

# HCB

January 15, 2020

**Title** Human Cultural Boundaries

**Version** 0.0.0

**Description** Creates seed populations with phoneme inventories that can grow, migrate, and create off-shoot populations. Phoneme inventories mutate when populations establish a new territory.

**License** What license it uses

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.1.1.9000

**Imports** mc2d, randomcoloR, uuid, numbers, philentropy, ade4

## R topics documented:

AddBeringStrait . . . . .	2
AddSegment . . . . .	3
AddShift . . . . .	3
AddSnakeBarriers . . . . .	4
Alternator . . . . .	4
BeringStraitPlot . . . . .	5
CardinalDirections . . . . .	5
DefineParameters . . . . .	6
Emigrate . . . . .	8
Extinction . . . . .	9
GeneratePhonemeProbabilities . . . . .	9
GenerateSeedLanguage . . . . .	9
GetAMutation . . . . .	10
GetASplitShiftJoinMut . . . . .	10
GetBering . . . . .	11
GetBeringCoords . . . . .	11
GetColorDistribution . . . . .	11
GetDist . . . . .	12
GetFactorDim . . . . .	12
GetGroups . . . . .	12
GetImmigrants . . . . .	13
GetRealPhonemeData . . . . .	13
GetTerritory . . . . .	14
GetXYCoords . . . . .	14
GroupBySeed . . . . .	14

HCBAAlternatorSimulation . . . . .	15
HCBSimulation . . . . .	15
HoritontalTransferRepeater . . . . .	15
HorizontalTransfer . . . . .	16
Initialize . . . . .	16
Lose . . . . .	17
MakeDistanceMap . . . . .	17
MakeLanguage . . . . .	17
MakePopulation . . . . .	18
Migration . . . . .	18
MigrationPlot . . . . .	19
NextStepDirections . . . . .	19
NextWave . . . . .	20
OneStepDirections . . . . .	20
PhonemeFrequencyPlots . . . . .	21
PhonemeMantel . . . . .	21
PhonemePopulationFrequencyPlots . . . . .	22
PopulationGrowth . . . . .	22
PopulationPlot . . . . .	23
RemoveHorizontalConnections . . . . .	23
RemoveVerticalConnections . . . . .	24
ResetPopulation . . . . .	24
SaveData . . . . .	24
ShiftDirections . . . . .	25
SnapshotPlot . . . . .	25
StepDirections . . . . .	25
UpdateExistingPhonemes . . . . .	26
UpdateStructuresMove . . . . .	26
UpdateStructuresRemove . . . . .	27
<b>Index</b>	<b>28</b>

---

AddBeringStrait	<i>Add Bering Strait</i>
-----------------	--------------------------

---

**Description**

Removes connections at the FirstStep stage of the Local structure to create "barriers" between cells. Bering Strait Barriers are designed to create structures similar to the Bering Strait entering North America, going through Central America, then opening up into South America.

**Usage**

```
AddBeringStrait(P, firstStep)
```

**Arguments**

- |           |  |
|-----------|--|
| P         | A list of parameters.                                |
| firstStep | The local directions created by OneStepDirections(). |

---

AddSegment

*Add Segment*


---

**Description**

Creates lines to show the Bering Straight borders.

**Usage**

AddSegment(P, a, b, top = FALSE)

**Arguments**

P	A list of parameters.
a	Start.
b	End.
Data	The Pre or Post output from an HBC simulation.

---

AddShift

*Add or Shift a Phoneme*


---

**Description**

Allows a language to either gain a new phoneme or shift an existing phoneme to better match another population's phoneme inventory.

**Usage**

AddShift(P, targetLanguage, languages, local, phonemeRelatedness, index)

**Arguments**

P	A list of parameters.
targetLanguage	The target language to be modified if possible.
languages	All languages
local	The local territories data structure.
index	The target territory whose language may change.
phonemeProbab	The probability of gaining each phoneme in the population.

---

AddSnakeBarriers	<i>Add Snake Barriers</i>
------------------	---------------------------

---

**Description**

Removes connections at the FirstStep stage of the Local structure to create "barriers" between cells. Snake Barriers are lines with length and spacing defined by the parameters. The barriers jut out from the east and west walls, alternating east, west, east, west. This creates a snaking, zig-zag pattern, hence the name.

