MetaCommunityMetrics.jl: A Julia Package for Spatiotemporal Metacommunity Analysis

25 September 2025

Summary

Community ecology studies how species coexist and change over time and space (Leibold et al. 2004). `MetaCommunityMetrics.jl` is a `Julia` (Bezanson et al. 2017) package that helps ecologists analyze large datasets of species observations to understand how community composition varies across different sites and time points. The package addresses key ecological questions in community ecology from observational data: What drives changes in community composition? How do species coexist locally and regionally over time? How do species dispersal abilities and habitat preferences shape community patterns? By implementing computationally optimized versions of established software and new complementary functions that are designed for handling large spatiotemporal datasets, this package enables ecologists to efficiently analyze large-scale spatiotemporal community data that would be computationally prohibitive with existing tools. We designed intuitive interfaces that would feel familiar to ecologists transitioning from `R`. Our function naming conventions closely match their `R` equivalents, and we maintained parameter naming and ordering consistency where possible. For example, if an `R` function accepts parameters named `species` and `site` in that order, our `Julia` implementation follows the same pattern, making it easier for ecologists to use our package. Table 1 shows a summary of that metrics and functions are avaliable in `MetaCommunityMetrics.jl`

Table 1: A summary of the metrics and functions available in `MetaCommunityMetrics.jl`.

Metrics	Functions	Details
Beta diversity decompositions in space and time	`beta_diversity()`	Re-implementation of `adespatial`(Dray et al. 2023)
	`spatial_beta_div()`	Re-implementation of `R` script from Guzman et al. (2022)

Metrics	Functions	Details
	`temporal_beta_div()`	Re-implementation of `R` script from Guzman et al. (2022)
Dispersal-niche continuum index	`create_groups()`	A novel grouping function for DNCI analysis
	`plot_groups()`	A novel visualization function for DNCI analysis
	`DNCI_multigroup()`	Re-implementation of `R` script from `DNCImper` (Gibert 2024)
Niche overlap index	`niche_overlap()`	Novel implementation
Occupied patches proportion	`prop_patche()`	Re-implementation of `R` script from Guzman et al. (2022)
Variability Metric	`CV_meta()`	Re-implementation from Wang et al. (2019)
Niche hypervolume measurements	`MVNH_det()`	Re-implementation from `MVNH`(Lu et al. 2021)
	`MVNH_dissimilarity()`	Re-implementation from `MVNH`(Lu et al. 2021)
	`average_MVNH_det()`	Novel extension of `MVNH_det()`
	`average_MVNH_dissimilarity()`	Novel extension of `MVNH_dissimilarity()`

Statement of need

Many valuable `R` (Core Team, n.d.) packages have been developed to aid in community analyses by providing ecological metrics that can summarize processes from biodiversity patterns and have been widely adopted in the field, such as `adespatial` (Dray et al. 2023), `codyn` (Hallett et al. 2020), and `vegan` (Oksanen et al. 2024). However, conducting community analyses in R is often computationally expensive, especially when working with large datasets, a limitation that becomes critical in workflows requiring repeated metric calculations. These include simulation studies for method validation, bootstrap resampling for uncertainty estimation (Efron 1992), and null model approaches for statistical testing (Gotelli and Graves 1996), all of which can create significant computational bottlenecks when processing multiple large datasets iteratively. To improve the efficiency of these computationally intensive analyses, `Julia`, which is known for its just-in-time compilation that optimizes code execution to levels comparable with lower-level languages like C or Fortran (Bezanson et al. 2017), yet, which are still underutilize by ecologists, become a logical choice for re-implementations. During development, we conducted a comprehensive review of existing `Julia` packages under the EcoJulia organization, a community-driven effort providing tools for ecological and environmental analysis, and the PoisotLab, which develop many `Julia` packages focusing on quantitative and computational ecology. We ensure that our package builds beyond the tools currently existing in `Julia` even if they share similar ecological concepts. For example, our beta diversity metrics share a similar concept, beta diversity, with `Diversity.jl` (Reeve and Harris 2024). However, our implementation decomposes beta diversity into species replacement and richness differences, which differentiates our approach from `Diversity.jl`, which emphasizes alpha, beta, and gamma diversity measures at the metacommunity and community levels.

