gazepoint_aoi_verification(  
  aoi_geometry,  
  gaze_data = NULL,  
  geometry_aoi_col = NULL,  
  geometry_stimulus_col = NULL,  
  x_min_col = NULL,  
  y_min_col = NULL,  
  x_max_col = NULL,  
  y_max_col = NULL,  
  x_col = NULL,  
  y_col = NULL,  
  width_col = NULL,  
  height_col = NULL,  
  gaze_x_col = NULL,  
  gaze_y_col = NULL,  
  gaze_stimulus_col = NULL,  
  screen_x_range = c(0, 1),  
  screen_y_range = c(0, 1),  
  facet_by_stimulus = TRUE,  
  show_labels = TRUE,  
  show_gaze = TRUE,  
  invert_y = TRUE,  
  point_alpha = 0.35,  
  point_size = 1.2,  
  line_width = 0.8,  
  label_size = 3  
)
```

Arguments

<code>aoi_geometry</code>	A data frame containing AOI geometry definitions.
<code>gaze_data</code>	Optional data frame containing gaze samples to overlay.
<code>geometry_aoi_col</code>	AOI label/name column in <code>aoi_geometry</code> .
<code>geometry_stimulus_col</code>	Optional stimulus/media column in <code>aoi_geometry</code> .

x_min_col	Optional AOI left/x-min column.
y_min_col	Optional AOI top/y-min column.
x_max_col	Optional AOI right/x-max column.
y_max_col	Optional AOI bottom/y-max column.
x_col	Optional AOI left/x column used with width_col.
y_col	Optional AOI top/y column used with height_col.
width_col	Optional AOI width column.
height_col	Optional AOI height column.
gaze_x_col	Optional gaze x-coordinate column.
gaze_y_col	Optional gaze y-coordinate column.
gaze_stimulus_col	Optional gaze stimulus/media column.
screen_x_range	Numeric length-2 vector defining the screen x range.
screen_y_range	Numeric length-2 vector defining the screen y range.
facet_by_stimulus	Logical. If TRUE, facet by stimulus/media when a stimulus column is available.
show_labels	Logical. If TRUE, draw AOI labels at AOI centres.
show_gaze	Logical. If TRUE, overlay gaze samples when gaze_data is supplied.
invert_y	Logical. If TRUE, reverse the y-axis so screen origin is at the top-left.
point_alpha	Alpha transparency for gaze points.
point_size	Size of gaze points.
line_width	Width of AOI rectangle borders.
label_size	Size of AOI labels.

Value

A ggplot object.

plot_gazepoint_cluster_results

Plot cluster-based permutation results

Description

Create a publication-ready time-course plot from the output of `run_gazepoint_cluster_permutation()`. The plot can show the mean condition difference, the time-wise test statistic, or both. Candidate time bins and cluster-level significant windows can be highlighted.

Usage

```
plot_gazepoint_cluster_results(
  result,
  plot_type = c("both", "difference", "statistic"),
  alpha = 0.05,
  significant_only = TRUE,
  show_clusters = TRUE,
  show_candidates = TRUE,
  show_threshold = TRUE,
  show_zero_line = TRUE,
  title = NULL,
  subtitle = NULL,
  x_label = "Time (ms)",
  y_label = NULL,
  line_width = 0.7,
  point_size = 1.8,
  cluster_alpha = 0.12
)
```

Arguments

result	A result object returned by run_gazepoint_cluster_permutation().
plot_type	Character. One of "both", "difference", or "statistic".
alpha	Cluster-level significance threshold used to decide which clusters are significant for plotting.
significant_only	Logical. If TRUE, only significant clusters are shaded. If FALSE, all observed clusters are shaded.
show_clusters	Logical. If TRUE, shade cluster windows.
show_candidates	Logical. If TRUE, mark time bins exceeding the cluster-forming threshold.
show_threshold	Logical. If TRUE, show the cluster-forming threshold on the statistic panel.
show_zero_line	Logical. If TRUE, add a horizontal zero reference line.
title	Optional plot title.
subtitle	Optional plot subtitle.
x_label	X-axis label.
y_label	Optional y-axis label. If NULL, a label is chosen automatically.
line_width	Width of the time-course line.
point_size	Size of candidate-bin points.
cluster_alpha	Transparency for shaded cluster windows.

Details

Cluster-based permutation tests are intended for time-course inference. They should not be used to discover a confirmatory time window and then test that same window again in a second confirmatory model.

Value

A ggplot object.

plot_gazepoint_gca *Plot observed and fitted Growth Curve Analysis trajectories*

Description

Plot observed and fitted pupil trajectories from a `gp3_gca_model` object. The function aggregates observed and fitted values by condition and time, and returns a `ggplot2` object.

Usage

```
plot_gazepoint_gca(
  model,
  data = NULL,
  time_col = "gca_time",
  observed_col = "gca_pupil",
  fitted_col = "gca_fitted",
  condition_col = "condition",
  subject_col = "subject",
  summarise = TRUE,
  show_observed = TRUE,
  show_fitted = TRUE,
  show_subjects = FALSE,
  interval = TRUE,
  title = NULL,
  point_size = 1.6,
  line_width = 0.8,
  alpha = 0.75
)
```

Arguments

<code>model</code>	A fitted object returned by <code>fit_gazepoint_gca()</code> , or a data frame containing observed and fitted values.
<code>data</code>	Optional data frame. If <code>NULL</code> and <code>model</code> is a <code>gp3_gca_model</code> , the model data are used.
<code>time_col</code>	Name of the time column.
<code>observed_col</code>	Name of the observed outcome column.
<code>fitted_col</code>	Name of the fitted-value column. If unavailable and <code>model</code> is a <code>gp3_gca_model</code> , fitted values are computed from the model.
<code>condition_col</code>	Name of the condition column.
<code>subject_col</code>	Optional subject column, used only when <code>show_subjects = TRUE</code> .

summarise	Logical. If TRUE, plot mean trajectories by condition and time. If FALSE, plot row-level values.
show_observed	Logical. If TRUE, include observed trajectory.
show_fitted	Logical. If TRUE, include fitted trajectory.
show_subjects	Logical. If TRUE, add faint subject-level observed trajectories when a subject column is available.
interval	Logical. If TRUE, add a standard-error ribbon around the observed mean trajectory when summarise = TRUE.
title	Optional plot title.
point_size	Point size for observed means.
line_width	Line width for trajectories.
alpha	Alpha value for observed points/lines.

Value

A ggplot2 object.

plot_gazepoint_model_predictions

Plot observed summaries and model-implied predictions

Description

Create a reporting plot that overlays observed outcome summaries with fitted/model-predicted trajectories. The helper is intentionally generic and can be used with linear models, GLMs, mixed models, GAMMs, GCA-style models, AOI GLMMs, and pupil LMMs when the fitted object supports predict().

Usage

```
plot_gazepoint_model_predictions(
  data,
  model = NULL,
  x_col,
  outcome_col,
  condition_col = NULL,
  group_cols = NULL,
  facet_cols = NULL,
  newdata = NULL,
  prediction_type = c("response", "link"),
  include_random_effects = FALSE,
  observed_summary_function = c("mean", "median"),
  ci = 0.95,
  show_observed = TRUE,
  show_observed_ci = TRUE,
```

```

  show_predictions = TRUE,
  show_prediction_ci = TRUE,
  point_alpha = 0.55,
  line_width = 1,
  name = "gazepoint_model_predictions"
)

```

Arguments

<code>data</code>	A data frame containing the observed data.
<code>model</code>	Optional fitted model object. If supplied, predictions are computed using <code>predict()</code> .
<code>x_col</code>	Column used on the x-axis, usually time or time bin.
<code>outcome_col</code>	Observed outcome column.
<code>condition_col</code>	Optional condition column used for colour/grouping.
<code>group_cols</code>	Optional additional grouping columns for observed and predicted trajectories.
<code>facet_cols</code>	Optional columns used for faceting.
<code>newdata</code>	Optional prediction grid. If NULL, predictions are computed on data and then summarised by <code>x/group/facet</code> .
<code>prediction_type</code>	Prediction scale passed to <code>predict()</code> . Common values are "response" and "link".
<code>include_random_effects</code>	Logical. For lme4 mixed models, FALSE requests population-level predictions via <code>re.form = NA</code> ; TRUE includes conditional random effects where possible.
<code>observed_summary_function</code>	Summary for observed outcomes. Options are "mean" and "median".
<code>ci</code>	Confidence level for observed and prediction intervals when standard errors are available.
<code>show_observed</code>	Logical. Plot observed summaries.
<code>show_observed_ci</code>	Logical. Plot observed normal-approximation intervals.
<code>show_predictions</code>	Logical. Plot model predictions when <code>model</code> is supplied.
<code>show_prediction_ci</code>	Logical. Plot prediction intervals when standard errors are available from <code>predict()</code> .
<code>point_alpha</code>	Alpha value for observed points.
<code>line_width</code>	Line width for prediction trajectories.
<code>name</code>	Character label stored in plot attributes.

Value

A ggplot object with attributes containing the observed summary, prediction summary, overview, and settings.

 plot_gazepoint_multiverse_results

Plot Gazepoint preprocessing multiverse results

Description

Create diagnostic plots from preprocessing multiverse summaries or from pupil/AOI multiverse result objects.

Usage

```
plot_gazepoint_multiverse_results(
  x,
  plot = c("status", "rows", "pupil_parameters", "aoi_denominators"),
  family = c("all", "pupil", "aoi"),
  title = NULL,
  show_labels = TRUE
)
```

Arguments

x	A gp3_multiverse_summary_results, gp3_pupil_multiverse_results, or gp3_aoi_multiverse_results object.
plot	Character. Plot type. One of "status", "rows", "pupil_parameters", or "aoi_denominators".
family	Character. Which family to show. One of "all", "pupil", or "aoi".
title	Optional plot title.
show_labels	Logical. If TRUE, show branch labels on the y-axis.

Value

A ggplot object.

 plot_gazepoint_pupil_preprocessing

Plot Gazepoint pupil preprocessing for one trial

Description

Create a visual audit plot for one selected subject, media item, trial, or trial-global identifier. The plot can show raw pupil, cleaned pupil, interpolated pupil, baseline-corrected pupil, smoothed pupil, and artifact flags.

Usage

```

plot_gazepoint_pupil_preprocessing(
  data,
  subject = NULL,
  media_id = NULL,
  trial = NULL,
  trial_global = NULL,
  condition = NULL,
  subject_col = "subject",
  media_col = "MEDIA_ID",
  trial_col = "trial",
  trial_global_col = "trial_global",
  condition_col = "condition",
  time_col = "time",
  raw_pupil_col = "pupil",
  clean_pupil_col = "pupil_clean",
  interpolated_pupil_col = "pupil_interpolated",
  baseline_pupil_col = "pupil_baseline_corrected",
  smoothed_pupil_col = "pupil_smoothed",
  artifact_col = NULL,
  artifact_reason_col = NULL,
  status_col = "pupil_interpolation_status",
  plot_style = c("faceted", "overlaid"),
  bin_width_ms = 50,
  max_event_marks = 150,
  point_size = 0.8,
  line_width = 0.35,
  alpha = 0.95
)

```

Arguments

<code>data</code>	A Gazepoint pupil data frame.
<code>subject</code>	Optional subject value to filter.
<code>media_id</code>	Optional media identifier value to filter.
<code>trial</code>	Optional trial value to filter.
<code>trial_global</code>	Optional global trial identifier value to filter.
<code>condition</code>	Optional condition value to filter.
<code>subject_col</code>	Name of the subject column.
<code>media_col</code>	Name of the media identifier column.
<code>trial_col</code>	Name of the trial column.
<code>trial_global_col</code>	Name of the global trial identifier column.
<code>condition_col</code>	Name of the condition column.
<code>time_col</code>	Name of the time column.

raw_pupil_col	Optional raw pupil column.
clean_pupil_col	Optional cleaned pupil column.
interpolated_pupil_col	Optional interpolated pupil column.
baseline_pupil_col	Optional baseline-corrected pupil column.
smoothed_pupil_col	Optional smoothed pupil column.
artifact_col	Optional artifact flag column. If NULL, the function tries pupil_artifact_flag, pupil_flag_invalid, and artifact_flag.
artifact_reason_col	Optional artifact-reason column. If NULL, the function tries pupil_artifact_reason, pupil_flag_reason, and artifact_reason.
status_col	Optional interpolation-status column.
plot_style	Either "faceted" or "overlaid".
bin_width_ms	Width of time bins in milliseconds. This is used only for visual smoothing of dense sample-level traces.
max_event_marks	Maximum number of artifact/interpolation rug marks to draw. Event marks are evenly thinned if there are more events.
point_size	Size control for artifact/interpolation rug marks.
line_width	Line width for pupil series.
alpha	Line and marker transparency.

Value

A ggplot2 plot object.

plot_gazepoint_pupil_status

Plot Gazepoint pupil preprocessing status

Description

Visualise observed, interpolated, missing, artifact, and other pupil-sample statuses over time or as grouped percentages.

Usage

```
plot_gazepoint_pupil_status(
  data,
  time_col = "time",
  pupil_col = NULL,
  status_col = "pupil_interpolation_status",
  interpolated_col = "pupil_was_interpolated",
  artifact_col = NULL,
  artifact_reason_col = NULL,
  group_cols = c("subject", "trial_global"),
  facet_cols = NULL,
  plot_type = c("timeline", "summary"),
  point_size = 0.7,
  alpha = 0.8,
  max_points = 50000
)
```

Arguments

<code>data</code>	A Gazepoint pupil data frame.
<code>time_col</code>	Name of the time column.
<code>pupil_col</code>	Optional pupil column used to detect remaining missing samples. If NULL, the function tries <code>pupil_smoothed</code> , <code>pupil_baseline_corrected</code> , <code>pupil_interpolated</code> , <code>pupil_clean</code> , and <code>pupil</code> .
<code>status_col</code>	Optional interpolation-status column.
<code>interpolated_col</code>	Optional logical interpolation flag column.
<code>artifact_col</code>	Optional artifact flag column. If NULL, the function tries <code>pupil_artifact_flag</code> , <code>pupil_flag_invalid</code> , and <code>artifact_flag</code> .
<code>artifact_reason_col</code>	Optional artifact-reason column. If NULL, the function tries <code>pupil_artifact_reason</code> , <code>pupil_flag_reason</code> , and <code>artifact_reason</code> .
<code>group_cols</code>	Character vector used to define timeline rows or summary groups.
<code>facet_cols</code>	Optional character vector of columns used for faceting.
<code>plot_type</code>	Either "timeline" or "summary".
<code>point_size</code>	Point size for timeline plots.
<code>alpha</code>	Point/column transparency.
<code>max_points</code>	Maximum number of rows to plot in timeline mode. If the input has more rows, rows are evenly thinned for plotting only.

Value

A ggplot2 plot object.

 plot_gazepoint_pupil_timecourse

Plot Gazepoint pupil time course

Description

Plot a binned pupil time course with a mean line and confidence band. The function can plot one overall time course or condition-wise time courses, with optional faceting by variables such as condition, media, AOI, subject, or trial.

Usage

