An Awesome Thesis That Will Prove to the Universe That I Really Deserve This Honorable Degree

by

Albert Richard Einstein, III

A.A.S., University of Southern Swampland, 1988
M.S., Art Therapy, University of New Mexico, 1991

THESIS

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science
Mathematics

The University of New Mexico

Albuquerque, New Mexico

December, 2017

Dedication

To my parents, Albert II and Gladys, for their support, encouragement and the Corvette they're giving me for graduation.

"A bird in hand is worth two in the bush" – Anonymous

Acknowledgments

I would like to thank my advisor, Professor Martin Sheen, for his support and some great action movies. I would also like to thank my dog, Spot, who only ate my homework two or three times. I have several other people I would like to thank, as well.[1]

[1]To my brother and sister, who are really cool.

An Awesome Thesis That Will Prove to the Universe That I Really Deserve This Honorable Degree

by

Albert Richard Einstein, III

A.A.S., University of Southern Swampland, 1988
M.S., Art Therapy, University of New Mexico, 1991

M.S., Mathematics, University of New Mexico, 2017

Abstract

The theory of relativity is a real "toughie" to prove, but with the help of my family and my great grandpa Al, this paper presents the proof in its entirety. Most of the math is correct, and the part about "warp speed" and "parallel universe" sounds very high-tech.

Contents

List of Figures viii

List of Tables xi

Glossary xiii

1 Introduction 1

1.1 Overview 1

1.2 Conclusions 1

2 Background Study 2

2.1 Power Law 3

2.2 CPFA 4

2.3 Genetic Algorithm 6

2.3.1 Selection 7

2.3.2 Crossover 7

2.3.3 Mutation 7

3 Method **10**

3.1 Setting Simulation Environment 10

3.2 Tuning Parameters using Genetic Algorithm 11

3.3 Generating Data Set for Analysis 15

3.4 Analyzing The Foraging Data 15

3.4.1 Creating Timeline for each type of distribution 16

3.5 Change Point Detection Algorithm 18

3.5.1 Calculating the Cumulative Sum 18

3.5.2 Detrending 19

3.6 Binary-Segmented Cumulative Sum 21

3.7 Verification of Change Points 21

3.8 Applying the best method on Field Data 23

4 Results **24**

4.1 Results from Simulation 24

4.2 Parameter setup for Change Point Detection Methods 29

5 Conclusion **32**

5.1 Overview 32

5.2 Conclusions 32

6 Appendicies **33**

6.1 Overview 33

6.2 Conclusions 33

List of Figures

2.1 Distribution of Seeds in the field experiment for power law distribution with power rank 5. Red Pile indicates one large pile of 256 seeds. 4 purple piles represent 4 large piles of 64 seeds, Green color represents 16 piles of 16 seeds and blue seeds are 256 random seeds . 3

3.1 Steps of Change Point Analysis 10

3.2 Example setup of a simulation environment for *P. Rugosus* with total 1024 seeds. The setup showing three different types of piles. One large pile of 256 seeds, Four piles of 64 seeds and sixteen piles of 16 seeds. 256 random seeds are distributed uniformly inside the ring. . 12

3.3 An example of a timeline for a distribution where numbers at the top represent the sliding window number. Values in the boxes are the rate of collection of seeds per window. 16

3.4 An example of timeline and change in foraging rate. The change in foraging rate is calculated by measuring the difference between the timeline windows 17

3.5 This figure demonstrates how the cumulative sum is calculated from the timeline of foraging rate. 18

3.6 An example of applying linear and constant detrending on the cumulative sum of a timeline from one simulated CPFA experiment. . 20

3.7 Four Different Change point Detection Method 23

4.1 Comparison of Change Point Detection Methods without Outliers for Pheromone Only Parameters. The time in seconds represents the difference between a pheromone laying event followed by a detection of change point in the foraging rate. 25

4.2 Enlarged view of efficiency of*Method* α. The time in seconds represents the difference between a pheromone laying event followed by a detection of change point in the foraging rate. 26

4.3 Efficiency chart for Change Point Detection Methods On a large pile of 256 seeds. This result is generated from the simulated data of pheromone only environment, configured for *P. Rugosus* 27

4.4 Efficiency chart for Change Point Detection Methods On four medium pile of 64 seeds. This result is generated from the simulated data of pheromone only environment, configured for *P. Rugosus* 28

4.5 Efficiency chart for Change Point Detection Methods On sixteen small pile of 16 seeds. This result is generated from the simulated data of pheromone only environment, configured for *P. Rugosus* . . 29

4.6 Comparison of Change Point Detection Methods without Outliers for All Parameters. The time in seconds represents the difference between a pheromone laying event followed by a detection of change point in the foraging rate. 30

4.7 Enlarged view of efficiency of*Method* α. The time in seconds represents the difference between a pheromone laying event followed by a detection of change point in the foraging rate. 30

4.8 Efficiency chart for Change Point Detection Methods On one large pile of 256 seeds. This result is generated from the simulated data of all parameter environment, configured for *P. Rugosus* 31

4.9 Efficiency chart for Change Point Detection Methods On one large pile of 256 seeds. This result is generated from the simulated data of all parameter environment, configured for *P. Rugosus* 31

List of Tables

2.1 Seven parameters and their initialization, that characterizes Central Place Foraging Algorithm 5

3.1 Environmental Setup of simulation for three species 11

3.2 Initialization of seven parameters of CPFA for three different environments 14

List of Algorithms

2.1 Genetic Algorithm at a glance. 8
3.1 Pseudo code for calculating cumulative sum. 19
3.2 Pseudocode for Binary Segmented Mean Cumulative Sum 22

Glossary

a_{lm}	Taylor series coefficients, where $l, m = \{0..2\}$
$A_{\mathbf{p}}$	Complex-valued scalar denoting the amplitude and phase.
A^T	Transpose of some relativity matrix.

Chapter 1

Introduction

1.1 Overview

The classic approach to proving a theorem is some really difficult mathematics. For the theory of relativity, I asked grandpa Al exactly how he proved it. He gave me a few hints, including some stuff about rest mass and big electro-motive force. I think he is really smart.

1.2 Conclusions

I conclude that this is a really short thesis.

Chapter 2

Background Study

Ants are social insect. These small tiny creatures have survived major extinction level events in the world. They have been evolved millions of years to survive in the worlds. Their strategies to find resources for their survival are fascinating. We ran several experiments on desert harvester ants at the field to observe how they forage. We have selected three different species of harvester Ants to observe their foraging strategies. Those ants are *P. Rugosus*, *P. Desertorum* and *P. Maricopa*. Our goal was to figure out how they find resources from an environment and what is the effect of the distribution of information on their foraging. So we ran experiments on ants. Seeds in the fields are distributed in a donut shape ring. The area of the food distribution is scaled with the colony size of ants. For example, Desertorum was the smallest in colony size (77 ± 296), so the donut ring radius of food was 1.5 to 3 meters, Rugosus has a colony size of 1712 ± 174. So the radius of food distribution for Rugosus was in 5-10 meters. The seeds are organized in a power law distribution around the nest.

2.1 Power Law

In power law distribution of food, seeds are distributed into multiple piles of different pile size. For example, for power rank 5 of power law distribution total number of seeds will be 1024. And it is divided into 4 types of 256 seeds in each type. One large pile of 256 seeds are placed all together in a certain position. Next 256 seeds are divided into 4 equal sizes of 64 seeds and placed around the nest. Next 256 seeds are equally divided into 16 piles of 16 seeds and placed around the nest inside the ring. Rest of the seeds are distributed uniformly around the nest.