`MetaCommunityMetrics.jl` delivers substantial computational advantages over existing tools. For example, the `beta.div.comp` function from the `R` package `adespatial` (Dray et al. 2023) has been re-implemented in our package, leading to faster execution times (up to ~ 45 times faster compared to its `R` equivalent) while maintaining methodological consistency (see Figure 1). The overall performance gain across all re-implementations, positions `MetaCommunityMetrics.jl` as a versatile solution for diverse ecological research needs, from standard biodiversity analyses to simulation studies and large-scale data analyses. Also, we extended the functionality of pre-existing R implementations to accept spatiotemporal datasets directly by adding temporal and community-level aggregation methods to metrics that were originally restricted to single timepoints and sites, eliminating the need for extensive data preprocessing. Specifically, beta diversity decompositions in `R` operate on a single metacommunity at a time, while niche hypervolume measurements in `R` calculate the niche volume of individual species and dissimilarity between single pairs of species. We not only re-implemented these original functions, but also extended their functionality by developing additional functions (i.e. `spatial_beta_div()` and `temporal_beta_div()`) that calculate beta diversity decompositions in space (by aggregating abundance or occupancy across time) and beta diversity decompositions in time (by aggregating across space), as well as functions that calculate average niche volume across all species (`average_MVNH_det()`) and average niche volume dissimilarity across all species pairs (`average_MVNH_dissimilarity()`). These extensions are particularly valuable when analyzing spatiotemporal datasets with multiple species. For calculating the DNCI, we also developed a function to group sites according to the grouping conditions suggested by (Vilmi et al. 2021), ensuring that site groupings fulfill the requirements for the calculation.

Additional handling of edge cases when calculating DNCI

Our DNCI implementation handles edge cases common in simulated datasets (e.g., single-species communities, insufficient permutation variation) by returning status flags that identify when standard DNCI calculation is not possible. See documentation for details on all five edge case types.

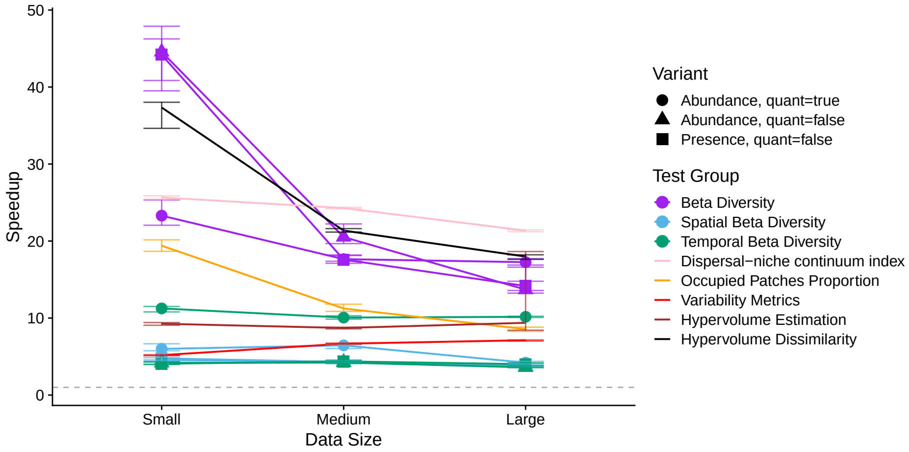

Figure 1: A plot showing the speedup of all benchmarked functions across the three datasets (small, medium and large). Speedup is calculated as the `R` median execution time divided by the `Julia` median execution time. Median speedup and its confidence interval of each comparison is shown. The grey dashed line shows where speedup equals to 1, that is both `R` and `Julia`'s implementations require the same amount of time. Details about this benchmarking can be found in the documentation under the benchmark results section.

Validation

All re-implementations were validated against their R equivalents (see documentation).

Examples

Using rodent metacommunity data as the sample data

We demonstrate the package using rodent data from the Portal Project (Ernest et al. 2018), a long-term Chihuahuan desert ecosystem study. The sample dataset includes 21 species across 24 sites from 2010-2023 (117 sampling events), with simulated spatial coordinates and environmental variables (temperature, precipitation). The data can be loaded with:

```
julia> load_sample_data()
```

The complete output of `load_sample_data()` is available in the documentation.

Illustration using the sample data

Here, we demonstrate analyzing spatiotemporal patterns in species composition (`CV_meta()`, `spatial_beta_div()`, `temporal_beta_div()`), determining assembly processes (`create_groups()`, `DNCI_multigroup()`), and quantifying niche overlap (`niche_overlap()`, `average_MVNH_det()`, `average_MVNH_dissimilarity()`).

