Appendix 1

List of functions in the Python Script

Module	Description	Script	Function
Filter 0. Birds	Filter using nor only SQL but also dynamic segmentation	pass_birdlist_ return_ numpyarray.py	pass_birdlist_ return_ numpyarray
Filter 1. Distance Time	Filter using nor only SQL but also dynamic segmentation	pass_twodistances_ return_twotimes_ tagIdentList_ numpyarray.py	pass_ twodistance_ return_time
Filter 2. Attributes	Filter using nor only SQL but also dynamic segmentation	filter_by_ attributes_ structured_ array.py	filter_by_ fields_where
Scatter plot & Histograms Graphics	Create graphics	scatter_histo.py	scatter_histo
Graphics by bird by season	Create graphics	barChartSeasBS.py	createBarChart BirdsSeason
Graphics by season by bird	Create graphics		createBarChart SeasonBirds
Mean Bird Speed and Std Dev by Season (BoxPlot)	Create graphics		createBox PlotSeason
Map	Create map	goose_tool.py	Not a function
Correlation	Create correlation matriz	correlation.py	my_cor
Summary Statistics	Create csv with summary statistics	summary_stats.py	
Season summary statistics	Create table with season summary statistics	season_summary.py	my_sumattrb
Linear Regression Model	Create the regression model. Export PDF	linear_regression.py	my_lnreg
Tool	Integrate all elements in an ArcGIS Pro toolbox	goose_tool.py	NA
Others	Return min and max values of distance grouping for bird names	min_max_ distance_by_ fielddelimiter.py	min_max_distance
	Return min and max values of IDs grouping for bird names	min_max_ids_by_ fielddelimiter.py	min_max_ids
	Return min and max values of time grouping for bird names	min_max_ timestamp_by_ fielddelimiter.py	min_max_ timestamp
	Return time values based on distance input values	pass_twodistances_ return_twotimes_ tagIdentList_ numpyarray.py	pass_twodistance_ return_time

Module	Description	Script	Function
	Return distances values based on time input values	pass_twotimes_ return_twodistances_ tagIdentList_ numpyarray.py	pass_twotime_ return_distance
	Project all shapefiles from WGS84 to 3857 inside gdb	Projectshape fromfolder_ tofc3857.py	NA
	Calculate cumulative distance for routes construction	calculate_ cumulativedistances_ in_pointsfc.py	NA

Appendix 2

User Manual

INSTALLATION

The easiest way to install and run the Goose Tool is:

1) Copy Folder "PIG" from Appendix 4 (Tool and Python Code) folder and paste to Disk "C:/"

2) Execute the project "Test.aprx"

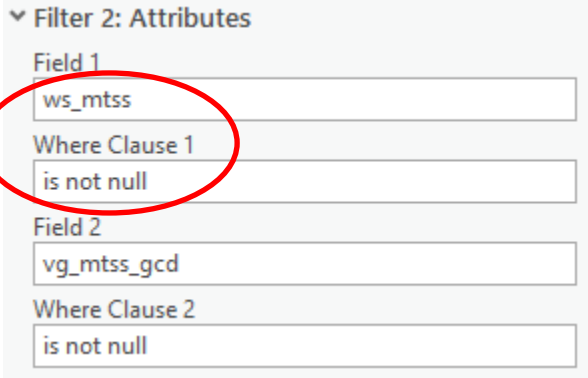
3) Look for the Toolbox named "Test"

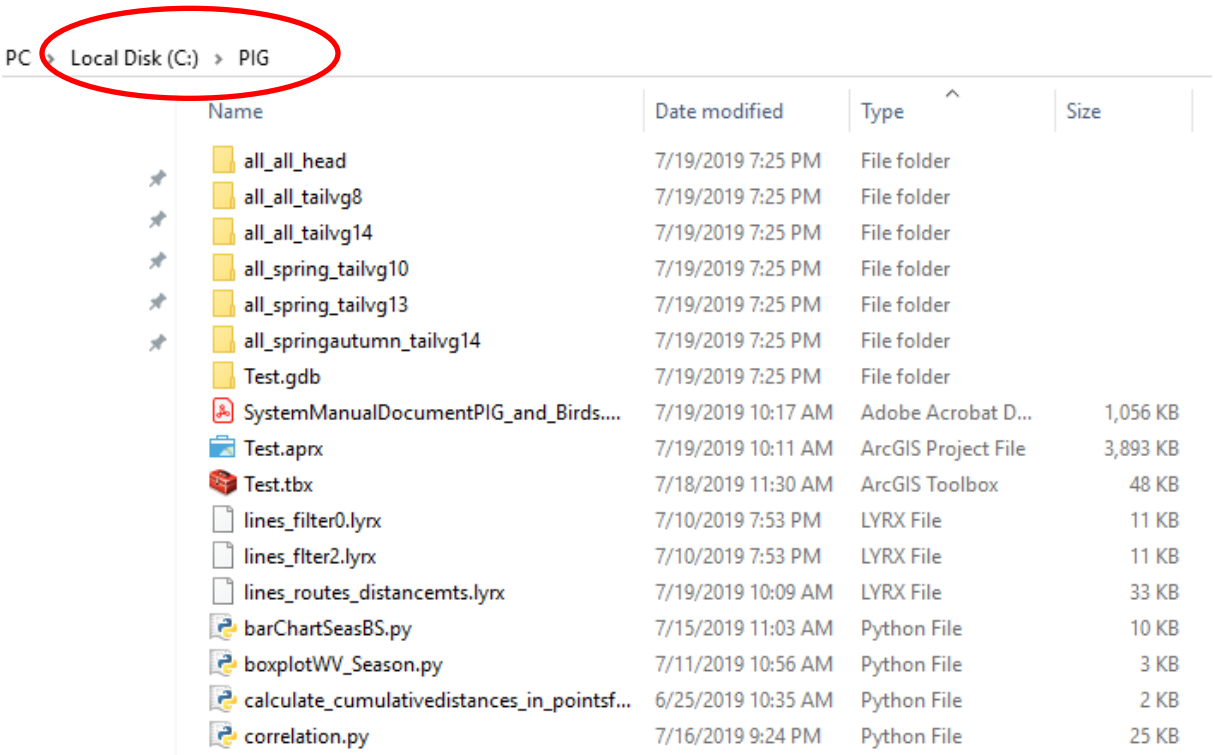
4) Excute Tool "Goose Analysis"

5) Run the tool with default parameters.

Note: Default values are configured to be working in path "C:/PIG"

Caution: The correlation function is sensible to NULL values in the variables. Be sure to use an *attribute filter* for the correlation variables, specially field by default ws_mtss. The attribute filter could be (WS_MTSS IS NOT NULL).




PC › Local Disk (C:) › PIG
Name
Date modified
Type
Size
all_all_head 7/19/2019 7:25 PM File folder
all_all_tailvg8 7/19/2019 7:25 PM File folder
all_all_tailvg14 7/19/2019 7:25 PM File folder
all_spring_tailvg10 7/19/2019 7:25 PM File folder
all_spring_tailvg13 7/19/2019 7:25 PM File folder
all_springautumn_tailvg14 7/19/2019 7:25 PM File folder
Test.gdb 7/19/2019 7:25 PM File folder
SystemManualDocumentPIG_and_Birds.... 7/19/2019 10:17 AM Adobe Acrobat D... 1,056 KB
Test.aprx 7/19/2019 10:11 AM ArcGIS Project File 3,893 KB
Test.tbx 7/18/2019 11:30 AM ArcGIS Toolbox 48 KB
lines_filter0.lyrx 7/10/2019 7:53 PM LYRX File 11 KB
lines_flter2.lyrx 7/10/2019 7:53 PM LYRX File 11 KB
lines_routes_distancemts.lyrx 7/19/2019 10:09 AM LYRX File 33 KB
barChartSeasBS.py 7/15/2019 11:03 AM Python File 10 KB
boxplotWV_Season.py 7/11/2019 10:56 AM Python File 3 KB
calculate_cumulativedistances_in_pointsf... 6/25/2019 10:35 AM Python File 2 KB
correlation.py 7/16/2019 9:24 PM Python File 25 KB