**Usage**

```
AddSnakeBarriers(P, firstStep)
```

**Arguments**

P	A list of parameters.
firstStep	The local directions created by OneStepDirections().

---

Alternator	<i>Migration</i>
------------	------------------

---

**Description**

A function wrapper that alternates between migration and horizontal transfer.

**Usage**

```
Alternator(P, S, repeats)
```

**Arguments**

P	A list of parameters.
S	A list of data structures.
repeats	how many times to repeat migration.

---

BeringStraitPlot	<i>Bering Strait Plot</i>
------------------	---------------------------

---

**Description**

Creates a plot that shows the Bering strait boundaries.

**Usage**

```
BeringStraitPlot(P, Data, colors = NA)
```

**Arguments**

P	A list of parameters.
Data	The Pre or Post output from an HBC simulation.
colors	A vector of colors of length equal to the number of seed populations.

---

CardinalDirections	<i>Cardinal Directions</i>
--------------------	----------------------------

---

**Description**

Calculates the territory indicies of locations around a target territory (also used for phoneme relatedness in the same way).

**Usage**

```
CardinalDirections(target, R, start, round, South, North, East, West, SE,  
NE, SW, NW)
```

**Arguments**

target	The territory around which to get local territories.
R	The number of rows.
start	How much to offset numbers (for phoneme structures).
round	Whether to get a "round" set of territories (N, S, E, W only) for phonemes or a square set of territories (includes diagonals) for distance.
SE	Whether to get the southeastern territory.
NE	Whether to get the northeastern territory.
SW	Whether to get the southwestern territory.
NW	Whether to get the northwestern territory.
south	Whether to get the southern territory.
north	Whether to get the northern territory.
east	Whether to get the eastern territory.
west	Whether to get the western territory.

DefineParameters

*Define Parameters***Description**

Creates a parameter data structure for running simulations.

**Usage**

```
DefineParameters(Rows = 40, Cols = 50, ChanceExpand = 0.8,
  PopulationStartIndex = c(1, 2), NumPopulationPhonemes = rep(NA,
    length(PopulationStartIndex)), UsePopSize = TRUE,
  IndividualsStEmSuEM = c(1000, 10, 20, NA), MutationRate = 0.15,
  PhonemeDitribution = c(12, 24, 133), Consonants = 750,
  Vowels = 100, MinConsonant = 6, MinVowel = 6,
  PhonemeProbabilityType = "RealMimic", GrowthMax = 5, GrowthMin = 5,
  Barriers = FALSE, BarrierLength = 30, BarrierBreaks = 4,
  MutationTypeChance = rep(1/5, 5), HorizontalRate = 0.1,
  Bias = TRUE, Steps = 1, HorizontalLocal = TRUE,
  NumberRandomHorizontal = 8, UpRoot = TRUE, Death = TRUE,
  Bering = FALSE, BeringLength = 20, MigrationSimSteps = 300,
  StationaryMutationRate = 0.01, UpRootMutationRate = 0.01,
  HorizontalSimSteps = 400, Waves = FALSE, Seed = NA)
```

**Arguments**

Rows	The number of rows in the world matrix.
Cols	The number of columns in the world matrix.
ChanceExpand	The chance that a population will either move or send off a group of individuals to found a new population.
PopulationStartIndex	The position in the matrix where each seed population starts. The number of seed populations is defined by the number of starting indicies.
NumPopulationPhonemes	The number of phonemes in each starting population. If set to NA, this is decided by sampling from a distribution with min, mode, and made on the values from the PhonemeDistribution arguement.
UsePopSize	Whether to take into account the the population size (number of people) when making decisions about moving, immigrating, the mutation rate ,and phoneme loss/addition biases.
IndividualsStEmSuEM	Four related parameters: 1) The number of individuals a seed population starts with, 2) the minumum number of individuals required to make a founder party to settle a new territory, 3) the minumum number of individuals that must stay behind when a founder party is sent off, and 4) the maximum number of individuals allowed to be in one founder party.
MutationRate	The rate at which phonemes mutate. E.g., if MutationRate==0.1, each phoneme in a populatio <sup>n</sup> phoneme inventory has a 10% chance to mutate. Note that when usePopSize==TRUE, this is better conceptualized as teh maximum mutation rate (larger populations have lower mutation rates).

