`CLarbor` documentation

Hedvig Skirgård and Siva Kalyan

04/04/2019

Introduction

When using distance measurements to compare languages, it can be necessary to not treat all variables as equally contributing to dissimilarity. This document outlines a set of functions that compute pairwise dependencies among the features in your data, and assign a dependency-adjusted weight to every feature, for use primarily in distance calculations. These functions work by producing Chu-Liu/Edmonds arborescences,[1] hence "`CLarbor`".[2]

Typological datasets often contain dependencies, either explicit logical dependencies or statistical ones. An example of a logical dependency is "number of noun classes" and "is masculine/feminine relevant for noun classes?": one necessarily presupposes the other. Common examples of statistical dependencies include the standard word-order dependencies (O V > NP P and the like). When comparing languages using such datasets, it is necessary to weight features differently, so as to avoid giving undue importance to features that are highly dependent on each other. The method described in this document serves to find the dependencies and give the user a way of weighting the features in their dataset when computing distances. We imagine that this is useful for databases of discrete typological features, but it can applied to cultural and other datasets with discrete features.

This is based on Hammarström & O'Connor's (2013) proposal for computing dependency-sensitive typological distances, which incorporates the Chu-Liu/Edmonds algorithm for finding a minimum spanning arborescence on a complete weighted graph (Chu & Liu 1965 and Edmonds 1967). We have reïmplemented Hammarström & O'Connor's technique in R, along with some extensions. In particular, we have adjusted the method for measuring feature dependencies so as to control for external variables such as language family and area.

Our implementation of Hammarström & O'Connor's dependency sensitive typological distances and the Chu-Liu/Edmonds-alogrithm

We begin with a data frame consisting of discrete typological features for a set of languages. For illustration, we use the Sahul survey (Reesink, Singer & Dunn 2009), which consists of 292 features for 200 languages of the Australia/New Guinea region. In addition, we produce a second data frame with information on language family and area for each of the languages in this dataset (compiled from Glottolog (Hammarström et al 2018) and AUTOTYP (Bickel et al 2017), respectively). In order to be able to control for language family and area simultaneously, we combine these two variables into a single one, so that we have one value for each unique combination of family and area.

On the basis of these two data frames, we measure the extent to which each feature predicts every other feature, i.e., the dependency of one feature on another or how predictable one feature is given another specific feature. Hammarström & O'Connor measure the dependency of B on A by computing the mutual information of A and B and dividing by the entropy of B. This tells us how much of the informational content of B is shared with A.[3] This creates an asymmetric dependency matrix, which can be interpreted as a complete, weighted directed graph where every node is a feature, every edge a dependency and edge weights are equal to the extent of the predictability/dependency.

[1] "Arborescence" is a term that refers to either a tree or a forest.

[2] Edmonds's account of this algorithm is more well-known, but Chu & Liu published earlier so we are using their initals.

[3] We thank Gereon A. Kaiping for explaining this to us.

There are other ways of controlling for external variables, the user is free to use another method such as interaction information or incorporate full phylogenetic relationships and not just the discrete family membership status. As long as the result is a complete, asymmetric dependency matrix of all features, the rest of the functions will be compatible with other ways of computing the dependencies. However, we should state that if users were to compute an asymmetric dependency matrix in a different fashion from what we have done here, the validity of the results may not hold; Hammarström & O'Connor's dependency measure has unique information-theoretic properties.

Given the complete weighted directed graph of feature dependencies, we follow Hammarström & O'Connor in using the Chu-Liu/Edmonds algorithm to compute a maximum spanning arborescence.[4] The Chu-Liu/Edmonds algorithm explores the directed graph and prunes edges in a systematic fashion until it ends up with the optimal maximum spanning arborescence, the structure of the graph that best explains the dependencies in the data without any circularity. The algorithm requires a starting point for the pruning, this node will end up as a root in the final arborescence.[5] For a full detailed explanation of how the Chu-Liu/Edmonds algorithm works, we refer the reader to Georgiadis (2003). The user can either specify a feature as the starting point or let the algorithm explore every feature as a starting point and compare all the resulting arborescences and report which is the optimal one. We have provided functions for doing both of these things.

The end result is a weighted arborescence, i.e. either one or a set of weighted trees. Each feature has (ideally) only one incoming edge which has a weight that represents how predictable the feature is given the source of the edge, with the exception of the root(s) of the tree(s) which has/have no incoming edge(s).[6] From this arborescence we can compute dependency weights, with the root(s) being assigned a weight of 0. The function that assumes a known root produces one object, namely an arborescence; the function that runs through all possible starting points returns two objects: the optimal arborescence and which starting point it was that produced it.

Given the nature of this algorithm, it can either produce one tree or several (a forest). A forest has multiple roots, however when the algorithm starts to explore the space it only starts from one place, i.e. even when the result is a forest there was only one starting point that gave rise to it. The dataset that was explored in Hammarström & O'Connor gave rise to only one dependency tree, so they do not discuss instances where the algorithm returns a forest. The demo here also produces trees, not forests. Other iterations of this experiment have generated forests, which gave us the opporunity to highlight this aspect of the Chu-Liu/Edmonds algorithm, but the final version did end up as trees for both the uncontrolled and controlled condition.

We now outline the steps in our workflow and the functions we provide.

Steps of the CLarbor workflow

There are 3 steps to the CLarbor workflow: (1) creating the complete dependency graph, (2) pruning this graph to the optimal maximum spanning arborescence and (3) computing the weight of each feature. Users can also use their own method for (1), as long as an asymmetrical complete dependency matrix is created. If steps (2) and (3) return instances where one node has more than one incoming edge, see the optional step (4).

1. Create a dependency matrix given the data, either with or without external variables. This dependency matrix can be understood as the adjacency matrix of a complete directed weighted dependency graph among features. The function `dependency_matrix_cond_MI(value_df, external_variables_df)` expects two data frames, one containing all the values of the features we are computing the dependencies over (`value_df`) and a second one which contains the external variables (`external_variables`). If there are no external variables, the function creates an external variable which is constant across the

[4] We tried using the `msArborEdmonds` function from the `optrees` package, but it didn't work, for reasons that are still unclear to us; also, the inner workings of this function were opaque. Thus we reïmplemented the algorithm ourselves, on the basis of the description in Georgiadis (2003). We would like to thank Harald Hammarström for his invaluable guidance and support in this process.

[5] The "starting point" for the Chu-Liu/Edmonds algorithm is conventionally refered to as the "root". However, when the final arborescence is a forest, it will only match up with one of the roots of the trees.

[6] For technical reasons, we may sometimes end up with a node having multiple incoming edges, all of which have exactly the same weight. However, even in this case, it is unproblematic to speak of "the weight of the incoming edge".

observations. The function also tests that the two dataframes do indeed both contain a column with the IDs of the observations and that these are the same across the two dataframes and are unique. It then computes the conditional mutual information between all pairs of features given the external variables, divided by conditional entropy of the target feature. The result is an asymmetric matrix that is designed to fit with the other functions. A high value for a feature pairing indicates that the source feature highly predicts the target feature. The diagonal of the matrix is therefore 1. It is possible to give another dependency matrix to the other functions, but it needs to be asymmetric.

