TSENAT: Tsallis entropy workflow

TSENAT developers

2026-02-01

Overview

This vignette demonstrates how to compute and apply Tsallis entropy to transcript-level expression data using `TSENAT`. It focuses on three practical goals: (1) compute per-sample, per-gene diversity measures across a range of sensitivity parameters `q`; (2) compare those measures between sample groups; and (3) visualize and inspect transcript-level patterns that explain differences.

Motivation

Many genes express multiple isoforms whose relative abundances can be multimodal or heavy-tailed. Standard mean-based comparisons may miss changes in isoform dominance or heterogeneity. Tsallis entropy provides a tunable way to weight rare versus dominant isoforms by varying `q`, producing a "q-curve" that reveals scale-dependent diversity differences.

High-level workflow

1. Preprocess counts and filter low-abundance transcripts.
2. Compute relative transcript proportions within each gene and evaluate Tsallis entropy across one or more `q` values.
3. Summarize per-gene values across samples (mean/median) and test for differences between groups (Wilcoxon or permutations).
4. Visualize q-curves and inspect transcript-level counts for top genes.

What is Tsallis entropy?

Tsallis entropy is a one-parameter family of diversity measures that generalizes Shannon entropy via a sensitivity parameter `q`. For a discrete probability vector $p = (p_1, \dots, p_n)$ it is defined as

$$S_q(p) = \frac{1 - \sum_{i=1}^{n} p_i^q}{q - 1}.$$

Why use Tsallis entropy for gene-expression data?

Tsallis entropy provides a compact, flexible summary of isoform-level heterogeneity that is particularly useful when biological signals act at different abundance scales; combined with careful preprocessing and resampling-based uncertainty quantification it is a practical and interpretable tool for transcriptomics analyses. Key advantages:

- Tunable sensitivity: changing `q` shifts emphasis between rare and dominant isoforms. Use small `q` (< 1) to probe rare-isoform heterogeneity and larger `q` (> 1) to focus on dominant-isoform behaviour; plotting `S_q` across a compact `q` grid (a “q-curve”) reveals scale-dependent differences between conditions.
- Interpretability: Tsallis maps to several familiar indices in limits (Shannon at `q=1`, richness-like at `q=0`, Gini–Simpson at `q=2`), so it unifies multiple perspectives on isoform diversity within one framework.
- Robust summary and diagnostics: combined with Hill numbers and q-curves, `S_q` supports concise reporting (effective isoform counts, slope/AUC summaries) and targeted follow-up (transcript-level plots).

Essential limiting cases:

- Limit $q \to 1$: Shannon entropy, $\lim_{q\to 1} S_q(p) = -\sum_i p_i \log p_i$.
- $q = 0$: richness-like (number of expressed isoforms minus one).
- $q = 2$: Gini–Simpson / collision index, $S_2 = 1 - \sum_i p_i^2$.
- Uniform maximum: for m expressed isoforms, $S_{q,\max}(m) = \frac{1-m^{1-q}}{q-1}$ and normalized $\tilde{S}_q = S_q / S_{q,\max}(m)$.

Use cases and interpretation:

- A gene with group separation only at low `q` suggests differences in low-abundance (rare) isoforms; separation at high `q` indicates changes in dominant isoforms. When both regimes differ, the q-curve will show divergence across the entire `q` range.
- Report both `S_q` (or `\tilde S_q`) and Hill numbers `D_q` for reader clarity: `D_q` converts entropies into an “effective number of isoforms” that is easy to interpret.

Example dataset

An example dataset is included for demonstration. Load the package and data:

```r
# Load packages
suppressPackageStartupMessages({
  library(TSENAT)
  library(ggplot2)
  library(SummarizedExperiment)
  library(mgcv)
})

## Note: `map_coldata_to_se()` is provided by the package and will be used below.

# Load required files
coldata_tsv <- system.file("extdata", "coldata.tsv", package = "TSENAT")
tx2gene_tsv <- system.file("extdata", "tx2gene.tsv", package = "TSENAT")

# Load dataset
data(tcga_brca_luma_dataset)

# Extract gene names and read count data (do not reference ts_se yet)
genes <- tcga_brca_luma_dataset[, 1]
readcounts <- tcga_brca_luma_dataset[, -1]

# Assign transcript IDs as rownames of `readcounts` so downstream transcript-level
# plotting functions can use them.
txmap <- utils::read.delim(tx2gene_tsv, header = TRUE, stringsAsFactors = FALSE)
```

```r
rownames(readcounts) <- as.character(txmap$Transcript)

# Read sample metadata
coldata_df <- read.table(coldata_tsv, header = TRUE, sep = "\t", stringsAsFactors = FALSE)

# Check gene names
head(genes)
#> [1] "MXRA8"   "MXRA8"   "MXRA8"   "MXRA8"   "MXRA8"   "C1orf86"

# Check read count dataset
dim(readcounts)
#> [1] 1100   40
head(readcounts[, 1:5])
#>           TCGA-A7-A0CH_N TCGA-A7-A0CH_T TCGA-A7-A0D9_N TCGA-A7-A0D9_T
#> MXRA8.1          2858.04         743.56         812.59           0.00
#> MXRA8.2           127.82          21.28          50.87          38.21
#> MXRA8.3           370.22          94.38         368.76          98.12
#> MXRA8.4          7472.00        3564.87        7647.76        3236.28
#> MXRA8.5            25.44          16.91           4.07           7.55
#> C1orf86.1           0.00           0.00           0.00           0.00
#>           TCGA-A7-A0DB_T
#> MXRA8.1          1508.90
#> MXRA8.2            30.10
#> MXRA8.3           167.89
#> MXRA8.4          6360.80
#> MXRA8.5            34.74
#> C1orf86.1           0.00

# Check the metadata
head(coldata_df)
#>           Sample Condition
#> 1 TCGA-A7-A0CH_N    Normal
#> 2 TCGA-A7-A0CH_T     Tumor
#> 3 TCGA-A7-A0D9_N    Normal
#> 4 TCGA-A7-A0D9_T     Tumor
#> 5 TCGA-A7-A0DB_N    Normal
#> 6 TCGA-A7-A0DB_T     Tumor
```

