

SPEAM User Manual: Purpose, Description, and User Instructions

Klaus Hofmann & Nikolaus Ritt, University of Vienna

(funded by the **Jubilee Fund of the City of Vienna**, Grant No. JF_2021-03_SPEAM)

SPEAM Version: 1.0 Beta (2023/10/13)

Version of the Manual: 1.0 (2023/10/13)

Contents:

Introduction and Rationale	1
Structure	8
Running the simulation	8

Introduction and Rationale

SPEAM is an agent-based simulation programme written in R. It allows researchers to pursue the question whether the distribution and the development of lexical stress patterns in a language may be explained as a function of eurhythmic preferences/constraints at the phrase/utterance level.

The basic idea is this: words with lexically assigned stress patterns combine to form utterances. These utterances are sensitive to preferences for speech to be rhythmical, and the sequences of stressed and unstressed syllables that arise when words combine may satisfy these preferences to a larger or to a lesser extent. When they satisfy them well, the lexical stress patterns carried by the words combined in this way are reinforced or entrenched. When they fail to satisfy them, they are destabilized and the probability that the words may adopt an alternative stress pattern increases. Thus, the lexicon winds up with a distribution of stress patterns whose combined expression in utterances satisfy rhythmic preferences sufficiently well more often than not, on average. This means that rhythmic preferences that apply in utterances are (at least part of) the reason why the lexical stress patterns of words are as they are.

This hypothesis differs in important ways from the established practice of accounting for word stress in terms of generalizations relating it to the syllabic structure of words as well as their syntactic class membership. The idea behind such approaches is that children acquiring a language abduce these generalizations from the evidence they are exposed to and learn to apply them when assigning word stress in their phonological grammars. It predicts that languages should have learnable and consistent stress rule systems, possibly with a relatively small set of exceptions. In contrast, the hypothesis that SPEAM is designed to investigate is that learners learn how often stress patterns help to produce rhythmically preferable utterance sequences and retain those patterns that, when combined in phrases, produce the preferable sequences more often than not. It does not rule out the possibility that words of the same syllabic structure and the same syntactic category should wind up with different stress patterns, reflecting the probability of these patterns to produce rhythmically well-formed utterances when combined with one another. Instead, it has the potential of predicting under what circumstances such stress pattern diversity is likely to establish itself in a language.

What makes that hypothesis interesting but at the same time difficult to assess is that word stress patterns cannot be assumed to reflect the effect of rhythmically based preferences directly. Instead, it needs to be taken into account that the impact of these preferences is mediated by the rhythmic contexts in which words come to be placed in when they combine with one another in actual utterances. When two words occur near each other, both of them affect the rhythmicity of the arising sequence, and how they do depends on how well their stress patterns fit together.

Consider, for example, the widely recognized preference for stressed and unstressed syllables to alternate. In a phrase like *se'vere de 'pression*, the iambic pattern of *se'vere* works well, but in *se'vere 'headache* it produces a clash with the first syllable of *'headache*. Since the ways in which words are combined are diverse, it is not straightforward to compute how their interactions with one another under constraints on rhythmicity should affect their stress patterns. In order to even only see what the hypothesis would predict, one does not only need know what types of rhythm one assumes to be preferred, and how strongly, but one also needs to know in what combinations words will occur, and how often. Only then can one try to calculate what types of stress patterns should - under the specific conditions - come to be stabilized in the lexicon and in what proportions, and only then can one begin to check if the predicted correlations between the relative frequencies of phrase types on the one hand, and

the relative frequencies of stress patterns in the lexicon, match those that can be observed in actual human languages. The purpose of SPEAM is to help with that task.

SPEAM is an agent based simulation, which models a population of word types (i.e. a lexicon) and the evolution of their stress patterns under rhythmically grounded constraints on the phrases they build. The lexicon it simulates consists of monosyllabic, disyllabic, and trisyllabic words, which can be nouns, adjectives, or verbs. The 'agents' in the simulation are the polysyllabic words, because only polysyllabic words have a choice regarding their stress pattern. The simulation allows them to choose any of the logically possible stress patterns they can take: i.e. disyllables may be stressed on the first or the last syllable, while trisyllables can be stressed on any of their three syllables. Also, each syllable may be either light or heavy (allowing four combinations for disyllables, and eight for trisyllables). The proportions of nouns, adjectives, and verbs, as well as the initial proportions of stress patterns and syllable weights, can be set separately for each run, so that different linguistic scenarios may be simulated.

