Biosim_Project Documentation

Release 1.1

Marius Skaug Kristiansen, Kristian Frafjord

January 20, 2016

CONTENTS

This is a simulation of animals of two species with

- a modifiable island
- two species:
- Carnivores
- Herbivores
- customizable parameters for island biomes and animals

CHAPTER

ONE

ISLAND

1.1 The island module

class biosim.island.**Island**(*island_map*)

Class object: Island.

- <var> = Island(String of landscape types)
- <var>.build_map() must run before simulation starts

Parameters **island_map** – Map of island as string of "biomes"

aging()
Runs ageing method in each cell

build_map()
Builds island based on input string. Island biomes are placed in a two dimensional array

death()
Runs death function in each cell

feeding()
Runs animal level feeding method on each animal in each cell

grow()
Runs through growth cycle for each cell in <var>.island. Runs only if the cell i capable of growing food

individuals()
Returns the number of carnivores and herbivores in each cell as array with each cell containing a dict over herbivores and carnivores

loss_of_weight()
Runs weightloss method in each cell

migration()
Runs the migration process for all cells. Starts at first cell in shuffled list. Resets the has_moved attribute of the animals when all animals have moved.

one_year()
Simulates one year progression and returns array containing the numbers of each species in each cell.

procreation()
Runs animal level breeding method on each animal in each cell

shuffle_coordinates()
Makes a shuffled list of the coordinates for all the cells on map.

Returns Shuffled list of coordinates

surrounding_cells(*coordinate*)
Calculates surrounding cells that animals can migrate to. If the neighboring cell is either mountain or ocean it does not append the selected coordinates

Parameters **coordinate** – A given coordinate

Returns List of surrounding cells which animals can migrate to.

surrounding_cells_relative_food(*y*, *x*, *species*)
Makes nested lists for the coordinates of the surrounding cells and their amount of relative food.

Parameters

- **y** – y coordinate for the center cell
- **x** – x coordinate for the center cell
- **species** – String with the name of the specie to calculate relative

food for. :return: nested list with surrounding cell and their calculated relative food

CHAPTER

TWO

LANDSCAPE

2.1 The landscape module

class biosim.landscape.**Desert**(*carnivores=None*, *herbivores=None*)
Landscape subclass Desert Inhabitable for herbivores, but carnivores can feed on herbivores in desert

class biosim.landscape.**Jungle**(*carnivores=None*, *herbivores=None*)
Landscape subclass Jungle. Habitable and food is replenished to maximum level each year

grow_food()
Replenishes the amount of food in the jungle cell to f_max

class biosim.landscape.**Landscape**(*carnivores=None*, *herbivores=None*)
Superclass Landscape

Constructor for Landscape.

Parameters **carnivores** – Instances of carnivores as list of

"Carnivore()" instances :param herbivores: Instances of herbivores as list of "Herbivore()" instances

age_cycle()
Each animals age is incremented by one year

avg_age()
Returns the average age of population

Returns ("herbivores age", "carnivores age")

avg_fitness()
Method used for testing Returns the average fitness of the population

Returns ("herbivores fitness", "carnivores fitness")

breeding_cycle()
Starts the breeding cycle for both species in a single cell If breeding is successful, the method appends a new animal of the same species to the list of animals

static **calc_fitness**(*animals*)
Makes a sorted list for animal fitness of the input list of animal instances. Highest fitness first.

Parameters **animals** – List of animal instances

Returns Sorted list of animal instances with fitness values in a

tuple consisting of (<class instance>, "fitness value")

death_cycle()
Starts the death-function for each animal. Removes animals who are "dead" (Animal death method returns "True")

feeding_cycle()
Starts the feeding cycle for herbivores in a single cell. Highest fitness first.

herbivore_weight()
Updates available food in cell for carnivore based on the total weight of herbivores in cell

migration_cycle_carn(*_list*)
Starts the migration cycle for the carnivores in the cell.

Parameters **_list** – Nested list with possible coordinates the carnivores may

move to and the relative food for each cell.

Returns List of animals that are migrating and their new position

migration_cycle_herb(*_list*)
Starts the migration cycle for the herbivores in the cell.

Parameters **_list** – Nested list with possible coordinates the herbivores may

move to and the relative food for each cell. :return: List of animals that are migrating and their new position

number_of_individuals()
Returns number of individuals in cell.

Returns dictionary containing number

of carnivores and number of herbivores

relative_food_carn()
Calculates the amount of relative food in cell for carnivores.

Returns Amount of relative food for carnivores

relative_food_herb()
Calculates the amount of relative food in cell for herbivores.