Data filtering and preprocessing

As a first step, before doing the diversity calculation, you might want to filter out genes with a low overall expression or limit the analysis to transcripts with a sufficient minimum expression level. Expression estimates of transcript isoforms with zero or low expression might be highly variable. For more details on the effect of transcript isoform prefiltering on differential transcript usage, see this paper.

Filter transcripts with fewer than 5 reads in more than 5 samples and update the `genes` vector accordingly.

```r
tokeep <- rowSums(readcounts > 5) > 5

readcounts <- readcounts[tokeep, ]
genes <- genes[tokeep]
```

We filter lowly-expressed transcripts to reduce noise and improve the stability of diversity estimates.

Compute normalized Tsallis entropy for a single `q` value.

```
# compute Tsallis entropy for q = 0.1
q <- 0.1
ts_se <- calculate_diversity(readcounts, genes, method = "tsallis", q = q, norm = TRUE)
head(assay(ts_se)[1:5, 1:5])
#>         TCGA-A7-A0CH_N TCGA-A7-A0CH_T TCGA-A7-A0D9_N TCGA-A7-A0D9_T
#> MXRA8        0.8616275      0.6609111      0.8156972      0.6633580
#> C1orf86      0.0000000      0.0000000      0.0000000      0.0000000
#> PDPN         0.3486679      0.3395884      0.0000000      0.3481643
#> ALDH4A1      0.8111864      0.4667361      0.8621495      0.5069312
#> HNRNPR       0.5204909      0.6683829      0.7395887      0.6537834
#>         TCGA-A7-A0DB_T
#> MXRA8        0.8036333
#> C1orf86      0.0000000
#> PDPN         0.3467400
#> ALDH4A1      0.5046596
#> HNRNPR       0.8287792
```

Map sample metadata (if available) into the `SummarizedExperiment` so plotting functions can use `sample_type` for grouping.

```
if (!is.null(coldata_df)) {
  ts_se <- map_coldata_to_se(ts_se, coldata_df)
}
```

Differential analysis

This section summarizes practical guidance for hypothesis testing and reporting when comparing diversity measures across sample groups.

The package exposes pragmatic pairwise tests for comparing per-gene diversity summaries between two groups. Two common choices are Wilcoxon-based tests and label-shuffle tests. Guidance:

- Wilcoxon rank-sum (unpaired) / Wilcoxon signed-rank (paired):
  - Use when you want a distribution-free test comparing central tendency (ranks/median) and sample sizes are moderate (typically 10 per group for stable asymptotic p-values).
  - Assumptions: independent observations within groups (or matched pairs for the signed-rank test); exchangeability under the null.
  - Handle ties: software returns approximate p-values when many ties exist; consider permutation p-values when ties or discreteness are extreme (small counts, many zeros).
  - Report effect sizes (median difference, Hodges–Lehmann estimator) and confidence intervals where possible; do not rely on p-values alone.
- Label-shuffle tests:
  - Use when sample sizes are small, distributional assumptions are questionable, or you prefer an exact/empirical null constructed from the observed data. Randomly shuffle group labels many times and recompute the test statistic (difference in medians or means).
  - Exchangeability requirement: permutations are valid when labels are exchangeable under the null (e.g., independent samples). For paired designs use paired permutations that shuffle within pairs or use sign flips for paired differences.

  - Practical settings: set `randomizations` to 1000 for routine use and 5000+ when estimating small p-values or when applying FDR across many genes. When possible, compute exact permutations (all labelings) for very small datasets.

Here we will use Wilcoxon, which is the adecuate option for speed and when sample sizes support asymptotic approximations; use permutation when small-sample accuracy or exact control of the null distribution is important.

