Language classifications as standardized Newick phylogenetic trees with branch length

Dan Dediu
Language and Genetics,
Max Planck Institute for Psycholinguistics,
Nijmegen, The Netherlands
Dan.Dediu@mpi.nl

April 5, 2017

Abstract

One of the best-known types of non-independence between languages is represented by genetic relationships due to descent from a common ancestor. While there are several classifications of languages into language families, each with its own advantages and disadvantages, they are relatively difficult to use by computational methods due to a lack of standardization. Moreover, certain advanced methods (such as phylogenetics) require not only the topology of the language family tree but also information concerning the amount of evolution that has happened on the tree represented as the branch lengths, and this information is usually missing. This paper presents a method that converts the language classifications provided by four widely-used databases (Ethnologue, WALS, AUTOTYP and Glottolog) into the *de facto* Newick standard, aligns the four most used conventions of unique identifiers for linguistic entities (ISO 639-3, WALS, AUTOTYP and Glottocode), and adds branch length information form a variety of sources (the tree's own topology, an externally given numeric constant or a distance matrix). The `R` scripts, input data and resulting Newick trees are provided in the associated Supplementary Materials in the hope that this will promote the use of advanced quantitative methods in answering questions concerning linguistic diversity and its temporal dynamics.

1 Introduction

Languages are not independent entities and the proper treatment of the various types of non-independence is crucial to drawing valid inferences (e.g., Ladd et al., 2015; Roberts and Winters, 2013). One of the best-known type of non-independence is due to shared ancestry (Campbell and Poser, 2008): the daughter languages tend to be more similar than expected due to the inheritance of characteristics from the mother language, similarity that tends to decrease with increasing temporal separation (this is also knwon as "Galton's problem" and

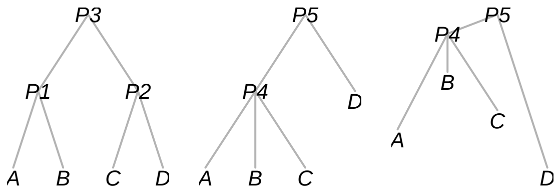

Figure 1: Three language families composed of the same four languages (*A*, *B*, *C* and *D*) but with different structures (left vs centre) and branch length (centre vs right). Time flows downwards from the proto-language at the top (*P3*, *P5* and *P5* respectively) towards the attested languages at the bottom. For example, in the leftmost tree languages *A* and *B* are more closely related than any is to language *C*. In the rightmost tree, language *B* has changed least since its most recent common ancestor (*P4*) with languages *A* and *B*.

applies more generally than linguistics; Mace and Pagel, 1994). Such related languages descending from a common *proto-language* form a *language family*, the internal structure of which is usually represented as a tree. In such a tree, the attested, present-day or recent, languages form the *leaves* (or *terminal nodes*) of the tree and the *internal nodes* represent extinct, mostly unattested, languages[1].

Reliably identifying such *genetic relationships* is a complex problem (Campbell and Poser, 2008; Bowern and Evans, 2014) and many controversies exist, not only in what concerns the so-called "macro-families" but also in the composition and internal structure of more accepted language families. For example, disagreements might exist in the actual set of languages belonging to the same family, in the internal relationships between these languages (the tree *topology*) and the amount of change (the *branch lengths*); see Figure 1.

There are three major difficulties facing modern quantitative methods that need to use such language classifications:

1. the existence of several such classifications,

2. the often non-standardized format these classifications are available in, and,

3. specifically for methods (e.g., phylogenetic) that take into account not only the toplogy of the tree but also the amount of change, the general absence of branch length estimates.

This paper offers a solution to these issues by proposing a standardized representation of language family trees from several classifications using the *de facto* standard *Newick* tree format[2], with added brach length estimates using multiple methods[3]. Here I briefly describe the data sources, methods and output

[1] Of course there are exceptions, such the inclusion of Latin – a well-attested extinct language – at the base of the Romance subfamily (e.g., Chang et al., 2015).

[2] This format is described in `http://evolution.genetics.washington.edu/phylip/newicktree.html`.

[3] *NB* for a few large families (including Indo-European, Austronesian, Bantu and Uto-Aztecan, with this list continuously growing) high-quality posterior samples of trees with branch length derived from cognacy judgments on the basic vocabulary (and even with cal-

formats, while the accompanying Supplementary Materials contain the actual primary data (wherever possible), the `R` code and the resulting Newick language family trees with branch length.

2 Data, methods and outputs

The language family topologies are given by the following four widely used language classifications: the Ethnologue (denoted in the following as **E**; Lewis et al., 2014), the World Atlas of Language Structures Online (WALS, **W**; Dryer and Haspelmath, 2013), AUTOTYP (**A**; Nichols et al., 2013) and Glottolog (**G**; Hammarström et al., 2014). For each of these resources, I downloaded the raw data containing the language classifications and converted them to Newick trees without branch length information.

2.1 Mapping between codes

However, before describing this transformation, it is important to discuss the issue of language *unique identifiers*. Currently, there are several methods for allocating unique (and hopefully also persistent) identifiers to linguistic entities (most often existing or recently extinct languages, but also dialects or proto-languages) and this is far from a simple problem. Here, four standards are relevant: ISO 639-3 codes (tree letters, denoted in the follwing as **i**; `http://www-01.sil.org/iso639-3`), WALS codes (three letters, **w**; `http://wals.info`), AUTOTYP LIDs (numeric, **a**; `http://www.autotyp.uzh.ch`), and Glottocodes (alphanumeric: four letters followed by four digits, **g**; `http://glottolog.org/glottolog/glottologinformation`). As a first step, I mapped these codes for all the linguistic entities present in the four databases, a process made possible by the fact that some of these also give other codes besides their primary one for the linguistic entities therein (Table 1).

Database	Primary code	Other codes
Ethnologue (**E**)	**i**	-
WALS (**W**)	**w**	**i g**
AUTOTYP (**A**)	**a**	**i g**
Glottolog (**G**)	**g**	**i**

Table 1: Codes present in the databases; most databases also give other codes besides their primary code. Legend for codes: **i** = ISO 639-3, **w** = WALS, **a** = AUTOTYP LID, and **g** = Glottocode.

This mapping is contained in the TAB-separated file `./output/code_mappings_iso_wals_autotyp_glottolog` which gives for each unique linguistic entity (the rows) the corresponding ISO 639-3 code (column "ISO"), WALS code (column "WALS"), AUTOTYP LID (column "AUTOTYP"), Glottocode (column "Glottolog"), the name as given by Ethnologue (column "Name.ethn"), by WALS (column "Name.ethn"), by AUTOTYP (column "Name.autotyp") and by Glottolog (column "Name.glottolog"),

ibaration data) using Bayesian phylogenetic methods (e.g. Bouckaert et al., 2012; Dunn et al., 2011) are available, but this is currently not the case for the vast majority of the families.

as well as the geographic coordinates (columns “Latitude” and ”Longitude”) in degrees as given by WALS and Glottolog[4].