Consonants	The number of possible consonants in existence. Default based on real phoneme data.
Vowels	The number of possible vowels in existence. Default based on real phoneme data.
MinConsonant	The minimum number of consonants that can be in a population's phoneme inventory. Default based on real phoneme data.
MinVowel	The minimum number of vowels that can be in a population's phoneme inventory. Default based on real phoneme data.
PhonemeProbabilityType	The method by which phoneme probabilities are established. Can be Real (uses the real data verbatim, and requires the correct number of consonants and phonemes), RealMimic (uses the real data to generate a new distribution of probability similar to the real data, can be used with any number of phonemes), Equal (all phonemes are equally likely to be known), Frequency (based on how common phonemes are across populations in the simulation), or Random (randomly generated).
GrowthMax	When an integer, the max number of individuals added or lost from populations every time step. When a fraction, the percent that population size increases or decreases each timestep. Can be positive or negative. Also note that if this is an integer, but GrowthMin is not, this will be considered a percentage (i.e. 2 = 200%).
GrowthMin	When an integer, the min number of individuals added or lost from each population every time step. When a fraction, the percent that a population size increases or decreases each timestep. Can be positive or negative. Also note that if this is an integer, but GrowthMax is not, this will be considered a percentage (i.e. 2 = 200%).
Barriers	Whether to create "snake barriers" that limit the direction of migration in the matrix.
BarrierLength	The width of snake barriers.
BarrierBreaks	The height of the space between snake barriers.
MutationTypeChance	The chance that each mutation type occurs. 1) Add, 2) Lose, 3) Split, 4) Join, and 5) Shift.
HorizontalRate	The fraction of the population that attempts to modify its phoneme inventory every horizontal timestep.
Bias	Whether to randomly bias mutations towards either gains or losses when populations are small. Set to true based on previously published data.
Steps	The number of distance steps away from a target location that are considered "local." Includes all 8 cardinal and ordinal directions around a target, so the local area is always a rectangle around the target location.
HorizontalLocal	Whether horizontal transfer occurs between local populations or globally. Set to FALSE as a control, as global horizontal transfer should abolish local patterns.
NumberRandomHorizontal	The number of locations to compare when HorizontalLocal==FALSE. Should be 8 when Steps==1, 24 when Steps==2, 48 when Steps==3, etc.
UpRoot	Whether established populations can move (TRUE) or they remain in place for the entire simulation (FALSE).

Death	Whether populations can die out.
Bering	Whether to employ barriers that mimic the Bering Strait and Americas.
BeringLength	An integer of length 0 to 24 that defines how long the bering straight is
MigrationSimSteps	The number of time steps to run each wave of migration.
StationaryMutationRate	The rate at which phonemes evolve in stationary populations during migration.
UpRootMutationRate	The rate at which phonemes evolve in populations that move as a unit during migration.
HorizontalSimSteps	The number of time steps to spend on horizontal transfer.
Waves	Whether migration occurs in waves or all seed populations are added at the same time. If TRUE, there is one wave for each seed population.
Seed	Sets a seed for reproducibility if an integer instead of NA.
PhonemeDistribution	The 1) min, 2) mode, and 3) max number of phonemes a population can have when sampling for seed population sizes and when preventing languages from gaining or losing too many phonemes. Defaults based on real phoneme data.

---

Emigrate

---

Emigrate

---

## Description

Picks which populations migrate, whether the entire population migrates or a founder party is sent off, and where the population migrates to. Allows only one population to enter a territory. When multiple populations attempt to enter the same territory, one is randomly chosen to do so while the rest stay put.

## Usage

```
Emigrate(P, occupied, local, populations)
```

## Arguments

P	A list of parameters.
occupied	The territories with a population on them.
local	The local territories data structure.
populations	The data for all existing populations.



---

Extinction

*Extinction*


---

**Description**

Tests which populations will die based on population size and random chance.

**Usage**

Extinction(populations, occupied)

**Arguments**

populations	The data for all existing populations.
occupied	The territories with a population on them.