```
plot_gazepoint_pupil_timecourse(
  data,
  pupil_col = NULL,
  time_col = "time",
  condition_col = "condition",
  facet_cols = NULL,
  bin_width_ms = 100,
  ci_level = 0.95,
  min_samples = 1,
  band_alpha = 0.2,
  line_width = 0.8
)
```

Arguments

data	A Gazepoint pupil data frame.
pupil_col	Name of the pupil column to plot. If NULL, the function tries pupil_smoothed, pupil_baseline_corrected, pupil_baseline_percent_change, pupil_interpolated, pupil_clean, and pupil.
time_col	Name of the time column.
condition_col	Optional condition column used for separate lines. If the column is missing or contains only missing values, the function plots a single "all_data" time course.
facet_cols	Optional character vector of columns used for faceting.
bin_width_ms	Width of time bins in milliseconds.
ci_level	Confidence level for the band.
min_samples	Minimum number of valid pupil samples required per time bin.
band_alpha	Transparency of the confidence band.
line_width	Width of the mean time-course line.

Value

A ggplot2 plot object.

plot_sampling_rate *Plot Gazeplot sampling-rate diagnostics*

Description

Creates a diagnostic plot of estimated sampling rate by participant/file and media stimulus.

Usage

```
plot_sampling_rate(  
  sampling,  
  user_col = "USER_FILE",  
  media_col = "MEDIA_ID",  
  expected_hz = 60,  
  hz_tolerance = 5  
)
```

Arguments

sampling	Sampling-rate table, usually from check_sampling_rate().
user_col	Column identifying the source/user file.
media_col	Column identifying the media/stimulus.
expected_hz	Expected sampling rate.
hz_tolerance	Allowed deviation from the expected sampling rate.

Value

A ggplot object.

plot_tracking_quality *Plot Gazeplot tracking-quality diagnostics*

Description

Creates a readable diagnostic plot of selected gaze and pupil validity percentages by participant/file and media stimulus.

Usage

```
plot_tracking_quality(  
  data,  
  metric_cols = NULL,  
  user_col = "USER_FILE",  
  media_col = "MEDIA_ID",  
  review_col = "review_required",  
  min_valid_pct = 70  
)
```

Arguments

<code>data</code>	A tracking-quality or flagged-quality table, usually from <code>summarise_tracking_quality()</code> or <code>flag_tracking_quality()</code> .
<code>metric_cols</code>	Validity percentage columns to plot. If NULL, the default is <code>FPOGV_valid_pct</code> and <code>RPV_valid_pct</code> , which provide a compact diagnostic view of gaze and right-pupil validity.
<code>user_col</code>	Column identifying the source/user file.
<code>media_col</code>	Column identifying the media/stimulus.
<code>review_col</code>	Optional column indicating whether a row requires review.
<code>min_valid_pct</code>	Vertical threshold line for acceptable validity.

Value

A ggplot object.

`plot_transition_heatmap`

Plot an AOI transition heatmap

Description

Plot an AOI transition heatmap

Usage

```
plot_transition_heatmap(transitions)
```

Arguments

<code>transitions</code>	Output of <code>compute_transition_matrix()</code> .
--------------------------	--

Value

A ggplot object.

```
prepare_gazepoint_aoi_gamm_data
```

Prepare AOI time-course data for GAMM analysis

Description

Prepare sample-level or binned Gazepoint AOI data for AOI time-course GAMM analysis. The function creates binned subject-by-condition-by-time summaries with binomial success/failure counts for target-AOI looking over time.

Usage

```
prepare_gazepoint_aoi_gamm_data(
  data,
  aoi_col = "aoi_current",
  target_aoi_values = NULL,
  outcome_col = NULL,
  subject_col = "subject",
  condition_col = "condition",
  time_col = "time",
  trial_col = NULL,
  time_bin_col = NULL,
  conditions = NULL,
  time_window = NULL,
  bin_size_ms = 50,
  denominator = c("valid", "all", "aoi_only"),
  valid_aoi_values = NULL,
  non_aoi_values = c("non_aoi"),
  missing_aoi_values = c("missing", ""),
  min_denominator_samples = 1,
  drop_invalid = TRUE,
  missing_condition_label = "all_data",
  outcome_label = "target_aoi"
)
```

Arguments

<code>data</code>	A data frame containing sample-level or binned AOI data.
<code>aoi_col</code>	Name of the AOI-state column. Used when <code>outcome_col = NULL</code> .
<code>target_aoi_values</code>	Character vector identifying target AOI values. Required when <code>outcome_col = NULL</code> .
<code>outcome_col</code>	Optional logical or 0/1 numeric column indicating target AOI looking at the sample level. If supplied, this takes priority over <code>aoi_col</code> and <code>target_aoi_values</code> .
<code>subject_col</code>	Name of the subject/participant column.

<code>condition_col</code>	Name of the condition column. If unavailable, a single fallback condition is created.
<code>time_col</code>	Name of the time column, in milliseconds.
<code>trial_col</code>	Optional trial identifier column.
<code>time_bin_col</code>	Optional existing time-bin column. If NULL, time bins are created from <code>time_col</code> using <code>bin_size_ms</code> .
<code>conditions</code>	Optional character vector of condition levels to retain and order.
<code>time_window</code>	Optional numeric vector of length 2 defining the retained time window in milliseconds.
<code>bin_size_ms</code>	Time-bin width in milliseconds when <code>time_bin_col</code> = NULL.
<code>denominator</code>	Denominator definition. "valid" uses non-missing AOI states, "all" uses all retained rows, and "aoi_only" uses only explicit AOI states.
<code>valid_aoi_values</code>	Optional character vector defining explicit AOI values for "aoi_only" denominators. If NULL, values beginning with "AOI" are treated as explicit AOIs, excluding <code>non_aoi_values</code> .
<code>non_aoi_values</code>	Character vector identifying non-AOI/background states.
<code>missing_aoi_values</code>	Character vector identifying missing AOI-state labels.
<code>min_denominator_samples</code>	Minimum number of denominator samples required per subject-condition-time bin.
<code>drop_invalid</code>	Logical. If TRUE, bins with zero or low denominators are removed from the returned data.
<code>missing_condition_label</code>	Fallback condition label when no usable condition column is available.
<code>outcome_label</code>	Descriptive label for the AOI-GAMM outcome.

Details

This helper is intended for modelling AOI time-course trajectories, such as target-AOI looking probability over time. It is separate from confirmatory AOI-window GLMMs and from cluster-based permutation tests.

Value

A tibble with standardised AOI-GAMM columns.

```
prepare_gazepoint_aoi_glmm_data
    Prepare AOI-window data for binomial GLMMs
```

Description

Prepare AOI-window summaries for confirmatory binomial mixed-effects modelling. The function creates success, failure, denominator, proportion, subject, condition, and window columns from output produced by `summarise_gazepoint_aoi_windows()`.

Usage

```
prepare_gazepoint_aoi_glmm_data(
  data,
  success_col = "n_target_samples",
  denominator = c("valid", "all", "aoi", "custom"),
  denominator_col = NULL,
  valid_denominator_col = "n_valid_denominator_samples",
  all_denominator_col = "n_window_samples",
  aoi_denominator_col = "n_aoi_samples",
  subject_col = "subject",
  condition_col = "condition",
  window_col = "window_label",
  window_start_col = "window_start_ms",
  window_end_col = "window_end_ms",
  group_cols = NULL,
  min_denominator_samples = 1,
  drop_invalid = TRUE,
  missing_condition_label = "all_data",
  outcome_label = "target"
)
```

Arguments

<code>data</code>	AOI-window summary data.
<code>success_col</code>	Column containing the success count. For target-looking models this is usually <code>n_target_samples</code> .
<code>denominator</code>	Denominator definition. Use "valid" for valid AOI-window denominator samples, "all" for all window samples, "aoi" for AOI-only samples, or "custom" with <code>denominator_col</code> .
<code>denominator_col</code>	Custom denominator column when <code>denominator = "custom"</code> .
<code>valid_denominator_col</code>	Column used when <code>denominator = "valid"</code> .
<code>all_denominator_col</code>	Column used when <code>denominator = "all"</code> .

aoi_denominator_col	Column used when denominator = "aoi".
subject_col	Subject/participant column.
condition_col	Optional condition column.
window_col	AOI-window label column.
window_start_col	Optional window-start column.
window_end_col	Optional window-end column.
group_cols	Optional extra grouping columns to keep/check.
min_denominator_samples	Minimum acceptable denominator.
drop_invalid	Logical. If TRUE, rows with invalid binomial counts or too-small denominators are removed.
missing_condition_label	Label used when condition is missing.
outcome_label	Label stored in the output to identify the modelled AOI outcome.

Value

A tibble of GLMM-ready AOI-window rows.

```
prepare_gazepoint_aoi_sequences
      Prepare Gazeptoint AOI sequences
```

Description

Create ordered AOI-state sequences from sample-level Gazeptoint AOI data or from the output of `summarise_gazeptoint_aoi_entries()`. The output is transition-ready and includes the current AOI state, previous state, next state, dwell time before transition, and self-transition flags.

Usage

```
prepare_gazeptoint_aoi_sequences(
  data,
  aoi_col = NULL,
  time_col = "time",
  group_cols = c("subject", "MEDIA_ID", "trial_global"),
  include_non_aoi = TRUE,
  non_aoi_values = c("non_aoi", "none", "background", "outside", "outside_aoi",
    "missing", "missing_aoi"),
  missing_aoi_label = "missing_aoi",
  include_terminal = TRUE
)
```

Arguments

<code>data</code>	A Gazepoint sample-level data frame or an AOI-entry table created by <code>summarise_gazepoint_aoi_entries</code> .
<code>aoi_col</code>	Name of the AOI-state column. Used only when <code>data</code> is sample-level data. If NULL, the function tries <code>aoi_current</code> , <code>AOI</code> , and <code>aoi_state</code> .
<code>time_col</code>	Name of the time column, in milliseconds. Used only when <code>data</code> is sample-level data.
<code>group_cols</code>	Character vector of columns defining independent AOI sequences, usually <code>subject/media/trial</code> .
<code>include_non_aoi</code>	Logical. If TRUE, non-AOI/background states are retained. If FALSE, non-AOI/background states are removed before sequence and transition fields are computed.
<code>non_aoi_values</code>	Character vector of AOI labels treated as background or non-AOI states.
<code>missing_aoi_label</code>	Label used when the AOI value is missing.
<code>include_terminal</code>	Logical. If TRUE, the final state of each sequence is retained with <code>next_state = NA</code> . If FALSE, terminal states are removed so that each output row represents an observed transition.

Value

A tibble with ordered AOI sequence and transition fields.

```
prepare_gazepoint_cluster_data
```

Prepare time-course data for cluster-based permutation tests

Description

Prepare sample-level or already binned Gazepoint time-course data for cluster-based permutation testing. The function standardises subject, condition, time-bin, outcome, sample-count, trial-count, and status columns. It can be used for AOI proportions, pupil time-course outcomes, or other continuous time-varying measures.

Usage

```
prepare_gazepoint_cluster_data(
  data,
  outcome_col,
  subject_col = "subject",
  condition_col = "condition",
  time_col = "time",
  trial_col = NULL,
  time_bin_col = NULL,
  conditions = NULL,
```

```

time_window = NULL,
bin_size_ms = 50,
aggregation = c("mean", "proportion", "sum", "median"),
min_samples_per_bin = 1,
paired = TRUE,
drop_invalid = TRUE,
missing_condition_label = "all_data",
outcome_label = "outcome"
)

```

Arguments

<code>data</code>	A data frame containing sample-level or binned time-course data.
<code>outcome_col</code>	Column containing the outcome to test. For AOI analyses this is often a 0/1 or logical AOI column. For pupil analyses this is often a processed pupil column.
<code>subject_col</code>	Subject/participant column.
<code>condition_col</code>	Optional condition column.
<code>time_col</code>	Time column in milliseconds.
<code>trial_col</code>	Optional trial identifier column.
<code>time_bin_col</code>	Optional existing time-bin column. If NULL, time bins are created from <code>time_col</code> and <code>bin_size_ms</code> .
<code>conditions</code>	Optional character vector of condition levels to keep. Cluster tests are usually pairwise, so this is typically length 2.
<code>time_window</code>	Optional numeric vector of length 2 giving the time range to retain, in milliseconds.
<code>bin_size_ms</code>	Bin size in milliseconds when <code>time_bin_col</code> = NULL.
<code>aggregation</code>	How to aggregate samples within subject-condition-time bins. Supported values are "mean", "proportion", "sum", and "median". "proportion" is equivalent to the mean of a numeric/logical 0/1 outcome.
<code>min_samples_per_bin</code>	Minimum number of samples required per subject-condition-time bin.
<code>paired</code>	Logical. If TRUE, retain only subjects with all retained condition levels.
<code>drop_invalid</code>	Logical. If TRUE, rows and bins that are not suitable for cluster testing are removed.
<code>missing_condition_label</code>	Label used when condition is missing or <code>condition_col</code> is unavailable.
<code>outcome_label</code>	Label stored in the output to identify the outcome.

Details

Cluster-based permutation tests are intended for time-course inference. They should not be used to discover a time window and then test that same window again as a confirmatory analysis.

Value

A tibble with standardised cluster-test preparation columns.

