REDUCING PASSENGER TRAVEL FUEL COST AND EMISSION: A CASE STUDY OF TANZANIA COMMUTER TRAINS

Joachim Julius Mwambeleko

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Engineering in Electrical Engineering

Suranaree University of Technology

Academic Year 2015

การลดค่าเชื้อเพลิงในการเดินทาง และลดการปล่อยมลพิษ: กรณีศึกษาของรถไฟในประเทศแทนซาเนีย

โจอะคิม จูเลียส มวมเบเลโก

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า

มหาวิทยาลัยเทคโนโลยีสุรนารี

ปีการศึกษา 2558

REDUCING PASSENGER TRAVEL FUEL COST AND EMISSION: A CASE STUDY OF TANZANIA COMMUTER TRAINS

Suranaree University of Technology has approved this thesis submitted in partial fulfillment of the requirements for a Master's Degree.

โจอะคิม จูเลียส มวมเบเลโก : การลดค่าเชื้อเพลิงในการเดินทาง และลดการปล่อยมลพิษ: กรณีศึกษาของรถไฟในประเทศแทนซาเนีย (REDUCING PASSENGER TRAVEL FUEL COST AND EMISSION: A CASE STUDY OF TANZANIA COMMUTER TRAINS) อาจารย์ที่ปรึกษา : รองศาสตราจารย์ ดร.ธนัดชัย กุลวรวานิชพงษ์, 261 หน้า

งานวิจัยนี้ศึกษาความเป็นไปได้ในการลดค่าเชื้อเพลิงการลดมลภาวะสำหรับการเดินทางของผู้โดยสารและการลดมลภาวะ โดยใช้รถไฟฟ้าแบบใช้แบตเตอรี่ (BEMUs: Battery Electric Multiple Units) แทนรถไฟดีเซลซึ่งวิ่งในระยะสั้น และใช้รถไฟของประเทศแทนซาเนีย (Tanzania commuter trains) เป็นกรณีศึกษา ความต้องการความรวดเร็วในการเดินทาง ปัจจัยด้านสิ่งแวดล้อม ราคาพลังงาน และการจราจรติดขัดที่เติบโตมากขึ้นเป็นประเด็นที่ละเอียดอ่อนซึ่งสามารถแก้ไขได้ด้วยระบบการขนส่งที่ยั่งยืน และมีประสิทธิภาพ เนื่องด้วยคุณสมบัติที่มีประสิทธิภาพสูง ค่าเชื้อเพลิง และการปล่อยแก๊สมลภาวะต่อผู้โดยสารต่อกิโลเมตรต่ำ รถไฟฟ้าเป็นหนึ่งในตัวเลือกที่ดีที่สุดในปัจจุบันต่อความท้าทายในการช่วยการเดินทางในพื้นที่เมือง อย่างไรก็ตามยังมีการใช้งานรถไฟดีเซลในระบบรางปกติ (non-electrified railway) และยังคงต้องใช้เวลาในการปรับระบบรางปกติให้เป็นระบบรางไฟฟ้าทั้งหมดระบบรางไฟฟ้าถูกนำมาใช้กับระบบรางมากที่สุดในที่ซึ่งมีการจราจรติดขัดมากพอที่จะคุ้มกับค่าใช้จ่ายที่สูงของระบบรางไฟฟ้า

ในการพยายามแก้ไขปัญหาการขนส่งภายในเมือง Dar es Salaam ที่มีการจราจรติดขัด รัฐบาลของประเทศแทนซาเนียเริ่มต้นใช้งานรถไฟดีเซลในเดือนตุลาคม ปี ค.ศ. 2012 หลังจากการปรับปรุงฟื้นฟูหัวรถจักร และขบวนรถซึ่งเคยนำไปใช้งานในพื้นที่ชนบท การให้บริการเดินรถมีเฉพาะในตอนเช้า ตอนเย็น และชั่วโมงเร่งด่วนในวันทำงาน แต่เนื่องด้วยประสิทธิภาพของรถไฟต่ำ โดยเฉพาะอย่างยิ่งในการเดินทางระยะสั้น ที่มีการจอดบ่อยครั้ง สิ้นเปลืองเชื้อเพลิง นำไปสู่ปัญหาค่าใช้จ่ายในการเดินรถที่สูงมากขึ้น

เนื่องด้วยความก้าวหน้าด้านเทคโนโลยีแบตเตอรี่ในปัจจุบัน จึงสามารถใช้ BEMUs แทนรถไฟดีเซลในระยะทางสั้นๆได้ ดังนั้นจึงสามารถลดค่าเชื้อเพลิงในการเดินทางของผู้โดยสาร และการปล่อยแก๊สมลภาวะได้ งานวิจัยนี้ใช้เทคนิคปัญญาประดิษฐ์เพื่อหาความเร็วที่เหมาะสมที่สุด สร้างแบบจำลองของรถไฟให้เป็น BEMU และจำลองผลการเคลื่อนที่โดยใช้ความเร็วที่เหมาะสมที่สุด วิเคราะห์การลดค่าเชื้อเพลิงในการเดินทางของผู้โดยสาร และการปล่อยแก๊สมลภาวะโดยใช้รถไฟของประเทศแทนซาเนียเป็นกรณีศึกษา

เมื่อเทียบกับรถไฟดีเซล TRL (Tanzania railways limited) พบว่า BEMU สามารถลดค่าเชื้อเพลิงในการเดินทางของผู้โดยสาร และการปล่อยแก๊สมลภาวะถึง 87.7% และ 70.89%

และเมื่อเทียบกับรถไฟดีเซล TAZARA (Tanzania and Zambia railway authority) พบว่า BEMU สามารถลดค่าเชื้อเพลิงในการเดินทางของผู้โดยสาร และการปล่อยแก๊สมลภาวะถึง 75.6% และ 42.25% ตามลำดับ ซึ่งจะเห็นว่าสามารถลดได้มากอย่างมีนัยสำคัญ ดังนั้น ผู้ออกนโยบาย เจ้าหน้าที่ที่เกี่ยวข้อง และผู้มีส่วนได้ส่วนเสียอื่นๆ ควรได้รับคำแนะนำเพื่อพิจารณาผลลัพธ์ที่ได้ของงานวิจัยนี้ เมื่อต้องการเสนอหรือร่างนโยบาย และแผนงานต่างๆ สำหรับระบบขนส่งในเมือง

สาขาวิชา<u>วิศวกรรมไฟฟ้า</u>
ปีการศึกษา 2558

ลายมือชื่อนักศึกษา [illegible]
ลายมือชื่ออาจารย์ที่ปรึกษา [illegible]

JOACHIM J. MWAMBELEKO : REDUCING PASSENGER TRAVEL FUEL COST AND EMISSION: A CASE STUDY OF TANZANIA COMMUTER TRAINS : THESIS ADVISOR : ASSOC. PROF. THANATCHAI KULWORAWANICHPONG, Ph.D., 261 PP.

DIESEL TRAINS/BATTERY ELECTRIC MULTIPLE UNITS/OPTIMAL SPEED PROFILE/PASSENGER TRAVELL FUEL COST AND EMISSION

This research studies the possibility of reducing passenger travel fuel cost and emission by replacing diesel commuter trains operating on short distances with Battery Electric Multiple Units (BEMUs), taking a case study of Tanzania commuter trains. Growing mobility demand, environmental concerns, energy prices, and traffic congestion in cities are becoming delicate issues that could be eased by more efficient and sustainable transportation systems. Having high efficiency, low fuel cost and emission levels per passenger kilometer, an electric train is currently one of the best solutions to urban mobility challenges. However, there are still diesel trains operating on non-electrified railway networks and it will take time to electrify the whole network. Electrification is most often applied to railways where the density of traffic movement is sufficient to justify the high fixed costs.

Trying to ease transport challenges within the congested city of Dar es Salaam, the government of Tanzania introduced two diesel commuter trains in October 2012; after the rehabilitation of the locomotives and coaches which were used for up-country journeys. The commuter train service is available only in the hours of the working days. Unfortunately, attributed by trains' low efficiency especially in short journeys with frequent stops, high fuel consumption leading to high operating cost has been a problem.

With the recent advancements in battery technology, BEMUs can replace diesel trains operating on short distances hence, reduce passenger travel fuel cost and emission. This research work therefore, uses artificial intelligence techniques to search for tram's optimal speed profiles, models the tram as a BEMU, simulates its movement based on the optimal speed profiles and, analyses the reduction in passenger travel fuel cost and emission taking a case study of Tanzania commuter trains.

Compared to Tanzania railways limited (TRL) diesel commuter train, the BEMU was found to reduce passenger travel fuel cost and emission by 87.7 %, and 70.89 % respectively. And compared to Tanzania and Zambia railway authority (TAZARA) diesel commuter train, the BEMU was found to reduce passenger travel fuel cost and emission by 75.6 %, and 42.25 % respectively. These are significant reductions; policymakers, city authorities and other stakeholders are therefore advised to consider the results from this research when proposing changes or making policies and plans for city transport systems.

School of Electrical Engineering

Academic Year 2015

Student's Signature [illegible]

Advisor's Signature [illegible]

ACKNOWLEDGEMENTS

The work contained in this thesis has been made possible through the generosity of several people in various ways.

I would like to express my deepest gratitude to my advisor, Assoc. Prof. Dr. Thanatchai Kulworawanichpong for his overly support, enlightening guidance, inspiration and encouragement throughout the course of this study, whilst allowing me a free room to work on my own thoughts.

My sincere appreciation is extended to Assoc. Prof. Dr. Kitti Attakitmongcol and Asst. Prof. Dr. Boonruang Marungsri who served on the thesis committee, for their insightful and invaluable comments upon the success of this work.

It is my pleasure to express my sincere gratitude to all Electrical Engineering students and staff for their love, kindness and support; and to TRL and TAZARA staff for their invaluable cooperation during data collection.

I am very grateful to Suranaree University of Technology, for the scholarship that covered my tuition fee during the period of the study.

I am greatly indebted to my lovely family on whose constant love and support I have relied. In a very special way I thank my lovely mother who untiringly endured all the hard times and difficulties supporting my studies. Finally, I would like to express my gratitude to all whom I have failed to mention here, but have generously supported me.

Joachim Julius Mwambeleko

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

Page

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF TABLES

LIST OF TABLES (Continued)

Table **Page**

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

SYMBOLS AND ABBREVIATIONS

BEMU	=	Battery Electric Multiple Unit
CO_2	=	Carbon Dioxide
DE	=	Differential Evolution
DoD	=	Depth of Discharge
EIA	=	Energy Information Administration
EMU	=	Electric Multiple Unit
ESR	=	Equivalent Series Resistance
EV	=	Electric Vehicle
EWURA	=	Tanzania Energy and Water Utilities Regulatory Authority
GA	=	Genetic Algorithm
HEV	=	Hybrid Electric Vehicle
IPEMU	=	Independently Powered Electric Multiple Unit
kph	=	Kilometer per hour
Li-ion	=	Lithium Ion
LTO	=	Lithium Titanate or Lithium Titanium Oxide
MEM	=	Tanzania Ministry of Energy and Minerals
min	=	Minutes
PbA	=	Lead Acid
PSO	=	Particle Swarm Optimization
SoC	=	State of Charge

SYMBOLS AND ABBREVIATIONS (Continued)

TAZARA	=	Tanzania and Zambia Railway Authority
TRL	=	Tanzania Railways Limited
TZR	=	Tanzania and Zambia Railway Authority
VOC	=	Open Circuit Voltage
VVVF	=	Variable Voltage Variable Frequency

CHAPTER I

INTRODUCTION

1.1 Background

Mobility is an essential human need. Since the dawn of human activity; quick, comfortable, and efficient means of transportation has always been a constant goal of every organized society. The Watt steam engine and later the steam locomotive was the major breakthrough in land transportation, in particular the railroad transportation. Due to technology advancements, today's railroad vehicles can be driven by internal combustion engines, electric traction motors, or the combination of the two. The rail transport is characterized by the guided movement of steel wheels on steel rails, resulting to a metal-to-metal contact that considerably reduces rolling resistance (Profillidis, 2006).

With the growing mobility demand, depletion of the limited petroleum resources, environmental concerns, energy prices, and traffic congestion in cities; the need for efficient and sustainable intracity transportation systems has never been more critical than it is today. Thanks to public transportation systems that are often proposed as a solution to problems caused by urban traffic congestion. Large scale adoption of electric driven public transportation may be a sustainable solution not only to urban traffic congestion but also to economic and environmental concerns in transportation (Li and Lo, 2014), and (Egbue and Long, 2012).

In Dar es Salaam, the Tanzania's largest and highly congested city; most of the intracity public transport is by minibuses. The minibuses tend to be extremely overcrowded during morning and evening rush hours. Keen to ease transport challenges within the city, the government of Tanzania introduced two diesel commuter trains in October 2012; after the rehabilitation of the dilapidated locomotives and coaches which were used for up-country journeys and then discarded for years. The commuter train service is available in the morning and evening rush hours of the working days. Unfortunately, attributed by trains' low efficiency especially in short journeys with frequent stops, high fuel consumption leading to high operating cost has been a challenge (TAZARA), and (Nachilongo, 2013).

Electric trains being more efficient, fuel-cost effective and environmental friendly than the traditional diesel trains are expected to reduce passenger travel fuel cost and emission level. However, the existence of diesel trains is due to the fact that, the railways in which they operate are not electrified. Not only in Tanzania, there still exist several non-electrified railway networks around the globe, and it will take time to electrify the whole network. Electrification is most often applied to railways where the density of traffic movement is sufficient to justify the high capital and maintenance costs. With the recent advancements in battery technology, this research studies and analyses the potential of BEMUs (Figure 1.1) in reducing passenger travel fuel cost and emission, taking Tanzania diesel commuter trains as a case study.

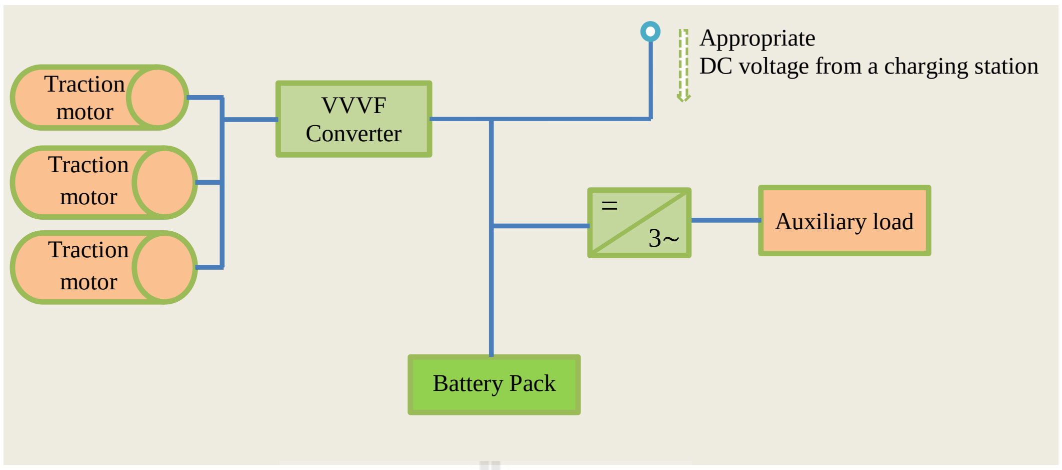


Figure 1.1 BEMU block diagram

1.2 Problem Definition

Electric multiple units (EMUs) are more efficient, fuel-cost effective and environmentally friendly than the traditional diesel trains. In short distances, with frequent stops, the efficiency of a traditional diesel train is even much lower than that of an EMU. However, there still exist several diesel trains operating on non-electrified railway networks, and it will take time to electrify the whole network. With the recent advancements in battery technology, BEMU may be operated on short distances, non-electrified railway networks. Thus, reduce passenger travel fuel cost and emission.

1.3 Research motivation

Despite all the advantages of an EMU over a traditional diesel train, diesel trains still operate on non-electrified railway networks and it will take time to electrify the entire network. Owing to their high power density, high energy density and excellent stability, lithium titanate (LTO) batteries have emerged as the leading candidates and proven success in vehicular applications. To a point that, battery powered electric buses are nowadays increasingly being manufactured. A very interesting fact is that, a train having low coefficient of rolling friction, consumes less energy per passenger kilometer than a bus for the same traffic.

This thesis work therefore, applies artificial intelligence techniques to search for a tram's optimal speed profile, models the tram as a BEMU, simulates its movement, and analyzes reduction in passenger travel fuel cost and emission, taking Tanzania diesel commuter trains as a case study.

1.4 Research objectives

The main objective of this research work is to study the possibility of BEMUs to provide passenger travel services in short distances, non-electrified railway networks and hence reduce passenger travel fuel cost and CO_2 emission. That is, the research attempts to answer the question "Is it possible for BEMUs to provide passenger travel services in short distances, non-electrified railway networks?, and should that be possible, by how much will they reduce fuel cost and CO_2 emission per passenger kilometer?". In so doing, the research takes a case study of Tanzania diesel commuter trains, and considers a short distance as a distance not greater than 30 km.

To achieve the main objective, the research is composed of the following specific objectives:-

a) To collect the performance data of the two Tanzania diesel commuter trains.

b) To establish tram's optimal speed profiles. As the two Tanzania commuter trains have different routes, this is to be done for each route.

c) To model the tram as a BEMU and simulates its movement in each route.

d) To analyze by how much the BEMU reduces passenger travel fuel cost and emission as compared to the Tanzania diesel commuter trains.

1.5 Methodology

Aiming at reducing passenger travel fuel cost and emission taking a case study of the two Tanzania diesel commuter trains, data related to the trains' performances are collected. Through available literatures, train kinematics and dynamics, electric traction systems and, global optimization methods are studied. Differential evolution (DE) and particle swarm optimization (PSO) algorithms are developed using MATLAB®. An EMU to be used in the study is chosen and then the MATLAB® built in genetic algorithm (GA) and, the developed DE and PSO algorithms are used to search for tram's optimal speed profiles taking into account, speed, time, and acceleration constrains. The speed profiles are compared and the best is taken.

The tram is then modelled as a BEMU, and based on the best speed profiles, tram movement is simulated. Given the diesel and electricity prices in Tanzania, and CO_2 emission factors; it is finally analyzed, to what extend the BEMU reduces passenger travel fuel cost and CO_2 emission.

1.6 Research Assumptions

This research is guided by the following assumptions:

a) Tram's gradient force is negligible.

   Dar es Salaam is in a coastal zone which has a fairly flat terrain, and the two commuter train routes are fairly flat. This makes train's gradient force to be very low.

b) A chosen tram is (i) powered by lithium titanate batteries, (ii) capable of regenerative braking and, (iii) can operate on either of the two commuter train routes.

c) Charging points have been installed at every terminal station, and at some of the selected intermediate stations as desired, such that, when the tram arrives at a charging station, a connection is made very fast and charging process begins.

1.7 Scope and Limitations of the Study

Carbon dioxide makes up 95% of all transportation-related greenhouse gas emissions. Relatively small amounts of methane and nitrous oxide are also emitted during fuel combustion. As far as emissions are concerned, this research takes into account only the carbon dioxide emission.

Moreover, to the extent that costs are concerned, this research focuses on fuel cost per passenger kilometer which is the main contributing factor to operating cost. To what extent a BEMU reduces passenger travel fuel cost, will be a way forward to analyze as to whether or not, the BEMUs are a cost effective alternative to diesel trains, serving short distances.

1.8 Expected Benefits

Results from this research will serve as guidelines to policy makers, city authorities and other stakeholders to make policies and plans, and take actions that will reduce passenger travel fuel cost and emission level.

1.9 Organization of the Thesis

The thesis comprises of five chapters. Background of the study, problem description, objectives, methodology, scope and significance of the research are presented in **Chapter I**.

Chapter II gives a brief review on traction systems, batteries, train kinematics and dynamics, induction motors and regenerative braking, and optimization. Information on Tanzania commuter trains and Tanzania electric power industry are also given in **Chapter II**.

In **Chapter III**, artificial intelligence techniques are applied to search for optimal speed profiles, and the best profiles are chosen to be used in tram movement simulation.

Tram movement simulation results are presented, analyzed and discussed in **Chapter IV**. And finally, a conclusion and recommendations are given in **Chapter V.**

CHAPTER II

GENERAL REVIEW

2.1 Introduction

Reducing traffic congestion, passenger travel fuel cost and emission is definitely a primary concern of every city authority. Public transport such as trains, significantly reduce traffic congestion. Replacing diesel commuter trains with independently powered electric multiple units (IPEMUs) will more than likely reduce not only traffic congestion but also passenger travel fuel cost and emission. In this chapter various aspects about IPEMUs and diesel commuter trains are presented and briefly discussed. Information concerning Tanzania commuter trains and electric power industry are also presented in this chapter.

2.2 Traction Systems and Carbon Dioxide Emission

Carbon dioxide makes up 95% of all transportation-related greenhouse gas emissions resulting from the combustion of petroleum-based products. Diesel fuel releases approximately 2.68 kg of CO_2 per liter burned in an internal combustion engine (ICE). The Nobel Prize winning 2007 intergovernmental panel on climate change report concluded that greenhouse gas emissions must be reduced by 50% to 85% by 2050 in order to limit global warming to four degrees Fahrenheit, thereby avoiding many of the worst impacts of climate change (Hodges, 2010).

Some of the methods to reduce CO_2 emission and hence mitigating the global warming as suggested by (Abu-Rub, et al., 2014) are:

- Promote all energy consumption in electrical form. Centralized fossil fuel power stations can effectively use emission control strategies;
- Extensively promote environmentally clean renewable energy systems;
- Replace internal combustion engine (ICE) vehicles by electric vehicles; and
- Promote mass transportation, particularly electric trains and buses.

Apart from reducing CO_2 emission, electric traction systems have several other advantages over the traditional diesel traction systems. Advantages of electric trains over diesel trains as given in (Steimel, 2008) include:

- Possibility of energy recovery when braking, less wear of brake shoes;
- Possible overload of electric machinery can be utilized;
- Easier maintenance, lower maintenance cost, higher number of operational hours; and
- Low energy cost, generally a cheaper traction unit.

As the saying goes, there are two sides to every coin. The disadvantage to electric trains is the high initial cost for catenaries and power-supply network. This is the obvious reason why some diesel trains are still in operation.

2.3 Tanzania Commuter Trains

Tanzania railway system comprises of two networks operated by two companies. Tanzania-Zambia Railway Authority (TAZARA or TZR), operates a single track 1067 mm gauge railway network from Dar es Salaam port, Tanzania to a town of Kapiri Mposhi Zambia. And Tanzania Railways limited (TRL) operates the other single track 1000 mm gauge railway network from Dar es Salaam port to the inland as shown in Figure 2.1.

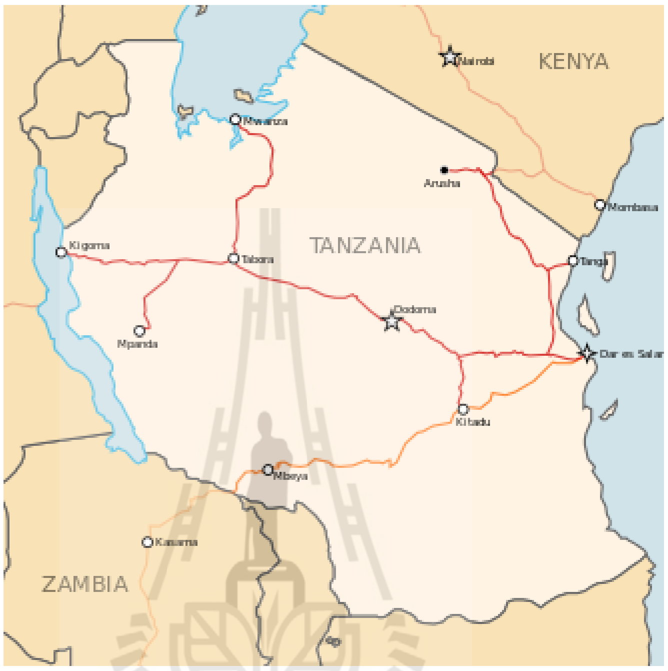


Figure 2.1 Railways in Tanzania (Wikipedia, 2015)

Following the traffic congestion and traveling hardships in the congested Tanzanian commercial city of Dar es Salam; the government of Tanzania introduced two diesel commuter trains in October 2012. Services are provided only in the morning and evening rush hours of the working days, by TAZARA (or TZR) and TRL using part of their respective railway networks. TAZARA commuter train (Figure 2.2) serves a route of 21 km from TAZARA Dar es Salaam main station to Mwakanga, while TRL commuter train (Figure 2.3) serves a route of 12.05 km from Dar es Salaam Central station to Ubungo-Maziwa as shown in Figure 2.4.

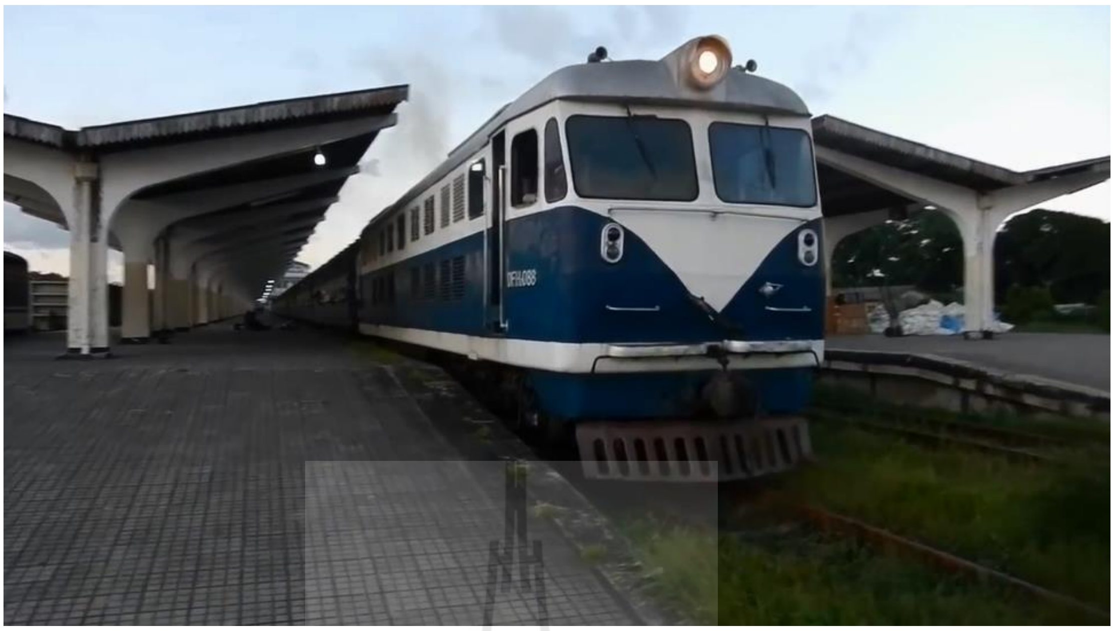


Figure 2.2 TAZARA commuter train, heading to Mwakanga

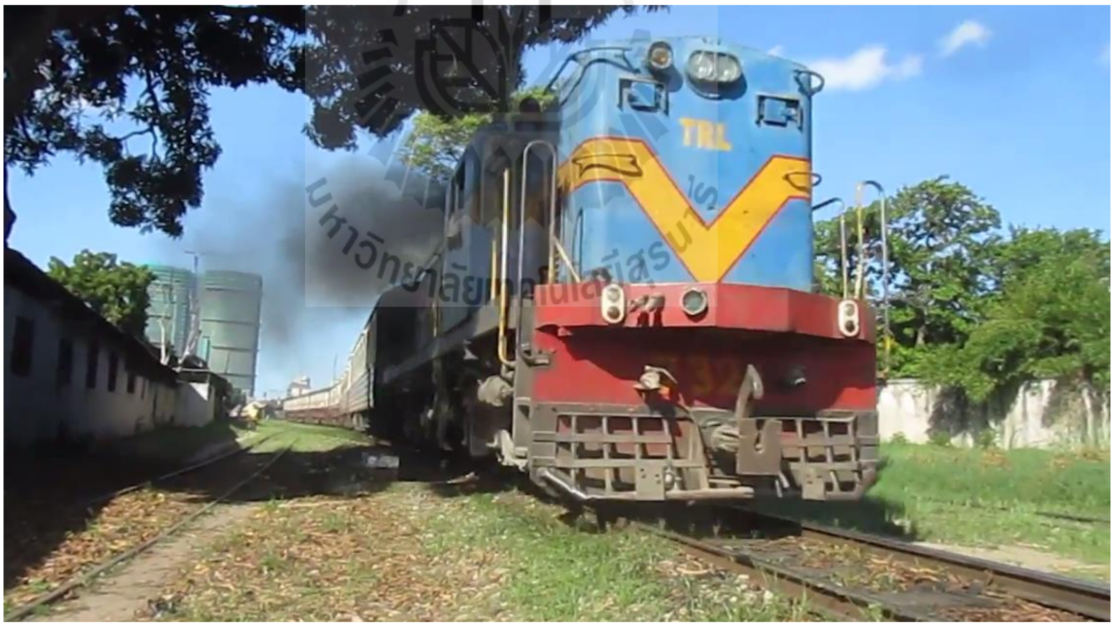


Figure 2.3 TRL commuter train, heading to Ubumgo-Maziwa

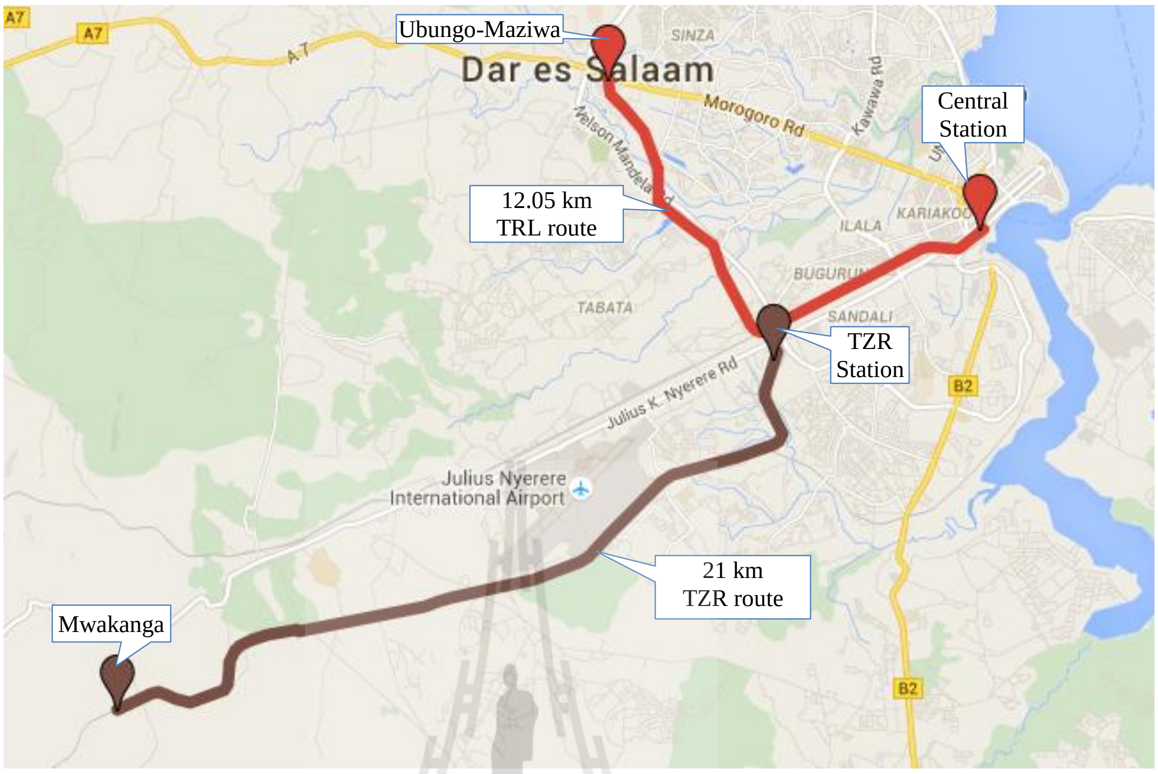


Figure 2.4 TRL and TAZARA commuter trains' routes

The trains were introduced after the rehabilitation of the dilapidated locomotives and coaches which were discarded for years. Attributed by low efficiency of the locomotives and coaches especially in short journeys with frequent stops; high fuel consumption leading to high operating cost has always been a challenge. The TRL commuter train was reported to have caused a loss of approximately US$ 334000 by November 2013, operating at a loss of approximately US$ 1000 per day (Nachilongo, 2013).