In each round of the simulation, two polysyllabic words (i.e. two agents) with their specific stress patterns are chosen (with a probability that reflects the frequency of their type) to combine with one another and to form a phrase. In that phrase the two words may either follow one another immediately, or may have monosyllabic items between them. As pointed out, these monosyllabic items do not represent agents themselves, because, having only one syllable, they have no choice as far as their stress patterns are concerned. Thus, their role in the simulation is reduced to representing the diversity of contexts in which polysyllabic 'agents' may come to co-occur. Between each two polysyllabic agents, SPEAM allows up to two monosyllables, each of which may either carry stress (i.e. simulate a major class word), or not (i.e. simulate a function word). Just as the initial composition of the lexicon, (i.e. the proportion of disyllabic or trisyllabic nouns, verbs, or adjectives with different types of stress patterns), also the relative probability of zero, one, or two monosyllables to occur between polysyllabic agents can be set separately for each run, and each part-of-speech (so that one take into account, for example, that adjectives may be more likely than verbs to occur immediately before nouns, because noun phrases following verbs may often begin with an unstressed determiner).

Once a phrase is formed, its rhythmic quality evaluated and the participating word types receive feedback in terms of effects on their relative 'fitness'. In SPEAM, the 'fitness' of a word with a specific stress pattern is expressed as number between zero and one, which represents the

relative frequency of the type in the lexicon and determines its likelihood of being used (i.e. of being selected in the simulation). The fitness of a word gets increased (by a determinable amount) whenever it is selected for 'use'. The rhythmic evaluation of the phrase it helps to build then adjusts that 'reward' by subtracting a certain percentage for any deviation from rhythmic perfection. Thus, the relative fitness (or proportional frequency) of word types and their stress patterns changes as the simulation unfolds.

Crucially, SPEAM makes it possible to specify a variety of different assumptions about what constitutes optimal or suboptimal rhythmic sequences, and thereby allows one to investigate what kind of stress pattern distributions these assumptions predict for the type of language one decides to model. More specifically, SPEAM allows one to determine the assumed impact of two primary and two secondary factors.

The first primary factor whose impact SPEAM allows one to model is the widely recognized preference for stressed and unstressed syllables to alternate with one another strictly (sometimes referred to as a preference for feet to be binary, c.f. the OT constrain `FOOTBIN`). Thus, any deviation from strict alternation (i.e. any sequence of stressed syllables (i.e. any 'clash') - or unstressed ones (i.e. any 'lapse') subtracts from the well-formedness of a phrase, and reduces the fitness of the involved stress patterns. The second major factor SPEAM takes into account, is the preference for stressed syllables to be heavy and for unstressed ones to be light (often referred to as the Weight-to-Stress principle, or 'WSP'). Basically, the fitness of polysyllabic items in a phrase incurs a reduction for any stressed syllable that is light, and for any unstressed one that is heavy. The relative penalty a phrase (and the polysyllabic agents in it) incurs for clashes and lapses on the one hand, and violations of the weight-to-stress principle on the other can be set in the simulation (even though in the present version of SPEAM, it is not possible to eliminate the effects of both constraints altogether at the same time. Instead, for the time being, one can manipulate only their relative effect - for more information, see below).

The first secondary factor that SPEAM incorporates reflects the potential impact of a preference for prominence peaks to occur at regular time intervals (sometimes referred to as a preference for isochronous feet). Since prominence peaks usually fall between the onset and the nucleus of a syllable, a stress clash between two syllables may count as less severe if the first syllable is heavy than if it is light, because a heavy rhyme will increase the time span between the two neighbouring stress peaks. By the same rationale, a lapse may count as relatively less severe, if

the syllables separating two stress peaks are light rather than heavy. - The other secondary factor takes the possibility into account that sequences of three stressed or unstressed syllables (i.e. double clashes or double lapses) may be rhythmically ‘repaired’ by promoting or demoting the one in the middle. Thus, in the famous line *Shall I compare thee to a summer’s day*, the sequence of three unstressed function words thee to a is ‘repaired’ by promoting the *thee* to a stress peak, yielding the perfectly iambic line *Shall 'I com 'pare thee 'to a 'summer's 'day*. Likewise, in the line *When I have seen by Time’s fell hand defaced*, the double clash in can be repaired by demoting *fell*, once again yielding perfectly iambic *When 'I have 'seen by 'Time's fell 'hand de 'faced*. SPEAM allows one to model the assumed impact of both of these secondary factors on the rhythmicality of phrases, and thereby on the effects it has on the fitness of the stress patterns involved in it.