```
prepare_gazepoint_eyetools_data
```

Prepare Gazepoint master data for eyetools-style workflows

Description

Convert a gp3tools master table into a dependency-free, eyetools-friendly sample-level table. The returned data frame keeps one row per sample and creates standard participant, trial, time, gaze-coordinate, binocular coordinate, pupil, AOI, fixation, event, validity, trackloss, and status columns.

Usage

```
prepare_gazepoint_eyetools_data(
  data,
  participant_col = NULL,
  trial_col = NULL,
  time_col = NULL,
  x_col = NULL,
  y_col = NULL,
  left_x_col = NULL,
  left_y_col = NULL,
  right_x_col = NULL,
  right_y_col = NULL,
  pupil_col = NULL,
  left_pupil_col = NULL,
  right_pupil_col = NULL,
  media_col = NULL,
  condition_col = NULL,
  aoi_col = NULL,
  fixation_col = NULL,
  event_col = NULL,
  validity_cols = NULL,
  trackloss_col = NULL,
  screen_x_range = c(0, 1),
  screen_y_range = c(0, 1),
  missing_aoi_label = "missing_aoi",
  keep_original_cols = TRUE
)
```

Arguments

<code>data</code>	A Gazepoint master table or sample-level gaze data frame.
<code>participant_col</code>	Participant/subject identifier column.
<code>trial_col</code>	Trial identifier column. If NULL, a trial identifier is created from <code>media_col</code> when available.

time_col	Sample time column.
x_col	Optional primary gaze x-coordinate column.
y_col	Optional primary gaze y-coordinate column.
left_x_col	Optional left-eye x-coordinate column.
left_y_col	Optional left-eye y-coordinate column.
right_x_col	Optional right-eye x-coordinate column.
right_y_col	Optional right-eye y-coordinate column.
pupil_col	Optional primary pupil column.
left_pupil_col	Optional left-eye pupil column.
right_pupil_col	Optional right-eye pupil column.
media_col	Optional media/stimulus identifier column.
condition_col	Optional condition/grouping column.
aoi_col	Optional AOI label/state column.
fixation_col	Optional fixation identifier column.
event_col	Optional event/marker column.
validity_cols	Optional validity columns used to define trackloss.
trackloss_col	Optional existing trackloss column. If supplied, it is used directly after coercion to logical.
screen_x_range	Numeric length-2 vector defining the screen x range.
screen_y_range	Numeric length-2 vector defining the screen y range.
missing_aoi_label	Label used for missing AOI values.
keep_original_cols	Logical. If TRUE, original columns are retained after the standard adapter columns.

Value

A tibble with class `gp3_eyetools_data`.

```
prepare_gazepoint_eyetrackingr_data
```

Prepare Gazepoint master data for eyetrackingR-style workflows

Description

Convert a `gp3tools` master table into a dependency-free, `eyetrackingR`-friendly sample-level table. The returned data frame keeps one row per gaze sample and creates standard participant, trial, time, gaze-coordinate, AOI, trackloss, and AOI-indicator columns.

Usage

```
prepare_gazepoint_eyetrackingr_data(
  data,
  participant_col = NULL,
  trial_col = NULL,
  time_col = NULL,
  aoi_col = NULL,
  x_col = NULL,
  y_col = NULL,
  media_col = NULL,
  condition_col = NULL,
  validity_cols = NULL,
  aoi_values = NULL,
  aoi_prefix = "aoi_",
  missing_aoi_label = "missing_aoi",
  non_aoi_values = c("outside", "none", "no_aoi", "non_aoi", "background", "off_aoi",
    "missing", "NA"),
  trackloss_col = NULL,
  keep_original_cols = TRUE
)
```

Arguments

<code>data</code>	A Gazepoint master table or sample-level gaze data frame.
<code>participant_col</code>	Participant/subject identifier column.
<code>trial_col</code>	Trial identifier column. If NULL, a trial identifier is created from <code>media_col</code> when available.
<code>time_col</code>	Sample time column.
<code>aoi_col</code>	AOI label/state column.
<code>x_col</code>	Optional gaze x-coordinate column.
<code>y_col</code>	Optional gaze y-coordinate column.
<code>media_col</code>	Optional media/stimulus identifier column.
<code>condition_col</code>	Optional condition/grouping column.
<code>validity_cols</code>	Optional validity columns used to define trackloss.
<code>aoi_values</code>	Optional AOI values for which logical indicator columns should be created. If NULL, values are detected from <code>aoi_col</code> .
<code>aoi_prefix</code>	Prefix for generated AOI indicator columns.
<code>missing_aoi_label</code>	Label used for missing AOI values.
<code>non_aoi_values</code>	Character values treated as non-AOI/background states.
<code>trackloss_col</code>	Optional existing trackloss column. If supplied, it is used directly after coercion to logical.
<code>keep_original_cols</code>	Logical. If TRUE, original columns are retained after the standard adapter columns.

Value

A tibble with class `gp3_eyetrackingr_data`.

```
prepare_gazepoint_fixation_aligned_data
      Prepare fixation- or saccade-contingent aligned Gazeport data
```

Description

Align Gazeport observations to a within-trial event such as first fixation, first target-AOI entry, first fixation to a target AOI, first saccade to a target AOI, or a custom event marker. The helper returns the original data with event-aligned time, event metadata, pre-event/post-event flags, and trial-level summaries that help separate event-driven looking from looks that were already present before the event.

Usage

```
prepare_gazepoint_fixation_aligned_data(
  data,
  time_col,
  participant_col = NULL,
  trial_col = NULL,
  aoi_col = NULL,
  target_aoi = NULL,
  fixation_col = NULL,
  saccade_col = NULL,
  event_col = NULL,
  event_value = NULL,
  alignment_event = c("first_target_entry", "first_fixation_to_target",
    "first_saccade_to_aoi", "first_fixation", "custom"),
  baseline_window = NULL,
  analysis_window = NULL,
  keep_unaligned = FALSE,
  name = "gazepoint_fixation_aligned_data"
)
```

Arguments

<code>data</code>	A data frame containing Gazeport samples, fixation rows, or trial-level time-course rows.
<code>time_col</code>	Time column.
<code>participant_col</code>	Optional participant column.
<code>trial_col</code>	Optional trial column.
<code>aoi_col</code>	Optional AOI column.

target_aoi	Optional character vector identifying the target AOI(s).
fixation_col	Optional fixation indicator column.
saccade_col	Optional saccade indicator column.
event_col	Optional custom event indicator column.
event_value	Optional value(s) in event_col defining the custom event. If NULL and alignment_event = "custom", event_col is interpreted as a logical-like indicator.
alignment_event	Alignment event. Options are "first_target_entry", "first_fixation_to_target", "first_saccade_to_aoi", "first_fixation", and "custom".
baseline_window	Optional numeric vector of length two giving the aligned-time baseline window, for example c(-200, 0).
analysis_window	Optional numeric vector of length two giving the aligned-time analysis window, for example c(0, 1000).
keep_unaligned	Logical. If FALSE, groups without an alignment event are removed from aligned_data. Their status remains in event_table.
name	Character label stored in the returned object.

Value

A list with class gp3_fixation_aligned_data.

```
prepare_gazepoint_gazer_data
```

Prepare Gazepoint master data for gazer-style workflows

Description

Convert a gp3tools master table into a dependency-free, gazer-friendly sample-level table. The returned data frame keeps one row per gaze sample and creates standard participant, trial, time, gaze-coordinate, pupil, AOI, fixation, validity, trackloss, and screen-bound status columns.

Usage

```
prepare_gazepoint_gazer_data(
  data,
  participant_col = NULL,
  trial_col = NULL,
  time_col = NULL,
  x_col = NULL,
  y_col = NULL,
  pupil_col = NULL,
  media_col = NULL,
  condition_col = NULL,
```

```

  aoi_col = NULL,
  fixation_col = NULL,
  validity_cols = NULL,
  trackloss_col = NULL,
  screen_x_range = c(0, 1),
  screen_y_range = c(0, 1),
  missing_aoi_label = "missing_aoi",
  keep_original_cols = TRUE
)

```

Arguments

<code>data</code>	A Gazepoint master table or sample-level gaze data frame.
<code>participant_col</code>	Participant/subject identifier column.
<code>trial_col</code>	Trial identifier column. If NULL, a trial identifier is created from <code>media_col</code> when available.
<code>time_col</code>	Sample time column.
<code>x_col</code>	Gaze x-coordinate column.
<code>y_col</code>	Gaze y-coordinate column.
<code>pupil_col</code>	Optional pupil column.
<code>media_col</code>	Optional media/stimulus identifier column.
<code>condition_col</code>	Optional condition/grouping column.
<code>aoi_col</code>	Optional AOI label/state column.
<code>fixation_col</code>	Optional fixation identifier column.
<code>validity_cols</code>	Optional validity columns used to define trackloss.
<code>trackloss_col</code>	Optional existing trackloss column. If supplied, it is used directly after coercion to logical.
<code>screen_x_range</code>	Numeric length-2 vector defining the screen x range.
<code>screen_y_range</code>	Numeric length-2 vector defining the screen y range.
<code>missing_aoi_label</code>	Label used for missing AOI values.
<code>keep_original_cols</code>	Logical. If TRUE, original columns are retained after the standard adapter columns.

Value

A tibble with class `gp3_gazer_data`.

```
prepare_gazepoint_gca_data
```

Prepare Gazepoint Growth Curve Analysis data

Description

Prepare binned pupil time-course data for Growth Curve Analysis (GCA). The function creates orthogonal polynomial time terms, preserves subject and condition information, and standardises key columns for later mixed-model fitting.

Usage

```
prepare_gazepoint_gca_data(
  data,
  pupil_col = "mean_pupil",
  time_col = "time_bin_center_ms",
  subject_col = "subject",
  condition_col = "condition",
  degree = 3,
  orthogonal = TRUE,
  time_window = NULL,
  valid_samples_col = "n_valid_samples",
  min_valid_samples = 1,
  weights_col = NULL,
  missing_condition_label = "all_data",
  drop_missing = TRUE
)
```

Arguments

<code>data</code>	A binned pupil time-course data frame, usually created by <code>prepare_gazepoint_pupil_gamm_data()</code> .
<code>pupil_col</code>	Name of the pupil outcome column.
<code>time_col</code>	Name of the time column.
<code>subject_col</code>	Name of the subject column.
<code>condition_col</code>	Name of the condition column. If unavailable or entirely missing, a single condition label is used.
<code>degree</code>	Number of polynomial time terms to create.
<code>orthogonal</code>	Logical. If TRUE, use orthogonal polynomial terms from <code>stats::poly()</code> . If FALSE, use raw powers of z-scored time.
<code>time_window</code>	Optional numeric vector of length 2 giving the time window to retain.
<code>valid_samples_col</code>	Optional column containing valid sample counts.
<code>min_valid_samples</code>	Minimum valid samples required per row when <code>valid_samples_col</code> is available.

weights_col Optional weights column to preserve for later modelling.
missing_condition_label Label used when condition values are missing.
drop_missing Logical. If TRUE, rows with missing outcome/time/subject values are removed.

Value

A tibble of class gp3_gca_data with standard GCA columns and polynomial time terms.

```
prepare_gazepoint_hmm_data
```

Prepare Gazepoint AOI/state sequences for HMM-style workflows

Description

Convert ordered Gazepoint AOI/state observations into a dependency-free hidden-Markov-model-ready structure. The helper creates ordered sequence data, transition tables, initial-state probabilities, transition-probability matrices, and observation/emission summaries. It does not fit an HMM and does not import external HMM packages.

Usage

```
prepare_gazepoint_hmm_data(  
  data,  
  state_col = NULL,  
  participant_col = NULL,  
  trial_col = NULL,  
  time_col = NULL,  
  observation_cols = NULL,  
  sequence_id_cols = NULL,  
  covariate_cols = NULL,  
  state_order = NULL,  
  exclude_states = c("missing", "missing_aoi", "missing_coordinate", "trackloss",  
    "track_loss"),  
  missing_state_label = NULL,  
  scale_numeric_observations = FALSE,  
  include_terminal_state = FALSE,  
  terminal_state_label = "END",  
  name = "gazepoint_hmm_data"  
)
```

Arguments

data A data frame containing ordered AOI/state observations.
state_col AOI/state column. If NULL, common AOI/state columns are detected automatically.

participant_col	Optional participant/subject column.
trial_col	Optional trial/sequence column.
time_col	Optional time/order column.
observation_cols	Optional observation columns to carry into the HMM data. If NULL, common gaze, pupil, fixation, and validity columns are detected automatically.
sequence_id_cols	Optional character vector of columns defining separate sequences. If NULL, participant and trial columns are used when available.
covariate_cols	Optional covariate columns to carry into the HMM data.
state_order	Optional preferred hidden-state order.
exclude_states	Character vector of states to exclude before sequence construction.
missing_state_label	Optional label used to retain missing states. If NULL, missing/blank states are removed.
scale_numeric_observations	Logical. If TRUE, z-scored versions of numeric observation columns are added with suffix <code>_z</code> .
include_terminal_state	Logical. If TRUE, each sequence contributes a final transition to <code>terminal_state_label</code> .
terminal_state_label	Terminal-state label.
name	Character label stored in the returned object.

Value

A list with class `gp3_hmm_data`.

```
prepare_gazepoint_pupillometryr_data
```

Prepare Gazepoint master data for pupillometryR-style workflows

Description

Convert a `gp3tools` master table into a dependency-free, `pupillometryR`-friendly sample-level pupil table. The returned data frame keeps one row per sample and creates standard participant, trial, time, pupil, condition, event, baseline, validity, trackloss, and status columns.

Usage

```
prepare_gazepoint_pupillometryr_data(
  data,
  participant_col = NULL,
  trial_col = NULL,
  time_col = NULL,
  pupil_col = NULL,
  media_col = NULL,
  condition_col = NULL,
  event_col = NULL,
  baseline_col = NULL,
  validity_cols = NULL,
  pupil_status_col = NULL,
  trackloss_col = NULL,
  invalid_pupil_status = c("missing", "artifact", "blink", "trackloss", "track_loss",
    "invalid", "excluded", "bad", "outlier"),
  keep_original_cols = TRUE
)
```

Arguments

<code>data</code>	A Gazepoint master table or sample-level gaze/pupil data frame.
<code>participant_col</code>	Participant/subject identifier column.
<code>trial_col</code>	Trial identifier column. If NULL, a trial identifier is created from <code>media_col</code> when available.
<code>time_col</code>	Sample time column.
<code>pupil_col</code>	Pupil column to export.
<code>media_col</code>	Optional media/stimulus identifier column.
<code>condition_col</code>	Optional condition/grouping column.
<code>event_col</code>	Optional event/marker column.
<code>baseline_col</code>	Optional baseline-period indicator column.
<code>validity_cols</code>	Optional validity columns used to define trackloss.
<code>pupil_status_col</code>	Optional pupil-status column used to mark invalid pupil samples.
<code>trackloss_col</code>	Optional existing trackloss column. If supplied, it is used directly after coercion to logical.
<code>invalid_pupil_status</code>	Character values in <code>pupil_status_col</code> treated as invalid pupil samples.
<code>keep_original_cols</code>	Logical. If TRUE, original columns are retained after the standard adapter columns.

Value

A tibble with class `gp3_pupillometryr_data`.

`prepare_gazepoint_pupil_gamm_data`*Prepare Gazepoint pupil GAMM data*

Description

Prepare binned pupil time-course data for GAMM modelling with `mgcv::bam()`. The function aggregates processed sample-level pupil data into subject-by- condition-by-time-bin rows and creates an `AR.start` indicator for autoregressive GAMM models.

Usage

```
prepare_gazepoint_pupil_gamm_data(  
  data,  
  pupil_col = NULL,  
  time_col = "time",  
  subject_col = "subject",  
  condition_col = "condition",  
  x_col = NULL,  
  y_col = NULL,  
  group_cols = c("subject", "condition"),  
  bin_width_ms = 50,  
  time_window = NULL,  
  min_valid_samples = 1,  
  missing_condition_label = "all_data"  
)
```

Arguments

<code>data</code>	A Gazepoint sample-level data frame, usually after pupil preprocessing, interpolation, baseline correction, and optional smoothing.
<code>pupil_col</code>	Name of the pupil column to aggregate. If <code>NULL</code> , the function tries common processed pupil columns such as <code>pupil_smoothed</code> , <code>pupil_baseline_corrected</code> , <code>pupil_interpolated</code> , <code>pupil_clean</code> , and <code>pupil_for_preprocessing</code> .
<code>time_col</code>	Name of the time column in milliseconds. If the requested column is not available, the function tries common alternatives.
<code>subject_col</code>	Name of the subject column. If unavailable, the function tries common participant identifiers.
<code>condition_col</code>	Name of the condition column. If unavailable or entirely missing, a single condition label is used.
<code>x_col</code>	Optional gaze x-coordinate column. If <code>NULL</code> , common x-coordinate columns are auto-detected when available.
<code>y_col</code>	Optional gaze y-coordinate column. If <code>NULL</code> , common y-coordinate columns are auto-detected when available.

group_cols	Columns defining independent time series before binning. Defaults to c("subject", "condition").
bin_width_ms	Width of time bins in milliseconds.
time_window	Optional numeric vector of length 2 giving the time window to retain before binning.
min_valid_samples	Minimum number of valid pupil samples required for a bin to be retained.
missing_condition_label	Label used when condition values are missing or when no usable condition column is available.

Value

A tibble with binned pupil time-course data for GAMM modelling.