---

GeneratePhonemeProbabilities

*Generate Phoneme Probabilities*


---

**Description**

Generates a vector of the probability to know each phoneme.

**Usage**

GeneratePhonemeProbabilities(P)

**Arguments**

P	A list of parameters.
---	-----------------------

---

GenerateSeedLanguage

*Generate Seed Language*


---

**Description**

Creates a language for a seed population.

**Usage**

GenerateSeedLanguage(P, phonemeProbab, seedNum)

**Arguments**

P	A list of parameters.
phonemeProbab	The probability of gaining each phoneme in the population.
seedNum	Which population seed is having it's language generated.

---

GetAMutation

*Get A Mutation*


---

### Description

Returns a new Add, Loss, Split, Join, or Shift mutation. Biases the mutations towards gaining or losing syllables when the parameter Bias==TRUE.

### Usage

```
GetAMutation(P, phonemes, phonemeProbab, phonemeRelatedness, gain)
```

### Arguments

P	A list of parameters.
phonemes	The phonemes currently in the language.
phonemeProbab	The probability of gaining each phoneme in the population.
phonemeRelatedness	The phoneme relatedness list.
gain	Whether to bias the mutations towards gaining syllables (TRUE) or losing them (FALSE).

---

GetASplitShiftJoinMut    *Get A Split Shift Join Mutation*


---

### Description

Recursively calls itself until a phoneme is found that can be used to generate the mutation type of interest. Returns Null if no phoneme can mutate appropriately.

### Usage

```
GetASplitShiftJoinMut(phonemes, phonemeRelatedness, unusable = NULL,
  type)
```

### Arguments

phonemes	Phonemes that can be mutated.
phonemeRelatedness	The phoneme relatedness list.
unusable	Phonemes that cannot be used to obtain the correct type of mutation.
type	Which kind of mutation to create: Split, Join, or Shift.

---

GetBering	<i>Get Bering positions</i>
-----------	-----------------------------

---

**Description**

returns a list of the points for the berring strait

**Usage**

```
GetBering(P)
```

**Arguments**

P	A list of parameters.
---	-----------------------

---

GetBeringCoords	<i>Get Bering Strait Coordinates</i>
-----------------	--------------------------------------

---

**Description**

Returns the hardcoded locations of the Bering Strait boundaries.

**Usage**

```
GetBeringCoords(P)
```

---

GetColorDistribution	<i>Get Colors</i>
----------------------	-------------------

---

**Description**

Creates a color gradient.

**Usage**

```
GetColorDistribution(P, Data, i, colors = c("coral1", "coral4"))
```

**Arguments**

P	A list of parameters.
Data	The Pre or Post output from an HBC simulation.
colors	A vector of beginning and ending colors for a gradient.

---

GetDist

*Get Distance*


---

**Description**

Get the euclidian distance between two territories.

**Usage**

```
GetDist(P, point1, point2)
```

**Arguments**

P	A list of parameters.
point1	One territory location.
point2	Another territory location.

---

GetFactorDim

*Get Factor Dimentions*


---

**Description**

Given a number of consonants or vowels, creates a data structure that is as square as possible.

**Usage**

```
GetFactorDim(nPhonemes)
```

**Arguments**

nPhonemes	The number of Phonemes (vowels or consonants).
-----------	--

---

GetGroups

*Get Groups*


---

**Description**

Returns the territories descended from each seed. Includes detailed ancestry data. Only works when Uproot and Death are FALSE.

**Usage**

```
GetGroups(P, Data)
```

**Arguments**

P	A list of parameters.
Data	The Pre or Post output from an HBC simulation.

---

`GetImmigrants`*Get Immigrants*

---

**Description**

Tests which populations immigrate, removes those that did not migrate from the migraton data. Splits the data into populations that sent off founder parties and those that moved as a single population.

**Usage**

```
GetImmigrants(P, occupied, local, populations)
```

**Arguments**

P	A list of parameters.
occupied	The territories with a population on them.
local	The local territories data structure.
populations	The data for all exising populations.

---

`GetRealPhonemeData`*Get Real Phoneme Data*

---

**Description**

Uses the real Phoneme data from Creanza..... UPDATE THIS!!!! to determine the phoneme probabilities.