USER INTERFACE

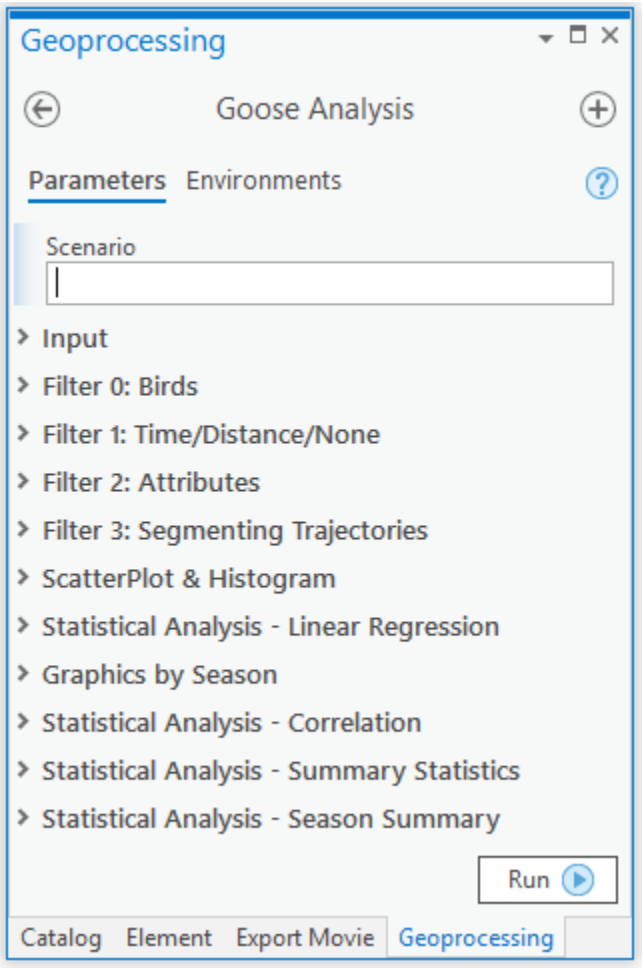
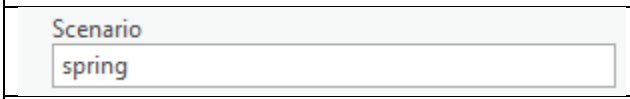
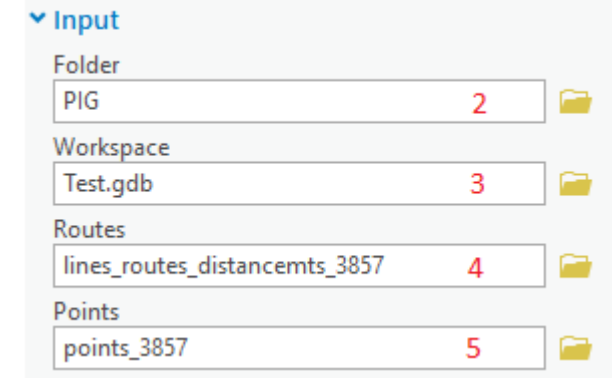
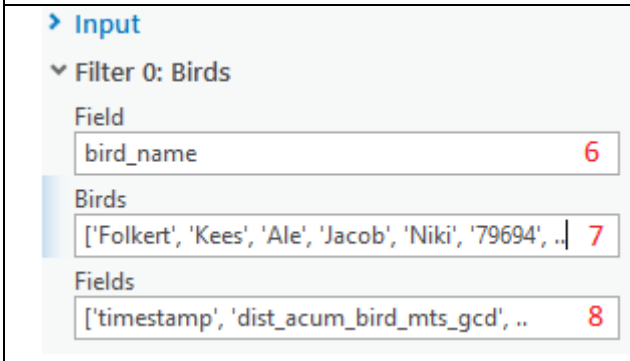
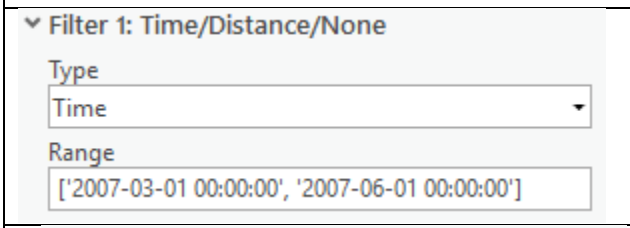
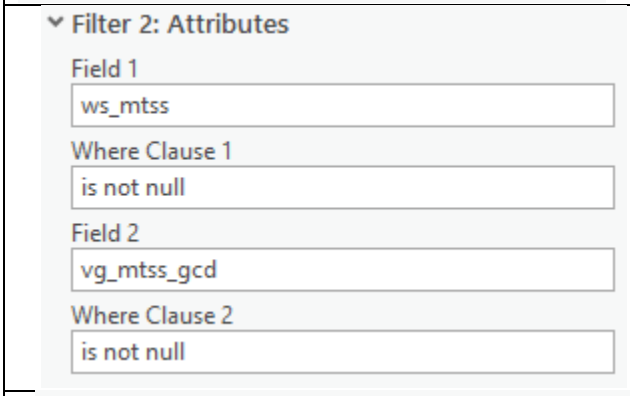
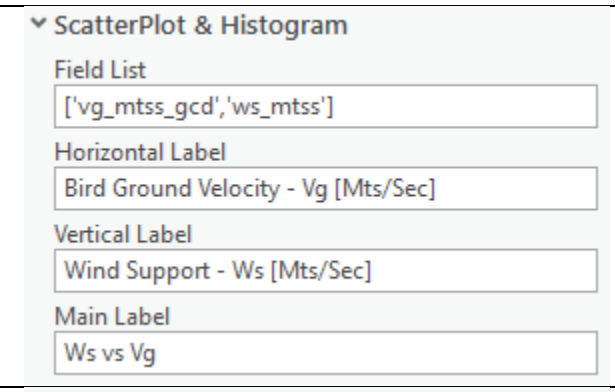
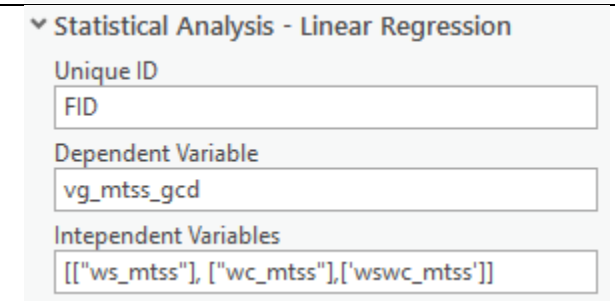
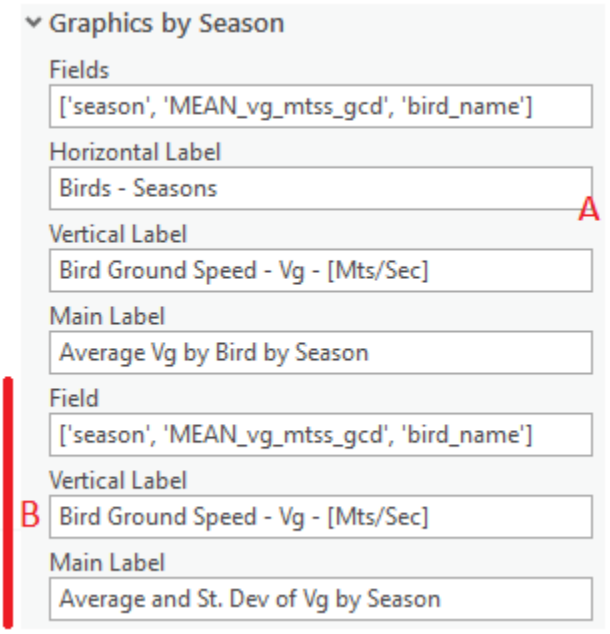
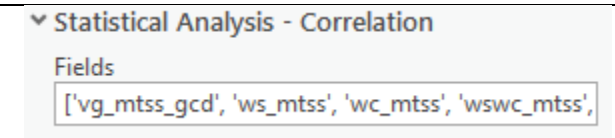
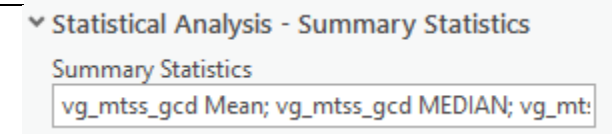