2.4 Batteries and their Applications in Electric Vehicles

Generally a battery is an assemblage of cells electrically connected in series and/or parallel to provide the desired voltage and current. The cells consist of positive and negative electrodes in an electrolyte. It is the chemical reaction between the electrodes and the electrolyte which generates direct current (DC) electricity. In the case of secondary (or rechargeable) batteries the chemical reaction can be reversed by reversing the current, and the battery returned to a charged state. In primary (or non-rechargeable) batteries, the chemical reaction is non-reversible (Larminie and Lowry, 2012).

In this section, battery parameters and three leading battery types: lead-acid (PbA), nickel metal hydride (Ni–MH) and, lithium-ion (Li-ion) batteries are briefly discussed. Information on battery electric multiple units (BEMUs) and, battery electrical model used in this thesis work are also briefly presented in this section.

2.4.1 Battery Parameters

Some of the basic definitions and aspects in the field of batteries include the following:

- *Capacity (C) and C-rate*

The battery capacity specifies the amount of electric charge a fully charged battery can supply before it is fully discharged. A more general unit for battery capacity is ampere-hour (Ah). A battery of m Ah can supply 1A current for m hours or in theory $\frac{m}{x}$ A for x hours. But in general, the battery capacity is dependent on discharge rate. Generally the higher the discharge rate, the lower capacity the battery will have, a degree to which the capacity is reduced will depend on the battery chemistry (battery type). The charge and discharge current of a battery is measured in C-rate, in order to normalize it against the battery capacity; if m is the battery capacity in Ah, nC rate will be equal to nm A (Mi., et al, 2011).

- *State of Charge (SOC) and Depth of Discharge (DoD)*

Battery state-of-charge (SoC) is the percentage of charge available from a battery relative to the entire capacity of the battery. On the other hand, depth-of-discharge (DoD) is the percentage of charge drawn from the battery relative to the entire battery capacity. The DoD parameter is often used in discharge pattern recommendations. A battery manufacturer might recommend not to exceed a certain level of DoD (depending on the type of the battery chemistry) in relation to battery lifetime issues (Abu-Rub., et al, 2014). If a battery is charged or discharged at C_r C-rate (in computer simulation the current and hence C_r is negative during charging and positive during discharge), from time t_o to t then, its SoC can be expressed as

$$SoC(t) = SoC(t_0) - 100\% \int_{t_0}^{t} C_r(\tau)\, d\tau \tag{2.1}$$

where SoC(t_o) is the initial SoC (SoC at time t_o).

- *Power and energy ratios*

Maximum power that a battery can deliver and nominal energy that can be stored in the battery are often expressed in per unit mass or volume of the battery. These ratios are important technological aspects in applications such as electric traction. Specific power [W/kg] is the maximum available battery power per unit mass and, power density [W/m^3] is the maximum available battery power per unit volume. The higher the ratios, the faster the battery can give and take energy. Specific power and power density determine the battery weight and size required to achieve a given charge or discharge rate. Specific energy [Wh/kg] is the nominal battery energy per unit mass and, energy density [Wh/m^3] is the nominal battery energy per unit volume. Along with the energy consumption of a vehicle, specific energy determines battery weight and energy density determines battery size required to achieve a given electric range.

- *Ampere-Hour (or Charge) Efficiency and Energy Efficiency*

The chemical reactions inside the battery during charge and discharge are not ideal and there are always losses involved. In other words, not all the energy used to charge the battery is available during discharge. Some of this energy is wasted in other forms of energy dissipation such as heat energy dissipation. Ampere-hour efficiency is the ratio between the electric charge given out during discharging a battery and the electric charge needed for the battery to return to the previous SoC. The typical values of charge efficiency range from 65 to 90%. The efficiency depends on various factors such as the battery type, temperature, and rate of charge. This parameter may be mentioned by other names such coulombic efficiency or charge acceptance. On the other hand, energy efficiency is the ratio of electrical energy supplied by a battery to the amount of charging energy required for the battery to return to its previous SoC. The efficiency decreases considerably if a battery is discharged and charged very quickly, it also reduces as the battery ages. Typically, the energy efficiency of a battery is in the range of 55–95% (Abu-Rub., et al, 2014) and (Mi, et al, 2011).

2.4.2 Lead Acid Batteries

PbA batteries are relatively old technology that maintains 40–45% of the battery market, mainly due to their extensive use as starting, lighting, and ignition (SLI) batteries in automobiles, trucks, and buses. They are also attractive for hybrid electric vehicles (HEVs) and energy storage applications owing to their relatively high round-trip efficiencies of 75–80%. However, for electric vehicles (EVs), more robust PbA batteries that withstand deep cycling and use either a gel or adsorbed glass mat (AGM) rather than a liquid electrolyte are used. Valve regulated lead acid (VRLA) batteries are modern PbA designs that immobilize the electrolyte using either highly porous and absorbent mats or a fumed silica gelling agent (Rahn and Wang, 2013) and (Larminie and Lowry, 2012)

In the PbA cells, the negative plates have a spongy lead as their active material, while the positive plates have an active material of lead dioxide. The plates are immersed (either fully or partially) in an electrolyte of dilute sulfuric acid. The sulfuric acid combines with the lead and the lead oxide to produce lead sulfate and water, electrical energy being released during the process. Both electrode reactions result in the formation of lead sulfate. The electrolyte gradually loses the sulfuric acid, and becomes more dilute. When being charged, the electrodes revert to lead and lead dioxide. The electrolyte also recovers its sulfuric acid, and the concentration rises. The overall chemical reaction for the lead acid battery is given as

$$\mathrm{PbO_2 + Pb + 2H_2SO_4 \xrightleftharpoons[charge]{discharge} 2PbSO_4 + 2H_2O} \tag{2.2}$$

A gas phase (when hydrogen and oxygen are released, leading to loss of electrolyte) may be present in the cell if it is overcharged or over-discharged. The formation of gas phase is a side reaction that is not desirable and can lead to unsafe conditions and/or reduced battery life. Lead-acid batteries can last a long time if they are charged and discharged properly. (Rahn and Wang, 2013), (Larminie and Lowry, 2012) and (Mi., et al, 2011). Lead, sufuric acid and plastic containers being considerlably inexpensive, makes PbA batteries relatively cheap. But, their main disadvantge with regard to electric traction is that, they have low power and energy ratios, and they and not abuse tolerant.

2.4.3 Nickel–Metal Hydride Batteries

The Ni–MH batteries offer higher performance at higher cost than VRLA batteries. They have very good cycle life and capacity, can be recharged very fast and, are safe and abuse tolerant. They have been heavily used in HEV applications, including the Toyota Prius. The disadvantages of Ni–MH batteries are high self-discharge rate and poor charge acceptance capability (low charge efficiency) at elevated temperatures (Rahn and Wang, 2013) and (Husain, 2003).

In the Ni-MH cell, the positive electrode contains nickel hydroxide as its principal active material and the negative electrode is mainly composed of hydrogen-absorbing nickel alloys. The cell has an electrically insulating separator, an alkaline electrolyte (e.g., a solution of potassium hydroxide, KOH), and a vented metal case. During discharge, nickel oxyhydroxide (NiOOH) is reduced to nickel hydroxide ($Ni(OH)_2$) in the positive electrode and in in the negative electrode, metal hydride (MH) is oxidized to the metal alloy (M); and during charge; the electrodes revert to nickel oxyhydroxide and metal hydride. The overall reversible chemical reaction for the Ni-MH battery cell is given as

$$\mathrm{MH+NiOOH} \underset{charge}{\overset{discharge}{\rightleftharpoons}} \mathrm{M+Ni(OH)_2} \qquad (2.3)$$

The basic principle is a reversible reaction in which hydrogen is bonded to the metal and then released as free hydrogen when required. An interesting feature of the Ni-MH cell is that, the composition of the electrolyte does not change during charge or discharge – both water and OH^- ions are created and used at exactly the same rate. The result is that the internal resistance and open-circuit voltage of the cell are much more constant during discharge than with PbA batteries (Rahn and Wang, 2013) and (Larminie and Lowry, 2012).

2.4.4 Lithium-Ion Batteries

Definitely a battery suitable for an electric vehicle (EV) is the one with high power and energy ratios, low self-discharge rate, low energy cost per cycle life and, abuse tolerant (high safety level). With convectional battery technologies (which have low power densities) such as lead-acid batteries; battery-supercapacitor hybrid systems including high energy batteries and high power supercapacitors can be used. Supercapacitors have higher power densities and longer cycle life than batteries. However, they are heavy (have low energy density) and have high self-discharge rates. Besides, following the development in Li-ion battery technologies, Li-ion is the most energy- and power-dense battery type. Features of Li-ion batteries are improving while prices decrease. Thus, a suitable battery can be used rather than a supercapacitor in order to maintain high power (Dincer, et al., 2015).

Lithium metal has high electrochemical reduction potential (3.045 V) and the lowest atomic mass (6.94), which shows promise for a battery of 3 V cell potential when combined with a suitable positive electrode. The negative electrode of a convectional Li-ion cell is made from carbon in the form of graphite or coke. The positive electrode is a lithium metallic oxide such as lithium cobalt oxide, lithium iron phosphate and lithium manganese oxide. The electrolyte is lithium salt in organic solvent. Li-ions (Li^+) alternatively move into and out of host lattices during charging and discharging cycles. This fundamental mechanism has led to the Li-ion battery's nick-name of “rocking-chair” battery. During cell discharge, the Li+ are released from the negative electrode and travels through the electrolyte toward the positive electrode. In the positive electrode, the lithium ions are quickly incorporated into the lithium compound material. The process is completely reversible; the overall reversible chemical reaction occurring in a Li-ion cell is

$$Li_xC + Li_{1-x}M_yO_z \underset{charge}{\overset{discharge}{\rightleftharpoons}} C + LiM_yO_2 \tag{2.4}$$

Unlike PbA and Ni-MH batteries, the Li-ion batteries have a much lower self-discharge rate, thus greatly increasing idle period capabilities. These bateries also have longer cycle life (Husain, 2003) and (Mi., et al, 2011)

The family of Li-ion batteries is vast, most common Li-ion batteries include lithium cobalt oxide (LCO), lithium manganese oxide (LMO), lithium nickel manganese cobalt oxide (NMC), lithium nickel cobalt aluminum oxide (NCA), lithium iron phosphate (LPO) and lithium titanate (LTO). A comparison between these Li-ion battery types is given in Table 2.1

Table 2.1 Comparison of Li-ion batteries (Stroe, et al., 2015).

Battery type	Specific capacity [mAh/g]	Nom. Voltage [V]	Energy density [Wh/kg]	Cycle life [cycles]
LCO	140	3.7	110 – 190	500 – 1000
LMO	146	3.8	100 – 120	1000
NMC	145	3.6	100 – 170	2000 – 3000
NCA	180	3.6	100 – 150	2000 – 3000
LPO	170	3.3	90 – 115	> 3000
LTO	170	2.2	60 – 70	> 5000

LTO battery is a new generation of Li-ion battery which uses lithium titanate oxide ($Li_4Ti_5O_{12}$) as negative electrode instead of graphite. The nanoparticles of lithium titanate oxide increase the electrode-electrolyte contact area and reduce the diffusion distance for lithium ions and electrons, thus reduce the polarization resistance and allow the quick charging and discharge (up to 10C rate) of lithium titanate cells. Lithium titanate oxide also saves other advantages, such as superior safety, outstanding cycling stability, negligible volume expansion in charging/discharging processes, excellent low temperature performance, low toxicity and low material cost. These remarkable advantages have made LTO battery a leading candidate for fast charging and power assist vehicular applications (Stroe, et al., 2015), (Liu, et al., 2015), and (Yao, et al., 2014).

A disadvantage of lithium-titanate batteries is that they have a lower cell voltage, which leads to a lower energy density than conventional lithium-ion battery technologies. However, this lower cell voltage prevents lithium deposition, even under overcharge conditions or with longer cycling, which results in enhanced safety and long life (Stroe, et al., 2015), and (Horiba, 2014).

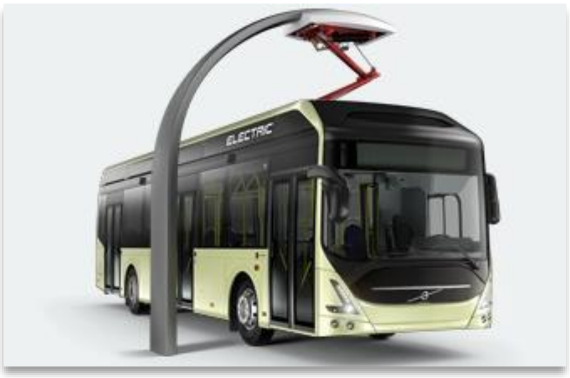
2.4.5 Lithium Titanate Batteries in Public Transportation Systems

Opportunity charging in public transportation systems, such as large capacity articulated electric bus project TOSA (Figure 2.5a), is using the LTO batteries high charging rate capability to partly recharge the batteries at selected bus stops in a record time of 15 seconds while passengers get on and off the bus. At the end of the bus line a 3 to 4 minutes ultrafast-charge is made to fully recharge the batteries (TOSA, 2013), and (ABB Communications, 2013).

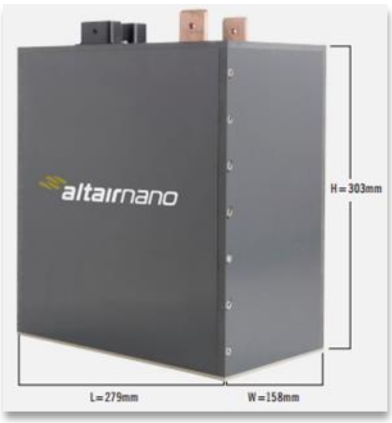
Škoda Perun HP electric bus (Figure 2.5b) is using a 78 kWh LTO battery pack. The battery pack can be recharged quickly (within 5-8 minutes) to the full capacity and the bus is ready to run other 30 km (Škoda Transportation). With LTO batteries the Volvo 7900 Electric bus (Figure 2.5c) is also using opportunity charging to fast recharge the batteries to full capacity (4 x 19 kWh) within 6 minutes (Volvo). The Volvo 7900 Electric Hybrid is also using LTO batteries, with opportunity charging the bus runs on electricity for 70% of a normal route, cuts energy consumption by 60% and reduces CO_2 emissions by 75% (Volvo). Shown in (Figure 2.5d), is a 27.4 kg, 1.4 kWh (24 V, 60 Ah) Altairnano LTO battery which complies with the UN 3090, UN 3480 transportation specifications (Altairnanno).

(a) TOSA articulated battery bus (TOSA, 2013).

(b) Škoda Perun HP electric bus (Škoda Transportation)



(c) Volvo 7900 Electric bus (Volvo)



(d) Lithium titanate battery (Altairnanno).

Figure 2.5 Lithium titanate battery in public transportation systems

Researches are still going on to further improve battery performance at low cost. In August 2015, a team of researchers from Tsinghua University, China; Massachusetts Institute of Technology and University of Wisconsin-Milwaukee, USA; revealed new findings in Li-ion battery technology that would increase battery cycle life, energy and power densities at low cost. These new findings which incorporate nano-scaled aluminum core with adjustable interspace and titanium oxide layer (AL/TiO_2) as anode, have been reported in a publication by (Li, et al., 2015).

It is very interesting that, owing to low coefficient of rolling resistance rail transit consumes less energy per passenger kilometre than road transport for the same traffic. Authours in (Mwambeleko, et al., 2015) present a publication that compares net traction energy consumption of a tram and a trolley bus; and conclude that, owing to the low rolling resistance the tram consumes only 57 percent of the energy consumed by the trolleybus per passenger kilometer. The author in (Profillidis, 2006) points out that; owing to the low rolling resistance, rail transport consumes less energy than road transport for the same traffic. This implies that, if the LTO batteries have proven success powering the electric buses, such as the articulated battery powered electric bus TOSA, they will prove even more success powering battery electric multiple units. And if battery powered buses are cost effective, then BEMUs will be even more cost effective.

2.4.6 Battery Electric Multiple Units

There have been various attempts to utilize battery powered railcars, notably British Rail operated a two car electric railcar (Figure 2.6) powered by a series of lead acid batteries on Deeside in Scotland from 1958 to 1962. Designed to take advantage of cheap hydroelectric power, the railcar successfully operated for a period of about four years prior to closure of the Deeside branch (UK Tram, 2012). Between 1955 and 1995 Deutsche Bahn (DB) railways successfully operated electric railcars (Figure 2.7) utilising lead-acid batteries (Wikipedia, 2015). Despite the relatively old battery technology, these battery railcars were successful. Over the past two decades battery technology has greatly improved.

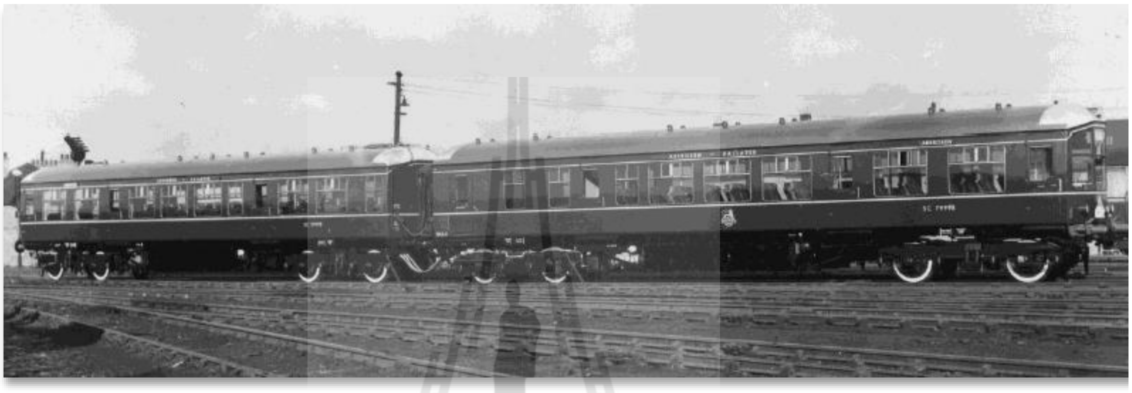


Figure 2.6 British rail battery powered rail car (UK Tram, 2012).

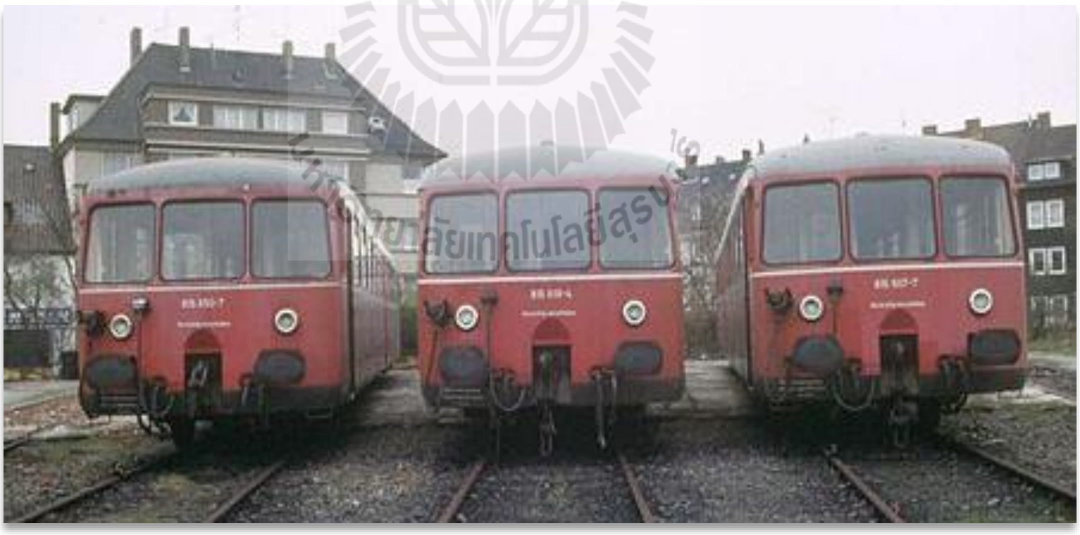


Figure 2.7 DB battery powered driving railcars (Wikipedia, 2015)

In May 2011, a Munich Stadler variobahn tram was reported to set a new world record by driving 16 km on a single charge of a 380 kg Li-ion battery pack

(Stadler, 2011). In Japan, the East Japan Railway Company (JR East) has been developing and testing “catenary and storage battery hybrid train system” since 2011. By March 2015, the EV-E301 series electric railcar (Figure 2.8) was in commercial operation capable of driving on a 20.4 km non-electrified section (Karasuyama line), using Li-ion batteries (Hirose, et al., 2012), and (Shiraki, et al., 2015).

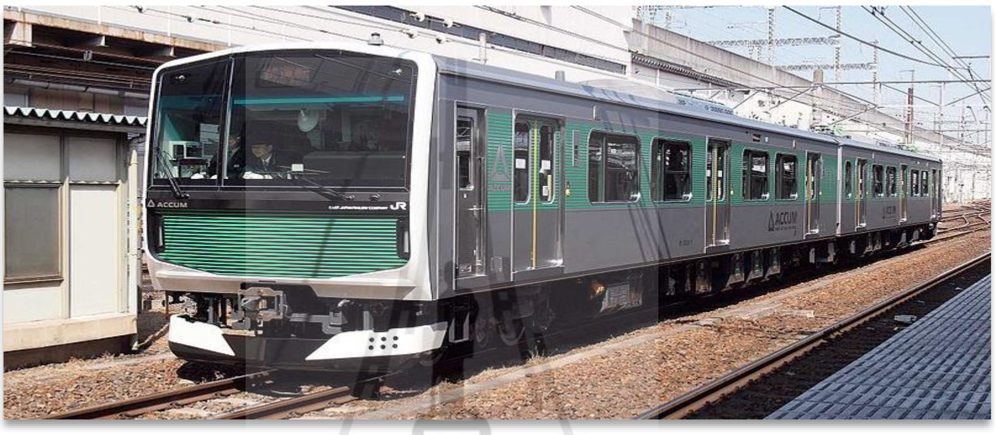


Figure 2.8 JR East Catenary and battery hybrid railcar EV-E301(Shiraki, et al., 2015).

In November 2015 Bombardier battery powered tram (Figure 2.9) set a range record when it successfully completed a 41.6 km catenary-free test run, the tram uses ‘*PRIMOVE’* battery system (Figure 2.10) which has been developed by Bombardier using nickel manganese cobalt (NMC) Li-ion cells (Bombardier Transportation, 2015).

Figure 2.9 Bombardier battery powered tram (Bombardier Transportation, 2015)

(a) Bombardier 49 kWh, 660 kg PRIMOVE battery

(b) Bombardier 30 kWh, 360 kg PRIMOVE battery

Figure 2.10 Bombardier PRIMOVE batteries (Bombardier Transportation, 2015)

It was reported in (Škoda Transportation, 2013) that "In the case of trams for the Turkish city of Konya, Škoda is going to use high-performance batteries with nano-lithium-titanium technology. The batteries are always recharged during tram operation under the trolley in just a few minutes."

2.4.7 Battery Electrical Model

There exist numerous techniques by which a battery electrical model can be derived. For this thesis work, the Thevenin equivalent battery model as shown in Figure 2.11 was considered sufficient. The battery model consists of an ideal battery with open-circuit voltage VOC_{Batt} in series with a constant equivalent series resistance ESR_{Batt}. From the given battery model, the output voltage U_{Batt} will drop with increase in discharge current and rise with the increase in charge current due to voltage drop across the resistor ESR_{Batt}. The output voltage U_{Batt} is then given as

$$U_{Batt} = VOC_{Batt} - I_{Batt}\mathrm{ESR}_{Batt} \tag{2.5}$$

where I_{Batt} is the battery current which is positive during discharge and negative during charging.

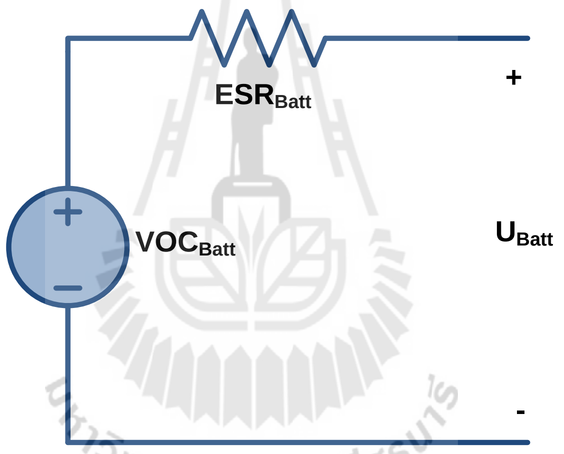


Figure 2.11 Battery equivalent electric model

As mentioned in section 1.6, in this thesis work the tram is assumed to be powered by lithium titanate batteries. A typical discharge curve of a Li-ion battery is shown in Figure 12. When the battery is fully charged its open-circuit voltage is higher than the nominal voltage. And, when the battery discharges, the open-circuit voltage drops exponentially to a certain level slightly above the nominal voltage where it stays fairly constant to about 80% DoD (depending on the rate of discharge). And when the battery is fully discharged its open-circuit voltage will be slightly lower than the nominal voltage. Thus the open-circuit voltage of a Li-ion battery stays reasonably constant throughout the discharge circle.

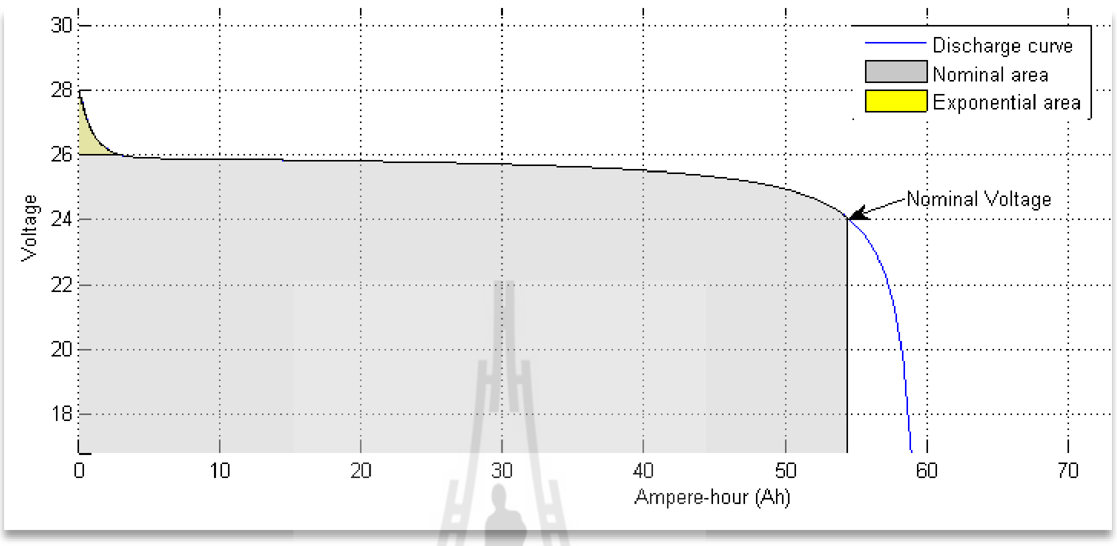


Figure 2.12 Typical Li-ion battery discharge curve

2.5 Electric Power Industry in Tanzania

The electricity supply industry in Tanzania is dominated by the Tanzania Electric Supply Company Limited (TANESCO), which is a vertically integrated utility, wholly owned by the government. TANESCO owns and operates the interconnected main grid (220 kV, 132 KV and 66 kV) connecting the major load centres (EWURA, 2015). As of May 2014, Tanzania's total installed generation capacity was 1,583 MW; composed of hydro 561 MW (35 %), natural gas 527 MW (34 %) and liquid fuel 495 MW (31 %) power plants; and the system losses were 19%. TANESCO also imports power from Uganda (10 MW), Zambia (5 MW) and Kenya (1MW) (MEM, 2014).