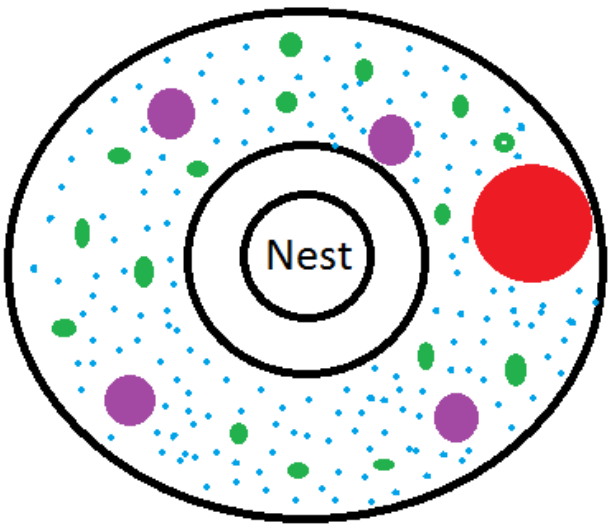


Figure 2.1: Distribution of Seeds in the field experiment for power law distribution with power rank 5. Red Pile indicates one large pile of 256 seeds. 4 purple piles represent 4 large piles of 64 seeds, Green color represents 16 piles of 16 seeds and blue seeds are 256 random seeds

2.2 CPFA

After the distribution of foods across the nest we observe their collection of seeds. It is observed that ants need longer time to discover the large pile with 256 seeds then the piles with small amount of seeds. But once the seeds are discovered, they started recruiting from the piles. Studies showed that information is transferred among the ants for larger piles with more seeds. We have plotted the collection of seeds from the field experiments and discovered ants walk randomly to collect seeds. Once they get the seed they bring it back to the nest. But if they discover a large food source they share the information with others to recruit from the food source. Once the information is shared more ants come into recruitments and while collecting this food they also share information with other ants. As soon as ants start recruiting from the food source, their foraging rate goes up. But the food source starts losing the number of seeds. As the number of seeds starts decreasing from the food source the recruitment of amount of ants also starts decreasing. Based on this behavior, an algorithm was proposed to simulate the behavior of ants. It is called Central Place Foraging Algorithm(CPFA). CPFA is an agent-based model where agents are programmed to follow ants strategy to collect seeds. In CPFA Ants follows three methods to collect seeds.

- **Random Walk:** Ants starts walking randomly from the nest. If they find any food they bring it back to nest
- **Use of Internal Memory:** They remember the last position where the food was found and can return to that place for further search of food. This is method is called site fidelity.
- **Use of Pheromone:** They lay a pheromone trail from the food source to the nest so that other ants can follow that trail to collect food from that source.

Initially, a search location is selected for each of the ants. Then ants start traveling to the search site. After reaching the search site, they perform either uninformed random walk to a random location or informed random walk to a known location based on site fidelity or pheromone. If no resource is found, then they return to the nest. If they find any resource they start to sense the local resource density. Based on the local resource density they decide whether to use site fidelity in future or to lay pheromone. After sensing the resource density, the return to the nest with the seed. While foraging whether the agents will use any of three strategies is governed by seven parameters. These are seven parameters for CPFA Parameters

Parameters	**Initialization Functions**
Probability of Switching to Searching	U(0,1)
Probability of Returning to Nest	U(0,1)
Uniform Search Variation	(0, 4 PI)
Rate of Informed Searched Decay	E(20,0)
Rate of Site Fidelity	E(20,0)
Rate of Laying Pheromone	E(20,0)
Rate of Pheromone Decay	E(20,0)

Table 2.1: Seven parameters and their initialization, that characterizes Central Place Foraging Algorithm

that are following a uniform distribution, higher the value of the parameter higher the probability of that event. For example, "probability of switching to searching" follows a uniform distribution. Higher the value of this parameter, higher the probability that the ant will switch to search. For the exponential distribution, higher the value of the parameter, lower the chance of using that feature. For example, if "Rate of Laying Pheromone" is zero, then it means that there is a higher chance of laying the pheromone. On the other hand, a value close to 20 means that chance of using the pheromone is very low. The Probability of switching to searching determines the chance of ants to switch to search for resources. When ants were not primed by site fidelity or pheromone information, they select a random location to visit. And search

for resources in the pre-determined location. If they cant find any food, they select another new location. The higher the value of switching to the searching parameter, the more they search. The Probability of returning to nest defines the chances of returning to nest for unsuccessful foraging trip. Ants determine the position of a search location by either using site fidelity, or pheromone or randomly. When the travel to a particular location they look for resources. If they can't find any resource, they select a new location to explore and travels to that place. They keep selecting a new place to explore for a certain period of time. The parameter "probability of returning to nest comes into play in this scenario. This parameter actually decides how long the ant will keep searching new places before returning to the nest. Higher the value of the parameter, higher the ant will explore the places for resources. The Rate of site fidelity comes into play when the ants use site fidelity to go to an informed location where the food already exists. Lower the value of this parameter, higher the chances of following the site fidelity. The Rate of laying pheromone is the probability of laying pheromone while returning to nest from a source location. This depends on sensing of local resource density. When the ants collect seeds from a location, they sense the density of resources in nearby areas. Lower the value of probability of laying pheromone, higher the chances of laying pheromone. The Probability of pheromone decay determines how fast the pheromone will evaporate. When the value of the parameter is low pheromone stays in the ground for longer time. And act as vice versa.

2.3 Genetic Algorithm

Genetic Algorithms are metaheuristic search algorithm inspired from natural selection and evolutionary genetics. As such they represent intelligent exploitation of a random search used to solve optimization problems. The basic techniques of GAs

are designed to simulate processes in natural systems necessary for evolution. The evolution usually starts from a population of randomly generated individuals strings that are analogous to the chromosome that we see in our DNA, and is an iterative process, with the population in each iteration called a generation. GAs simulate the survival of the fittest among individuals over the consecutive generation for solving a problem. Each individual represents a point in a search space and a possible solution. The individuals in the population are then made to go through a process of evolution. GA proceeds through the solution domain by evaluating the fitness function. The whole process of evolution is divided into three major section- Selection, Crossover & Mutation.

2.3.1 Selection

In selection, the individuals with better fitness are selected. Fitness function or objective function is the way to evaluate how close the genome is, to a better solution. Once individuals are selected with better fitness, crossover and mutation is applied over them.

2.3.2 Crossover

A crossover site along the bit strings is randomly chosen. The values of the two strings are exchanged up to this point. The two new offspring created from this crossover are put into the next generation of the population.

2.3.3 Mutation

With some low probability, a portion of the new individuals will have some of their bits flipped. Its purpose is to maintain diversity within the population and inhibit

premature convergence. Mutation alone induces a random walk through the search space. Mutation and selection create a parallel, noise-tolerant, hill-climbing algorithms. The process is followed until a common termination condition is reached. This termination condition can be either

- A solution which fulfills minimum criteria
- Evolution reaching a desired number of generation
- Reaching Maximum Fitness
- Reaching the maximum computation time
- Convergence of the solution Or
- A combination of any of this.

In Short, the GA Algorithm can be defined as bellow

Algorithm 2.1 Genetic Algorithm at a glance.

1: Initialize the population randomly
2: Determine the fitness of the first generation
3: **while** Desired solution is obtaind **do**
4: Select elite population with the best fitness
5: Create a new population by crossover and mutation among the elite population
6: Evaluate fitness of the population
7: **end while**

Although GA is very effective in finding the solution in a large special domain, it has some drawbacks. If the length of the chromosome increases, the solution

space may increase exponentially, which may increase the time to reach the solution domain. Also, GA mutation and crossover in an undesired region can produce useless genomes which can push the solution to some local maxima. Besides evaluating repeated genome in the same generation can increase the evaluation time. GAs also cant solve efficiently where the fitness function measure is a Boolean value. Evolving the GA fitness function in first come-first serve manner can increase the bottleneck of the problem.