Figure 1. User interface – Goose Analysis Tool

Section	Description of user input
	Scenario: Name of the scenario
	2- Folder: Output base folder 3- Workspace: Input and output file geodatabase 4- Routes: Routes (XYM) of tracked path 5- Points: Tracked bird points with all variables as fields
	6- Field with the bird ID. It could be tag_ident or bird_name 7- List of bird ID. It could be unique values inside tag_ident of bird_name 8- Fields needed for the analysis inside tracked bird points (input 5 in this table) : timestamp, cumulative distance between point and object id
	For type of filter 1, the user can select Time, Distance or None. Either Distance or Time, its range need to be written in Range text box.
	Name of fields and where clause to use as filter by attributes. It is possible to use two fields to filter by attributes Example 1: ws_mtss is not null vg_mtss_gcd is not null Example 2: ws_mtss > 0 vg_mtss_gcd > 2
	Not implemented. Future work.

	Variables and graphic labels to create the scatter plot/histograms graphic.
	Unique ID: existing integer unique ID (different to OBJECTID) inside tracked point feature class (input 4 in this table) Dependent variable Independent variables
	Variables and graphic labels to create graphics by season. Section A: First part A is for the graphic bird-season (X axe with Bird names), Section B: Second part B is for the graphic season-bird (X axe with Season names)
	List of fields to calculate matrix of correlation
	List of fields and statistical parameters names to calculate summary statistics

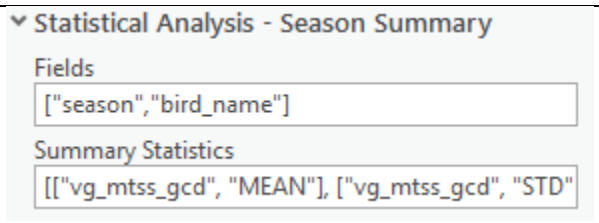
	Fields: List of fields inside Birds point track (input 4 in this table). Summary Statistics: List of fields and statistical parameters names to calculate summary statistics by fields provided in “Fields”

Table 1. Description of the user interface for the tool.

Appendix 3

System Manual Documents

Contents

Application description

Introduction

For the python project a dataset (from Movebank) of tracked birds (coordinates, timestamp, bird identifier, others) was provided in Shapefile format (points and lines). In addition to this provided information, it was necessary to annotate (download supplementary variables from Movebank), join and process additional information related to wind velocity, so the starting point to construct the application was the initial information stored in the tracking dataset plus extra information related to wind.

A well-known method for working with lines geometries in GIS is linear referencing, which in addition to common information stored in every vertex, related to X, Y and Z (optional) coordinates, allows the user to pile Measure information. A linear geometry in GIS vector format, with X, Y, Z, and M, or X, Y, and M information stored in each vertex is called a **route**. In our context one example of a route could be a detailed line that represents the migratory path of a tracked bird, starting from point A and finishing in point B, and storing the cumulative flying distance in each vertex, starting in A with value zero. So it is possible to know with certainty, for instance, where exactly was located the bird when its flying distance was 12.4 kilometers. If in the point database of the tracked path there is available information about time, then is possible, using some lines of code, to use together distances in the route and time information to locate the bird at a specific time (a point event along the line) or in times intervals (line events along the route). The advantage of routes usage in GIS, is the possibility of query and display linear and point

events along the routes based on the stored measure information, this process is called dynamic segmentation.

The application is based in the use of linear referencing and dynamic segmentation to display user filters based on bird's identifiers, distances, times and attributes. This filters are processed serially, orderly one after the other. The use of dynamic segmentation has the only purpose of graphical display of every filter as a set of lines and points, but for the actual analysis of the dataset the results of every filter are transferred to the next one using a list of IDs that satisfy previous query.

The tool could have been done without linear referencing and dynamic segmentation, but then the detailed spatial location, as sets of lines and/or points of each query, it could not have been easily deployed.

Dataset Input for the Tool

Dataset	Description	Example
Birds tracks (line geometry)	Feature class of lines with bird tracks in route (X, Y, Measure) format (linear referencing), this is, in every vertex of the line are stored values of X (east coordinate), Y (north coordinate), and M (cumulative distance in meters)	C:\PIG\Test.gdb\lines_routes_distancemts_3857
Birds tracks (point geometry)	Feature class of points with all the tracked and annotated variables stored as its attributes.	C:\PIG\Test.gdb\points_3857

Dataset Filters

The concept behind this tool to analyze Birds tracks is the use of dynamic segmentation (based on linear referencing, this is lines with X, Y, and Measure values) to display graphically some filters that the user can perform according to Table 1, nonetheless linear referencing and dynamic segmentation are meant only to display graphically the filters as a set of lines or points, but to proceed to the data analysis phase, the result of the filter are transferred as a list of IDs that satisfied the parameters of the query.

Filter Name	Description	Result of dynamic segmentation	Result for next step in the analysis procedure
Filter 0	Filter the dataset by **birds** identifiers	Lines depicting the birds paths that satisfied the users list of birds to analyze	List of IDs that satisfied the filter 0
Filter 1	Allow the user to filter by **time** or **distance**, but in addition includes Filter 0	Lines depicting the birds paths that satisfied the users list of birds and the distance or time specified by the user.	List of IDs that satisfied the filter 1
Filter 2	Allow the user to filter by **attributes**, but in addition includes Filter 0	Lines depicting the birds paths that satisfied the users list of birds and the attributes filter specified by the user.	List of IDs that satisfied the filter 2