```
prepare_gazepoint_pupil_window_model_data
```

Prepare pupil-window data for confirmatory mixed models

Description

Prepare pupil-window summaries or pupil trial-feature tables for confirmatory window-level modelling. The function standardises subject, condition, window, trial/media identifiers, outcome, valid-sample counts, total-sample counts, valid-sample proportions, weights, and model-readiness status columns.

Usage

```
prepare_gazepoint_pupil_window_model_data(
  data,
  outcome_col = "mean_pupil",
  subject_col = "subject",
  condition_col = "condition",
  window_col = "window_label",
  window_start_col = "window_start_ms",
  window_end_col = "window_end_ms",
  trial_col = NULL,
  media_col = "media_id",
  valid_samples_col = "n_valid_pupil",
  total_samples_col = "n_samples",
  min_valid_samples = 5,
  min_valid_prop = 0.7,
  drop_invalid = TRUE,
  missing_condition_label = "all_data",
  outcome_label = "pupil"
)
```

Arguments

data	Pupil-window summary data.
outcome_col	Column containing the pupil outcome to model. The default is mean_pupil.
subject_col	Subject/participant column.
condition_col	Optional condition column. Common aliases such as condition, Condition, and CONDITION are detected when available.
window_col	Pupil-window label column.
window_start_col	Optional window-start column.
window_end_col	Optional window-end column.
trial_col	Optional trial identifier column.
media_col	Optional media/stimulus identifier column. Common aliases such as media_id and MEDIA_ID are detected when available.
valid_samples_col	Optional column containing the number of valid pupil samples in the window. Common aliases such as n_valid_pupil and n_valid_samples are detected when available.
total_samples_col	Optional column containing the total number of samples in the window. Common aliases such as n_samples and n_window_samples are detected when available.
min_valid_samples	Minimum acceptable number of valid pupil samples.
min_valid_prop	Minimum acceptable valid-sample proportion.
drop_invalid	Logical. If TRUE, rows with invalid or low-quality model inputs are removed.
missing_condition_label	Label used when condition is missing.
outcome_label	Label stored in the output to identify the modelled pupil outcome.

Value

A tibble of pupil-window rows prepared for confirmatory modelling.

```
prepare_gazepoint_semimarkov_data
```

Prepare Gazept AI sequences for semi-Markov modelling

Description

Convert ordered AOI/state observations into state-visit and transition-level semi-Markov data. Consecutive repeated states can be collapsed into dwell episodes, producing one row per state visit with dwell duration and next-state information.

Usage

```
prepare_gazepoint_semimarkov_data(
  data,
  state_col = NULL,
  participant_col = NULL,
  trial_col = NULL,
  time_col = NULL,
  duration_col = NULL,
  sequence_id_cols = NULL,
  covariate_cols = NULL,
  exclude_states = c("missing", "missing_aoi", "missing_coordinate", "trackloss",
    "track_loss"),
  missing_state_label = NULL,
  collapse_repeated_states = TRUE,
  include_terminal_states = TRUE,
  terminal_next_state_label = "END",
  name = "gazepoint_semimarkov_data"
)
```

Arguments

<code>data</code>	A data frame containing ordered AOI/state observations.
<code>state_col</code>	AOI/state column. If NULL, common AOI/state columns are detected automatically.
<code>participant_col</code>	Optional participant/subject column.
<code>trial_col</code>	Optional trial/sequence column.
<code>time_col</code>	Optional time/order column.
<code>duration_col</code>	Optional sample-duration column. If supplied, dwell durations are computed by summing this column within each state visit.
<code>sequence_id_cols</code>	Optional character vector of columns defining separate sequences. If NULL, participant and trial columns are used when available.
<code>covariate_cols</code>	Optional character vector of covariate columns to carry into the state-visit and transition tables using the first value within each state visit.
<code>exclude_states</code>	Character vector of states to exclude before creating state visits.
<code>missing_state_label</code>	Optional label used to retain missing states. If NULL, missing/blank states are removed.
<code>collapse_repeated_states</code>	Logical. If TRUE, consecutive repeated states within a sequence are collapsed into a single dwell episode.
<code>include_terminal_states</code>	Logical. If TRUE, the final state visit in each sequence is retained as a transition to <code>terminal_next_state_label</code> .

terminal_next_state_label	Label used for the terminal next state.
name	Character label stored in the returned object.

Value

A list with class `gp3_semimarkov_data`.

read_gazepoint	<i>Read a Gazepoint all-gaze or fixation CSV export</i>
----------------	---

Description

Reads Gazepoint all-gaze and fixation CSV exports, standardises timestamped column names, and removes empty trailing columns produced by Gazepoint exports.

Usage

```
read_gazepoint(path, standardise_names = TRUE, drop_empty_cols = TRUE)
```

Arguments

path	Path to a Gazepoint CSV export.
standardise_names	Logical. If TRUE, standardise <code>TIME(...)</code> and <code>TIMETICK(...)</code> headers.
drop_empty_cols	Logical. If TRUE, remove empty trailing or unnamed columns created by Gazepoint export formatting.

Value

A tibble with attributes `gp3_file_type` and `gp3_source_file`.

read_gazepoint_folder	<i>Read multiple Gazepoint CSV exports from a folder</i>
-----------------------	--

Description

Reads all Gazepoint all-gaze or fixation CSV exports in a folder that match a filename pattern and combines them into one tibble.

Usage

```
read_gazepoint_folder(
  folder,
  pattern = "\\*.csv$",
  source_col = "USER_FILE",
  recursive = FALSE,
  ...
)
```

Arguments

folder	Path to the folder containing Gazepoint CSV exports.
pattern	Regular expression used to select files. For example, "_all_gaze*.csv\$" or "_fixations*.csv\$".
source_col	Name of the column storing the source filename.
recursive	Logical. If TRUE, search subfolders recursively.
...	Additional arguments passed to read_gazepoint().

Value

A tibble containing all matching files combined row-wise.

read_gazepoint_summary

Read a Gazepoint Analysis Data Summary export

Description

Parses the multi-section Data_Summary_export_*.csv file into metadata, aoi_summary, and aoi_by_user tables.

Usage

```
read_gazepoint_summary(path)
```

Arguments

path	Path to Data_Summary_export_*.csv.
------	------------------------------------

Value

A list with metadata, aoi_summary, and aoi_by_user.

 recalibrate_gazepoint_gaze

Offline gaze recalibration using known target coordinates

Description

Apply an offline drift-correction shift to Gazepoint gaze coordinates using known fixation/check-target coordinates. For each group, the helper estimates the horizontal and vertical gaze offset from valid target samples and applies the correction to all gaze samples in the same group.

Usage

```
recalibrate_gazepoint_gaze(
  data,
  x_col,
  y_col,
  target_x_col,
  target_y_col,
  time_col = NULL,
  grouping_cols = NULL,
  calibration_col = NULL,
  calibration_value = NULL,
  method = c("median_shift", "mean_shift"),
  min_valid_points = 3L,
  max_shift = NULL,
  output_x_col = "gaze_x_recalibrated",
  output_y_col = "gaze_y_recalibrated",
  dx_col = "gaze_recalibration_dx",
  dy_col = "gaze_recalibration_dy",
  shift_col = "gaze_recalibration_shift",
  error_before_col = "gaze_error_before_recalibration",
  error_after_col = "gaze_error_after_recalibration",
  status_col = "gaze_recalibration_status",
  overwrite = FALSE,
  name = "gazepoint_gaze_recalibration"
)
```

Arguments

<code>data</code>	A data frame containing gaze and target coordinates.
<code>x_col</code>	Horizontal gaze coordinate column.
<code>y_col</code>	Vertical gaze coordinate column.
<code>target_x_col</code>	Known horizontal target coordinate column.
<code>target_y_col</code>	Known vertical target coordinate column.
<code>time_col</code>	Optional time column used only for stable ordering.

grouping_cols	Optional grouping columns used to estimate one correction per participant, trial, block, stimulus, or other unit.
calibration_col	Optional column identifying rows to use for estimating the correction.
calibration_value	Optional value in calibration_col identifying calibration/check rows. If calibration_col is supplied and calibration_value = NULL, logical TRUE rows are used.
method	Shift estimator. "median_shift" uses median target-minus-gaze offsets; "mean_shift" uses mean offsets.
min_valid_points	Minimum valid target/gaze pairs required per group.
max_shift	Optional maximum Euclidean correction shift. If exceeded, the shift is reported but not applied.
output_x_col	Corrected horizontal gaze output column.
output_y_col	Corrected vertical gaze output column.
dx_col	Estimated horizontal correction column.
dy_col	Estimated vertical correction column.
shift_col	Estimated Euclidean shift-distance column.
error_before_col	Row-wise gaze-to-target error before correction.
error_after_col	Row-wise gaze-to-target error after correction.
status_col	Row-level recalibration status column.
overwrite	Logical. If FALSE, existing output columns are protected.
name	Character label stored in object attributes.

Details

This helper is useful only when known target coordinates are available, for example from calibration checks, fixation targets, validation targets, or drift-check trials.

Value

A tibble with recalibrated gaze columns and recalibration attributes.

```
recommend_gazepoint_exclusions
```

Recommend trial and participant exclusions

Description

Create explicit trial-level and participant-level exclusion recommendations from Gazepoint sample-level quality information. The helper can use validity flags, gaze-coordinate missingness, pupil missingness, and optional artifact flags to produce transparent exclusion tables.

Usage

```

recommend_gazepoint_exclusions(
  data,
  participant_col,
  trial_col = NULL,
  condition_col = NULL,
  validity_col = NULL,
  x_col = NULL,
  y_col = NULL,
  pupil_col = NULL,
  artifact_col = NULL,
  min_trial_samples = 10L,
  max_trial_missing_prop = 0.5,
  max_trial_artifact_prop = 0.5,
  min_participant_trials = 2L,
  min_participant_valid_trials = 1L,
  max_participant_missing_prop = 0.5,
  max_participant_artifact_prop = 0.5,
  require_both_gaze_coordinates = TRUE,
  name = "gazepoint_exclusion_recommendations"
)

```

Arguments

<code>data</code>	A data frame containing sample-level or trial-level data.
<code>participant_col</code>	Participant identifier column.
<code>trial_col</code>	Optional trial identifier column.
<code>condition_col</code>	Optional condition column retained in summaries.
<code>validity_col</code>	Optional logical/numeric/character validity column.
<code>x_col</code>	Optional horizontal gaze coordinate column.
<code>y_col</code>	Optional vertical gaze coordinate column.
<code>pupil_col</code>	Optional pupil column.
<code>artifact_col</code>	Optional logical/numeric/character artifact flag column.
<code>min_trial_samples</code>	Minimum samples required per trial.
<code>max_trial_missing_prop</code>	Maximum missing/unusable sample proportion per trial.
<code>max_trial_artifact_prop</code>	Maximum artifact proportion per trial.
<code>min_participant_trials</code>	Minimum total trials required per participant.
<code>min_participant_valid_trials</code>	Minimum retained trials required per participant.

max_participant_missing_prop	Maximum missing/unusable sample proportion per participant.
max_participant_artifact_prop	Maximum artifact proportion per participant.
require_both_gaze_coordinates	Logical. If both gaze columns are supplied, should a sample be usable only when both coordinates are finite?
name	Character label stored in object attributes.

Details

This function recommends exclusions only. It does not remove rows.

Value

A list with overview, trial recommendations, participant recommendations, an explicit exclusion table, and settings.

run_gazepoint_aoi_multiverse

Run a Gazepoint AOI preprocessing multiverse

Description

Run all AOI preprocessing branches defined by `create_gazepoint_preprocessing_multiverse()`. Each branch creates AOI-window summaries and then prepares binomial AOI GLMM data using the branch-specific denominator and minimum denominator threshold.

Usage

```
run_gazepoint_aoi_multiverse(
  data,
  multiverse,
  branch_ids = NULL,
  windows,
  time_col = "time",
  aoi_col = "aoi_current",
  subject_col = "subject",
  condition_col = NULL,
  group_cols = NULL,
  target_aoi_values,
  distractor_aoi_values = NULL,
  success_col = "n_target_samples",
  outcome_label = "target",
  keep_outputs = TRUE,
  stop_on_error = FALSE
)
```

Arguments

data	A Gazepoint master table or sample-level AOI table.
multiverse	A gp3_preprocessing_multiverse object returned by create_gazepoint_preprocessing_multiverse().
branch_ids	Optional character vector of AOI branch IDs to run.
windows	Numeric vector or labelled window table passed to summarise_gazepoint_aoi_windows().
time_col	Time column.
aoi_col	AOI-state column.
subject_col	Subject column.
condition_col	Optional condition column.
group_cols	Optional grouping columns for AOI-window summaries.
target_aoi_values	Target AOI values.
distractor_aoi_values	Optional distractor AOI values.
success_col	Success-count column passed to prepare_gazepoint_aoi_glimm_data().
outcome_label	Outcome label passed to AOI helpers.
keep_outputs	Logical. If TRUE, keep branch outputs in branch_outputs.
stop_on_error	Logical. If TRUE, stop when a branch fails. If FALSE, record the branch error and continue.

Value

A list with class gp3_aoi_multiverse_results containing overview, branch results, optional branch outputs, and settings.

run_gazepoint_cluster_permutation

Run paired cluster-based permutation tests

Description

Run a paired cluster-based permutation test on time-course data prepared by prepare_gazepoint_cluster_data(). The function tests whether two conditions diverge over time while controlling cluster-level inference using a permutation distribution of maximum cluster statistics.

Usage