Chapter 3

Method

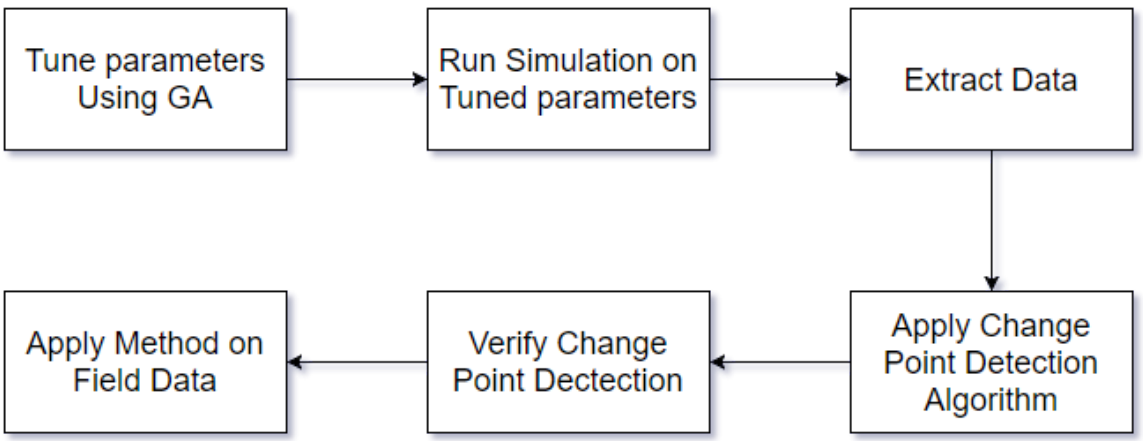


Figure 3.1: Steps of Change Point Analysis

3.1 Setting Simulation Environment

We tried to setup the simulation environment as like as the field experiment environment. So we distributed the resources in Donut Shapes. We had three different setups for three different species of ants. For *Rugosus* the distribution of the re-

Species	**Number of Seeds**	**Radius of Seed Distribution**
P. Rugosus	1024	5-10 meter
P. Maricopa	128	1-3 meter
P. Desertorum	128	1-3 meter

Table 3.1: Environmental Setup of simulation for three species

sources was in power law and the power rank was 5. Total 1024 seeds are distributed in 4 different densities. 256 seeds are piled up in one single pile. There were 4 piles of 64 seeds in each. 16 piles had 16 seeds each and rest of the 256 seeds were distributed uniformly among the nest in the donut shape ring. Although in the field experiment, the area of the donut shape was proportional to the colony size. In here we tried to keep the area of the donut shape ring constant. The inner radius of the ring was 5 meter and the outside radius was 10 meter. The total duration of each experiment was 90 minutes (We collected data from field experiments for 90 minutes only). The total arena size was 20×20 meter. We kept the arena into this size and bounded the agents to search in this arena. The setup is varied for *Maricopa* and *Desertorum*. The following table represents the environmental setup of simulations for *P. Rugosus*, *P. Maricopa* and *P. Desertorum*.

3.2 Tuning Parameters using Genetic Algorithm

To analyze the data, we have tuned the parameters of CPFA. As stated above in the background study, enormous amount of parameters for CPFA can be used to evaluate the fitness. But we have used the genetic algorithm to achieve the optimum set of parameters. We have divided the simulations into three categories to tune the GA for three different environments.

1. **Pheromone Only Parameters:** For this type we have restricted the use of

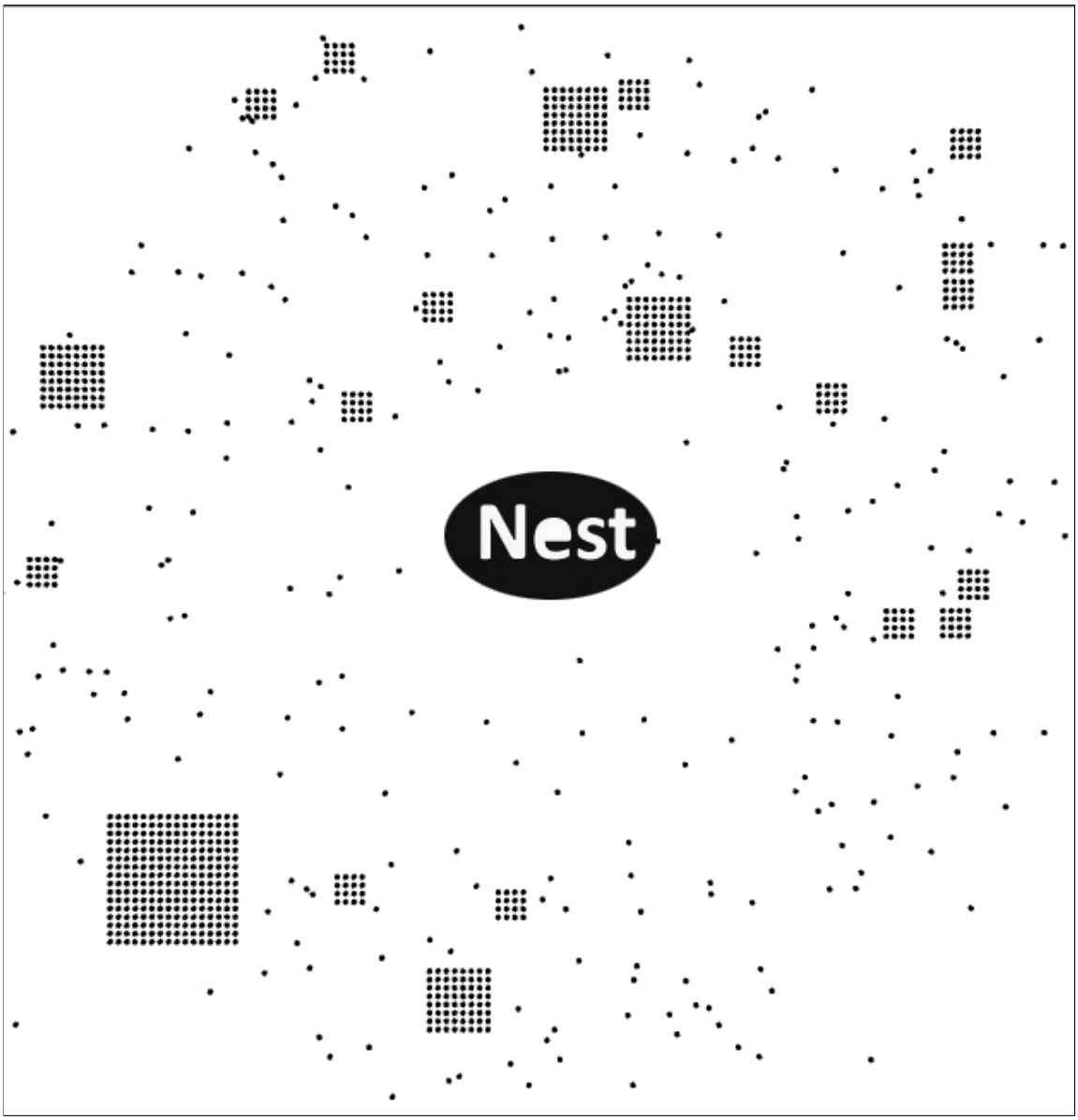


Figure 3.2: Example setup of a simulation environment for *P. Rugosus* with total 1024 seeds. The setup showing three different types of piles. One large pile of 256 seeds, Four piles of 64 seeds and sixteen piles of 16 seeds. 256 random seeds are distributed uniformly inside the ring.

site fidelity. Which means that the probability of using site fidelity is 20. Rest of the parameters evolved using the GA.

2. **Site fidelity Only Parameters:** In this types of experiment we restricted the use of pheromone. For this case ants can only use site fidelity and random walk to collect resources

3. **Using Both site fidelity and pheromone:** This environment represents the actual field experiment condition. Here agents use both site fidelity and pheromone along with the random walk.

For each type of environment, we have tuned the parameters to obtain maximum fitness using Genetic Algorithm. Initially, we have created the one hundred set of parameters. And then evaluated their fitness. The best genome is selected for crossover and mutation for next generation. We continued this process until we obtain the best fitness genome or parameter set. The GA is terminated either when the parameters are converged, or it reaches to generation 50. For each parameter set in a population, fitness is tested for four different random seeds. The Random seed is the variable which controls the variables of a simulation. After evaluation of each random seed for one parameter set, we have calculated the average seed collection to define the fitness of that particular set of parameters. These random seeds are basically numbers which are fixed for each generation. For each generation, we have selected four different numbers for the random seeds and then evaluated all the parameters for those values. For GA we can confine particular parameters to a certain range of values by setting this in evolver.cpp file. For Example, for "Pheromone Only" Environment we restricted the values of site fidelity to $(20, 20)$ where each value inside the parenthesis represents upper bound and lower bound. To calculate the fitness for each parameter set it takes evaluating the fitness function for four times due to four different random seeds, which means for each generation it needs evaluating the objective function for 400 times. So over 50 generations, it will need 2000 evaluations of the objective function. This can take a lot of time if we perform the evaluation sequentially. To remove this bottleneck, we have used

multi-threading of genetic algorithm by evaluating multiple objective functions simultaneously. We have used GA Lib Genetic Algorithm Package. The software for this work used the GAlib genetic algorithm package, written by Matthew Wall at the Massachusetts Institute of Technology. The MPI version was written by Andrew Rasmussen https://github.com/andyras/GAlib-mpi/blob/master/LICENSE who modified the code from https://github.com/BORJA/GAlib-mpi. The evolver.cpp file is used to initialize the GA parameters and pipe the parameters to the GA Lib. Detail of initial parameter settings of Genetic Algorithm for three different setup is given in the following table.

