01 Randomizing Language Features in Tamgwa

by **armytag**

When and how to roll the dice

For a long time I have struggled to get very far with making a conlang, too bewildered by the vast sea of linguistic possibilities to actually choose how this one language in front of me should work. So for my newest language **Tamgwa**, I decided to take the process out of my head until it's take enough shape for me to work with. In other words, I randomly selected a lot of features of **Tamgwa** and started to fill in the gaps and smooth out the edges based on what was generated. There was no Artificial Intelligence involved in this process, just good old fashioned chance which lead to some interesting developments along the way. I hope this article will give you some ideas about how to use randomization in your own language creation process.

Why use randomization?

At first glance, randomization might seem at odds with the very spirit of conlanging. Whether we make languages to flesh out our fictional worlds, to explore the edges of linguistic and philosophical expression, to further the noble cause of international communication, or simply for the glee this process brings us, we never want what we do to be a random mess. There is almost always a *purpose* behind our endeavor, an idea pushing forward all the effort we inevitably dedicate to this pursuit. If all we cared about were the phonologies, inflection tables, and word lists, we might spend a few hours with Vulgarlang or ChatGPT seeing what combinations they spit out.[1] But that kind of activity is never going to sustain our interest for very long, certainly not for the months and years that many conlangers put into their creations. Clearly there is something to this process fuels the passion of a conlanger, which randomization along can never fulfill.

So is there any place for randomization in the conlanging process? The answer, of course, is yes! Randomization works best as a tool in a conlanger's toolbelt, augmenting their creation process without dominating it. This can happen in several ways. First, randomization

[1]If this actually is what you're interested in, I recommend looking at real languages anyway. The diversity in natural languages far outshines anything these tools can offer.

can help a conlanger generate content like phonemes, words, grammar structures, and more. 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.

All these situations and more are excellent times to use a bit of randomization. Just remember that randomization is only a tool to assist with the actual goal: creating a language.

Laying the foundation

Now that I've convinced you of the merits of a bit of randomization, you still need to know how to use it. To that end, I will walk through the process I used for starting **Tamgwa** with almost entirely randomized features. The simplest way to randomize a feature is to list a few variations of how that feature could look, and then toss a coin, roll some dice, or otherwise randomly choose which variation to move forward with. For example, we could take the list of the types of writing systems from Wikipedia (abjad, abugida, alphabetical, and logographic/syllabic) and roll a 4-sided die to decide which system to implement. The result might be at odds with the structure or phonology of the language, such as a syllabary for language with complex syllable structure, but that is precisely how randomization challenges our biases in language construction. We can always put the random result aside if it simply isn't working with the rest of the language.

For **Tamgwa**, rather than using Wikipedia I instead used the invaluable World Atlas of Language Structures (WALS).[2] 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 various realizations of the language feature along with the number of languages using that 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 an array of language features in a way that mimics the tendencies of natural languages.

To illustrate this process, we'll randomize how **Tamgwa** encodes noun plurality using WALS chapters 33 & 34. Chapter 33 covers how plurality is encoded, such as using affixes like English's *-s*. The authors list the following possibilities and number of languages which employ that option.

[2]https://wals.info

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 the Randomizer Range column, which simply shows how our random number should map back to the feature options.[3] Now all we have to do is generate a number between 1 and 1066 (inclusive) and see which range it falls within. For **Tamgwa** I generated the number 814, so **Tamgwa** will encode plurality with an independant word rather than some kind of morphological inflection. We can use the same process on Chapter 34: Occurrence of Nominal Plurality to randomize *when* this plural word should be used. Hopefully we don't randomly select the option ”No nominal plural”, since we've already established that **Tamgwa** will use a plural word; we could either not include this option or re-randomize if we get it. Cutting to the end, I ultimately generated the random number 118 for this feature which corresponds to the option ”Plural in all nouns, always optional”. Thus in **Tamgwa** all nouns can be optionally marked for plurality by adding a plural word somewhere in the sentence.

For **Tamgwa** I used this process to randomize features for most of the WALS chapters. This provides a wide grammatical basis upon which to build the rest of the language. I've listed some of the randomized features below. Of course some contradictions are likely to arise from these completely random features, which can be seen in this list as well. These contradictions are not always obstacles however, and they can sometimes lead to developing interesting new features to resolve them. I will go into an example of this next.