2. Create a Chu-Liu arborescence (tree/forest). This step has two functions, depending on whether the user wants to supply a specific feature as the starting point (root) for the Chu-Liu/Edmonds pruning algorithm or have the optimal starting point computed based on all possibilities. The function `CLarbor_from_dep_matrix_known_root(dep_matrix, root)` is to be used when the root is known, it returns the optimal maximum spanning arborescence. `CLarbor_from_dep_matrix_check_all_roots(dep_matrix)` computes the arborescences given all possible starting points and returns two objects: the optimal arborescence and its starting point. Both functions take the output of step 1, the dependency matrix, and create a directed acyclic graph where the edge weights are equal to the dependency weights. The first function takes significantly less time to run than the second.

3. Compute the dependency weight of each feature as the weight of its unique incoming edge (or 0 if there is no incoming edge). The function `make_Clarbor_df(weighted_tree)` takes the output of step 2 and creates a data frame which lists all the edges and their weights. NB that sometimes the Chu-Liu/Edmonds algorithm can actually produce an "arborescence" in which a node has two or more incoming edges. This happens if the edges are of exactly the same weight. Given that, it is possible that the result of this function has more rows than there are features in the starting dataset.

4. (Optional. If the result from step 3 containts instances where a node has more than one incoming edge, you can use `dplyr::distinct()`. See explanation below.)

Demonstration using the Sahul dataset

In order to illustrate the value of our approach, we run it on a dataset that we are free to share - the Sahul-survey (Reesink, Singer & Dunn 2009)[7] - and demonstrate the dependency-sensitive distances using NeighborNet, trees, and MDS. The Sahul-dataset contains 292 typological features coded for over 200 languages. These languages represent 45 top-level genetic units ("families") and 10 areas, which when combined result in 57 unique values for the external variable. We include the dataset in this GitHub repo, including a mapping between the feature numbers, questions and abbreviations.

Going through the steps with the Sahul-dataset

Steps 1 to 3 are implemented in the `demo_CLarbor_with_Sahul_uncontrolled.R` and `demo_CLarbor_with_Sahul_controlle` scripts; step 4 is implemented (for both uncontrolled and controlled conditions) in `demo_CLarbor_graph.R` (since it is only relevant for visualisation).

1. Creating the dependency matrix. We have the 292 features over 200 languages in one tsv-file, `demo_data/Sahul_structure_wide.tsv`, this is our `value_df`. We also have a dataframe of external variables, these are the names of the top-level genetic units (language families) and areas. The ID for each language is found in the first column in each dataframe, and they passed all the checks for being identical accross data frames and unique in each dataframe. The external variable for linguistic area is based on the classifications in the AUTOTYP project and aims to delimit areas of linguistic and cultural exchange/history (i.e. not Glottolog macroareas). Computing a dependency matrix based on this data took appriximately 10 minutes. We computed both uncontrolled and controlled dependency matrices.

[7] We are grateful to Michael Dunn and the other coauthors for letting us use this dataset.

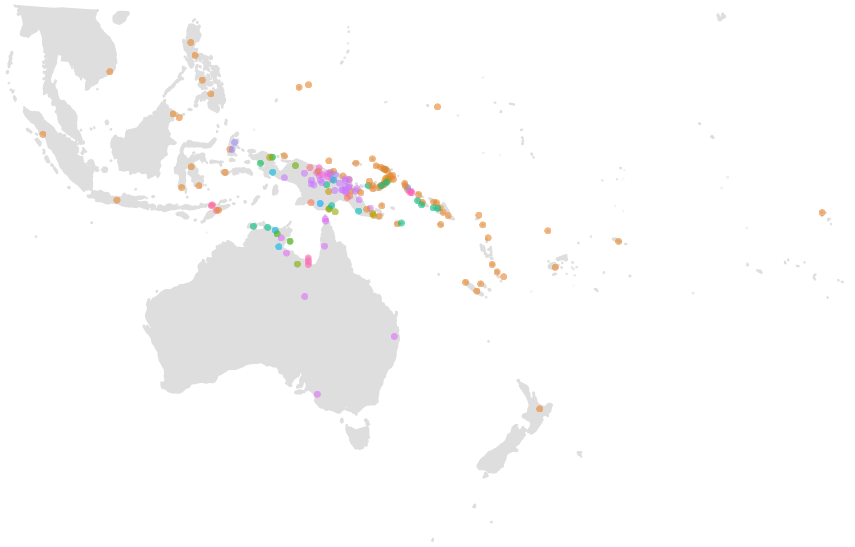


Figure 1: Map of the locations of the languages in the Sahul sample (minus Dutch)

2. In this instance, we do not know which is the best root (starting point) to give to the Chu-Liu/Edmonds algorithm, so we use `CLarbor_from_dep_matrix_check_all_roots(dep_matrix)`. Running this function over this dataset took 50 hours on a computer with four cores at 3.2 GHz. The result was 2 objects: one tree and the optimal starting point "`SAHUL067`" (for uncontrolled), or "`SAHUL4.3.09a`" (for controlled). Furthermore, the function kindly supplies us with an estimation of how much of our dataset mass is predictable (once the external variables are controlled for).
3. We run `make_CLarbor_df(weighted_CLarbor)` over the arborescence from step 2. The dependency dataframe from our arborescence does contain several cases where a feature node had more than one incoming edge, hence more than one row per feature in the data frame. Therefore, we apply the optional step 4.
4. We end up with a dataframe that in some instances has more than one incoming edge for certain nodes, which is not ideal for our visualisations. These edges are of equal weight, making the Chu-Liu/Edmonds algorithm unable to select which one to remove. We reduce these edges ourseleves by running `dplyr::distinct()` on the `to` column (making sure to set the `.keep_all` argument to TRUE), which produces a dataframe with only rows that have a unique value in the `to` column. This gives us a graph where each node only has one incoming edge, represented as a data frame where every row has a unique value in the `to` column.

Now we have a Chu-Liu/Edmonds arborescence (in our case trees) and weights for each feature in our dataset. We can use these weights when computing distances and in other analysis. We will now visualise these results, both when we control for external variables and when we don't, as well as the corresponding unweighted equivalents to demonstrate what the method is capable of.

Data wrangling notes

There are 3 files that you need for running this analysis.

1. **`value_df`**

A wide data table of your features and values. Features as columns with the first column being an identifier for the observations. This table should have end with "wide.tsv".

2. `external_variables_df`

A table with the external variables of the observations. In our case, a large table of meta-information about Glottolog languoids. You can select any other file here, as long as the first column contains identifiers that can be linked to the wide data table (1).