CPFA Parameters	**Pheromone Only**	**Sitefidelity Only**	**All Parameters**
Probability of Switching to Searching	U(0,1)	U(0,1)	U(0,1)
Probability of Returning to Nest	U(0,1)	U(0,1)	U(0,1)
Uniform Search Variation	(0, 4 PI)	(0, 4 PI)	(0, 4 PI)
Rate of Informed Searched Decay	E(20,0)	E(20,0)	E(20,0)
Rate of Site Fidelity	E(20,20)	E(20,0)	E(20,0)
Rate of Laying Pheromone	E(20,0)	E(20,20)	E(20,0)
Rate of Pheromone Decay	E(20,0)	E(20,20)	E(20,0)

Table 3.2: Initialization of seven parameters of CPFA for three different environments

3.3 Generating Data Set for Analysis

Once the parameters are tuned for three different environments, we have generated the data for our analysis using these parameter sets. For each of the experiment, we have extracted pick up and drop off time for each seed, location of each seed in the arena. Pick up time represents the time when the seed is picked by the ant from the location of the seed and drop time is when it is dropped off at the nest. We have tagged each ant with distinct ID. For each seed, we also have extracted which ant has collected that seed. Also, we have tracked when the pheromone is laid, and followed, and when the site fidelity is followed. We have assigned distinct ID number to each pile so that when a pheromone trail is laid we can track which pile the trail is coming from. For each type of environment, we have simulated 500 experiments and generated data mentioned above. We varied the value of random seed for each experiment while keeping the CPFA parameters constant for a particular environment. We also varied the position of seeds for each experiment. Each experiment was performed for 90 minutes. While generating the data for "Pheromone only parameters", We did not extract any site fidelity data, because we tuned all the parameters not to use site fidelity data. Similarly, for "site fidelity only" experiments we did not extract any pheromone data as there was no pheromone. We have collected both site fidelity data and pheromone data when we have used both methods together for collecting resources.

3.4 Analyzing The Foraging Data

After generating all the data from the simulation we have tried to observe how the ants collect seeds from different food distributions. We observed that it takes some time for them to discover the larger piles. But once they discover it, they start

to collect seeds from those piles. And they use site fidelity and pheromone for this recruitment. Once they start collecting this seeds we see an increase in their foraging rate. So we tried to detect those changes in their foraging rate by applying the change point detection algorithm.

3.4.1 Creating Timeline for each type of distribution

We have studied each experiment separately to analyze the change in their foraging rate. For each experiment, we studied foraging rate for each type of pile individually. To study foraging rate for each pile we have created a timeline for each type of distribution. For example, we have divided the whole time of the experiments into

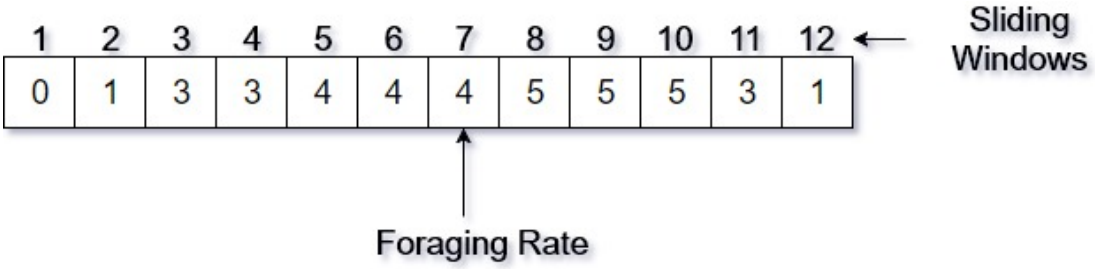


Figure 3.3: An example of a timeline for a distribution where numbers at the top represent the sliding window number. Values in the boxes are the rate of collection of seeds per window.

sliding windows. Where each sliding window represents the rate of collection of seeds for that time. The length of the sliding window is 60 seconds for *P. Rugosus*. And then we slided the window by 10 seconds so that we can get the rate of collecting seeds by that window. So if the experiment is for 90 minutes (5400 seconds), we kept the length of the sliding windows for 60 seconds and slided it by 10 seconds, we get total 540 sets of data where we calculated their foraging rate for a particular pile. Once we have created the timeline for each experiment for a particular distribution of seeds we used Change point detection algorithm to detect the change in the rate

of collection of seeds. Another method we have created the timeline is by taking into consideration the change in the rate of foraging. In this method for creating the timeline instead of foraging rate, we take into consideration the change in foraging rate.

The length of the window and the sliding amount for the window is different for each species. We have systematically varied this two parameters to fine tune the change point detection algorithms. Values for each of the parameters will be discussed further in the result section.

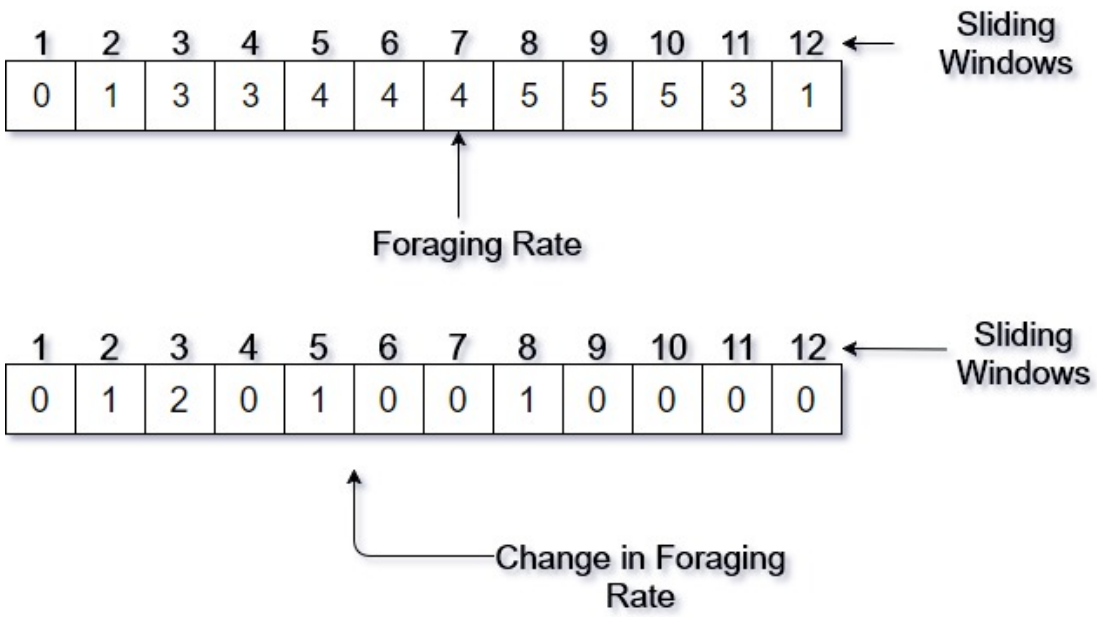


Figure 3.4: An example of timeline and change in foraging rate. The change in foraging rate is calculated by measuring the difference between the timeline windows

3.5 Change Point Detection Algorithm

The change point detection algorithm is divided into two parts. First part is the adding rate of collecting seeds to calculate the cumulative sum and detrend for smoothing. And the second part is applying the change point detection algorithm. We have used binary segmented cumulative sum method to determine the change points.

3.5.1 Calculating the Cumulative Sum

The calculation of cumulative sum is basically adding the foraging rate in each window. The following figure and algorithm show how the cumulative sum is calculated.

