Package: tlsR (via r-universe)

May 24, 2026

Type Package

Title Detection and Spatial Analysis of Tertiary Lymphoid Structures

Version 0.3.0

Date 2026-04-02

Description Fast, reproducible detection and quantitative analysis of tertiary lymphoid structures (TLS) in multiplexed tissue imaging. Implements Independent Component Analysis Trace (ICAT) index, local Ripley's K scanning, automated K Nearest Neighbor (KNN)-based TLS detection, and T-cell clusters identification as described in Amiryousefi et al. (2025) <doi:10.1101/2025.09.21.677465>.

License MIT + file LICENSE

URL https://github.com/labsyspharm/tlsR

Depends R (>= 4.0.0)

Imports dbscan (>= 1.1-10), fastICA (>= 1.2-3), FNN (>= 1.1.3), spatstat.explore (>= 3.0-0), spatstat.geom (>= 3.0-0), ggplot2 (>= 3.4.0), rlang (>= 1.0.0), grDevices, graphics, stats, methods

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Ali Amiryousefi [aut, cre] (ORCID: <https://orcid.org/0000-0002-6317-3860>), Jeremiah Wala [aut] (ORCID: <https://orcid.org/0000-0001-6591-1620>), Peter Sorger [ctb] (ORCID: <https://orcid.org/0000-0002-3364-1838>)

Maintainer Ali Amiryousefi <ali_amiryousefi@hms.harvard.edu>

Repository https://cran.r-universe.dev

Date/Publication 2026-04-23 22:05:46 UTC

RemoteUrl https://github.com/cran/tlsR

RemoteRef HEAD

RemoteSha b0b0fc313de9296349ed3d052363c19e058d5aa5

Contents

`tlsR-package` *tlsR: Detection and Spatial Analysis of Tertiary Lymphoid Structures*

Description

Fast, reproducible detection and quantitative analysis of tertiary lymphoid structures (TLS) in multiplexed tissue imaging data.

Typical workflow

1. Load or prepare a named list of data frames (`ldata`), one per tissue sample. Each data frame must contain columns `x`, `y` (spatial coordinates in microns), and `phenotype` (character: `"B cell"` / `"T cell"` / other).
2. Run `detect_TLS` to label B+T co-localised regions.
3. (Optional) Run `scan_clustering` to identify windows of significant immune clustering via local Ripley's L.
4. Run `calc_icat` to score the internal linearity/organisation of each detected TLS.
5. Run `detect_tic` to identify T-cell clusters outside TLS.
6. Use `summarize_TLS` to obtain a tidy summary table.
7. Use `plot_TLS` to produce publication-ready spatial plots.

Author(s)

Maintainer: Ali Amiryousefi <ali_amiryousefi@hms.harvard.edu> (ORCID)

Authors:

- Jeremiah Wala <jeremiah_wala@dfci.harvard.edu> (ORCID)

Other contributors:

- Peter Sorger (ORCID) [contributor]

References

Amiryousefi et al. (2025) doi:10.1101/2025.09.21.677465

See Also

Useful links:

- https://github.com/labsyspharm/tlsR

`calc_icat` *Calculate ICAT (Independent Component Analysis Trace) Index for TLS*

Description

Quantifies the spatial spread and linear organisation of cells within a detected TLS. FastICA is applied to the (x, y) coordinates of TLS cells to estimate independent components; the mixing matrix is used to reconstruct the data, and the ICAT index is defined as the normalised trace-standard-deviation of the reconstructed coordinates.

The index is always non-negative because it measures the average spatial spread per cell rather than the signed trace of the mixing matrix (which can be negative due to ICA sign ambiguity). Higher values indicate a more spatially extended, structured cluster.

Usage

```
calc_icat(patientID, tlsID, ldata = NULL)
```

Arguments

`patientID`	Character. Sample name in `ldata`.
`tlsID`	Numeric or integer. TLS identifier (value of `tls_id_knn` for the TLS of interest).
`ldata`	Named list of data frames, or `NULL` to use the global `ldata` object (deprecated; pass explicitly).

