01 Randomizing Language Features in Tamgwa

by **armytag**

When and how to roll the dice

There are times when the language creation process reaches a creative block and progress becomes frustratingly slow. This might happen for a number of reasons, such as indecision at the early stage of a language or exhaustion in a later stage. In situations like this the language creator may look for methods to push past the mental obstacle, including randomly choosing some features. This article outlines how the randomization approach was used with **Tamgwa** to quickly generate a whole array of language features from the very beginning.

What is randomization?

Randomization, as used in this article, refers to a process in which a set of possibilities are narrowed down in a random manner. Examples of this might be creating a list of choices and rolling some dice to select from it, or writing options on a deck of flash cards to draw from. Digital tools can often help with randomization, especially for conditional or rule-based features like syllable and word generation which depend on a language's phonotactics. Artificial Intelligence tools like ChatGPT could also be considered randomization tools under this definition, but they were not used in the randomization process of **Tamgwa** outlined in this article so they have been ignored here; AI tools are certainly not needed for randomization, as there are a variety of simpler techniques available.

Randomization offers several benefits when it comes to language creation. It works best as a tool in a language creator's toolbelt, augmenting their creation process without dominating it. This can happen in several ways. First, randomization can help a language creator generate content like phonemes, words, grammar structures, *etc*. This benefit should be obvious to anyone who winces remembering the half-finished dictionary they haven't touched in months. Second, randomization can counter a conlanger's linguistic biases. Few of us are familiar with more than a handful of languages, and that limited exposure can likewise limit the form our conlangs take. If you can't help but make Englishy conlangs, chance can steer you elsewhere. Lastly, randomization can help overcome the indecision we sometimes feel in the face of endless linguistic possibility. There are so many ways that human language conveys ideas, and sometimes it can be almost impossible to wade through the myriad options to settle on the **one** this language will use.

Laying the foundation

To exemplify the randomization process, I will walk through the method I used for starting **Tamgwa** with almost entirely randomized features. To reiterate the definition above, randomization here will always involve first identifying a set of possible options, then randomly selecting from that set. For creating a list of possibilities I relied heavily on the invaluable World Atlas of Langauge Structures (WALS).[1] WALS has a collection of chapters and associated maps showing the distribution of a variety of language features, from phonology to morphology to syntax and more. Each chapter lists the various realizations of that language feature along with the number of languages using each variation. These numbers give an idea of how widespread each variation is and allow us to weight the randomization in a more naturalistic way. In other words, we can use the chapters in WALS to randomize a set of language features that mimics the tendencies of natural languages.

A typical randomization procedure can been seen in how noun plurality in **Tamgwa** was randomized using WALS chapters 33 & 34. Chapter 33 covers how plurality is encoded, such as using affixes like English's *-s*. The WALS authors list the following possibilities along with the number of languages which employ each option.

Language Feature	Language Count	Randomizer Range
Plural prefix	126	1 - 126
Plural suffix	513	127 - 639
Plural stem change	6	640 - 645
Plural tone	4	646 - 649
Plural by complete stem reduplication	8	650 - 657
Morphological, but no method primary	60	658 - 717
Plural word	170	718 - 887
Plural clitic	81	888 - 968
No plural	98	969 - 1066

Table 01.1: WALS Chapter 33: Coding of Nominal Plurality

I have added a Randomizer Range column, which simply shows how a random number should map back to the feature options.[2] In this case a number can be randomly generated between 1 and 1066 (inclusive) to see which range it falls within. For **Tamgwa** the random number was 814, so **Tamgwa** will encode plurality with an independant word rather than some kind of morphological inflection. The same process can be used with Chapter 34, *Occurrence of Nominal Plurality* to randomize when the plural word should be used. The random number for this feature was ultimately 118 which corresponds to the option ”Plural in all nouns, always optional”, thus all nouns in **Tamgwa** can be optionally marked for plurality by adding the plural word somewhere in the sentence. It is worth noting that sometimes there can be conflict between randomized features, such as if chapter 34 had been randomized to ”No nominal plural” despite the fact that **Tamgwa** already has a plural word based on the feature in chapter 33. In that case one could either exclude the conflicting option beforehand, or re-randomize if it is selected.

