

Package: fpROC (via r-universe)

May 14, 2026

Type Package

Title Fast Partial Receiver Operating Characteristic (ROC) Test for Ecological Niche Modeling

Version 0.1.0

Maintainer Luis Osorio-Olvera <luismurao@gmail.com>

Description Provides optimized 'C++' code for computing the partial Receiver Operating Characteristic (ROC) test used in niche and species distribution modeling. The implementation follows Peterson et al. (2008) <[doi:10.1016/j.ecolmodel.2007.11.008](https://doi.org/10.1016/j.ecolmodel.2007.11.008)>. Parallelization via 'OpenMP' was implemented with assistance from the 'DeepSeek' Artificial Intelligence Assistant (<<https://www.deepseek.com/>>).

License GPL-3

Encoding UTF-8

LazyData true

NeedsCompilation yes

SystemRequirements GNU make, OpenMP, TBB

URL <https://luismurao.github.io/fpROC/>

BugReports <https://github.com/luismurao/fpROC/issues>

Imports Rcpp (>= 1.0.7), stats, terra

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.3.2

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Config/pak/sysreqs libgdal-dev gdal-bin libgeos-dev make libproj-dev libsqlite3-dev

Repository <https://luismurao.r-universe.dev>

Date/Publication 2025-07-18 00:13:50 UTC

RemoteUrl <https://github.com/luismurao/fproc>

RemoteRef HEAD

RemoteSha 968b62119d48b758202a4917b69e126faef6ca7e

Contents

auc_metrics	2
auc_parallel	3
summarize_auc_results	5
trap_roc	7

Index	9
--------------	----------

auc_metrics	<i>Calculate Partial and complete Area Under the Curve (AUC) Metrics</i>
-------------	--

Description

Computes partial AUC ratios between model predictions and random curves at a specified threshold, with options for sampling and iterations. Handles both numeric vectors and SpatRaster inputs.

Usage

```
auc_metrics(
  test_prediction,
  prediction,
  threshold = 5,
  sample_percentage = 50,
  iterations = 500,
  compute_full_auc = TRUE
)
```

Arguments

test_prediction	Numeric vector of test prediction values (e.g., model outputs)
prediction	Numeric vector or SpatRaster object containing prediction values
threshold	Percentage threshold for partial AUC calculation (default = 5)
sample_percentage	Percentage of test data to sample (default = 50)
iterations	Number of iterations for estimating bootstrap statistics (default = 500)
compute_full_auc	Logical. If TRUE, the complete AUC values will be computed

Details

Partial ROC is calculated following Peterson et al. (2008; [doi:10.1016/j.ecolmodel.2007.11.008](https://doi.org/10.1016/j.ecolmodel.2007.11.008)). The function calculates partial AUC ratios by:

1. Validating input types and completeness
2. Handling NA values and SpatRaster conversion

3. Checking for prediction variability
4. Computing AUC metrics using optimized C++ code

When prediction values have no variability (all equal), the function returns NA values with a warning.

Value

A list containing:

- If input has no variability: List with NA values for AUC metrics
- Otherwise: Matrix of AUC results.

References

Peterson, A.T. et al. (2008) Rethinking receiver operating characteristic analysis applications in ecological niche modeling. *Ecol. Modell.*, 213, 63–72.

Examples

```
# With numeric vectors
test_data <- rnorm(100)
pred_data <- rnorm(100)
result <- fpROC::auc_metrics(test_prediction = test_data, prediction = pred_data)

# With SpatRaster
library(terra)
r <- terra::rast(ncol=10, nrow=10)
values(r) <- rnorm(terra::ncell(r))
result <- fpROC::auc_metrics(test_prediction = test_data, prediction = r)
```

auc_parallel

Parallel AUC and partial AUC calculation with optimized memory usage

Description

Computes bootstrap estimates of partial and complete AUC using parallel processing and optimized binning.

Usage

```
auc_parallel(  
  test_prediction,  
  prediction,  
  threshold = 5,  
  sample_percentage = 50,  
  iterations = 500L,  
)
```

```

    compute_full_auc = TRUE,
    n_bins = 500L
  )

```

Arguments

test_prediction	Numeric vector of test prediction values
prediction	Numeric vector of model predictions (background suitability data)
threshold	Percentage threshold for partial AUC calculation (default = 5.0)
sample_percentage	Percentage of test data to sample in each iteration (default = 50.0)
iterations	Number of bootstrap iterations (default = 500)
compute_full_auc	Boolean indicating whether to compute complete AUC (default = TRUE)
n_bins	Number of bins for discretization (default = 500)

Details

This function implements a highly optimized AUC calculation pipeline: 1. Cleans input data (removes non-finite values) 2. Combines background and test predictions 3. Performs range-based binning (discretization) 4. Computes cumulative distribution of background predictions 5. Runs bootstrap iterations in parallel: - Samples test predictions - Computes sensitivity-specificity curves - Calculates partial and complete AUC

Key optimizations: - OpenMP parallelization for binning and bootstrap - Vectorized operations using Armadillo

Value

A numeric matrix with ‘iterations’ rows and 4 columns containing:

- auc_complete: Complete AUC (NA when compute_full_auc = FALSE)
- auc_pmodel: Partial AUC for the model (sensitivity > 1 - threshold/100)
- auc_prand: Partial AUC for random model (reference)
- ratio: Ratio of model AUC to random AUC (model/reference)

Partial AUC

The partial AUC focuses on the high-sensitivity region defined by: Sensitivity > 1 - (threshold/100)

See Also

[summarize_auc_results](#) for results processing, [trap_roc](#) for integration method