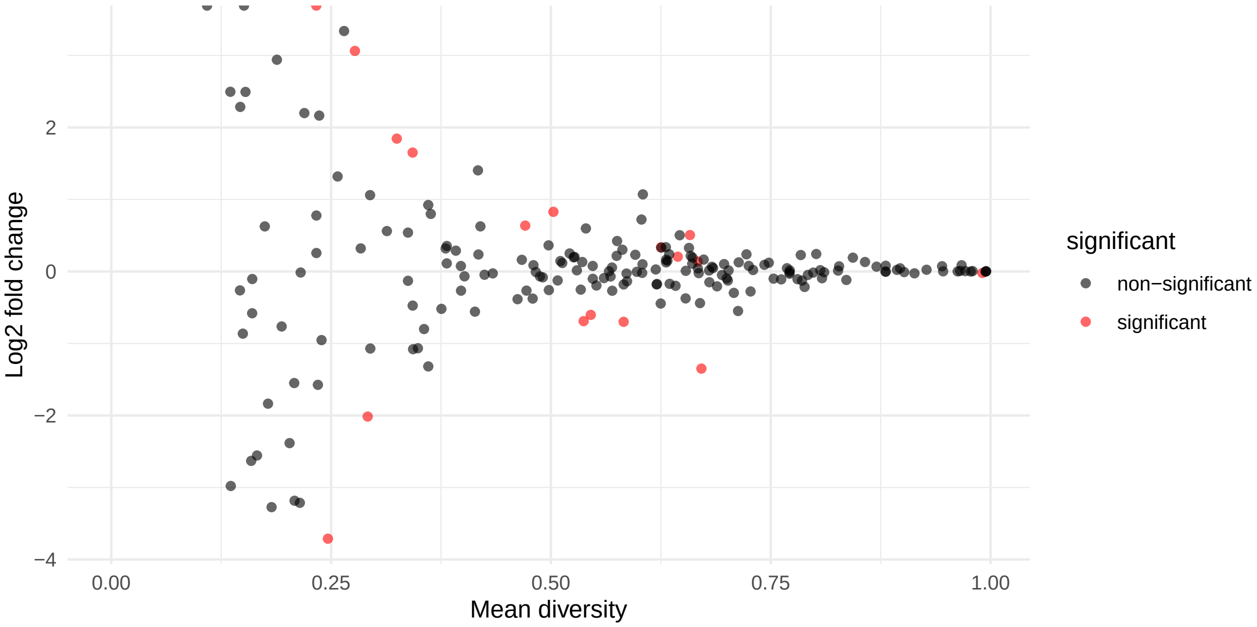
Before running differential tests we summarize per-gene diversity values across samples and explicitly define sample groups. The example below performs a pairwise comparison between two conditions (e.g., `Normal` vs `Tumor`) using a non-parametric Wilcoxon test on the per-gene diversity summaries. Adjust the `control` and `test` parameters in `calculate_difference()` to suit your experimental design.

```r
# create a sample grouping vector inferred from sample names
# account for per-q column names like 'Sample_q=0.01'
sample_base_names <- sub("_q=.*", "", colnames(assay(ts_se)))
samples <- as.character(SummarizedExperiment::colData(ts_se)$sample_type)

# prepare diversity table as data.frame with gene names in first column
div_df <- as.data.frame(assay(ts_se))
div_df <- cbind(genes = rowData(ts_se)$genes, div_df)

# samples are matched pairs (Normal/Tumor), so use a paired test
res <- calculate_difference(div_df, samples, control = "Normal",
                            method = "mean", test = "wilcoxon",
                            paired = TRUE)
# sort results by adjusted p-value
if ("adjusted_p_values" %in% colnames(res)) {
  res <- res[order(res$adjusted_p_values), , drop = FALSE]
}
head(res)
#>        genes Tumor_mean Normal_mean mean_difference log2_fold_change
#> 6   C1orf213  0.3783117   0.9639752     -0.58566342      -1.34942042
#> 17     S1PR1  0.9836170   0.9976878     -0.01407073      -0.02049165
#> 87      MBD2  0.5199277   0.1655574      0.35437027       1.65097918
#> 98      OSR1  0.7722536   0.5440455      0.22820806       0.50534727
#> 101    GFPT1  0.1156779   0.4675860     -0.35190810      -2.01511871
#> 126     KIF9  0.5729707   0.3685433      0.20442740       0.63662729
#>     raw_p_values adjusted_p_values
#> 6   0.0002357454         0.0269389
#> 17  0.0007285418         0.0269389
#> 87  0.0006535852         0.0269389
#> 98  0.0005801974         0.0269389
#> 101 0.0008919014         0.0269389
#> 126 0.0006356242         0.0269389
```

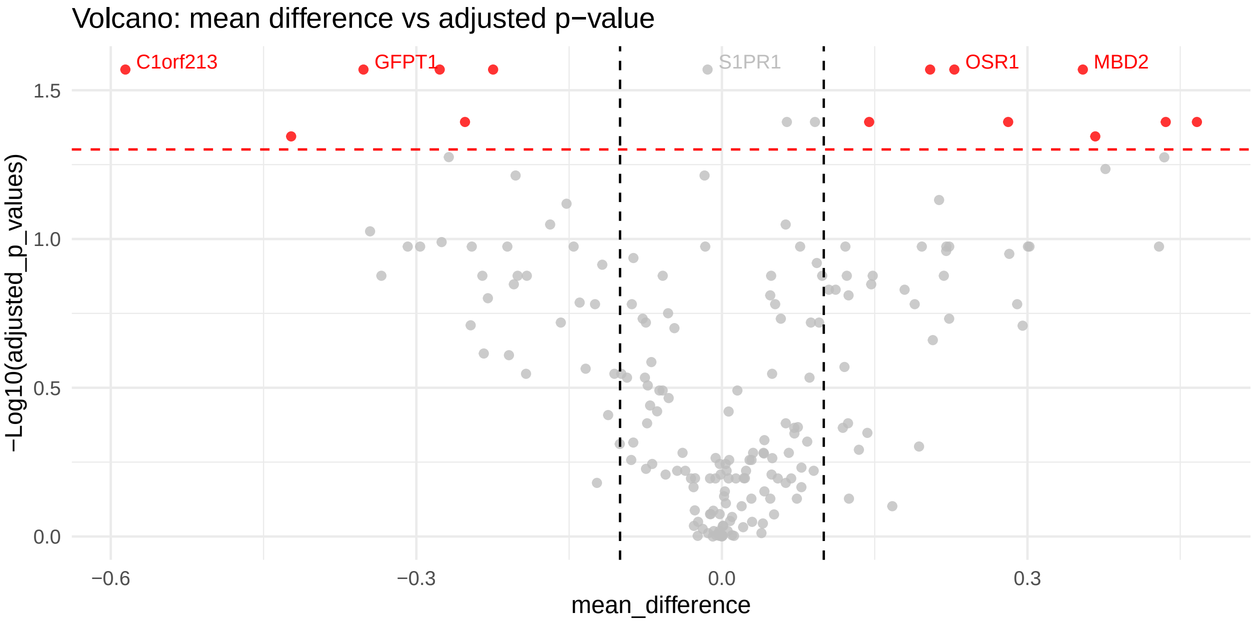
Generate diagnostic plots to summarize per-gene effect sizes:

- The MA plot shows the relationship between mean expression and log-fold change, helping to spot genes with large effect sizes across expression levels.
- The volcano plot displays effect size versus significance, useful for highlighting candidates by thresholding magnitude and adjusted p-value.

```r
# MA plot using helper
p_ma <- plot_ma(res)
print(p_ma)
#> Warning: Removed 1 row containing missing values or values outside the scale
#> range (`geom_point()`).
```



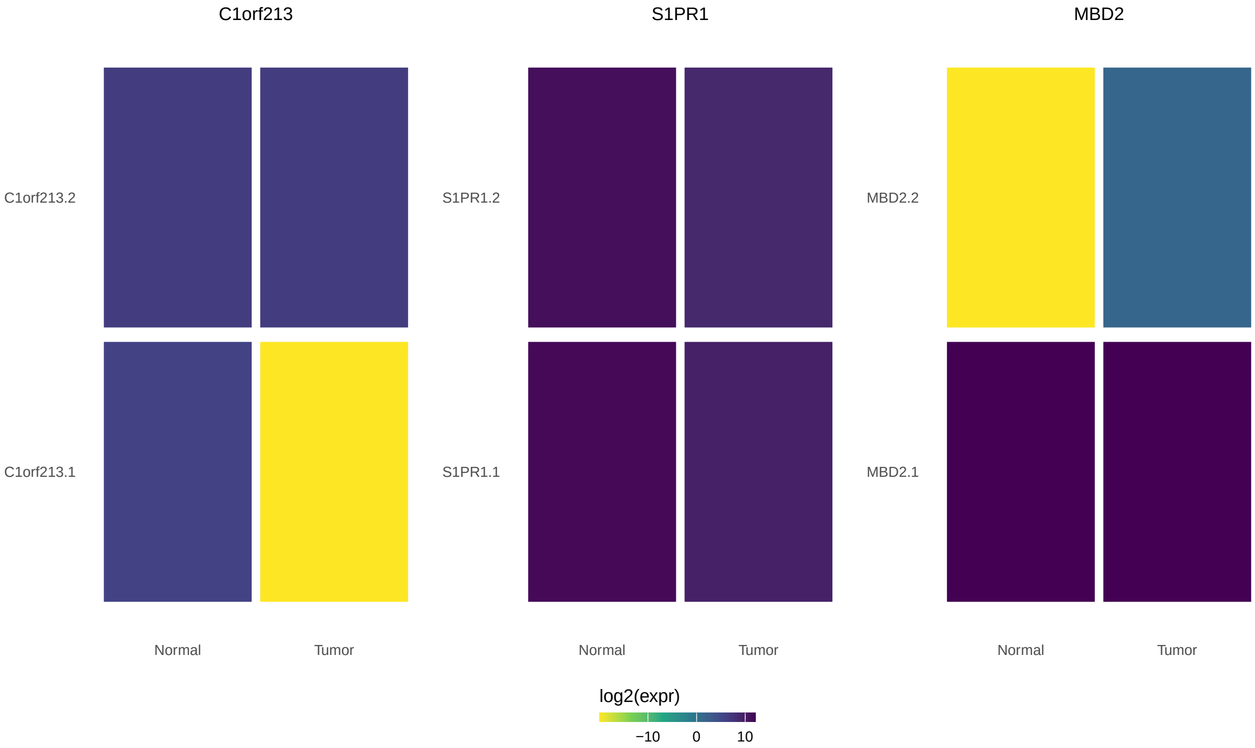
```r
# Volcano plot: mean difference vs -log10(adjusted p-value)
if (all(c("mean_difference", "adjusted_p_values") %in% colnames(res))) {
  p_volcano <- plot_volcano(res)
  print(p_volcano)
}
```



Plot top 3 genes from the single-q differential analysis

Here we visualize transcript-level expression for the most significant genes identified in the previous step. The `plot_top_transcripts()` helper expects transcript-level counts (rows = transcripts) with rownames matching the `tx2gene` mapping. We plot the top three genes by significance to inspect isoform-level patterns that may explain the diversity differences.

```
sig_res <- res[res$adjusted_p_values < 0.05, , drop = FALSE]
top_genes <- head(sig_res$genes, 3)
sample_base_names <- sub("_q=.*", "", colnames(assay(ts_se)))
samples_vec <- as.character(SummarizedExperiment::colData(ts_se)$sample_type)
p_comb <- plot_top_transcripts(readcounts, gene = top_genes,
                               samples = samples_vec, tx2gene = txmap,
                               top_n = NULL)
print(p_comb)
```



Inspect transcript-level counts for the top genes to understand isoform patterns that may drive diversity differences.