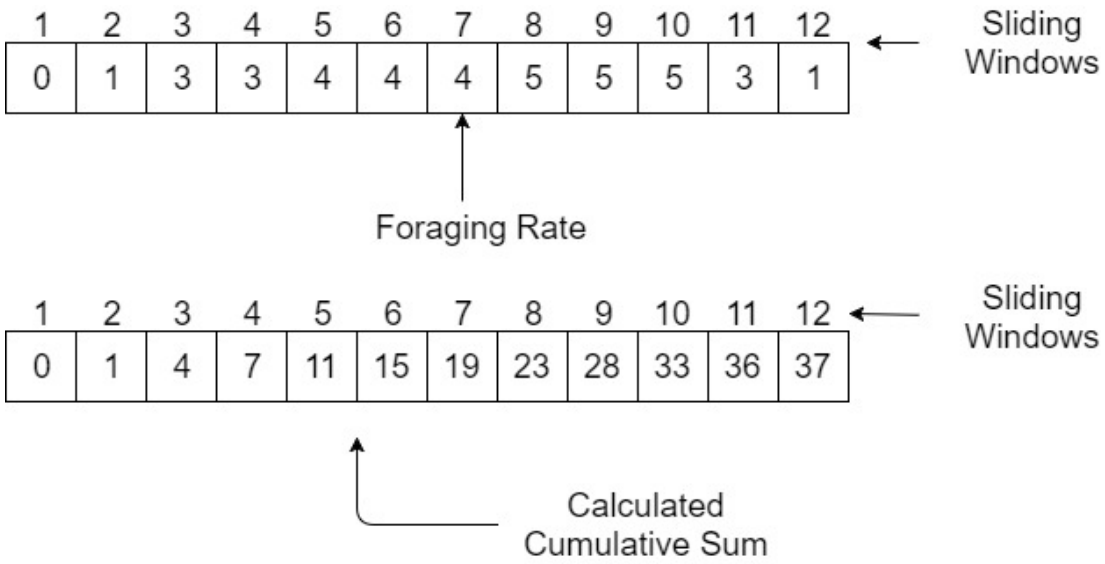


Figure 3.5: This figure demonstrates how the cumulative sum is calculated from the timeline of foraging rate.

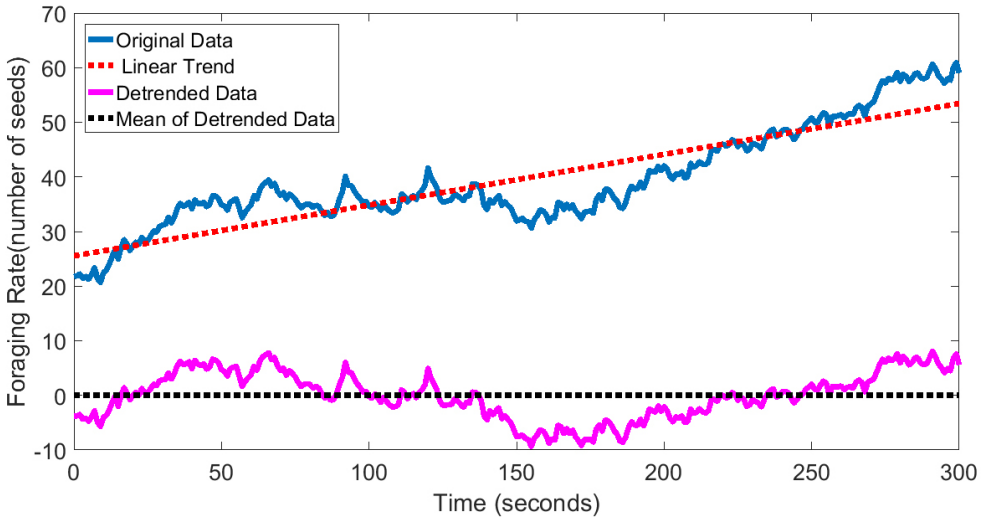
Algorithm 3.1 Pseudo code for calculating cumulative sum.

```
1: Sum=0
2: for i=1:Number of Sliding Window do
3:     Sum= Sum + window(i)
4:     CumulativeSum(i)=Sum
5: end for
```

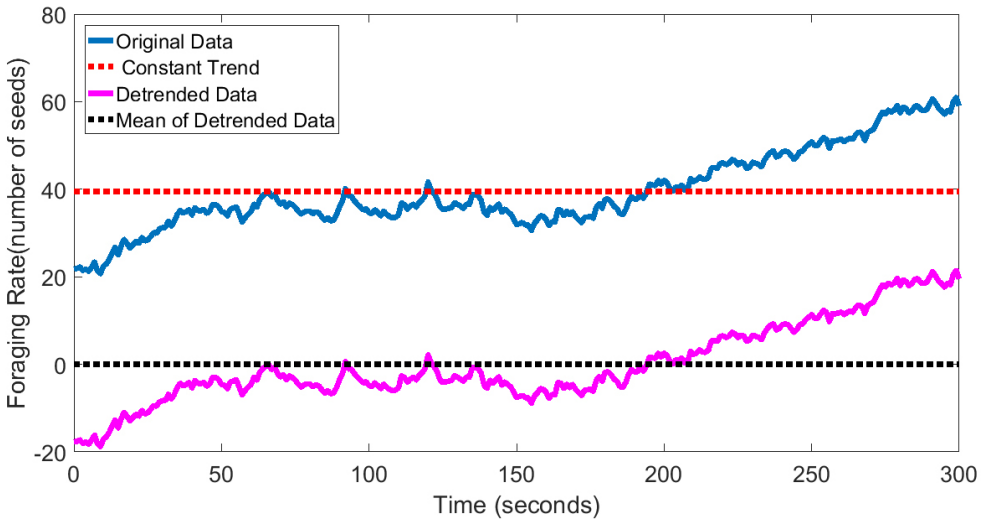
3.5.2 Detrending

In time series trend means a lazy and gradual change in properties over the whole interval of the event. It is sometimes implicitly defined as a long-term change in the mean. It can also be referred as a change in some statistical properties. Usually, periodic and seasonal components and abrupt fluctuations and other parts were studied separately. Modern analysis techniques frequently treat the series without such routine decomposition, it is still required to consider the trend separately. The removal of a trend in a statistical or mathematical operation of time series is called detrending. It is often applied to remove features which are obsolete or unimportant. In time series analysis, detrending is also used in preprocessing step to prepare data set for further analysis. There are several methods of detrending. Linear trends in mean can be truncated by subtracting a least-square-fit straight line. Different procedures are used for more complicated trend. For example, the cubic smoothing spline is commonly used in dendrochronology to fit and remove ring-width trend that might not be linear, or not even monotonically increasing or decreasing over time. It is important to understand the effect of detrending on spectral properties of time series before trying to remove the trend from the time series. Before applying the change point detection algorithm, we have applied detrending algorithm to remove the trend from the time series. We used linear detrending and constant detrending to observe the effect of detrending in our time series data. Linear detrending removes

the linear trend from the data where constant detrending removes the mean from the data. Here is an example, which shows how linear and constant detrending affects the time series.



(a) Linear Detrending



(b) Constant Detrending

Figure 3.6: An example of applying linear and constant detrending on the cumulative sum of a timeline from one simulated CPFA experiment.

3.6 Binary-Segmented Cumulative Sum

Binary Segmentation is one of the most established search method used for detecting the change point. This method extends any single change point method to multiple change points by iteratively repeating the method on different subsets of the sequence. To perform binary segmentation, we rst apply the chosen single change point detection method to the entire data set, if no change point is found then we are done. If a change point is detected, call this , then the data is split into two segments, timeline$[1 : t]$ and timeline$[t + 1 : n]$. We then apply the single change point method to the two segments and repeat iteratively. We stop when no more change points are detected.

Binary segmentation is a very fast algorithm with complexity $O(n \log n)$ to detect the changes. But the major disadvantage of its computational correctness is that it gives us only an approximation of changes. It is not guaranteed that the binary segmentation method will find us the optimum solution. Also due to iterative nature of this algorithm, it may not detect changes in small changes. Which is why to verify the how well this method is performing, we have verified the results with the simulated data. The pseudo code for the binary segmentation algorithm is given in Algorithm 3.2.

3.7 Verification of Change Points

As we have simulated data, and we know when the pheromones and site fidelity are used in simulations we can certainly verify how efficient is our change points detection algorithms are. So to check how efficient is our algorithms to detect change points, we divided the detection of change points into 4 categories.