2.2 Building the tree topologies

A second step is represented by the gathering of the raw data concerning the structure of the language families and exporting them as pure tree topologies in Newick format (without any branch length information). Each database poses its own challanges as they tend to use particular representations of the genetic relationships between languages. To standardize the process of topology extraction, conversion, exporting to and importing from file, I have written a collection of `R` (R Core Team, 2014) types and functions (file `FamilyTrees.R`) which extends the *de facto* standard for representing phylogenetic trees in `R` as objects of class `phylo` (package `ape2004`; **?**).

The list below summarizies the format of the raw data and its acquisition:

Ethnologue (Lewis et al., 2014) as opposed to the other three databases, the language classification data here is not provided in an easily downloadable form; instead, the Ethnologue website provides[5] (as of February 2015) a webpage (`http://www.ethnologue.com/browse/families`) containing a list with all the language families and links to their respective webpages (e.g., `http://www.ethnologue.com/subgroups/afro-asiatic`). These family webpages were further downloaded and parsed in order to extract the tree structure of the family, as well as the group names and the language names and ISO 639-3 codes[6];

WALS Online (Dryer and Haspelmath, 2013) provides the whole database (including language name, codes, geographic coordinates but also values for more than 130 typological features; `http://wals.info/static/download/wals-language.csv.zip`) under a Creative Commons Attribution-NonCommercial-NoDerivs 2.0 Germany (CC BY-NC-ND 2.0 DE; `http://creativecommons.org/licenses/by-nc-nd/2.0/de/deed.en`); here the important columns are WALS, ISO 639-3 and Glottolog codes, the language names and its “genus” and “family”, resulting in a rather flat three-levels structure;

AUTOTYP (Nichols et al., 2013) the AUTOTYP trees are freely available for download (`http://www.autotyp.uzh.ch/available.html`), use and distribution provided that its source is clearly cited; the format of the language families is similar to the WALS in the sense that each language (row) contains the language names, the AUTOTYP LID, the Glottolog and the ISO 639-3 codes, as well as the “stock”, “mbranch”, “sbranch”, “ssbranch” and “lsbranch” names, each denoting more and more superificial levels (i.e., the “stock” is highest levels corresponding to the language family), and in some cases such an intermediate level might be missing;

[4] When there is a discrepance greater than 1° between the two, WALS wins.

[5] Under a set of conditions contained in the Terms of Use (`www.ethnologue.com/terms-use`) which allow “portions” of the data to be used for “research or educational purposes”.

[6] The data used in this paper and included in the supplementary materials was harvested in Feburary 2015.

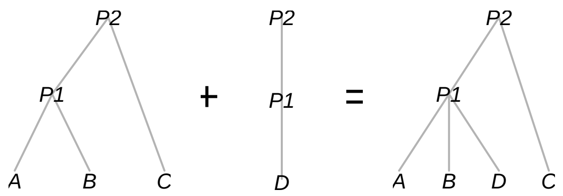

Figure 2: The leftmost partial family tree already exists in the forest when a new language *D* from subfamily *P1* in family *P2* (thus with full path $P2 \rightarrow P1 \rightarrow D$) is added, resulting in the rightmost tree.

Glottolog (Hammarström et al., 2014) as opposed to the other three databases, Glottlog provides the family trees already in a standardized Newick format (`http://glottolog.org/static/trees/tree-glottolog-newick.txt`) under a Creative Commons Attribution-ShareAlike 3.0 Unported License (CC BY-SA 3.0]; `http://creativecommons.org/licenses/by-sa/3.0`) license; here I only expanded the language codes with WALS and AUTOTYP.

The basic idea behind building the standardized tree topologies from these diverse formats[7] is to maintain a forest of (partially) built language family trees to which a new full path from a proto-language to a language is added. The algorithm first tries to identify an already present tree that contains the deeper part of the path (i.e., say adding "Indo-European → Germanic → North-West Germanic → Enligsh" would identify an already existing partial Indo-European tree) and, if so, adds the new (recent) part of the path to the tree. In this manner, the forest of all language families in the database is iteratively built from the ground up (Figure 2).

Table 3 gives various summaries concerning the language family tree topologies successfully converted for each database.

Measure	**E**	**W**	**A**	**G**
# trees	147	214	403	435
# leaves total	7492	2607	2926	15772
Avg leaves	51.0	12.2	7.3	36.3
Max leaves	1545	371	340	3254
Avg levels	4.8	4.0	3.4	4.5
Min levels	3	4	3	3
Max levels	16	4	7	20

Table 2: Various summaries concerning the topologies (no branch length) of the language family trees extracted from the four databases ("normal" GA); **E** = Ethnologue, **W** = WALS, **A** = AUTOTYP, and **G** = Glottolog; "#" stands for "number of..."; the leaves (or non-internal nodes) are various types of lects (most often languages).

[7] Except for Glottolog, which provides a Newick format that requires only very light processing.

Measure	**E**	**W**	**A**	**G**
# trees	147	214	403	435
# leaves total	7492	2607	2926	15772
Avg leaves	51.0	12.2	7.3	36.3
Max leaves	1545	371	340	3254
Avg levels	4.8	4.0	3.4	4.5
Min levels	3	4	3	3
Max levels	16	4	7	20

Table 3: Various summaries concerning the topologies (no branch length) of the language family trees extracted from the four databases (”slow” GA); **E** = Ethnologue, **W** = WALS, **A** = AUTOTYP, and **G** = Glottolog; “#” stands for “number of...”; the leaves (or non-internal nodes) are various types of lects (most often languages).

2.3 The Newick trees and the naming convention

An interesting question concerns the format in which these tree topologies (and later, branch lengths) should be exported. I opted for the *de facto* standard Newick tree format[8] widely used in evolutionary biology, read and exported by many software packages and libraries, and able to represent rooted and unrooted trees, with our without leaf and internal node names, and with or without branch lengths. The basic idea is that subtrees are enclosed within parentheses “()” and the branch length is given as a number immediately following the branch and separated from it by “:”. For example, the leftmost tree in Figure 2 can be represented as (language = leaf, proto-languages or groups = internal nodes, for simplicity all branches have the same length of 1):

Representation	Comments
((,),);	just the structure
((*A*,*B*),*C*);	with leaf names
((*A*,*B*)*P1*,*C*)*P2*;	with group names
((*A*:1,*B*:1),*C*:1);	with branch length
((*A*:1,*B*:1)*P1*:1,*C*:2)*P2*:1;	with everything

The language and group/proto-language names must include not only the actual name as given by the particular classification (which could very well differ between classifications; as a trivial example, Ethnologue calls the language with code ISO 639-3 “English” while Glottolog calls it “Standard English”, but there are much more dramatic differences between the databases), but also the various unique identifiers this linguistic entity might have. Therefore, I opted for a standardized node name that follows the convention:

‘NAME [i-I][w-W][a-A][g-G]’

where CAPITAL LETTERS denote variables. NAME is the entity name as given by the classification[9], followed by a SPACE and the four unique codes

[8] See `http://evolution.genetics.washington.edu/phylip/newicktree.html` and `http://en.wikipedia.org/wiki/Newick_format` for details on the actual format.