If you wish to use another external variables file, changes this line in demo_tidying.R:

```
external_variables_fn <- "data/Glottolog_lookup_table_Heti_edition.tsv"
```

3. Feature meta-data file. This is used for the graphs of the CLarbors. It needs to have the following columns: Feature_ID, Abbreviation and group. Feature_ID needs to correspond to the column names in value_df

Results of demo

Our functions produce three distance matrices for all the languages in our data.

1. unweighted
2. weighted but with no external variables controlled for
3. weighted and with external variables controlled for

In this section we will interrogate the results by looking at the changes in distances between these, which features are most predictable and visualise the distances matrices as neigbour-joining trees, NeighbourNets and in terms of multidimensional scaling.

The features all get assigned a weight according to how predictable they are, given the CLarbor. In table 1 we have the least and most predictable features given the uncontrolled CLarbor. A low value and high placement in the table means that the feature is not very predictable, and conversely a high value and a low placement means that the feature is very predictable. Unsurprisingly, the known logical dependencies in the data show up among the highest predictability values. The feature "What is the counting system?" with the options "decimal", "quinary" "body-tallying", "vigisemal" and "minimal" is very good at predicting the binary question "Is there a body-part tallying system?". You can see the mapping between feature questions and abbreviations in the file `Sahul_ID_desc.tsv`.

from	to	dependency
FutTenseV	VSuppletion	0.0182060
VVarPerson	WordFinalC	0.0480637
TAMPersonPortmanteau	RealisIrreal	0.0507526
ComplementClauseSpeech	12PronConfl	0.0550464
VOtherIncorp	VLocativeDir	0.0633023
...	...	...
OrderNumN	NumN	1
OrderDem	DemN	1
OrderPosdPosr	PossrPossd	1
OrderSV	IntransSV	1
Counting	CountBody	1

Table 1: Top and bottom 5 features in terms of predictability in the weighted uncontrolled CLarbor

In table 2, we see the top and bottom features given the controlled CLarbor, i.e. when we have taken area and family into account. We still see what can be viewed as rather trivial links, minimal-augmented pronoun systems have for example by necessity a distinction between inclusive and exlcusive pronouns. However, there are other links here that are less obvious, and where we would appreciate input in the interpretation.

from	to	dependency
OSuffix	BenMarkedV	0.2356166
PunctualCont	ModalAux	0.2367853
RealisIrreal	WordFinalC	0.2379070
OrderDem	RealisIrreal	0.2394622
RealisIrreal	OnsetCClust	0.2411139
...	...	...
AffixCaus	MinAug	1
IntransVS	MinAug	1
PhonVelarFricOrGlide	MinAug	1
PhonVelarFricOrGlide	NumDecl	1
InclExclDist	MinAug	1

Table 2: Top and bottom 5 features in terms of predictability in the weighted controlled CLarbor

Having looked now at the differences between the dependencies, we turn to what this analysis brings in relation to the observations, the languages. We are now comparing the distance matrices generated by the unweighted feature set, the uncontrolled weighted and the controlled weighted. We make 2 comparisons, firstly between the unweighted and weighted uncontrolled and secondly between the weighted uncontrolled and the weighted controlled. The average unweighted distance between two languages is 0.5928564; the average weighted uncontrolled distance is 0.5967084; and the average weighted controlled distance is 0.6208452. All of these are highly significantly different from each other (unweighted vs. weighted uncontrolled: $t = -5.094$, $df = 28347$, $p = 3.527 \times 10^{-7}$; weighted uncontrolled vs. weighted controlled: $t = -32.596$, $df = 28390$, $p < 2.2 \times 10^{-16}$).

In addition to comparing the distances overall, we can look at individual pairs of languages, and see which pairs move closer together or farther apart as we move from one condition to the next. We have presented some of these results in tables 3 and 4. We have also visualised this on a map, where every point is a language and the colour representes the average distance that that language has beeen "moved". A red color indicates that the language was among the languages that on average moved further apart from everyone else, and a blue color indicates that it moved closer to everyone else.

	Language 1	Language 2	difference
Closer	Kaulong	Saposa	-0.06073913
	Kuot	Nalik	-0.05726712
	Iaai	Nalik	-0.05507513
	Manam	Saposa	-0.05338823
	Iloko	Pele-Ata	-0.05284266
	...	...	...
Further apart	Roviana	Southern Sama	0.05557907
	Grass Koiari	Nggem	0.05593593
	Klon	Namia	0.05708976
	Rennell-Bellona	Southern Sama	0.05723085
	Bauzi	Duna	0.05778703

Table 3: Changes in distances of language pair between unweighted and weighted uncontrolled distances (top and bottom 5).

When inspecting these tables and maps, do note that these differences are not huge. Furthermore, note the recurrence of Nalik (among languages that grew closer together), and Southern Sama (among languages that grew further apart). In particular, Southern Sama is moving away from two of its Austronesian Oceanic cousins. Remember, we haven't yet controlled for family membership or area, but the Austronesian family is still so dominant in our sample that it is likely to drive dependencies nontheless. I.e., Southern Sama is

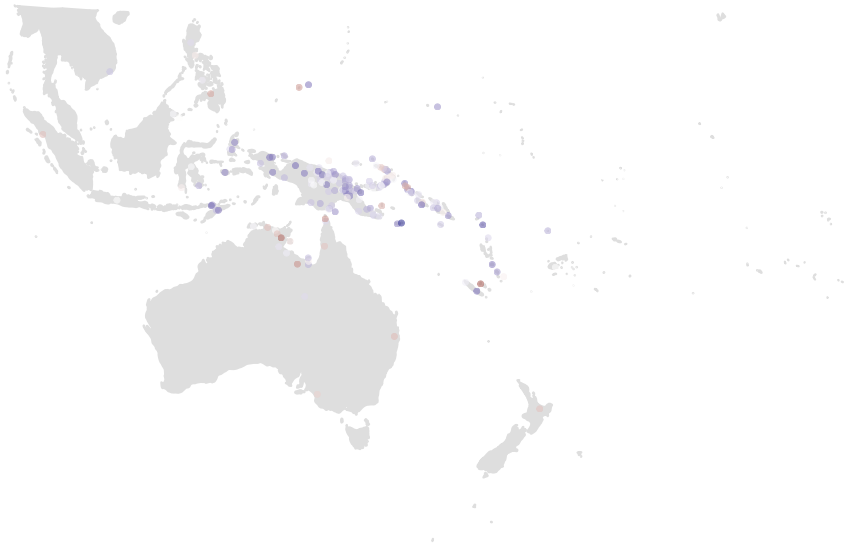


Figure 2: Differences between 1 and 2

indeed very similar to Roviana and Renell-Bellona on account of their all being 'well-behaved' Austronesian languages, but once those traits become less important it moves further away from them.