2.6 Train Mechanics

A clear understanding of the properties of motion of objects, forces and torques and, their effect on motion is a prerequisite to understand train movement and energy consumption. In this section the theory behind train kinematics, train dynamics and traction energy consumption are briefly presented.

2.6.1 Train Kinematics

Train kinematics can be well understood from its speed profile curve drawn as speed versus time as shown in Figure 2.13. The slope of the curve represents acceleration or deceleration as the case may be, and area under the curve gives the distance travelled. The speed profile curve generally consists of four operating modes namely: i) accelerating or motoring mode, ii) constant speed or cruising mode, iii) coasting or freewheeling mode and, iv) braking mode.

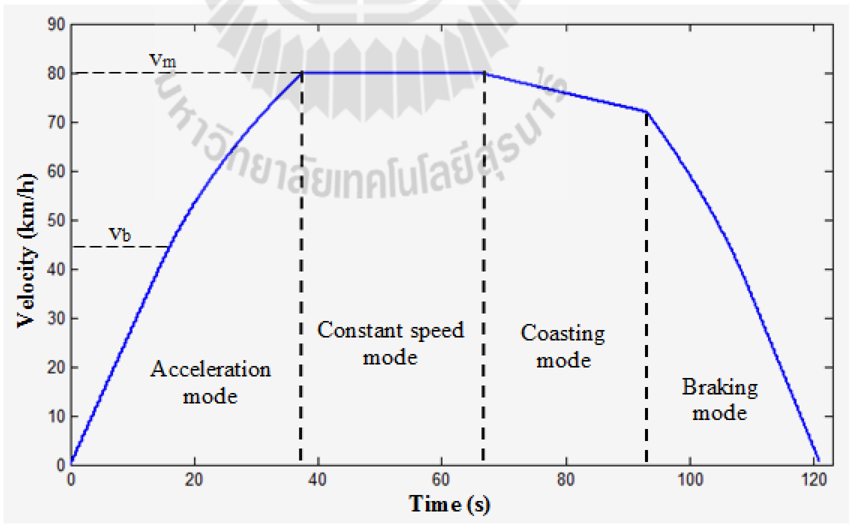


Figure 2.13 General train speed profile curve

Normally when a train starts moving, it accelerates with a constant acceleration to a certain speed called base speed v_b. From the base speed to maximum desired speed v_m tractive power is kept constant and tractive force reduces inversely to speed, till it balances that due to resistance to the train motion (Steimel, 2008), and (Rajput, 2006). To avoid slipping, the tractive force is always kept lower than or equal to the maximum allowable force, depending on the adhesion between the driving wheels and the track. A typical tractive force curve is shown in Figure 2.10.

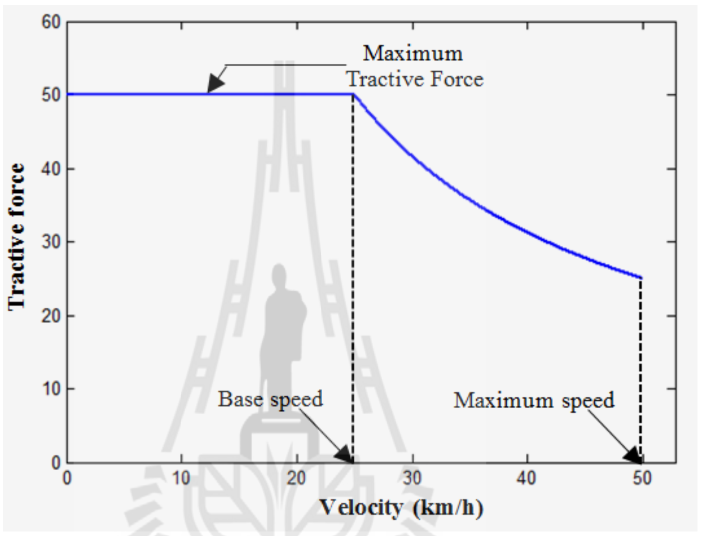


Figure 2.14 Typical tractive force curve

When the tractive force balances the forces opposing the motion, the train moves with a constant speed (zero acceleration) drawing constant power. As a technique to minimize tractive energy consumption, before brakes are applied, tractive power is disconnected and the train is left to move with its own kinetic energy, this is called coasting or freewheeling. The train speed starts decreasing on account of resistance to motion. Coasting has long been recognized as one of the techniques to save energy, though it increases interstation run time (Rajput, 2006), and (Goodman, 2008).

Finally, brakes are applied and the train is brought to a stop. As with the tractive force, braking force is also limited to a maximum allowable force. A typical braking force curve is as shown in Figure 2.15, the negative implies braking (force in the opposite direction to motion). Modern trains use regenerative braking to recapture train's kinetic energy. Some trains just use rheostatic braking, where the braking energy is dissipated in a bank of resistors as heat. As opposed to friction braking dynamic braking (regenerative or rheostatic braking) has proven to minimize maintenance cost of the braking system (Jiang, et al., 2014).

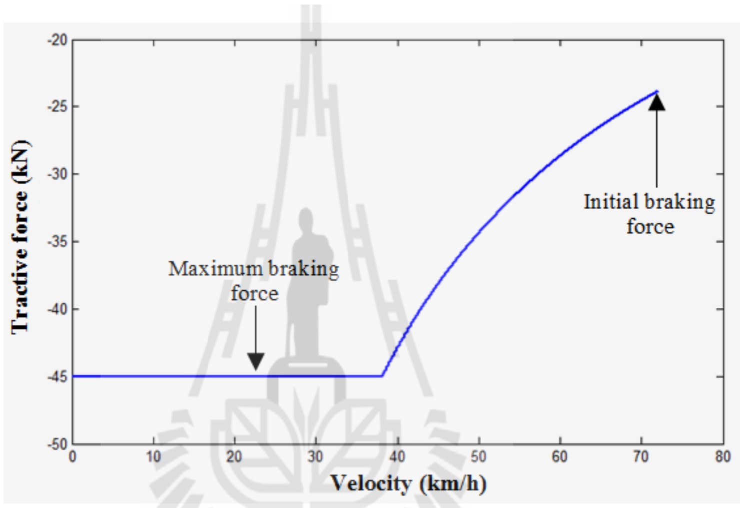


Figure 2.15 Typical braking force curve

2.6.2 Train Dynamics

Just like any other vehicle, train's movement calculations are governed by the Newton's laws of motion. A free body diagram showing the forces acting upon an uphill moving train is shown in Figure 2.16.

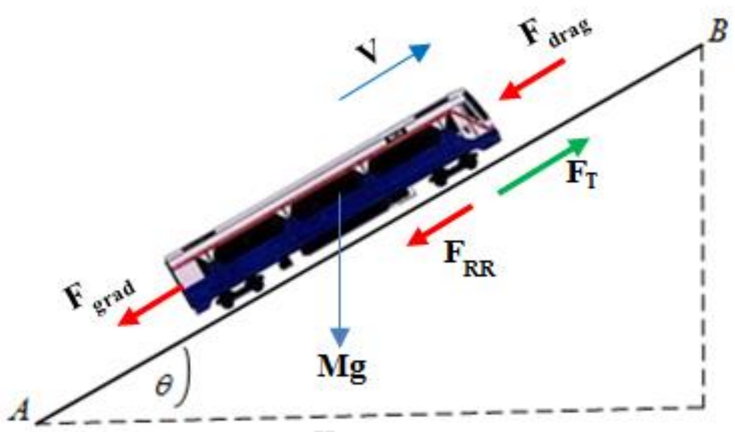


Figure 2.16 Free body diagram of an uphill moving train

The relationship between the forces is expressed as

$$F_T = m\frac{dv}{dt} + F_R \tag{2.6}$$

$$F_R = F_{RR} + F_{grad} + F_{drag} \tag{2.7}$$

where F_T is the vehicle tractive force (N), F_R is the net resistance force against the vehicle's motion (N), F_{RR} is the vehicle rolling resistance force (N), F_{drag} is the aerodynamic drag force (N), F_{grad} is the vehicle gravitational (gradient) force (N), m is the vehicle effective mass (kg), dv is the change in velocity (m/s) and dt is the change in time (s) (Mwambeleko, et al, 2015).

The vehicle effective mass takes into account the rotary allowance and passenger load. Due to the fact that, when a train accelerates along a track the total mass (tare mass + passenger or freight mass) is accelerated linearly but the rotating parts are also accelerated in a rotational sense. The parts involved are usually the wheelsets, gears and motors, the effect of the latter being magnified by the gear-ratio squared (assuming the motors are geared to rotate faster than the wheels, which is normally the case). It is usual to express this rotational inertia effect as an increase in the effective linear mass of the train called the 'rotary allowance' and expressed as a fraction of the tare weight of the train. Thus, the train effective mass *m* is given as

$$m = m_{tare}\left(1+\lambda\right) + m_{pass\_fr} \tag{2.8}$$

where m_{tare} is the tare mass, λ is the rotary allowance and m_{pass_fr} is the passenger or freight load. The value of the rotary allowance varies from 0.05 to 0.15 depending on the number of motored axles, the gear ratio and the type of car construction. (Goodman, 2008) and (Kulworawanichpong, 2003)

2.6.2.1 Aerodynamic drag, $\mathbf{F_{drag}}$

The aerodynamic resistance force results from three basic effects: i) the pressure different in front and behind the vehicle, ii) skin friction representing the surface roughness of the vehicle body and iii) internal flow of air entering the internal parts of the vehicle. It is common to express the aerodynamic resistance force in the basic form as

$$F_{drag} = \frac{1}{2}\rho_{air} C_d A_f V_{air}^2 \tag{2.9}$$

where ρ_{air} is the air density (kgm^{-3}), C_d is an aerodynamic drag coefficient, A_f is the projected vehicle frontal area (m^2) and, V_{air} is the speed of air relative to the vehicle body (ms^{-1}) (Kulworawanichpong and Punpaisarn, 2014).

The density of air does of course vary with temperature, altitude and humidity. However, a value of 1.25 kgm^{-3} is a reasonable value to use in most cases (Larminie and Lowry, 2012).

2.6.2.2 Gradient force, F$_{grad}$

Gradient force is the force needed to drive a vehicle along the slope, which is simply the component of the vehicle weight that acts along the slope. It is positive when the vehicle is moving uphill and negative when the vehicle is moving downhill**,** mathematically expressed as

$$F_{grad} = \pm mgsin\theta \tag{2.10}$$

where m is the vehicle effective mass (kg), g is the acceleration due to gravity (9.81ms^{-2}) and, θ is the slope angle.

2.6.2.3 Rolling Resistance Force, F$_{RR}$

Rolling resistance is the resistance to motion of the rotating parts. At the most elementary level the rolling resistance force is mathematically expressed as

$$F_{RR} = \mu mg \tag{2.11}$$

where mg is the wheel load and μ is the rolling resistance coefficient. In some literatures, Equation (2.9) and Equation (2.11) are combined in one equation expressing the resistance to motion due to aerodynamics and coefficient of rolling friction as a function of velocity, which is called Davis' equation as in (Gillespie, 1992), and (Lu, et al., 2013).

2.6.2.4 Tractive force, F_T:

A train moves through the application of tractive force F_T which is produced at a driving wheel-rail interface; it is normally restricted in such a way $F_T \le F_{T\_Max}$, where F_{T_Max} is the tractive force beyond which slipping occurs as previously discussed. The maximum tractive force depends on the adhesion between a driving wheel and a rail, the greater the adhesion the greater the maximum tractive force. Usually the force needed to start slipping is greater than that needed to continue slipping.

2.6.3 Traction Energy Consumption Computation

In a computer simulation, the approach to calculate the tractive energy consumed by a train travelling from one station to the next is as summarized in step (i) through step (vii) below

I. Variables are initialized, e.g. $t = 0;\ v(t) = 0;\ s(t) = 0$;

II. Train acceleration is determined;

III. At every time step (dt), a value of velocity (v) and change in distance (ds) are calculated as $v = u + adt$ and $ds = udt + 0.5adt^2$, where u and v are previous and current velocities respectively and a is the acceleration;

IV. Resistance forces (F_{RR}, F_{grag} and, F_{grad}) are calculated, and then the tractive force (T_F) is obtained;

V. The output mechanical tractive power (P_{MT}), and input electrical tractive power (P_{ET}) are calculated as $P_{MT} = T_F v$ and $P_{ET} = \frac{P_{MT}}{\eta}$ respectively and then electrical tractive energy consumption (E_T) is computed as $E_T = P_{ET} dt$; where η is the input electrical tractive power (P_{ET}) to the output mechanical tractive power (P_{MT}) conversion efficiency;

VI. The data are stored; and

VII. The values of time (t), velocity (u), and distance (s) are updated as $t = t + dt$, $u = v$ and $s = s + ds$ respectively.

VIII. The cycle (step II to step VII) repeats till a vehicle reaches a stopping point.

These procedures are presented in a flow chart in Figure 2.17.

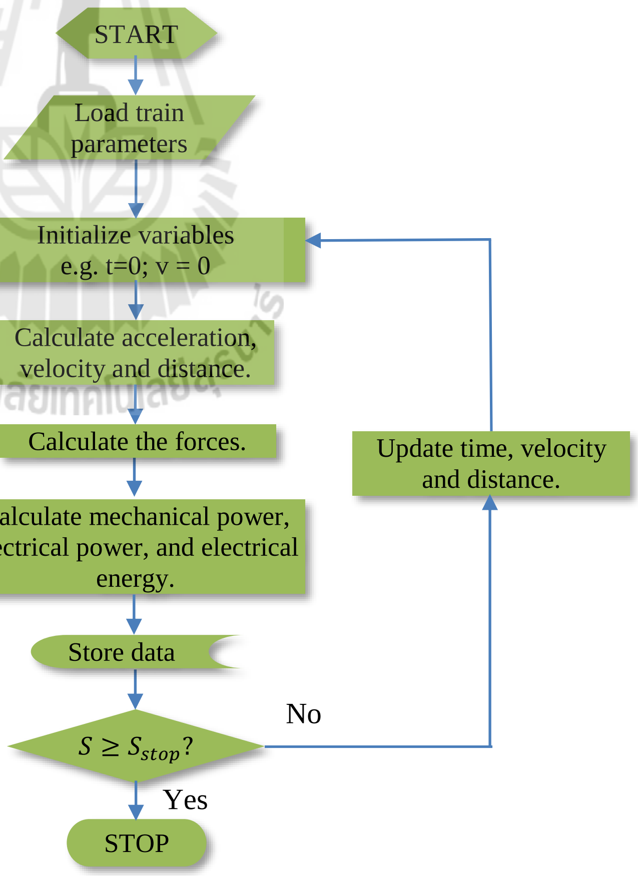

START
Load train parameters
Initialize variables e.g. t=0; v = 0
Calculate acceleration, velocity and distance.
Calculate the forces.
Calculate mechanical power, electrical power, and electrical energy.
Store data
$S \geq S_{stop}$?
No
Yes
STOP
Update time, velocity and distance.

Figure 2.17 Train energy consumption calculation flow chart

2.7 Induction Motor in Electric Traction Systems

The asynchronous or induction motor- especially in the robust form with squirrel cage rotor has always been the “work horse” of electric drive technology and thus the ideal traction means for railway technical engineers (Steimel, 2008).

The induction motor can be operated in three modes: motoring, generating, and plugging. If the stator terminals are connected to a three-phase supply, the stator windings will produce rotating magnetic field and the rotor will rotate in the direction of the stator rotating magnetic field. This is the natural (motoring) mode of operation. If the speed of rotor is higher than the synchronous speed (due to either the inertial of a drive system e.g. a vehicle going downhill or, decrease in synchronous speed e.g. when frequency of the supply is lowered) and the rotor rotates in the same direction as the stator rotating field, the motor will produce a generating torque—that is, a torque acting opposite to the rotation of the rotor (or acting opposite to the stator rotating magnetic field).

In electric traction, it is this this regenerative mode of the induction motor that is used to provide dynamic braking. The motor may be fed from a variable-frequency drive (VFD) or a variable-voltage-variable-frequency (VVVF) supply to control speed of the drive system. To stop the drive system, the frequency of the supply is gradually reduced, such that the instantaneous speed of the drive system is higher than the synchronous speed. As a result, the generating action of the induction motor will cause the power flow to reverse and the kinetic energy of the drive system will be fed back to the supply. If the drive system is required to stop very quickly, changing the motor’s terminal phase sequence will cause the stator rotating field to rotate opposite to the rotation of the rotor, producing the plugging operation (Sen, 2014).

2.8 Regenerative Braking

Generally there are two main types of braking in traction systems: dynamic (electric) braking and the more traditional friction braking. In dynamic braking, as a traction motor is turned into a generator, the generating torque provides the necessary braking force. Depending on how the generated current is used, the dynamic braking can be either rheostatic or regenerative braking. In rheostatic braking, the current is dissipated in banks of resistors. This type of braking is valuable on heavy-haul diesel-electric locomotives and some dc fed (either by third rail or catenary) electric trains that do not have sufficient onboard storage facilities, running on routes with extensive down grades. When regenerative braking is employed, instead of dissipating the train kinetic energy in the bank of resistors as heat energy, this energy is either feed back to the grid, or stored in the storage facilities such as capacitors and batteries. These storage facilities may be installed onboard (in the train) and/or at a stopping station.

Recent studies have shown that up to 40% of the energy supplied to electrical rail guided vehicles could be recovered through regenerative braking. The more frequently a train stops, the more it can benefit from regenerative braking. Apart from energy saving, regenerative braking reduces the need for friction brakes, and offer substantial savings in brake maintenance. The full-stop commuter services at Birmingham and Manchester in the United Kingdom are able to use regenerative braking. With regenerative braking being enabled, their disk brake pad life was around 18 months; when the electric braking was switched off, the pad life reduced to 18 days (Barrero, et al., 2008), and (Jiang, et al., 2014).

2.9 Optimization methods

In engineering practice, optimization is an act, process, or methodology of finding the best value(s) of the decision variable(s) under given conditions which define a set of available alternatives. Optimization normally aims at minimizing system cost, operational cost, or required effort. In some cases optimization may also aim at maximizing profit. A function of the decision variables is written to reflect what is to be optimized, and optimization techniques are used to search for the best values of the decision variables.

Optimization methods are mainly classified as traditional (or classical) and modern. Traditional optimization methods such as golden search, Newton Raphson, and steepest descent have no ability to escape from a local optimal, once any optimum point (be it local or global) is found, the methods do converge. In most cases, that optimum point will be the one closest to the starting point. On the other hand the modern optimization methods have great ability to escape from the local optimal and converge to a global optimal. This section discusses the following modern optimization methods: Genetic algorithms (GAs), Particle swarm optimization (PSO), and differential evolution (DE).

2.9.1 Genetic Algorithm

Genetic Algorithms (GAs) were first presented systematically by J. Holland in 1992. But, the idea of evolutionary computing was introduced in the 1960s by I. Rechenberg in his work “Evolution strategies” which was then developed by others (Holland, 1992) and (Sanghvi., et al, 2014).

GAs are inspired by the laws of natural selection and genetics. Thus they use the concept of Darwin’s theory of evolution, which is based on the rule of survival of the fittest (Vas, 1999). To solve a problem, GAs randomly initialize a population of individuals (chromosomes) and probabilistically modifies the population by three genetic operators: reproduction (selection), crossover and mutation. In such a way the population number is maintained with the fittest survivals. The GAs involve three basic steps i) Population initialization, ii) Fitness evaluation, and iii) New generation formation, summarized as follows:

I. Given a solution space, individuals (possible-solutions) are randomly created. An individual is characterized by fixed-length binary bit string which is called a chromosome.

II. The individuals are evaluated by means of fitness function

III. The new generation is then formed by applying the three genetic operators

- First a selection (reproduction) operator is applied where evaluated individuals are selected with probability proportional to their fitness values; the most commonly used technique is the 'roulette wheel selection' where the probability of an individual to be selected is $P_i = \frac{f_i}{\sum_{k=1}^{n} f_k}; \left(k = 1, 2, \ldots, n\right)$ where n is the population size, and f_i the fitness function. The selected individuals are reproduced (copied one or more times to obtain a mating pool) such that the population number is maintained.
- Secondly individuals in the mating pool are paired randomly to obtain couples (parents) and the crossover operator is applied.

- Crossover operator is then followed by a mutation operator. The mutation rate is normally kept low so that good chromosomes are preserved. Mathematically the mutation ensures that, given any population, the entire search space is connected (Vas, 1999).

IV. The cycle repeats form step ii until convergence. In computer simulation it is a common practice to use two parameters to decide if the convergence criterion is satisfied. These parameters are usually called tolerance function (TolFun) and stall generation limit (StallGenLimit). If the change in objective function value is less or equal to the TolFun, StallGenLimit times for consecutive iterations; then the algorithm stops. Maximum number of iterations and maximum allowed run time are normally integrated with the convergence criterion such that any of the three can stop the algorithm.

The GA flowchart can be presented as shown in Figure 2.18.

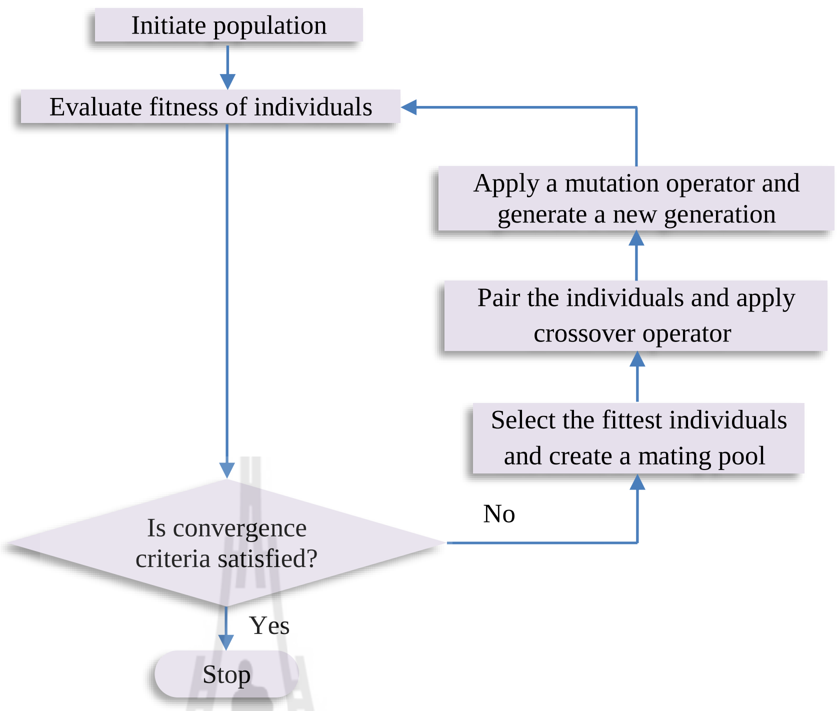

Initiate population
Evaluate fitness of individuals
Apply a mutation operator and generate a new generation
Pair the individuals and apply crossover operator
Select the fittest individuals and create a mating pool
No
Is convergence criteria satisfied?
Yes
Stop

Figure 2.18. Genetic algorithm flow chart

2.9.2 Particle Swarm Optimization

Particle swarm optimization (PSO) is an intelligent optimization algorithm, originally developed by Kennedy and Eberhart in 1995. The algorithm was developed based on the behavior of a colony or swarm of insects, such as ants, termites, bees, and wasps; a flock of birds; or a school of fish. The particle swarm optimization algorithm mimics the behavior of these social organisms. The word *particle* denotes, for example, a *bee* in a colony or a *bird* in a flock. Each individual or particle in a swarm behaves in a distributed way using its own intelligence and the collective or group intelligence of the swarm. As such, if one particle discovers a good path to *food*, the rest of the swarm will also be able to follow the good path instantly even if their location is far away in the swarm. Optimization methods based on swarm intelligence are called behaviorally inspired algorithms as opposed to the genetic algorithms, which are called evolution-based procedures (Zhang and Wei, 2009) and (Rao, 2009).

Like GA, PSO algorithm is based on population; it can be used to solve many complex optimization problems, which are nonlinear, non-differentiable and multi-modal. Unlike GA, PSO algorithm does not rely on genetic operators like selection, crossover and, mutation operators to operate individuals; it optimizes the population through information exchange among individuals. The most prominent merit of PSO is its fast convergence speed. In addition, PSO algorithm can be realized simply for less parameters need adjusting (Zhang and Wei, 2009) and (Yan., et al, 2013).

In multivariable optimization context, the swarm is assumed to be of specified or fixed size with each particle located initially at random locations in the multidimensional design (solution) space. Each particle having two characteristics: a *position* and a *velocity*; wanders around in the solution space and remembers the best position (in terms of the food source or objective function value) it has discovered. The particles communicate information on good positions to each other and adjust their individual positions and velocities based on the information received on the good positions. Thus the behavior of the swarm is based on a combination of three simple factors:

i. Cohesion—stick together
ii. Separation—don't come too close
iii. Alignment—follow the general heading of the swarm

The PSO algorithm therefore, simulates a random search in the design space for the optimal value of the objective function. As such, gradually over many iterations, the particles go to the target (or optimal value of the objective function) (Rao, 2009).

If the search space is d-dimensional, the ith particle of the swarm can be represented by a d-dimensional position vector as $x_i = (x_{i1}, x_{i2}, \ldots, x_{id})$. The velocity of the particle is denoted by $v_i = (v_{i1}, v_{i2}, \ldots, v_{id})$. The best visited position of the ith particle is presented as $P_{ibest} = \left(P_{i1}, P_{i2}, \ldots, P_{id}\right)$ and the best position explored so far is presented as $P_{gbest} = \left(P_{g1}, P_{g2}, \ldots, P_{gd}\right)$ (Lazinica, 2009).

The velocity (v_i) and position (x_i) of the ith particle are updated using Equation (2.12) and Equation (2.13) respectively.

$$V_{i,new} = \omega v_i + c_1 r_1 \left(P_{ibest} - x_i\right) + c_2 r_2 \left(P_{gbest} - x_i\right) \tag{2.12}$$

$$x_{i,new} = x_i + V_{i,new} \tag{2.13}$$

where c_1 and c_2 are positive acceleration constants (or learning factors) to control the influence of the cognitive and social information respectively and, $r_1$and r_2 are random numbers uniformly distributed between 0 and 1. Some researchers have shown that setting c_1 and c_2 equal to 2 gets the best overall performance. ω is the inertia weight which shows the effect of previous velocity vector on the new vector (Sadri and Suen, 2006), and (Lazinica, 2009).

Actually, the original PSO algorithm did not include the inertia weight ω, and it was found that usually the rate of decrease of particle velocities is not good enough to converge the particles to an optimal position as the iterative process progresses. Some of the particles could easily miss or skip the optimal position increasing the total computational time. The inertia concept was then introduced for the first time by Shi and Eberhart in 1999 to dampen the velocities over time (or iterations), enabling the swarm to converge more accurately and efficiently.

Usually, the value of ω is assumed to decrease linearly from 0.9 to 0.4 as the iterative process progresses. Thus, in the j^th iteration the value of ω can be given as

$$\omega_j = \omega_{initial} - \left(\omega_{initial} - \omega_{final}\right)\frac{j}{j_{max}} \tag{2.14}$$

or, in some literature it is given as

$$\omega_j = \left(\omega_{initial} - \omega_{final}\right)\left(\frac{j_{max} - j}{j_{max}}\right) + \omega_{final} \tag{2.15}$$

where j_{max} is the maximum number of iteration. When $j = 0$, (if generation or iteration counting starts from zero) the value of ω_j will be equal to $\omega_{initial}$, that is 0.9 and, when $j = j_{max}$, the value of ω_j will be equal to ω_{final}, that is 0.4. A larger value of ω promotes global exploration and a smaller value promotes a local search (fine tuning), this can be realized from Equation (2.12) and Equation (2.13). PSO algorithms with Linearly Decreasing Inertia Weight are commonly abbreviated as LDIW-PSO (Rao, 2009) and (Adewumi and Arasomwan, 2014).