Compare between q values

Compute normalized Tsallis entropy for two q values (0.1 and 2) to compare scale-dependent behavior.

```
# compute Tsallis entropy for q = 1 (normalized)
q <- c(0.1, 2)
ts_se <- calculate_diversity(readcounts, genes, method = "tsallis",
                             q = q, norm = TRUE)
head(assay(ts_se)[1:5, 1:5])
```

```
#>          TCGA-A7-A0CH_N_q=0.1 TCGA-A7-A0CH_T_q=0.1 TCGA-A7-A0D9_N_q=0.1
#> MXRA8               0.8616275            0.6063802            0.6609111
#> C1orf86             0.0000000            0.0000000            0.0000000
#> PDPN                0.3486679            0.6619058            0.3395884
#> ALDH4A1             0.8111864            0.7478309            0.4667361
#> HNRNPR              0.5204909            0.6176153            0.6683829
#>          TCGA-A7-A0D9_T_q=0.1 TCGA-A7-A0DB_T_q=0.1
#> MXRA8               0.3925463            0.8156972
#> C1orf86             0.0000000            0.0000000
#> PDPN                0.5044987            0.0000000
#> ALDH4A1             0.2841620            0.8621495
#> HNRNPR              0.6856195            0.7395887
```

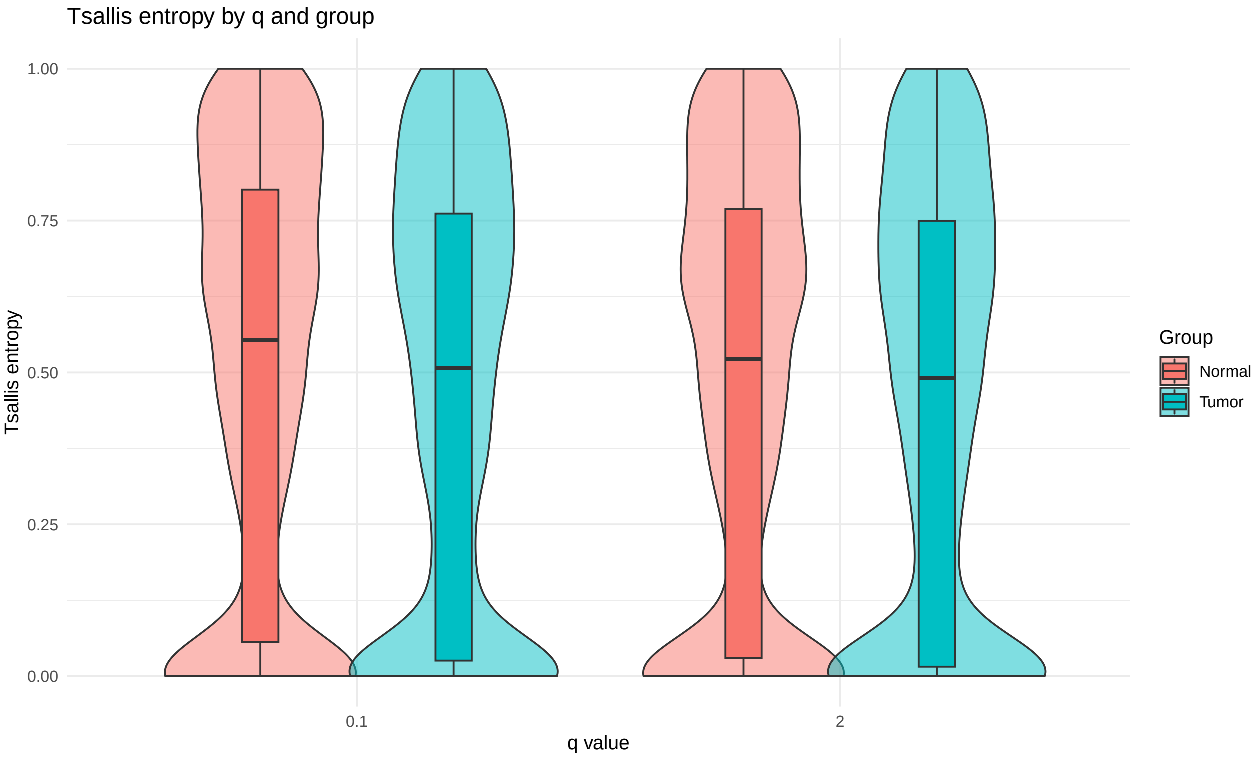
It allows to compare values across `q` for the same gene to see whether diversity differences are scale-dependent (e.g., significant at `q=0.1` but not at `q=2` suggests rare-isoform-driven changes).

Map optional sample metadata into the multi-q `SummarizedExperiment` so plotting helpers have access to `sample_type` for grouping.