Finally, SPEAM also allows one to determine how the feedback incurred from the rhythmic quality of a phrase is distributed between the two polysyllables involved in building the phrase. This possibility is built into the simulation because one cannot be certain - a priori - that it will be distributed equally, and there reasons to suspect that it is not. In English, for example, potential stress clashes in phrases such as *Prin 'cess 'Mary*, or *Ho 'tel 'Cali 'fornia*, are typically ‘repaired’ by shifting the stress in the first word backwards, yielding *'Princess 'Mary*, And *'Ho tel 'Cali 'fornia*, respectively.

Thus, SPEAM allows one to set (and thereby to control for) two different sets of independent variables that may affect the fitness (or the relative frequency) of word stress patterns in the lexicon of a language. On the one hand, these are the factors that determine the likelihood of polysyllabic items to co-occur in different types of phrasal constellations. They reflect the composition of the lexicon (i.e. the number of mono- and polysyllabic nouns, verbs and adjectives, and the number of unstressed function words), as well as any assumptions one wants to make about syntactically conditioned probabilities of their co-occurrence. On the other hand, SPEAM allows one to incorporate one’s estimates concerning the potential effects of a variety of constraints grounded in rhythmical preferences.

Once one has set the initial parameters in a way that strikes one as plausible or theoretically interesting, one can also decide the number of rounds, or ‘generations’, for which one would like the simulation to run, as well the number of times one wants it to run. Then SPEAM runs.

How does SPEAM report the results of a run? Basically, it produces a set of huge tables (the number depends on how many runs one has ordered) in plain text format (labelled ‘1.txt’ through ‘n.txt’), where columns represent all of the 96 possible types of part-of-speech and stress-pattern combinations and where rows represent the proportions of the lexicon they constitute after each round of the simulation. The number of rows (=rounds) depends on one’s settings. Table (1) below represents a small section of such a table for the sake of illustration.

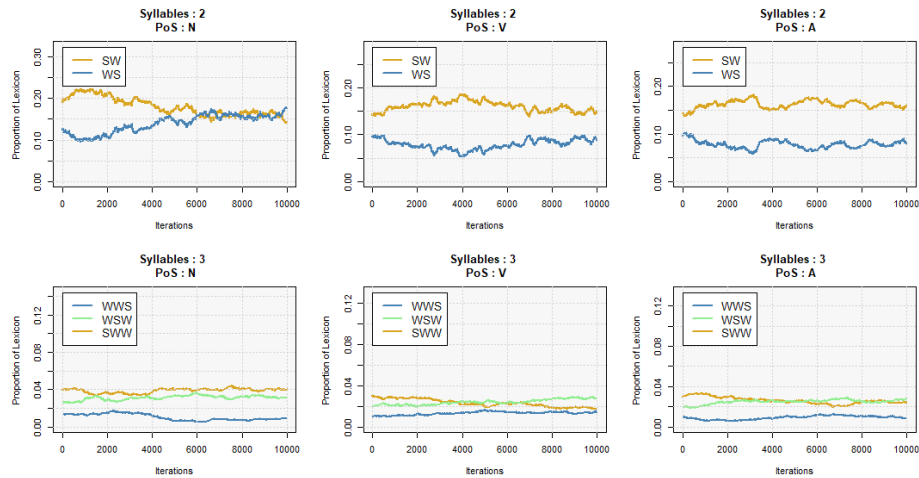
Table (1)