Apart from linearly reducing the value of the inertia weight ω, another technique used to converge the particles in PSO algorithm is to set the maximum velocity of the particles and reduce its value as the iterative process progresses. Using such a technique, the inertia weight ω is normally assigned a constant number between 0 and 1. The authors in (Chan and Tiwari, 2007) have shown that, the algorithm will have a better chance to converge if the following two conditions are satisfied.

$$\begin{cases} 0 < c_1 + c_2 < 4 \\ \dfrac{(c_1 + c_2)}{2} - 1 < \omega < 1 \end{cases} \tag{2.16}$$

Over the years, several improvements to PSO algorithm have been proposed and proven successfully in various applications. Some of these improvements are presented in publications by (Yan., et al, 2013), (Zhang and Wei, 2009), (Liang., et al, 2010), (Mei., et al, 2010) and (Adewumi and Arasomwan, 2014). Generally, the PSO algorithm can be summarized as follows:

I. Given a solution space, *N* particles of the swarm are randomly initialized. There is no generally accepted number of particles needed in the optimization process. However, 20 – 50 particles are commonly used in literature (Adewumi and Arasomwan, 2014). A small swarm size will reduce the total number of function evaluations needed to find a solution. But with too small swarm size it is likely to take longer to find a solution or, in some cases, the algorithm may not be able to find a solution at all. Initially all particles have zero velocity, as iterations progress all particles will be moving towards the optimal point with a velocity given by the velocity-updating function, different PSO algorithms have different velocity-updating functions, the common function used to update the particle velocities is given in Equation (2.12);

II. For each particle, its fitness value $f(x_i)$ is computed using its current position x_i. The computed fitness value is compared to the particle's best value achieved so far $f(P_{ibest})$;

III. Particle's best position and value are updated if the particle's current position x_i is better than its best position found so far $f(P_{ibest})$. That is, assuming it is a minimization problem, then;

$if\ f(x_i) < f(P_{ibest})$; then $P_{ibest} = x_i\ and\ f(P_{ibest}) = f(x_i)$

IV. The best fitness of all particles are compared, and the particle with the best of the best fitness is considered to be at the global best position (P_{gbest});

V. Velocity of each particle is updated; and

VI. Each particle is moved to a new position.

VII. The cycle repeats form step II to step VI, until convergence criterion is satisfied. As previously mentioned, in computer simulation it is a common practice to use two parameters to define the convergence criterion. These parameters are usually called tolerance function (TolFun) and stall generation limit (StallGenLimit). If the change in objective function value is less or equal to the TolFun, StallGenLimit times for consecutive iterations; then the algorithm stops. Maximum number of iterations or maximum allowed run time are normally integrated with the convergence criterion to stop the algorithm.

The PSO algorithm procedure is shown as a flow chart in Figure 2.19.

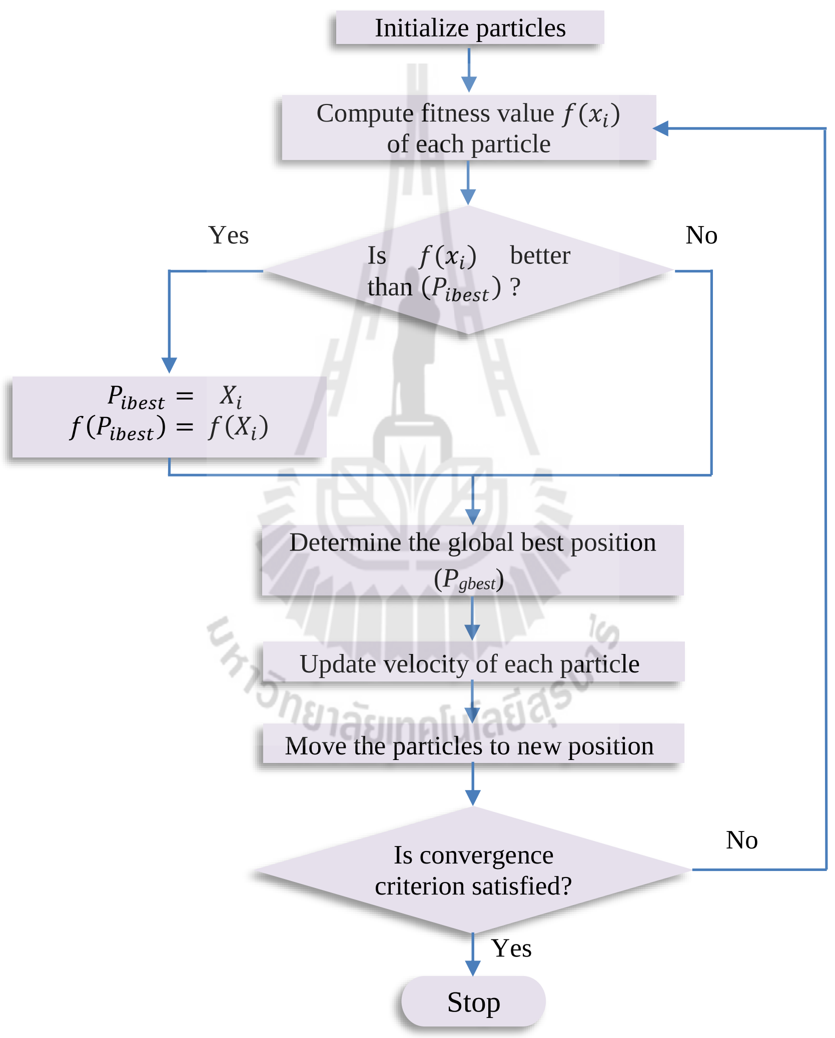

Initialize particles
Compute fitness value $f(x_i)$ of each particle
Is $f(x_i)$ better than (P_{ibest}) ?
Yes
No
$P_{ibest} = X_i$
$f(P_{ibest}) = f(X_i)$
Determine the global best position (P_{gbest})
Update velocity of each particle
Move the particles to new position
Is convergence criterion satisfied?
No
Yes
Stop

Figure 2.19 Particle swarm algorithm flow chart

2.9.3 Differential Evolution

Differential evolution (DE) proposed by Storn and Price in 1995, is a very simple but very powerful stochastic global optimizer for continuous search domain. It has been proven a robust global optimizer and has been successfully applied to many global optimization problems with superior performance in both widely used benchmark function and real-world applications. Like all Evolutionary Algorithms (EAs), DE is a stochastic population-based search method that employs repeated cycles of recombination and selection to guide the population towards the vicinity of global optimal. However, the DE algorithm is unique in the sense that it uses a *differential mutation* operator (before crossover) to perturb parent vectors in the current generation. A parent vector from the current generation is called *target* vector, a mutant vector obtained through the differential mutation operation is known as *donor* vector and finally an offspring formed by recombining (crossing over) the target with the donor vector is called *trial* vector.

A one-to-one greedy selection scheme between the trial vectors and the target vectors produces parent vectors of the next generation. Depending on the way the parent vector is mutated to generate the donor vector and eventually the trial vector, there exist many trial vector generation strategies and consequently many DE variants. The conceptual and algorithmic simplicity, high convergence characteristics and robustness of DE has made it one of the popular techniques for real-valued parameter optimization (Jeyakumar and Velayutham, 2009) and, (Das and Suganthan, 2011).

One of the most frequently used mutation strategy is the DE/rand/1/bin, according to which; DE starts by randomly creating a population of *N* individuals in a d-dimensional solution space. In the current iteration the ith individual of the population is represented by a d-dimensional vector as $x_{i,g} = (x_{1,i,g}, x_{2,i,g}, \ldots, x_{d,i,g})$, where g is the current iteration (generation), if G is the maximum number of generations then, $g = 1, 2, \ldots, G$ and $i = 1, 2, \ldots, N$. Once the population is initialized; DE employs repeated cycles of mutation, crossover and selection to guide the population towards a global optimal.

According to DE/rand/1/bin version, DE creates a mutant vector $v_{i,g}$ for each target vector by adding a weighted difference between two randomly selected vectors $x_{l,g}$ and $x_{m,g}$ to a third randomly selected vector $x_{k,g}$ as

$$v_{i,g} = v_{k,g} + \eta(v_{l,g} - v_{m,g}) \tag{2.17}$$

where $i \neq k \neq l \neq m$. The mutation (scaling) factor η is a positive user defined real number, typically < 1, used to avoid search stagnation (Chengfu, et al., 2012), and (Ahadzadeh and Menhaj, 2014).

Following the mutation phase, to enhance diversity in the searching process, the crossover operation is performed on $x_{i,g}$ (target vector) and $v_{i,g}$ (mutant vector) to generate a trial vector $u_{i,g} = (u_{1,i,g}, u_{2,i,g}, \ldots, u_{d,i,g})$. Binomial (uniform) is the most commonly used crossover method in DE, in which the number of trial vector parameters inherited from the mutant vector has a (nearly) binomial distribution. The scheme may be outlined as

$$u_{i,j,g} = \begin{cases} v_{i,j,g} & \textit{if rand} \leq \textit{Cr or } j = \textit{jrand} \\ x_{i,j,g} & \textit{otherwise} \end{cases} \tag{2.18}$$

where $i \in [1,2,\ldots, N]$, $j \in [1, 2,\ldots, d]$, rand is a uniformly distributed random number in the interval 0 to 1, *jrand* is a randomly selected integer from 1 to d and *Cr* is crossover probability in the interval 0 to 1 (Das and Suganthan, 2011) and, (Ahadzadeh and Menhaj, 2014).

There after the selection operator is applied to compare the fitness values of the target vector $x_{i,g}$ and the trial vector $u_{i,g}$ and determine which of the two vectors survives for the next generation (g+1). If it is a minimization problem, the scheme may be outlined as

$$x_{i,g+1} = \begin{cases} u_{i,g} & \text{if } f\left(u_{i,g}\right) \leq f\left(x_{i,g}\right) \\ x_{i,g} & \text{otherwise} \end{cases} \tag{2.19}$$

If the trial vector (child) $u_{i,g}$ has a fitness value smaller than or equal to that of the target vector (parent) $x_{i,g}$ it replaces the corresponding in the next generation, otherwise the parent survives to the next generation.

Generally, the DE algorithm can be summarized as follows:

I. Given a solution space, individuals (possible-solutions) are randomly initialized;

II. Fitness values of the individuals are determined;

III. The new generation is then formed by applying mutation, crossover and selection operators;

- Firstly a mutation operator is applied, and mutant vectors (donors) are obtained,
- Secondly a crossover operator is applied between mutant vectors and parent vectors to obtain trial vectors (children),

- Crossover operator is then followed by a selection operator which determines which individuals from children and parents' population survive to the next generation.

IV. Step II and III are repeated until convergence criterion is satisfied. Like in GA and PSO algorithms, tolerance function (TolFun) and stall generation limit (StallGenLimit) parameters are normally used to define convergence criterion. Maximum number of iterations and maximum allowed run-time are normally integrated with the convergence criterion such that any of the three that is reached first, will stop the algorithm.

The basic DE algorithm procedure is presented in Figure 2.20

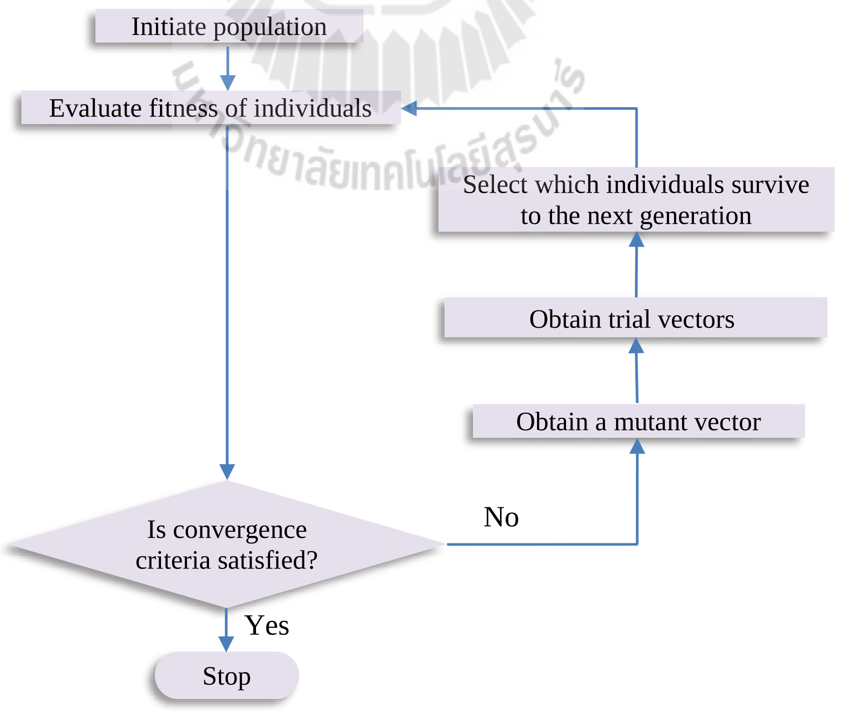

Initiate population
Evaluate fitness of individuals
Select which individuals survive to the next generation
Obtain trial vectors
Obtain a mutant vector
No
Is convergence criteria satisfied?
Yes
Stop

Figure 2.20 Differential evolution algorithm flow chart

CHAPTER III

SPEED PROFILE OPTIMIZATION RESULTS

3.1 Introduction

Optimal train designing aiming at minimizing train weight, auxiliary loads, aerodynamic and frictional forces without sacrificing passenger comfortability and train's passenger carrying capacity is definitely the best thing to do; in order to minimize energy consumption per passenger-kilometer. However, given train parameters; energy consumed by a train travelling from one station to the next, and eventually one terminal to the other is far from being fixed. Since the number of stops and subsequent speed profiles (driving cycles) have an immense influence on energy demand.

In this chapter, the summarized speed profile optimization results are presented. The optimization was done in MATLAB® environment using three optimization algorithms: MATLAB® built in GA and, developed DE and PSO; which had been tested using three benchmark functions: Ackley's, Rosenbrock's and, Rastrigin's functions; and proven successful. The algorithms testing results are presented in appendix B. Given the two routes: TRL and TAZARA (TZR); the three optimization techniques were then used to search for optimal speed profiles for each route taking into account time, velocity and acceleration constrains. Comparing the obtained optimal speed profiles, the best speed profile for each route was established. The optimization results for every section (interstation distance) of each route are presented in appendix C. It should be noted that, in the appendix B and appendix C the MATLAB® built in GA, plots only the mean and the best value of the fitness function, while the developed DE and PSO were programed to plot the worst, mean, and best values of the fitness function. The worst fitness value was added in the plot for the purpose of this thesis work so as to evaluate (i) How much energy would have been consumed with the worst speed profile and, (ii) the convergence of the individuals (or particles). For each algorithm, the number of individuals (or particles) was set to 50, and the number of maximum iteration was also set to 50 as it can be seen in the “Options.m” file in appendix D. For a particular iteration if the worst and mean fitness values are not plotted that means at least one of the individuals (or particles) was invalid (did not comply with all the constrains) and hence it was discarded (its fitness value was set to infinite).

3.2 Objective Function

In minimizing energy to be consumed by a tram travelling form one station to the next, the objective function was written as

$$y = min \int_{t_{start}}^{t_{stop}} \left(\frac{(ma(t) + F_R)v(t)}{\eta} + AL \right) dt \tag{3.1}$$

Subject to

$$v_m \le 13.9$$

$$0.3 \le a \le 0.6$$

$$(0.4 \le b_{point} \le 1) \quad ; v_b = b_{point} v_m$$

$$t_{bs} \le \begin{cases} 150/km & if\ distance \le 1km \\ 120/km & otherwise \end{cases}$$

where t_{start} is the time at which the tram starts moving from its current station, t_{stop} is the time at which the tram stops at the next station, m is the tram effective mass (kg), a is the acceleration (m/s^2), F_R is the force opposing the motion (N), η is the electrical – mechanical tractive power conversion efficiency, v is the velocity (m/s), AL is the auxiliary load, dt is the change in time (s); v_m is the maximum velocity (m/s), b_{point} is the braking point, v_b is the braking velocity (m/s), and t_{bs} is the traveling time between stations (sec).

The objective was to obtain optimal values of three parameters: maximum velocity (V_m), starting acceleration (a), and braking point (b_{point}) as shown in Figure 3.1. The optimal values of these parameters define the optimal speed profile hence, minimum energy consumption.

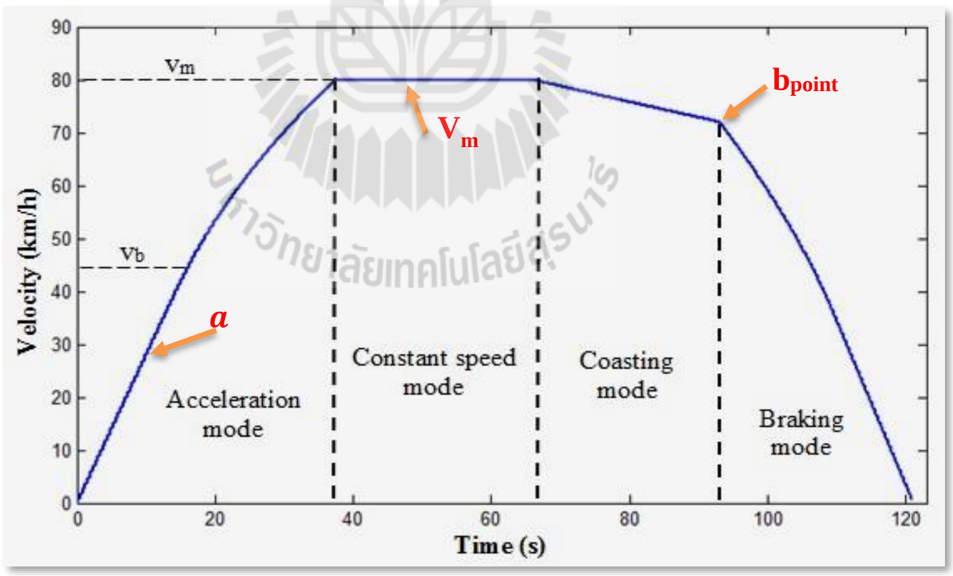


Figure 3.1 Typical train speed profile

3.3 Routes and Train Details

The two routes' stations and distances between them are as shown in Figure 3.2. The TRL route covers 12.05 km from Central Station to Ubungo-Maziwa with a total of eight (8) stations; and the TAZARA route covers 21 km from TAZARA Station to Mwakanga with a total of ten (10) stations.

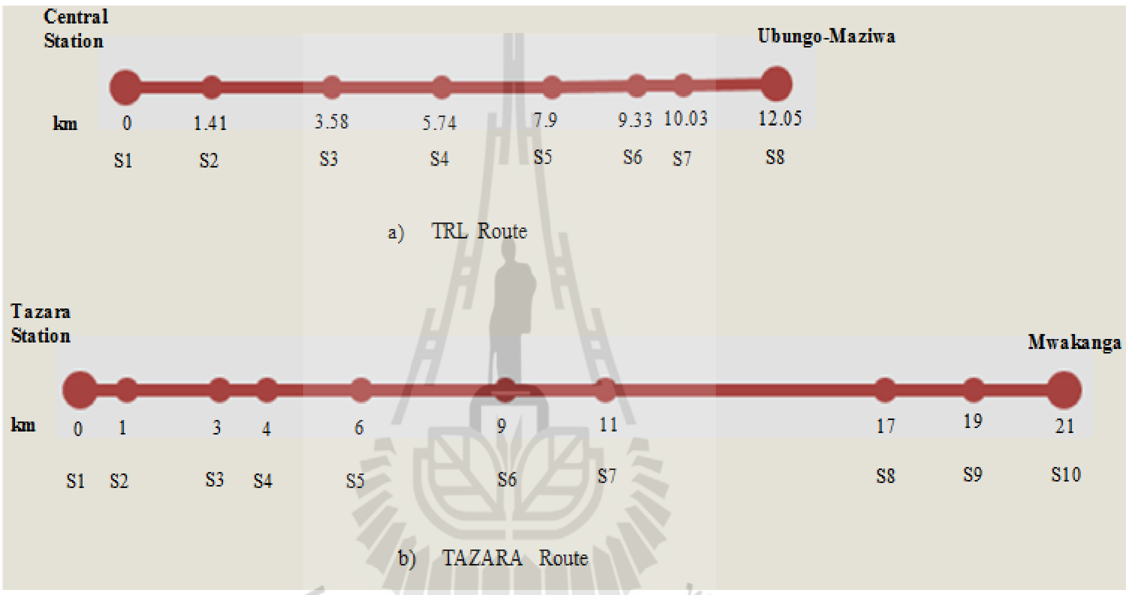
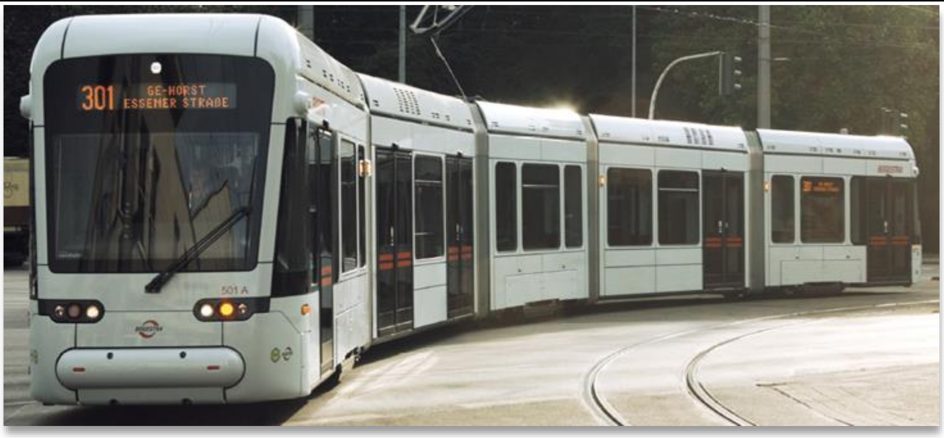
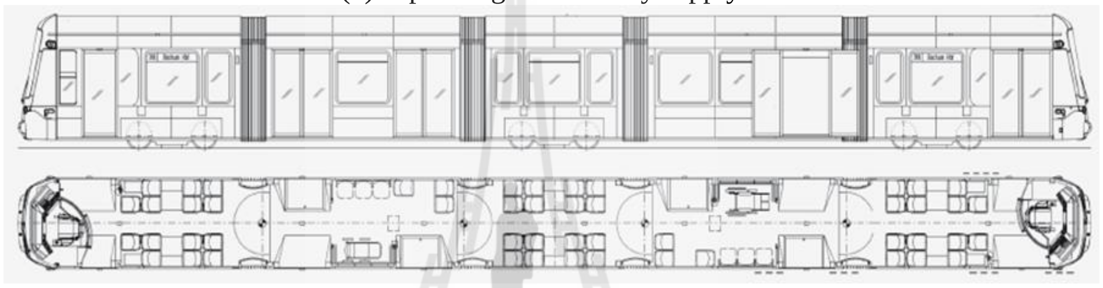


Figure 3.2 TRL and TAZARA commuter train routes

For this research work a Stadler variobahn bidirectional tram as shown in Figure 3.3 was chosen. The tram specifications are given in Table 3.1. In this thesis, this catenary fed tram is assumed to be powered by lithium titanate batteries.



(a) Operating via catenary supply



(b) Construction overview

Figure 3.3 Stadler variobahn bidirectional tram (Stadler, 2014)

Table 3.1 Stadler variobahn tram details

Attribute		Value
Maximum vehicle speed		70 km/h
Passenger capacity	56 Sitting	312
	256 Standing (8 pers/m^2)	
Tram weight		38800 kg
Catenary's voltage		600/750 V
Drive/ Powered wheel /trailer wheel		8 x 45 kW/8/4
Vehicle length/ width/ height		29.62 m /2.3 m/3.35 m
Gauge		1000 mm

Taking average weight of a person as 60.7 kg , train rolling resistance coefficient as 0.006, and aerodynamic drag coefficient as 0.6 (Barrero, et al., 2008), tram parameters used in optimization were taken as given in Table 3.2.

Table 3.2 Tram parameters used in optimization

Attribute	Value
Vehicle effective weight (0.11 rolling allowance, passenger load, batteries)	62830 kg
Rolling resistance coefficient	0.006
Aerodynamic drag coefficient	0.6
Frontal area	7.7 m^2
Energy conversion efficiency	0.6
Average auxiliary load	10 kW

3.4 Optimal Speed Profile for TRL Route

As mentioned earlier, GA, DE and PSO optimization techniques were used to search for the optimal values of the maximum velocity, starting acceleration, and braking point. Comparing the obtained speed profiles, the best speed profile for the TRL route was established. The optimization results for TRL route were as presented in Table 3.3 through Table 3.6 and, in Figure 3.4. Figure 3.5 was plotted just to give a clue to the size of the required power source.

Table 3.3 TRL route; speed profile values as optimized by GA

Stations	From -To	S1-S2	S2-S3	S3-S4	S4-S5	S5-S6	S6-S7	S7-S8
	Distance(km)	**1.41**	**2.17**	**2.16**	**2.16**	**1.43**	**0.7**	**2.02**
Parameters	Maximum velocity (km/h)	44.07	39.08	39.29	39.22	42.56	33.95	40.64
	Stating acceleration (m/s^2)	0.54	0.54	0.51	0.53	0.50	0.49	0.54
	Braking point	0.58	0.40	0.40	0.40	0.61	0.91	0.40
Energy consumed (kWh)		3.45	4.88	4.86	4.86	3.49	2.08	4.57
Travelling time (min)		2.82	4.34	4.32	4.32	2.86	1.75	4.04
Total energy consumed (kWh)								**28.19**
Total travelling time (min)								**24.44**

Table 3.4 TRL route; speed profile values as optimized by DE

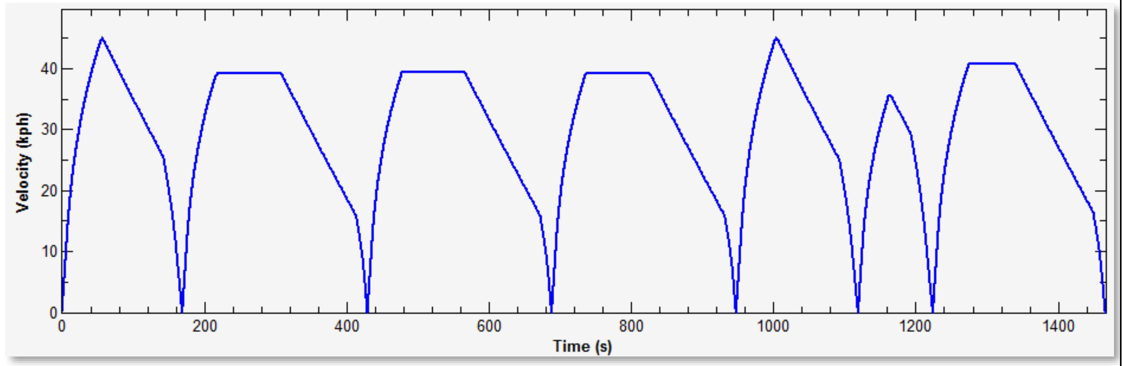
Stations	From -To	S1-S2	S2-S3	S3-S4	S4-S5	S5-S6	S6-S7	S7-S8
	Distance(km)	**1.41**	**2.17**	**2.16**	**2.16**	**1.43**	**0.7**	**2.02**
Parameters	Maximum velocity (km/h)	45.05	39.11	39.34	39.19	44.91	35.62	40.68
	Stating acceleration (m/s^2)	0.50	0.53	0.52	0.54	0.54	0.53	0.49
	Braking point	0.56	0.40	0.40	0.40	0.55	0.82	0.40
Energy consumed (kWh)		3.44	4.88	4.86	4.86	3.47	2.02	4.57
Travelling time (min)		2.82	4.34	4.31	4.32	2.86	1.75	4.04
Total energy consumed (kWh)								**28.09**
Total travelling time (min)								**24.44**

Table 3.5 TRL route; speed profile values as optimized by PSO

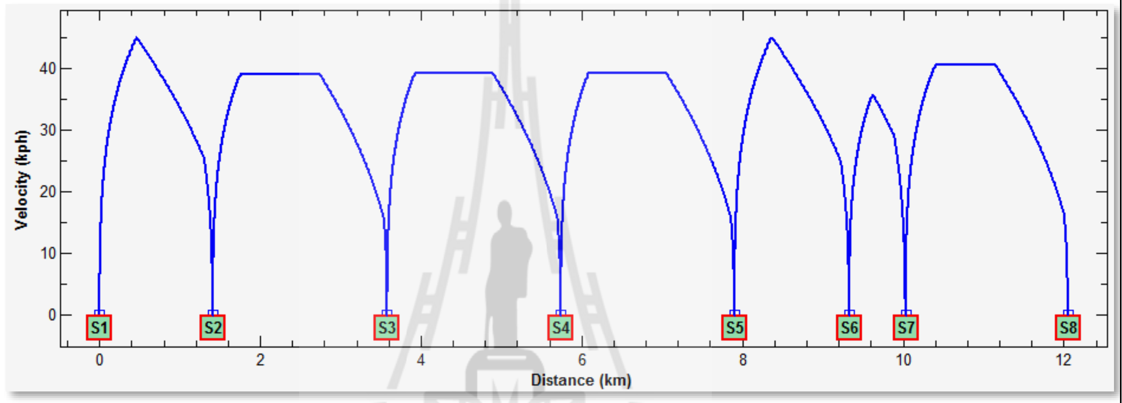
Stations	From -To	S1-S2	S2-S3	S3-S4	S4-S5	S5-S6	S6-S7	S7-S8
	Distance(km)	**1.41**	**2.17**	**2.16**	**2.16**	**1.43**	**0.7**	**2.02**
Parameters	Maximum velocity (km/h)	44.47	39.11	39.33	44.15	41.70	35.91	42.40
	Stating acceleration (m/s^2)	0.40	0.54	0.48	0.30	0.42	0.48	0.39
	Braking point	0.65	0.40	0.40	0.40	0.70	0.81	0.40
Energy consumed (kWh)		3.55	4.88	4.86	4.90	3.59	2.02	4.58
Travelling time (min)		2.82	4.34	4.32	4.32	2.86	1.75	4.04
Total energy consumed (kWh)								**28.38**
Total travelling time (min)								**24.44**

Table 3.6 TRL route; the optimal speed profile values

Stations	From -To	S1-S2	S2-S3	S3-S4	S4-S5	S5-S6	S6-S7	S7-S8
	Distance(km)	**1.41**	**2.17**	**2.16**	**2.16**	**1.43**	**0.7**	**2.02**
Parameters	Maximum velocity (km/h)	45.05	39.11	39.34	39.19	44.91	35.62	40.68
	Stating acceleration (m/s^2)	0.50	0.54	0.52	0.54	0.54	0.53	0.49
	Braking point	0.56	0.40	0.40	0.40	0.55	0.82	0.40
Energy consumed (kWh)		3.44	4.88	4.86	4.86	3.47	2.02	4.57
Travelling time (min)		2.82	4.34	4.31	4.32	2.86	1.75	4.04
Optimization Method		DE	PSO	DE	DE	DE	DE	DE
Total energy consumed (kWh)								**28.09**
Total travelling time (min)								**24.44**



(a) Optimal speed (kph) against time (s) for TRL route



(b) Optimal speed (kph) against distance (km) for TRL route

Figure 3.4 Optimal speed profile for TRL route

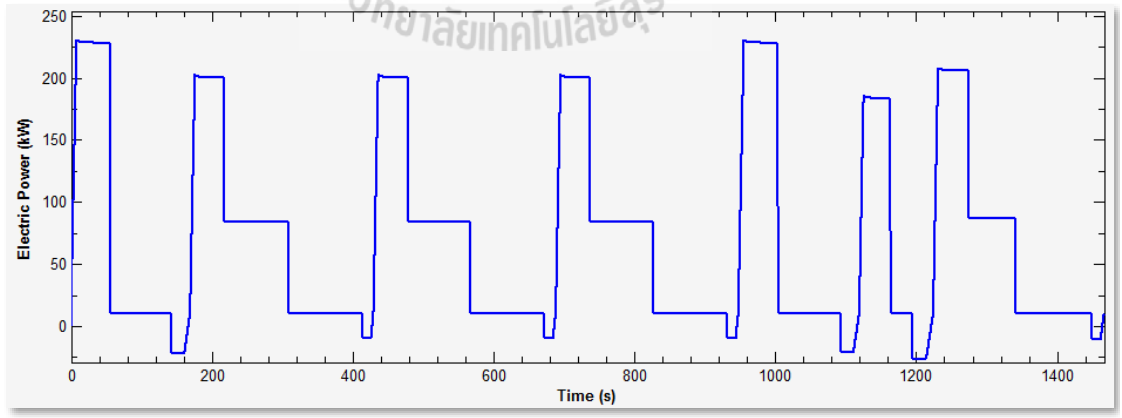


Figure 3.5 Tram power demand corresponding to the TRL route optimal speed profile

3.5 Optimal Speed Profile for TAZARA Route

In a similar fashion to what was done with the TRL route, the same was done with the TAZARA route. The GA, DE and PSO optimization techniques were used to search for the optimal values of the maximum velocity, starting acceleration, and braking point. Comparing the obtained speed profiles, the best speed profile for TAZARA route was established. The optimization results for TAZARA route were as presented in Table 3.7 through Table 3.10 and, in Figure 3.6. Figure 3.7 was plotted just to give a clue to the size of the required power source.