[9] Given that some characters have a special meaning in the Newick format, I have enforced the following character substitutions: ,→. ’→‘ (→{)→} TAB→SPACE :→| ;→| and char-

I (ISO 639-3), W (WALS), A (AUTOTYP) and G (Glottocode), where each and all can be missing or can have multiple values (in which case the values are separated by "-"). A few examples are (WALS classification, Indo-European family):

'German {Zurich}
[i-gsw][w-gzu][a-1305-1306-1307-1308-1309-1310][g-swis1247]'
'Urdu [i-urd][w-urd][a-2671][g-urdu1245]'
'Romani {Sepecides} [i-][w-rse][a-][g-]'
'Germanic [i-][w-][a-][g-]'.

2.4 The branch length methods

The methods I used to add branch lengths to the tree topologies can be divided into:

a) methods that depend only on the topology: (1) constant, (2) proportional and (3) grafen,

b) methods that generate the branch length and topology from a distance matrix: (4) nj, and

c) methods that map a given distance matrix onto the topology: (5) nnls and (6) ga.

The methods of type (a) only need a tree topology T (and possibly a numeric constant $k > 0$). Method (1) computes branch lengths such that the sum of the branch lengths for every *root* → *leaf* path in the tree is equal to the constant k, meaning that the same amount of evolution k has happened on all branches. For example, for the leftmost tree in Figure 2 and $k = 1.0$, the resulting tree is

$$((A : 0.667, B : 0.667)P1 : 0.333, C : 1)P2;$$

Method (2) simply gives each branch the same length k such that the amount of evolution on a path is proportional to the number of splits on that path; here the result is

$$((A : 1, B : 1)P1 : 1, C : 1)P2;$$

Method (3) is a reimplementation of Grafen (1989) whereby first each node is given a 'height' defined as the number of leaves of its subtree minus 1 (0 for the leaves), after which branch lengths are computed as the difference between the height of the lower and the upper nodes of the branch; our tree is then:

$$((A : 1, B : 1)P1 : 1, C : 2)P2;$$

Method (4) is the only one of type (b) used here and is a clasic method in phylogenetics, the so-called "Neighbor-Joining" (or NJ) alorithm (Saitou and Nei, 1987), essentially a clustering method that iteratively joins taxa into higher groupings (see `en.wikipedia.org/wiki/Neighbor_joining` for a good explanation). Given a language family topology T and a distance matrix between a set of languages D, I extract the languages in T and the submatrix of distances

acters with diacritics into their plain form (e.g., á→a and ã→a) while leaving unaltered the other characters.

between them D_T (*NB* it is possible that not for all pairs of languages there is a distance defined in D, resulting in a submatrix D_T with missing data for those pairs of languages), and then use NJ (as implemented by `R`'s function `njs()` in package `ape` version 3.2; Paradis et al., 2004) to construct the corresponding phylogenetic tree. Thus, this method does not consider the actual topology in T but only the set of languages and the distances between them. For our example and the distance matrix (please note that the distances are given only between the languages – the leaves – and do not concern the proto-languages – the internal nodes)

$$D = \begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 0 & 2.1 & 3.9 \\ 2.1 & 0 & 4.2 \\ 3.9 & 4.2 & 0 \end{pmatrix} \end{array}$$

which approximates the distances between the tree languages in the right-most tree of Figure 2 assuming method (1) with $k = 2.0$, we have the NJ tree

$$(C : 3, B : 1.2, A : 0.9);$$

It is important to note that NJ does not know about the internal structure of the original family tree (in this case the *P1* internal node) and it might produce very different topologies from the ones given by the actual classifications.

Methods (5) and (6) try to use both the given language family's tree topology T and the information contained in the inter-language distance matrix D by computing branch lengths that best approximate the original distances in D (i.e., if one creates a new distance matrix between the languages D' by adding up the total branch lengths one needs to travel in the tree from one language to the other, then $D' \approx D$). Method (5) computes the branch lengths by using a non-negative least squares approach as implemented by `R`'s function `nnls.tree()` in package `phangorn` version 1.99 (Schliep, 2011), resulting in this case in

$$((A : 1.05, B : 1.05)P1 : 0.975, C : 2.02)P2;$$

Finally, method (6) estimates the branch lengths using a standard genetic algorithm (as implemented by `R`'s function `ga()` in package `GA` version 2.2 Scrucca, 2013). Given a topology T with n branches, I need to compute n real positive numbers, each representing the length of a branch in T such that the resulting distance matrix D' is a good approximation of the original distances D. In this genetic algorithms approach, I defined the "genome" as composed of n real-valued "genes", and the "fitness function" for a particular such genome $G = (g_1, g_2, ..., g_n)$ computes the SSE (sum of squared errors) between the original distances D and the current distances $D'(G)$ between languages if the topology T had the branch lengths g_1, g_2, ... g_n. The genetic algorithm finds the best solution $G^* = (g_1^*, g_2^*, ..., g_n^*)$ that minimizes the fitness function (the SSE) using a population size of 100 individuals for at most 10,000 iterations (or when the fitness does not change for 100 iterations). For our example, some possible trees could be

$$((A : 0.901, B : 1.2)P1 : 1.13, C : 1.87)P2;$$

$$((A : 0.9, B : 1.2)P1 : 1.15, C : 1.85)P2;$$

$$((A : 0.9, B : 1.2)P1 : 1.91, C : 1.09)P2;$$

Please note that due to the random nature of the genetic algorithm and possibly the non-uniqueness of the solution (multiple optima), the best solution might vary between runs. Methods (5) and (6) have similar goals and produce very similar results, but approach them in very different ways; method (5) is less robust than method (6) (it fails for certain topologies and distance matrices), while method (6) is much slower, especially for very large trees, and might produce non-unique solutions.

2.5 The distance matrices

There are many potentially meaningful distances between languages, and while the framework and `R` code introduced here can accomodate new ones, I have used in this paper the following:

a) distances based on vocabulary: (1) ASJP16,

b) distances based on geography: (2) great-circle,

c) distances based on WALS: (3) gower and (4) euclidean, with and without missing data imputation,

d) distance based on AUTOTYP: (5) gower with missing data using only the variables with a single datapoint per language (this distance was computed by Balthasar Bickel), and

e) distances based on the tree topology: Maurits and Griffiths (2014)'s "genetic method" applied to the WALS (6), Ethnologue (7), Glottolog (8) and AUTOTYP (9) classifications.