- **Catagory A:** Change point detection within 10 seconds of pheromone laying

Algorithm 3.2 Pseudocode for Binary Segmented Mean Cumulative Sum

Input:A set of data of the form ($value_1$,$value_2$,$value_3$...)
    A test statistic $\tau(.)$
    An estimator of the changepoint position $\tau(.)$
    A rejection threshold β
Initialize: Let $C = \phi$, and $S = [1 : n]$
while $S \neq \phi$ **do**
    Choose an element of S
    Denote this element as $[s, t]$
    if $\tau(ys : t) < \beta$ **then**
        remove$[s, t] from S$
    end if
    if $\tau(ys : t) \geq \beta$ **then**
        remove$[s, t] from S$
        calculate $r = \tau(ys : t) + s - 1$,
        add r to C
        if $r \neq s$ **then**
            add $[s, r]$ to S
        end if
        if $r = t - 1$ **then**
            $[r + 1, t]$ to S
        end if
    end if
end while

events,

- **Catagory B:** Change point detections within 11-300 seconds of pheromone laying events,

- **Catagory C:** Change point detections after more than 300 seconds of pheromone laying events and
- **Catagory D:** Change point detected but no pheromone laying events has happened.

3.8 Applying the best method on Field Data

We applied change point detection on 4 different types of the data set. This lead us to evaluate the performance of change point detection algorithm for four different methods. After validating four different methods that we have applied to simulation

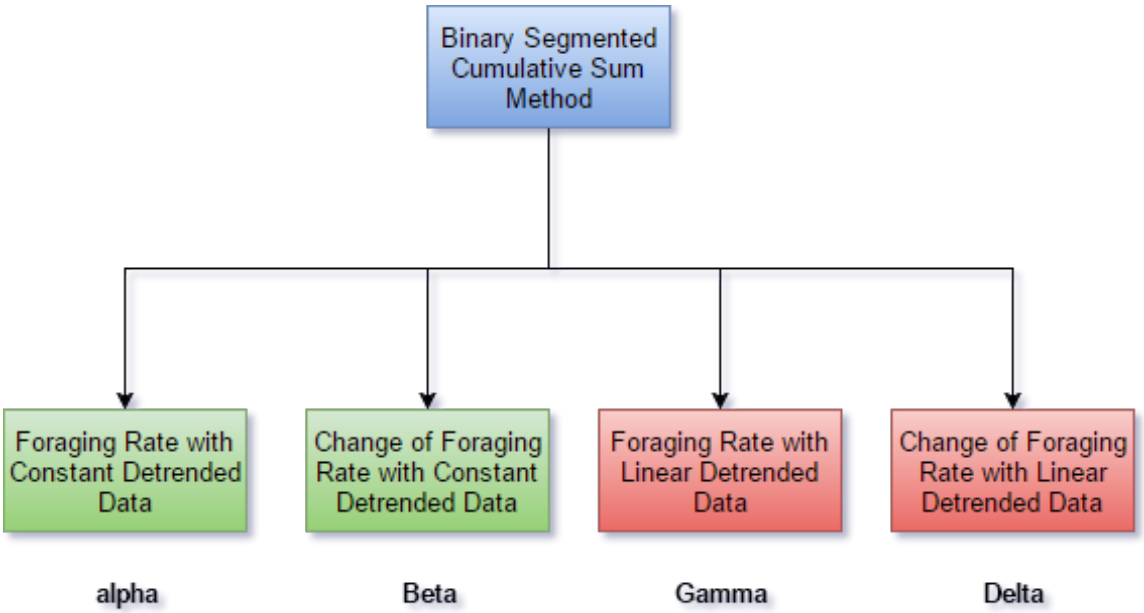


Figure 3.7: Four Different Change point Detection Method

data. We select the best method for them based on the performance on four categories stated above. And apply the best method in our field data.

Chapter 4

Results

4.1 Results from Simulation

We configured the simulation settings as close as we can with the experimental setup of *P. Rugosus*, *P. Maricopa* and *P. Desertorum*. Then we have compared the result of method *alpha*, *beta*, *gamma* and *delta*. based on the time between a pheromone laying event followed by the detection of change point.

Figure 4.1 compares the results of four different methods on a pheromone only simulated environment for *P. Rugosus*. We can see that, *Method* α detects change points which is closer to the pheromone laying event. The result of *Method* β is closer to the *Method* α, but Method γ and Method δ doesn't perform well in this environmental settings.

We have illustrated the result of *Method alpha* in figure 4.2. The average difference of pheromone laying event followed by a change point detection for *Method* α is 47.5, 60 and 81.5 for one pile, four large piles and sixteen piles. It is also observed that, the range of time difference between a pheromone laying event followed by a change point detection event is also significantly reduced. We observe similar phenomenon

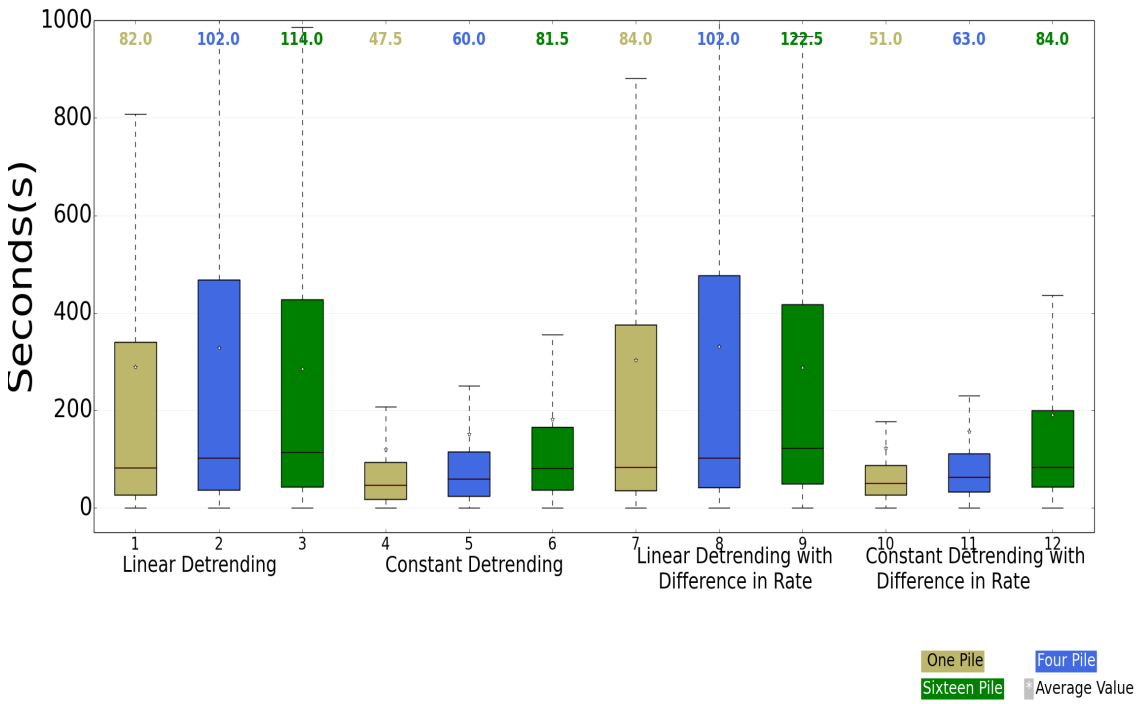


Figure 4.1: Comparison of Change Point Detection Methods without Outliers for Pheromone Only Parameters. The time in seconds represents the difference between a pheromone laying event followed by a detection of change point in the foraging rate.

for *P. Maricopa* and *P. Desertorum*. The box plots of comparison for these two species are provided in the Appendices. The outliers are removed from the figure to make the figure more visible. The detailed figure with the outliers are provided in the appendices.