```
run_gazepoint_cluster_permutation(
  data,
  condition_order = NULL,
  n_permutations = 1000,
  cluster_threshold = 2,
```

```

tail = c("two_sided", "greater", "less"),
cluster_stat = c("sum_abs_t", "sum_t", "size"),
min_time_bins = 1,
seed = NULL,
paired = TRUE
)

```

Arguments

data	Cluster-ready data produced by <code>prepare_gazepoint_cluster_data()</code> .
condition_order	Optional character vector of length 2 defining the two conditions and their order. The tested difference is condition 2 minus condition 1.
n_permutations	Number of sign-flip permutations.
cluster_threshold	Absolute t-statistic threshold for forming candidate clusters. For <code>tail = "greater"</code> or <code>tail = "less"</code> , the same positive threshold is used in the requested direction.
tail	Direction of the test. <code>"two_sided"</code> tests positive and negative clusters. <code>"greater"</code> tests condition 2 greater than condition 1. <code>"less"</code> tests condition 2 less than condition 1.
cluster_stat	Cluster statistic. <code>"sum_abs_t"</code> sums absolute t-statistics within a cluster. <code>"sum_t"</code> sums signed t-statistics and then uses the absolute value for cluster-level inference. <code>"size"</code> uses the number of time bins.
min_time_bins	Minimum number of adjacent time bins required for a cluster to be retained.
seed	Optional random seed for reproducible permutations.
paired	Logical. Currently only paired within-subject sign-flip permutation is supported.

Details

Cluster-based permutation tests are intended for time-course inference. They should not be used to discover a confirmatory time window and then test that same window again in a second confirmatory model.

Value

A list containing observed time-course statistics, observed clusters, the permutation distribution, settings, and status fields.

```
run_gazepoint_eyetools_fixation_detection
```

Run optional eyetools fixation and saccade detection

Description

Prepare Gazepoint sample-level gaze data for the optional eyetools package and, when eyetools is installed, run fixation and/or saccade detection using `eyetools::fixation_dispersion()`, `eyetools::fixation_VTI()` and/or `eyetools::saccade_VTI()`.

Usage

```
run_gazepoint_eyetools_fixation_detection(
  data,
  participant_col = NULL,
  trial_col = NULL,
  time_col = NULL,
  x_col = NULL,
  y_col = NULL,
  condition_col = NULL,
  stimulus_col = NULL,
  method = c("dispersion", "vti", "saccade", "all"),
  sample_rate = NULL,
  threshold = 100,
  min_dur = 150,
  min_dur_sac = 20,
  disp_tol = 100,
  NA_tol = 0.25,
  smooth = FALSE,
  drop_missing = TRUE,
  progress = FALSE,
  name = "gazepoint_eyetools_fixation_detection"
)
```

Arguments

<code>data</code>	A data frame containing sample-level gaze data.
<code>participant_col</code>	Participant/subject column. If NULL, common names are detected.
<code>trial_col</code>	Trial column. If NULL, common names are detected.
<code>time_col</code>	Time column. If NULL, common names are detected.
<code>x_col</code>	Horizontal gaze coordinate column. If NULL, common names are detected.
<code>y_col</code>	Vertical gaze coordinate column. If NULL, common names are detected.
<code>condition_col</code>	Optional condition/group column.
<code>stimulus_col</code>	Optional stimulus/media column.
<code>method</code>	Detector branch. Options are "dispersion", "vti", "saccade", and "all".
<code>sample_rate</code>	Optional sample rate passed to velocity-threshold functions.
<code>threshold</code>	Velocity threshold passed to velocity-threshold functions.
<code>min_dur</code>	Minimum fixation duration in milliseconds.
<code>min_dur_sac</code>	Minimum saccade duration in milliseconds.
<code>disp_tol</code>	Dispersion tolerance in pixels.
<code>NA_tol</code>	Missing-data tolerance passed to <code>fixation_dispersion()</code> .
<code>smooth</code>	Logical; passed to <code>fixation_VTI()</code> .
<code>drop_missing</code>	Logical. If TRUE, rows with non-finite time, x, or y are removed before detector execution.

progress	Logical. Passed to eyetools detector functions.
name	Character label stored in the returned object.

Details

This helper is an optional external-detector branch. It does not replace the main gp3tools summaries or AOI/transition workflows.

Value

A list with class `gp3_eyetools_fixation_detection`.

```
run_gazepoint_gazer_crosscheck
  Run an optional gazeR pupil-preprocessing cross-check
```

Description

Prepare Gazepoint pupil data for the optional gazer package and, when gazer is installed, run a conservative pupil-preprocessing cross-check using gazeR-style blink extension, smoothing/interpolation, optional baseline correction, and optional downsampling.

Usage

```
run_gazepoint_gazer_crosscheck(
  data,
  participant_col = NULL,
  trial_col = NULL,
  time_col = NULL,
  pupil_col = NULL,
  condition_col = NULL,
  message_col = NULL,
  blink_col = NULL,
  hz = 60,
  fillback = 100,
  fillforward = 100,
  smooth_n = 5,
  step_first = c("smooth", "interpolate"),
  interpolation_type = "linear",
  maxgap = Inf,
  baseline_window = NULL,
  baseline_event = NULL,
  baseline_dur = 100,
  baseline_method = "sub",
  bin_length = NULL,
  name = "gazepoint_gazer_crosscheck"
)
```

Arguments

<code>data</code>	A data frame containing Gazepoint or gp3tools pupil time-series data.
<code>participant_col</code>	Participant/subject column. If NULL, common names are detected.
<code>trial_col</code>	Trial column. If NULL, common names are detected.
<code>time_col</code>	Time column. If NULL, common names are detected.
<code>pupil_col</code>	Pupil column. If NULL, common processed/raw pupil columns are detected.
<code>condition_col</code>	Optional condition column. If NULL, common names are detected; otherwise "all_data" is used.
<code>message_col</code>	Optional message/event column.
<code>blink_col</code>	Optional blink/trackloss column.
<code>hz</code>	Sampling rate passed to gazeR functions.
<code>fillback</code>	Blink-extension window before missing/blink samples, in ms.
<code>fillforward</code>	Blink-extension window after missing/blink samples, in ms.
<code>smooth_n</code>	Smoothing window parameter passed to <code>gazer::smooth_interpolate_pupil()</code> .
<code>step_first</code>	Processing order passed to <code>gazer::smooth_interpolate_pupil()</code> .
<code>interpolation_type</code>	Interpolation type passed to <code>gazer::smooth_interpolate_pupil()</code> .
<code>maxgap</code>	Maximum gap passed to <code>gazer::smooth_interpolate_pupil()</code> .
<code>baseline_window</code>	Optional numeric vector of length 2 passed to <code>gazer::baseline_correction_pupil()</code> .
<code>baseline_event</code>	Optional event label passed to <code>gazer::baseline_correction_pupil_msg()</code> .
<code>baseline_dur</code>	Baseline duration used with <code>baseline_event</code> .
<code>baseline_method</code>	Baseline method used with <code>baseline_event</code> .
<code>bin_length</code>	Optional bin length passed to <code>gazer::downsample_gaze()</code> .
<code>name</code>	Character label stored in the returned object.

Details

This helper is a cross-check branch. It is not intended to replace the main gp3tools pupil preprocessing pipeline.

Value

A list with class `gp3_gazer_crosscheck`.

```
run_gazepoint_model_leave_one_out
```

Run leave-one-unit model sensitivity analysis

Description

Refit the same model repeatedly while removing one participant, item, stimulus, trial, or other analysis unit at a time. The helper compares leave-one-out estimates with the full-data model to assess whether a key effect is driven by a single unit.

Usage

```
run_gazepoint_model_leave_one_out(
  data,
  unit_col,
  fit_function,
  extract_function = NULL,
  effect_terms = NULL,
  min_rows = 2L,
  keep_models = FALSE,
  name = "gazepoint_model_leave_one_out"
)
```

Arguments

<code>data</code>	A data frame used for model fitting.
<code>unit_col</code>	Column identifying the unit to leave out, for example subject, participant, item, stimulus, or trial.
<code>fit_function</code>	Function that takes one data frame argument and returns a fitted model.
<code>extract_function</code>	Optional function that takes a fitted model and returns a data frame of effects. If NULL, a default coefficient extractor is used for common model objects.
<code>effect_terms</code>	Optional character vector of terms/effects to retain in the sensitivity summary.
<code>min_rows</code>	Minimum number of rows required after leaving one unit out.
<code>keep_models</code>	Logical. If TRUE, keep the full model and refitted models.
<code>name</code>	Character label stored in the returned object.

Details

This is a generic robustness wrapper. It can be used with linear models, GLMs, mixed models, GAMMs, GCA models, AOI GLMMs, pupil LMMs, or any custom model as long as a fitting function is supplied.

Value

A list with class `gp3_model_leave_one_out_sensitivity`.

 run_gazepoint_pupil_multiverse

Run a Gazepoint pupil preprocessing multiverse

Description

Run all pupil preprocessing branches defined by `create_gazepoint_preprocessing_multiverse()`. Each branch can apply pupil artifact flagging, interpolation, baseline correction, smoothing, and optional pupil-window summarisation.

Usage

```
run_gazepoint_pupil_multiverse(
  data,
  multiverse,
  branch_ids = NULL,
  pupil_col = NULL,
  time_col = NULL,
  group_cols = NULL,
  summarise_windows = FALSE,
  windows = NULL,
  keep_outputs = TRUE,
  stop_on_error = FALSE
)
```

Arguments

<code>data</code>	A Gazepoint master table or processed pupil table.
<code>multiverse</code>	A <code>gp3_preprocessing_multiverse</code> object returned by <code>create_gazepoint_preprocessing_multiverse()</code> .
<code>branch_ids</code>	Optional character vector of pupil branch IDs to run.
<code>pupil_col</code>	Optional pupil column passed to downstream preprocessing helpers when supported.
<code>time_col</code>	Optional time column passed to downstream preprocessing helpers when supported.
<code>group_cols</code>	Optional grouping columns passed to downstream preprocessing helpers when supported.
<code>summarise_windows</code>	Logical. If TRUE, summarise each processed pupil branch into pupil analysis windows.
<code>windows</code>	Optional windows passed to <code>summarise_gazepoint_pupil_windows()</code> when <code>summarise_windows = TRUE</code> .
<code>keep_outputs</code>	Logical. If TRUE, keep processed branch data in <code>branch_outputs</code> .
<code>stop_on_error</code>	Logical. If TRUE, stop when a branch fails. If FALSE, record the branch error and continue.

Value

A list with class `gp3_pupil_multiverse_results` containing overview, branch results, optional branch outputs, and settings.

run_gazepoint_workflow

Run a complete Gazepoint analysis workflow

Description

Reads Gazepoint all-gaze and fixation exports from a folder, computes sampling-rate checks, tracking-quality summaries, quality flags, and AOI-level metrics, and optionally exports output tables, diagnostic plots, and an HTML diagnostic report.

Usage

```
run_gazepoint_workflow(
  export_dir,
  all_gaze_pattern = "_all_gaze\\.csv$",
  fixation_pattern = "_fixations\\.csv$",
  check_file_pairs = TRUE,
  group_cols = c("USER_FILE", "MEDIA_ID"),
  user_col = "USER_FILE",
  sample_rate = 60,
  min_gaze_valid_pct = 70,
  min_pupil_valid_pct = 70,
  expected_hz = 60,
  hz_tolerance = 5,
  min_duration_sec = NULL,
  output_dir = NULL,
  prefix = "gazepoint",
  overwrite = TRUE,
  save_plots = FALSE,
  plot_output_dir = NULL,
  create_report = FALSE,
  report_file = NULL,
  report_title = "Gazepoint diagnostic report",
  report_plot_dir = NULL,
  report_max_rows = 30
)
```

Arguments

`export_dir` Folder containing Gazepoint CSV export files.

`all_gaze_pattern` Regular expression for selecting all-gaze files.

fixation_pattern	Regular expression for selecting fixation files.
check_file_pairs	Logical. If TRUE, check that each participant/source has both an all-gaze file and a fixation file before importing.
group_cols	Columns used for grouped sampling and tracking-quality summaries.
user_col	Column name used to identify the source/user file.
sample_rate	Sampling rate used for approximate sample-based AOI viewed time.
min_gaze_valid_pct	Minimum acceptable FPOGV validity percentage.
min_pupil_valid_pct	Minimum acceptable pupil validity percentage.
expected_hz	Expected sampling rate.
hz_tolerance	Allowed deviation from the expected sampling rate.
min_duration_sec	Optional minimum acceptable recording duration in seconds.
output_dir	Optional folder where output CSV files should be written.
prefix	Filename prefix used when exporting output tables, plots, and report.
overwrite	Logical. If FALSE, stop when output files already exist.
save_plots	Logical. If TRUE, save standard diagnostic plots.
plot_output_dir	Optional folder where diagnostic plots should be saved. If NULL, output_dir is used.
create_report	Logical. If TRUE, create an HTML diagnostic report.
report_file	Optional path to the HTML report. If NULL, a report file is created in output_dir using prefix.
report_title	Title used in the HTML diagnostic report.
report_plot_dir	Optional folder for plots used inside the HTML report.
report_max_rows	Maximum number of rows shown in report preview tables.

Value

A named list containing file-pair checks, imported data, analysis tables, quality flags, written table paths, written plot paths, and written report path.

save_gazepoint_plots *Save standard Gazepoint diagnostic plots*

Description

Saves standard diagnostic plots produced from gp3tools workflow outputs.

Usage