Table 1. Filter descriptions of the application

Tool User Inputs

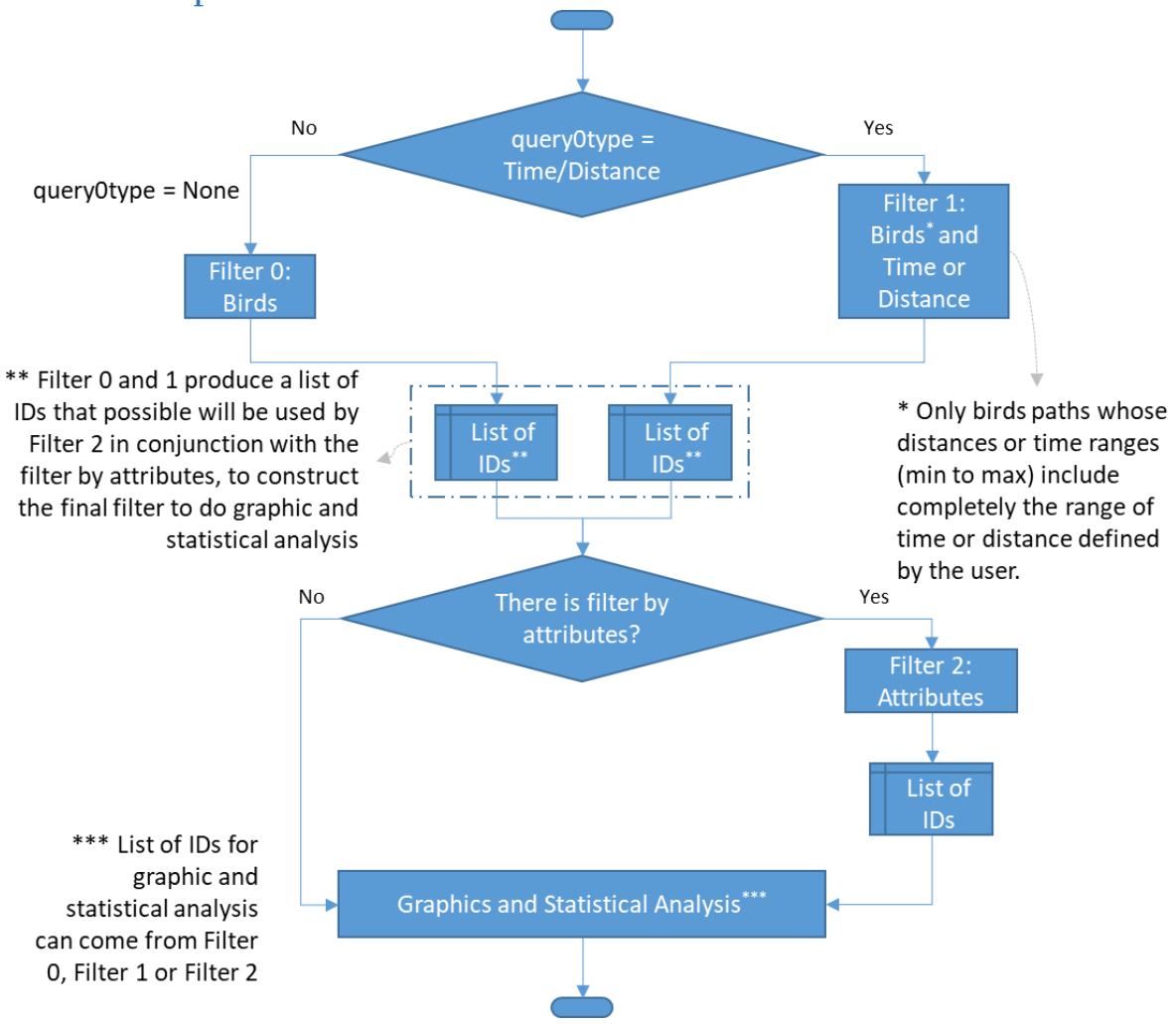
All the user text inputs are taken from the toolbox using arcpy.GetParameterAsText(ID) according to Table 2. All the user text inputs that start and end with square brackets ([,]), are converted to array (python code) using the keyword **eval**, for instance 'eval(user text input)'.

ID	Name	Category	Default Value	Description
0	scenario		spring	
1	folder	Input	Aprx path	
2	workspace	Input	Aprx path/Test.gdb	
3	routes	Input	Aprx path/Test.gdb /lines_routes_distancemts_3857	
4	points	Input	Aprx path/Test.gdb /points_3857	
5	field	Filter 0: Birds	bird_name	
6	birds	Filter 0: Birds	['Folkert', 'Kees', 'Ale', 'Jacob', 'Niki', '79694', '79698']	
7	query0type	Filter 1: Time/Distance/None	Time. Possible values: Distance, Time, None	
8	queryrange	Filter 1: Time/Distance/None	['2007-03-01 00:00:00', '2007-06-01 00:00:00']	
9	field1	Filter 2: Attributes	ws_mtss	
10	whereclause1	Filter 2: Attributes	is not null	
11	field2	Filter 2: Attributes	vg_mtss	
12	whereclause2	Filter 2: Attributes	is not null	
13	segmenttrack	Filter 3: Segmenting Trajectories	false	
14	fieldlistscatterhist	ScatterPlot & Histogram	['vg_mtss','ws_mtss']	
15	horizontallabel	ScatterPlot & Histogram	Bird Ground Velocity - Vg [Mts/Sec]	
16	verticallabel	ScatterPlot & Histogram	Wind Support - Ws [Mts/Sec]	
17	mainlabel	ScatterPlot & Histogram	Ws vs Vg	
18	integeruniqueid	Statistical Analysis – Linear Regression	FID	
19	dependentvariable	Statistical Analysis – Linear Regression	vg_mtss	
20	independentvariables	Statistical Analysis – Linear Regression	[["vw_mtss"], ["ws_mtss"], ["wc_mtss"]]	
21	fieldsforcursor	Graphics by Season	['season', 'MEAN_vg_mtss', 'bird_name']	
22	horizontallabel1	Graphics by Season	Birds - Seasons	
23	verticallabel1	Graphics by Season	Bird Ground Speed - Vg - [Mts/Sec]	
24	mainlabel1	Graphics by Season	Average Vg by Bird by Season	
25	fieldsforcursor2	Graphics by Season	['season', 'MEAN_vg_mtss', 'bird_name']	
26	verticallabel2	Graphics by Season	Bird Ground Speed - Vg -	

ID	Name	Category	Default Value	Description
			[Mts/Sec]	
27	mainlabel2	Graphics by Season	Average and St. Dev of Vg by Season	
28	fieldlistcorrelation	Correlation	['vw_mtss','ws_mtss', 'wc_mtss', 'vg_mtss', 'va_mtss']	
29	summarystatistics	Statistical Analysis – Summary Statistics	vw_mtss MEAN; vw_mtss MEDIAN; vw_mtss STD; vw_mtss VARIANCE; vg_mtss Mean; vg_mtss MEDIAN; vg_mtss STD; vg_mtss VARIANCE; va_mtss Mean; va_mtss MEDIAN; va_mtss STD; va_mtss VARIANCE	
30	fieldlistseasonsum2	Statistical Analysis – Season Summary	["season","bird_name"]	
31	summarystatistics2	Statistical Analysis – Season Summary	[["vw_mtss", "MEAN"], ["vw_mtss", "STD"], ["vw_mtss", "VARIANCE"], ["vg_mtss", "MEAN"], ["vg_mtss", "STD"], ["vg_mtss", "VARIANCE"], ["va_mtss", "MEAN"], ["va_mtss", "STD"], ["va_mtss", "VARIANCE"]]	
32	infields	Filter 0: Birds	['timestamp', 'distancemts', 'OBJECTID']	

Table 2. User Inputs

Filters Descriptions



Filter 0

Filter Name	When?	Script	Function
Filter 0	User input 7 is equal to None. query0type = 'None'	pass_birdlist_return_numpyarray.py	pass_birdlist_return_numpyarray

Input	User input ID or other	Example
inFcPoints	4	C:\PIG\Test.gdb\points_3857
fieldDelimiter	5	bird_name
inFields	[5, 32]	[bird_name, 'timestamp', 'distancemts', 'OBJECTID']
birds	6	['Folkert', 'Kees', 'Ale', 'Jacob', 'Niki', '79694', '79698']

Output	Description	Example

Filter Name	When?	Script	Function
myResutlNumpyArray2		**Array**. The formatted array to create the lines event table that will represent all the lines that cover the points that satisfied this filter 0.	('Folkert', 'LINE', 0., 19874140.67592672, '2007-01-31 13:00:00', '2007-11-15 14:00:00', 2883, 3921), ('Kees', 'LINE', …), ('Ale', …), ('Jacob', …), ('Niki', …), ('79694', …), ('79698', …)
scenario + "_birds_lineevents_ table_filter0"		**Table** of events created with myResutlNumpyArray2	C:\PIG\Test.gdb\spring_ birds_lineevents_ table_filter0 **Note**: This result is not created inside of the function pass_birdlist_return_numpyarray
scenario + "_birds_lineevents_ fc_filter0"		**Feature class** of lines after applying dynamic segmentation of table of events described in previous rom. Always stored inside the dataset 'scenario'	C:\PIG\Test.gdb\spring\spring_ birds_lineevents_ table_filter0 **Note**: This result is not created inside of the function pass_birdlist_return_numpyarray
fieldForNextFilter		The **name of the field** identifier inside the user input 4. The next list of IDs (idsForNextFilter) belongs to this field.	OBJECTID
idsForNextFilter		**Array** with list of all IDs inside the user input 4, that satisfy this filter 0.	(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, ..)