Method (1) uses the distances between languages provided by The Automated Similarity Judgment Program version 16 (ASJP16; Wichmann et al., 2013) and the ASJP software (version 2.1), freely available under a Creative Commons Attribution 3.0 (CC BY 3.0, `http://creativecommons.org/licenses/by/3.0`) license from the authors' website `http://asjp.clld.org`. These distances are computed on the basis of standardized short wordlists transcribed in a reduced set of symbols using a normalized Levenstein distance (for details see Bakker et al., 2009). I further processed and converted this database into a distance matrix between languages using ISO 639-3 codes as language identifiers[10], resulting in a 3932 × 3932 matrix with no missing data.

Method (2) computes the geographic (great circle) distances between the languages' geographic coordinates using `R`'s function `distm()` in package `geosphere` version 1.3 (Hijmans, 2014), resulting in a 7494 × 7494 matrix with no missing data.

Methods (3) and (4) use the WALS typological database to compute distances between languages using their feature values. I used the methods implemented by `R`'s function `daisy()` in package `cluster` version 2.0.1 (Maechler et al., 2015), namely "gower" (method 3; Gower, 1971) which standardizes each feature to the $[0, 1]$ interval, and "euclidean" (method 4) which computes the

[10] This conversion required limited manual editing including the replacement of some non-ASCII characters in the language descriptors and some of the 26-character language identifiers exported by the ASJP v2.1 software.

standard Euclidean distance in an n-dimensional real space. Given the enormously high proportion of missing data in the WALS database (85.1% cells), I have computed these distances also doing a simple missing data imputation whereby the missing data was replaced by the mode (i.e., the most frequent value) of the corresponding typological variable. With these, I obtained the follwing distance matrices: gower with missing data (2679×2679, 48.9% missing data cells), gower with missing data imputation (2679×2679, no missing data), euclidean with missing data (2679×2679, 48.9% missing data cells), and euclidean with missing data imputation (2679×2679, no missing data).

Method (5) uses the AUTOTYP typological database to compute distances between languages using their feature values. This method also uses `R`'s function `daisy()` in package `cluster` version 2.0.1 (Maechler et al., 2015) with argument "gower" (Gower, 1971), without missing data imputation, resulting in a 2928×2928 distance matrix with 57.6% missing data cells.

Methods (6) – (9) use the "genetic method" introduced in Maurits and Griffiths (2014) whereby branch lengths are assign based on the topology of the family tree in such a way that languages that share k intermediate nodes on their paths from the root are separated by the distance $d = M - \sum_{i=1}^{n} \alpha^i$ (with M being the maximum possible diatance and α empirically fixed at 0.69); it is important to note that by defintion these distances are not defined for pairs of languages that belog to different families and are defined for any pair of languages that belogn to the same family (therefore the percent of missing data is uninformative in this case). I reimplemented this method in `R`[11] and applied it to each of the four classifications, resulting in the follwing distance matrices: MG2015 using the WALS classification (2607×2607), the Ethnologue classification (7492×7492), the Glottolog classification (15772×15772), and the AUTOTYP classification (2926×2926).

2.6 The family trees with branch length

Thus, for each combination of *classification* (Ethnologue, WALS, AUTOTYP, Glottolog) by *method* (no branch length, constant, proportional, grafen, nj, nnls, ga) and, for the last three methods, also by *distance* matrix (asjp16, great-circle, wals-gower, wals-gower+imputation, wals-euclidean, wals-euclidean+imputation, autotyp-gower, mg2015+wals, mg2015+ethnologue, mg2015+glottolog, mg2015+autotyp), I produced a set of phylogenetic trees in Newick format as described above. Each of these sets was saved in two formats: a TAB-separated CSV file, and a Nexus file, containing essentially the same information but easier to load into various phylogenetic software packages.

The first format is a standard TAB-separated CSV file with a standardized name of the form `CLASSIFICATION-newick-METHOD&PARAMETERS.csv` (e.g., `autotyp-newick-nj+autotyp.csv` and `glottolog-newick-nnls+wals(gower,mode).csv`) in the `./output/CLASSIFICATION/` directory. It contains the language families (one per row, except the first row which is the header), and for each family it gives the family name (as defined by the classification), the success or failure of the method, some relevant comments (for examply, why the methods has failed), and the actual tree in Newick format (or an empty string ‘’ if the method has failed).

[11] Many thanks to Luke Maurits for helping to clarify the inner workings of the method.

The second format is a standard Nexus file (Maddison et al., 1997) with a standardized name of the form `CLASSIFICATION-nexus-METHOD&PARAMETERS.nex` (e.g., `autotyp-nexus-nj+autotyp.nex` and `glottolog-nexus-nnls+wals(gower,mode).nex`) in the `./output/CLASSIFICATION/` directory. These Nexus files contain only the `trees` block with the `translate` list[12] and the named trees (the language family names as given by the classification) in Newick format.

Summaries about these trees are given in Appendix A.

To explore the (dis)agreement between the trees produced by these methods, for each language family (within a given classification as families do not generally mean the same thing across classifications) and pair of methods, I computed two distances[13] between these corresponding trees: one that considers only how similar they are in their topology ("PH85", Penny and Hendy, 1985; Rzhetsky and Nei, 1992) and another that also takes into account the branch length ("score", Kuhner and Felsenstein, 1994). For details please see Appendix B.

2.7 A note on robustness

An important question concerns the robustness of these branch-length inference methods against violations of the conditions on the distances matrix d. A matrix must meet four conditions for it to be a true distance matrix:

1. the diagonal is zero: $d_{ii} = 0$ for all $1 < i < N$;
2. the off-diagonal is positive: $d_{ij} \geq 0$ for all $1 < i \neq j < N$;
3. the matrix is symmetric: $d_{ij} = d_{ji}$ for all $1 < i, j < N$;
4. the triangle inequality is satisfied: $d_{ij} \leq d_{ik} + d_{kj}$ for all $1 < i, j, k < N$.

We have generated a set of matrices (bases on our test distances matrix D) that violate each of the conditions individually (and all of them), as follows

The original test distances matrix:

$$D = \begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 0 & 2.1 & 3.9 \\ 2.1 & 0 & 4.2 \\ 3.9 & 4.2 & 0 \end{pmatrix} \end{array}$$

Violating (1): non-zero elements on the diagonal:

$$D_d = \begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 3 & 2.1 & 3.9 \\ 2.1 & 1.3 & 4.2 \\ 3.9 & 4.2 & 2 \end{pmatrix} \end{array}$$

Violating (2): off-diagonal negative elements:

$$D_n = \begin{array}{c} \\ A \\ B \\ C \end{array} \begin{array}{c} \begin{array}{ccc} A & B & C \end{array} \\ \begin{pmatrix} 0 & -2.1 & 3.9 \\ -2.1 & 0 & 4.2 \\ 3.9 & 4.2 & 0 \end{pmatrix} \end{array}$$

[12] The `R` script is capable to generate or not the `translate` list, by default it does in order to increase human readablity and make the files importable by some phylogenetic software.