Returns Amount of relative food for herbivore

classmethod **set_parameters**(*new_params*)
Updates parameters. Raises ValueError if values are invalid

Parameters **new_params** – New set of parameters as dictionary

weightloss_cycle()
Each animal loses weight according to formula; “eta” * “weight”

class biosim.landscape.**Mountain**(*carnivores=None*, *herbivores=None*)
Landscape subclass Mountain Impassable terrain for both species

class biosim.landscape.**Ocean**(*carnivores=None*, *herbivores=None*)
Landscape subclass Ocean Impassable terrain for both species

class biosim.landscape.**Savannah**(*carnivores=None*, *herbivores=None*)
Landscape subclass Savannah. Habitable, but food grows at a reduced rate.

grow_food()
Replenishes the amount of food in Savannah cell according to formula

CHAPTER

THREE

ANIMALS

3.1 The animals module

class biosim.animals.**Animal**(*weight=None*, *age=0*, *coordinates=(1, 1)*)

Superclass "Animal" for herbivores and carnivores

Parameters

- **weight** – Default None results in gaussian distribution of weight
- **age** – The starting age of animal
- **coordinates** – Starting coordinates of the animal

ageing()

Increment age by one per year

breeding(*individuals*)

Calculates if the animal will give birth based on animals present in cell, weight of animal and set parameters

Parameters **individuals** – number of individuals in cell

Returns Returns birth weight if it gives birth or None.

check_migrate()

Check if the animal wants to migrate based on set parameters

Returns True if animal will migrate

death()

Calculates if the animal dies or not based on fitness and set parameters

Returns True if the animal dies, False otherwise

migrate(*_list*)

Calculates if the herbivore will migrate and returns either the new coordinates or the current coordinates.

Parameters **_list** – Nested list of tuples with surrounding positions as first

element and relative food as second element. :return: New coordinates for the animal if it migrates or the old if it does not.

classmethod **set_parameters**(*new_params*)

Updates parameters. Raises ValueError if values are invalid

Parameters **new_params** – New set of parameters as dictionary

update_fitness()

Re-calculates the animal's fitness based on age and weight

weightloss()

Recalculates the animals weight according to "eta" and original weight

class biosim.animals.**Carnivore**(*weight=None*, *age=0*, *coordinates=(1*, *1)*)
Animal subclass Carnivore

feeding(*herbivores*)
Calculate if the carnivore will feed based on its own fitness and the fitness of the herbivore, and gain weight. Removes eaten herbivores.

Parameters **herbivores** – List of herbivores in cell

Returns Updated list of herbivores in cell after eating

class biosim.animals.**Herbivore**(*weight=None*, *age=0*, *coordinates=(1*, *1)*)
Animal subclass Herbivore.

feeding(*available_food*)
Herbivore feeding method. The animal will feed based on amount of available food in cell, and returns the result. If the amount left in cell is less than the animals "hunger", the animal will eat the remaining amount and returns 0

Parameters **available_food** – available food before eating

Returns new amount of food left after eating

CHAPTER

FOUR

SIMULATION

4.1 The simulation module

class biosim.simulation.**BioSim**(*island_map=None*, *ini_pop=None*, *seed=None*)
Class BioSim Main simulation class for biosim project

Constructor creates island from given map, and adds initial population to island.

All parameters needed to create animals are contained within ini_pop.

Parameters **island_map** – String containing letters representing each

type of landscape :param ini_pop: List of initial_populations. [pop1, pop2] :param seed: Seed for rng

add_population(*population*)
Adds given population to cells. Location stored in given population :param population: list of populations

animal()
Returns total number of animals per type :return anim: Dict of animals per type

animals_by_species()
Prints number of Animals per type on island

static heatmap(*island_results*)
Returns two nested lists; [row[cell]], one for herbivore population and one for carnivore population. :param island_results: Array containing population data for one year :return kart_herb, kart_carn: Map over populations

per_cell_animal_count()
Prints the total amount of animals in island

plot_update(*years*, *abscissa*, *ordinate*, *colour_herb*, *colour_carn*)
Updates the plot n_steps years :param years: Number of years to plot :param abscissa: Xrange :param ordinate: Yrange :param colour_herb: Colour of heatmap (default None) :param colour_carn: Colour of heatmap (default None)

simulate(*num_steps=100*, *vis_steps=1*, *img_steps=2000*, *abscissa=None*, *ordinate=20000*, *colour_herb=None*, *colour_carn=None*)
Runs simulation num_steps number of years

Parameters

- **num_steps** – Number of years to simulate
- **vis_steps** – Number of years between each time results are drawn
- **img_steps** – Number of years between each .png file created
- **abscissa** – Length of x-axis
- **ordinate** – Length of y-axis

- **colour_herb** – Colour for chart of herbivore population
- **colour_carn** – Colour for chart of herbivore population

total_number_of_animals()
Prints the total number of animals on island

years_simulated()
Prints the total number of years simulated

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

b