Table 3. Description of Filter 0

Filter 1

Filter Name	When?	Script	Function
Filter 1	User input 7 is equal to Distance query0type = 'Distance'	pass_twodistances_return_twotimes_ tagIdentList_numpyarray.py	pass_twodistance_ return_time
	User input 7 is equal to Time. query0type = 'Time'	pass_twotimes_return_twodistances_ tagIdentList_numpyarray.py	pass_twotime_return_ distance

Input	User input ID or other	Example
inFcPoints	4	C:\PIG\Test.gdb\points_3857
fieldDelimiter	5	bird_name
inFields	[5, 32]	[bird_name, 'timestamp', 'distancemts', 'OBJECTID']
queryrange	8	['2007-03-01 00:00:00', '2007-06-

Filter Name	When?	Script	Function
			01 00:00:00']
			[0, 10000]
myNewFieldDelimiterList		New list of Birds after check if every line that represents a bird identifier has the range of values defined in queryrange, completely inside its minimum and maximum values of distance or time.	['Folkert', 'Kees', 'Ale', 'Jacob', 'Niki']

Output	Description	Example
myResutlNumpyArray2	**Array**. The formatted array to create the lines event table that will represent all the lines that cover the points that satisfied this filter 1.	('Ale', 'LINE', 217815.81012144, 9334512.59814454, '2007-03-01 00:00:00', '2007-06-01 00:00:00', 21, 337), ('Kees', 'LINE', …), ('Folkert', …), ('Niki', …), ('Jacob', …)
scenario + "_distance_ lineevents_table_filter1"	**Table**. If query0type = 'Distance', this table of events is created with myResutlNumpyArray2	C:\PIG\Test.gdb\spring_distance_ lineevents_table_filter1 **Note**: This result is not created inside of the function pass_twodistance_return_time
scenario + "_time_ lineevents_table_filter1"	**Table**. If query0type = 'Time', this table of events is created with myResutlNumpyArray2	C:\PIG\Test.gdb\spring_time_ lineevents_table_filter1 **Note**: This result is not created inside of the function pass_twotime_return_ distance
scenario + "_distance_ lineevents_table_filter1"	**Feature Class** always stored inside the dataset 'scenario'. If query0type = 'Distance', this line feature class of events is created with myResutlNumpyArray2	C:\PIG\Test.gdb\spring|spring_ distance_lineevents_table_filter1 **Note**: This result is not created inside of the function pass_twodistance_return_time
scenario + "_time_ lineevents_table_filter1"	**Feature Class** always stored inside the dataset 'scenario'. If query0type = 'Time', this line feature class of events is created with myResutlNumpyArray2	C:\PIG\Test.gdb\spring\spring_ time_lineevents_table_filter1 **Note**: This result is not created inside of the function pass_twotime_return_distance
fieldForNextFilter	The **name of the field** identifier inside the user input 4. The next list of IDs (idsForNextFilter) belongs to this field.	OBJECTID
idsForNextFilter	**Array** with list of all IDs inside the user input 4, that satisfy this filter 1.	(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30,

Filter Name	When?	Script	Function
		31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, ..)	

Table 4. Description of Filter 1. This include also Filter 0 but after checking min and max values of time or distance.

Filter 2

Filter Name	When?	Script	Function
Filter 2	Occurs if there are attributes to filter. User inputs 9 to 12.	filter_by_attributes_ structured_array.py	filter_by_fields_where

Input	User input ID or other	Example
inFcPoints	4	C:\PIG\Test.gdb\points_3857
fieldDelimiter	5	bird_name
inFields	[5, 32]	[bird_name, 'timestamp', 'distancemts', 'OBJECTID']
fieldDelimiter2	Name of a field that represents an object identifier inside user input 4. Taken from the output fieldForNextFilter coming from Filter 0 or Filter 1	OBJECTID
fieldDelimiterList2	List of IDs coming from Filter 0 or Filter 1 and stored in variable idsForNextFilter.	21, 22, 23, 850, 851, …, 1360, 1363
field1	The name of a field to construct a filter by attributes. User input 9	ws_mtss
condition1	The condition belonging to previous field. User input 10	is not null
field2	The name of a field to construct a filter by attributes. User input 11	vg_mtss
condition2	The condition belonging to previous field. User input 12	is not null

Output	Description	Example
myResutlNumpyArray2	**Array**. The formatted array to create lines and point event tables that that represent the output of filter 2. If the event table is type lines, then all the points in user input 4 that touch those lines will be part of the filter result. If the event table is type points, then all the points in user input 4 that	('Ale', 'LINE', 255603.26108147, 256831.89446036, '2007-03-10 13:00:00', '2007-03-10 14:00:00', 21, 22), ('Ale', 'POINT', 264814.55609887, 264814.55609887, '2007-03-12 06:00:00', '2007-03-12 06:00:00', 25, 25), ('Ale', 'POINT', ...), ('Ale', 'POINT', ...), ('Ale', 'LINE', ...), ('Ale', 'LINE', ...), ('Ale', 'LINE', ...),

Filter Name	When?	Script	Function
		have the same coordinates will be part of the filter result. The result of this array could be for event type lines, points or both.	('Ale', 'POINT', ...)
scenario + "_attributes_ lineevents_table_filter2"		This **table** of line events is created with myResutlNumpyArray2, using only those rows of the array whose type is LINE.	C:\PIG\Test.gdb\spring_attributes_ lineevents_table_filter2 **Note**: This result is not created inside of the function filter_by_fields_where
scenario + "_attributes_ pointevents_table_filter2"		This **table** of point events is created with myResutlNumpyArray2, using only those rows of the array whose type is POINT.	C:\PIG\Test.gdb\spring_attributes_ pointevents_table_filter2 **Note**: This result is not created inside of the function filter_by_fields_where
scenario + "_attributes_ lineevents_fc_filter2"		This **feature class** of line events is created with myResutlNumpyArray2, using only those rows of the array whose type is LINE.	C:\PIG\Test.gdb\spring\spring_ attributes_lineevents_fc_filter2 **Note**: This result is not created inside of the function filter_by_fields_where
scenario + "_attributes_ pointevents_fc_filter2"		This **feature class** of point events is created with myResutlNumpyArray2, using only those rows of the array whose type is POINT.	C:\PIG\Test.gdb\spring\spring_ attributes_pointevents_fc_filter2 **Note**: This result is not created inside of the function filter_by_fields_where
fieldForNextFilter		The name of the field identifier to which next list of IDs belongs.	OBJECTID
idsForNextFilter		Array with list of all IDs that satisfy this filter 1	(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, ..)

Table 5. Description of Filter 2. This filter does not include anything related to bird names.

Graphic analysis

Graphic	Script and Function	Description
Scatter plot & Histograms Graphics	scatter_histo.py / scatter_histo	This function will create a plot with three graphics. One main graphic for a scatterplot, then two histograms, one for each variable used in the axes of the scatterplot.
Input	**Description**	**Example**
fc_or_table	User input 4	C:\PIG\Test.gdb\points_3857
myFieldList	User input 14	['vg_mtss', 'ws_mtss']. First field is for the horizontal graphic (also horizontal axe) and second field is

		for vertical graphic (also vertical axe)
myFieldToFilter	User input 5. This filter is not used in this function as the system already did the necessary filters. See parameters fieldDelimiter2 and fieldDelimiterList2.	myFieldToFilter = “”
myListOfValuesToFilter	User input 6, but possibly after filter values with min and max of distance or time (Filter 1). This filter is not used in this function as the system already did the necessary filters. See parameters fieldDelimiter2 and fieldDelimiterList2.	myListOfValuesToFilter = []
varticalLabel	Label for the vertical axe of the graphic. It depends of the second field of the array defined in myFieldList (user input 14)	Wind Support (Ws)
mainlabel	Title of the graphic	Ws vs Vg
horizontalLabel	Label for the horizontal axe of the graphic. It depends of the first field of the array defined in myFieldList (user input 14)	Bird Ground Velocity (Vg)
my_new_file	Name for the output graphic. Inside the folder (user input 1) the system will create the folder scenario (user input 0) and inside it will place the graphic	C:\PIG\spring\ spring_scatterplot_histo.png In this path the value of scenario (user input 0) is ‘spring’.
field1	The function can do filter by attributes, but as the system already did the filter (if the user selects to do so), these parameters where not used (value = “”). See parameters fieldDelimiter2 and fieldDelimiterList2.	field1 = “”
condition1		condition1 = “”
field2		field2 = “”
condition2		condition2 = “”
fieldForNextFilter	The name of the field identifier to which next list of IDs belongs.	OBJECTID
idsForNextFilter	Array with list of all IDs to be used to filter points (in user input 4), then construct the graphic only with this filtered points.	(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, ..)
Output	**Description**	**Example**
my_new_file	Name of the output graphic. Inside the folder (user input 1) the system will create the folder scenario (user input 0) and inside it will place the graphic	C:\PIG\spring\ spring_scatterplot_histo.png In this path the value of scenario (user input 0) is ‘spring’.