We have also evaluated four different methods based on the four categories mentioned in section 3.7. As we can see from figure 4.3 in constant detrending method, 15% of the time change points are detected within 10 seconds of laying pheromone, whereas, in linear detrending method, it is only 9%. Constant detrending method also has significant improvement over linear detrending method in *Catagory two*. 74% of the

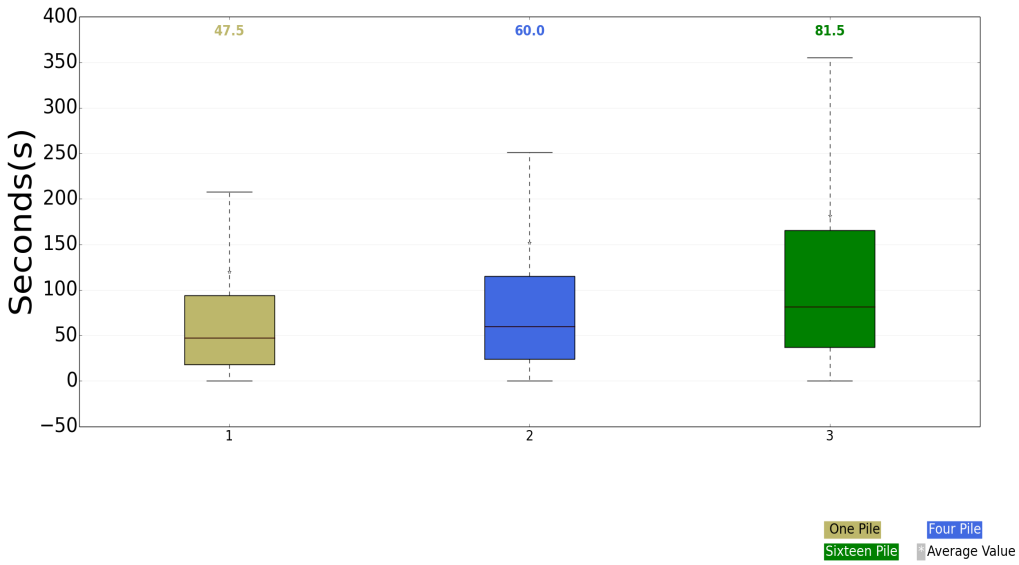


Figure 4.2: Enlarged view of efficiency of*Method* α. The time in seconds represents the difference between a pheromone laying event followed by a detection of change point in the foraging rate.

time, changes points are detected with eleven to three hundred seconds of laying pheromone while using constant detrending method, on the other hand, using linear detrending method it is only 50%. In *catagory three* the error is 6% for *method* α, whereas, it is 21% for *method* γ. Performance of *Method* α and *Method* β is similar. if we consider categories one and two. But *Method* α does significantly better than *Method* β in detecting change points for four pile and sixteen pile in the simulated environment. From figure 4.4 and figure 4.5 we can observe that, the performance of *Method* α combined in category one and two is better than performance of *Method* β combined in these two categories Also from figure 4.4 and 4.5 we can see that *Method* α does better in category three than *Method* β. If we look at the performance of the four methods over these categories in phermone only simulated environments, for *P. Maricopa* and *P. Desertorum* we observe similar pattern of performances. The results for *P. Maricopa* and *P. Desertorum* provided in the appendices Figure 4.6

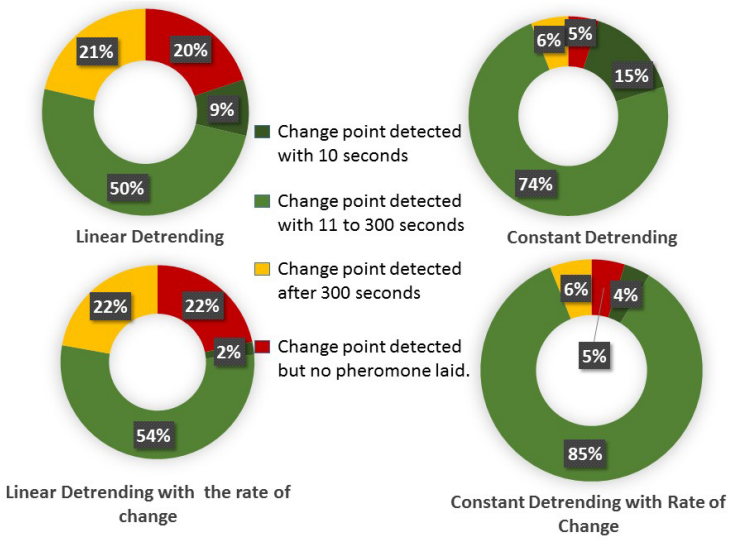


Figure 4.3: Efficiency chart for Change Point Detection Methods On a large pile of 256 seeds. This result is generated from the simulated data of pheromone only environment, configured for *P. Rugosus*

shows the comparison of four different methods on a simulated environment with all parameters for *P. Rugosus*. From this figure it is clearly visible that *Method* α does better in detecting the change points than any other methods. As a matter of fact, Method α does better in all parameter environment than the pheromone only environment. If we take a closer look at the performance of *Method* α we see that the average time to detect a change point after a pheromone laying event is 29 seconds, 40 seconds and 48 seconds for one pile, four pile and sixteen pile respectively. We took a look at the efficiency chart of all four methods for all parameter environmental settings in figure 4.8. This figure is generated from the performance evaluation of change point detection methods on one large pile of 256 seeds for *P. Rugosus*. We see that, *method* α*]* and *method* β does significantly better than the other two methods. But in category three and four *method* α does better than *method* β. We observe that, the error is 0% for *method* α, whereas it is 1% for *method* β. *method* α also surpluses *method* β in the combined result of category 1 and 2. We observer,

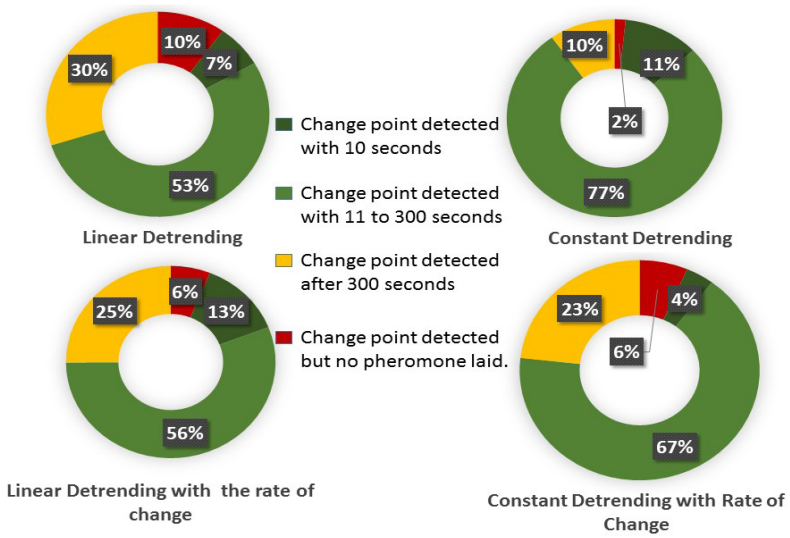


Figure 4.4: Efficiency chart for Change Point Detection Methods On four medium pile of 64 seeds. This result is generated from the simulated data of pheromone only environment, configured for *P. Rugosus*

the similar pattern of performances for four piles and sixteen piles. The performance rating is similar for *P. Maricopa* and *P. Desertorum*. Further details of the results are provided in the appendices. Since we have seen that, *method* α does better in previous two environmental settings, for the sitefidelity only parameters, we compare the result of *method* α and *method* γ for the four categories. The result in figure 4.9 is produced from the site fidelity only simulated environment. Since, no pheromone is laid in this environment we calculate time difference between change points detection and the event of following sitefidelity by an agent. Although our none of the methods are as efficient as they were in the previous two environmental settings, we compared two methods. We observe that, *method* α does better in all categories than *method* β. We observe similar performance pattern for four medium piles and sixteen small piles. Simulated environment of *P. Desertorum* and*P. Maricopa* gives us similar pattern of results.

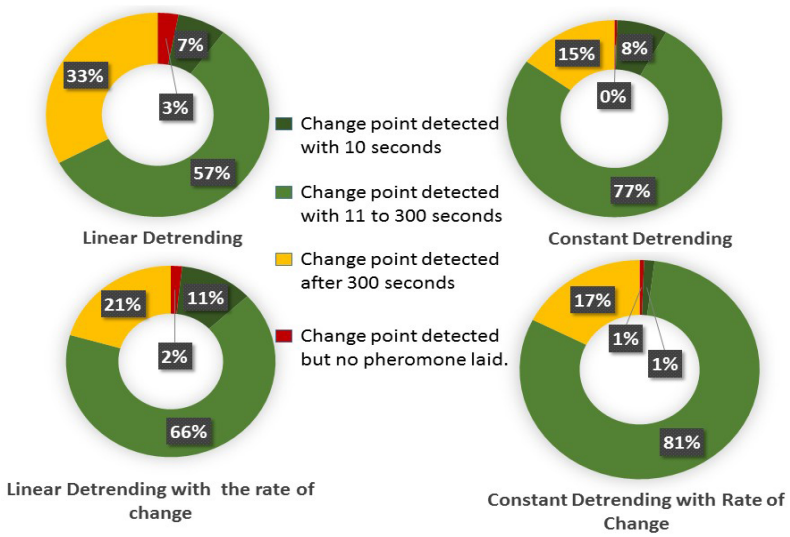


Figure 4.5: Efficiency chart for Change Point Detection Methods On sixteen small pile of 16 seeds. This result is generated from the simulated data of pheromone only environment, configured for *P. Rugosus*

4.2 Parameter setup for Change Point Detection Methods

As we know that our change point detection algorithms depends on the values of 3 different parameters, sliding window size, sliding amount and number of change points, we have systematically varied this parameters to obtain the optimum efficiencies for all three methods. For tuning the sliding window size, we have varied it from 40-120 seconds with an interval of 10 seconds. We have varied the sliding amount from 10-30 seconds with an interval of 10 seconds. For each of the species we have observed the simulations visually to track down, how many times ants discover a particular type of pile and noted that number.