**Usage**

```
GetRealPhonemeData(nPhoneme, actual, vowel = FALSE)
```

**Arguments**

nPhoneme	The number of phonemes (vowels or consonants).
actual	Whether the data is Real (TRUE) or RealMimic (FALSE).
vowel	If true load the vowel data, otherwise load the cosonant data.

---

GetTerritory	<i>Get Territory</i>
--------------	----------------------

---

**Description**

Returns a territory a population can migrate to or NA if none are available.

**Usage**

```
GetTerritory(local, open)
```

**Arguments**

local	Territories that are within reach of the target territory.
open	Which territories can be migrated to (i.e. no other population currently resides there).

---

GetXYCoords	<i>Get XY Coordinates</i>
-------------	---------------------------

---

**Description**

Converts territory numbers into X,Y coordinates.

**Usage**

```
GetXYCoords(P, territories)
```

**Arguments**

P	A list of parameters.
territories	A vector of territory indices to convert into X,Y coordinates.

---

GroupBySeed	<i>Group By Seed</i>
-------------	----------------------

---

**Description**

Returns the territories descended from each seed. Works for all sims, but lacks ancestry data.

**Usage**

```
GroupBySeed(P, Data)
```

**Arguments**

P	A list of parameters.
Data	The Pre or Post output from an HBC simulation.

---

HCBAlternatorSimulation

*Human Cultural Boundaries Alternator Simulation*


---

**Description**

Runs a simulation where each time step alternates between migration and horizontal transfer.

**Usage**

HCBAlternatorSimulation(P)

**Arguments**

P                      A list of parameters.

---

HCSimulation

*Human Cultural Boundaries Simulation*


---

**Description**

Runs a simulation where migration and horizontal transfer are completely separate.

**Usage**

HCSimulation(P)

**Arguments**

P                      A list of parameters.

---

HoritontalTransferRepeater

*Horitontal Transfer Repeater*


---

**Description**

A wrapper for the horizontal transfer process. After migration, allow populations to exchnage phoneme information, losing or gaining syllables based on other populations in the simulation. Occurs HSims number of time steps.

**Usage**

HoritontalTransferRepeater(P, S, repeats)

**Arguments**

P                      A list of parameters.  
S                      A list of the data structures.  
repeats              how many times to repeat horizontal transfer

---

HorizontalTransfer	<i>Horizontal Transfer</i>
--------------------	----------------------------

---

### Description

A function wrapper that get the language to modify and allows the phoneme change to either add/shift or remove a phoneme if this can be done.

### Usage

```
HorizontalTransfer(P, languages, local, phonemeRelatedness, phonemeProbab,
index)
```

### Arguments

P	A list of parameters.
languages	All languages.
local	The local territories data structure.
phonemeRelatedness	The phoneme relatedness list.
phonemeProbab	The probability of gaining each phoneme in the population.
index	The target territory whose language may change.

---

Initialize	<i>Initialize</i>
------------	-------------------

---

### Description

The function wrapper that makes calls to create the population and phoneme data structures and then populates them with initial data.

### Usage

```
Initialize(P)
```

### Arguments

P	A list of parameters.
---	-----------------------



---

Lose

*Lose a Phoneme*


---

**Description**

Allows a language to either lose a phoneme to better match other populations.

**Usage**

Lose(P, targetLanguage, phonemeProbab)

**Arguments**

P                      A list of parameters.  
targetLanguage      The target language to be modified if possible.  
phonemeProbab      The probability of gaining each phoneme in the population.

---

MakeDistanceMap

*Make Distance Map*


---

**Description**

Creates a distance map based on the euclidian diatcnnes between territories.

**Usage**

MakeDistanceMap(P)

**Arguments**

P                      A list of parameters.

---

MakeLanguage

*Make Language*


---

**Description**

Copies and mutates the parent's langauge to create a new language for a founder party.

**Usage**

MakeLanguage(P, phonemeProbab, phonemeRelatedness, language, popSize, rate)

**Arguments**

P	A list of parameters.
phonemeProbab	The probability of gaining each phoneme in the population.
phonemeRelatedness	The phoneme relatedness list.
language	The parent language to mutate into a new language.
popSize	The number of individuals in the parent pouлтаion.
rate	The mutation rate.

