Tableaux for the larger model incorporating geminate inputs

Supplement to the article "Saltation and P-map", submitted to *Phonology*

1. Purpose

In the submitted text we give a simplified analysis focusing on the pattern of saltation. Here, we incorporate geminate inputs /appa/ and /abba/. For the latter, there are two possible outputs, [abba] and [aβa]. We first analyze each with a separate grammar (in order to obtain diagnostic tableaux); these were obtained using the Constraint Demotion algorithm in OTSoft 2.3.2. Then, we combine the two grammars into a single grammar that generates free variation (section 4), and lastly checked that this grammar performs correctly (section 5).

2. Grammar generating the output /abba/ → [aβa]

2.1 Rankings

Stratum	**Constraint**
Stratum #1	*MAP(b-β)
	*MAP(pp-β)
Stratum #2	*V[-CONT]V
	*MAP(pp-b)
Stratum #3	*β
	*MAP(pp-p)
	*MAP(pp-bb)
	*MAP(bb-β)
Stratum #4	*V[-voice]V
	*MAP(p-β)
	*MAP(bb-b)
Stratum #5	*MAP(p-b)
Stratum #6	*b

2.2 Tableaux

/apa/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ aβa						*					*		
aba				*!								*	*
apa				*!					*				

/aba/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ aba				*									*
apa				*					*!			*	
aβa	*!					*							

/aβa/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ aβa						*							
apa				*!					*		*		
aba	*!			*									*

/pa/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ pa													
ba												*!	*
βa						*!					*		

/ba/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ ba													*
pa												*!	
βa	*!					*							

/βa/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ pa											*		
βa						*!							
ba	*!												*

/appa/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ appa				*					*				
apa				*			*!		*				
abba				*	*!							*	*
aba			*!	*									*
aβa		*!				*							

/abba/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ aβa						*		*					
abba				*!									**
aba				*!						*			*

- The following are not "real" tableaux but are used to enforce the assumption that ranking will be P-map-compliant in the absence of contradictory data.

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ Winner												*	
Loser											*!		

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ Winner										*			
Loser			*!										

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ Winner					*								
Loser			*!										

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ Winner												*	
Loser			*!										

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ Winner							*						
Loser			*!										

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ Winner			*										
Loser		*!											

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ Winner							*						
Loser		*!											

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ Winner								*					
Loser		*!											

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*V[-cont]V	*Map(pp-bb)	*β	*Map(pp-p)	*Map(bb-β)	*V[-voice]V	*Map(bb-b)	*Map(p-β)	*Map(p-b)	*b
☞ Winner										*			
Loser								*!					

3. Grammar generating the output /abba/ → [abba]

Stratum	Constraints
Stratum #1	*MAP(b-β)
	*MAP(pp-β)
Stratum #2	*MAP(pp-b)
	*MAP(bb-β)
Stratum #3	*V[-CONT]V
	*MAP(pp-p)
	*MAP(pp-bb)
	*MAP(bb-b)
Stratum #4	*V[-voice]V
	*β
Stratum #5	*MAP(p-β)
Stratum #6	*MAP(p-b)
Stratum #7	*b

3.1 Tableaux

/apa/	*MAP(b-β)	*MAP(pp-β)	*MAP(pp-b)	*MAP(bb-β)	*V[-CONT]V	*MAP(pp-p)	*MAP(bb-b)	*MAP(pp-bb)	*β	*V[-voice]V	*MAP(p-β)	*MAP(p-b)	*b
☞ aβa									*		*		
aba					*!							*	*
apa					*!					*			

/aba/	*MAP(b-β)	*MAP(pp-β)	*MAP(pp-b)	*MAP(bb-β)	*V[-CONT]V	*MAP(pp-p)	*MAP(bb-b)	*MAP(pp-bb)	*β	*V[-voice]V	*MAP(p-β)	*MAP(p-b)	*b
☞ aba					*								*
apa					*					*!		*	
aβa	*!								*				

/aβa/	*MAP(b-β)	*MAP(pp-β)	*MAP(pp-b)	*MAP(bb-β)	*V[-CONT]V	*MAP(pp-p)	*MAP(bb-b)	*MAP(pp-bb)	*β	*V[-voice]V	*MAP(p-β)	*MAP(p-b)	*b
☞ aβa									*				
apa					*!					*	*		
aba	*!				*								*

/pa/	*MAP(b-β)	*MAP(pp-β)	*MAP(pp-b)	*MAP(bb-β)	*V[-CONT]V	*MAP(pp-p)	*MAP(bb-b)	*MAP(pp-bb)	*β	*V[-voice]V	*MAP(p-β)	*MAP(p-b)	*b
☞ pa													
ba												*!	*
βa									*!		*		