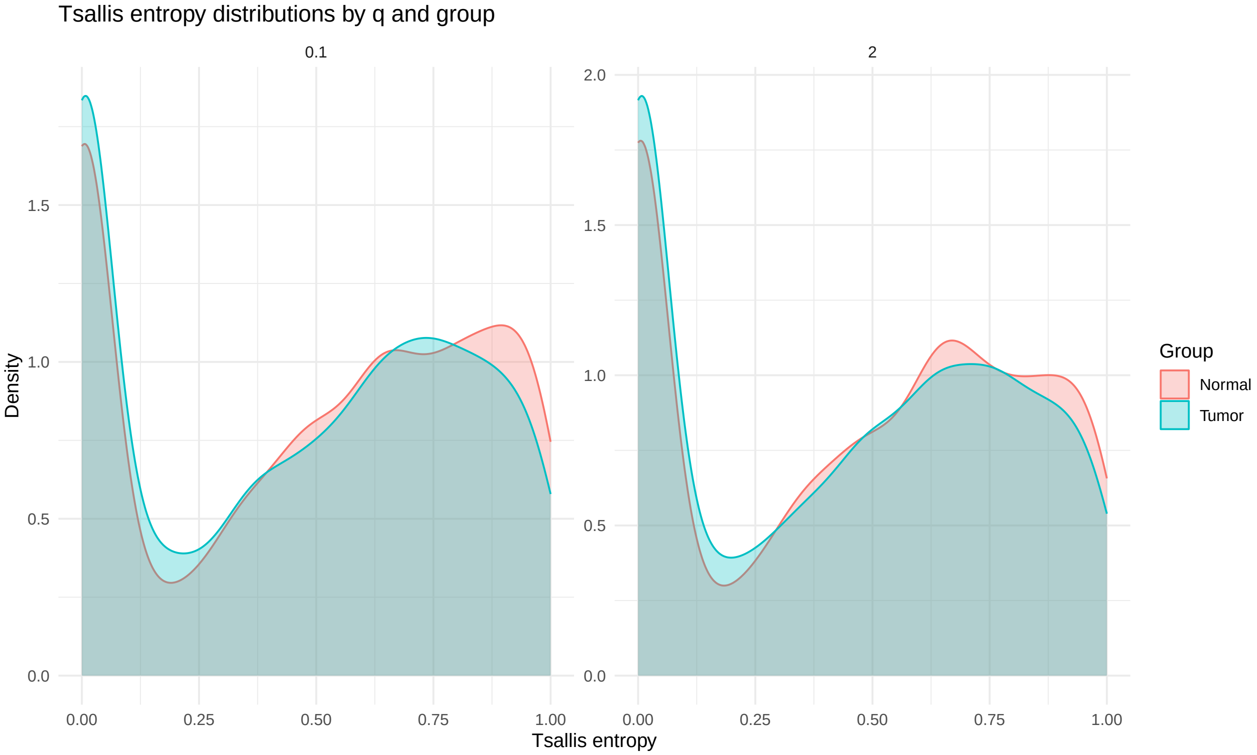
```
if (!is.null(coldata_df)) {
  ts_se <- map_coldata_to_se(ts_se, coldata_df)
}
```

Create violin and density plots summarizing diversity across multiple `q` values to show distributional differences between groups.

```
## Use package plot helpers for multi-q violin and density
p_violin <- plot_tsallis_violin_multq(ts_se, assay_name = "diversity")
print(p_violin)
```



```r
p_density <- plot_tsallis_density_multq(ts_se, assay_name = "diversity")
print(p_density)
```



Linear-model interaction and shape tests for q-curves

Now we will compute normalized Tsallis entropy across a sequence of `q` values.

```r
# compute Tsallis entropy for a sequence of values (normalized)
qvec <- seq(0.01, 2, by = 0.1)
ts_se <- calculate_diversity(readcounts, genes, method = "tsallis",
                             q = qvec, norm = TRUE)
head(assay(ts_se)[1:5, 1:5])
#>         TCGA-A7-A0CH_N_q=0.01 TCGA-A7-A0CH_T_q=0.01 TCGA-A7-A0D9_N_q=0.01
#> MXRA8               0.9840684             0.8500785             0.7527194
#> C1orf86             0.0000000             0.0000000             0.0000000
#> PDPN                0.3348515             0.3502196             0.3659083
#> ALDH4A1             0.9759153             0.7976183             0.6982381
#> HNRNPR              0.5653272             0.5164881             0.4849368
#>         TCGA-A7-A0D9_T_q=0.01 TCGA-A7-A0DB_T_q=0.01
#> MXRA8               0.6818220             0.6303117
#> C1orf86             0.0000000             0.0000000
#> PDPN                0.3818895             0.3981326
#> ALDH4A1             0.6447807             0.6181672
#> HNRNPR              0.4655343             0.4548022
```