Details

The ICAT index is computed as follows:

1. Centre the (x, y) coordinates of TLS cells.
2. Run `fastICA` with 2 components.
3. Reconstruct $\hat{X} = SA^T + \mu$.
4. Let v_1, v_2 be the marginal variances of $\hat{X}$.
5. $$\text{ICAT} = 100 \times \frac{\sqrt{v_1 + v_2 + 2\sqrt{v_1 v_2}}}{\text{nrow}(X)}$$

If the requested TLS contains fewer than 3 cells, or FastICA does not converge, the function returns `NA_real_` with an informative message rather than throwing an error.

Value

A single non-negative numeric value (the ICAT index), or `NA_real_` if computation is not possible (fewer than 3 cells, or FastICA did not converge).

Examples

```
data(toy_ldata)
ldata <- detect_TLS("ToySample", k = 30, ldata = toy_ldata)
if (max(ldata[["ToySample"]]$tls_id_knn, na.rm = TRUE) > 0) {
  icat <- calc_icat("ToySample", tlsID = 1, ldata = ldata)
  print(icat)
}
```

`detect_tic` *Detect T-cell Immune Clusters (TIC) on the Tissue*

Description

Applies HDBSCAN to T cells that lie outside of previously detected TLS regions to identify spatially compact T-cell clusters (TIC). Phenotype labels `"T cell"` and `"T cells"` are both accepted.

Usage

```
detect_tic(sample, min_pts = 10L, min_cluster_size = 10L, ldata = NULL)
```

Arguments

sample	Character. Sample name in ldata.
min_pts	Integer. HDBSCAN minPts parameter: minimum cluster size (default 10). Smaller values detect more, smaller clusters.
min_cluster_size	Integer. Minimum number of T cells for a HDBSCAN cluster to be retained; smaller clusters are merged back into noise (label 0). Default 10.
ldata	Named list of data frames, or NULL to use the global ldata object (deprecated; pass explicitly).

Value

The input ldata list with the sample data frame augmented by one new column:

tcell_cluster_hdbscan Integer. 0 = noise / not a T-cell cluster; positive integer = TIC cluster ID. Non-T-cell rows receive NA.

Examples

```
data(toy_ldata)
ldata <- detect_TLS("ToySample", k = 30, ldata = toy_ldata)
ldata <- detect_tic("ToySample", ldata = ldata)
table(ldata[["ToySample"]]$tcell_cluster_hdbscan, useNA = "ifany")
```

---

detect_TLS *Detect Tertiary Lymphoid Structures using a KNN-density approach*

Description

Identifies TLS candidates by finding regions of high local B-cell density that also contain a sufficient number of nearby T cells (B+T co-localisation). Phenotype labels "B cell" and "B cells" (and their T-cell equivalents) are both accepted.

Usage

```
detect_TLS(
  LSP,
  ldata,
  k = 30L,
  bcell_density_threshold = 10,
  min_B_cells = 50L,
  min_T_cells_nearby = 10L,
  max_distance_T = 50,
  expand_distance = 80
)
```

Arguments

LSP	Character. Sample name in ldata.
k	Integer. Number of nearest neighbours used for density estimation (default 30, calibrated for 0.325 um/px imaging).
bcell_density_threshold	Numeric. Minimum average 1/k-distance (in microns) for a B cell to be considered locally dense (default 15).
min_B_cells	Integer. Minimum B cells per candidate TLS cluster (default 50).
min_T_cells_nearby	Integer. Minimum T cells within max_distance_T microns of the candidate cluster centre (default 30).
max_distance_T	Numeric. Search radius (microns) for T-cell proximity check (default 50).
expand_distance	Integer. The extended values from the boundary of the deteced B-cells clusters that the T cells are bieng integrated (default 80).
ldata	Named list of data frames, or NULL to use the global ldata object (deprecated; pass explicitly).

Value

The similarly formatted ldata list, with the data frame for LSP augmented by three new columns:

tls_id_knn	Integer. 0 = non-TLS cell; positive integer = TLS cluster ID.
tls_center_x	Numeric. X coordinate of the TLS centre for TLS cells; NA otherwise.
tls_center_y	Numeric. Y coordinate of the TLS centre for TLS cells; NA otherwise.