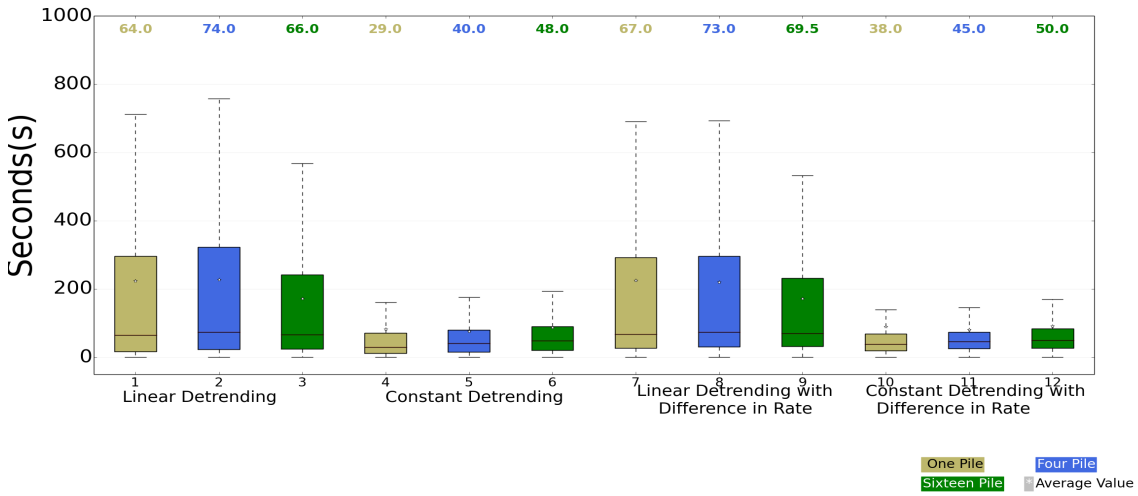


Figure 4.6: Comparison of Change Point Detection Methods without Outliers for All Parameters. The time in seconds represents the difference between a pheromone laying event followed by a detection of change point in the foraging rate.

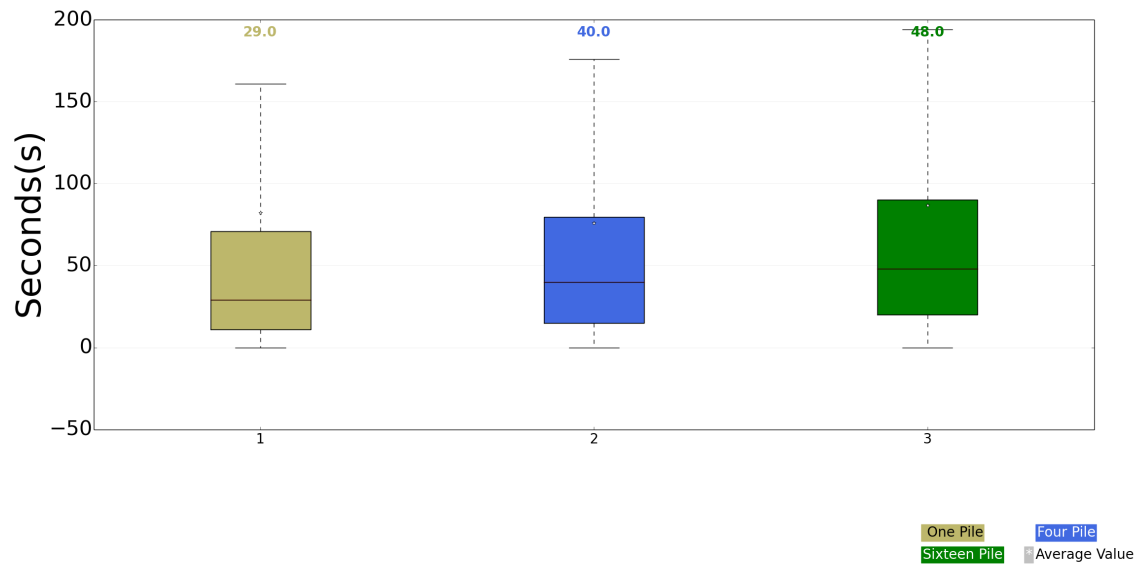


Figure 4.7: Enlarged view of efficiency of*Method* α. The time in seconds represents the difference between a pheromone laying event followed by a detection of change point in the foraging rate.

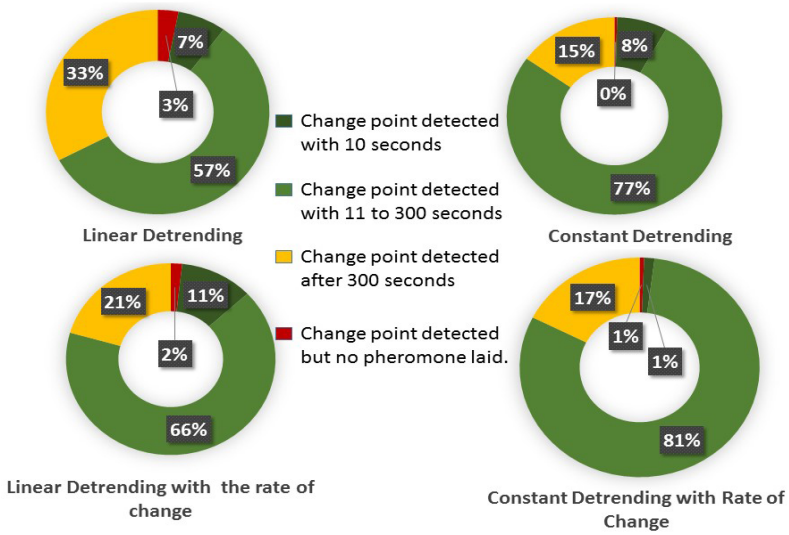


Figure 4.8: Efficiency chart for Change Point Detection Methods On one large pile of 256 seeds. This result is generated from the simulated data of all parameter environment, configured for *P. Rugosus*

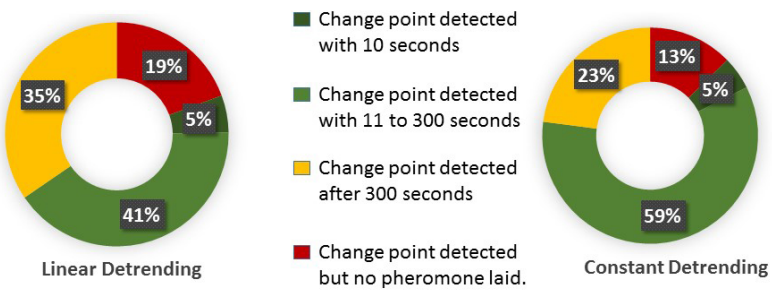


Figure 4.9: Efficiency chart for Change Point Detection Methods On one large pile of 256 seeds. This result is generated from the simulated data of all parameter environment, configured for *P. Rugosus*

Chapter 5

Conclusion

5.1 Overview

The classic approach to proving a theorem is some really difficult mathematics. For the theory of relativity, I asked grandpa Al exactly how he proved it. He gave me a few hints, including some stuff about rest mass and big electro-motive force. I think he is really smart.

5.2 Conclusions

I conclude that this is a really short thesis.

Chapter 6

Appendicies

6.1 Overview

The classic approach to proving a theorem is some really difficult mathematics. For the theory of relativity, I asked grandpa Al exactly how he proved it. He gave me a few hints, including some stuff about rest mass and big electro-motive force. I think he is really smart.

6.2 Conclusions

I conclude that this is a really short thesis.