- 1
- 2

Personal disagreement

The randomized features of Tamgwa are (surprisingly) mostly consistent, outlining an analytic language with very little inflectional morphology. There are some still some inconsistencies however, the most glaring of which is between the features for chapters 26 and 102. Verbs in Tamgwa must have ”person marking of both A and P arguments” according to the chapter 102 feature. Yet in order for Tamgwa to have ”little to no inflectional morphology”, the **affix index** must be at most 2. According to the description in chapter 26, affixes marking A add 2 to the affix index and affixes marking P add an additional 1, so it would seem that Tamgwa must necessarily have an affixing index of at least 3 which would fully contradict the value generated for this feature.

Complete contradictions like this could be easily overcome by simply rerandomizing some or all of the features. In this case however, there are some details in the WALS definitions

[3]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

that can 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 and both of these features can be satisfied.

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[4], one of the verb arguments is marked by an affix on the verb and the other person 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) Sorani Kurdish

dat		**benim**	
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)

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** uses a simpler system, where the clitic 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 not.

Producing Pronouns

Based on the person-marking system outlined above, a full set of pronouns, affixes, and clitics can be generated. It was previously determined that **Tamgwa** will need indepent pronoun words with person-number stems and a clusivity distinction in 1PL pronouns. It will have nominative and accusative forms for the pronouns to match the A and P person-markers, and will also have a form to indicate possession; these will be called nominative, accusative, and genetive forms (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, such as the online tool Vulgarlang[5], but Tamgwa uses a Python script that is tailored to its phonology and

[4]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

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

phonotactics. In this case the script generated 35 words that were randomly assigned to the different pronoun forms needed, which can be seen in the table below.

	NOM	Affix	ACC	Clitic	GEN
1SG	hnja	ki	gya	hnja	hma
2SG	lam	man	ba	be	pu
3SG	le	na	bwa	nji	pe
1PL.INC	ga	fe	hmuhl	pix	re
1PL.EXC	pwa	tenj	hla	le	pu
2PL	bihl	tu	bya	bes	hnjanj
3PL	xa	ti	pwax	njar	bwa

Table 01.2: Random pronoun table

Although there are some similar (even identical) words in the table, there is no immediate pattern. To add more consistency after the randomization, the words can be manually rearrange. For example, all of the words beginning with **hnj** or **nj** could be moved into the same row, or words with closed syllables could be moved to the GEN column. The final rearranged table for Tamgwa is printed below, with obligatory plural marker **tyan**[6] added to NOM and ACC forms of plural pronouns.

	NOM	Affix	ACC	Clitic	GEN
1SG	hnja	-nji	hnja	=njar	hnjanj
2SG	fe	-pe	pwax	=pwa	pix
3SG	bes	-hla	hmuhl	=hma	man
1PL.INC	tu tyan	-ti	re tyan	=na	tenj
1PL.EXC	xa tyan	-ki	gya tyan	=ga	bya
2PL	bwa tyan	-be	bwa tyan	=ba	bihl
3PL	pu tyan	-le	pu tyan	=le	lam

Table 01.3: Rearranged pronoun table

Revealing Results

To fully illustrate Tamgwa's new person-marking syntax there needs to be a set of preverbal tense-aspects words for the clitics to attach to. Once again pulling from real-world examples, **Tamgwa** will mirror the same set of aspect words as described in Timothy Ajani's 2001 paper on Yoruba.[7] According to this analysis, we will need 4 aspect words marking Realis Imperfective, Irrealis Perfective, and Irrealis Imperfective, and Relational aspects. The Realis Perfective is unmarked and thus assumed when none of these explicit aspect words occur. The aspect words can co-occur to create more nuanced aspectual distinctions, such as Realis Imperfective and Irrealis Imperfective together denoting a Habitual aspect. For more details on this system, please refer to Ajani's paper.

The same Python script 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.

[6] *N.B.* **tyan** is an optional plural marker for all nouns, but is only obligatory for plural pronouns. The feature was randomly generated for Chapter 35 of WALS.

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

Aspect	Word
RLS.PFV	**<none>**
RLS.IPFV	**kwa**
IRR.PFV	**hna**
IRR.IPFV	**pi**
REL	**gya**

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) **kyamunjipwa** Tamgwa

kyamu	-nji	=pwa
see	-1SG.NOM	=2SG.ACC

"I saw you."

(5)

kwapwa		**kyamunji**	
kwa	=pwa	kyamu	-nji
RLS.IPFV	=2SG.ACC	*see*	-1SG.NOM

"I see you."

(6)

pipwa		**kyamunji**	
pi	=pwa	kyamu	-nji
IRR.IPFV	=2SG.ACC	*see*	-1SG.NOM

"I will see you."

(7)

pipwa		**kwa**	**kyamunji**	
pi	=pwa	kwa	kyamu	-nji
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.