Word Type:	1	2	3	4	...	96
Round:	N_SW_LL	N_WS_LL	N_SW_HL	N_WS_HL	...	V_WWS_HHL
1	0.000388779	0.01773511	0.0584343	0.029005326	...	0.001759077
2	6.846E-05	0.000528813	0.02710081	0.000461802		7.14792E-05
2	0.03788945	9.29024E-05	0.19209249	0.020800744		0.01283002
4	0.00041266	0.000210095	0.12105268	0.007286984		0.175919849

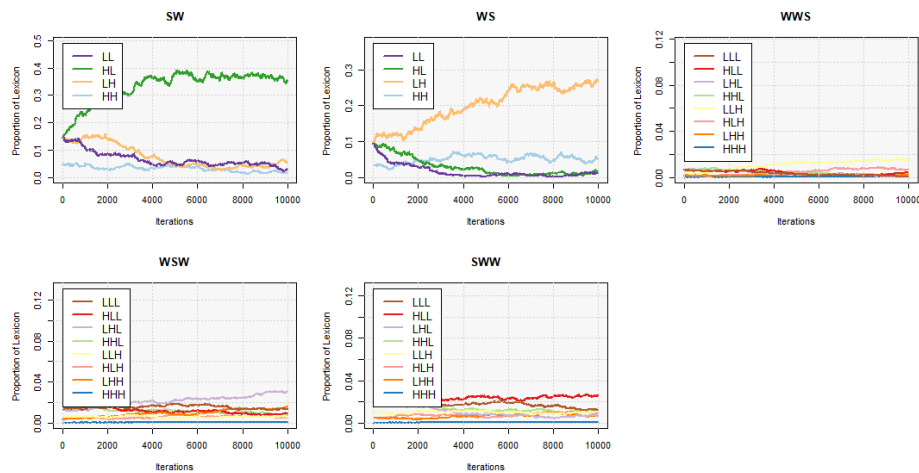
The number of times SPEAM should run the simulation can also be set. When SPEAM is asked to run more than once, it produces a corresponding number of output files, which are likely to differ from one another, because SPEAM also incorporates an element of randomness. In cases of multiple runs, SPEAM also produces an output file named ‘average.txt’, which averages over the outputs of all individual runs. While it may be useful to look at that average in cases where individual runs produce only slightly different outputs, it can be utterly misleading when they don’t. This means that it is necessary to inspect individual outputs in order to assess how seriously the average can be taken.

From the information contained in the output tables it is possible to derive trajectories representing the relative frequencies of all stress pattern types among polysyllabic nouns, verbs, and adjectives as they are likely to evolve under the initial parameters settings that one has chosen. Since these are difficult to interpret when they are represented as huge lists of numbers, SPEAM calculates these trajectories automatically and (b) produces a number of plots representing them, such as the ones in (1) and (2) below:

(1) Distribution of stress patterns among di- and trisyllabic nouns, verbs and adjectives:



(2) Distribution of syllable weights among stress pattern types:



An alternative way of producing visualisations is by means of Excel-Book templates that are provided as addenda to SPEAM. In order to use them, all one needs to do is copy and paste the contents of any of the result files (i.e., '1.txt', ... 'n.txt', 'or 'average.txt') into the sheet labelled 'Import' that the Excel-Book templates contain, pointing to the upper left corner of the sheet. The rest happens automatically.

Thus, SPEAM can show - in the form of fairly comprehensible graphic representations - what the hypothesis that the distribution of word stress patterns in a lexicon is determined by constraints on the rhythmality of utterances would actually predict, given one's specific







assumptions about these constraints, and given language specific probabilities of different (syntactically conditioned) types of word sequences to occur.

The following, more technical parts of this manual, explain the basic design of the simulation, as well as the way in which initial parameters can be set for different runs.

Structure

The programme consists of several sub-scripts which refer to one another. The individual scripts reflect the division into several functional elements of the programme (parameter input, payoff table generation, simulation, output generation and saving, graph generation) and primarily serve convenience of coding. For running the simulation, the individual sub-scripts are immaterial.

In its present version SPEAM involves the following subscripts. The one that needs to be opened when wanting to run SPEAM is the one labelled “SPEAM_input_console.R”. How parameters can be set in it, is explained below.