Table 3.7 TAZARA route; speed profile values as optimized by GA

Stations		S1-S2	S2-S3	S3-S4	S4-S5	S5-S6	S6-S7	S7-S8	S8-S9	S9-S10
	From -To	**S1-S2**	**S2-S3**	**S3-S4**	**S4-S5**	**S5-S6**	**S6-S7**	**S7-S8**	**S8-S9**	**S9-S10**
	Distance(km)	**1**	**2**	**1**	**2**	**3**	**2**	**6**	**2**	**2**
Parameters	Maximum velocity (km/h)	35.31	39.32	33.71	40.38	35.32	40.61	35.42	40.68	40.67
	Stating acceleration (m/s^2)	0.54	0.48	0.51	0.48	0.53	0.51	0.53	0.51	0.48
	Braking point	0.50	0.45	0.55	0.42	0.41	0.41	0.40	0.41	0.41
Energy consumed (kWh)		2.40	4.54	2.42	4.53	6.64	4.53	13.05	4.53	4.53
Travelling time (min)		2.50	4.00	2.50	4.00	6.00	4.00	11.08	4.00	4.00
Total energy consumed (kWh)										**47.17**
Total travelling time (min)										**42.08**

Table 3.8 TAZARA route; speed profile values as optimized by DE

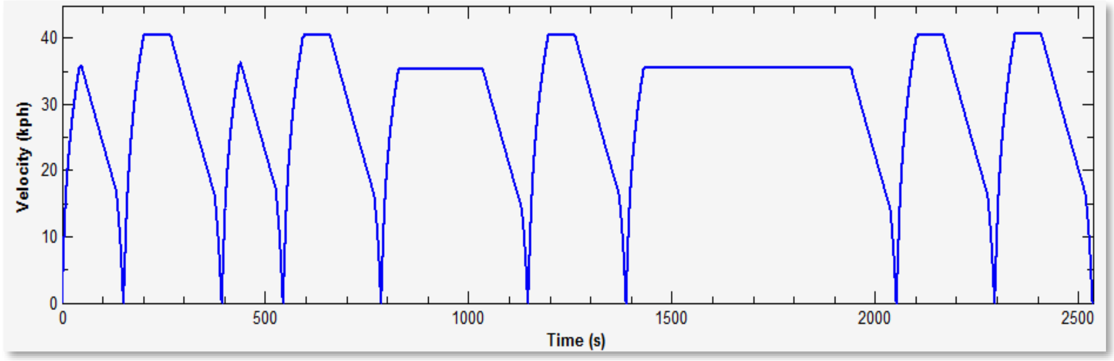
Stations	**From -To**	**S1-S2**	**S2-S3**	**S3-S4**	**S4-S5**	**S5-S6**	**S6-S7**	**S7-S8**	**S8-S9**	**S9-S10**
	Distance(km)	**1**	**2**	**1**	**2**	**3**	**2**	**6**	**2**	**2**
Parameters	Maximum velocity (km/h)	36.14	40.77	36.48	40.78	35.42	40.77	35.53	40.80	40.77
	Stating acceleration (m/s^2)	0.49	0.52	0.54	0.54	0.50	0.50	0.51	0.48	0.51
	Braking point	0.48	0.40	0.47	0.40	0.40	0.40	0.40	0.40	0.40
Energy consumed (kWh)		2.40	4.52	2.40	4.52	6.63	4.52	13.05	4.52	4.52
Travelling time (min)		2.49	4.00	2.50	4.00	6.00	4.00	11.06	4.00	4.00
Total energy consumed (kWh)										**47.10**
Total travelling time (min)										**42.05**

Table 3.9 TAZARA route; speed profile values as optimized by PSO

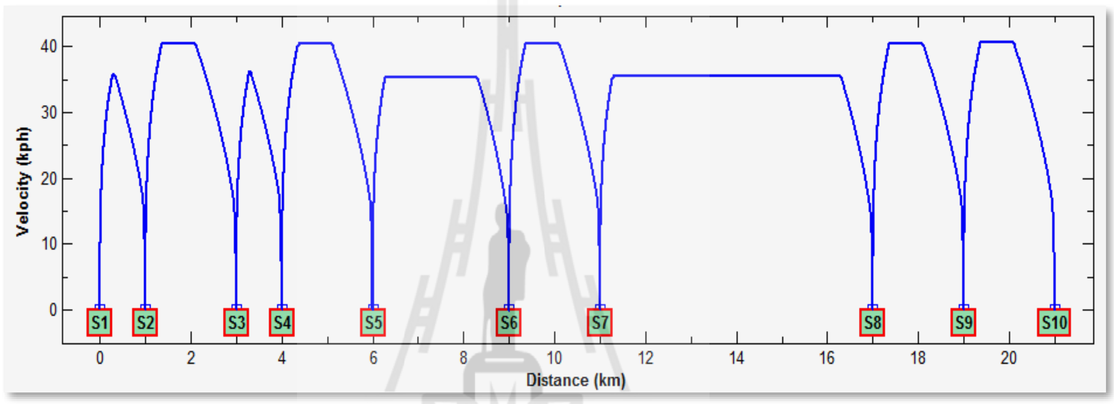
Stations	From -To	S1-S2	S2-S3	S3-S4	S4-S5	S5-S6	S6-S7	S7-S8	S8-S9	S9-S10
	Distance(km)	1	2	1	2	3	2	6	2	2
Parameters	Maximum velocity (km/h)	36.36	40.77	36.48	40.85	37.37	40.77	35.67	40.77	40.77
	Stating acceleration (m/s^2)	0.51	0.51	0.54	0.48	0.30	0.51	0.30	0.50	0.49
	Braking point	0.48	0.40	0.48	0.40	0.40	0.40	0.40	0.40	0.40
Energy consumed (kWh)		2.40	4.52	2.40	4.52	6.66	4.52	13.07	4.52	4.52
Travelling time (min)		2.50	4.00	2.50	4.00	5.99	4.00	11.21	4.00	4.00
Total energy consumed (kWh)										**47.14**
Total travelling time (min)										**42.19**

Table 3.10 TAZARA route; the optimal speed profile values

Stations	From -To	S1-S2	S2-S3	S3-S4	S4-S5	S5-S6	S6-S7	S7-S8	S8-S9	S9-S10
	Distance(km)	**1**	**2**	**1**	**2**	**3**	**2**	**6**	**2**	**2**
Parameters	Maximum velocity (km/h)	36.14	40.77	36.48	40.78	35.42	40.77	35.53	40.80	40.77
	Stating acceleration (m/s^2)	0.49	0.52	0.54	0.54	0.50	0.50	0.51	0.48	0.51
	Braking point	0.48	0.40	0.47	0.40	0.40	0.40	0.40	0.40	0.40
Energy consumed (kWh)		2.40	4.52	2.40	4.52	6.63	4.52	13.05	4.52	4.52
Travelling time (min)		2.49	4.00	2.50	4.00	6.00	4.00	11.06	4.00	4.00
Method		DE	DE	DE	DE	DE	DE	DE	DE	DE
Total energy consumed (kWh)										**47.10**
Total travelling time (min)										**42.05**



(a) Optimal speed (kph) against time (s) for TAZARA route



(b) Optimal speed (kph) against distance (km) for TAZARA route

Figure 3.6 Optimal speed profile for TAZARA route

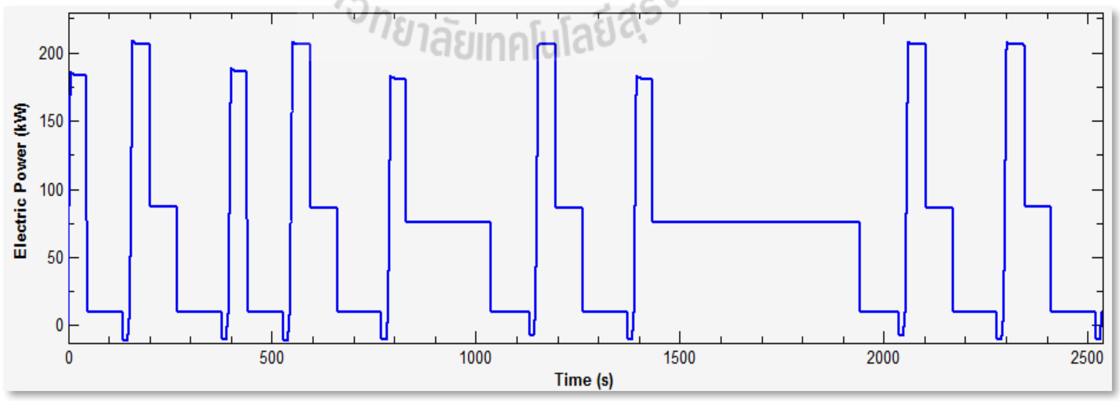


Figure 3.7 Tram power demand corresponding to the TAZARA route optimal speed profile

3.6 Discussion

From the optimal speed profiles for TRL and TAZARA routes, the following can be observed:-

- Though the differences in the best objective function values were very small (in fact immeasurably small), DE seemed to outperform GA and PSO. Looking at it from another angle, the fact that the best objective function values were very close (almost the same) implies that, the found position (variable vector), was real the best position in the given solution space.
- As explained previously, the MATLAB® built in GA, plots the mean and the best value of the fitness function, while the developed DE and PSO were programed to plot the worst, mean, and best values of the fitness function. And, invalid individuals were discarded (their fitness values were set to infinite). Since DE does a one-to-one greedy selection between the trial vectors and the target vectors to generate parent vectors for the next generation as explained in section 2.9.3; the only way an invalid individual will go to the next generation is if it was compared to another invalid individual, otherwise all individuals in the next generation will be valid and there will be no infinite value of a fitness function. This is why; in most cases DE plots all the three values: the worse, the mean, and the best fitness values while, GA and PSO plots only the best fitness values due to the presence of an infinite value in a given iteration as it can be realized in appendix C. This is

also a reason for DE to perform better than GA and PSO especially if a function is noisy. However, if a function is not noisy, PSO has a better chance to outperform DE and GA due to the fact that the swarm will be easily directed towards the global optimal, this can be realized in appendix B.

- Seeing that in most cases, DE plots all the three values: the worse, the mean, and the best fitness values, the significance of optimization can be well realized from DE fitness value plots as given in appendix C. For example in Figure C.2, the difference between the worst and best value is about 1 kWh.
- The longer the interstation distance, the higher the energy consumption, and travelling time. This is very obvious.
- With the same interstation distances; the optimal speed profile, energy consumption, and travelling time were also more or less the same. This implies that there was a single global optimal and the optimization method(s) was good enough to find it.
- With short interstation distance, the entire time band was used. For the same time constrain; acceleration and deceleration time loss makes time more critical if the interstation distance is short.
- For very short interstation distances, coasting was preferred over cruising, i.e. going to high speed and then free-wheel (coast) was better than going to average speed and cruise. This may also imply that the

longer the coasting the less the energy consumed, that’s why in most cases the entire time band was utilized.

- If time constrain is tight, decreasing the starting acceleration or maximum speed, or both; will decrease the peak power on the expense of decreasing the coasting time, which will result to increase in energy consumption. Depending on how much is the increase in energy consumption, this may be done if the power source has low power density. However, for this thesis work that was unnecessary.

CHAPTER IV

TRAM MOVEMENT RESULTS AND ANALYSIS

4.1 Introduction

In this chapter tram movement simulation results are presented and discussed alongside the passenger travel fuel cost and emission. Having got the best speed profiles for each of the two tram routes (in chapter III), the tram was then modelled as a BEMU and simulated using MATLAB®. For each route there are two directions, these directions were termed as *odd* and *even*. A direction is odd if a tram is moving away from the city (main station), and even if the tram is moving towards the city center. And for convenient purposes, stations were named as S1, S2,…Sn, where S1 is the main station, in this case the terminal station in the city. Charging points were assumed to be installed at every terminal station and at some of the intermediate stations of the TAZARA route as desired. If an intermediate station is not a charging station, the tram stops (dwells) for one minute, and if it is a charging station, the tram dwells for one minute plus time reserved. Time reserved is the difference between maximum allowed time window and the actual time used. At a terminal station the tram dwells for ten minutes within which batteries are charged to full charge. To easily change the route and direction during the simulation, a graphical user interface as shown in Figure 4.1 was created

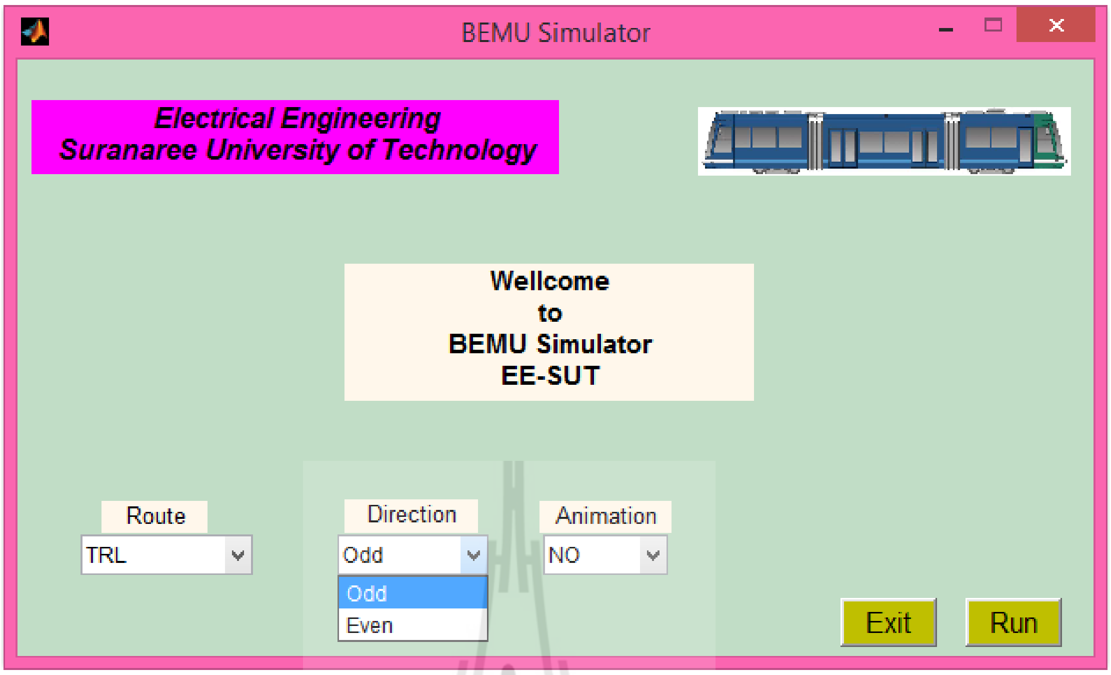


Figure 4.1 Graphical user interface for BEMU simulation

4.2 Tram and Battery Details

Tram and battery details used in the simulation were as given in Table 4.1 and Table 4.2 respectively.

Table 4.1 Tram details used in simulation

Attribute	Value
Tram effective weight	62830 kg
Rolling resistance coefficient	0.006
Aerodynamic drag coefficient	0.6
Frontal area	7.7 m^2
Energy conversion efficiency	0.6
Average auxiliary load	10 kW
Passenger capacity	312

Table 4.2 Lithium titanate battery details used in simulation (Altairnanno)

Attribute	**Value**	**Remarks (Battery pack value)**
Voltage range/nominal	17 - 27.5 V / 24 V	459 - 742.5 V / 648 V
Nominal capacity	60 Ah	120 Ah
Typical discharge energy at 1C rate,25°C, CCCV charging	1.4 kWh	77.76 kWh
Maximum continuous charge/discharge current	360 A / 360 A	720 A / 720 A
Pulse charge/discharge current (10 sec pulse)	Up to 600 A	Up to 1200 A
Internal impedance	4 mΩ	54 mΩ
Weight	27.4 kg	1480 kg
Life characteristics		
Cycle life at 2C charge and 2C discharge, 100% DoD, 25°C		>16,000 to 80% initial capacity
Cycle life at 2C charge and 2C discharge, 100% DoD, 55°C		>4,000 to 80% initial capacity

A battery pack consisted of two (2) modules in parallel, where a single module consisted of twenty seven (27) batteries in series. Thus, the battery pack had a nominal energy capacity of 77.76 kWh at 648 V. At a charging station batteries are charged at 4.5C rate if the SoC is below 80%, and at 3C rate otherwise.

4.2.1 BEMU on TRL Route

As shown in Figure 4.2, the TRL route is 12.05 km long running from Central-station (S1) to Ubungo-maziwa (S8). Charging points were assumed to be installed at the terminal stations S1 and S8, where batteries are fully charged. Tram movement from S1 to S8 was termed as *odd* direction while that from S8 to S1 was termed as *even* direction as previously explained. Simulation results for the odd direction in which the tram starts moving from S1 with fully charged batteries all the way to S8 (stopping for one minute at each intermediate station), were as presented in Figure 4.3 through Figure 4.6, the simulation results are also given in Table 4.3. When the tram reaches at a terminal station S8, batteries are fast charged to full charge within ten minutes. As previously mentioned, batteries are charged at 4.5C rate if the SoC is below 80%, and at 3C rate otherwise. This can be seen in Figure 4.4 and Figure 4.5.

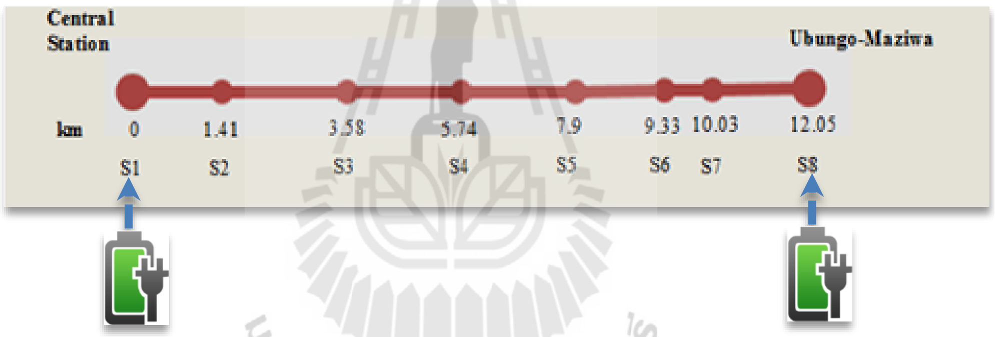


Figure 4.2 TRL route, charging points installed at terminal stations

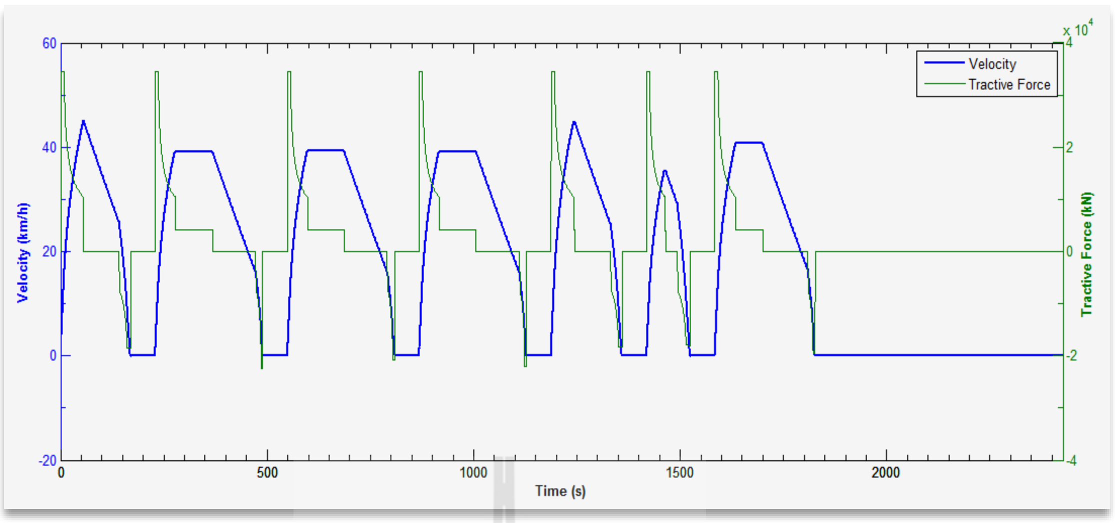


Figure 4.3 TRL route, odd direction, velocity and tractive force profiles

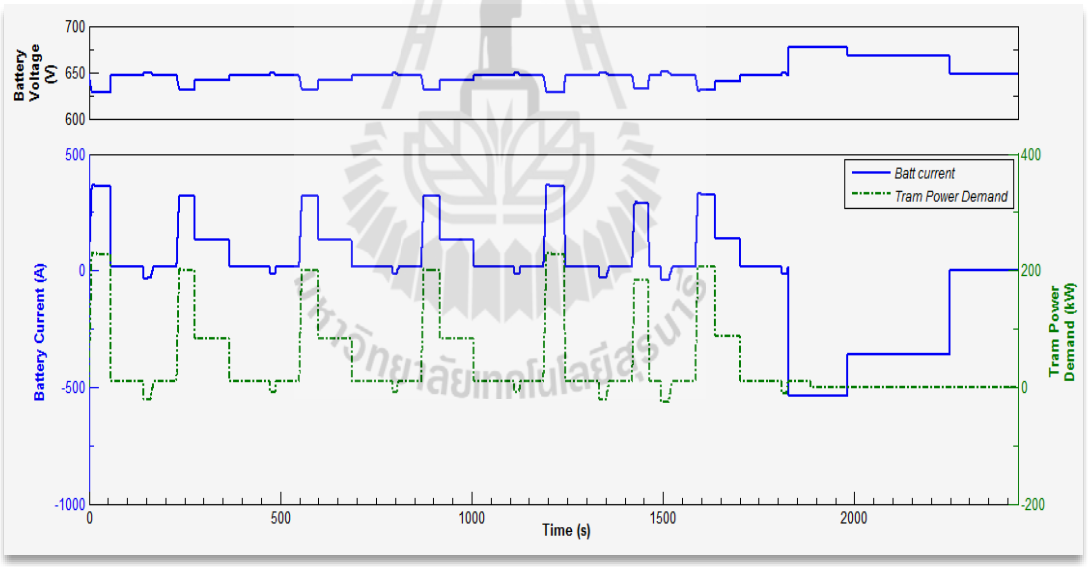


Figure 4.4 TRL route, odd direction, battery current and voltage, and tram power demand profiles.

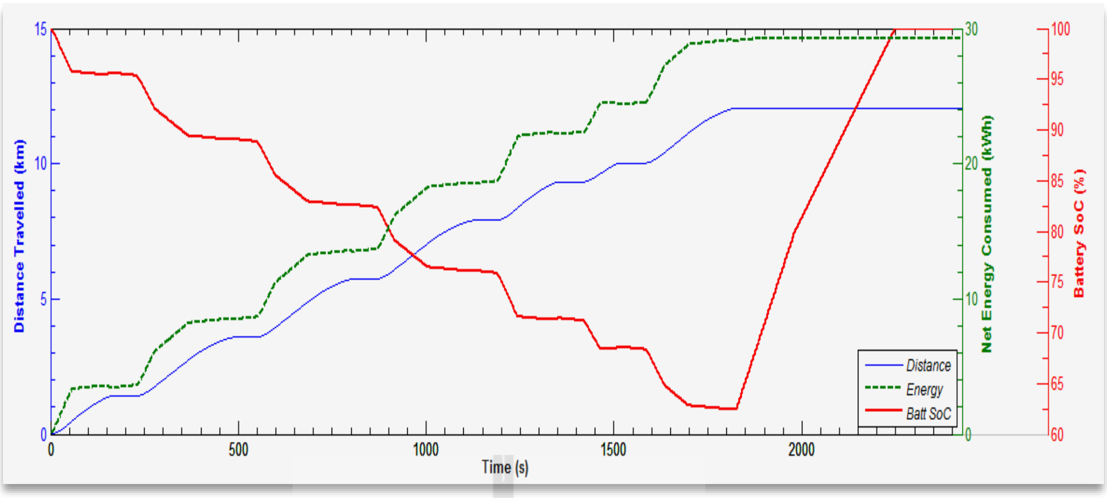


Figure 4.5 TRL route, odd direction, distance travelled, net energy consumed and battery state of charge profiles

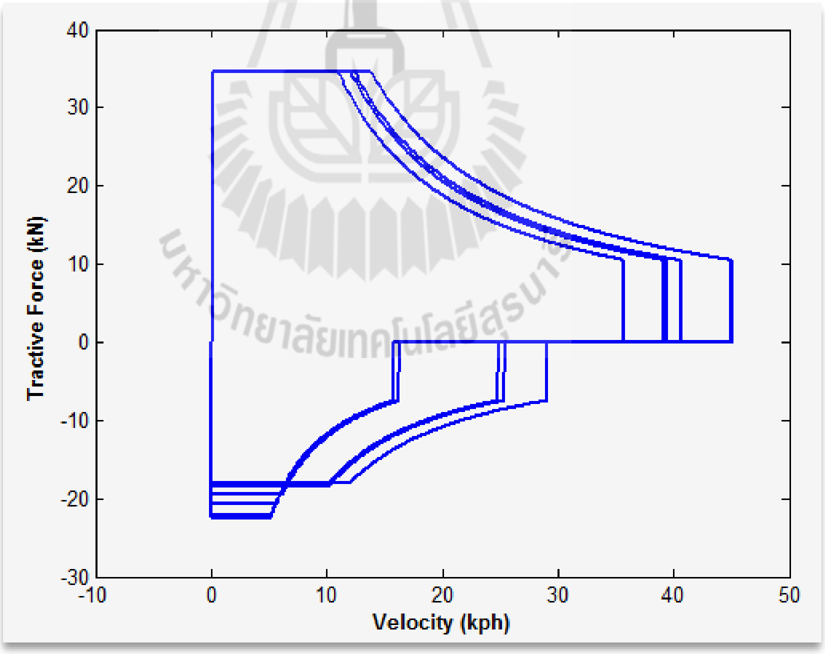


Figure 4.6 TRL route, odd direction, tractive effort curves

Table 4.3 TRL route, odd direction, interstation and cumulative simulation results

Stations	From -To	S1-S2	S2-S3	S3-S4	S4-S5	S5-S6	S6-S7	S7-S8
	Distance(km)	**1.41**	**2.17**	**2.16**	**2.16**	**1.43**	**0.7**	**2.02**
Parameters	Maximum velocity (km/h)	45.05	39.11	39.34	39.19	44.91	35.62	40.68
	Stating acceleration (m/s^2)	0.50	0.54	0.52	0.54	0.54	0.53	0.49
	Braking velocity (km/h)	25.33	15.64	15.74	15.68	24.84	29.06	16.27
Net energy consumed between stations (kWh)		3.44	4.88	4.86	4.86	3.47	2.02	4.57
Net energy consumed between stations plus dwelling (kWh)		3.61	5.04	5.02	5.02	3.64	2.19	4.73
Cumulative net energy consumed (kWh)		3.61	8.65	13.67	18.70	22.33	24.53	**29.26**
Time between stations (min)		2.82	4.34	4.31	4.32	2.86	1.75	4.04
Time between stations plus dwelling time (min)		3.82	5.34	5.32	5.32	3.86	2.75	14.04
Cumulative time (min)		3.82	9.16	14.47	19.79	23.65	26.40	**40.44**

As the track was assumed to be flat, the simulation results for the even direction as presented in Figure 4.7 through Figure 4.10 and in Table 4.4 were similar to the ones in the odd direction, but flipped left to right. The tractive effort curves remain completely the same. It should be noted that, before changing direction the battery pack has to be fully charged at a terminal station. Terminal station dwell time is an important parameter of system performance, service reliability and quality. In this thesis work, the terminal dwell time was set to ten minutes.