```
save_gazepoint_plots(  
  flagged_quality = NULL,  
  sampling = NULL,  
  output_dir,  
  prefix = "gazepoint",  
  overwrite = TRUE,  
  width = 9,  
  height_quality = 6,  
  height_sampling = 5,  
  dpi = 300  
)
```

Arguments

flagged_quality	Flagged tracking-quality table, usually from flag_tracking_quality() or run_gazepoint_workflow()
sampling	Sampling-rate table, usually from check_sampling_rate() or run_gazepoint_workflow().
output_dir	Folder where plot files should be saved.
prefix	Filename prefix used for saved plot files.
overwrite	Logical. If FALSE, stop when output plot files already exist.
width	Plot width in inches.
height_quality	Tracking-quality plot height in inches.
height_sampling	Sampling-rate plot height in inches.
dpi	Plot resolution.

Value

A tibble with plot names and written file paths.

 smooth_gazepoint_pupil

Smooth Gazepoint pupil data

Description

Applies sample-based rolling smoothing to a Gazepoint pupil time series, typically after `flag_gazepoint_pupil()`, `interpolate_gazepoint_pupil()`, and optionally `baseline_correct_gazepoint_pupil()`. The function preserves the original pupil column and adds smoothed-output columns.

Usage

```
smooth_gazepoint_pupil(
  data,
  pupil_col = NULL,
  time_col = NULL,
  group_cols = c("subject", "media_id"),
  window_samples = 5,
  method = c("mean", "median"),
  align = c("center", "right", "left"),
  min_points = 1,
  preserve_missing = TRUE
)
```

Arguments

<code>data</code>	A Gazepoint master table, preferably after pupil preprocessing.
<code>pupil_col</code>	Optional name of the pupil column to smooth. If NULL, the function detects one of <code>pupil_baseline_corrected</code> , <code>pupil_baseline_percent_change</code> , <code>pupil_interpolated</code> , <code>pupil_for_preprocessing</code> , <code>mean_pupil</code> , <code>pupil</code> , <code>pupil_raw</code> , <code>left_pupil</code> , or <code>right_pupil</code> .
<code>time_col</code>	Optional name of the time column. If NULL, the function detects one of <code>time_ms</code> , <code>time</code> , <code>time_orig</code> , or <code>time_orig_ms</code> .
<code>group_cols</code>	Character vector of grouping columns used to keep smoothing within independent time series. Defaults to <code>c("subject", "media_id")</code> . Columns such as <code>"trial"</code> or <code>"trial_global"</code> can be added when available. Use <code>character(0)</code> for global smoothing.
<code>window_samples</code>	Number of samples in the rolling smoothing window. Defaults to 5.
<code>method</code>	Smoothing statistic. One of <code>"mean"</code> or <code>"median"</code> . Defaults to <code>"mean"</code> .
<code>align</code>	Window alignment. One of <code>"center"</code> , <code>"right"</code> , or <code>"left"</code> . Defaults to <code>"center"</code> .
<code>min_points</code>	Minimum number of finite values required inside a window to return a smoothed value. Defaults to 1.
<code>preserve_missing</code>	Logical. If TRUE, rows with missing/non-finite input remain missing in <code>pupil_smoothed</code> . Defaults to TRUE.

Value

A tibble containing the original data plus pupil-smoothing columns.

Examples

```
pupil_data <- tibble::tibble(  
  subject = rep("P1", 6),  
  time = c(0, 16, 32, 48, 64, 80),  
  pupil_baseline_corrected = c(0.00, 0.10, 0.20, 0.10, 0.00, -0.10)  
)  
  
smoothed <- smooth_gazepoint_pupil(  
  pupil_data,  
  pupil_col = "pupil_baseline_corrected",  
  time_col = "time",  
  group_cols = "subject",  
  window_samples = 3,  
  method = "mean"  
)  
  
smoothed
```

standardise_gazepoint_names

Standardise Gazepoint column names

Description

Converts timestamped Gazepoint headers such as TIME(2026/02/20 00:53:57.275) to TIME, converts TIMETICK(f=10000000) to TIMETICK, trims whitespace, and removes empty columns created by trailing commas in Gazepoint exports.

Usage

```
standardise_gazepoint_names(x)
```

Arguments

x A data frame or character vector of column names.

Value

If x is a data frame, the same data frame with standardised names and empty Gazepoint columns removed. If x is a character vector, a character vector.

summarise_aoi_samples *Summarise sample-level AOI viewing*

Description

Computes transparent AOI metrics from sample-level rows. These may not exactly reproduce Gaze-point Analysis summary metrics; use `read_gazepoint_summary()` when official Gaze-point summary values are available.

Usage

```
summarise_aoi_samples(  
  data,  
  group_cols = "MEDIA_ID",  
  aoi_col = "AOI",  
  time_col = "TIME"  
)
```

Arguments

<code>data</code>	A Gaze-point all-gaze data frame.
<code>group_cols</code>	Grouping columns.
<code>aoi_col</code>	AOI column name.
<code>time_col</code>	Time column name.

Value

A tibble with AOI sample count, TTFF, and approximate dwell time.

summarise_fixations *Summarise fixation-level AOI metrics*

Description

Summarise fixation-level AOI metrics

Usage

```
summarise_fixations(data, group_cols = "MEDIA_ID", aoi_col = "AOI")
```

Arguments

<code>data</code>	A Gaze-point fixation data frame.
<code>group_cols</code>	Grouping columns.
<code>aoi_col</code>	AOI column name.

Value

A tibble with fixation counts and summed fixation duration by AOI.

```
summarise_gazepoint_aoi
```

Summarise Gazepoint AOI metrics from gaze and fixation exports

Description

Combines sample-level AOI viewing information from all-gaze data with fixation-level AOI metrics from fixation data.

Usage

```
summarise_gazepoint_aoi(
  gaze_data,
  fixation_data,
  user_col = "USER_FILE",
  sample_rate = 60
)
```

Arguments

<code>gaze_data</code>	A Gazepoint all-gaze data frame imported with <code>read_gazepoint()</code> .
<code>fixation_data</code>	A Gazepoint fixation data frame imported with <code>read_gazepoint()</code> .
<code>user_col</code>	Name of the column identifying the user file. Default is "USER_FILE".
<code>sample_rate</code>	Assumed sampling rate used to approximate viewed time from sample counts.

Value

A tibble with one row per user, media, and AOI.

```
summarise_gazepoint_aoi_entries
```

Summarise Gazepoint AOI entry episodes

Description

Convert sample-level AOI states into AOI entry episodes. An entry starts whenever the AOI state changes within a subject, media, trial, or other grouping unit.

Usage

```

summarise_gazepoint_aoi_entries(
  data,
  aoi_col = NULL,
  time_col = "time",
  group_cols = c("subject", "MEDIA_ID", "trial_global"),
  include_non_aoi = TRUE,
  non_aoi_values = c("non_aoi", "none", "background", "outside", "outside_aoi",
    "missing", "missing_aoi"),
  missing_aoi_label = "missing_aoi"
)

```

Arguments

<code>data</code>	A Gazepoint master/sample-level data frame.
<code>aoi_col</code>	Name of the AOI-state column. If NULL, the function tries <code>aoi_current</code> , <code>AOI</code> , and <code>aoi_state</code> .
<code>time_col</code>	Name of the time column, in milliseconds.
<code>group_cols</code>	Character vector of columns defining independent sequences, usually <code>subject/media/trial</code> .
<code>include_non_aoi</code>	Logical. If TRUE, non-AOI/background episodes are retained. If FALSE, they are removed after entry order and neighbouring states have been computed.
<code>non_aoi_values</code>	Character vector of AOI labels treated as background or non-AOI states.
<code>missing_aoi_label</code>	Label used when the AOI value is missing.

Value

A tibble with one row per AOI entry episode.

`summarise_gazepoint_aoi_transitions`

Summarise Gazepoint AOI transition features

Description

Summarise AOI transitions at the trial or group level. The function can work from sample-level Gazepoint AOI data, AOI-entry tables created by `summarise_gazepoint_aoi_entries()`, or AOI-sequence tables created by `prepare_gazepoint_aoi_sequences()`.

Usage

```

summarise_gazepoint_aoi_transitions(
  data,
  aoi_col = NULL,
  time_col = "time",
  group_cols = c("subject", "MEDIA_ID", "trial_global"),
  include_non_aoi = TRUE,
  target_aoi_values = NULL,
  distractor_aoi_values = NULL,
  non_aoi_values = c("non_aoi", "none", "background", "outside", "outside_aoi",
    "missing", "missing_aoi"),
  missing_aoi_label = "missing_aoi"
)

```

Arguments

<code>data</code>	A Gazepoint sample-level data frame, AOI-entry table, or AOI-sequence table.
<code>aoi_col</code>	Name of the AOI-state column. Used only when data is sample-level data. If NULL, the function tries <code>aoi_current</code> , <code>AOI</code> , and <code>aoi_state</code> .
<code>time_col</code>	Name of the time column, in milliseconds. Used only when data is sample-level data.
<code>group_cols</code>	Character vector of columns defining independent AOI sequences, usually <code>subject/media/trial</code> .
<code>include_non_aoi</code>	Logical. If TRUE, non-AOI/background states are retained before transition summaries are computed. This is useful for background-to-target and target-to-background features.
<code>target_aoi_values</code>	Optional character vector defining target AOI labels.
<code>distractor_aoi_values</code>	Optional character vector defining distractor AOI labels.
<code>non_aoi_values</code>	Character vector of AOI labels treated as background or non-AOI states.
<code>missing_aoi_label</code>	Label used when the AOI value is missing.

Value

A tibble with one row per group and trial-level AOI transition features.

```
summarise_gazepoint_aoi_trial_features
      Summarise Gazepoint AOI trial features
```

Description

Create trial-level AOI features from sample-level Gazepoint AOI data or from AOI-entry tables created by `summarise_gazepoint_aoi_entries()`. The output includes AOI dwell, entry, TTFF, revisit, and transition features.

Usage

```
summarise_gazepoint_aoi_trial_features(
  data,
  aoi_col = NULL,
  time_col = "time",
  group_cols = c("subject", "MEDIA_ID", "trial_global"),
  include_non_aoi = TRUE,
  target_aoi_values = NULL,
  distractor_aoi_values = NULL,
  non_aoi_values = c("non_aoi", "none", "background", "outside", "outside_aoi",
    "missing", "missing_aoi"),
  missing_aoi_label = "missing_aoi"
)
```

Arguments

<code>data</code>	A Gazepoint sample-level data frame, AOI-entry table, or compatible AOI table.
<code>aoi_col</code>	Name of the AOI-state column. Used only when data is sample-level data. If NULL, the function tries <code>aoi_current</code> , <code>AOI</code> , and <code>aoi_state</code> .
<code>time_col</code>	Name of the time column, in milliseconds. Used only when data is sample-level data.
<code>group_cols</code>	Character vector of columns defining independent trials, usually <code>subject/media/trial</code> .
<code>include_non_aoi</code>	Logical. If TRUE, non-AOI/background states are kept when computing trial-duration and transition features.
<code>target_aoi_values</code>	Optional character vector defining target AOI labels.
<code>distractor_aoi_values</code>	Optional character vector defining distractor AOI labels.
<code>non_aoi_values</code>	Character vector of AOI labels treated as background or non-AOI states.
<code>missing_aoi_label</code>	Label used when the AOI value is missing.

Value

A tibble with one row per trial/group and AOI trial-level features.

```
summarise_gazepoint_aoi_windows
```

Summarise Gazepoint AOI samples within predefined time windows

Description

Summarise sample-level AOI states into predefined analysis windows. This is intended for confirmatory AOI window modelling, especially binomial target-looking models where target samples are modelled relative to a denominator such as all valid window samples.

Usage

```
summarise_gazepoint_aoi_windows(
  data,
  windows,
  time_col = "time",
  aoi_col = NULL,
  subject_col = "subject",
  condition_col = "condition",
  group_cols = NULL,
  target_aoi_values = NULL,
  distractor_aoi_values = NULL,
  non_aoi_values = c("non_aoi", "none", "background", "outside", "outside_aoi",
    "missing", "missing_aoi"),
  window_label_col = "window_label",
  window_start_col = "window_start_ms",
  window_end_col = "window_end_ms",
  include_right_endpoint = FALSE,
  missing_condition_label = "all_data",
  missing_aoi_label = "missing_aoi"
)
```

Arguments

<code>data</code>	A Gazepoint master table or sample-level gaze data.
<code>windows</code>	Numeric breakpoints, for example <code>c(0, 500, 1000, 2000)</code> , or a data frame with window labels and start/end columns.
<code>time_col</code>	Name of the time column.
<code>aoi_col</code>	Optional AOI-state column. If <code>NULL</code> , the function attempts to detect <code>aoi_current</code> , <code>AOI</code> , or <code>aoi_state</code> .
<code>subject_col</code>	Name of the subject column.
<code>condition_col</code>	Optional condition column.

group_cols	Additional grouping columns. Defaults to subject, condition, media, trial-global, and trial columns when available.
target_aoi_values	Character vector identifying target AOIs.
distractor_aoi_values	Character vector identifying distractor AOIs.
non_aoi_values	Character vector identifying background/non-AOI states.
window_label_col	Window label column when windows is a data frame.
window_start_col	Window start column when windows is a data frame.
window_end_col	Window end column when windows is a data frame.
include_right_endpoint	Logical. If TRUE, include the right endpoint of each window.
missing_condition_label	Label used when condition is missing.
missing_aoi_label	Label used when AOI state is missing.

Value

A tibble with one row per group and AOI window.

summarise_gazepoint_clusters
Summarise cluster-based permutation results

Description

Create compact reporting tables from the output of `run_gazepoint_cluster_permutation()`. The function returns an overview table, all observed clusters, significant clusters, time-course summary, permutation-distribution summary, settings table, and circularity warning.

Usage

```
summarise_gazepoint_clusters(  
  result,  
  alpha = 0.05,  
  round_digits = NULL,  
  include_timecourse = TRUE  
)
```

Arguments

result	A result object returned by <code>run_gazepoint_cluster_permutation()</code> .
alpha	Cluster-level significance threshold.
round_digits	Optional number of digits for rounding numeric reporting columns. If NULL, no rounding is applied.
include_timecourse	Logical. If TRUE, include the full observed time-course table in the returned object.

Details

Cluster-based permutation tests are intended for time-course inference. They should not be used to discover a confirmatory time window and then test that same window again in a second confirmatory model.

Value

A list of summary tables.

summarise_gazepoint_emmeans

Summarise estimated marginal means and contrasts

Description

Create manuscript-ready estimated marginal means and pairwise contrast tables from fitted models used in `gp3tools` workflows.

Usage