[13] As implemented by `R`'s function `dist.topo()` in package `ape` v.3.2 (Paradis et al., 2004).

Violating (3): asymmetric matrix:

$$D_s = \begin{array}{c|ccc} & A & B & C \\ \hline A & 0 & 2.1 & 3.9 \\ B & 3.3 & 0 & 4.2 \\ C & 3.9 & 1.3 & 0 \end{array}$$

Violating (4): the triangle inequality:

$$D_t = \begin{array}{c|ccc} & A & B & C \\ \hline A & 0 & 5.1 & 1.9 \\ B & 5.1 & 0 & 2.2 \\ C & 1.9 & 2.2 & 0 \end{array}$$

Violating all conditions (1)-(4) simultaneously:

$$D_a = \begin{array}{c|ccc} & A & B & C \\ \hline A & 3.1 & -5.1 & 1.9 \\ B & 5.1 & 0 & 2.6 \\ C & 1.9 & 2.2 & 0 \end{array}$$

For each of the three methods (4, 5 and 6) that take a distances matrix as a parameter, we first tested if the method crashes when fed one of these "bad" distances matrix and second, how close to the true branch lengths the estimated trees are.

2.7.1 Robustness of nj

The trees are (in the order: original D, D_d, D_n, D_s, D_t, and D_a):

$$D : (C : 3, B : 1.2, A : 0.9);$$

$$D_d : (C : 3, B : 1.2, A : 0.9);$$

$$D_n : (C : 1.95, B : 2.25, A : 1.95);$$

$$D_s : (C : 0.95, B : 0.35, A : 2.95);$$

$$D_t : (C : 0, B : 3.2, A : 2.9);$$

$$D_a : (C : 0, B : 3.2, A : 2.9);$$

2.7.2 Robustness of nnls

The trees are (in the order: original D, D_d, D_n, D_s, D_t, and D_a):

$$D : ((A : 1.05, B : 1.05)P1 : 0.975, C : 2.02)P2;$$

$$D_d : ((A : 1.05, B : 1.05)P1 : 0.975, C : 2.02)P2;$$

$$D_n : ((A : 0, B : 0)P1 : 2.02, C : 2.02)P2;$$

$$D_s : ((A : 1.42, B : 1.42)P1 : 0, C : 1.42)P2;$$

$$D_t : ((A : 1.53, B : 1.53)P1 : 0, C : 1.53)P2;$$

$$D_a : ((A : 1.53, B : 1.53)P1 : 0, C : 1.53)P2;$$

2.7.3 Robustness of ga

The trees are (in the order: original D, D_d, D_n, D_s, D_t, and D_a):

$$D : ((A : 0.901, B : 1.2)P1 : 1.13, C : 1.87)P2;$$

$$D : ((A : 0.9, B : 1.2)P1 : 1.15, C : 1.85)P2;$$

$$D : ((A : 0.9, B : 1.2)P1 : 1.91, C : 1.09)P2;$$

$$D_d : ((A : 0.9, B : 1.2)P1 : 1.28, C : 1.72)P2;$$

$$D_n : ((A : 0.0331, B : 0.0324)P1 : 2.13, C : 1.86)P2;$$

$$D_s : ((A : 1.93, B : 0.775)P1 : 1.17, C : 0.808)P2;$$

$$D_t : ((A : 2.22, B : 2.47)P1 : 0.0554, C : 0.023)P2;$$

$$D_a : ((A : 0.00925, B : 0.179)P1 : 0.702, C : 1.34)P2;$$

2.7.4 Robustness: conclusions

In conclusions, all three methods can deal with violations of the distance matrix conditions gracefully, neither of them crashes and the trees produced still seem meaningful.

2.8 The influence of GA parameters on the branch length

The parameters of the GA (`GA.MAXITER` = the maximum number of iterations to run, `GGA.POPSIZE` = the population size, and `GGA.CONSTANTRUN` = the number of consecutive generations with the same fitness needed to stop the search prematurely) may in theory have an important impact on the solution found (i.e., branch length) and certainly on the computational costs necessary for this solution to be found. Therefore, I ran two conditions, as follows:

normal: `GA.MAXITER` = 10000, `GGA.POPSIZE` = 100, and `GGA.CONSTANTRUN` = 100; and

slow: `GA.MAXITER` = 50000, `GGA.POPSIZE` = 150, and `GGA.CONSTANTRUN` = 200.

The computational costs are very different, and, for comparison, I ran the three conditions on the same compute cluster node using a dedicated CPU for each classification, and the reported times are wall clock (i.e., real) times: 'normal' required about 10 days, while 'slow' was forcefully stopped after 52 days (when all trees converged except for glottolog+mg2015).

How do the estimated branch lengths compare?

We compared 'normal' and 'slow' by computing the Pearson's r, paired t-test, and Euclidean distance between the corresponding branch lengths for each family tree for each classification and distance matrix. Pearsons correlations vary between 0.13 and 1.00, with mean 0.98 and median 1.00. The Euclidean distances vary between 0.00 and 2229.63, with mean 4.23 and median 0.02. Only 247 paired t-tests (out of 3022 successful comparisons) are significant at the 0.05 level.

Interestingly, the Pearson correlation between the fitness values for the 'normal' and 'slow' is r = 0.93, p = 0, with the paired t-test not significant: t(3182.0)

= 1.52, p = 0.1298. Moreover, the number of generations required to reach this fitness optimum are correlated: r = 0.89, p = 0, but 'slow' requires significantly more generations than 'normal' as shown by the paired t-tests: mean diff. normal - slow = -1174.61, t(3182.0) = -20.14, p = 5.178e-85.

Thus, it seems that the much higher computational costs required by 'slow' are not justified in terms of better fit, and the resulting branch lengths are very similar. Therefore, the 'normal' parameters `GA.MAXITER` = 10000, `GGA.POPSIZE` = 100, and `GGA.CONSTANTRUN` = 100, are enough for our purposes.