---

MakePopulation	<i>Make Population</i>
----------------	------------------------

---

**Description**

Generates new population based on the parent population.

**Usage**

MakePopulation(P, population)

**Arguments**

P	A list of parameters.
population	The population data used to make a new population.

---

Migration	<i>Migration</i>
-----------	------------------

---

**Description**

Main simulation function. Allows populations to migrate, split, and die.

**Usage**

Migration(P, S, repeats)

**Arguments**

P	A list of parameters.
S	A list of data structures.
repeats	how many times to repeat migration.

---

MigrationPlot	<i>Migration Plot</i>
---------------	-----------------------

---

**Description**

Shows the expansion of populations from the seed population. Only works when Uproot and Death are FALSE.

**Usage**

```
MigrationPlot(P, Data, groups = NA, colors = NA)
```

**Arguments**

P	A list of parameters.
Data	The Pre or Post output from an HBC simulation.
groups	Group structure of which territories were descended from what population seed.
colors	A vector of colors of length equal to the number of seed populations.

---

NextStepDirections	<i>Next Step Directions</i>
--------------------	-----------------------------

---

**Description**

Expands the Steps list one more step out.

**Usage**

```
NextStepDirections(firstStep, currentStep, start = 0)
```

**Arguments**

firstStep	The original StepOne.
currentStep	StepOne in its current state.
start	How much to offset numbers (for phoneme structures).

---

NextWave	<i>Next Wave</i>
----------	------------------

---

**Description**

Adds the seed data for the next wave to the population and language dataframes.

**Usage**

```
NextWave(P, S, i)
```

**Arguments**

P	A list of parameters.
S	A list of data structures.
i	The number of the next wave.

---

OneStepDirections	<i>One Step Directions</i>
-------------------	----------------------------

---

**Description**

Creates the FirstStep data structure.

**Usage**

```
OneStepDirections(R, C, start = 0, round = FALSE)
```

**Arguments**

R	The number of rows.
C	The number of columns.
start	How much to offset numbers (for phoneme structures).
round	whether to make the spacing Round (Phonemes) or Square (Territories).

---

PhonemeFrequencyPlots    *Phoneme Frequency Plots*


---

**Description**

Shows how common each phonemes is in the simulation color coded by population.

**Usage**

```
PhonemeFrequencyPlots(P, Data, groups = NA, colors = NA)
```

**Arguments**

P	A list of parameters.
Data	The Pre or Post output from an HBC simulation.
groups	Group structure of which territories were descended from what population seed.
colors	A vector of colors of length equal to the number of seed populations.

---

PhonemeMantel            *Phoneme Mantel*


---

**Description**

Performs both a Hamming and Jaccard Mantel test.

**Usage**

```
PhonemeMantel(P, Data, repeats = 100)
```

**Arguments**

P	A list of parameters.
Data	The Pre or Post output from an HBC simulation.
repeats	How many times to repeat the analysis.

---

PhonemePopulationFrequencyPlots

*Phoneme Population Frequency Plots*


---

### Description

Shows how common each phonemes is in the simulation color coded by population.

### Usage

```
PhonemePopulationFrequencyPlots(P, Data, groups = NA, colors = NA,
sort = TRUE)
```

### Arguments

P	A list of parameters.
Data	The Pre or Post output from an HBC simulation.
groups	Group structure of which territories were descended from what population seed.
colors	A vector of colors of length equal to the number of seed populations.#' @param sort Whether to sort the data from most to least frequent phoneme.

---

PopulationGrowth

*Population Growth*


---

### Description

Adds new individuals to existing populations when population size is used in the simulation.

### Usage

```
PopulationGrowth(P, populationSizes, occupied)
```

### Arguments

P	A list of parameters.
populationSizes	The number of people live on each territory.
occupied	The indices of territories with people living on them.

---

PopulationPlot

*Population Plot*


---

**Description**

Population Plot

**Usage**

```
PopulationPlot(P, Data, groups = NA, colors = NA)
```

**Arguments**

P	A list of parameters.
groups	Group structure of which territories were descended from what population seed.
colors	A vector of colors of length equal to the number of seed populations.