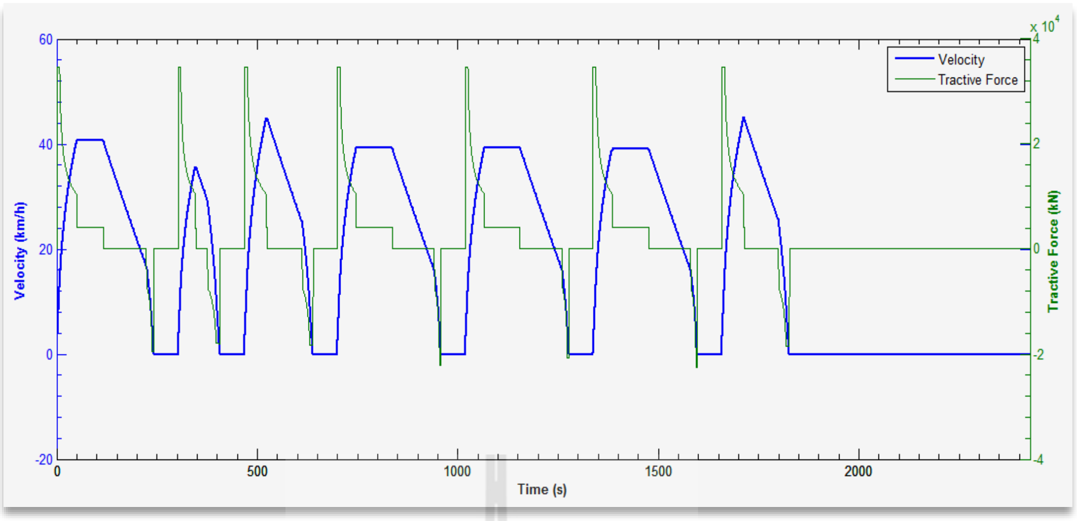


Figure 4.7 TRL route, even direction, velocity and tractive force profiles

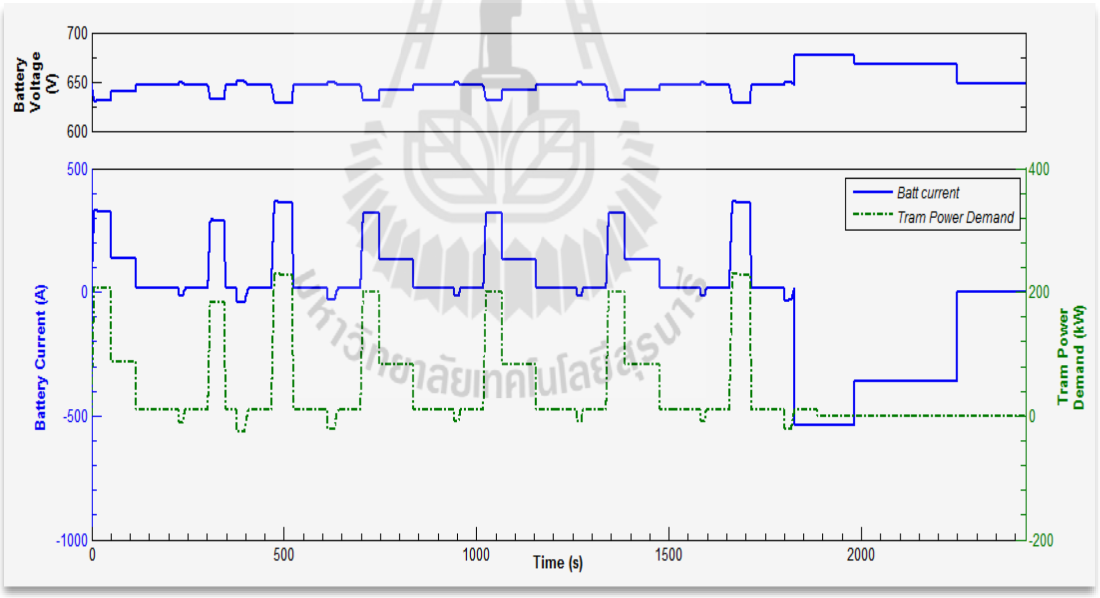


Figure 4.8 TRL route, even direction, battery current and voltage, and tram power demand profiles.

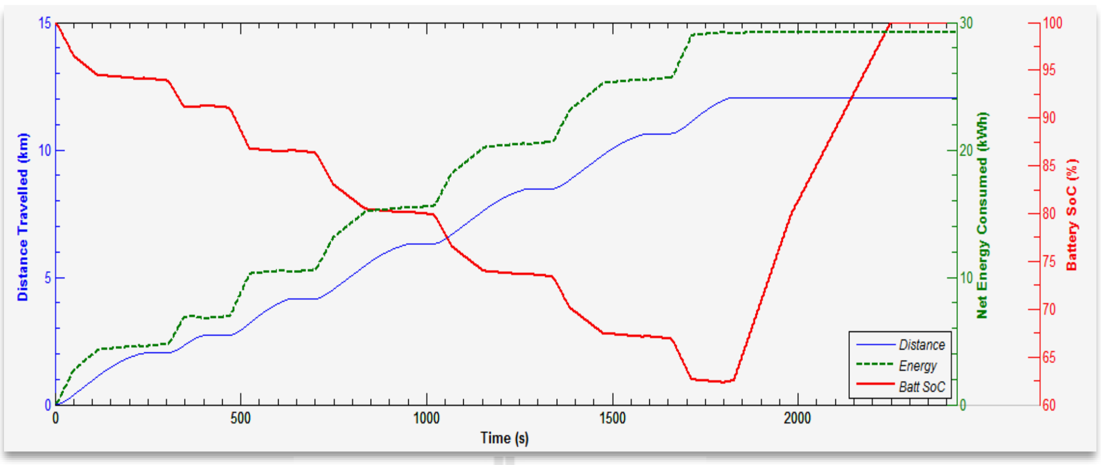


Figure 4.9 TRL route, even direction, distance travelled, net energy consumed and battery state of charge profiles.

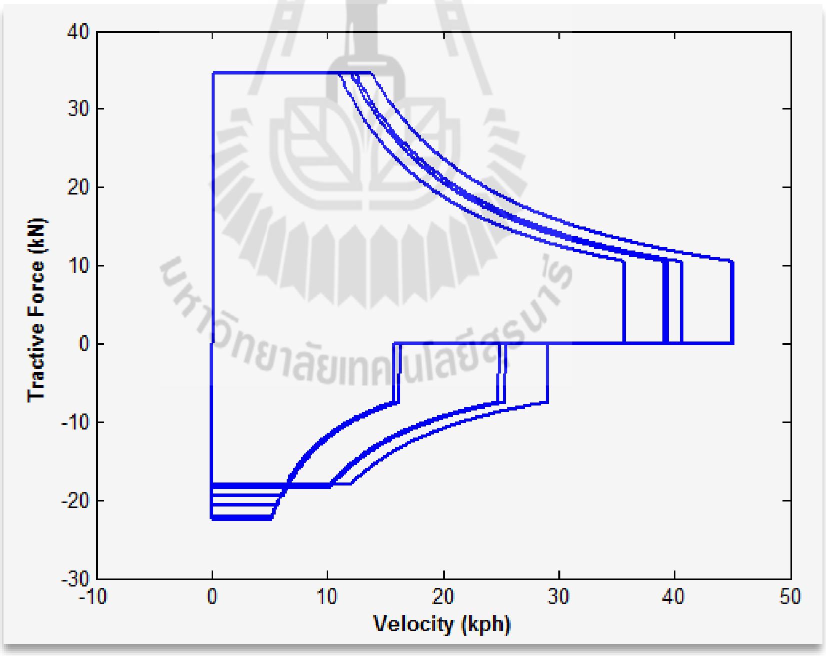


Figure 4.10 TRL route, even direction, tractive effort curves

Table 4.4 TRL route, even direction, interstation and cumulative simulation results

Stations	**From -To**	**S8-S7**	**S7-S6**	**S6-S5**	**S5-S4**	**S4-S3**	**S3-S2**	**S2-S1**
	Distance(km)	**2.02**	**0.7**	**1.43**	**2.16**	**2.16**	**2.17**	**1.41**
Parameters	Maximum velocity (km/h)	40.68	35.62	44.91	39.19	39.34	39.11	45.05
	Stating acceleration (m/s^2)	0.49	0.53	0.54	0.54	0.52	0.54	0.50
	Braking velocity (km/h)	16.27	29.06	24.84	15.68	15.74	15.64	25.33
Net energy consumed between stations (kWh)		4.57	2.02	3.47	4.86	4.86	4.88	3.44
Net energy consumed between stations plus dwelling (kWh)		4.73	2.19	3.64	5.02	5.02	5.04	3.61
Cumulative net energy consumed (kWh)		4.73	6.92	10.56	15.58	20.61	25.65	**29.26**
Time between stations (min)		4.04	1.75	2.86	4.32	4.31	4.34	2.82
Time between stations plus dwelling time (min)		5.04	2.75	3.86	5.32	5.32	5.34	12.82
Cumulative time (min)		5.04	7.79	11.65	16.97	22.28	27.62	**40.44**

4.2.2 BEMU on TAZARA Route

As shown in Figure 4.10, TAZARA route is 21 km long (9 km longer than the TRL route) running from Tazara-Station (S1) to Mwakanga (S10). Charging points were assumed to be installed at the terminal stations: S1 and S10 (where batteries are fully charged) and at three of the intermediate stations: S5, S7 and S8. (where batteries are partially charged). Simulation results for the odd direction (S1 to S10) were as presented in Figure 4.11 through Figure 4.15 and, in Table 4.5.

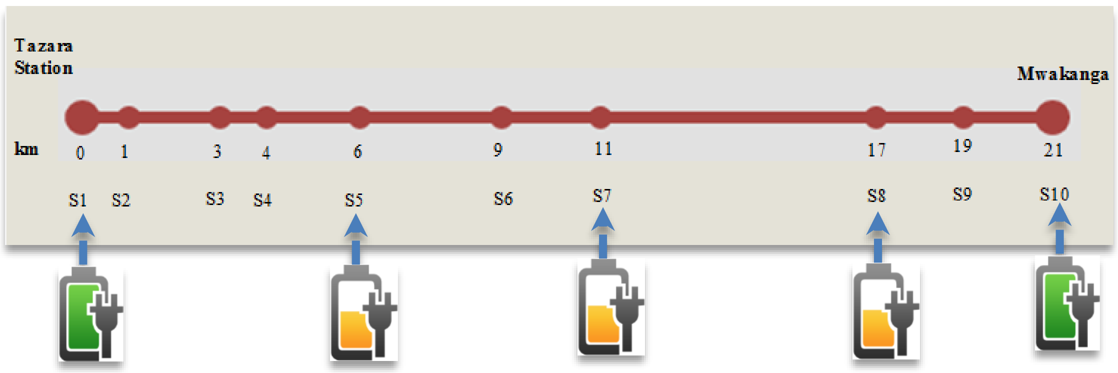


Figure 4.11 TAZARA route and charging points

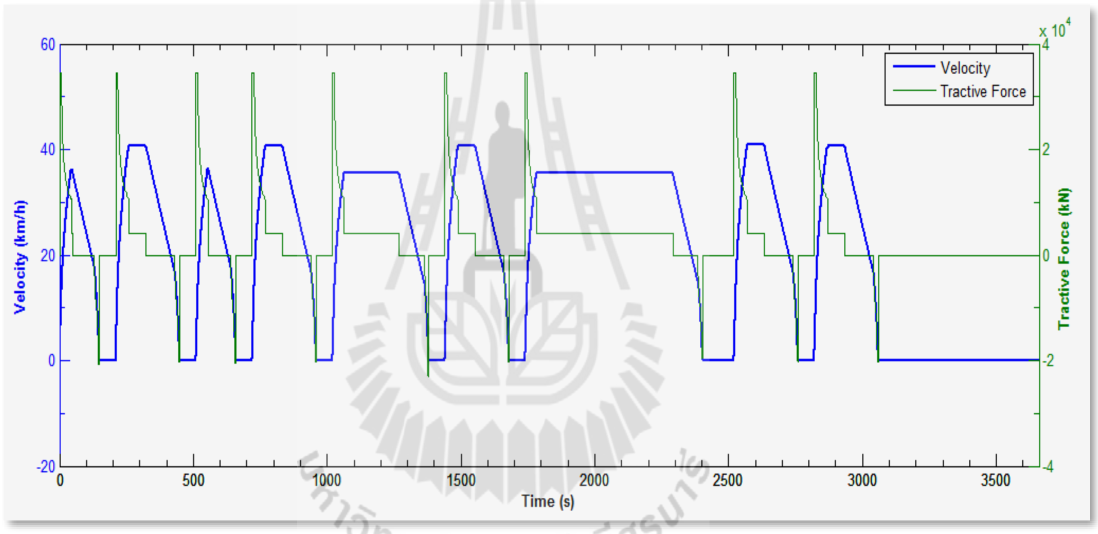


Figure 4.12 TAZARA route, odd direction, velocity and tractive force profiles

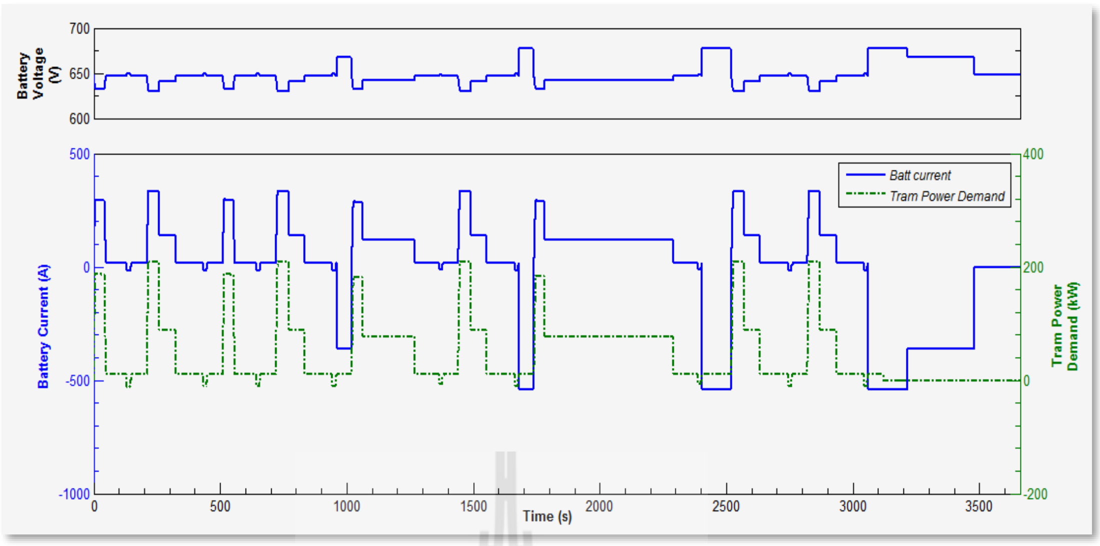


Figure 4.13 TAZARA route, odd direction, battery pack current and voltage, and tram power demand profiles.

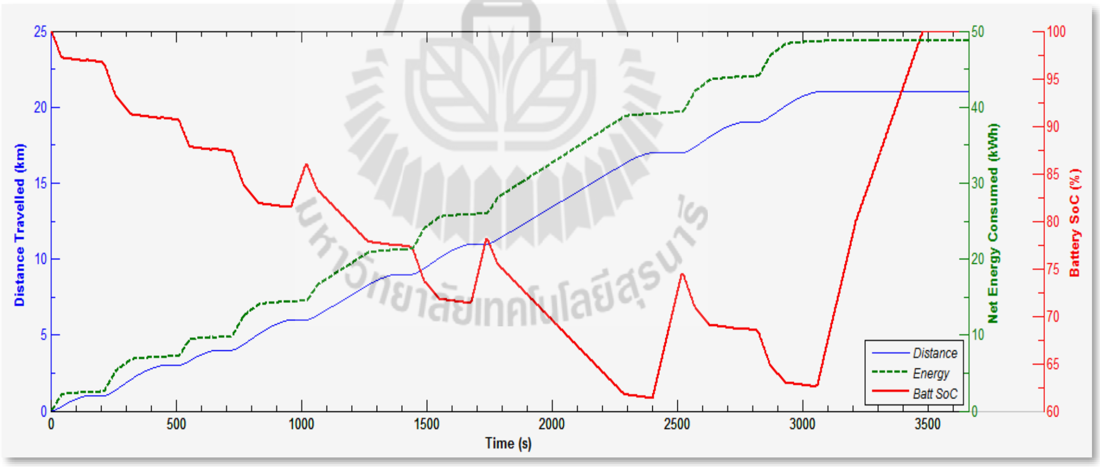


Figure 4.14 TAZARA route, odd direction, distance travelled, net energy consumed and battery state of charge profiles

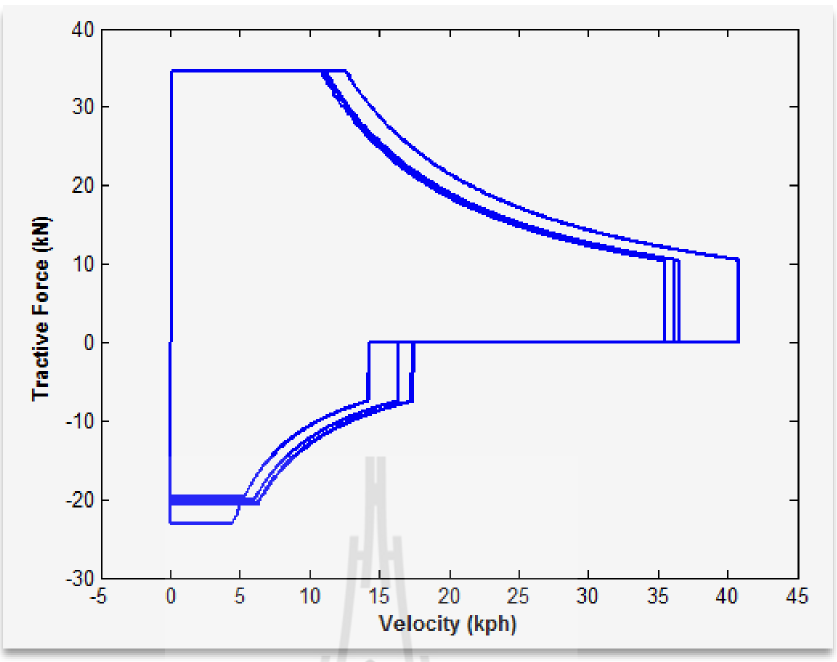


Figure 4.15 TAZARA route, odd direction, tractive effort curves

Table 4.5 TAZARA route, odd direction, interstation and cumulative simulation results

Stations	**From -To**	**S1-S2**	**S2-S3**	**S3-S4**	**S4-S5**	**S5-S6**	**S6-S7**	**S7-S8**	**S8-S9**	**S9-S10**
	Distance(km)	**1**	**2**	**1**	**2**	**3**	**2**	**6**	**2**	**2**
Parameters	Maximum velocity (km/h)	36.14	40.77	36.48	40.78	35.42	40.77	35.53	40.80	40.77
	Stating acceleration (m/s^2)	0.49	0.52	0.54	0.54	0.50	0.50	0.51	0.48	0.51
	Braking velocity (km/h)	17.42	16.31	17.29	16.31	14.17	16.31	14.21	16.32	16.31
Net energy consumed between stations (kWh)		2.40	4.52	2.40	4.52	6.63	4.52	13.05	4.52	4.52
Net energy consumed between stations plus dwelling (kWh)		2.56	4.69	2.56	4.69	6.80	4.69	13.37	4.69	4.69
Cumulative net energy consumed (kWh)		2.56	7.26	9.82	14.51	21.31	26.00	39.38	44.07	**48.76**
Time between stations (min)		2.49	4.00	2.50	4.00	6.00	4.00	11.06	4.00	4.00
Time between stations plus dwelling time (min)		3.49	5.00	3.50	5.01	7.00	5.00	13.00	5.00	14.00
Cumulative time (min)		3.49	8.50	12.00	17.00	24.00	29.00	42.00	47.00	**61.00**

Just like with the TRL route, as the track was assumed to be flat, the simulation results for the even direction as presented in Figure 4.16 through Figure 4.20 and in Table 4.6 were similar to the ones in the odd direction, but flipped left to right. Tractive effort curves also remain completely the same. The only difference that can noted is that, in even direction the interstation charging takes longer at S7 (and not at S8 as it was the case with odd direction), due to time reserved travelling from S8 to S7. Before changing direction, the tram dwells at a terminal station for ten minutes and the battery pack is fully charged.

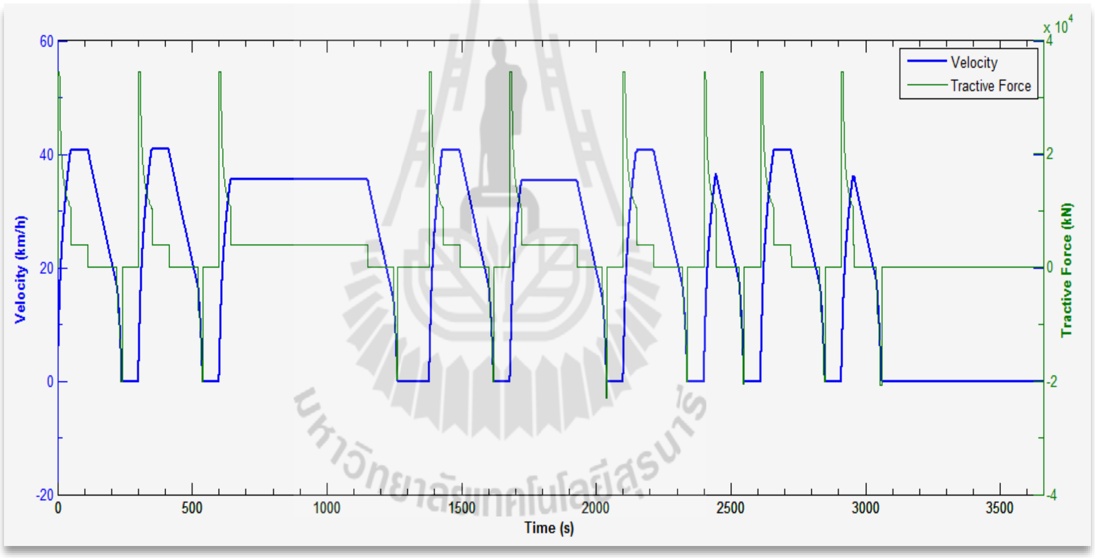


Figure 4.16 TAZARA route, even direction, velocity and tractive force profiles

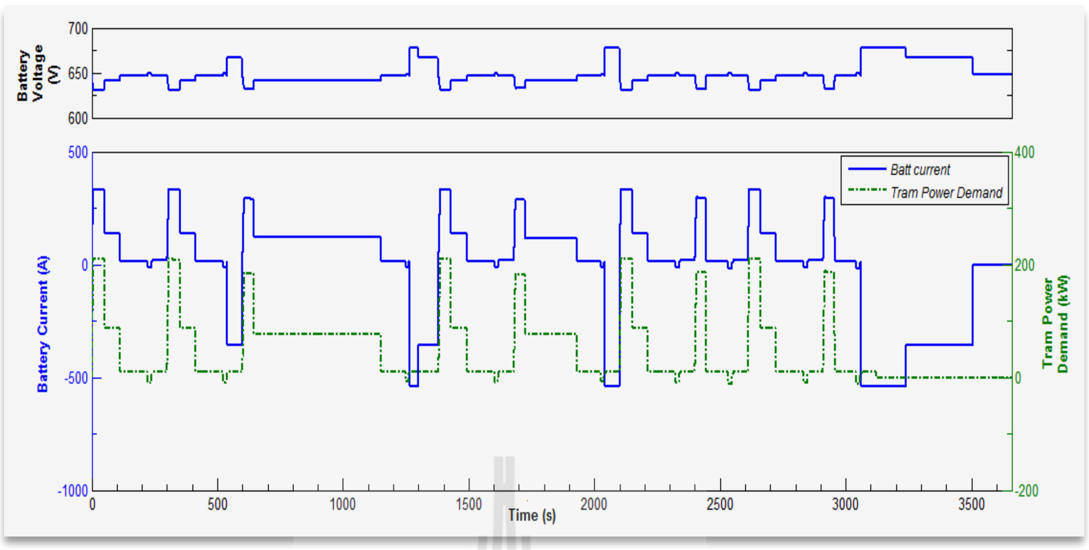


Figure 4.17 TAZARA route, even direction, battery current and voltage, and tram power demand profiles

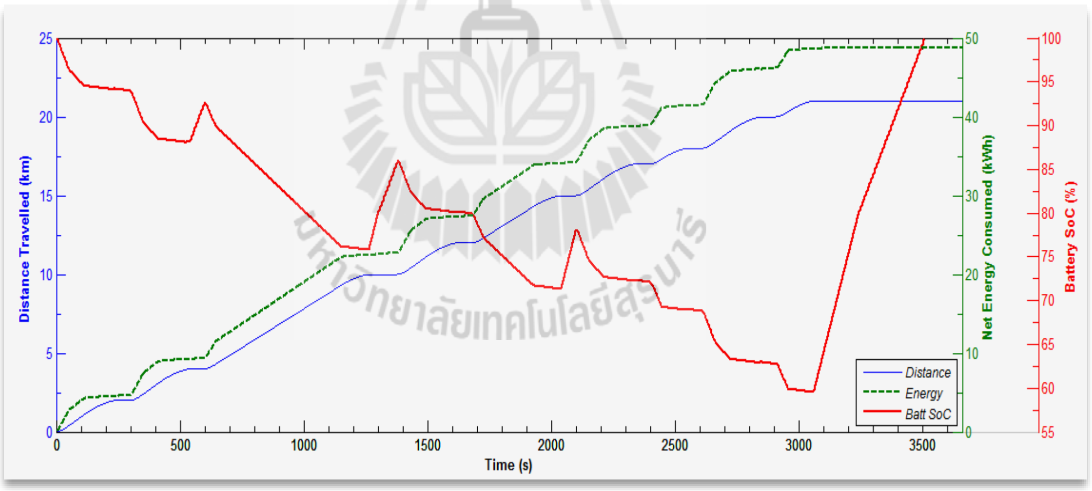


Figure 4.18 TAZARA route, odd direction, distance travelled, net energy consumed and battery state of charge

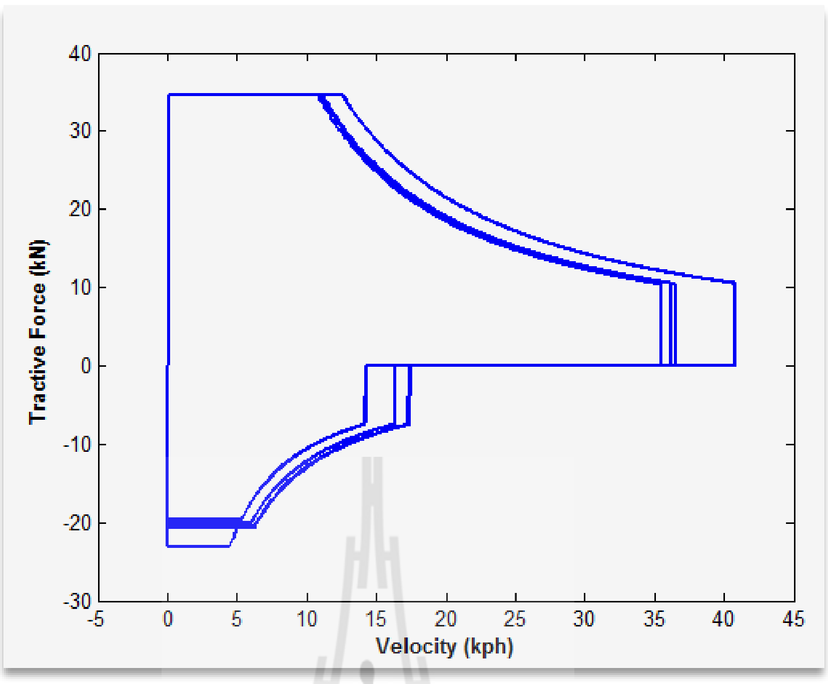


Figure 4.19 TAZARA route, even direction, tractive effort curves

Table 4.6 TAZARA route, even direction, interstation and cumulative simulation results

Stations	**From -To**	**S10-S9**	**S9-S8**	**S8-S7**	**S7-S6**	**S6-S5**	**S5-S4**	**S4-S3**	**S3-S2**	**S2-S1**
	Distance(km)	**2**	**2**	**6**	**2**	**3**	**2**	**1**	**2**	**1**
Parameters	Maximum velocity (km/h)	40.77	40.80	35.53	40.77	35.42	40.78	36.48	40.77	36.14
	Stating acceleration (m/s^2)	0.51	0.48	0.51	0.50	0.50	0.54	0.54	0.52	0.49
	Braking velocity (km/h)	16.31	16.32	14.21	16.31	14.17	16.31	17.29	16.31	17.42
Net energy consumed between stations (kWh)		4.52	4.52	13.05	4.52	6.63	4.52	2.40	4.52	2.40
Net energy consumed between stations plus dwelling (kWh)		4.69	4.69	13.37	4.69	6.80	4.69	2.56	4.69	2.56
Cumulative net energy consumed (kWh)		4.69	9.38	22.76	27.45	34.25	38.94	41.50	46.19	**48.75**
Time between stations (min)		4.00	4.00	11.06	4.00	6.00	4.00	2.50	4.00	2.49
Time between stations plus dwelling time (min)		5.00	5.00	13.00	5.00	7.00	5.00	3.50	5.00	12.50
Cumulative time (min)		5.00	10.00	23.00	28.00	35.00	40.00	43.50	48.50	**60.99**

On TRL route the tram consumes approximately 29.3 kWh net energy on either direction and, on TAZARA route the net energy consumption is approximately 48.8 kWh on either direction. In fuel cost and emission analysis a charger efficiency of 0.9 was included, so that the analysis is based on the energy received from the grid. Energy recaptured from regenerative braking is very low, almost negligible as the tram starts breaking at very low speed.