[1] https://wals.info

[2] To create the Randomizer Ranges yourself, simply take the high end of the previous range, add 1 to it to get the low end of the new range, and add the number of languages to it to get the high end of the new range

When first creating **Tamgwa** I used the process above to randomize features for most of the WALS chapters. This provided a wide grammatical basis upon which to build the rest of the language. Some of the features naturally contradicted one another, as one would expect from a completely random result, but not all of those contradictions proved to be intractable. The next section describes an interesting feature that developed in order to resolve one such conflict.

Personal disagreement

The randomized features of **Tamgwa** are (surprisingly) mostly consistent, giving shape to an analytic language with very little inflectional morphology. There are still some inconsistencies however, the most glaring of which was between the features for chapters 26 and 102. Chapter 26, *Prefixing vs. Suffixing in Inflectional Morphology*, uses a so-called *affix index* to measure how much a language uses affixes. The authors list all the features which contribute to calculating the *affix index* in the chapter's description, but what is relevant here is that affixes marking a verb's agent (A) add 2 to the *affix index* and affixes marking its patient (P) add 1. If the total *affix index* is 2 or less then the language is categorized under ”little to no inflectional morphology”, which is the case for **Tamgwa**. However, the randomized result for chapter 102, *Verbal Person Marking*, specifies that verbs in **Tamgwa** must have ”person marking of both A and P arguments” which would seem to necessitate an affixing index of at least 3 and thus contradict the value generated for chapter 26.

Although this contradiction seems to require one or both of the features to be re-randomized, there are some details in the WALS definitions that can actually help resolve the contradiction. In chapter 26, the authors clarify that person-marking clitics that can also attach to non-verbs are not included as part of the *affix index* calculation. On the other hand, chapter 102 does count clitics in its analysis as long as the clitics can sometimes attach to the verb. Therefore, if at least one of the person markers is a clitic that sometimes attaches to verbs and sometimes to non-verbs, then the *affix index* can be kept to 2 or less while still marking the verb for both A and P.

An real-world example of a language with verbal person marking that uses affixes and clitics together is the Sorani dialect of Kurdish. According to the analysis by Gharib and Pye[3], one of the verb arguments is marked by an affix on the verb and the other argument is marked by a clitic on either the verb or a preceding adverb. Whether the clitic is marking the agent (A) or the patient (P) of the action depends on whether it is using the Perfective or Imperfective aspect, as seen in the example below.

(1) **dat** **benim** Sorani Kurdish

da	=t	beni	-m
IPFV	=OBL.2SG	*see*.IPFV	-1SG

“I see you.”

(2) **benemeet**

beni	=m	-eet
see.PFV	=OBL.1SG	-2SG

“I saw you.”

[3] Gharib, H., & Pye, C. (2023) The clitic status of person markers in Sorani Kurdish. Kansas Working Papers in Linguistics, 39, 57-65. https://doi.org/10.17161/1808.27692

(3) **benetm**
bene =t -m
see.PFV =OBL.2SG -1SG

"You saw me." (Gharib & Pye 2023)

In Sorani Kurdish the oblique clitic is used to mark the patient of imperfective verbs and the agent of perfective verbs. **Tamgwa** can use a simpler system, where the clitic marks the same verbal argument and attaches to the first verb/adverb in the verb phrase regardless of aspect. Thus for our purposes, it is sufficient to see that examples (1) and (3) show the OBL.2SG clitic attaching to the aspect marker if it is present, and the verb if it is not.

Producing Pronouns

Based on the person-marking system outlined above, a full set of pronouns, affixes, and clitics can be generated. **Tamgwa** pronouns were randomized to use indepent words with person-number stems and a clusivity distinction in 1PL pronouns. These will have nominative and accusative forms to match the separate A and P person-markers, as well as a form to indicate possession; the forms will be called nominative, accusative, and genetive forms respectively (although there is no case system for general nouns). Altogether this entails 7 person-number combinations and 5 forms (3 cases, 1 affix, and 1 clitic form) for a total of 35 distinct "words".

There are plenty of ways to generate a list of words and syllables, including digital tools such as Vulgarlang[4], but for **Tamgwa** I have created a Python script that is tailored to its phonology and phonotactics. This script was used to generate 35 words that were randomly assigned to the different pronoun forms needed, as shown in the table below.