Table 6. Scatter Plot & Histogram function

Graphic	Script and Function	Description
Graphics by bird by season	barChartSeasBS.py / createBarChartBirdsSeason	This function will create bar plots with average values of one variable (mainly Vg) by bird and by season.
Input	**Description**	**Example**
wspace	Source table to construct the graphics. Table with the summary statistics by season by bird	C:\PIG\Test.gdb\ spring_season_sum_stat
myFieldList	User input 21. List of fields to construct the filter.	['season', 'MEAN_vg_mtss', 'bird_name'].
my_new_file	Path to output PNG graphics.	C:\PIG\spring\ spring_barchartbirdseason.png
horizontallabel	String with the horizontal label of the graphic	Birds - Seasons
verticallabel	String with the vertical label of the graphic	Bird Ground Speed – Vg [Mts/Sec]
mainlabel	String with the main label of the graphic	Average Vg by Bird by Season
Output	**Description**	**Example**
my_new_file	Path to output PNG graphics.	C:\PIG\spring\ spring_barchartbirdseason.png

Table 7. Function to create graphics by bird by season

Graphic	Script and Function	Description
Graphics by season by bird	barChartSeasBS.py / createBarChartSeasonBirds	This function will create bar plots with average values of one variable (mainly Vg) by season and by bird.
Input	**Description**	**Example**
wspace	Source table to construct the graphics. Table with the summary statistics by season by bird	C:\PIG\Test.gdb\ spring_season_sum_stat
myFieldList	User input 21. List of fields to construct the filter.	['season', 'MEAN_vg_mtss', 'bird_name'].
my_new_file	Path to output PNG graphics.	C:\PIG\spring\ spring_barchartseasonbird.png
horizontallabel	String with the horizontal label of the graphic. Note that this label is the same user input as graphic represented in previous table (Table 7)	Birds - Seasons
verticallabel	String with the vertical label of the graphic. Note that this label is the same user input as graphic	Bird Ground Speed – Vg [Mts/Sec]

	represented in previous table (Table 7)	
mainlabel	String with the main label of the graphic. Note that this label is the same user input as graphic represented in previous table (Table 7)	Average Vg by Bird by Season
Output	**Description**	**Example**
my_new_file	Path to output PNG graphics.	C:\PIG\spring\ spring_barchartseasonbird.png

Table 8. Function to create graphics by season by bird

Graphic	**Script and Function**	**Description**
Mean Bird Speed and Std Dev by Season (BoxPlot)	barChartSeasBS.py / createBoxPlotSeason	This function will create box plots with average values and standard deviation ranges of one variable (mainly Vg) grouping by season.
Input	**Description**	**Example**
wspace	Source table to construct the graphics. Table with the summary statistics by season by bird	C:\PIG\Test.gdb\ spring_season_sum_stat
myFieldList	User input 25. List of fields to construct the filter.	['season', 'MEAN_vg_mtss', 'bird_name'].
my_new_file	Path to output PNG file to store the graphic.	C:\PIG\spring\ spring_boxplotseason.png
verticallabel	String with the vertical label of the graphic. User input 26.	Bird Ground Speed – Vg [Mts/Sec]
mainlabel	String with the main label of the graphic. User input 27.	Average and St. Dev of Vg by Season
Output	**Description**	**Example**
my_new_file	Path to output PNG graphics.	C:\PIG\spring\ spring_barchartseasonbird.png

Table 9. Function to create boxplot graphic by season

Graphic	**Script and Function**	**Description**
Map	goose_tool.py	Code to create the map in format PNG. It is not a function, it is code inside the main python file of the tool.

Table 10. Code to create the final Map

Statistical analysis

Analysis	Script and Function	Description
Correlation	correlation.py / my_cor	To create a text file with the correlation matrix between variables.
Input	**Description**	**Example**
fc_or_table	User input 4	C:\PIG\Test.gdb\points_3857
myFieldList	User input 28	['vw_mtss','ws_mtss', 'wc_mtss', 'vg_mtss', 'va_mtss']
myFieldToFilter	User input 5. This filter is not used in this function as the system already did the necessary filters. See parameters fieldDelimiter2 and fieldDelimiterList2.	myFieldToFilter = ""
myListOfValuesToFilter	User input 6, but possibly after filter values with min and max of distance or time (Filter 1). This filter is not used in this function as the system already did the necessary filters. See parameters fieldDelimiter2 and fieldDelimiterList2.	myListOfValuesToFilter = []
fieldDelimiter2	The name of the field identifier to which next list of IDs belongs.	OBJECTID
fieldDelimiterList2	Array with list of all IDs to be used to filter points (in user input 4), then construct the correlation matrix only with this filtered points.	(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, ..)
Output	**Description**	**Example**
corr_report	Output **text file** with the correlation matrix. Note: This file is not created inside the function.	C:\PIG\spring\ spring_correlation.txt

Table 11. Correlation function

Analysis	Script and Function	Description
Season summary statistics	season_summary.py / my_sumattrb	To create a table with summary statistics by season by bird. This table will be used to construct graphics by season and by bird.
Input	**Description**	**Example**
fc_or_table	User input 4	C:\PIG\Test.gdb\points_3857
myFieldToFilter	User input 5. This filter is not used in this function as the system already did the necessary filters. See parameters fieldDelimiter2 and fieldDelimiterList2.	myFieldToFilter = ""

myListOfValuesToFilter	User input 6, but possibly after filter values with min and max of distance or time (Filter 1). This filter is not used in this function as the system already did the necessary filters. See parameters fieldDelimiter2 and fieldDelimiterList2.	myListOfValuesToFilter = []
theTable	Table path to create the summary statistics by season by bird	C:\PIG\Test.gdb\ spring_season_sum_stat
summaryStatistics	User input 31. All the summary statistics that the function will perform based on next parameter (summaryFields)	[["vw_mtss", "MEAN"], ["vw_mtss", "STD"], ["vw_mtss", "VARIANCE"], ["vg_mtss", "MEAN"], ["vg_mtss", "STD"], ["vg_mtss", "VARIANCE"], ["va_mtss", "MEAN"], ["va_mtss", "STD"], ["va_mtss", "VARIANCE"]]
summaryFields	User input 30. The base fields used by the function to construct the summary statistics.	["season","bird_name"]
fieldDelimiter2	The name of the field identifier to which next list of IDs belongs.	OBJECTID
fieldDelimiterList2	Array with list of all IDs to be used to filter points (in user input 4), then construct the season summary statistics only with this filtered points.	(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, ..)
Output	**Description**	**Example**
theTable	Table with the summary statistics by season by bird	C:\PIG\spring\ spring_season_sum_stat