3 Conclusions

This paper describes a flexible method for producing standardized language family trees in the Newick format with branch length using a variety of linguistic classifications, methods and distances between languages. Accompanying this paper as Supplementary Online Materials is an archive (tar.xz) containing (where possible given the licensing terms) the input data, the `R` code, the output files, and the source of this paper (written using `Sweave` Leisch, 2002), as well as short descriptions (`ReadMe.txt` files) and license terms. My own `R` code is released under a GPL v2 license, is relatively well-commented and tested, and is free to use and modify as long as the terms of the license are respected and this paper is cited[14]. Especially the high-level functions in `./code/FamilyTrees.R` might be useful for manipulating such trees and applying new distance matrices to family tree topologies; the file `./code/StandardizedTrees.r` can be consulted as an example of using them and it also contains useful functions.

Acknowledgments

Many thanks to the authors of the databases used here for making their data freely available, to Balthasar Bickel for computing the AUTOTYP distance and agreeing to making it freely available with this paper, to Luke Maurits for clariying their "genetic method", and to Harald Hammarström and Seán Roberts for discussions and feedback. During this project I was funded by an NWO (Netherlands Organisation for Scientific Research) VIDI grant number 016.124.315.

References

Bakker, D., Müller, A., Velupillai, V., Wichmann, S., Brown, C. H., Brown, P., Egorov, D., Mailhammer, R., Grant, A., and Holman, E. W. (2009). Adding typology to lexicostatistics: A combined approach to language classification. *Linguistic Typology*, 13(1).

Bouckaert, R., Lemey, P., Dunn, M., Greenhill, S. J., Alekseyenko, A. V., Drummond, A. J., Gray, R. D., Suchard, M. A., and Atkinson, Q. D. (2012). Mapping the origins and expansion of the Indo-European language family. *Science*, 337(6097):957–960.

[14] Why not an `R` package, you migth ask? I feel this application is very specific and the code is mainly intended to be changed and adapted (or just serve as inspiration) for other specific problems the users might have, instead of being used as it is.

Bowern, C. and Evans, B. (2014). *The Routledge Handbook of Historical Linguistics*. Routledge.

Campbell, L. and Poser, W. J. (2008). *Language Classification: History and Method*. Cambridge University Press.

Chang, W., Cathcart, C., Hall, D., and Garrett, A. (2015). Ancestry-constrained phylogenetic analysis supports the Indo-European steppe hypothesis. *Language*, 91(1):194–244.

Dryer, M. S. and Haspelmath, M., editors (2013). *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig. `http://wals.info`.

Dunn, M., Greenhill, S. J., Levinson, S. C., and Gray, R. D. (2011). Evolved structure of language shows lineage-specific trends in word-order universals. *Nature*, 473:79–82.

Gower, J. C. (1971). A general coefficient of similarity and some of its properties. *Biometrics*, 27(4):857–871.

Grafen, A. (1989). The phylogenetic regression. *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, 326(1233):119–157.

Hammarström, H., Forkel, R., Haspelmath, M., and Nordhoff, S., editors (2014). *Glottolog 2.3*. Max Planck Institute for Evolutionary Anthropology, Leipzig. `http://glottolog.org`.

Hijmans, R. J. (2014). *geosphere: Spherical Trigonometry*. R package version 1.3-11.

Kuhner, M. K. and Felsenstein, J. (1994). A simulation comparison of phylogeny algorithms under equal and unequal evolutionary rates. *Molecular Biology and Evolution*, 11(3):459–468.

Ladd, D. R., Roberts, S. G., and Dediu, D. (2015). Correlational studies in typological and historical linguistics. *Annual Review of Linguistics*, 1(1):221–241.

Leisch, F. (2002). Sweave: Dynamic generation of statistical reports using literate data analysis. In Härdle, W. and Rönz, B., editors, *Compstat 2002 — Proceedings in Computational Statistics*, pages 575–580. Physica Verlag, Heidelberg. ISBN 3-7908-1517-9.

Lewis, M. P., Simons, G. F., and Fennig, C. D., editors (2014). *Ethnologue: Languages of the World*. Dallas, Tex.: SIL International, 17 edition. `http://www.ethnologue.com`.

Mace, R. and Pagel, M. (1994). The comparative method in anthropology. *Current Anthropology*, 35:549–564.

Maddison, D. R., Swofford, D. L., and Maddison, W. P. (1997). Nexus: An extensible file format for systematic information. *Systematic Biology*, 46(4):590–621.

Maechler, M., Rousseeuw, P., Struyf, A., Hubert, M., and Hornik, K. (2015). cluster: Cluster analysis basics and extensions. R package version 2.0.1.

Maurits, L. and Griffiths, T. L. (2014). Tracing the roots of syntax with Bayesian phylogenetics. *Proceedings of the National Academy of Sciences*, 111(37):13576–13581.

Nichols, J., Witzlack-Makarevich, A., and Bickel, B. (2013). The AUTOTYP genealogy and geography database: 2013 release. Electronic database, `http://www.autotyp.uzh.ch`.

Paradis, E., Claude, J., and Strimmer, K. (2004). APE: analyses of phylogenetics and evolution in R language. *Bioinformatics*, 20:289–290.

Penny, D. and Hendy, M. D. (1985). The use of tree comparison metrics. *Systematic Biology*, 34(1):75–82.

R Core Team (2014). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. `http://www.R-project.org`.

Roberts, S. and Winters, J. (2013). Linguistic diversity and traffic accidents: Lessons from statistical studies of cultural traits. *PLoS ONE*, 8(8):e70902.

Rzhetsky, A. and Nei, M. (1992). A simple method for estimating and testing minimum-evolution trees. *Mol. Biol. Evol*, 9(5):945–967.

Saitou, N. and Nei, M. (1987). The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4(4):406–425.

Schliep, K. (2011). phangorn: phylogenetic analysis in r. *Bioinformatics*, 27(4):592–593.

Scrucca, L. (2013). GA: A package for genetic algorithms in R. *Journal of Statistical Software*, 53(4):1–37.

Wichmann, S., Müller, A., Wett, A., Velupillai, V., Bischoffberger, J., Brown, C. H., Holman, E. W., Sauppe, S., Molochieva, Z., Brown, P., Hammarström, H., Belyaev, O., List, J.-M., Bakker, D., Egorov, D., Urban, M., Mailhammer, R., Carrizo, A., Dryer, M. S., Korovina, E., Beck, D., Geyer, H., Epps, P., Grant, A., and Valenzuela, P. (2013). The ASJP database (version 16). `http://asjp.clld.org/`.

Appendix A: Family trees summaries

This Appendix contains summaries concerning the language family trees generated using the classifications, methods and distances discussed in this paper, using the same conventions as in Table 3.

Table 4 gives the summaries for the *constant* method with $k = 1.00$.

Parameter	Measure	E	W	A	G
k=1.00	# trees	147	214	403	435
	# trees success	147	214	403	435
	# leaves total	7492	2607	2926	15772
	Avg leaves	51.0	12.2	7.3	36.3
	Max leaves	1545	371	340	3254
	Avg levels	4.8	4.0	3.4	4.5
	Min levels	3	4	3	3
	Max levels	16	4	7	20

Table 4: Summaries for constant=1.00.