 SPEAM_functions.R	R File
 SPEAM_input_arrange.R	R File
 SPEAM_input_console.R	R File
 SPEAM_model.R	R File
 SPEAM_plot.R	R File
 SPEAM_preparations.R	R File

(Note: the only other file one might want to open is the one labelled “SPEAM_plot.R”. By default it is set to plot the average outcome of a number of runs. If you want to plot individual runs, you need to search for “average” and replace it with the appropriate number, i.e., “1” though “n”, depending.)

Running the simulation

Currently (Sept. 2023), the simulation does not feature a graphical user interface (GUI). This means that the user has to enter various parameters directly into the R script before the programme can be executed. However, this only involves manipulation of the sub-script called “SPEAM_input_console.txt”. The remaining sub-scripts need not - and should not - be modified.

In the sub-script “SPEAM_input_console.txt”, various input and interaction parameters can be modified to investigate their effect on stress pattern evolution according to the rhythmic context hypothesis as described above. The individual parameters are described below:

1) Set working directory

setwd(“”):

Enter path to directory where all subscripts and relevant input files (see below) are located. By default, the working directory is set to the location of the source file of “SPEAM_input_console.txt”.

2) Rhythmic contexts

The distributions of rhythmic contexts, which in the model act as a selective pressure for the evolution of lexical stress patterns, can be provided in various forms (i.e. *a-c* below), differing with regard to the granularity of the available input information. For example, if an external file is provided (cf. options *a-b* below), the distribution of parts-of-speech in the contexts may be specified independently, while if the proportions of context types are entered manually (cf. option *c* below), the parts-of-speech will be divided up equally across context types.

a) *ctxt.data.path:*

Fill with path to provide a set of coded concordances (i.e. corpus concordances annotated for rhythmic prominence; for exact format, see example file “*ctxt.data.input.txt*”). The relevant context distributions are calculated automatically from this input set. Leave empty string (= “”) to select input options b) or c).

b) *ctxt.frq.path:*

Fill with filename to provide the distributions of contexts in a tabled format (i.e. context types with FRQ of occurrence; for exact format, see example file “*ctxt.frq.input.txt*”). - Leave string empty (= “”) to select input options a) or c).

c) *ctxt_1 ... ctxt_7:*

Each line stands for one of 7 possible rhythmic context types. The value provided for each represents its share in the total distribution of context types.

Fill each context type with a value {0, 1}, such that the sum over all context types $\text{sum}(\text{ctxt}_1, \dots, \text{ctxt}_7)$ is exactly 1. If individual context types are left empty, the

context types have the occurrence probability of 0.0, which means that they do not exist in the simulation.

3) Lexicon (Agents).

The distributions of lexical types, which in the model act as agents selecting from among a number of possible stress patterns, may be provided in various forms (i.e. *a-b* below), differing in the amount of granularity of input information provided. For example, if a path is selected (option *a*), parts-of-speech proportions can be specified independently for each syllable count, while if proportions of nouns, verbs, disyllables etc. are specified manually (option *b*), this is not possible.

a) *lex.path*:

Fill with path (and filename) to provide a lexicon with all relevant attributes filled (e.g. as modeled on corpus data). Leave empty to select input option *b*).

b) Enter values for lexical type attributes (PoS, SyllNo, StrPat, Weight)

- pN: Proportion of nouns in the lexicon.
- pV: Proportion of verbs in the lexicon. Proportion of adjectives is automatically calculated as the difference between 1 and the sum over the proportions of nouns and verbs (i.e. $1 - \text{sum}(\text{pN}, \text{pV})$).
- p2S: Proportion of disyllables in the lexicon. Proportion of trisyllables (p3S) is automatically calculated as the difference between 1 and the proportion of disyllables (i.e. $1 - \text{p2S}$).
- pSW: Initial proportion of left-strong patterns ('trochaic stress') in the lexicon, which is subject to change in the simulation. Proportion of right-strong patterns ('iambic stress') is automatically calculated as the difference between 1 and the proportion of left-strong patterns (i.e. $1 - \text{pSW}$).
- pWWS: Initial proportion of right-strong trisyllabic patterns ('ultimate stress') in the lexicon, which is subject to change in the simulation.
- pWSW: Initial proportion of medially strong trisyllabic patterns ('penultimate stress') in the lexicon, which is subject to change in the simulation. Proportion of left-strong trisyllabic patterns ('ante-penultimate stress') is automatically calculated as the difference between 1 and the sum over the proportions of right-strong and medially strong patterns (i.e. $1 - \text{sum}(\text{pWWS}, \text{pWSW})$).