Here are the languages whose distances changed the most in the transition from weighted uncontrolled to weighted controlled:

	Language 1	Language 2	difference
Closer	Qaqet	Sudest	-0.02808764
	Orya	Saposa	-0.02789271
	Arammba	Ngandi	-0.02770343
	Mali	Ramoaaina	-0.02766718
	Bandjalangic	Ngandi	-0.02698233
	...	...	...
Further apart	Tagalog	Tainae	0.06600770
	Hiligaynon	Tainae	0.06738921
	Arosi	Futuna-Aniwa	0.06876116
	Klon	Rotuman	0.06929129
	Klon	Maori	0.07565244

Table 4: Changes in distances of language pair between weighted uncontrolled and weighted controlled distances (top and bottom 5).

Ngandi, a Gunwinyguan language of Australia, is moving closer to both a fellow Australian but Pama-Nyungan language - Bandjalangic - and also to a language across the Torres Starit - Arammba of the Morehead-Wasur family of Southern New Guinea. On the other end of the spectrum, we have a Timor-Alor-Pantar language - Klon - that is moving away from two Austronesian languages - Maori and Rotuman.

Chu-Liu graph (optimal maximum spanning arboresecence of dependency matrix)

In this demo, we ran the Chu-Liu/Edmonds algorithm over both the adjacency matrix of dependencies that were not controlled for area or family, as well as the one that was. In this section, we will investigate the arborescences resulting from the uncontrolled and the controlled ajdacency matrices respectively.

Furthermore, as has been discussed earlier, it is possible for the resulting Chu-Liu/Edmonds arborescnece to include nodes with more than one incoming edge. This happens if the incomoing edges are of identical weight. A graph of this kind cannot be represented as a tree. This is the case with our demo data. However, since

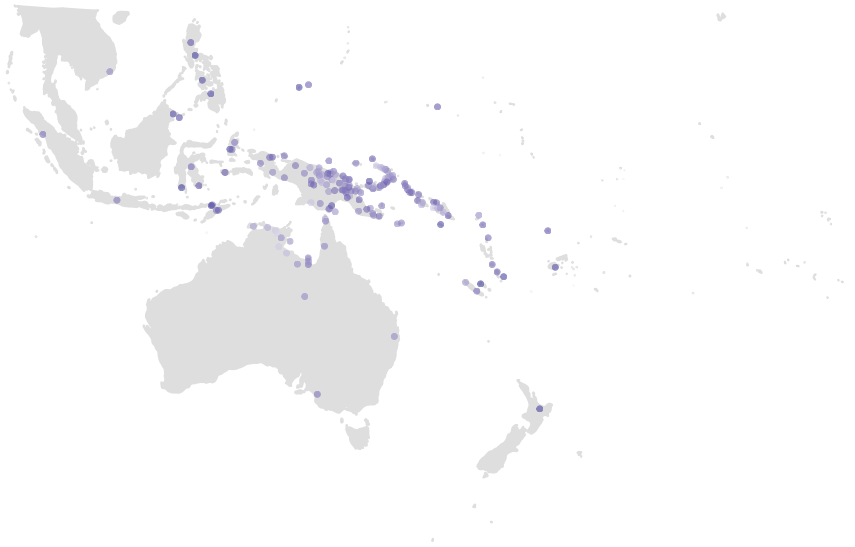
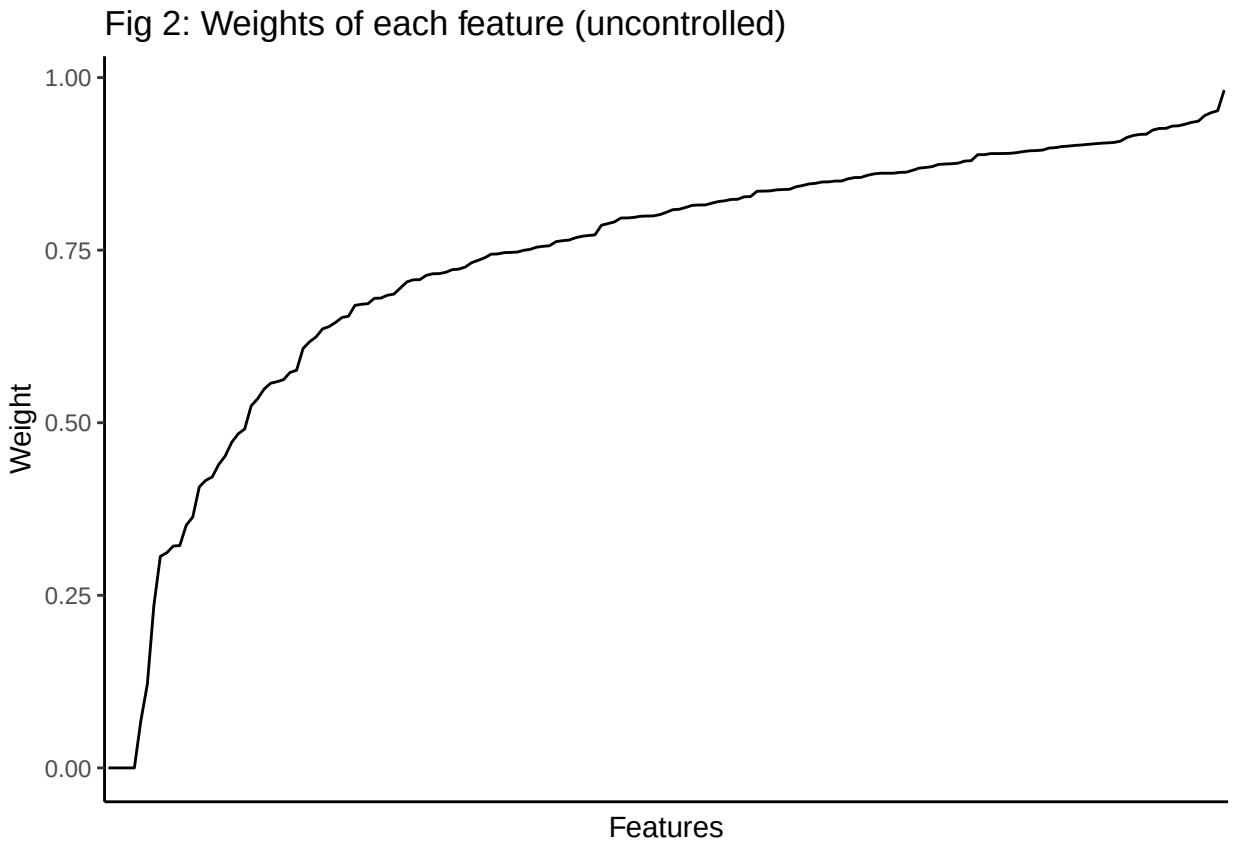