Examples

```
# Use a 70% sample of the data to keep CRAN check time under 10s.
# TLS detection requires sufficient cell density; 70% preserves
# the spatial structure needed for reliable detection.
# For production use, run on the full dataset (see \donttest{} below).
data(toy_ldata)
set.seed(42)
idx <- sample(nrow(toy_ldata[["ToySample"]]),
              size = floor(0.7 * nrow(toy_ldata[["ToySample"]])))
sub_ldata <- list(ToySample = toy_ldata[["ToySample"]][idx, ])
ldata <- detect_TLS("ToySample", k = 30, ldata = sub_ldata)
table(ldata[["ToySample"]]$tls_id_knn)
plot(ldata[["ToySample"]]$x, ldata[["ToySample"]]$y,
     col = ifelse(ldata[["ToySample"]]$tls_id_knn > 0, "red", "gray"),
     pch = 19, cex = 0.5, main = "Detected TLS (70% sample)")


# Full dataset with default settings
data(toy_ldata)
ldata <- detect_TLS("ToySample", k = 30, ldata = toy_ldata)
table(ldata[["ToySample"]]$tls_id_knn)
```

```
plot(ldata[["ToySample"]]$x, ldata[["ToySample"]]$y,
     col = ifelse(ldata[["ToySample"]]$tls_id_knn > 0, "red", "gray"),
     pch = 19, cex = 0.5, main = "Detected TLS in toy data")
```

---

plot_TLS *Plot Spatial Map of TLS and T-cell Clusters*

Description

Produces a ggplot2 scatter plot of cell positions, coloured by TLS membership, T-cell cluster membership, and background phenotype.

Background (non-TLS, non-TIC) cells are rendered with a lower alpha to keep them visually recessive, while TIC cells are drawn slightly larger than TLS cells so they stand out without dominating the plot.

Usage

```
plot_TLS(
  sample,
  ldata = NULL,
  show_tic = TRUE,
  point_size = 0.5,
  alpha = 0.7,
  bg_alpha = 0.25,
  tic_size_mult = 1.8,
 tls_palette = c("#0072B2", "#009E73", "#CC79A7", "#D55E00", "#56B4E9", "#F0E442"),
  tic_colour = "#E69F00",
  bg_colour = "grey80"
)
```

Arguments

sample	Character. Sample name in ldata.
ldata	Named list of data frames, or NULL to use the global ldata object (deprecated; pass explicitly).
show_tic	Logical. Colour T-cell clusters (if detect_tic has been run) in a distinct colour? Default TRUE.
point_size	Numeric. Base point size for TLS cells and background cells (default 0.5). TIC cells are drawn at point_size * tic_size_mult.
alpha	Numeric. Point transparency for TLS and TIC cells (default 0.7).
bg_alpha	Numeric. Point transparency for background (non-TLS, non-TIC) cells (default 0.25). Reducing this value pushes background cells further behind the foreground structure.
tic_size_mult	Numeric. Multiplier applied to point_size for TIC cells so they appear slightly larger than background and TLS cells (default 1.8).

tls_palette	Character vector of colours for TLS IDs. Recycled if there are more TLS than colours. Default uses a colourblind-friendly palette.
tic_colour	Character. Colour for T-cell cluster cells (default "#E69F00").
bg_colour	Character. Colour for non-TLS, non-TIC cells (default "grey80").

Value

A ggplot object (invisibly). The plot is also printed unless the return value is assigned.

Examples

```
data(toy_ldata)
ldata <- detect_TLS("ToySample", k = 30, ldata = toy_ldata)

  p <- plot_TLS("ToySample", ldata = ldata)
```

---

scan_clustering *Scan Tissue for Local Immune Cell Clustering (Ripley's L Heatmap)*

Description

Applies a sliding-window Ripley's L analysis across the tissue to produce a spatial clustering map. For each window a *K-integral* index is computed as the mean positive excess of the observed L function over its theoretical CSR value. When plot = TRUE a base-graphics spatial map is drawn with LOESS-smoothed L-excess curves and numeric CI labels overlaid inside each qualifying window, plus a legend identifying point and curve colours.