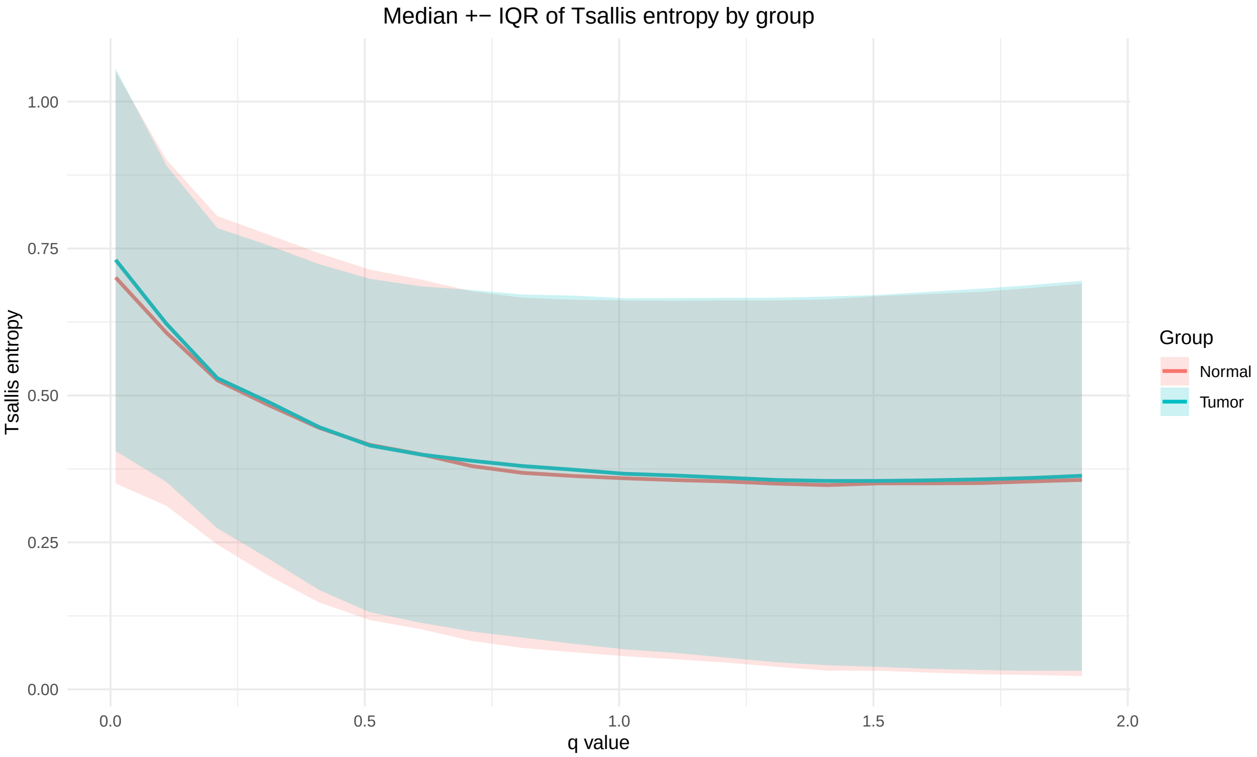
Map the sample metadata into the multi-q `SummarizedExperiment` as before.

```
if (!is.null(coldata_df)) {
  ts_se <- map_coldata_to_se(ts_se, coldata_df)
}
```

With the q-sequence we can produces a q-curve per sample and gene. These curves show how diversity emphasis shifts from rare to dominant isoforms as `q` increases and form the basis for interaction tests. The q-curve shows entropy as a function of `q`. Diverging curves between groups indicate scale-dependent diversity differences. Separation at low `q` implies differences in rare isoforms; separation at high `q` signals differences in dominant isoforms.

Plot the Tsallis q-curve (entropy vs q) to visualize how diversity changes with `q` across sample groups.

```
# q-curve: median ± IQR across q values by group
p3 <- plot_tsallis_q_curve(readcounts, genes, q_values = qvec)
print(p3)
```



Comparing entire q-curves between groups asks whether the relationship between diversity and `q` differs by condition. The package implements three complementary approaches; below are practical notes to guide choice and parameter selection.

- Linear interaction model (`entropy ~ q * group`):
  - Interprets group differences as slope (q-by-group interaction) differences across the evaluated q-grid. Use when the q-curve is approximately linear over the chosen range and sample sizes are modest.
  - Advantages: simple, interpretable interaction coefficient, fast.
  - Caveats: will miss localized nonlinear differences (e.g., only at low q values).
- GAM-based comparison (`mgcv`):

  - Fit smooth functions of `q` with group-specific terms (e.g., a common smooth plus group-by-smooth deviations) and compare nested models with an approximate F-type test (anova.gam).
  - Choose spline basis dimension `k` relative to the number of distinct `q` values (keep `k` small for sparse grids). Inspect diagnostic plots and concur that fitted smooths reflect biological signal rather than overfitting noise.
  - Requires sufficient observations per gene across samples and q values to estimate smooth terms reliably.

- FPCA-based test (functional PCA on q-curves):
  - Treat each sample's q-curve as a functional object, compute principal components across q, and test group differences on leading PC scores.
  - Fast and robust when dominant curve modes capture group differences, but it reduces the curve to a few components and may miss localized effects confined to a narrow q-range.
  - Address missing q points by sensible imputation (column means or spline interpolation) before PCA; require minimal q coverage across samples to obtain stable PCs.

This test fits a simple linear model per gene to assess whether the slope of the q-curve differs between sample groups. It can help identify genes whose response to the entropy scale parameter `q` is group-specific, indicating changes in isoform-dominance dynamics rather than only an overall shift in diversity.

```
# ensure the SummarizedExperiment contains sample names with group suffixes
# (the function infers group from sample name suffix _N -> Normal)
lm_res <- calculate_lm_interaction(ts_se, sample_type_col = "sample_type", min_obs = 8)
#> [calculate_lm_interaction] method=linear
#> [calculate_lm_interaction] parsed samples and groups; starting per-gene fits
head(lm_res)
#>      gene p_interaction adj_p_interaction
#> 1 PPP2R1A     0.7454960         0.9997906
#> 2    MDH1     0.7721388         0.9997906
#> 3    ZZZ3     0.7844040         0.9997906
#> 4   ACACB     0.7869660         0.9997906
#> 5  CAPZA2     0.7916957         0.9997906
#> 6  EFTUD2     0.7948379         0.9997906
```