4.3 Passenger travel fuel cost and emission analysis

To analyze passenger travel fuel cost and emission, electricity and diesel prices and emission factors as given in Table 4.3 were used.

Table 4.7 Electricity and diesel prices and emission factors

Attribute		Value
Diesel	Price (US$/liter)	1
	CO_2 Emission (kg /liter burn)	2.68
Electricity	Price (US$ / kWh)	0.082
	CO_2 Emission (kg/kWh distributed)	0.52

As according to the Tanzania Energy and Water Utilities Regulatory Authority (EWURA), the diesel price in Dar es Salaam in the month of August 2015 was approximately. US$ 1 (EWURA, 2015). On the other hand, the electricity price in Tanzania as given by Tanzania Electric Supply Company Limited (TANESCO) is approximately. US$ 0.082 per kWh, for a customer type T3_MV (TANESCO, 2014). Diesel fuel releases approximately. 2.68 kg of CO_2 per liter burned and, as of May 2014, the CO_2 emission per kWh distributed in Tanzania was approximately. 0.52 kg; taking kg-CO_2/kWh emission factors from natural gas and liquid fuel power plants as 0.549 and 0.817 respectively (EIA, 2015).

Extracted from appendix B, TRL and TAZARA diesel trains' operational data are as summarized in Table 4.4 and Table 4.5 respectively.

Table 4.8. TRL diesel train operational data

Train Schedule and Occupancy					
Morning			**Evening**		
Time	Direction	Percentage Full	Time	Direction	Percentage Full
05:00	Odd	0 %	16:00	Odd	90 %
06:00	Even	90 %	17:00	Even	60 %
07:00	Odd	50 %	18:00	Odd	100 %
08:00	Even	100 %	19:00	Even	50 %
09:00	Odd	60 %	20:00	Odd	80 %
10:00	Even	80 %	21:00	Even	40 %
Train Performance					
Passenger carrying capacity		720 Sitting			1440
		720 Standing (8 per/m^2)			
Train average full					960 Passengers (66.67 %)
Average fuel consumption (Litres of diesel)					1125 liters per day
		7.85 liters /km			(93.75 liters/one way trip)
Average journey time					45 minutes
Average station stopping time					3 minutes

Table 4.9. TAZARA diesel train operational data

Train Schedule and Occupancy					
Morning			Evening		
Time	Direction	Percentage Full	Time	Direction	Percentage Full
04:45	Odd	0 %	16:00	Odd	30 %
05:25	Even	100 %	17:00	Even	20 %
06:40	Odd	20 %	18:05	Odd	90 %
07:40	Even	100 %	19:10	Even	30 %
08:45	Odd	20 %	20:10	Odd	90 %
09:45	Even	30 %	21:10	Even	10 %

Train Performance		
Passenger carrying capacity	640 Sitting	1200
	560 Standing (8 per/m^2)	
Train average full		540 Passengers (45 %)
Average fuel consumption		840 liters per day
(Liters of diesel)		3.3333 liters /km (70 liters/one way trip)
Average journey time		≥ 50 minutes
Average station stopping time		≥ 2 minutes

Thus, fuel cost and emission per passenger kilometer were as given in Figure 4.13 and Figure 4.14 respectively. Compared to TRL train, the BEMU reduces fuel cost and CO_2 emission per passenger kilometer by 87.7%, and 70.89 % respectively. And compared to TAZARA train the BEMU reduces fuel cost and CO_2 emission per passenger kilometer by 75.6 %, and 42.25 % respectively.

From Table 4.4 and Table 4.5 it ca be seen that, TAZARA train has lower diesel consumption per passenger kilometer than TRL train; i.e. TAZARA train has higher fuel efficiency than TRL train. This is the reason why compared to TRL train, the BEMU reduces more fuel cost and CO_2 emission per passenger kilometer than when compared to TAZARA train. From Figure 4.13 and Figure 4.14, it can be seen that the BEMU fuel cost and emission per passenger kilometer are lower on TRL route than they are on TAZARA route. One of the reasons is that the TRL route has higher ridership than TAZARA route.

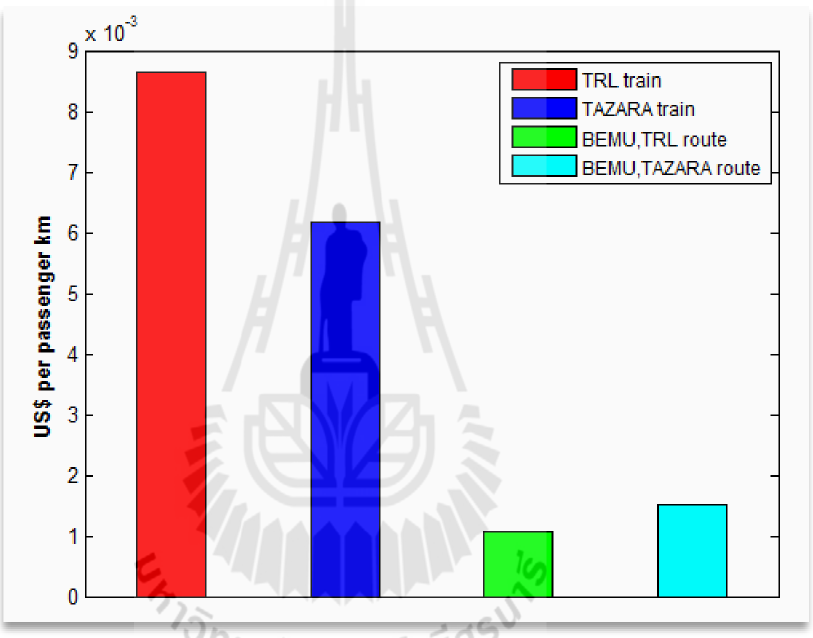


Figure 4.20 Fuel cost per passenger kilometer

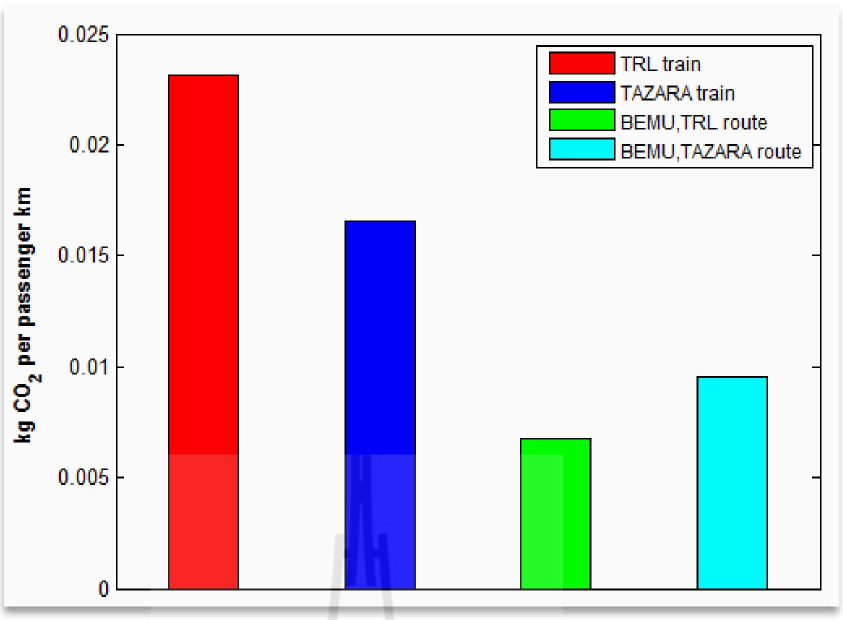


Figure 4.21 Carbon dioxide emission per passenger kilometer

CHAPTER V

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

This research has been driven by the facts that electric trains are more efficient, fuel-cost effective and environmental friendly than the traditional diesel trains and, battery technology has advanced and proven success in vehicular applications. The research therefore, is built upon three main pillars: optimal speed profile, BEMU and, passenger travel fuel cost and emission.

To minimize amount of energy required to move a tram from one station to the next, and eventually one terminal to the other, speed profile optimization was done. Three optimization methods were used, from which an optimal speed profile was established. As the tram was to be powered by batteries which have limited energy capacity, optimization was vital.

Modeled as a BEMU, tram movement was simulated, and it was found that a 648 V, 120 Ah LTO battery pack is quite sufficient to power the tram with 312 passenger capacity, and approximately. 63 ton effective weight, along a 12 km route and 21 km route. On the 21 km route, partial charging at some of the intermediate stations as passengers get on and off the tram was found to be convenient, so as to keep the battery pack at high state of charge, above 50%.

Compared to TRL diesel train, the BEMU was found to reduce fuel cost and emission per passenger kilometer by 87.7 percent, and 70.89 percent respectively. And compared to TAZARA diesel train, the BEMU was found to reduce fuel cost and emission per passenger kilometer by 75.6 percent, and 42.25 percent respectively. These reductions are significant; policy makers, city authorities and other stakeholders are thereby called to consider the results from this research when making policies and plans for city transport systems.

5.2 Recommendations

Further improvement on this research would be to:

(i) Include within a case study, a diesel train with a very good efficiency and/or, model and simulate a diesel train assuming very good efficiency;

(ii) Include a design of suitable charging infrastructures for the intermediate and terminal stations;

(iii) Analyze which is a better option between: (a) charging at intermediate stations so as to minimize battery pack size and, (b) increasing the battery pack size so that charging takes place only at terminal stations;

(iv) From (ii) above, if option ‘a’ is better than ‘b’, then evaluate whether it is convenient or not to set different dwell times, for different intermediate stations according to the average number of commuters at a station, so that, intermediate charging stations are the ones where a tram dwells for long;

(v) Evaluate whether it is convenient or not to have a short catenary system installed at every intermediate station to allow the tram accelerate while drawing power from the catenary. A short catenary at every intermediate station will make every intermediate station a charging station and significantly reduce the battery pack size. Batteries will be mainly used during cruising; and

(vi) Include investment cost and other operational costs such as maintenance cost, and estimate a break-even point.

REFERENCES

ABB Communications. (2013, May 30). **Large-capacity, flash-charging, battery-powered pilot bus takes to the street**. Retrieved September 17, 2015, from http://www.abb.com/cawp/seitp202/9315e568e4c6a1f8c1257b7400302fcd.aspx

Abu-Rub, H., Malinowski, M., and Al-Haddad, K. (2014). **Power Electronics for Renewable Energy Systems, Transportation and Industrial Applications** . West Sussex, United Kingdom: John Wiley & Sons Ltd.

Adewumi, A. O., and Arasomwan, A. M. (2014). Improved Particle Swarm Optimization based on Greedy and Adaptive Features. **2014 IEEE Symposium on Swarm Intelligence (SIS)** (pp. 1 - 6). Orlando, FL: IEEE.

Ahadzadeh, B., and Menhaj, M. B. (2014). A Modified Differential Evolution Algorithm Based on a New Mutation Strategy and Chaos Local Search for Optimization Problems. **Proceedings of the 2014 4th IEEE International eConference on Computer and Knowledge Engineering (ICCKE)** (468 - 473). Mashhad.

Altairnanno. (n.d.). **24 V 60 Ah Battery Module**. Retrieved December 30, 2015, from http://www.altairnano.com/products/battery-module/

Barrero, R., Mierlo, J. V., and Tackoen, X. (2008). Energy savings in Public Transport. **IEEE Vehicular Technology Magazine,Vol.3(3): 26-36.**

Bombardier Transportation. (2015). **Trams and e-buses PRIMOVE battery optimised for maximum performance and lifetime**. Berlin, Germany.

Bombardier Transportation. (2015, November 3). **Bombardier's Battery Powered Tram Sets Range Record**. (Bombardier) Retrieved February 11, 2016, from http://www.bombardier.com/en/media/newsList/details.BT-20151103-Bombardiers-Battery-Powered-Tram-Sets-Range-Record-01.bombardiercom.html

Chan, F. T., and Tiwari, M. K. (2007). **Swarm Intelligence, Focus on Ant and Particle Swarm Optimization**. Vienna: INTECH.

Chengfu, S., Haiyan, Z., and Liqing, C. (2012). Improved differential evolution algorithms. **Proceedings of the 2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE)** (142 - 145). Zhangjiajie.

Das, S., and Suganthan, P. N. (2011). Differential Evolution: A Survey of the State-of-the-Art. **IEEE Transactions on Evolutionary Computation.** Vol.15(1): 4-31.

Dincer, I., Colpan, C. O., and Ezan, M. A. (2015). **Progress in Clean Energy, Volume 2: Novel Systems and Applications**. Springer.

Egbue, O., and Long, S. (2012). Barriers towide spread adoption of electric vehicles: Ananalysis of consumer attitudes andperceptions. **Energy Policy.** Vol.48: 717-729.

EIA. (2015, March 30). **Frequently Asked questions**. (Energy Information Administration (EIA)) Retrieved January 31, 2016, from https://www.eia.gov/tools/faqs/faq.cfm?id=74&t=11

EWURA. (2015). **Electricity**. Retrieved February 7, 2016, from EWURA Web site: http://144.76.33.232/?page_id=130

EWURA. (2015, August 5). **Cap Prices for Petroleum Products with Effect from 5th August 2015**. Dar es Salaam, Tanzania.

Gillespie, T. D. (1992). **Fundametals of vehicle dynamics**. Warrendale, PA , Pennsylvania: Society of Automotive Engineers.

Goodman, C. (2008). Overview of electric railway systems and the calculation of train performance. **Electric Traction Systems**. Manchester.

Griffiths, P. (2012, September 5). **Technology Briefing Paper Catenary Free Tram Operation**. United Kingdom.

Hirose, H., Yoshida, K., & Shibanuma, K. (2012). Development of Catenary and Storage Battery Hybrid Train System. **Proceedings of the IEEE-Electrical Systems for Aircraft, Railway and Ship Propulsion (ESARS)**. 2012.

Hodges, T. (2010, January). **Public Transportation's Role in Responding to Climate Change**. United States of America.

Holland, J. H. (1992). Genetics Algorithms: Computer programs that "evolve" in ways that resemble natural selection can solve complex problems even their creators do not fully understand. **Scientific American**, 66-72.

Horiba, T. (2014). Lithium-Ion Battery Systems. **Proceedings of the IEEE** 102(6), 939 - 950.

Husain, I. (2003). **Electric and Hybrid Vehicles Design Fundamentals**. Washington, D.C., United States of America: CRC Press LLC.

Jeyakumar, G., and Shunmuga, V. C. (2009). A Comparative Performance Analysis of Differential Evolution and Dynamic Differential Evolution Variants.

Proceedings of the IEEE-World Congress on Nature & Biologically Inspired Computing (463 - 468). Coimbatore.

Jiang, Y., Liu, J., Tian, W., Shahidehpour, M., and Krishnamurthy, M. (2014). Energy Harvesting for the Eletrification of Railway Stations. **IEEE Electrification Magazine**, Vol.2(3): 39-48.

Kulworawanichpong, T. (2003). **Optimising AC Electric Railway Power Flows With Power Electronic Control:** PhD Thesis. The University of Birmingham.

Kulworawanichpong, T., and Punpaisarn, S. (2014, May). Dynamic Simulation of Electric Bus Vehicle. **The Standard International Journals (The SIJ).** Vol.2(3): 99-104.

Larminie, J., & Lowry, J. (2012). **Electric Vehicle Technology Explained** (2nd ed.). West Sussex,, United Kingdom: John Wiley & Sons Ltd.

Lazinica, A. (2009). **Particle Swarm Optimization**. Vienna, Austria: In-Tech.

Li, S., Niu, J., Zhao, Y. C., So, K. P., Wang, C., Wang, C. A., et al. (2015). High-rate aluminium yolk-shell nanoparticle anode for Li-ion battery with long cycle life and ultrahigh capacity. **Nature communications**, 1-7.

Li, X., and Lo, H. K. (2014). An energy-efficient scheduling and speed control approach for metro rail operations. **Transportation Research Part B**. Vol.64: 73–89.

Liang, S., Song, S., Kong, L., and Cheng, J. (2010). An Improved Particle Swarm Optimization Algorithm and Its Convergence Analysis. **2010 Second International Conference on Computer Modeling and Simulation** (pp. 138 - 141). Sanya, Hainan: IEEE.

Liu, S., Jiang, J., Shi, W., Zeyu, M., Wang, L. Y., & Guo, H. (2015). Butler–Volmer-Equation-Based Electrical Model for High-Power Lithium Titanate Batteries Used in Electric Vehicles. **IEEE Transactions on Industrial Electronics**, *62*(12), 7557-7568.

Liu, S., Jiang, J., Shi, W., Zeyu, Ma., Wang, L. Y., and Guo, H. (2015). Butler–Volmer-Equation-Based Electrical Model for High-Power Lithium Titanate Batteries Used in Electric Vehicles. **IEEE Transactions on Industrial Electronics.** Vol.62(12): 7557-7568.

Lu, S., Hillmansen, S., Ho, T. k., and Robert, C. (2013, June). Single Train Trajectory Optimization. **IEEE Transactions On Intelligent Transportation Systems.** Vol.12(2), 743-750.

Mashadi, B., & Crolla, D. (2012). **Vehicle Powertrain Systems**. United Kingdom: Wiley.

Mei, C., Liu, G., & Xiao, X. (2010). Improved particle swarm optimization algorithm and its global convergence analysis. **2010 Chinese Control and Decision Conference (CCDC)** (pp. 1662 - 1667). Xuzhou: IEEE.

MEM. (2014, June 30). **Tanzania Electricity Supply Industry Reform Strategy and Roadmap 2014 - 2015**. Dar es Salaam Tanzania.

Mi, C., Masrur, M. A., and Gao, D. W. (2011). **Hybrid Electric Vehicles: Principles and Applications with Practical Perspectives**. West Sussex: John Wiley & Sons, Ltd.

Mwambeleko, J. J., Kulworawanichpong, T., and Greyson, K. A. (2015). Tram and trolleybus net traction energy consumption comparison. **Proceedings of the**

IEEE 18th International Conference on Electrical Machines and Systems (ICEMS2015). Pattaya.

Nachilongo, H. N. (2013, November 11). **A train ride worth Sh2m loss**. (Mwananchi Communications Ltd) Retrieved June 5, 2015, from http://www.thecitizen.co.tz/News/A-train-ride-worth-Sh2m-loss/-/1840392/2068448/-/acnhks/-/index.html

Profillidis, V. (2006). **Railway Management and Engineering** - Third Edition. Wey Court East: Ashgate Publishing Limited.

Rahn, C. D., and Wang, C.-Y. (2013). **Battery Systems Engineering**. West Sussex: John Wiley & Sons, Ltd.

Rajput, R. (2006). **Utilisation of Electrical Power: Including Electrical Drives and Electric Traction**. New Delhi: Laxmi Publications (P) ltd.

Rao, S. S. (2009). **Engineering Optimization Theory and Practice** (4th ed.). New Jersey, United States of America: John Wiley & Sons, Inc.

Sadri, J., and Suen, C. Y. (2006). A Genetic Binary Particle Swarm Optimization Model. **Proceedings of the IEEE Congress on Evolutionary Computation**, (656-663). Vancouver.

Sanghvi, R. C., Vashi, A. S., Patolia, H. P., and Jivani, R. G. (2014). Multi-Objective Optimization of Two-Stage Helical Gear Train Using NSGA-II. **Journal of Optimization**, 1-8.

Sen, P. C. (2014). **Principles of Electric Machines and Power Electronics**. Ontario: John Wiley & Sons Inc.

Shiraki, N., Tokito, K., & Yokozutsumi, R. (2015). Propulsion system for catenary and storage battery hybrid electric railcar series EV-E301. **Proceedings of the IEEE-International Conference on Electrical Systems for Aircraft, Railway, Ship Propulsion and Road Vehicles (ESARS).** Aachen.

Škoda Transportation. (2013, May 6). **Škoda transportation will deliver catenary-free trams to turkey**. Retrieved January 4, 2016, from Škoda Transportation Web site: http://www.skoda.cz/en/press-room/news/skoda-transportation-will-deliver-catenary-free-trams-to-turkey/

Škoda Transportation. (n.d.). **The Electric Skoda Perun HP**. (Škoda Transportation) Retrieved September 17, 2015, from http://www.skoda.cz/cs/produkty/elektricke-a-hybridni-autobusy/elektrobus-skoda-hp-perun/

Stadler Pankow GmbH. (2011, May 25). **Battery-powered tram for Munich produced by Stadler Pankow GmbH sets a new world record**. Retrieved January 5, 2016, from http://www.stadlerrail.com/en/news/2011/05/25/tram-without-overhead-line-a-new-world-record/

Stadler Rail Group. (2014, July 5). **Low-floor light rail vehicle, type Variobahn for BOGESTRA**. Bussnang, Switzerland.

Steimel, A. (2008). **Electric Traction - Motive Power and Energy Supply**. Munich: Oldenbourg Industrieverlag GmbH.

Stroe, A.-I., Swierczynski, M., Stroe, D.-I., and Teodorescu, R. (2015). Performance Model for High-Power Lithium Titanate Oxide Batteries based on Extended Characterization Tests. **Proceedings of the IEEE- Energy Conversion Congress and Exposition (ECCE)**. Montreal, QC.

TANESCO. (2014, June 1). **Tariffs.** Retrieved June 25, 2015, from http://www.tanesco.co.tz/index.php?option=com_docman&task=cat_view&gid=36&Itemid=221

TAZARA. (n.d.). **Passenger Services**. Retrieved December 22, 2015, from http://tazarasite.com/?page_id=106

TOSA. (2013). **Flash Mobility. Clean City. Smart Bus.** (TOSA) Retrieved September 17, 2015, from http://www.tosa2013.com/en#/

Vas, P. (1999). **Artificial intelligence based electrical machines and drives: Application of fuzzy, neural, fuzzy-neural, and genetic algorithms based techniques**. New York, United States: Oxford University Press.

Volvo. (n.d.). **The all-new Volvo 7900 Electric – go where people want to go**. (Volvo) Retrieved January 5, 2016, from http://www.volvobuses.com/bus/global/en-gb/products_services/buses/City%20buses/volvo_7900_electric/Pages/introduction.aspx

Volvo. (n.d.). *Volvo 7900* **Electric Hybrid - silent, clean and available**. (Volvo) Retrieved January 5, 2016, from http://www.volvobuses.com/bus/global/en-gb/products_services/buses/City%20buses/volvo_7900_electric_hybrid/Pages/introduction.aspx

Wikipedia. (2015, April 1). **Rail transport in Tanzania**. Retrieved December 5, 2015, from Wikipedia Website: https://en.wikipedia.org/wiki/Rail_transport_in_Tanzania

Wikipedia. (2015, Novermber 21). **Battery electric multiple unit**. (Wikipedia) Retrieved January 6, 2016, from Wikipedia Web Site: https://en.wikipedia.org/wiki/Battery_electric_multiple_unit

Yan, X., Wu, Q., Liu, H., and Huang, W. (2013). An Improved Particle Swarm Optimization Algorithm and Its Application. **IJCSI International Journal of Computer Science Issues**, 316-324.

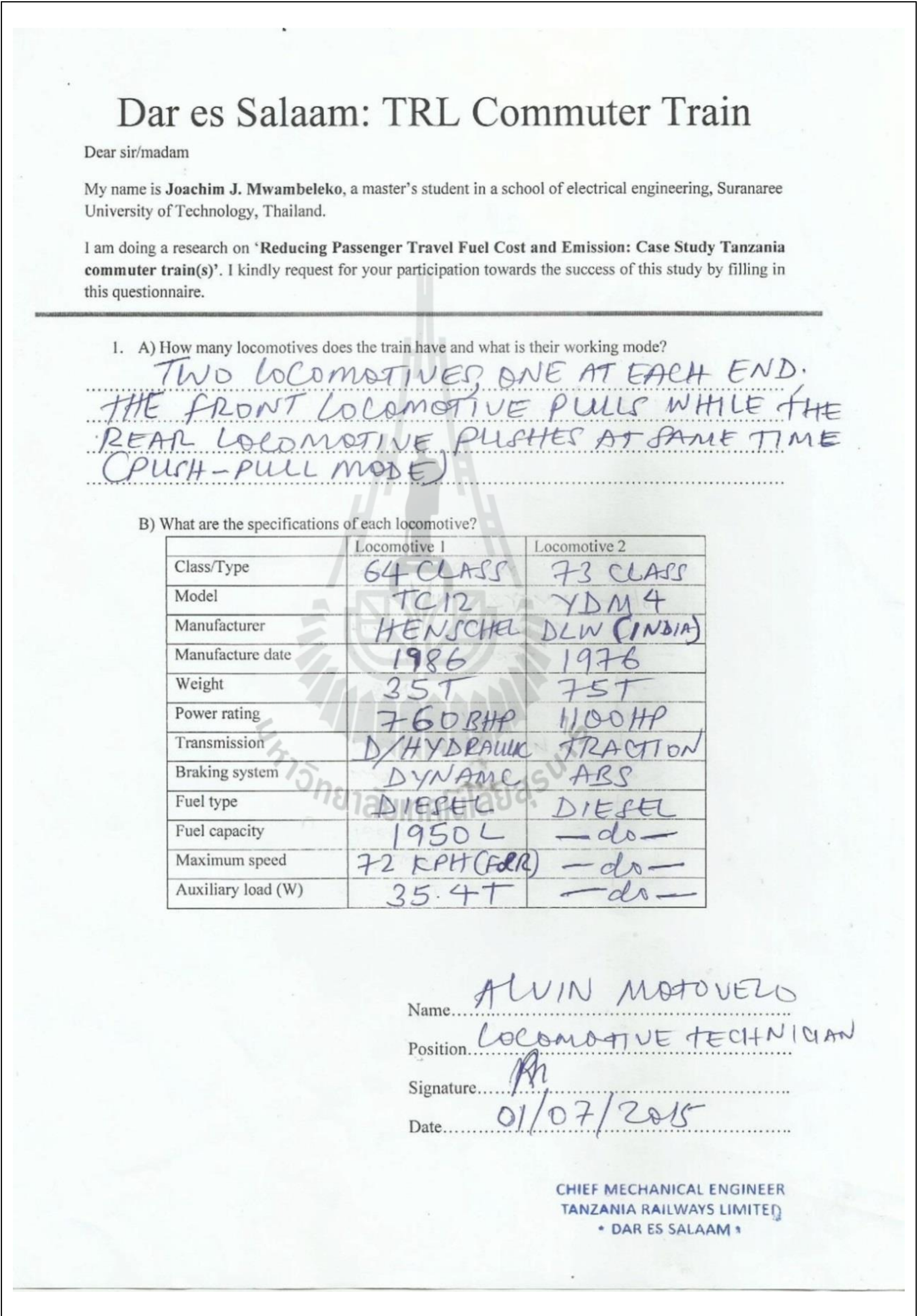
Yao, L. W., Aziz, J., Kong, P. Y., Idris, N., & Alsofyani, I. (2014). Modeling of Lithium Titanate Battery for Charger Design. **Proceedings of the IEEE Australasian Universities Power Engineering Conference (AUPEC)** (1-5). Perth: IEEE.

Zhang, J.-w., & Wei, X. (2009). An Improved Particle Swarm Optimization Algorithm and its Application for Solving Traveling Salesman Problem. **2009 World Congress on Computer Science and Information Engineering** (pp. 612 - 616). Los Angeles, CA: IEEE.

APPENDIX A

TANZANIA COMMUTER TRAINS TECHNICAL AND PERFORMANCE DATA

A.1 TRL Commuter Train Performance Data



Dar es Salaam: TRL Commuter Train

Dear sir/madam

My name is **Joachim J. Mwambeleko**, a master's student in a school of electrical engineering, Suranaree University of Technology, Thailand.

I am doing a research on '**Reducing Passenger Travel Fuel Cost and Emission: Case Study Tanzania commuter train(s)**'. I kindly request for your participation towards the success of this study by filling in this questionnaire.