First, we will look at how abundance varies across time on species-level and community-level at both local and regional scales using `CV_meta()`:

```
julia> using MetaCommunityMetrics
julia> df = load_sample_data()
julia> CV_meta(df.Abundance, df.Sampling_date_order, df.plot,
         df.Species)
1×4 DataFrame
 Row  CV_s_l   CV_s_r    CV_c_l    CV_c_r
      Float64  Float64   Float64   Float64

   1  1.48859  0.944937  0.718266  0.580183
```

Variability in abundance decreases from local to regional scales (comparing `CV_s_l` to `CV_s_r` and `CV_c_l` to `CV_c_r`) and from species to community levels (comparing `CV_s_l` to `CV_c_l` and `CV_s_r` to `CV_c_r`), indicating greater stability at broader spatial scales and higher organizational levels.

Now, we analyze compositional changes using `spatial_beta_div()` and `temporal_beta_div()`. The beta diversity decompositions of the rodent metacommunity in space based on species abundances:

```
julia> spatial_beta_div(df.Abundance, df.Sampling_date_order,
         df.plot, df.Species; quant = true)
1×3 DataFrame
 Row  spatial_BDtotal  spatial_Repl  spatial_RichDif
      Float64          Float64       Float64

   1         0.264822      0.121882         0.142939
```

The beta diversity decompositions of the rodent metacommunity in time based on species abundances:

```
julia> temporal_beta_div(df.Abundance, df.Sampling_date_order,
         df.plot, df.Species; quant = true)
1×3 DataFrame
 Row  temporal_BDtotal  temporal_Repl  temporal_RichDif
      Float64           Float64        Float64

   1          0.311222      0.0995483          0.211674
```

Spatial composition differences are driven equally by replacement and abundance differences (values are similar), while temporal differences are primarily driven by abundance changes (higher value), suggesting seasonal patterns or disturbance over time (Podani and Schmera 2011; Legendre 2014).

Now, we compute DNCI using `DNCI_multigroup()` to determine the relative contribution of dispersal versus niche processes. The function automatically filters absent or ubiquitous species and handles empty sites.

Before computing the DNCI, we need to group sites using `create_groups()` to make sure we have at least two groups, a minimum of five sites per group, and that the variation in the number of species and sites per group does not exceed 40% and 30%, respectively. We can proceed as follows:

```
julia> grouping_result = create_groups(df.Sampling_date_order,
         df.Latitude, df.Longitude, df.plot, df.Species, df.Presence)
```

`grouping_result` returns a dictionary indexed by `Sampling_date_order`. Time points not meeting grouping requirements are assigned `missing`. We visualize grouping at time point 60:

```
julia> grouping_result[60]
```

Now, we will use `plot_groups()` to visualize the grouping at `Sampling_date_order()` = 60:

```
julia> plot_groups(grouping_result[60].Latitude, grouping_result[60].Longitude,
         grouping_result[60].Group; output_file = "groups.svg")
```

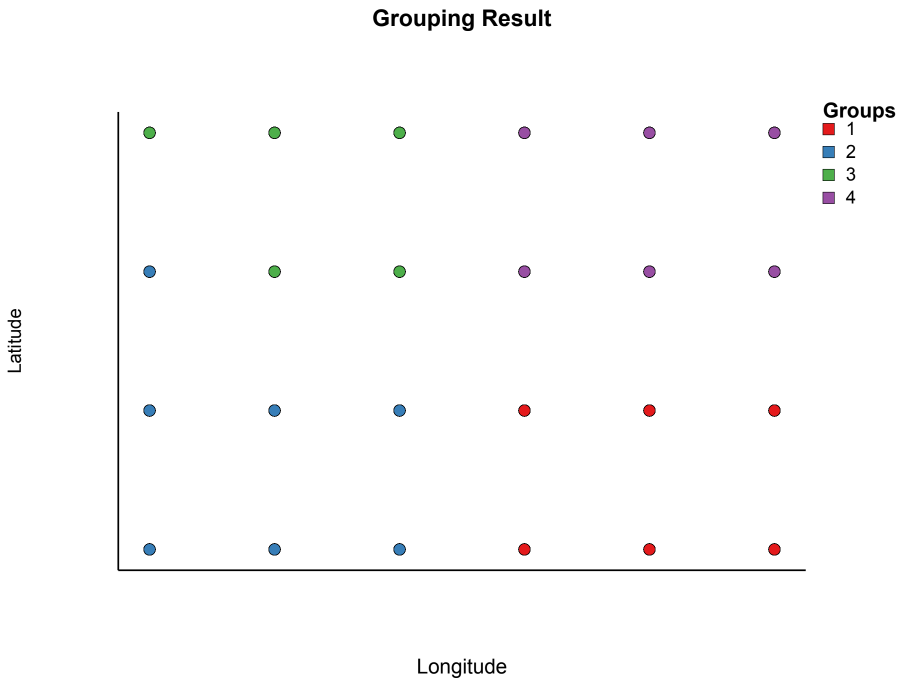

Figure 2: A plot showing the location of sites and their groups. Different colors represent different groups at `Sampling_date_order` = 60.