- p2Wght: Initial distribution of syllable weight types across disyllables (LL, HL, LH, HH), which is subject to change in the simulation. Sum over values needs to be ≤ 1 , where proportion of type HH is difference btw. 1 and sum over proportions of all other weight types (i.e. $HH = 1 - \text{sum}(LL, HL, LH)$).
- p3Wght: Initial distribution of syllable weight types across trisyllables (LLL, HLL, LHL, HHL, LLH, HLH, LHH, HHH), which is subject to change in the simulation. Sum over values needs to be ≤ 1 , where proportion of type HHH is difference btw. 1 and sum over proportions of all other weight types (i.e. $HHH = 1 - \text{sum}(LLL, HLL, LHL, HHL, LLH, HLH, LHH)$).

4) Interaction parameters

a) General parameters:

- *phi*: Basic increment ϕ , which is manipulated by all subsequent interaction parameters. If all other parameters are set to neutral, the fitness attribute in the agents participating in one round will maximally be augmented by ϕ . If ϕ is set too high, simulation will fluctuate too much for interactions to have discernible effect; if set too low, the effect may not become visible before a very large number of rounds have passed. Also, the value for ϕ and the values for noise/jit need to be co-adjusted, such that any randomness introduced by the latter does not outweigh the effect of the former. A value for $\phi \approx 0.015$ and jitter ≈ 0.0005 have proven functional. These values are set as defaults.
- *payoff-ratio*: The parameter *payoff-ratio* controls the relative impact of the payoff garnered through the evaluation of rhythmic interactions (including all interactions parameters above) vs. the payoff garnered through the evaluation of word-level weight attributes (weight-to-stress). Payoff from weight evaluation is highest when stress falls on heavy syllables while light syllables are unstressed. Weight-to-stress is evaluated separately from rhythmic well-formedness. If set to 0.5, both payoff mechanisms contribute equally to the final payoff garnered by an agent, i.e. the parameter is effectively neutralized. If set to 0, only word-level weight evaluation contributes to the final payoff and rhythmic well-formedness is disregarded. If set to 1, only rhythmic well-formedness contributes to the final payoff and word-level weight attributes are disregarded.

- *mult*: The parameter *mult* controls how relatively damaging more than one violation against weight-to-stress is compared to one violation.
- *noise*: The parameter *noise* is the first of two parameters that can be used to introduce some degree of randomness into the process. *Noise* does so by selecting an agent with random configuration of attributes, irrespective of the FRQ attribute of that agent type. This mirrors ‘random mutations’ in a population of agents. The parameter takes values ranging from 0 to 1. If set to 0, random agents are never selected, if set to 1, random agents are always selected.
- *jit.fct*: The parameter *jitter.factor* is the second of two parameters that can be used to introduce some degree of randomness into the process. *Jitter factor* does so by altering the FRQ attribute of all agent types after each round by a random factor. The parameter takes values ranging from 0 to 1. If set to 0, the FRQ attribute is left unaltered. The size of the impact of *jitter.factor* on the process critically depends on the size of *phi*, i.e. the basic increment. A value for $\varphi \approx 0.015$ and *jitter* ≈ 0.0005 have proven functional. These values are set as defaults.
- *nTimes*: The parameter *nTimes* determines the number of iterations a simulation is set to run through. If set too small, the simulation may not run long enough for the system to settle into a stable state.
- *para.sample*: The parameter *para.sample* determines the number of simulations that will be run. Depending on the values given to the input parameters (for contexts, interactions, ...) the simulations will run on the same parameter configuration or assume a different parameter configuration (within a pre-specified range of values) for each run of the simulation.

b) Interaction parameters:

Interaction parameters control the amount of payoff distributed to agents.

- *alpha*: The parameter α controls how much of any positive payoff generated will be distributed to each of the two agents participating in an interaction. If set to 0.5, the payoff is distributed equally between the agents, if set to 1, Player 1 receives all of the payoff, if set to 0, Player 2 receives all of the payoff.