---

RemoveHorizontalConnections

*Remove Horizontal Connections*


---

**Description**

Affects local territories below/South (and perhaps to the Southeast and Southwest) the target territory (index) and above/North (perhaps Northwest and Northeast) of index +1.

**Usage**

```
RemoveHorizontalConnections(R, index, firstStep, right = TRUE,  
  left = TRUE)
```

**Arguments**

R	The number of rows in the population matrix.
index	The target territory.
firstStep	The local directions created by OneStepDirections().
right	Whether to remove the right diagonal.
left	Whether to remove the left diagonal.

---

RemoveVerticalConnections

*Remove Vertical Connections*


---

### Description

Affects local territories right/East (and perhaps to the Northeast and Southeast) the target territory (index) and left/West (perhaps Northwest and Southwest) of index + R.

### Usage

```
RemoveVerticalConnections(R, index, firstStep, above = TRUE,
  below = TRUE)
```

### Arguments

R	The number of rows in the population matrix.
index	The target territory.
firstStep	The local directions created by OneStepDirections().
above	Whether to remove the upper diagonal.
below	Whether to remove the lower diagonal.

---

ResetPopulation

*Reset Population*


---

### Description

Resets phoeneme and population data in the original territory when an entire population moes to a new territory.

### Usage

```
ResetPopulation()
```

---

SaveData

*Save Data*


---

### Description

Saves just the language data from the simulation to .csv files.

### Usage

```
SaveData(Data, filename)
```



---

ShiftDirections	<i>Shift Directions</i>
-----------------	-------------------------

---

**Description**

Returns the relationships between phonemes with an offset of start.

**Usage**

```
ShiftDirections(nPhonemes, start = 0)
```

**Arguments**

nPhonemes	The number of phonemes.
start	Where to start number the phonemes (0 for consonants, number of consonants +1 for vowels).

---

SnapshotPlot	<i>Snapshot Plot</i>
--------------	----------------------

---

**Description**

Shows which territories are populated and from which seed they decended.

**Usage**

```
SnapshotPlot(P, Data, colors)
```

**Arguments**

P	A list of parameters.
Data	The Pre or Post output from an HBC simulation.
colors	A vector of colors, one for each seed.

---

StepDirections	<i>Step Directions</i>
----------------	------------------------

---

**Description**

A wrapper that calls StepOne(), add barriers if required, then expands StepOne as many steps as the Steps parameter calls for.

**Usage**

```
StepDirections(P)
```

**Arguments**

P	A list of parameters.
---	-----------------------

---

`UpdateExistingPhonemes`*Update Existing Phonemes*

---

**Description**

Change the language to incorporation new mutations.

**Usage**

```
UpdateExistingPhonemes(existingPhonemes, newMut, index)
```

**Arguments**

<code>newMut</code>	The new mutation generated by <code>GetAMutation()</code> .
<code>index</code>	Whether to enact changes to the language based on the first or section member of the mutation structure.
<code>ExistingPhonemes</code>	The Phonemes currently in the language.

---

`UpdateStructuresMove`    *Update Structures Move*

---

**Description**

Copies population data from one territory to another when the entire population migrates and then erases the original data.

**Usage**

```
UpdateStructuresMove(S, move, former)
```

**Arguments**

<code>S</code>	A list of data structures.
<code>move</code>	The indices of territories
<code>former</code>	The indices of territories

---

`UpdateStructuresRemove`*Update Structures Remove*

---

**Description**

Deletes population and language information for specified territories.

**Usage**

`UpdateStructuresRemove(S, remove)`

**Arguments**

S	A list of data structures.
Remove	The indices of territories whose data should be erased.