Usage

```
scan_clustering(
  ws = 500,
  sample,
  phenotype = c("T cells", "B cells", "Both"),
  plot = TRUE,
  creep = 1L,
  min_cells = 10L,
  min_phen_cells = 5L,
  label_cex = 1.1,
  ldata = NULL
)
```

Arguments

ws	Numeric. Window side length in microns (default 500).
sample	Character. Sample name in ldata.
phenotype	One of "T cells", "B cells", or "Both".

plot	Logical. Draw the spatial clustering map? (default TRUE).
creep	Integer. Grid density factor; creep = 2 overlaps adjacent windows by half a window width, producing a smoother map (default 1).
min_cells	Integer. Minimum total cell count required in a window before it is analysed (default 10).
min_phen_cells	Integer. Minimum phenotype-specific cell count per window (default 5).
label_cex	Numeric. Base character expansion for the CI numeric labels drawn inside each window (default 1.1). Increase this value if labels appear too small for your screen or output resolution.
ldata	Named list of data frames, or NULL to use the global ldata object (deprecated; pass explicitly).

Details

The K-integral clustering index for window w is:

$$\mathrm{CI}_w = \frac{1}{N_+} \sum_{i:\, L_i > L_{\mathrm{theo},\, i}} (L_i - L_{\mathrm{theo},i})$$

where N_+ is the number of spatial lags where the observed L exceeds the theoretical CSR value.

When plot = TRUE the map shows:

- All cells as small light-grey points.
- Phenotype cells (T cells green, B cells red).
- Navy dashed grid lines marking window boundaries.
- A LOESS-smoothed L-excess curve inside each qualifying window.
- A bold numeric CI label centred in the window.
- A legend identifying all point and curve colours.

When phenotype = "Both" two side-by-side panels are produced - one for B cells and one for T cells - so the two clustering maps can be compared directly on the same spatial layout.

Value

A named list with elements B and/or T (depending on phenotype), each containing the Lest objects for all qualifying windows of that phenotype. Returned invisibly when plot = TRUE.

Examples

```
data(toy_ldata)

  L_models <- scan_clustering(
    ws        = 200,
    sample    = "ToySample",
    phenotype = "B cells",
    plot      = TRUE,
    ldata     = toy_ldata
```

```
)
cat("B-cell windows analysed:", length(L_models$B), "\n")

# Side-by-side B and T cell panels
L_both <- scan_clustering(
  ws        = 200,
  sample    = "ToySample",
  phenotype = "Both",
  plot      = TRUE,
  ldata     = toy_ldata
)
```

`summarize_TLS` *Summarize Detected TLS Across Samples*

Description

Produces a tidy `data.frame` with one row per sample summarising the number of detected TLS, their sizes, and (optionally) ICAT scores.

Usage

```
summarize_TLS(ldata, calc_icat_scores = FALSE)
```

Arguments

`ldata`	Named list of data frames as returned by `detect_TLS` (and optionally `detect_tic`).
`calc_icat_scores`	Logical. Should ICAT scores be computed for each TLS and appended as a list-column? Default `FALSE`.

Value

A `data.frame` with columns:

`sample` Sample name.

`n_TLS` Number of TLS detected.

`total_cells` Total cells in the sample.

`TLS_cells` Number of cells assigned to any TLS.

`TLS_fraction` Fraction of all cells that are TLS cells.

`mean_TLS_size` Mean cells per TLS (`NA` if n_TLS = 0).

`n_TIC` Number of T-cell clusters detected by `detect_tic` (`NA` if not yet run).

`icat_scores` List-column of ICAT scores per TLS (only when `calc_icat_scores = TRUE`).

Examples

```
data(toy_ldata)
ldata <- detect_TLS("ToySample", k = 30, ldata = toy_ldata)
summarize_TLS(ldata)
```

`toy_ldata` *Toy Multiplexed Imaging Data*

Description

A small synthetic dataset mimicking multiplexed tissue imaging data, used in package examples and tests. The list contains one sample named `"ToySample"`.

Usage

```
toy_ldata
```

Format

A named list with one element:

`ToySample` A `data.frame` with the following columns:

- `x` Numeric. X coordinate in microns.
- `y` Numeric. Y coordinate in microns.
- `phenotype` Character. Cell phenotype label. Values are `"B cell"`, `"T cell"`, and `"Other"`.

Source

Synthetically generated for package examples.

References

Amiryousefi et al. (2025) doi:10.1101/2025.09.21.677465

Examples

```
data(toy_ldata)
str(toy_ldata)
table(toy_ldata[["ToySample"]]$phenotype)
```

Index