```
summarise_gazepoint_emmeans(
  model,
  specs,
  by = NULL,
  model_name = NULL,
  type = "response",
  contrast_method = "pairwise",
  adjust = "tukey",
  conf_level = 0.95,
  include_contrasts = TRUE
)
```

Arguments

model	A fitted model object, or a gp3tools fit object containing a \$model element.
specs	Character vector or formula passed to emmeans::emmeans().
by	Optional character vector of grouping variables passed to emmeans::emmeans().
model_name	Optional model label used in returned tables.
type	Scale passed to emmeans summaries. Common values are "link" and "response".
contrast_method	Contrast method passed to emmeans::contrast().
adjust	Multiplicity adjustment for contrasts.
conf_level	Confidence level.
include_contrasts	Logical. If TRUE, compute contrasts.

Details

The function uses the optional emmeans package. If emmeans is not installed, the function returns structured skipped tables rather than failing. This keeps emmeans as an optional suggested dependency.

Value

A list with overview, emmeans, contrasts, and settings. The returned object has class gp3_emmeans_summary.

summarise_gazepoint_fixation_trials

Summarise Gazepoint fixation trial features

Description

Create trial-level fixation features from Gazepoint fixation-level data. The function supports common Gazepoint fixation export columns such as FPOGS, FPOGD, FPOGX, FPOGY, FPOGID, FPOGV, and AOI, as well as already-standardised columns.

Usage

```
summarise_gazepoint_fixation_trials(
  data,
  group_cols = NULL,
  fixation_id_col = NULL,
  start_col = NULL,
  duration_col = NULL,
  x_col = NULL,
  y_col = NULL,
  valid_col = NULL,
  aoi_col = NULL,
```

```

start_time_unit = c("auto", "ms", "s"),
duration_unit = c("auto", "ms", "s"),
valid_only = TRUE,
include_non_aoi = TRUE,
target_aoi_values = NULL,
distractor_aoi_values = NULL,
non_aoi_values = c("non_aoi", "none", "background", "outside", "outside_aoi",
  "missing", "missing_aoi"),
missing_aoi_label = "missing_aoi"
)

```

Arguments

<code>data</code>	A Gazepoint fixation-level data frame.
<code>group_cols</code>	Character vector of columns defining independent trials. If NULL, the function tries to infer sensible grouping columns from participant, media, and trial columns.
<code>fixation_id_col</code>	Optional fixation ID column. If NULL, the function tries FPOGID, <code>fixation_id</code> , and related names.
<code>start_col</code>	Optional fixation start-time column. If NULL, the function tries FPOGS, <code>fixation_start_time</code> , <code>time</code> , <code>TIME</code> , and <code>TIMETICK</code> .
<code>duration_col</code>	Optional fixation-duration column. If NULL, the function tries FPOGD, <code>fixation_duration_ms</code> , <code>fixation_duration</code> , and related names.
<code>x_col</code>	Optional fixation x-coordinate column.
<code>y_col</code>	Optional fixation y-coordinate column.
<code>valid_col</code>	Optional fixation-validity column. If detected and <code>valid_only = TRUE</code> , invalid fixations are removed.
<code>aoi_col</code>	Optional AOI column. If NULL, the function tries <code>AOI</code> , <code>aoi_current</code> , and <code>aoi_state</code> .
<code>start_time_unit</code>	Unit for the start-time column: "auto", "ms", or "s".
<code>duration_unit</code>	Unit for the duration column: "auto", "ms", or "s".
<code>valid_only</code>	Logical. If TRUE, invalid fixations are removed when a validity column is available.
<code>include_non_aoi</code>	Logical. If TRUE, non-AOI/background fixations are included. If FALSE, they are removed before summaries are computed.
<code>target_aoi_values</code>	Optional character vector defining target AOI labels.
<code>distractor_aoi_values</code>	Optional character vector defining distractor AOI labels.
<code>non_aoi_values</code>	Character vector of AOI labels treated as background or non-AOI states.
<code>missing_aoi_label</code>	Label used when the AOI value is missing.

Value

A tibble with one row per trial/group and fixation-level features.

```
summarise_gazepoint_fixed_effects
  Summarise fixed effects from fitted models
```

Description

Create a compact manuscript-ready fixed-effect summary table from common models used in gp3tools workflows.

Usage

```
summarise_gazepoint_fixed_effects(
  model,
  model_name = NULL,
  conf_level = 0.95,
  exponentiate = FALSE,
  drop_intercept = FALSE
)
```

Arguments

<code>model</code>	A fitted model object, or a gp3tools fit object containing a <code>\$model</code> element.
<code>model_name</code>	Optional model label used in the returned table.
<code>conf_level</code>	Confidence level for Wald confidence intervals.
<code>exponentiate</code>	Logical. If TRUE, exponentiate estimates and confidence intervals. This is useful for logistic models when reporting odds ratios.
<code>drop_intercept</code>	Logical. If TRUE, remove the intercept row.

Details

The function supports `lm`, `glm`, `lme4` mixed models, and `mgcv` GAM/BAM objects. It can also accept a gp3tools fit object containing a `$model` element. Confidence intervals are computed using a Wald approximation from the estimate and standard error so that the function remains lightweight and fast for mixed models.

Value

A tibble with fixed-effect estimates, standard errors, test statistics, p-values when available, confidence intervals, significance stars, and status fields.

 summarise_gazepoint_multiverse_results

Summarise Gazepoint preprocessing multiverse results

Description

Summarise pupil and/or AOI preprocessing multiverse result objects created by `run_gazepoint_pupil_multiverse()` and `run_gazepoint_aoi_multiverse()`.

Usage

```
summarise_gazepoint_multiverse_results(..., results = NULL)
```

Arguments

<code>...</code>	One or more multiverse result objects.
<code>results</code>	Optional named list of multiverse result objects.

Value

A list with class `gp3_multiverse_summary_results` containing overview, branch summary, failure summary, and settings tables.

 summarise_gazepoint_pupil

Summarise Gazepoint pupil data

Description

Creates compact pupil-quality and pupil-distribution summaries from a Gazepoint master sample-level table created by `as_gazepoint_master()` or `create_gazepoint_master()`. This function is intended as the first pupil preprocessing gate before interpolation, filtering, baseline correction, or pupil-based modelling.

Usage

```
summarise_gazepoint_pupil(
  master,
  group_cols = c("subject", "media_id"),
  pupil_col = NULL,
  time_col = NULL,
  missing_pupil_col = NULL,
  min_pupil = 0,
  max_pupil = Inf,
  outlier_k = 1.5
)
```

Arguments

master	A Gazepoint master sample-level table.
group_cols	Character vector of grouping columns. Defaults to <code>c("subject", "media_id")</code> using internally standardised names. Use <code>character(0)</code> for an overall summary.
pupil_col	Optional name of the pupil column to summarise. If NULL, the function detects one of <code>mean_pupil</code> , <code>pupil</code> , <code>pupil_raw</code> , <code>left_pupil</code> , or <code>right_pupil</code> .
time_col	Optional name of the time column. If NULL, the function detects one of <code>time_ms</code> , <code>time</code> , <code>time_orig</code> , or <code>time_orig_ms</code> .
missing_pupil_col	Optional name of the missing-pupil flag column. If NULL, the function uses <code>missing_pupil</code> when available.
min_pupil	Minimum plausible pupil value. Defaults to 0.
max_pupil	Maximum plausible pupil value. Defaults to Inf. Use narrower values, such as 1 and 9, only when the pupil column is known to be measured in millimetres.
outlier_k	Multiplier for IQR-based outlier detection. Defaults to 1.5.

Value

A tibble with pupil-quality and pupil-distribution summaries.

Examples

```

master <- gazepoint_example_master
master <- create_gazepoint_master(
  gaze_data = gazepoint_example_master,
  screen_width_px = 1920,
  screen_height_px = 1080
)

summarise_gazepoint_pupil(master)
summarise_gazepoint_pupil(master, group_cols = "subject")
summarise_gazepoint_pupil(master, group_cols = character(0))

```

summarise_gazepoint_pupil_trial_features

Summarise Gazepoint pupil trial-level features

Description

Convert sample-level Gazepoint pupil time series into trial-level pupil features for statistical modelling.

Usage

```
summarise_gazepoint_pupil_trial_features(
  data,
  group_cols = c("subject", "trial_global"),
  pupil_col = NULL,
  time_col = "time",
  interpolated_col = "pupil_was_interpolated",
  artifact_col = NULL,
  artifact_reason_col = NULL,
  early_window = c(0, 500),
  middle_window = c(500, 1500),
  late_window = c(1500, 3000),
  min_valid_samples = 1
)
```

Arguments

<code>data</code>	A Gazepoint pupil data frame.
<code>group_cols</code>	Character vector of grouping columns. The default is <code>c("subject", "trial_global")</code> .
<code>pupil_col</code>	Name of the processed pupil column to summarise. If <code>NULL</code> , the function tries <code>pupil_smoothed</code> , <code>pupil_baseline_corrected</code> , <code>pupil_baseline_percent_change</code> , <code>pupil_interpolated</code> , <code>pupil_clean</code> , and <code>pupil</code> .
<code>time_col</code>	Name of the time column.
<code>interpolated_col</code>	Optional logical interpolation flag column.
<code>artifact_col</code>	Optional artifact flag column. If <code>NULL</code> , the function tries to detect <code>pupil_artifact_flag</code> , <code>pupil_flag_invalid</code> , or <code>artifact_flag</code> .
<code>artifact_reason_col</code>	Optional artifact-reason column. If <code>NULL</code> , the function tries to detect <code>pupil_artifact_reason</code> , <code>pupil_flag_reason</code> , or <code>artifact_reason</code> .
<code>early_window</code>	Numeric vector of length 2 defining the early window in milliseconds.
<code>middle_window</code>	Numeric vector of length 2 defining the middle window in milliseconds.
<code>late_window</code>	Numeric vector of length 2 defining the late window in milliseconds.
<code>min_valid_samples</code>	Minimum number of valid pupil samples required for a trial to be labelled "ok".

Details

The function summarises one row per trial or other user-defined grouping. It computes mean pupil, peak pupil, time-to-peak, AUC, early/middle/late window means, valid-sample percentage, interpolation percentage, artifact percentage, and missingness summaries.

Value

A tibble with one row per trial/group.

 summarise_gazepoint_pupil_windows

Summarise Gazepoint pupil responses within time windows

Description

Aggregates processed Gazepoint pupil data into user-defined analysis windows, typically after `flag_gazepoint_pupil()`, `interpolate_gazepoint_pupil()`, `baseline_correct_gazepoint_pupil()`, and `smooth_gazepoint_pupil()`. The function can summarise raw, interpolated, baseline-corrected, percent-change, or smoothed pupil columns.

Usage

```
summarise_gazepoint_pupil_windows(
  data,
  pupil_col = NULL,
  time_col = NULL,
  windows = c(0, 500, 1000, 2000),
  group_cols = c("subject", "media_id"),
  include_window_end = FALSE,
  min_valid_samples = 1
)
```

Arguments

<code>data</code>	A Gazepoint master table or processed pupil table.
<code>pupil_col</code>	Optional name of the pupil column to summarise. If NULL, the function detects one of <code>pupil_smoothed</code> , <code>pupil_baseline_corrected</code> , <code>pupil_baseline_percent_change</code> , <code>pupil_interpolated</code> , <code>pupil_for_preprocessing</code> , <code>mean_pupil</code> , <code>pupil</code> , <code>pupil_raw</code> , <code>left_pupil</code> , or <code>right_pupil</code> .
<code>time_col</code>	Optional name of the time column used for assigning samples to windows. If NULL, the function detects one of <code>time_relative_ms</code> , <code>relative_time_ms</code> , <code>event_time_ms</code> , <code>time_ms</code> , <code>time</code> , <code>time_orig</code> , or <code>time_orig_ms</code> .
<code>windows</code>	Window definitions. Either a numeric vector of breakpoints, such as <code>c(0, 500, 1000, 2000)</code> , or a data frame with window start and end columns. Supported names include <code>window_start_ms</code> , <code>window_start</code> , <code>start_ms</code> , or <code>start</code> , and <code>window_end_ms</code> , <code>window_end</code> , <code>end_ms</code> , or <code>end</code> . A <code>window_label</code> or <code>label</code> column is optional.
<code>group_cols</code>	Character vector of grouping columns. Standard roles such as <code>"subject"</code> , <code>"media_id"</code> , <code>"trial"</code> , and <code>"trial_global"</code> are internally standardised when available. Other columns, such as <code>"condition"</code> or <code>"AOI"</code> , can also be used if present in data. Use <code>character(0)</code> for overall window summaries.
<code>include_window_end</code>	Logical. If FALSE, windows are left-closed and right-open: <code>[start, end)</code> . If TRUE, the end point is included: <code>[start, end]</code> . Defaults to FALSE.

`min_valid_samples`

Minimum number of finite pupil samples required for a window to be labelled "valid". Defaults to 1.

Value

A tibble with one row per group-by-window combination present in the data.

Examples

```
pupil_data <- tibble::tibble(
  subject = rep("P1", 6),
  media_id = rep("M1", 6),
  time = c(0, 250, 500, 750, 1000, 1250),
  pupil_smoothed = c(0.00, 0.05, 0.10, 0.08, 0.03, 0.00)
)

pupil_windows <- summarise_gazepoint_pupil_windows(
  pupil_data,
  pupil_col = "pupil_smoothed",
  time_col = "time",
  windows = c(0, 500, 1000, 1500),
  group_cols = c("subject", "media_id")
)

pupil_windows
```

`summarise_gazepoint_workflow`

Summarise a Gazepoint workflow result

Description

Creates a compact one-row summary from a result object returned by `run_gazepoint_workflow()`. This is useful for quickly checking how many rows, file pairs, flagged recordings, exported tables, exported plots, and reports were produced by the workflow.

Usage

```
summarise_gazepoint_workflow(results)
```

Arguments

`results` A named list returned by `run_gazepoint_workflow()`.

Value

A tibble with one row containing workflow-level summary counts.

Examples

```
results <- run_gazepoint_workflow(  
  export_dir = system.file("extdata", "gazepoint_realistic_demo_exports", package = "gp3tools"),  
  output_dir = file.path(tempdir(), "gp3_outputs"),  
  prefix = "study1",  
  save_plots = TRUE,  
  create_report = TRUE  
)  
  
summarise_gazepoint_workflow(results)
```

`summarise_tracking_quality`*Summarise Gazepoint tracking quality*

Description

Summarise Gazepoint tracking quality

Usage

```
summarise_tracking_quality(data, group_cols = "MEDIA_ID")
```

Arguments

<code>data</code>	A Gazepoint data frame.
<code>group_cols</code>	Grouping columns.

Value

A tibble with validity percentages for available validity columns.

`tidy_gazepoint_model_summary`*Create a tidy model summary for manuscript tables*

Description

Create a compact model-summary object from common fitted models used in `gp3tools` workflows. The function combines model metadata, fixed-effect summaries, and optional model diagnostics into one structured object.

Usage