/ba/	*MAP(b-β)	*MAP(pp-β)	*MAP(pp-b)	*MAP(bb-β)	*V[-CONT]V	*MAP(pp-p)	*MAP(bb-b)	*MAP(pp-bb)	*β	*V[-voice]V	*MAP(p-β)	*MAP(p-b)	*b
☞ ba													*
pa												*!	
βa	*!								*				

/βa/	*MAP(b-β)	*MAP(pp-β)	*MAP(pp-b)	*MAP(bb-β)	*V[-CONT]V	*MAP(pp-p)	*MAP(bb-b)	*MAP(pp-bb)	*β	*V[-voice]V	*MAP(p-β)	*MAP(p-b)	*b
☞ pa											*		
βa									*!				
ba	*!												*

/appa/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*Map(bb-β)	*V[-cont]V	*Map(pp-p)	*Map(bb-b)	*Map(pp-bb)	*β	*V[-voice]V	*Map(p-β)	*Map(p-b)	*b
☞ appa					*					*			
abba					*			*!				*	*
apa					*	*!				*			
aba			*!		*								*
aβa		*!							*				

/abba/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*Map(bb-β)	*V[-cont]V	*Map(pp-p)	*Map(bb-b)	*Map(pp-bb)	*β	*V[-voice]V	*Map(p-β)	*Map(p-b)	*b
☞ abba					*								**
aba					*		*!						*
aβa				*!					*				

- The following are not "real" tableaux but are used to enforce the assumption that ranking will be P-map-compliant in the absence of contradictory data.

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*Map(bb-β)	*V[-cont]V	*Map(pp-p)	*Map(bb-b)	*Map(pp-bb)	*β	*V[-voice]V	*Map(p-β)	*Map(p-b)	*b
☞ Winner												*	
Loser											*!		

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*Map(bb-β)	*V[-cont]V	*Map(pp-p)	*Map(bb-b)	*Map(pp-bb)	*β	*V[-voice]V	*Map(p-β)	*Map(p-b)	*b
☞ Winner							*						
Loser			*!										

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*Map(bb-β)	*V[-cont]V	*Map(pp-p)	*Map(bb-b)	*Map(pp-bb)	*β	*V[-voice]V	*Map(p-β)	*Map(p-b)	*b
☞ Winner								*					
Loser			*!										

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*Map(bb-β)	*V[-cont]V	*Map(pp-p)	*Map(bb-b)	*Map(pp-bb)	*β	*V[-voice]V	*Map(p-β)	*Map(p-b)	*b
☞ Winner												*	
Loser			*!										

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*Map(bb-β)	*V[-cont]V	*Map(pp-p)	*Map(bb-b)	*Map(pp-bb)	*β	*V[-voice]V	*Map(p-β)	*Map(p-b)	*b
☞ Winner						*							
Loser			*!										

*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*Map(bb-β)	*V[-cont]V	*Map(pp-p)	*Map(bb-b)	*Map(pp-bb)	*β	*V[-voice]V	*Map(p-β)	*Map(p-b)	*b	*Map(b-β)
		*!											

*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*Map(bb-β)	*V[-cont]V	*Map(pp-p)	*Map(bb-b)	*Map(pp-bb)	*β	*V[-voice]V	*Map(p-β)	*Map(p-b)	*b	*Map(b-β)
						*							
		*!											