	NOM	Affix	ACC	Clitic	GEN
1SG	ŋ̊a	ki	gʲa	ŋ̊a	m̥a
2SG	lam	man	ba	bə	pu
3SG	lə	na	bʷa	ŋi	pə
1PL.INC	ga	fə	m̥uɬ	pix	rə
1PL.EXC	pʷa	təŋ	ɬa	lə	pu
2PL	biɬ	tu	bʲa	bəs	ŋ̊aŋ
3PL	xa	ti	pʷax	ŋar	bʷa

Table 01.2: Random pronoun table

Although there are some similar (even identical) words in the table, there is no immediate pattern. These words were manually rearranged after the randomization to add more consistency. For example, all of the words beginning with **ŋ̊** or **ŋ** were moved into the same row, or words with closed syllables were moved to the GEN column. The final rearranged table for **Tamgwa** is printed below, with obligatory plural marker **tʲan**[5] added to NOM and ACC forms of plural pronouns.

[4] https://www.vulgarlang.com

[5] *N.B.* **tʲan** is an optional plural marker for all nouns, but is obligatory for plural pronouns. The feature was randomly generated for WALS chapter 35.

	NOM	Affix	ACC	Clitic	GEN
1SG	ŋ̊a	-ŋi	ŋ̊a	=ŋar	ŋ̊aŋ
2SG	fə	-pə	pʷax	=pʷa	pix
3SG	bəs	-ɬa	m̥uɬ	=m̥a	man
1PL.INC	tu tʲan	-ti	rə tʲan	=na	təŋ
1PL.EXC	xa tʲan	-ki	gʲa tʲan	=ga	bʲa
2PL	bʷa tʲan	-bə	bʷa tʲan	=ba	biɬ
3PL	pu tʲan	-lə	pu tʲan	=lə	lam

Table 01.3: Rearranged pronoun table

Revealing Results

In addition to these pronominal forms, a set of preverbal tense-aspects words are required for the clitics to attach to in order to fully illustrate **Tamgwa**'s new person-marking syntax. Once again pulling from real-world examples, **Tamgwa** will mirror the set of aspect distinctions described in Timothy Ajani's 2001 paper on Yoruba.[6] According to this analysis, we will need 4 aspect words marking Realis Imperfective, Irrealis Perfective, Irrealis Imperfective, and Relational aspects. The Realis Perfective is unmarked and thus assumed when none of these separate aspect words occur. The aspect words can also occur together to create more nuanced aspectual distinctions, such as Realis Imperfective and Irrealis Imperfective denoting a Habitual aspect when used together. For more details on this system, please refer to Ajani's paper.

The same Python script used to generate the pronouns can be used to generate these 4 aspect words. However, because the number of words is relatively small, the script can be set to generate 2-3 times as many words and the final choices can be manually selected. The results are listed in the table below.

Aspect	Word
RLS.PFV	<none>
RLS.IPFV	kʷa
IRR.PFV	n̥a
IRR.IPFV	pi
REL	gʲa

Table 01.4: Aspect words in Tamgwa

These aspect words can now be used to test the person-marking clitic behaviour in some example sentences below. Remember that the clitics will attach to the end of the first word in the aspect-verb structure.

(4) **kʲamuŋipʷa** Tamgʷwa
kʲamu -ŋi =pʷa
see -1SG.NOM =2SG.ACC

"I saw you."

[6] Ajani, T. 2001. *Aspect in Yoruba and Nigerian English*

(5) **kʷapʷa kʲamuŋi**
kʷa =pʷa kʲamu -ŋi
RLS.IPFV =2SG.ACC *see* -1SG.NOM
"I see you."

(6) **pipʷa kʲamuŋi**
pi =pʷa kʲamu -ŋi
IRR.IPFV =2SG.ACC *see* -1SG.NOM
"I will see you."

(7) **pipʷa kʷa kʲamuŋi**
pi =pʷa kʷa kʲamu -ŋi
IRR.IPFV =2SG.ACC RLS.IPFV *see* -1SG.NOM
"I generally (habitually) see you."

Conclusion

Hopefully this article has given you some ideas about how to use randomization to augment your conlanging process. You can randomize everything from syntax to morphology to phonology to lexicon and more. This is particularly useful if you're struggling with choice paralysis or not sure where to start with certain part of your language, and you can even go as far as bootstrapping most of a language's features through random generation. You can also go about doing your randomization in whatever way you'd like. You write down some options on flash cards to randomly draw from. However you choose to use randomization, if at all, I wish you luck and can't wait to see what you come up with.