Figure 3: Differences between 2 and 3



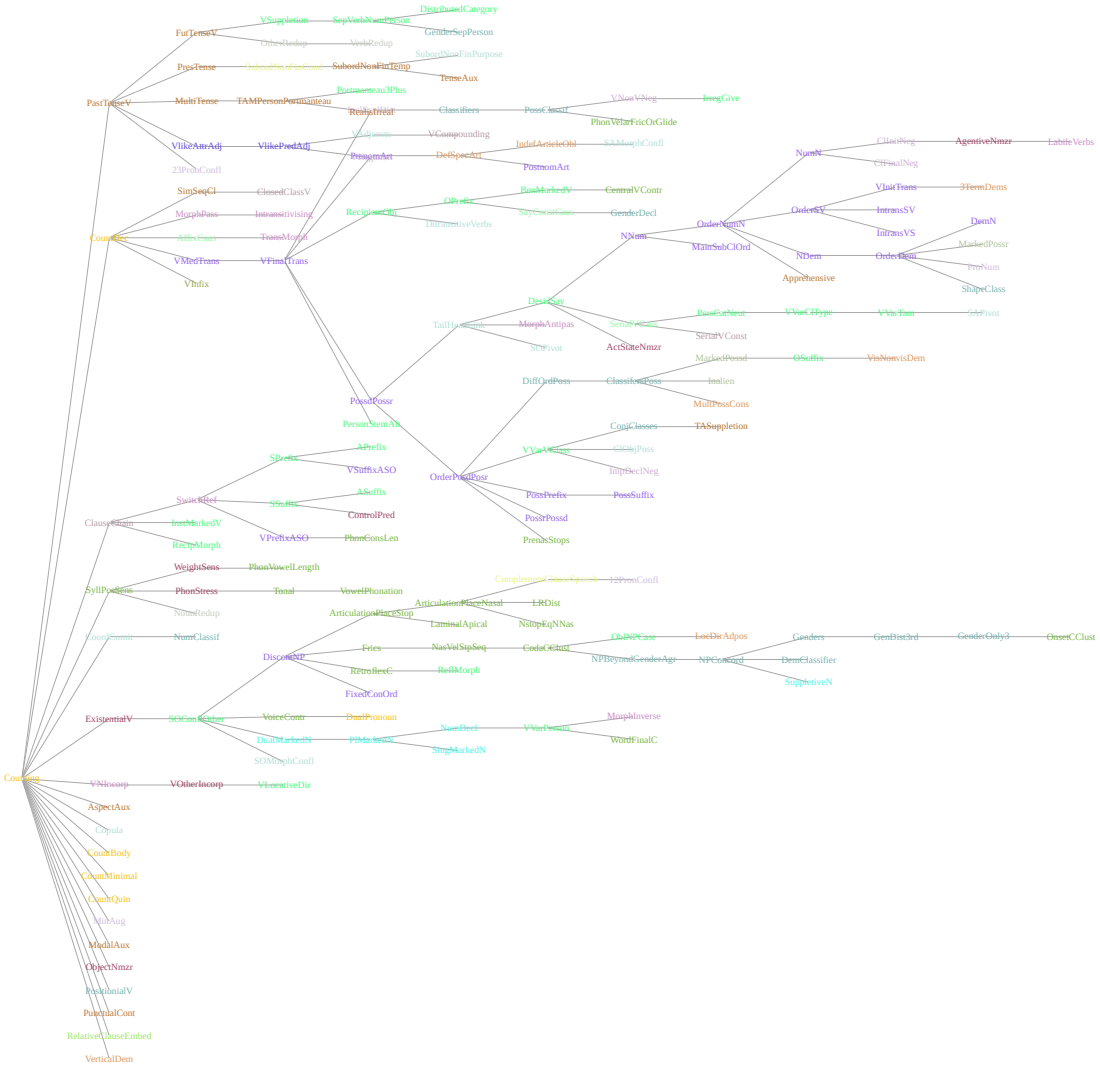


Figure 4: Pruned Chu-Liu graph of Sahul features (uncontrolled)