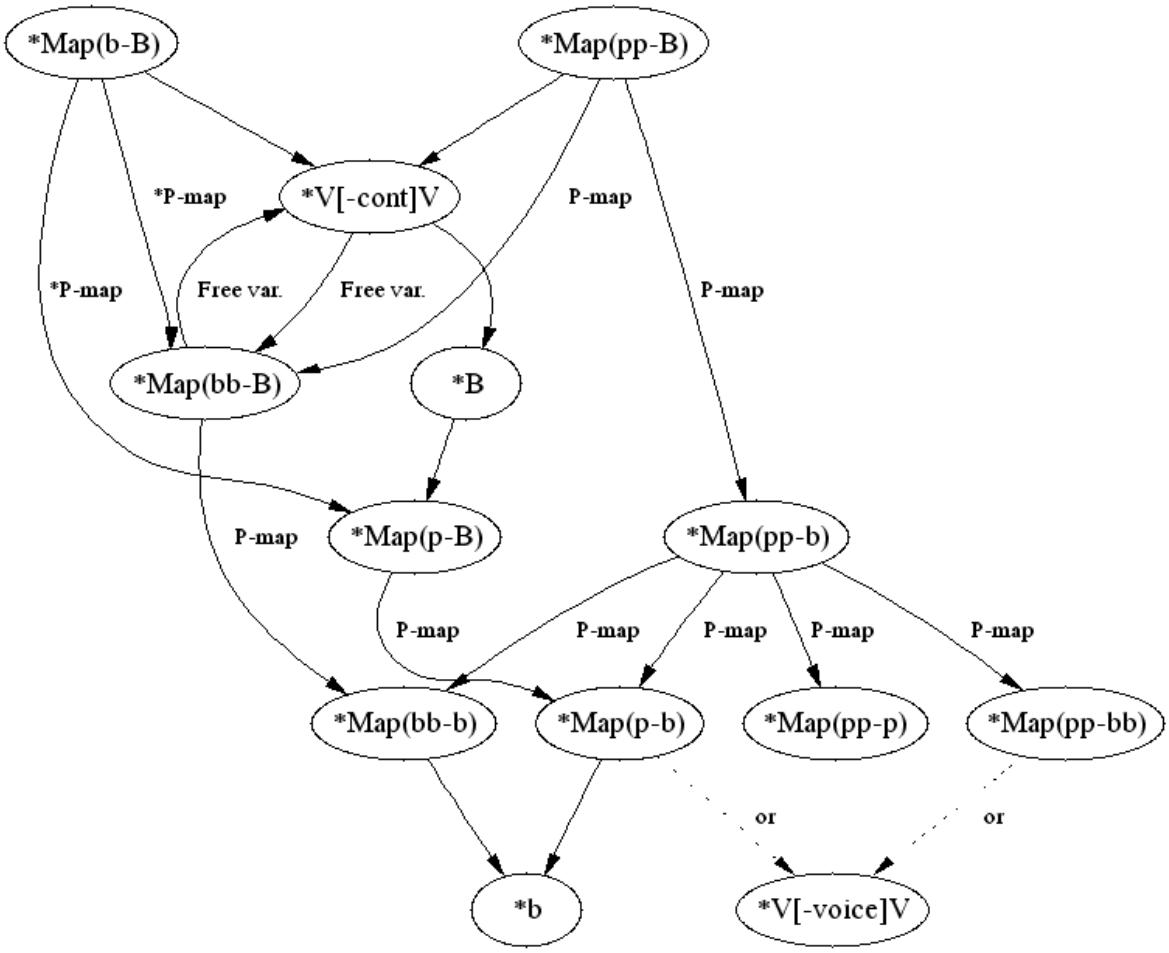
/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*Map(bb-β)	*V[-cont]V	*Map(pp-p)	*Map(bb-b)	*Map(pp-bb)	*β	*V[-voice]V	*Map(p-β)	*Map(p-b)	*b
☞ Winner				*									
Loser		*!											

/EnforcePMap/	*Map(b-β)	*Map(pp-β)	*Map(pp-b)	*Map(bb-β)	*V[-cont]V	*Map(pp-p)	*Map(bb-b)	*Map(pp-bb)	*β	*V[-voice]V	*Map(p-β)	*Map(p-b)	*b
☞ Winner							*						
Loser				*!									

[continues next page]

4. Full Hasse diagram incorporating both possible outputs for /abba/

We produced this by running the Fusional Reduction Algorithm (Brasoveanu and Prince 2011) on both of our input files, then collating the pairwise rankings obtained. There is only one case of *required* differential ranking, shown with the arrows labeled "Free var." This free ranking obtains the observed free variation between [abba] and [aβa] for /abba/.



5. Making sure the final grammar works

To verify that this grammar is adequate, we used rankings to create an analogous grammar in the framework of Stochastic OT (Boersma 1998, Boersma and Hayes 2001). Specifically, for any two constraints that are in a ranking relationship in the Hasse diagram, there is a ranking value difference of at least 20. Moreover, where there is free ranking (one single pair of constraints), the ranking values are the same. The following values satisfy these criteria:

Constraint	**Ranking value**
*V[-cont]V	100
*V[-voice]V	20
*B	80
*b	20
*Map(p-B)	60
*Map(p-b)	40
*Map(b-B)	120
*Map(pp-p)	40
*Map(pp-bb)	40
*Map(pp-b)	60
*Map(pp-B)	120
*Map(bb-b)	40
*Map(bb-B)	100

Using OTSoft 2.3.2, we tested this grammar by running it on 100,000 inputs for each candidate. For the free variation pair /abba/ → [abba, aβa], this generated 50,094 [abba] outputs and 49,906 [aβa] outputs. For all other inputs, the correct output was generated 100,000 times. We conclude that our grammar generates the intended output forms correctly.