Examples

```
# Basic usage with random data
set.seed(123)
bg_pred <- runif(1000) # bg predictions
test_pred <- runif(500) # Test predictions

# Compute only partial AUC metrics (500 iterations)
results <- auc_parallel(test_pred, bg_pred,
                       threshold = 5.0,
                       iterations = 100) # Reduced for example

# View first 5 iterations
head(results, 5)

# Summarize results (assume complete AUC was not computed)
summary <- summarize_auc_results(results, has_complete_auc = FALSE)

# Interpretation:
# - auc_pmodel: Model's partial AUC (higher is better)
# - auc_prand: Random model's partial AUC
# - ratio: Model AUC / Random AUC (>1 indicates better than random)

# Compute both partial and complete AUC
full_results <- auc_parallel(test_pred, bg_pred,
                             compute_full_auc = TRUE,
                             iterations = 100)
```

summarize_auc_results *Summarize Bootstrap AUC Results*

Description

Computes aggregated statistics from bootstrap AUC iterations. This function processes the raw output of [auc_parallel](#) to produce meaningful summary metrics of the partial ROC test.

Usage

```
summarize_auc_results(auc_results, has_complete_auc)
```

Arguments

auc_results Numeric matrix output from [auc_parallel](#) (dimensions: n_iterations x 4)

has_complete_auc Boolean indicating whether complete AUC was computed in the bootstrap iterations (affects first summary column)

Details

This function: 1. Filters iterations with non-finite ratio values (handles bootstrap failures) 2. Computes means for each AUC metric across valid iterations 3. Calculates proportion of iterations where model outperforms random (ratio > 1). This way of computing the the p-value of the test.

Special handling: - Returns all NAs if no valid iterations exist - First column (complete AUC) depends on has_complete_auc parameter - Handles NaN/Inf values safely by filtering

Value

A numeric matrix with 1 row and 5 columns containing:

- mean_complete_auc: Mean of complete AUC values (NA if not computed)
- mean_pauc: Mean of partial AUC values for the model
- mean_pauc_rand: Mean of partial AUC values for random model (reference)
- mean_auc_ratio: Mean of AUC ratios (model/random)
- prop_ratio_gt1: Proportion of iterations where ratio > 1 (performance better than random)

Interpretation Guide

- mean_auc_ratio > 1: Model generally outperforms random predictions - prop_ratio_gt1 = 1.9: 90 - mean_pauc: Absolute performance measure (higher = better discrimination)

See Also

[auc_parallel](#) for generating the input matrix

Examples

```
# Basic usage with simulated results
set.seed(123)
# Simulate bootstrap output (100 iterations x 4 metrics)
auc_matrix <- cbind(
  complete = rnorm(100, 0.85, 0.05), # Complete AUC
  pmodel   = rnorm(100, 0.15, 0.03), # Partial model AUC
  prand    = rnorm(100, 0.08, 0.02), # Partial random AUC
  ratio    = rnorm(100, 1.9, 0.4)    # Ratio
)

# Summarize results (assuming complete AUC was computed)
summary <- summarize_auc_results(auc_matrix, has_complete_auc = TRUE)

# Typical output interpretation:
# - mean_complete_auc: 0.85 (good overall discrimination)
# - mean_pauc: 0.15 (absolute partial AUC)
# - mean_pauc_rand: 0.08 (random expectation)
# - mean_pAUCratio: 1.9 (model 90% better than random)
# - p_value: 0.98 (98% of iterations showed model > random)

# Real-world usage with actual AUC function output
```

```

# First run bootstrap AUC calculation
bg_pred <- runif(1000)
test_pred <- runif(500)
auc_output <- auc_parallel(
  test_prediction = test_pred,
  prediction = bg_pred,
  iterations = 100
)

# Then summarize results (complete AUC not computed in this case)
summary <- summarize_auc_results(auc_output, has_complete_auc = FALSE)

# Print summary statistics
colnames(summary) <- c("mean_complete_auc", "mean_pauc",
  "mean_pauc_rand", "mean_pAUCratio", "p_value")
print(summary)

# Expected output structure:
#   mean_complete_auc mean_pauc mean_pauc_rand mean_pAUCratio p_value
# [1,]                NA    0.152         0.083          1.83         0.94

```

trap_roc

Calculate Area Under Curve (AUC) using trapezoidal rule

Description

Computes the area under a curve using the trapezoidal rule of numerical integration.

Usage

```
trap_roc(x, y)
```

Arguments

x	Numeric vector (arma::vec) of x-coordinates (should be sorted in increasing order)
y	Numeric vector (arma::vec) of y-coordinates corresponding to x-coordinates

Details

The trapezoidal rule approximates the area under the curve by dividing it into trapezoids. For each pair of adjacent points $(x[i], y[i])$ and $(x[i+1], y[i+1])$, it calculates the area of the trapezoid formed. The total AUC is the sum of all these individual trapezoid areas.

Special cases: - Returns 0 if there are fewer than 2 points (no area can be calculated) - Handles both increasing and decreasing x values (though typically x should be increasing for ROC curves)

Value

A numerical value representing the computed area under the curve as a double precision value.

See Also

[integrate](#) for R's built-in integration functions

Examples

```
# R code example:  
x <- c(0, 0.5, 1, 1.5, 2)  
y <- c(0, 0.7, 0.9, 0.95, 1)  
trap_roc(x, y)
```

Index

`auc_metrics`, [2](#)

`auc_parallel`, [3](#), [5](#), [6](#)

`integrate`, [8](#)

`summarize_auc_results`, [4](#), [5](#)

`trap_roc`, [4](#), [7](#)