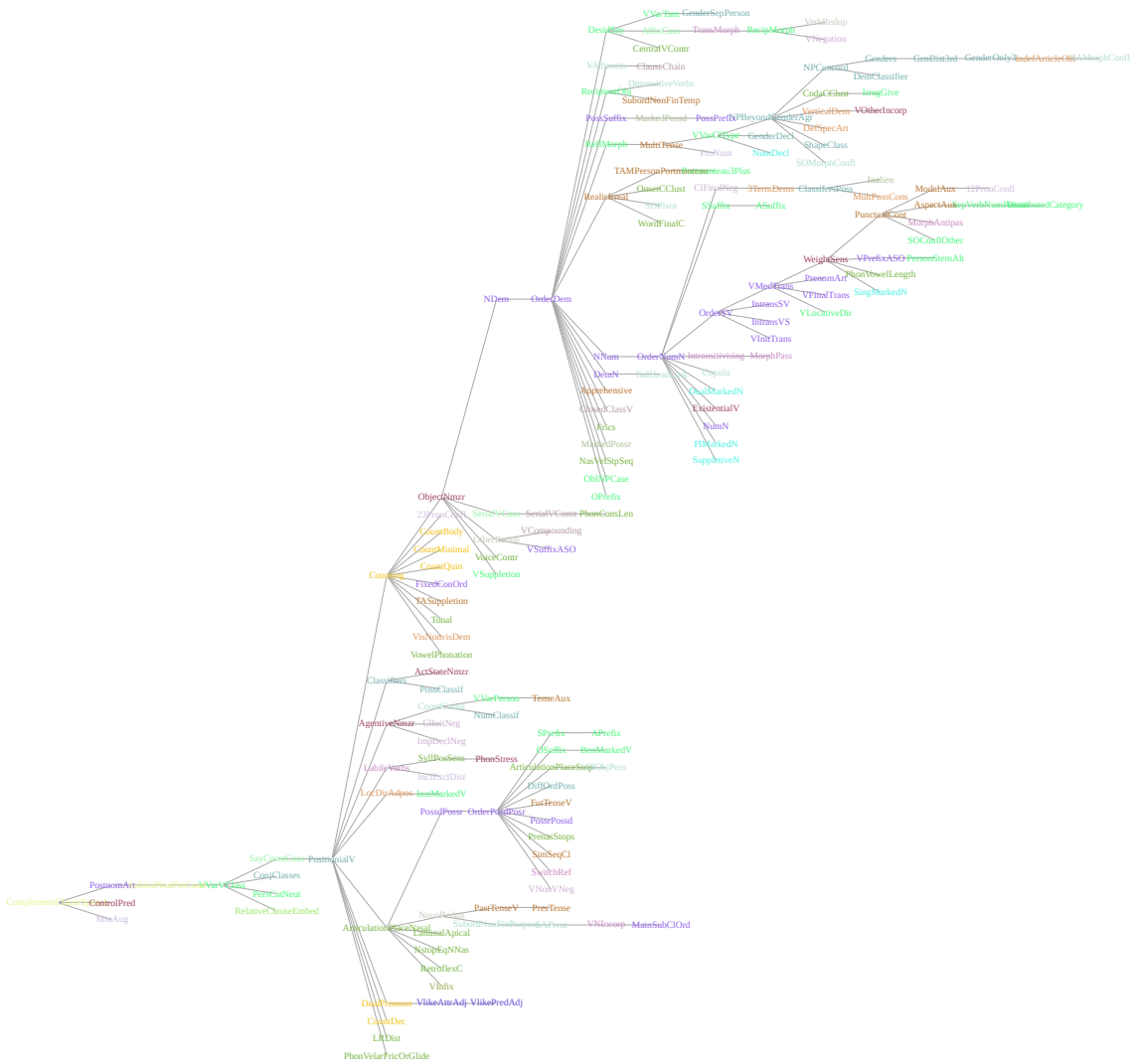
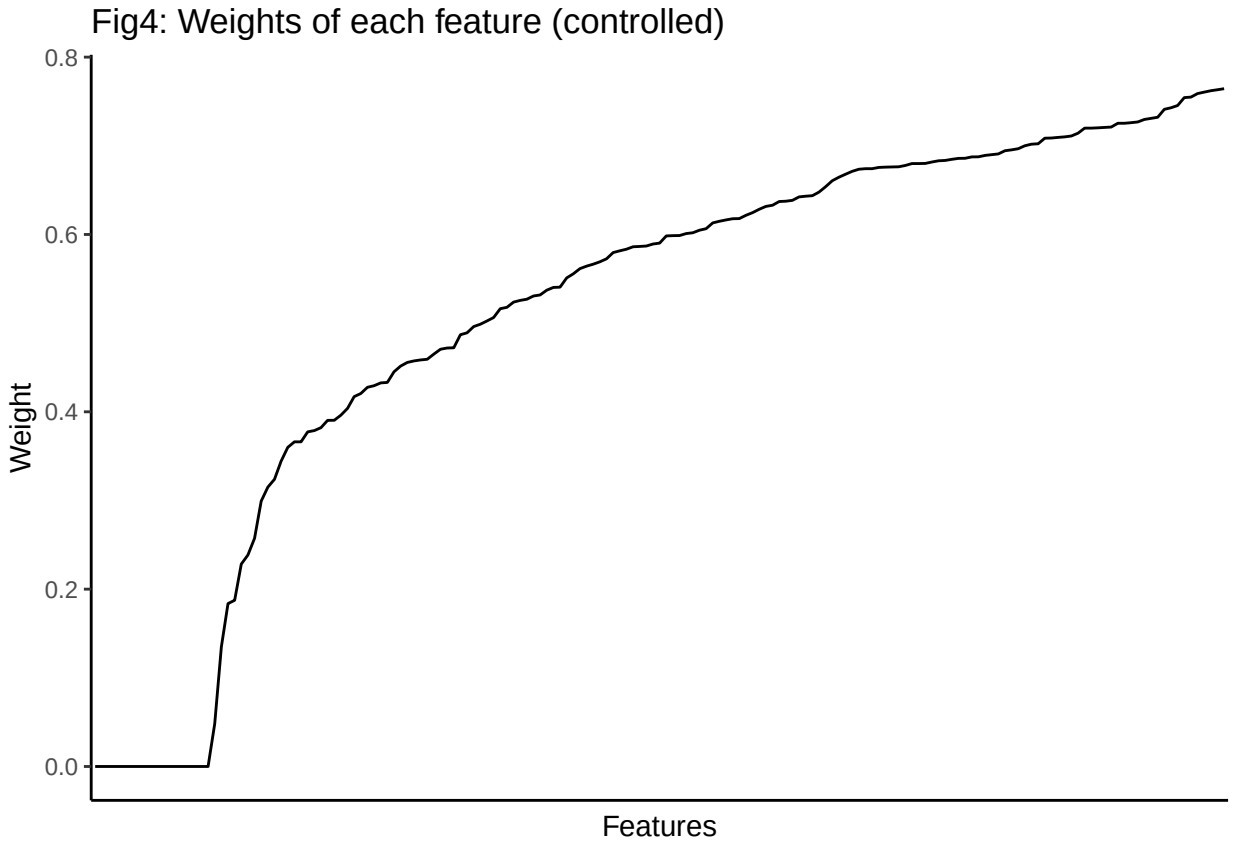


Figure 5: pruned Chu-Liu graph of Sahul features (controlled)



Distance-based visualisations

We have made three different kinds of visualisations of the distance matrix of languages in the dataset given the dependency weights of the features computed by the CLarbor functions: Neighbour-joining trees, NeigbourNets and MDS plots. All visualisations are based on Gower-distances, calculated with the `daisy` function from the `cluster` package.

For reference, we will present all combinations of the following:

1. unweighted distnaces
2. weighted but with no external variables distances
3. weighted and with external variables distances

a. neighbour joining trees
b. neighbour nets
c. multidimensional Scaling (MDS)

Neighbour-joining trees

(See next page.)

NeighborNets (SplitsTree)

(See following pages.)

Comparing Fig 6 and 7, we can see that Southern Sama has now moved further apart from Oceanic Austronesian languages and instead closer to more western Austronesian languages. Overall however, the basic structure of the network doesn't change that much.