Table 5 gives the summaries for the *proportional* method with $k = 1.00$.

Parameter	Measure	E	W	A	G
k=1.00	# trees	147	214	403	435
	# trees success	147	214	403	435
	# leaves total	7492	2607	2926	15772
	Avg leaves	51.0	12.2	7.3	36.3
	Max leaves	1545	371	340	3254
	Avg levels	4.8	4.0	3.4	4.5
	Min levels	3	4	3	3
	Max levels	16	4	7	20

Table 5: Summaries for proportional=1.00.

Table 6 gives the summaries for the *grafen* method.

Parameter	Measure	E	W	A	G
-	# trees	147	214	403	435
	# trees success	147	214	403	435
	# leaves total	7492	2607	2926	15772
	Avg leaves	51.0	12.2	7.3	36.3
	Max leaves	1545	371	340	3254
	Avg levels	4.8	4.0	3.4	4.5
	Min levels	3	4	3	3
	Max levels	16	4	7	20

Table 6: Summaries for grafen.

Table 7 gives the summaries for the *nj* method with the various distance matrices.

Parameter	**Measure**	**E**	**W**	**A**	**G**
asjp16	# trees	147	214	403	435
	# trees success	82	70	108	86
	# leaves total	3810	1973	2035	1926
	Avg leaves	46.5	28.2	18.8	22.4
	Max leaves	789	297	306	430
	Avg levels	7.4	7.1	6.3	5.6
	Min levels	2	2	2	2
	Max levels	33	21	24	25
geo	# trees	147	214	403	435
	# trees success	102	79	128	141
	# leaves total	7124	2425	2547	4501
	Avg leaves	69.8	30.7	19.9	31.9
	Max leaves	1510	370	337	830
	Avg levels	17.2	11.5	8.6	10.4
	Min levels	2	2	2	2
	Max levels	220	67	69	162
wals(gower)	# trees	147	214	403	435
	# trees success	49	49	75	40
	# leaves total	1611	1807	1577	691
	Avg leaves	32.9	36.9	21.0	17.3
	Max leaves	306	325	314	127
	Avg levels	9.4	9.8	7.1	6.4
	Min levels	2	2	2	2
	Max levels	31	41	46	21
wals(euclidean)	# trees	147	214	403	435
	# trees success	49	49	75	39
	# leaves total	1611	1807	1577	676
	Avg leaves	32.9	36.9	21.0	17.3
	Max leaves	306	325	314	127
	Avg levels	8.8	9.7	6.5	6.4
	Min levels	2	2	2	2
	Max levels	44	45	31	24
wals(gower,mode)	# trees	147	214	403	435
	# trees success	76	79	121	64
	# leaves total	2231	2442	2229	945
	Avg leaves	29.4	30.9	18.4	14.8
	Max leaves	337	371	314	128
	Avg levels	11.0	11.1	8.5	7.3
	Min levels	2	2	2	2
	Max levels	64	66	50	28
wals(euclidean,mode)	# trees	147	214	403	435
	# trees success	76	79	121	64
	# leaves total	2231	2442	2229	945
	Avg leaves	29.4	30.9	18.4	14.8
	Max leaves	337	371	314	128
	Avg levels	11.6	12.0	8.8	8.0
	Min levels	2	2	2	2
	Max levels	64	88	59	42

Parameter	**Measure**	**E**	**W**	**A**	**G**
autotyp	# trees	147	214	403	435
	# trees success	30	38	48	32
	# leaves total	365	462	559	211
	Avg leaves	12.2	12.2	11.6	6.6
	Max leaves	57	53	59	23
	Avg levels	6.2	6.2	5.6	4.1
	Min levels	2	2	2	2
	Max levels	19	23	19	12
mg2015(wals)	# trees	0	214	0	0
	# trees success	0	79	0	0
	# leaves total	0	2442	0	0
	Avg leaves	0.0	30.9	0.0	0.0
	Max leaves	0	371	0	0
	Avg levels	0.0	14.7	0.0	0.0
	Min levels	0	2	0	0
	Max levels	0	163	0	0
mg2015(ethnologue)	# trees	147	0	0	0
	# trees success	104	0	0	0
	# leaves total	7419	0	0	0
	Avg leaves	71.3	0.0	0.0	0.0
	Max leaves	1545	0	0	0
	Avg levels	15.8	0.0	0.0	0.0
	Min levels	2	0	0	0
	Max levels	129	0	0	0
mg2015(glottolog)	# trees	0	0	0	435
	# trees success	0	0	0	222
	# leaves total	0	0	0	15507
	Avg leaves	0.0	0.0	0.0	69.9
	Max leaves	0	0	0	3254
	Avg levels	0.0	0.0	0.0	11.1
	Min levels	0	0	0	2
	Max levels	0	0	0	77
mg2015(autotyp)	# trees	0	0	403	0
	# trees success	0	0	128	0
	# leaves total	0	0	2605	0
	Avg leaves	0.0	0.0	20.4	0.0
	Max leaves	0	0	340	0
	Avg levels	0.0	0.0	10.1	0.0
	Min levels	0	0	2	0
	Max levels	0	0	162	0

Table 7: Summaries for nj.

Table 8 gives the summaries for the *nnls* method with the various distance matrices.

Parameter	**Measure**	**E**	**W**	**A**	**G**
asjp16	# trees	147	214	403	435
	# trees success	100	91	144	123
	# leaves total	3846	2015	2107	2000
	Avg leaves	38.5	22.1	14.6	16.3
	Max leaves	789	297	306	430
	Avg levels	4.4	3.4	3.3	4.0
	Min levels	2	2	2	2
	Max levels	15	4	7	15
geo	# trees	147	214	403	435
	# trees success	132	108	172	193
	# leaves total	7184	2483	2635	4605
	Avg leaves	54.4	23.0	15.3	23.9
	Max leaves	1510	370	337	830
	Avg levels	4.7	3.9	3.7	4.1
	Min levels	2	2	2	2
	Max levels	15	4	7	17
wals(gower)	# trees	147	214	403	435
	# trees success	88	97	149	89
	# leaves total	1017	1329	1130	486
	Avg leaves	11.6	13.7	7.6	5.5
	Max leaves	100	220	115	38
	Avg levels	3.8	3.1	3.0	3.1
	Min levels	2	2	2	2
	Max levels	11	4	7	6
wals(euclidean)	# trees	147	214	403	435
	# trees success	88	97	149	89
	# leaves total	1017	1329	1130	486
	Avg leaves	11.6	13.7	7.6	5.5
	Max leaves	100	220	115	38
	Avg levels	3.8	3.1	3.0	3.1
	Min levels	2	2	2	2
	Max levels	11	4	7	6
wals(gower,mode)	# trees	147	214	403	435
	# trees success	97	109	166	101
	# leaves total	2273	2502	2319	1019
	Avg leaves	23.4	23.0	14.0	10.1
	Max leaves	337	371	314	128
	Avg levels	4.3	4.0	3.3	3.7
	Min levels	2	4	2	2
	Max levels	15	4	7	12
wals(euclidean,mode)	# trees	147	214	403	435
	# trees success	97	109	166	101
	# leaves total	2273	2502	2319	1019
	Avg leaves	23.4	23.0	14.0	10.1
	Max leaves	337	371	314	128
	Avg levels	4.3	4.0	3.3	3.7
	Min levels	2	4	2	2
	Max levels	15	4	7	12