Table 12. Season summary statistics function

Analysis	**Script and Function**	**Description**
Linear Regression Model	linear_regression.py / my_lnreg	Create the linear regression model to test independent variables (mainly Ws, Wc) against dependent variables (mainly Vg). But the user can select the variables.
Input	**Description**	**Example**
fc_or_table	User input 4	C:\PIG\Test.gdb\points_3857
u_id	Integer unique ID **field** inside user input 4. This value is passed by the user in input 18, and it is required to exist previously in user input 4.	FID
data_store	The output point **feature class** path with the values of the regression model.	C:\PIG\Test.gdb\spring\ spring_regression

d_var	Dependent variable	vg_mtss
i_var	Independent variables	[["ws_mtss"], ["wc_mtss"]]
coeff_op	**Table** to store the coefficients of the linear regression model. The full path to an table that will receive model coefficients, standardized coefficients, standard errors, and probabilities for each explanatory variable.	C:\PIG\Test.gdb\spring_reg_coeff
diag_op	The full path to an **table** that will receive model summary diagnostics.	C:\PIG\Test.gdb\spring_reg_diag_op
res_report	The path to the **PDF** file with the report. This report file includes model diagnostics, graphs, and notes to help you interpret the OLS results.	C:\PIG\spring\spring_reg_result.pdf
myFieldToFilter	User input 5. This filter is not used in this function as the system already did the necessary filters. See parameters fieldDelimiter2 and fieldDelimiterList2.	myFieldToFilter = ""
myListOfValuesToFilter	User input 6, but possibly after filter values with min and max of distance or time (Filter 1). This filter is not used in this function as the system already did the necessary filters. See parameters fieldDelimiter2 and fieldDelimiterList2.	myListOfValuesToFilter = []
fieldDelimiter2	The name of the field identifier to which next list of IDs belongs.	OBJECTID
fieldDelimiterList2	Array with list of all IDs to be used to filter points (in user input 4), then construct the linear regression model only with this filtered points.	(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, ..)
Output	**Description**	**Example**
data_store	The output point **feature class** path with the values of the regression model.	C:\PIG\Test.gdb\spring\ spring_regression
coeff_op	**Table** with the resulting coefficients of the linear regression model. It contains model coefficients, standardized coefficients, standard errors, and probabilities for each explanatory variable.	C:\PIG\Test.gdb\spring_reg_coeff
diag_op	**Table** that with the model summary diagnostics.	C:\PIG\Test.gdb\spring_reg_diag_op
res_report	**PDF** file with the report. This report file includes model diagnostics,	C:\PIG\spring\spring_reg_result.pdf

	graphs, and notes to help you interpret the OLS results.	

Table 13. Linear regression model function

User Interface

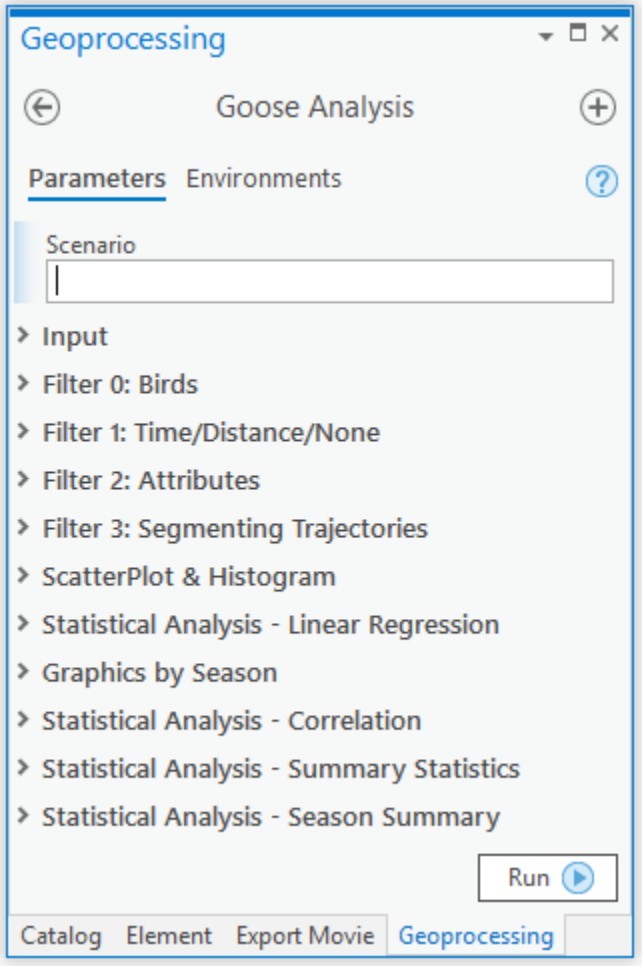
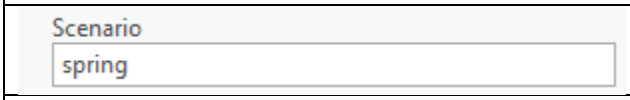
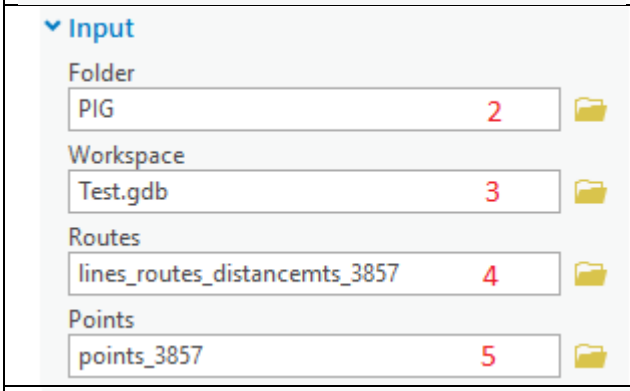
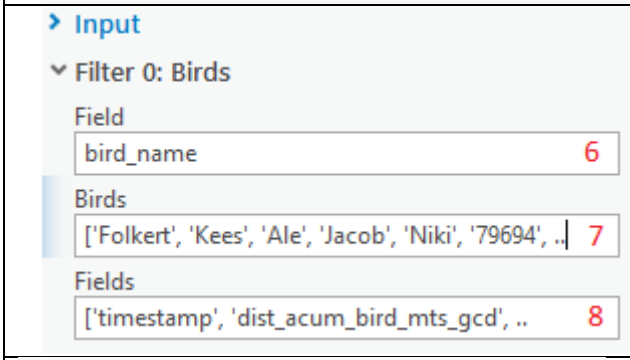
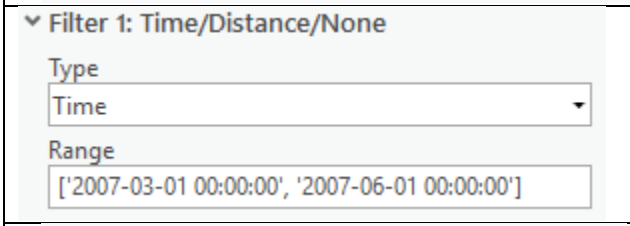
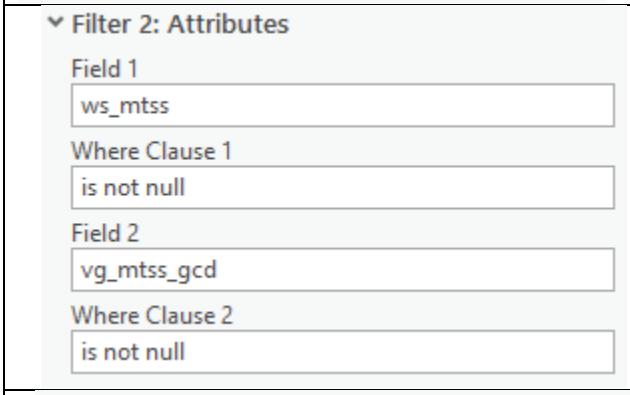
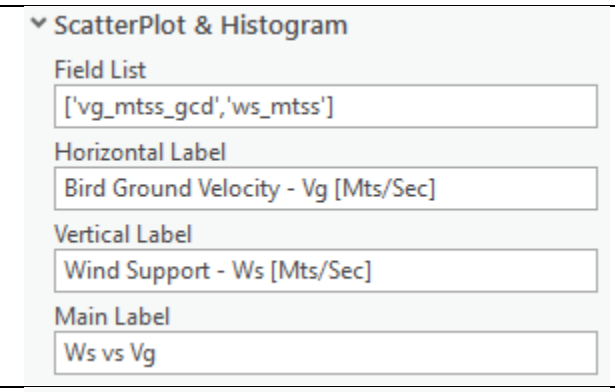
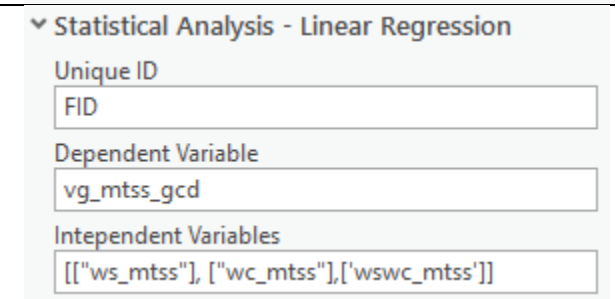
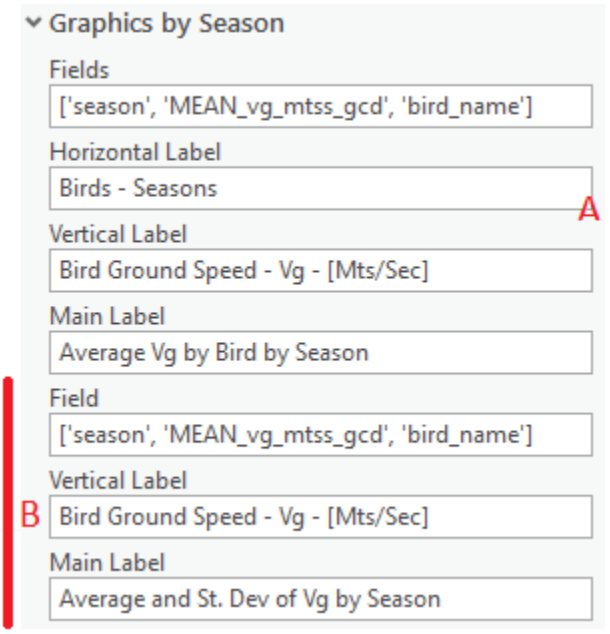
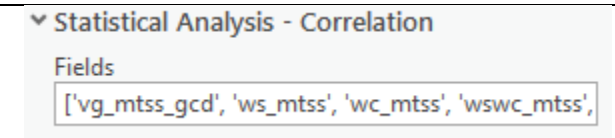
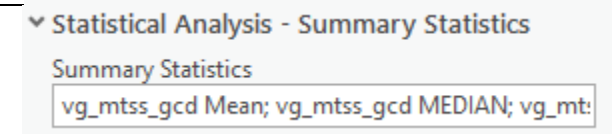