Practical notes and recommendations

Quick checklist (what to do and why):

- Preprocess: filter transcripts never observed or add a small, documented pseudocount (e.g., +1) — this stabilizes proportions and avoids undefined powers when computing `p_i^q`.
- Choose `q` grid: pick a compact set spanning rare → dominant regimes (suggested: 0.1, 0.2, 0.5, 1, 1.5, 2). Plot q-curves to inspect where groups diverge rather than relying on a single `q`.
- Reporting scale: present Hill numbers `D_q` (intuitive "effective isoforms") and/or normalized entropy $\tilde{S}_q$; report both effect size (median/mean difference) and variability (CI or bootstrap SE).
- Hypothesis testing: use Wilcoxon for speed (asymptotic) and label-shuffle (permutation) for small samples or exact inference. For permutations use `randomizations` 1000 (5000 when estimating very small p-values); use paired permutations for matched designs.
- Multiple testing: apply FDR correction (Benjamini–Hochberg) and show raw/adjusted p-values alongside effect sizes so readers can judge practical relevance.
- Interaction testing: explore slopes/AUC first; use linear interaction for simple slope differences, GAMs for nonlinear shapes, and FPCA for fast low-dimensional summaries — always report key parameters (`k`, `randomizations`).

- Design & technical tips: prefer paired/matched designs when available, ensure adequate biological replication, use accurate quantifiers (Salmon/kallisto or UMI-based methods), and correct for confounders or include them in models.
- Design examples: time-series or longitudinal sampling to capture dynamics; perturbation gradients (dose–response) to reveal transitions; single-cell or spatial (UMI-based) to study heterogeneity; paired/matched designs to control subject-level variability.
- Technical checklist: use accurate transcript quantifiers, aim for sufficient sequencing depth for isoform inference, filter or pseudocount low-abundance transcripts, and apply batch/confounder correction prior to downstream testing. Test across a compact `q` grid and report the grid.

Session info

```
sessionInfo()
#> R version 4.5.2 (2025-10-31)
#> Platform: x86_64-conda-linux-gnu
#> Running under: Ubuntu 22.04.5 LTS
#> 
#> Matrix products: default
#> BLAS/LAPACK: /home/nouser/miniconda3/lib/libopenblasp-r0.3.30.so;  LAPACK version 3.12.0
#> 
#> locale:
#>  [1] LC_CTYPE=es_ES.UTF-8       LC_NUMERIC=C              
#>  [3] LC_TIME=de_DE.UTF-8        LC_COLLATE=es_ES.UTF-8    
#>  [5] LC_MONETARY=de_DE.UTF-8    LC_MESSAGES=es_ES.UTF-8   
#>  [7] LC_PAPER=de_DE.UTF-8       LC_NAME=C                 
#>  [9] LC_ADDRESS=C               LC_TELEPHONE=C            
#> [11] LC_MEASUREMENT=de_DE.UTF-8 LC_IDENTIFICATION=C       
#> 
#> time zone: Europe/Berlin
#> tzcode source: system (glibc)
#> 
#> attached base packages:
#> [1] stats4    stats     graphics  grDevices utils     datasets  methods  
#> [8] base     
#> 
#> other attached packages:
#>  [1] mgcv_1.9-4                  nlme_3.1-168               
#>  [3] SummarizedExperiment_1.40.0 Biobase_2.70.0             
#>  [5] GenomicRanges_1.62.1        Seqinfo_1.0.0              
#>  [7] IRanges_2.44.0              S4Vectors_0.48.0           
#>  [9] BiocGenerics_0.56.0         generics_0.1.4             
#> [11] MatrixGenerics_1.22.0       matrixStats_1.5.0          
#> [13] ggplot2_4.0.1               TSENAT_0.1                 
#> 
#> loaded via a namespace (and not attached):
#>  [1] tidyr_1.3.2         SparseArray_1.10.8  lattice_0.22-7     
#>  [4] digest_0.6.39       magrittr_2.0.4      evaluate_1.0.5     
#>  [7] grid_4.5.2          RColorBrewer_1.1-3  fastmap_1.2.0      
#> [10] Matrix_1.7-4        ggrepel_0.9.6       tinytex_0.58       
#> [13] purrr_1.2.1         viridisLite_0.4.2   scales_1.4.0       
#> [16] abind_1.4-8         cli_3.6.5           rlang_1.1.7        
#> [19] XVector_0.50.0      splines_4.5.2       withr_3.0.2        
#> [22] DelayedArray_0.36.0 yaml_2.3.12         otel_0.2.0
#> [25] S4Arrays_1.10.1     tools_4.5.2         dplyr_1.1.4
#> [28] vctrs_0.7.1         R6_2.6.1            lifecycle_1.0.5
#> [31] pkgconfig_2.0.3     pillar_1.11.1       gtable_0.3.6
#> [34] glue_1.8.0          Rcpp_1.1.1          xfun_0.56
#> [37] tibble_3.3.1        tidyselect_1.2.1    knitr_1.51
#> [40] farver_2.1.2        patchwork_1.3.2     htmltools_0.5.9
#> [43] rmarkdown_2.30      labeling_0.4.3      compiler_4.5.2
#> [46] S7_0.2.1
```