Parameter	**Measure**	**E**	**W**	**A**	**G**
autotyp	# trees	147	214	403	435
	# trees success	83	91	125	86
	# leaves total	858	884	703	452
	Avg leaves	10.3	9.7	5.6	5.3
	Max leaves	101	102	105	28
	Avg levels	3.8	3.1	3.0	3.3
	Min levels	2	2	2	2
	Max levels	10	4	7	7
mg2015(wals)	# trees	0	214	0	0
	# trees success	0	109	0	0
	# leaves total	0	2502	0	0
	Avg leaves	0.0	23.0	0.0	0.0
	Max leaves	0	371	0	0
	Avg levels	0.0	4.0	0.0	0.0
	Min levels	0	4	0	0
	Max levels	0	4	0	0
mg2015(ethnologue)	# trees	147	0	0	0
	# trees success	134	0	0	0
	# leaves total	7479	0	0	0
	Avg leaves	55.8	0.0	0.0	0.0
	Max leaves	1545	0	0	0
	Avg levels	4.9	0.0	0.0	0.0
	Min levels	3	0	0	0
	Max levels	16	0	0	0
mg2015(glottolog)	# trees	0	0	0	435
	# trees success	0	0	0	274
	# leaves total	0	0	0	15611
	Avg leaves	0.0	0.0	0.0	57.0
	Max leaves	0	0	0	3254
	Avg levels	0.0	0.0	0.0	5.4
	Min levels	0	0	0	3
	Max levels	0	0	0	20
mg2015(autotyp)	# trees	0	0	403	0
	# trees success	0	0	174	0
	# leaves total	0	0	2697	0
	Avg leaves	0.0	0.0	15.5	0.0
	Max leaves	0	0	340	0
	Avg levels	0.0	0.0	3.9	0.0
	Min levels	0	0	3	0
	Max levels	0	0	7	0

Table 8: Summaries for nnls.

Table 9 gives the summaries for the *ga* method with the various distance matrices.

Parameter	**Measure**	**E**	**W**	**A**	**G**
asjp16	# trees	147	214	403	435
	# trees success	100	88	141	123
	# leaves total	3846	1999	2091	2000
	Avg leaves	38.5	22.7	14.8	16.3
	Max leaves	789	297	306	430
	Avg levels	4.4	3.4	3.3	4.0
	Min levels	2	2	2	2
	Max levels	15	4	7	15
geo	# trees	147	214	403	435
	# trees success	132	107	169	193
	# leaves total	7184	2481	2619	4605
	Avg leaves	54.4	23.2	15.5	23.9
	Max leaves	1510	370	337	830
	Avg levels	4.7	3.9	3.8	4.1
	Min levels	2	2	2	2
	Max levels	15	4	7	17
wals(gower)	# trees	147	214	403	435
	# trees success	80	78	126	71
	# leaves total	998	1290	1065	447
	Avg leaves	12.5	16.5	8.5	6.3
	Max leaves	100	220	115	38
	Avg levels	3.9	3.0	3.1	3.4
	Min levels	2	2	2	2
	Max levels	11	4	7	6
wals(euclidean)	# trees	147	214	403	435
	# trees success	80	78	126	71
	# leaves total	998	1290	1065	447
	Avg leaves	12.5	16.5	8.5	6.3
	Max leaves	100	220	115	38
	Avg levels	3.9	3.0	3.1	3.4
	Min levels	2	2	2	2
	Max levels	11	4	7	6
wals(gower,mode)	# trees	147	214	403	435
	# trees success	97	109	162	99
	# leaves total	2273	2502	2299	1012
	Avg leaves	23.4	23.0	14.2	10.2
	Max leaves	337	371	314	128
	Avg levels	4.3	4.0	3.4	3.7
	Min levels	2	4	2	2
	Max levels	15	4	7	12
wals(euclidean,mode)	# trees	147	214	403	435
	# trees success	97	109	162	99
	# leaves total	2273	2502	2299	1012
	Avg leaves	23.4	23.0	14.2	10.2
	Max leaves	337	371	314	128
	Avg levels	4.3	4.0	3.4	3.7
	Min levels	2	4	2	2
	Max levels	15	4	7	12

Parameter	**Measure**	**E**	**W**	**A**	**G**
autotyp	# trees	147	214	403	435
	# trees success	73	71	100	68
	# leaves total	835	832	646	412
	Avg leaves	11.4	11.7	6.5	6.1
	Max leaves	101	102	105	28
	Avg levels	4.0	3.1	3.0	3.5
	Min levels	2	2	2	2
	Max levels	10	4	7	7
mg2015(wals)	# trees	0	214	0	0
	# trees success	0	109	0	0
	# leaves total	0	2502	0	0
	Avg leaves	0.0	23.0	0.0	0.0
	Max leaves	0	371	0	0
	Avg levels	0.0	4.0	0.0	0.0
	Min levels	0	4	0	0
	Max levels	0	4	0	0
mg2015(ethnologue)	# trees	147	0	0	0
	# trees success	134	0	0	0
	# leaves total	7479	0	0	0
	Avg leaves	55.8	0.0	0.0	0.0
	Max leaves	1545	0	0	0
	Avg levels	4.9	0.0	0.0	0.0
	Min levels	3	0	0	0
	Max levels	16	0	0	0
mg2015(glottolog)	# trees	0	0	0	435
	# trees success	0	0	0	274
	# leaves total	0	0	0	15611
	Avg leaves	0.0	0.0	0.0	57.0
	Max leaves	0	0	0	3254
	Avg levels	0.0	0.0	0.0	5.4
	Min levels	0	0	0	3
	Max levels	0	0	0	20
mg2015(autotyp)	# trees	0	0	403	0
	# trees success	0	0	174	0
	# leaves total	0	0	2697	0
	Avg leaves	0.0	0.0	15.5	0.0
	Max leaves	0	0	340	0
	Avg levels	0.0	0.0	3.9	0.0
	Min levels	0	0	3	0
	Max levels	0	0	7	0

Table 9: Summaries for ga.