Figure 2 shows the grouping result at `Sampling_date_order` = 60.

We can then join `df` with `grouping_result[60]` to obtain the matrix and group assignment column for calculating DNCI at `Sampling_data_order` = 60.

```
julia> group_df = @pipe df |>
                  filter(row -> row[:Sampling_date_order] == 60, _) |>
                  select(_, [:plot, :Species, :Presence]) |>
                  innerjoin(_, grouping_result[60], on = [:plot => :Site,
                  :Species], makeunique = true)|>
                  select(_, [:plot, :Species, :Presence, :Group]) |>
                  unstack(_, :Species, :Presence, fill = 0)
julia> comm= @pipe group_df |>
             select(_, Not([:plot,:Group])) |>
             Matrix(_)
julia> Random.seed!(1234)
julia> DNCI_result = DNCI_multigroup(comm, group_df.Group, 1000; Nperm_count = false)
6×6 DataFrame
 Row  group1  group2  DNCI      CI_DNCI  S_DNCI    status
      Int64   Int64   Float64   Float64  Float64   String

   1       1       2  -3.41127  2.17348  1.08674   normal
   2       1       3  -2.44866  2.05951  1.02976   normal
   3       1       4  -2.3671   2.45697  1.22848   normal
   4       2       3  -2.65022  2.28931  1.14466   normal
   5       2       4  -3.0168   2.43496  1.21748   normal
   6       3       4  -1.83521  1.9589   0.979449  normal
```

The DNCI values at `Sampling_date_order` = 60 do not differ significantly from zero for group pairs 1-4 and 3-4, while the rest are significantly smaller than zero. This indicates that most local communities in this metacommunity are dominated by dispersal processes, while both dispersal and niche processes contribute evenly to some local communities at `Sampling_date_order` = 60. To draw conclusions about the entire metacommunity over time, we suggest running this function for all available time points and averaging across all group pairs at all time points.

We will now investigate how species share their niche spaces across all sites and time points using `niche_overlap()`, `prop_patches()`, `average_MVNH_det()`, and `average_MVNH_dissimilarity()`. We will start with `niche_overlap()`:

```
julia> niche_overlap(df.Abundance, df.Species, df.plot, df.Sampling_date_order)
1×3 DataFrame
 Row  mean_niche_overlap_index  min_niche_overlap_index  max_niche_overlap_index
      Float64                   Float64                  Float64

   1                 0.0923816                      0.0                 0.406837
```

Then, we will use `prop_patches()`:

```
julia> prop_patches(df.Presence, df.Species, df.plot)
1×3 DataFrame
 Row  mean_prop_patches  min_prop_patches  max_prop_patches
      Float64            Float64           Float64


   1           0.734649         0.0833333               1.0
```

Finally, we will calculate the average niche volume and average niche volume dissimilarity across all species using `average_MVNH_det()` and `average_MVNH_dissimilarity()`, respectively. Note that these functions, unlike the others, require environmental data, which is already available in our sample dataset (i.e., temperature and precipitation). These environmental variables were already standardized and checked for normality before being stored as our sample data. Note that users need to (1) standardized their environmental variables and transform them to normal distribution (if necessary); and (2) remove singletons before using these functions. The following demonstrates how we use these functions:

```
julia> df = @pipe load_sample_data() |>
                      groupby(_, :Species) |>
                      filter(row -> sum(row.Presence) > 1, _)|>
                      DataFrame(_)
julia> env_data = @pipe df |>
           select(_, [:standardized_temperature, :standardized_precipitation])
julia> average_MVNH_det(env_data, df.Presence, df.Species; var_names =
         ["Temperature", "Precipitation"])
1.2103765096417536
julia> average_MVNH_dissimilarity(env_data, df.Presence, df.Species;
         var_names = ["Temperature", "Precipitation"])
0.030599429364544443
```

High spatial overlap (`mean_prop_patches` ∼ 0.73) combined with low temporal overlap (`mean_niche_overlap_index` ∼ 0.09) and high niche similarity (`average_MVNH_dissimilarity` ∼ 0.03) suggests temporal niche partitioning, where species occupy similar sites at different times (Albrecht and Gotelli 2001; Lear et al. 2021).