Figure 1. User interface – Goose Analysis Tool

Section	Description of user input
	Scenario: Name of the scenario
	2- Folder: Output base folder 3- Workspace: Input and output file geodatabase 4- Routes: Routes (XYM) of tracked path 5- Points: Tracked bird points with all variables as fields
	6- Field with the bird ID. It could be tag_ident or bird_name 7- List of bird ID. It could be unique values inside tag_ident of bird_name 8- Fields needed for the analysis inside tracked bird points (input 5 in this table) : timestamp, cumulative distance between point and object id
	For type of filter 1, the user can select Time, Distance or None. Either Distance or Time, its range need to be written in Range text box.
	Name of fields and where clause to use as filter by attributes. It is possible to use two fields to filter by attributes Example 1: ws_mtss is not null vg_mtss_gcd is not null Example 2: ws_mtss > 0 vg_mtss_gcd > 2
	Not implemented. Future work.

	Variables and graphic labels to create the scatter plot/histograms graphic.
	Unique ID: existing integer unique ID (different to OBJECTID) inside tracked point feature class (input 4 in this table) Dependent variable Independent variables
	Variables and graphic labels to create graphics by season. Section A: First part A is for the graphic bird-season (X axe with Bird names), Section B: Second part B is for the graphic season-bird (X axe with Season names)
	List of fields to calculate matrix of correlation
	List of fields and statistical parameters names to calculate summary statistics

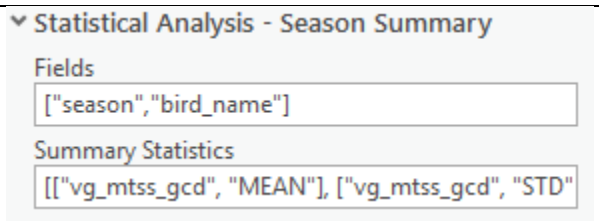
	Fields: List of fields inside Birds point track (input 4 in this table). Summary Statistics: List of fields and statistical parameters names to calculate summary statistics by fields provided in "Fields"

Table 1. Description of the user interface for the tool.

Appendix 4

Task Distribution

MEMBERS: Alexys Rodríguez, Annette Achieng, Hassan Omar, Senait Meles, Somnath Chaudhuri

Table 1. Overall project participation

Project Component	Participation
Initial literature review and project design	Alexys, Som
Movebank Annotation	Hassan, Alexys
Data preprocessing	Alexys
Ideas, project design, model	Som, Alexys, Hassan, Senait, Annette
Python Coding	Senait, Hassan, Som, Alexys
Toolbox	Alexys
Scenario Modeling and Output Analysis	Som, Alexys
Presentation	Hassan, Som, Alexys
Documentation	Annette, Som, Alexys, Senait

Table 2. Coding and Tool implementation

Module	Description	Script	Function	Author	
				Static	Dynamic
Filter 0. Birds	Filter using nor only SQL but also dynamic segmentation	pass_birdlist_ return_ numpyarray.py	pass_birdlist_ return_ numpyarray	Alexys	
Filter 1. Distance Time	Filter using nor only SQL but also dynamic segmentation	pass_twodistances_ return_twotimes_ tagIdentList_ numpyarray.py	pass_ twodistance_ return_time		
Filter 2. Attributes	Filter using nor only SQL but also dynamic segmentation	filter_by_ attributes_ structured_ array.py	filter_by_ fields_where		
Scatter plot & Histograms Graphics	Create graphics	scatter_histo.py	scatter_histo	Hassan	
Graphics by bird by season	Create graphics	barChartSeasBS.py	createBarChart BirdsSeason	Senait	Alexys
Graphics	Create graphics		createBarChart	Senait	Alexys

Module	Description	Script	Function	Author	
				Static	Dynamic
by season by bird			SeasonBirds	Alexys	
Mean Bird Speed and Std Dev by Season (BoxPlot)	Create graphics		createBox PlotSeason	Senait	Alexys
Map	Create map	goose_tool.py	Not a function	Alexys	
Correlation	Create correlation matriz	correlation.py	my_cor	Som	Som Alexys
Summary Statistics	Create csv with summary statistics	summary_stats.py		Som	Som Alexys
Season summary statistics	Create table with season summary statistics	season_summary.py	my_sumattrb	Som	
Linear Regression Model	Create the regression model. Export PDF	linear_regression.py	my_lnreg	Som	Som Alexys
Tool	Integrate all elements in an ArcGIS Pro toolbox	goose_tool.py	NA	Alexys	
Others	Return min and max values of distance grouping for bird names	min_max_ distance_by_ fielddelimiter.py	min_max_distance		
	Return min and max values of IDs grouping for bird names	min_max_ids_by_ fielddelimiter.py	min_max_ids		
	Return min and max values of time grouping for bird names	min_max_ timestamp_by_ fielddelimiter.py	min_max_ timestamp		
	Return time values based on distance input values	pass_twodistances_ return_twotimes_ tagIdentList_ numpyarray.py	pass_twodistance_ return_time		
	Return	pass_twotimes_	pass_twotime_		

Module	Description	Script	Function	Author	
				Static	Dynamic
	distances values based on time input values	return_twodistances_ tagIdentList_ numpyarray.py	return_distance		
	Project all shapefiles from WGS84 to 3857 inside gdb	Projectshape fromfolder_ tofc3857.py	NA		
	Calculate cumulative distance for routes construction	calculate_ cumulativedistances_ in_pointsfc.py	NA		

Table 3. Report Writing

Section	Participation
Introduction	Annette, Senait, Hassan, Som, Alexys
Methodology	Alexys
Output results and Analysis	Som
Discussion	Alexys, Som
Appendix: Tool document	Alexys
Appendix: Output results	Som, Alexys