- *rho*: The parameter ρ controls the degree to which repaired violations (e.g. SSS > SwS) are penalized relative to non-repairable violations (e.g. SWSSW). If set to $\frac{2}{3}$, payoff is distributed equally to agents participating in interactions with repaired or non-repairable violations. This reflects the fact that repairable violations always involve two violations at a time (e.g. SSS consists of two consecutive instances of SS). If set to 1, only agents in interactions with non-repairable violations receive payoff, if set to 0, only agents in interactions with repaired violations receive payoff.
- *lambda*: The parameter λ controls the degree to which lapses (i.e. WW) count as less (or possibly more) detrimental violations than clashes (i.e. SS). If set to 0.5, payoff is distributed equally to agents participating in interactions with clashes or lapses, i.e. the parameter is effectively neutralized. If set to 0, agents in interactions with lapses will receive the maximal allocable, while agents in interactions with clashes will not receive any payoff. If set to 1, agents in interactions with clashes will receive the maximal allocable payoff, while agents in interactions with lapses will not receive any payoff.
- *omega*: The parameter ω controls the degree to which syllable weight factors into the evaluation of clashes (i.e. SS). Weight-optimal clashes are clashes in which the first clashing syllable is heavy (i.e. S_HS) under the assumption that such a configuration promotes isochrony and is therefore rhythmically preferred. If set to 0.5, payoff is distributed equally to agents participating in interactions with weight-optimal and weight-suboptimal clashes, i.e. the parameter is effectively neutralized. If set to 0, agents in interactions with weight-optimal clashes will receive the maximal allocable payoff, while agents in interactions with weight-suboptimal clashes will not receive any payoff. If set to 1, agents in interactions with weight-suboptimal clashes will receive the maximal allocable payoff, while agents in interactions with weight-optimal clashes will not receive any payoff.
- *omicron*: The parameter o controls the degree to which syllable weight factors into the evaluation of lapses (i.e. WW). Weight-optimal lapses are lapses in which both lapsing syllables are light (i.e. W_LW_L) under the assumption that such a configuration promotes isochrony and is therefore

rhythmically preferred. If set to 0.5, payoff is distributed equally to agents participating in interactions with weight-optimal and weight-suboptimal lapses, i.e. the parameter is effectively neutralized. If set to 0, agents in interactions with weight-optimal lapses will receive the maximal allocable payoff, while agents in interactions with weight-suboptimal lapses will not receive any payoff. If set to 1, agents in interactions with weight-suboptimal lapses will receive the maximal allocable payoff, while agents in interactions with weight-optimal lapses will not receive any payoff.

- *gamma*: The parameter γ controls the degree to which syllable weight factors into the evaluation of repaired clashes, i.e. demotions ($SSS > SwS$). Weight-optimal demotions are demotions in which the demoted syllable is light (i.e. $Sw_L S$) under the assumption that heavy syllables carry intrinsic prominence and are therefore less likely to become demoted. If set to 0.5, payoff is distributed equally to agents participating in interactions with weight-optimal and weight-suboptimal demotions, i.e. the parameter is effectively neutralized. If set to 0, agents in interactions with weight-optimal demotions will receive the maximal allocable payoff, while agents in interactions with weight-suboptimal demotions will not receive any payoff. If set to 1, agents in interactions with weight-suboptimal demotions will receive the maximal allocable payoff, while agents in interactions with weight-optimal demotions will not receive any payoff.
- *epsilon*: The parameter ϵ controls the degree to which syllable weight factors into the evaluation of repaired lapses, i.e. promotions ($WWW > WsW$). Weight-optimal promotions are promotions in which the promoted syllable is heavy (i.e. $WS_H W$) under the assumption that heavy syllables carry intrinsic prominence and are therefore more likely to become promoted. If set to 0.5, payoff is distributed equally to agents participating in interactions with weight-optimal and weight-suboptimal promotions, i.e. the parameter is effectively neutralized. If set to 0, agents in interactions with weight-optimal promotions will receive the maximal allocable payoff, while agents in interactions with weight-suboptimal promotions will not receive any payoff. If set to 1, agents in interactions with

weight-suboptimal promotions will receive the maximal allocable payoff, while agents in interactions with weight-optimal promotions will not receive any payoff.