# Index

## \*Topic **Barriers**

- AddBeringStrait, [2](#)
- AddSnakeBarriers, [4](#)
- GetBering, [11](#)
- RemoveHorizontalConnections, [23](#)
- RemoveVerticalConnections, [24](#)

## \*Topic **Directions**

- CardinalDirections, [5](#)
- GetFactorDim, [12](#)
- NextStepDirections, [19](#)
- OneStepDirections, [20](#)
- ShiftDirections, [25](#)
- StepDirections, [25](#)

## \*Topic **Horizontal**

- AddShift, [3](#)
- HoritontalTransferRepeater, [15](#)
- HorizontalTransfer, [16](#)
- Lose, [17](#)

## \*Topic **Language**

- GetAMutation, [10](#)
- GetASplitShiftJoinMut, [10](#)
- MakeLanguage, [17](#)
- UpdateExistingPhonemes, [26](#)

## \*Topic **Phonemes**

- GeneratePhonemeProbabilities, [9](#)
- GenerateSeedLanguage, [9](#)
- GetRealPhonemeData, [13](#)

## \*Topic **Plotting**

- AddSegment, [3](#)
- BeringStraitPlot, [5](#)
- GetBeringCoords, [11](#)
- GetColorDistribution, [11](#)
- GetDist, [12](#)
- GetGroups, [12](#)
- GetXYCoords, [14](#)
- GroupBySeed, [14](#)
- MakeDistanceMap, [17](#)
- MigrationPlot, [19](#)
- PhonemeFrequencyPlots, [21](#)
- PhonemeMantel, [21](#)
- PhonemePopulationFrequencyPlots, [22](#)
- PopulationPlot, [23](#)

- SaveData, [24](#)

- SnapshotPlot, [25](#)

## \*Topic **Population\_Dynamics**

- Emigrate, [8](#)
- Extinction, [9](#)
- GetImmigrants, [13](#)
- GetTerritory, [14](#)
- MakePopulation, [18](#)
- PopulationGrowth, [22](#)
- ResetPopulation, [24](#)
- UpdateStructuresMove, [26](#)
- UpdateStructuresRemove, [27](#)

## \*Topic **SimParam**

- Alternator, [4](#)
- DefineParameters, [6](#)
- HCBAAlternatorSimulation, [15](#)
- HCBSimulation, [15](#)
- Migration, [18](#)
- NextWave, [20](#)

## \*Topic **Wrapper**

- Initialize, [16](#)
- StepDirections, [25](#)

- AddBeringStrait, [2](#)

- AddSegment, [3](#)

- AddShift, [3](#)

- AddSnakeBarriers, [4](#)

- Alternator, [4](#)

- BeringStraitPlot, [5](#)

- CardinalDirections, [5](#)

- DefineParameters, [6](#)

- Emigrate, [8](#)

- Extinction, [9](#)

- GeneratePhonemeProbabilities, [9](#)

- GenerateSeedLanguage, [9](#)

- GetAMutation, [10](#)

- GetASplitShiftJoinMut, [10](#)

- GetBering, [11](#)

- GetBeringCoords, [11](#)

- GetColorDistribution, [11](#)

GetDist, [12](#)  
GetFactorDim, [12](#)  
GetGroups, [12](#)  
GetImmigrants, [13](#)  
GetRealPhonemeData, [13](#)  
GetTerritory, [14](#)  
GetXYCoords, [14](#)  
GroupBySeed, [14](#)  
  
HCBAlternatorSimulation, [15](#)  
HCBSimulation, [15](#)  
HoritontalTransferRepeater, [15](#)  
HorizontalTransfer, [16](#)  
  
Initialize, [16](#)  
  
Lose, [17](#)  
  
MakeDistanceMap, [17](#)  
MakeLanguage, [17](#)  
MakePopulation, [18](#)  
Migration, [18](#)  
MigrationPlot, [19](#)  
  
NextStepDirections, [19](#)  
NextWave, [20](#)  
  
OneStepDirections, [20](#)  
  
PhonemeFrequencyPlots, [21](#)  
PhonemeMantel, [21](#)  
PhonemePopulationFrequencyPlots, [22](#)  
PopulationGrowth, [22](#)  
PopulationPlot, [23](#)  
  
RemoveHorizontalConnections, [23](#)  
RemoveVerticalConnections, [24](#)  
ResetPopulation, [24](#)  
  
SaveData, [24](#)  
ShiftDirections, [25](#)  
SnapshotPlot, [25](#)  
StepDirections, [25](#)  
  
UpdateExistingPhonemes, [26](#)  
UpdateStructuresMove, [26](#)  
UpdateStructuresRemove, [27](#)