```
tidy_gazepoint_model_summary(
  model,
  model_name = NULL,
  conf_level = 0.95,
  exponentiate = FALSE,
  drop_intercept = FALSE,
  include_diagnostics = TRUE,
  use_dharma = FALSE,
  dharma_simulations = 250,
  seed = 123
)
```

Arguments

model	A fitted model object, or a gp3tools fit object containing a \$model element.
model_name	Optional model label used in returned tables.
conf_level	Confidence level for Wald confidence intervals.
exponentiate	Logical. If TRUE, exponentiate fixed-effect estimates and confidence intervals.
drop_intercept	Logical. If TRUE, remove the intercept from the fixed-effect table.
include_diagnostics	Logical. If TRUE, include model diagnostics when supported.
use_dharma	Logical. If TRUE, request optional DHARMA diagnostics.
dharma_simulations	Number of DHARMA simulations.
seed	Random seed used before DHARMA simulation.

Value

A list with overview, model_info, fixed_effects, diagnostics, and settings. The returned object has class gp3_model_summary.

transform_gazepoint_aoi_empirical_logit

Transform AOI proportions to empirical logits

Description

Convert bounded AOI proportions into empirical logits for linear mixed models, growth-curve analysis, or other approximately Gaussian time-course models.

Usage

```
transform_gazepoint_aoi_empirical_logit(
  data,
  numerator_col = NULL,
  denominator_col = NULL,
  proportion_col = NULL,
  correction = 0.5,
  pseudo_denominator = 1,
  output_col = "aoi_empirical_logit",
  adjusted_proportion_col = "aoi_proportion_adjusted",
  raw_proportion_col = "aoi_proportion_raw",
  numerator_output_col = "aoi_numerator",
  denominator_output_col = "aoi_denominator",
  status_col = "aoi_empirical_logit_status",
  overwrite = FALSE,
  name = "gazepoint_aoi_empirical_logit"
)
```

Arguments

<code>data</code>	A data frame containing AOI proportions or AOI count data.
<code>numerator_col</code>	Optional numerator column, for example number of samples or fixations inside the AOI.
<code>denominator_col</code>	Optional denominator column, for example total valid samples or total fixations in the window.
<code>proportion_col</code>	Optional bounded AOI proportion column. If <code>numerator_col</code> and <code>denominator_col</code> are supplied, the raw proportion is computed from those columns. If only <code>proportion_col</code> is supplied, a pseudo-denominator is used and recorded in the output.
<code>correction</code>	Positive correction constant added to numerator and non-AOI count. The common empirical-logit correction is 0.5.
<code>pseudo_denominator</code>	Positive pseudo-denominator used only when <code>proportion_col</code> is supplied without <code>denominator_col</code> .
<code>output_col</code>	Name of the empirical-logit output column.
<code>adjusted_proportion_col</code>	Name of the adjusted proportion output column.
<code>raw_proportion_col</code>	Name of the raw proportion output column.
<code>numerator_output_col</code>	Name of the numerator output column used in the transformation.
<code>denominator_output_col</code>	Name of the denominator output column used in the transformation.
<code>status_col</code>	Name of the row-level transformation status column.

overwrite	Logical. If FALSE, the function errors when output columns already exist in data.
name	Character label stored in object attributes.

Details

Binomial GLMMs are usually preferable when numerator and denominator counts are available. This helper is intended for sensitivity analyses, GCA-style models, and linear time-course summaries where a transformed AOI proportion is needed.

Value

A tibble with empirical-logit transformation columns added. The object has class `gp3_aoi_empirical_logit_data`.

```
validate_gazepoint_master
```

Validate a Gazepoint master sample table

Description

Performs formal validation checks on a Gazepoint master sample-level table created by [as_gazepoint_master\(\)](#) or [create_gazepoint_master\(\)](#). This function is intended as a gate between master-table construction and more advanced steps such as pupil preprocessing, AOI modelling, or statistical analysis.

Usage

```
validate_gazepoint_master(
  master,
  min_valid_sample_pct = 75,
  max_missing_gaze_pct = 25,
  max_missing_pupil_pct = 50,
  max_offscreen_gaze_pct = 25,
  require_pupil = FALSE,
  require_aoi = FALSE,
  fail_on_error = FALSE
)
```

Arguments

<code>master</code>	A Gazepoint master sample-level table.
<code>min_valid_sample_pct</code>	Minimum acceptable percentage of valid gaze samples. Defaults to 75.
<code>max_missing_gaze_pct</code>	Maximum acceptable percentage of missing gaze samples. Defaults to 25.
<code>max_missing_pupil_pct</code>	Maximum acceptable percentage of missing pupil samples. Defaults to 50.

max_offscreen_gaze_pct	Maximum acceptable percentage of off-screen gaze samples. Defaults to 25.
require_pupil	Logical. If TRUE, the validation fails when no usable pupil column is present. Defaults to FALSE.
require_aoi	Logical. If TRUE, the validation fails when no real AOI samples are present. Defaults to FALSE.
fail_on_error	Logical. If TRUE, the function aborts when one or more validation checks fail. Defaults to FALSE.

Value

A named list with:

summary One-row validation summary.

checks A tibble containing all validation checks.

failed_checks Validation checks with status "fail".

warning_checks Validation checks with status "warning".

column_map Detected column mapping used for validation.

Examples

```

master <- gazepoint_example_master
master <- create_gazepoint_master(
  gaze_data = gazepoint_example_master,
  screen_width_px = 1920,
  screen_height_px = 1080
)

validation <- validate_gazepoint_master(master)

validation$summary
validation$checks

```

write_gazepoint_outputs

Write standard Gazepoint analysis outputs

Description

Convenience wrapper for exporting standard gp3tools outputs such as sampling checks, tracking quality summaries, flagged quality rows, and AOI tables.

Usage

```
write_gazepoint_outputs(  
  sampling = NULL,  
  quality = NULL,  
  flagged_quality = NULL,  
  aoi_table = NULL,  
  output_dir,  
  prefix = "gazepoint",  
  overwrite = TRUE  
)
```

Arguments

sampling	Sampling-rate table, usually from <code>check_sampling_rate()</code> .
quality	Tracking-quality table, usually from <code>summarise_tracking_quality()</code> .
flagged_quality	Flagged quality table, usually from <code>flag_tracking_quality()</code> .
aoi_table	AOI summary table, usually from <code>summarise_gazepoint_aoi()</code> .
output_dir	Folder where CSV files should be written.
prefix	Optional filename prefix.
overwrite	Logical. If FALSE, stop when files already exist.

Value

A tibble with table names and written file paths.

Index

* datasets

- gazeport_example_aoi_geometry, [76](#)
 - gazeport_example_aoi_windows, [77](#)
 - gazeport_example_fixations, [78](#)
 - gazeport_example_master, [79](#)
 - gazeport_example_pupil_windows, [80](#)
- as_gazeport_master, [4](#)
- as_gazeport_master(), [20](#), [54](#), [70](#), [148](#), [156](#)
- audit_gazeport_aoi_coding_matrix, [6](#)
- audit_gazeport_aoi_geometry, [8](#)
- audit_gazeport_aoi_margin_sensitivity, [10](#)
- audit_gazeport_aoi_overlap, [12](#)
- audit_gazeport_aoi_window_denominators, [13](#)
- audit_gazeport_condition_quality_imbalance, [14](#)
- audit_gazeport_design_balance, [15](#)
- audit_gazeport_event_sync, [16](#)
- audit_gazeport_exclusion_flow, [17](#)
- audit_gazeport_gaze_signal_quality, [18](#)
- audit_gazeport_master, [20](#)
- audit_gazeport_master(), [54](#)
- audit_gazeport_post_exclusion_balance, [21](#)
- audit_gazeport_pupil_baseline, [22](#)
- audit_gazeport_pupil_drift, [24](#)
- audit_gazeport_pupil_gaps, [25](#)
- audit_gazeport_pupil_imbalance, [26](#)
- audit_gazeport_pupil_overlap_risk, [27](#)
- audit_gazeport_pupil_reliability, [29](#)
- audit_gazeport_stimulus_luminance, [30](#)
- baseline_correct_gazeport_pupil, [31](#)
- baseline_correct_gazeport_pupil(), [22](#), [135](#), [151](#)
- check_gazeport_file_pairs, [32](#)
- check_gazeport_model_convergence, [33](#)
- check_gazeport_model_overdispersion, [34](#)
- check_gazeport_model_singularity, [34](#)
- check_gazeport_real_data_readiness, [35](#)
- check_sampling_rate, [37](#)
- classify_gazeport_export, [37](#)
- compare_gazeport_nested_models, [38](#)
- compute_gazeport_aoi_transition_matrix, [39](#)
- compute_gazeport_time_varying_transition_matrix, [40](#)
- compute_transition_matrix, [41](#)
- create_gazeport_analysis_decision_audit, [42](#)
- create_gazeport_markovchain_object, [43](#)
- create_gazeport_master, [44](#)
- create_gazeport_master(), [20](#), [54](#), [70](#), [148](#), [156](#)
- create_gazeport_preprocessing_multiverse, [45](#)
- create_gazeport_preprocessing_registry, [46](#)
- create_gazeport_preprocessing_registry(), [73](#)
- create_gazeport_report, [48](#)
- create_gazeport_reporting_checklist, [49](#)
- diagnose_gazeport_gamm, [50](#)
- diagnose_gazeport_glm, [51](#)
- estimate_gazeport_divergence_point, [52](#)
- export_gazeport_master_audit, [54](#)
- export_gazeport_model_tables, [55](#)
- export_gazeport_tables, [56](#)

- fit_gazepoint_aoi_gamm, [57](#)
- fit_gazepoint_aoi_model_sensitivity, [59](#)
- fit_gazepoint_aoi_window_glmm, [60](#)
- fit_gazepoint_gca, [62](#)
- fit_gazepoint_pupil_gamm, [63](#)
- fit_gazepoint_pupil_pfe_gamm, [64](#)
- fit_gazepoint_pupil_window_lmm, [66](#)
- fit_gazepoint_pupil_window_sensitivity, [67](#)
- fit_gazepoint_transition_count_nb_sensitivity, [69](#)
- flag_gazepoint_pupil, [70](#)
- flag_gazepoint_pupil(), [31](#), [81](#), [135](#), [151](#)
- flag_gazepoint_pupil_artifacts, [72](#)
- flag_gazepoint_pupil_artifacts(), [81](#)
- flag_gazepoint_pupil_hampel, [74](#)
- flag_tracking_quality, [75](#)
- gazepoint_example_aoi_geometry, [76](#)
- gazepoint_example_aoi_windows, [77](#)
- gazepoint_example_fixations, [78](#)
- gazepoint_example_master, [79](#)
- gazepoint_example_pupil_windows, [80](#)
- inspect_gazepoint_columns, [80](#)
- interpolate_gazepoint_pupil, [81](#)
- interpolate_gazepoint_pupil(), [26](#), [31](#), [135](#), [151](#)
- interpolate_gazepoint_pupil_pchip, [82](#)
- plot_gazepoint_aoi_gamm, [83](#)
- plot_gazepoint_aoi_transition_matrix, [85](#)
- plot_gazepoint_aoi_verification, [86](#)
- plot_gazepoint_cluster_results, [87](#)
- plot_gazepoint_gca, [89](#)
- plot_gazepoint_model_predictions, [90](#)
- plot_gazepoint_multiverse_results, [92](#)
- plot_gazepoint_pupil_preprocessing, [92](#)
- plot_gazepoint_pupil_status, [94](#)
- plot_gazepoint_pupil_timecourse, [96](#)
- plot_sampling_rate, [97](#)
- plot_tracking_quality, [97](#)
- plot_transition_heatmap, [98](#)
- prepare_gazepoint_aoi_gamm_data, [99](#)
- prepare_gazepoint_aoi_glmm_data, [101](#)
- prepare_gazepoint_aoi_sequences, [102](#)
- prepare_gazepoint_cluster_data, [103](#)
- prepare_gazepoint_eyetools_data, [105](#)
- prepare_gazepoint_eyetrackingr_data, [106](#)
- prepare_gazepoint_fixation_aligned_data, [108](#)
- prepare_gazepoint_gazer_data, [109](#)
- prepare_gazepoint_gca_data, [111](#)
- prepare_gazepoint_hmm_data, [112](#)
- prepare_gazepoint_pupil_gamm_data, [115](#)
- prepare_gazepoint_pupil_window_model_data, [116](#)
- prepare_gazepoint_pupillometryr_data, [113](#)
- prepare_gazepoint_semimarkov_data, [117](#)
- read_gazepoint, [119](#)
- read_gazepoint_folder, [119](#)
- read_gazepoint_summary, [120](#)
- recalibrate_gazepoint_gaze, [121](#)
- recommend_gazepoint_exclusions, [122](#)
- run_gazepoint_aoi_multiverse, [124](#)
- run_gazepoint_cluster_permutation, [125](#)
- run_gazepoint_eyetools_fixation_detection, [126](#)
- run_gazepoint_gazer_crosscheck, [128](#)
- run_gazepoint_model_leave_one_out, [130](#)
- run_gazepoint_pupil_multiverse, [131](#)
- run_gazepoint_workflow, [132](#)
- run_gazepoint_workflow(), [152](#)
- save_gazepoint_plots, [134](#)
- smooth_gazepoint_pupil, [135](#)
- smooth_gazepoint_pupil(), [151](#)
- standardise_gazepoint_names, [136](#)
- summarise_aoi_samples, [137](#)
- summarise_fixations, [137](#)
- summarise_gazepoint_aoi, [138](#)
- summarise_gazepoint_aoi_entries, [138](#)
- summarise_gazepoint_aoi_transitions, [139](#)
- summarise_gazepoint_aoi_trial_features, [141](#)
- summarise_gazepoint_aoi_windows, [142](#)
- summarise_gazepoint_clusters, [143](#)
- summarise_gazepoint_emmeans, [144](#)
- summarise_gazepoint_fixation_trials, [145](#)
- summarise_gazepoint_fixed_effects, [147](#)

summarise_gazepoint_multiverse_results,
148
summarise_gazepoint_pupil, 148
summarise_gazepoint_pupil_trial_features,
149
summarise_gazepoint_pupil_windows, 151
summarise_gazepoint_workflow, 152
summarise_tracking_quality, 153

tidy_gazepoint_model_summary, 153
transform_gazepoint_aoi_empirical_logit,
154

validate_gazepoint_master, 156
validate_gazepoint_master(), 54

write_gazepoint_outputs, 157