Future Directions

We will apply an approximate Bayesian computation framework to infer community assembly processes from long-term phytoplankton monitoring data. This project will depend on `MetaCommunityMetrics.jl` to calculate summary statistics in its pipeline.

Acknowledgement

We are grateful to the authors of the pre-existing `R` packages/implementations that we have implemented in `MetaCommunityMetrics.jl`. We express our gratitude to everyone who has tried out this package and provided feedback on how to improve it. LMG acknowledges funding from USC and Cornell University, YYJC acknowledges funding from the Wrigley Institute.

References

Albrecht, M, and NJ Gotelli. 2001. "Spatial and Temporal Niche Partitioning in Grassland Ants." *Oecologia* 126 (1): 134–41.

Bezanson, Jeff, Alan Edelman, Stefan Karpinski, and Viral B Shah. 2017. "'Julia': A Fresh Approach to Numerical Computing." *SIAM Review* 59 (1): 65–98. https://doi.org/10.1137/141000671.

Core Team, 'R'. n.d. *'R': A Language and Environment for Statistical Computing.*

Dray, Stéphane, David Bauman, Guillaume Blanchet, et al. 2023. *'Adespatial': Multivariate Multiscale Spatial Analysis.* https://CRAN.R-project.org/package=adespatial.

Efron, Bradley. 1992. "Bootstrap Methods: Another Look at the Jackknife." In *Breakthroughs in Statistics: Methodology and Distribution.* Springer.

Ernest, SK Morgan, Glenda M Yenni, Ginger Allington, et al. 2018. "The Portal Project: A Long-Term Study of a Chihuahuan Desert Ecosystem." *BioRxiv*, 332783.

Gibert, Corentin. 2024. *'DNCImper': Assembly Process Identification Based on SIMPER Analysis.* https://github.com/gibertcorentin/DNCImper.

Gotelli, Nicholas J, and Gary R Graves. 1996. *Null Models in Ecology.* Smithsonian Institution Press.

Guzman, Laura Melissa, Patrick L Thompson, Duarte S Viana, et al. 2022. "Accounting for Temporal Change in Multiple Biodiversity Patterns Improves the Inference of Metacommunity Processes." *Ecology* 103 (6): e3683.

Hallett, Lauren, Meghan L. Avolio, Ian T. Carroll, et al. 2020. *'Codyn': Community Dynamics Metrics.* https://doi.org/10.5063/F1N877Z6.

Lear, Karissa O, Nicholas M Whitney, John J Morris, and Adrian C Gleiss. 2021.

“Temporal Niche Partitioning as a Novel Mechanism Promoting Co-Existence of Sympatric Predators in Marine Systems.” *Proceedings of the Royal Society B* 288 (1954): 20210816.

Legendre, Pierre. 2014. “Interpreting the Replacement and Richness Difference Components of Beta Diversity.” *Global Ecology and Biogeography* 23 (11): 1324–34.

Leibold, Mathew A, Marcel Holyoak, Nicolas Mouquet, et al. 2004. “The Metacommunity Concept: A Framework for Multi-Scale Community Ecology.” *Ecology Letters* 7 (7): 601–13.

Lu, Muyang, Kevin Winner, and Walter Jetz. 2021. “A Unifying Framework for Quantifying and Comparing n-Dimensional Hypervolumes.” *Methods in Ecology and Evolution* 12 (10): 1953–68.

Oksanen, Jari, Gavin L. Simpson, F. Guillaume Blanchet, et al. 2024. *‘Vegan‘: Community Ecology Package*. https://vegandevs.github.io/vegan/.

Podani, János, and Dénes Schmera. 2011. “A New Conceptual and Methodological Framework for Exploring and Explaining Pattern in Presence-Absence Data.” *Oikos* 120 (11): 1625–38.

Reeve, Robert, and Charles Harris. 2024. *‘Diversity.jl‘ (V0.5.15)*. https://doi.org/10.5281/zenodo.12790658.

Vilmi, Annika, Corentin Gibert, Gilles Escarguel, et al. 2021. “Dispersal-Niche Continuum Index: A New Quantitative Metric for Assessing the Relative Importance of Dispersal Versus Niche Processes in Community Assembly.” *Ecography* 44 (3): 370–79.

Wang, Shaopeng, Thomas Lamy, Lauren M Hallett, and Michel Loreau. 2019. “Stability and Synchrony Across Ecological Hierarchies in Heterogeneous Metacommunities: Linking Theory to Data.” *Ecography* 42 (6): 1200–1211.