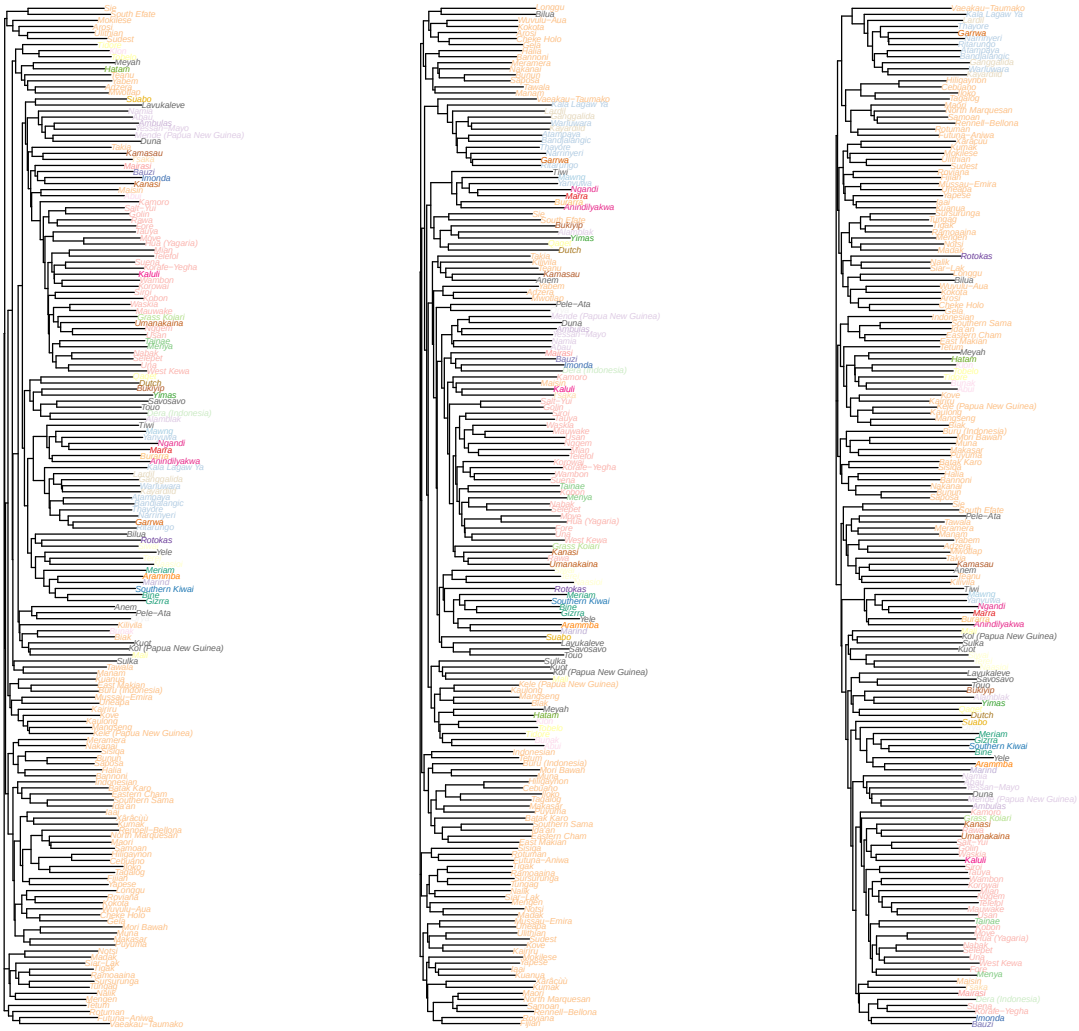


Figure 6: Neighbour-joining trees

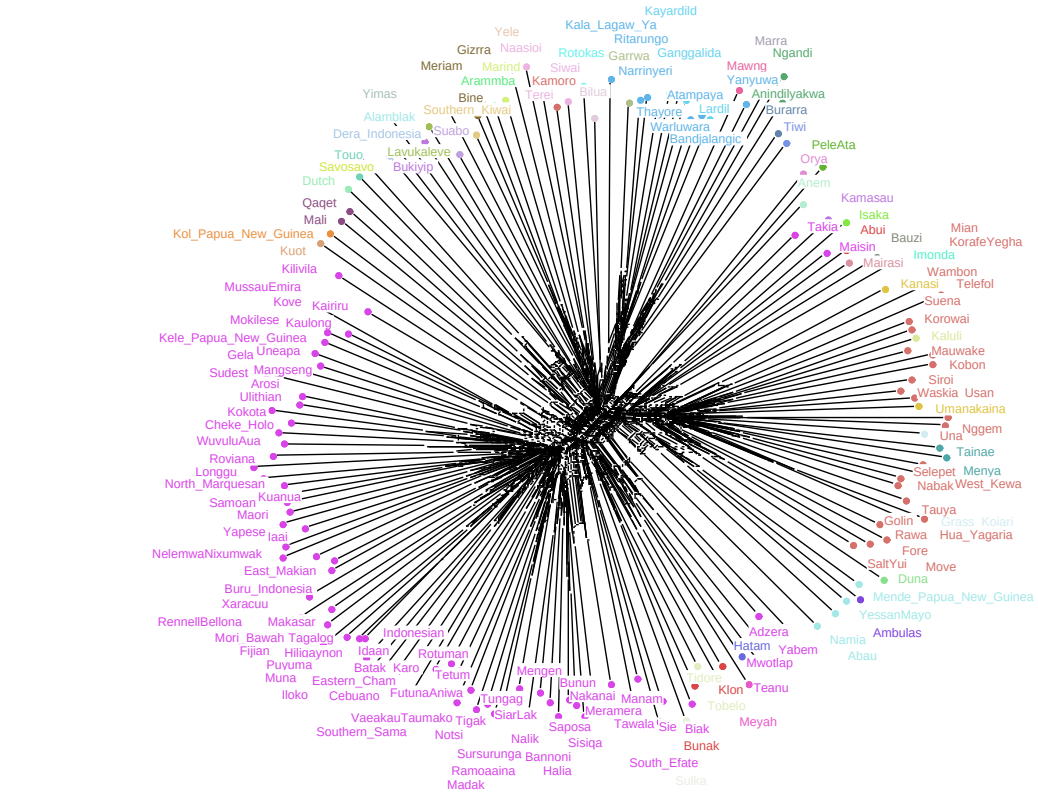


Figure 7: Neighbournet generated by SplitsTree from the unweighted distance matrix of the Sahul-survey

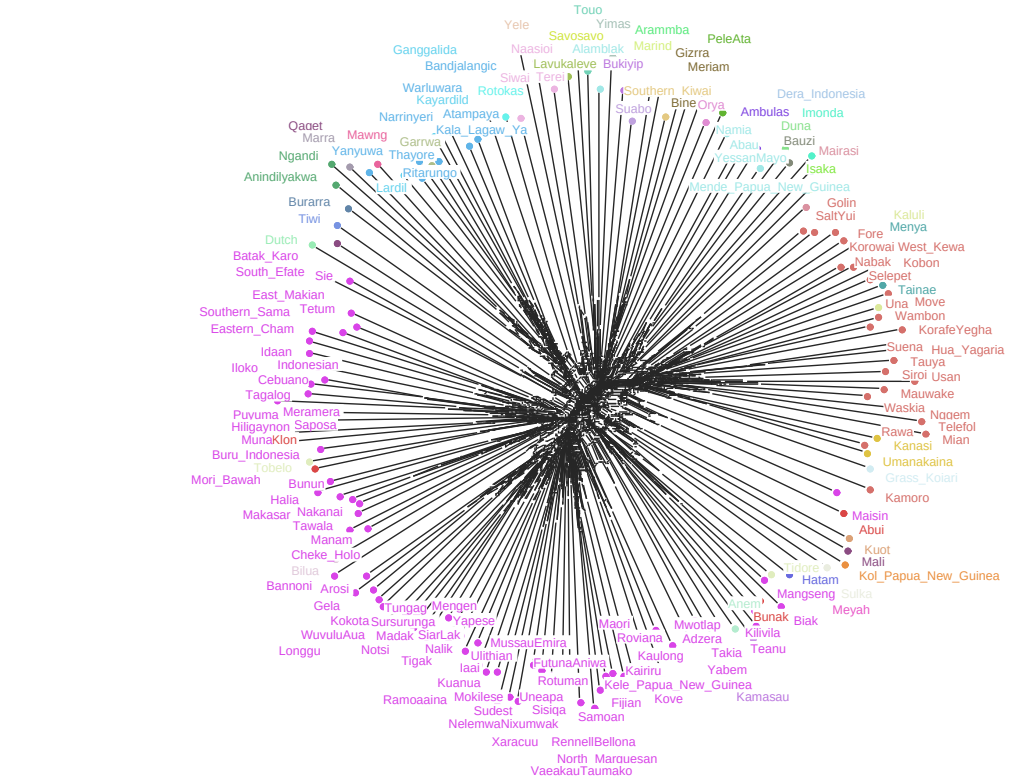


Figure 8: Neighbournet generated by SplitsTree from the weighted uncontrolled distance matrix of the Sahul-survey

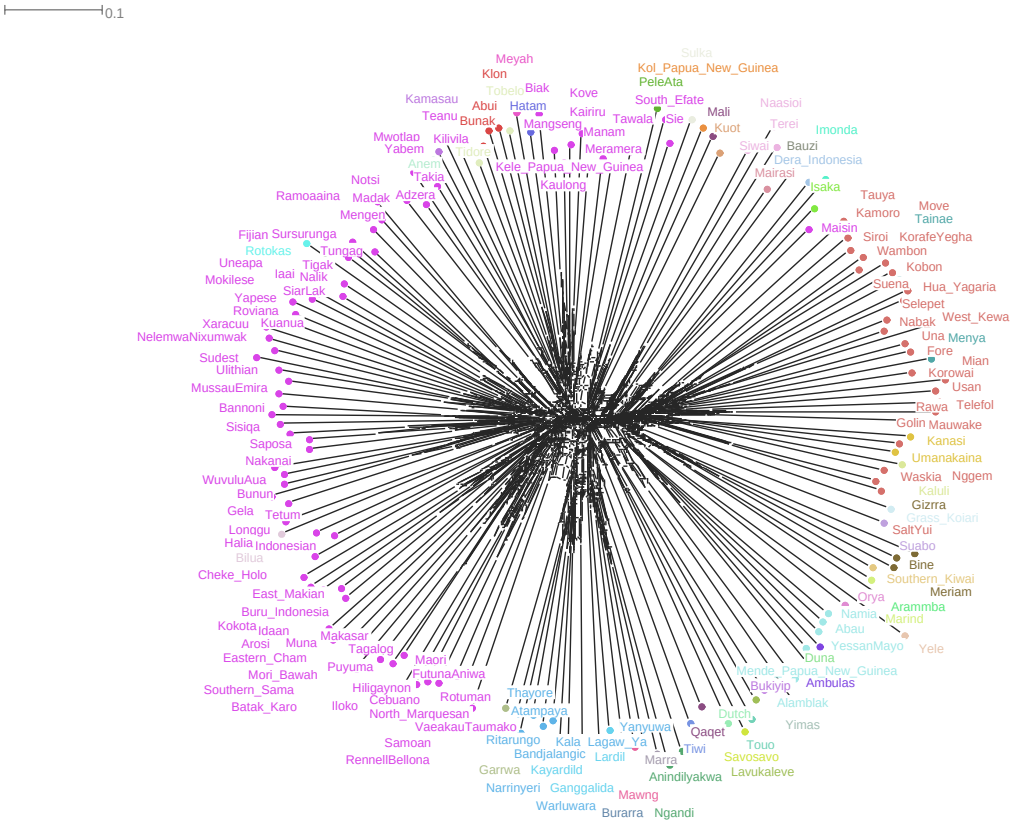


Figure 9: Neighbournet generated by SplitsTree from the weighted controlled distance matrix of the Sahul-survey

Dimensionality reduction: multidimensional scaling (MDS)

(See following pages.)

Summary

The purpose of this project was to extend Hammarström & O'Connor's dependency-sensitive distance in a way that factors out feature dependencies that are due to a particular language family or linguistic area. This refinement was not relevant to Hammarström & O'Connor's original study, as their illustration was based on languages from a single language family and linguistic area. We tested our approach on data from the Sahul survey, which (in addition to being easily available to us) was ideal for our purposes in that it is decidedly *unbalanced* in terms of both families and areas.

We have deliberately refrained from interpreting our results in too much detail, as we are hoping that our more knowledgeable readers will help us with this. To this end, we have made sure to provide several different types of visualisations (both cladistic and non-cladistic) of our weighted/unweighted and controlled/uncontrolled distances.

References

Bickel, B., J. Nichols, T. Zakharko, A. Witzlack-Makarevich, K. Hildebrandt, M. Rießler, L. Bierkandt, F. Zúñiga & B. Lowe (2017) The AUTOTYP typological databases. Version 0.1.0 https://github.com/autotyp/autotyp-data/tree/0.1.0

Chu, Y & T. Liu. (1965) On the shortest arborescence of a directed graph. *Scientia Sinica*.

Edmonds, J. (1967) Optimum Branchings. *Journal of Research of the National Bureau of Standards*. 233-240

Georgiadis, L. (2003) Arborescence optimization problems solvable by Edmonds' algorithm. *Theoretical Computer Science* 71: 233–240.

Hammarström, H. & Forkel, R. & Haspelmath, M. (2018) Glottolog 3.0. Jena: Max Planck Institute for the Science of Human History.

Hammarström, H & L. O'Connor. (2013) Dependency sensitive typological distance. in Borin & Saxena (eds) *Approaches to Measuring Linguistic Differences*. De Gruyter Mouton.

Reesink, G., R. Singer & M. Dunn (2009) Explaining the linguistic diversity of Sahul using population methods. PloS Biology 7(11). e1000241.

Reesink, G. & M, Dunn. (2012) Systematic typological comparison as a tool for investi- gating language history. In Nicholas Evans & Marian Klamer (eds.), Language Documenta- tion & Conservation Special Publication No 5: Melanesian Languages on the Edge of Asia: Challenges for the 21st Century, .

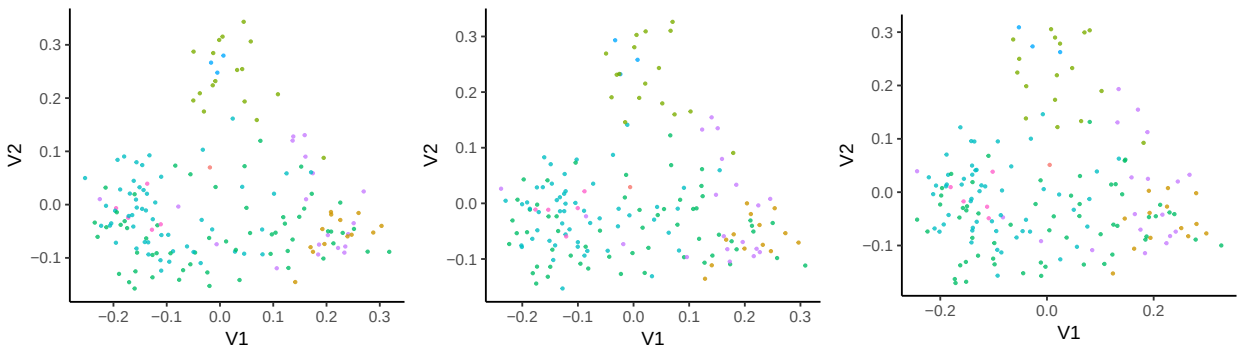


Figure 10: The first and second dimensions of the MDS analysis of each of the three datasets, coloured by macroarea