1. A) How many locomotives does the train have and what is their working mode?

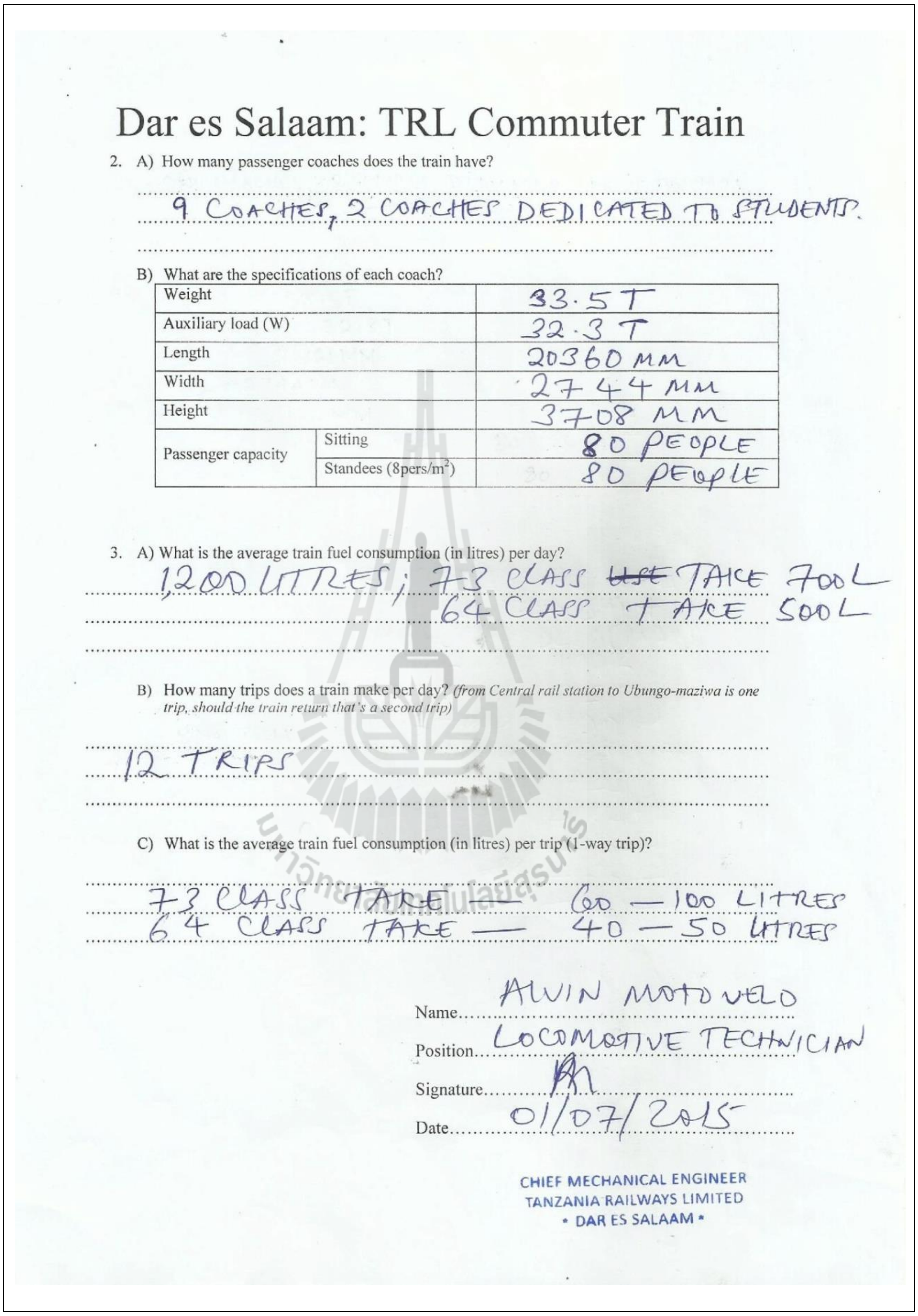
TWO LOCOMOTIVES, ONE AT EACH END. THE FRONT LOCOMOTIVE PULLS WHILE THE REAR LOCOMOTIVE PUSHES AT SAME TIME (PUSH-PULL MODE)

B) What are the specifications of each locomotive?

	Locomotive 1	Locomotive 2
Class/Type	64 CLASS	73 CLASS
Model	TC12	YDM 4
Manufacturer	HENSCHEL	DLW (INDIA)
Manufacture date	1986	1976
Weight	35T	75T
Power rating	760BHP	1100HP
Transmission	D/HYDRAULIC	TRACTION
Braking system	DYNAMIC	ABS
Fuel type	DIESEL	DIESEL
Fuel capacity	1950L	—do—
Maximum speed	72 KPH (FOR)	—do—
Auxiliary load (W)	35.4T	—do—

Name: ALVIN MOTOVELO
Position: LOCOMOTIVE TECHNICIAN
Signature: [illegible]
Date: 01/07/2015

CHIEF MECHANICAL ENGINEER
TANZANIA RAILWAYS LIMITED
• DAR ES SALAAM •



Dar es Salaam: TRL Commuter Train

2. A) How many passenger coaches does the train have?

9 COACHES, 2 COACHES DEDICATED TO STUDENTS.

B) What are the specifications of each coach?

Weight		33.5 T
Auxiliary load (W)		32.3 T
Length		20360 MM
Width		2744 MM
Height		3708 MM
Passenger capacity	Sitting	80 PEOPLE
	Standees (8pers/m^2)	80 PEOPLE

3. A) What is the average train fuel consumption (in litres) per day?

1,200 LITRES; 73 CLASS ~~USE~~ TAKE 700L
64 CLASS TAKE 500L

B) How many trips does a train make per day? *(from Central rail station to Ubungo-maziwa is one trip, should the train return that's a second trip)*

12 TRIPS

C) What is the average train fuel consumption (in litres) per trip (1-way trip)?

73 CLASS TAKE — 60 — 100 LITRES
64 CLASS TAKE — 40 — 50 LITRES

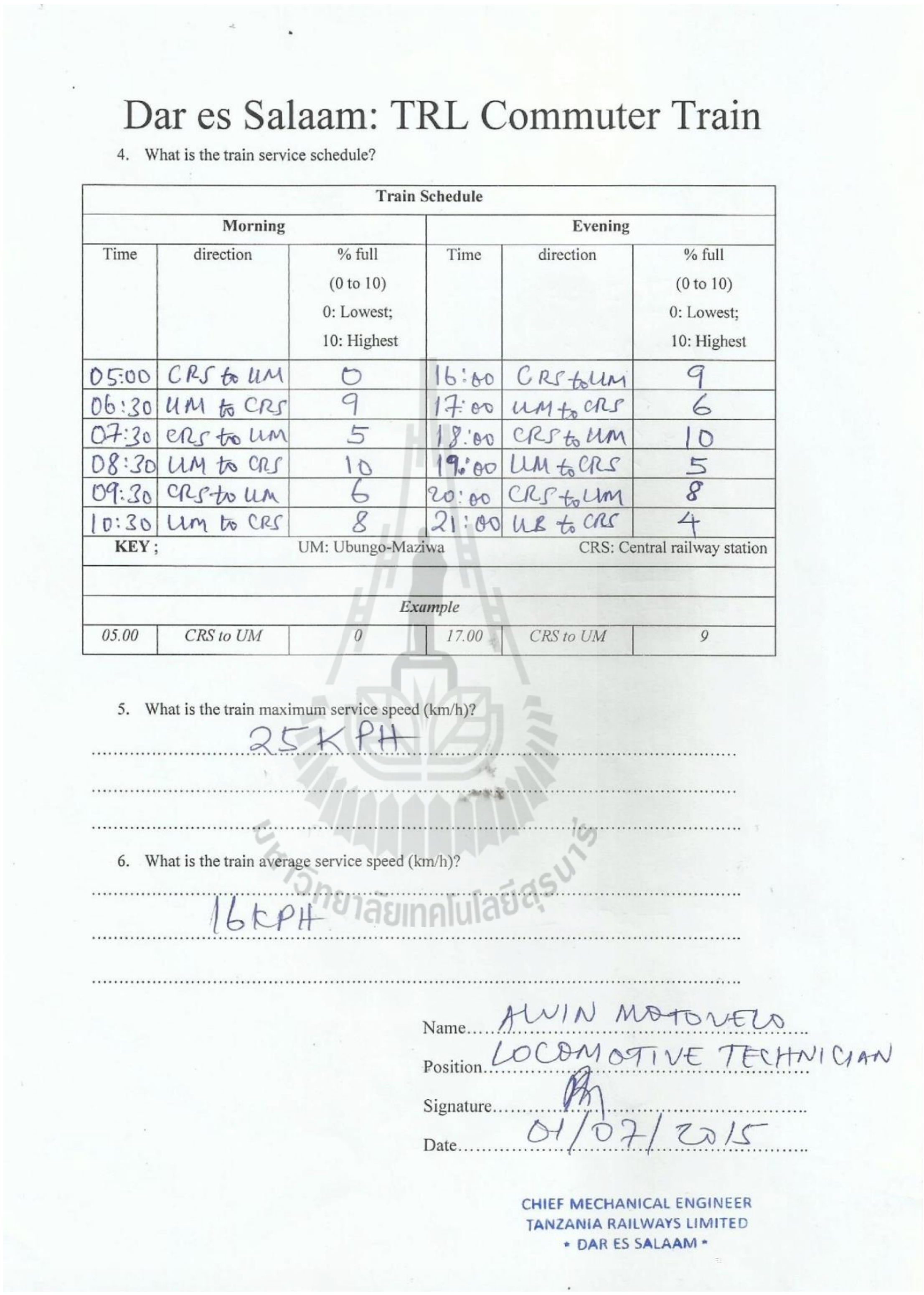
Name: ALVIN MOTOVELO

Position: LOCOMOTIVE TECHNICIAN

Signature: [illegible]

Date: 01/07/2015

CHIEF MECHANICAL ENGINEER
TANZANIA RAILWAYS LIMITED
• DAR ES SALAAM •



Dar es Salaam: TRL Commuter Train

4. What is the train service schedule?

Train Schedule					
Morning			**Evening**		
Time	direction	% full (0 to 10) 0: Lowest; 10: Highest	Time	direction	% full (0 to 10) 0: Lowest; 10: Highest
05:00	CRS to UM	0	16:00	CRS to UM	9
06:30	UM to CRS	9	17:00	UM to CRS	6
07:30	CRS to UM	5	18:00	CRS to UM	10
08:30	UM to CRS	10	19:00	UM to CRS	5
09:30	CRS to UM	6	20:00	CRS to UM	8
10:30	UM to CRS	8	21:00	UB to CRS	4
KEY ;		UM: Ubungo-Maziwa		CRS: Central railway station	
Example					
05.00	*CRS to UM*	*0*	*17.00*	*CRS to UM*	*9*

5. What is the train maximum service speed (km/h)?

25KPH

6. What is the train average service speed (km/h)?

16KPH

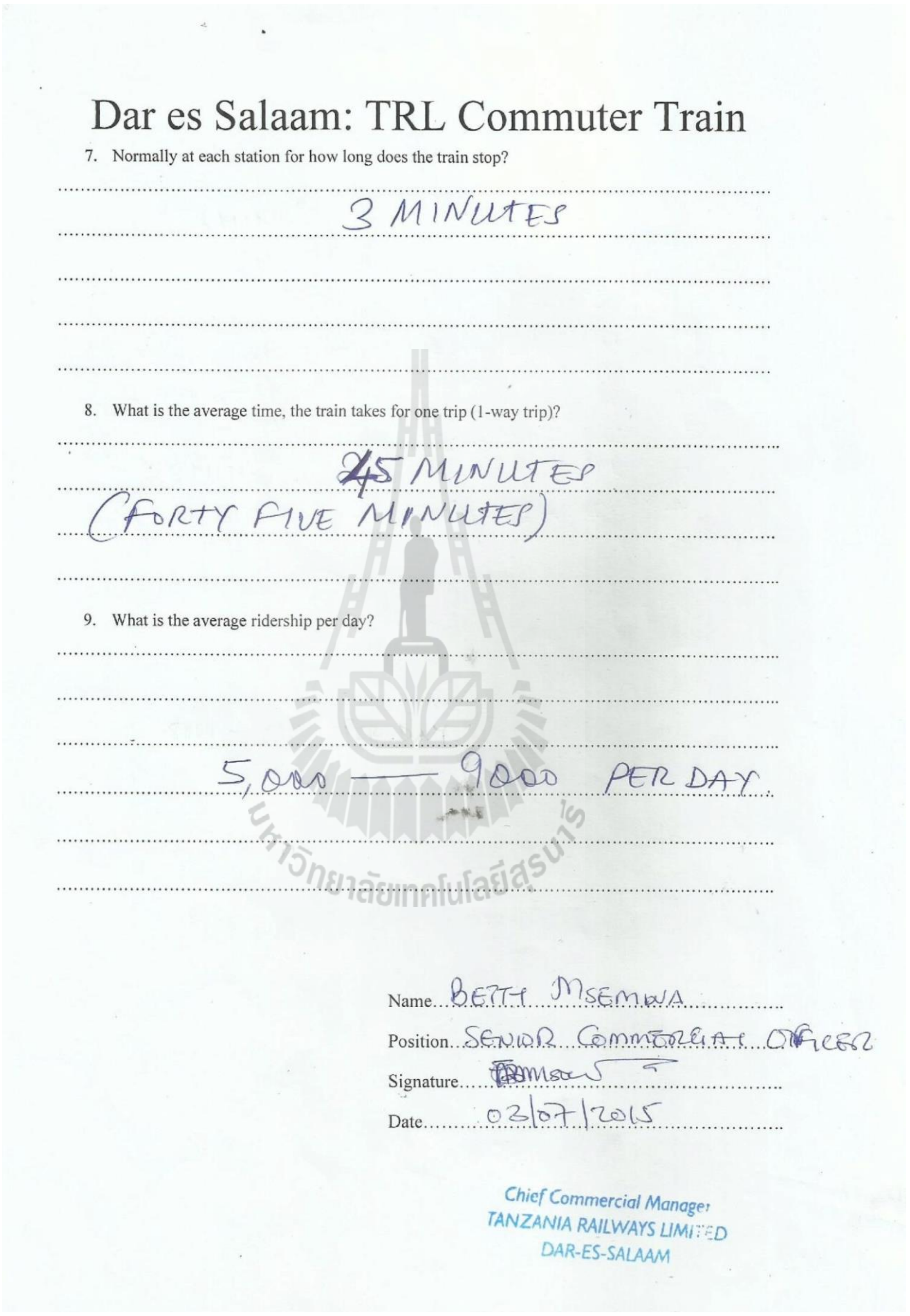
Name: AWIN MOTONELO

Position: LOCOMOTIVE TECHNICIAN

Signature: [illegible]

Date: 01/07/2015

CHIEF MECHANICAL ENGINEER
TANZANIA RAILWAYS LIMITED
• DAR ES SALAAM •



Dar es Salaam: TRL Commuter Train

7. Normally at each station for how long does the train stop?

3 MINUTES

8. What is the average time, the train takes for one trip (1-way trip)?

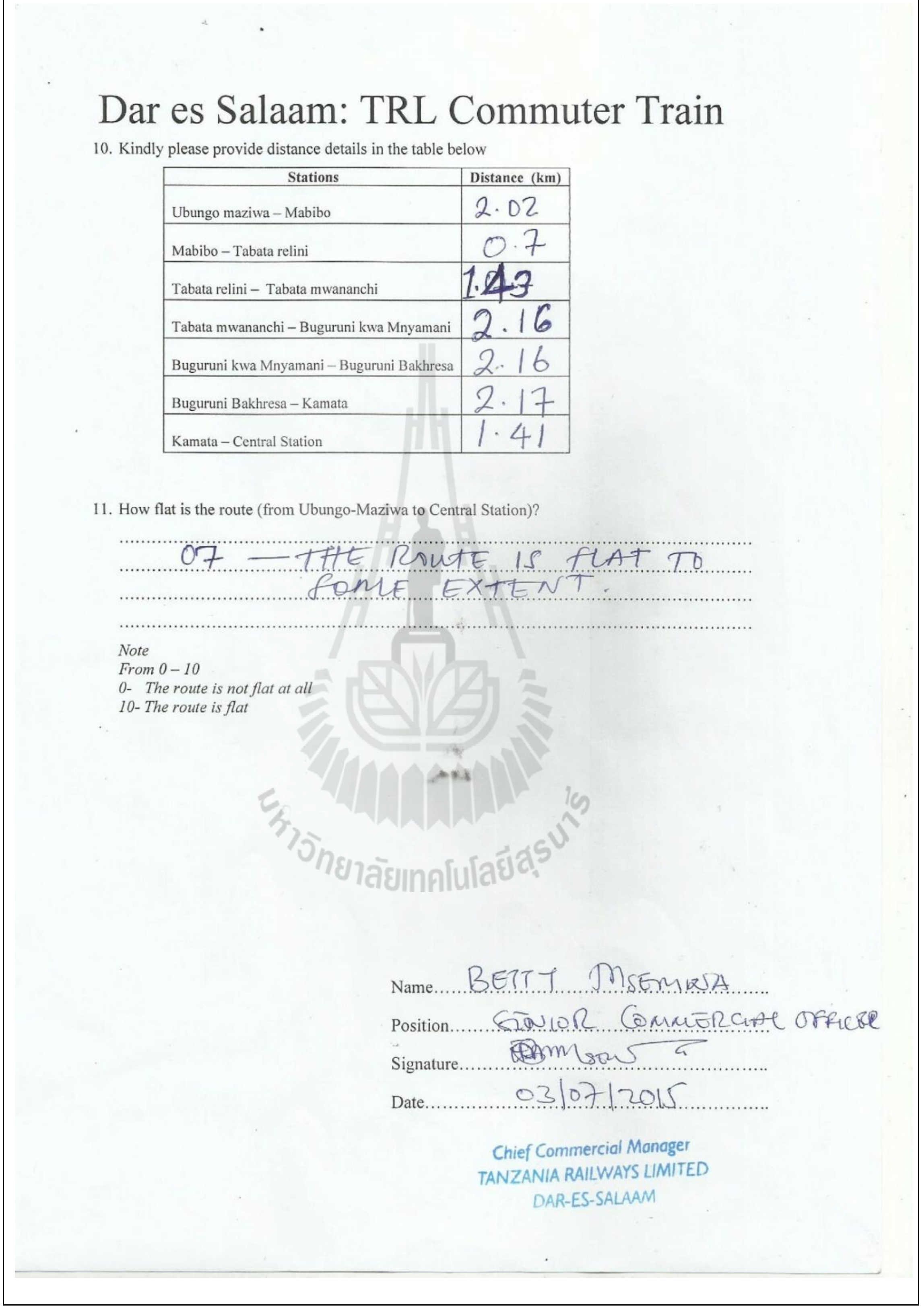
45 MINUTES
(FORTY FIVE MINUTES)

9. What is the average ridership per day?

5,000 — 9000 PER DAY

Name: BETTY MSEMWA
Position: SENIOR COMMERCIAL OFFICER
Signature: [illegible]
Date: 03/07/2015

Chief Commercial Manager
TANZANIA RAILWAYS LIMITED
DAR-ES-SALAAM



Dar es Salaam: TRL Commuter Train

10. Kindly please provide distance details in the table below

Stations	Distance (km)
Ubungo maziwa – Mabibo	2.02
Mabibo – Tabata relini	0.7
Tabata relini – Tabata mwananchi	1.43
Tabata mwananchi – Buguruni kwa Mnyamani	2.16
Buguruni kwa Mnyamani – Buguruni Bakhresa	2.16
Buguruni Bakhresa – Kamata	2.17
Kamata – Central Station	1.41

11. How flat is the route (from Ubungo-Maziwa to Central Station)?

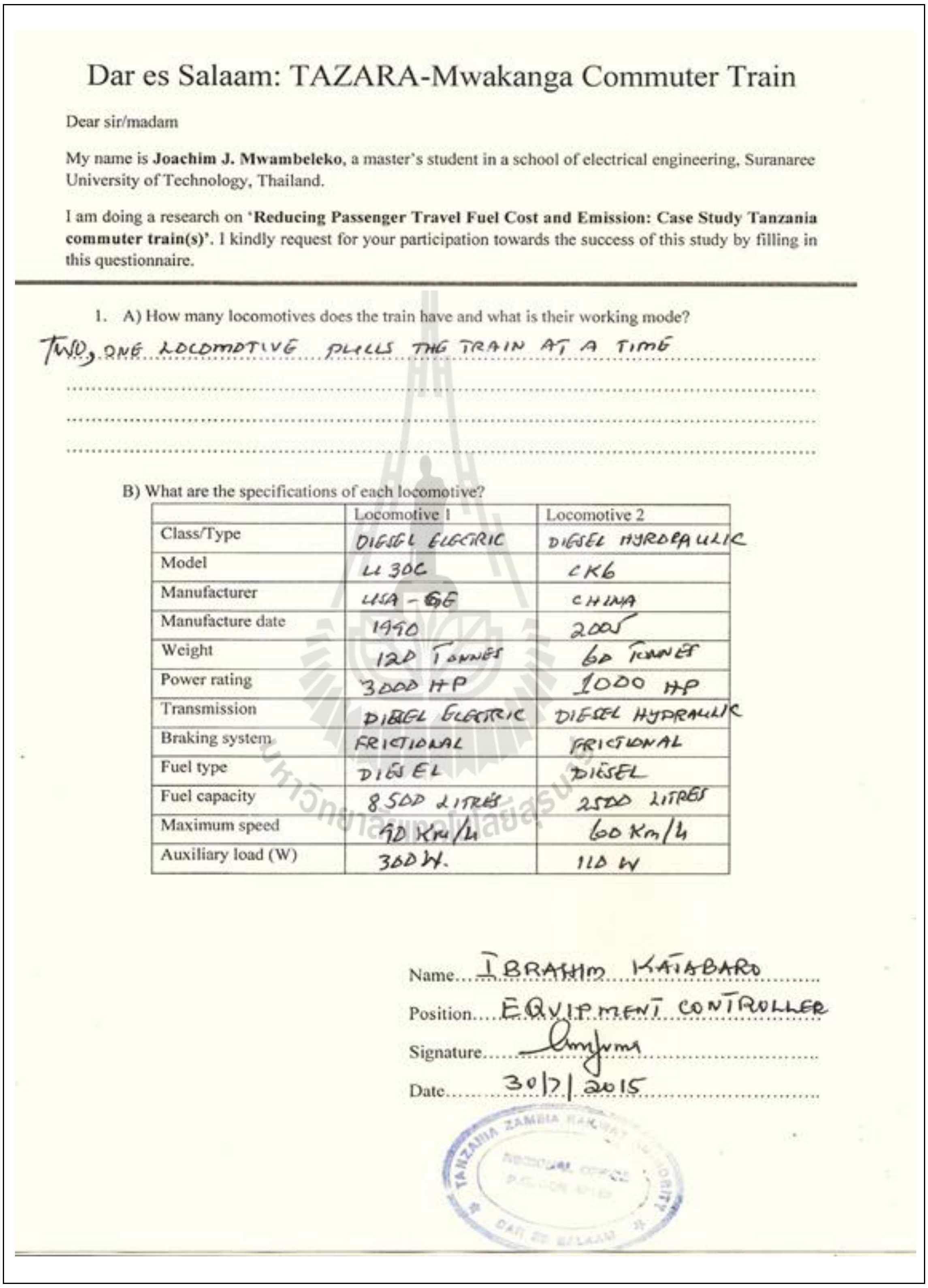
07 — THE ROUTE IS FLAT TO SOME EXTENT.

Note
From 0 – 10
0- The route is not flat at all
10- The route is flat

Name: BETTY MSEMWA
Position: SENIOR COMMERCIAL OFFICER
Signature: [illegible]
Date: 03/07/2015

Chief Commercial Manager
TANZANIA RAILWAYS LIMITED
DAR-ES-SALAAM

A.2 TAZARA Commuter Train Performance Data



Dar es Salaam: TAZARA-Mwakanga Commuter Train

Dear sir/madam

My name is **Joachim J. Mwambeleko**, a master's student in a school of electrical engineering, Suranaree University of Technology, Thailand.

I am doing a research on '**Reducing Passenger Travel Fuel Cost and Emission: Case Study Tanzania commuter train(s)**'. I kindly request for your participation towards the success of this study by filling in this questionnaire.

1. A) How many locomotives does the train have and what is their working mode?

TWO, ONE LOCOMOTIVE PULLS THE TRAIN AT A TIME

B) What are the specifications of each locomotive?

	Locomotive 1	Locomotive 2
Class/Type	DIESEL ELECTRIC	DIESEL HYRDRAULIC
Model	U 30C	CK6
Manufacturer	USA - GE	CHINA
Manufacture date	1990	2005
Weight	120 TONNES	60 TONNES
Power rating	3000 HP	1000 HP
Transmission	DIESEL ELECTRIC	DIESEL HYDRAULIC
Braking system	FRICTIONAL	FRICTIONAL
Fuel type	DIESEL	DIESEL
Fuel capacity	8500 LITRES	2500 LITRES
Maximum speed	90 Km/h	60 Km/h
Auxiliary load (W)	300 W.	110 W

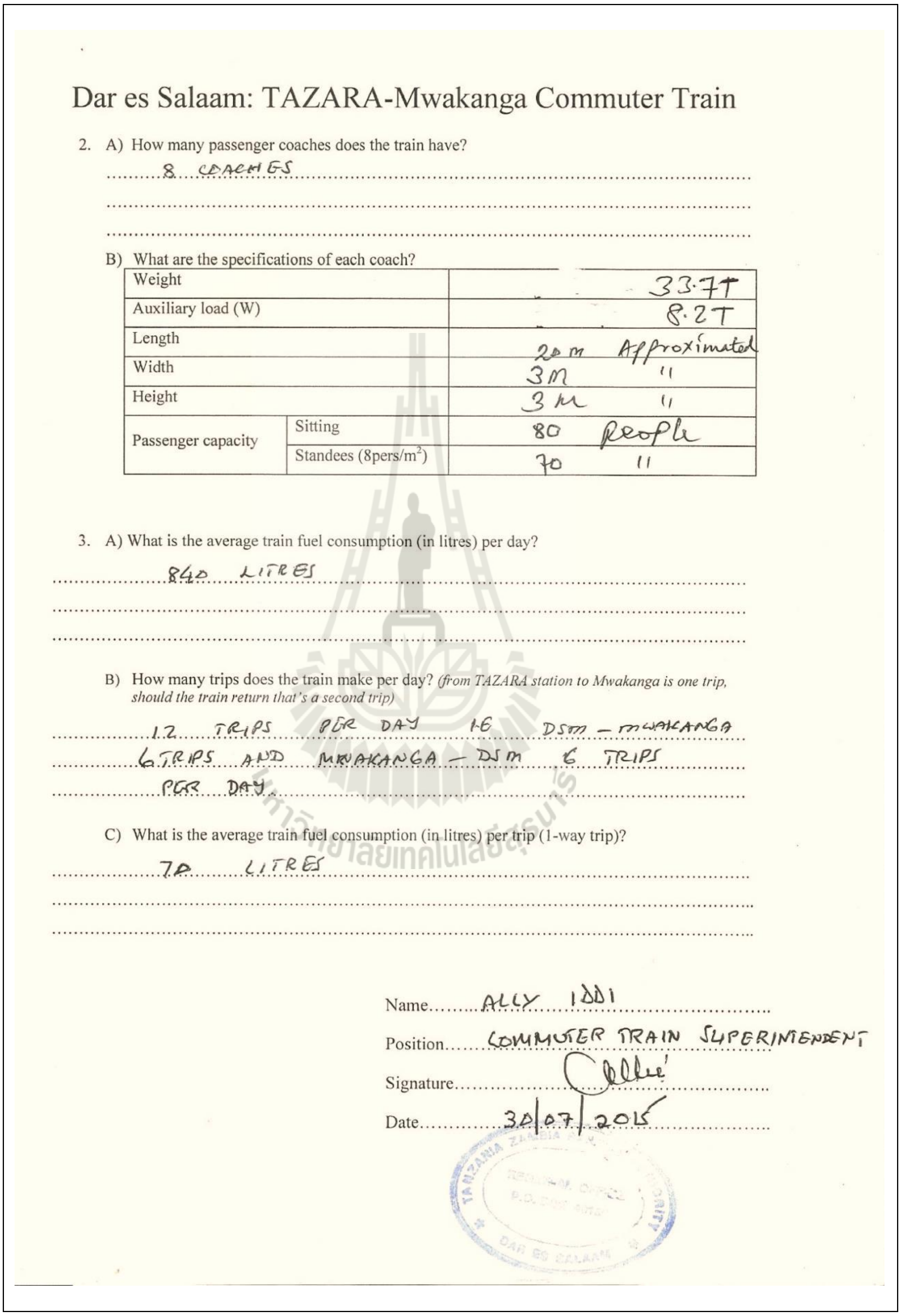
Name: IBRAHIM KATABARO

Position: EQUIPMENT CONTROLLER

Signature: [illegible]

Date: 30/7/2015

TANZANIA ZAMBIA RAILWAY AUTHORITY
REGIONAL OFFICE
DAR ES SALAAM



Dar es Salaam: TAZARA-Mwakanga Commuter Train

2. A) How many passenger coaches does the train have?

8 COACHES

B) What are the specifications of each coach?

Weight		33.7T
Auxiliary load (W)		8.2T
Length		20 m Approximated
Width		3 m "
Height		3 m "
Passenger capacity	Sitting	80 People
	Standees (8pers/m^2)	70 "

3. A) What is the average train fuel consumption (in litres) per day?

840 LITRES

B) How many trips does the train make per day? *(from TAZARA station to Mwakanga is one trip, should the train return that's a second trip)*

12 TRIPS PER DAY I.E DSM – MWAKANGA 6 TRIPS AND MWAKANGA – DSM 6 TRIPS PER DAY.

C) What is the average train fuel consumption (in litres) per trip (1-way trip)?

70 LITRES

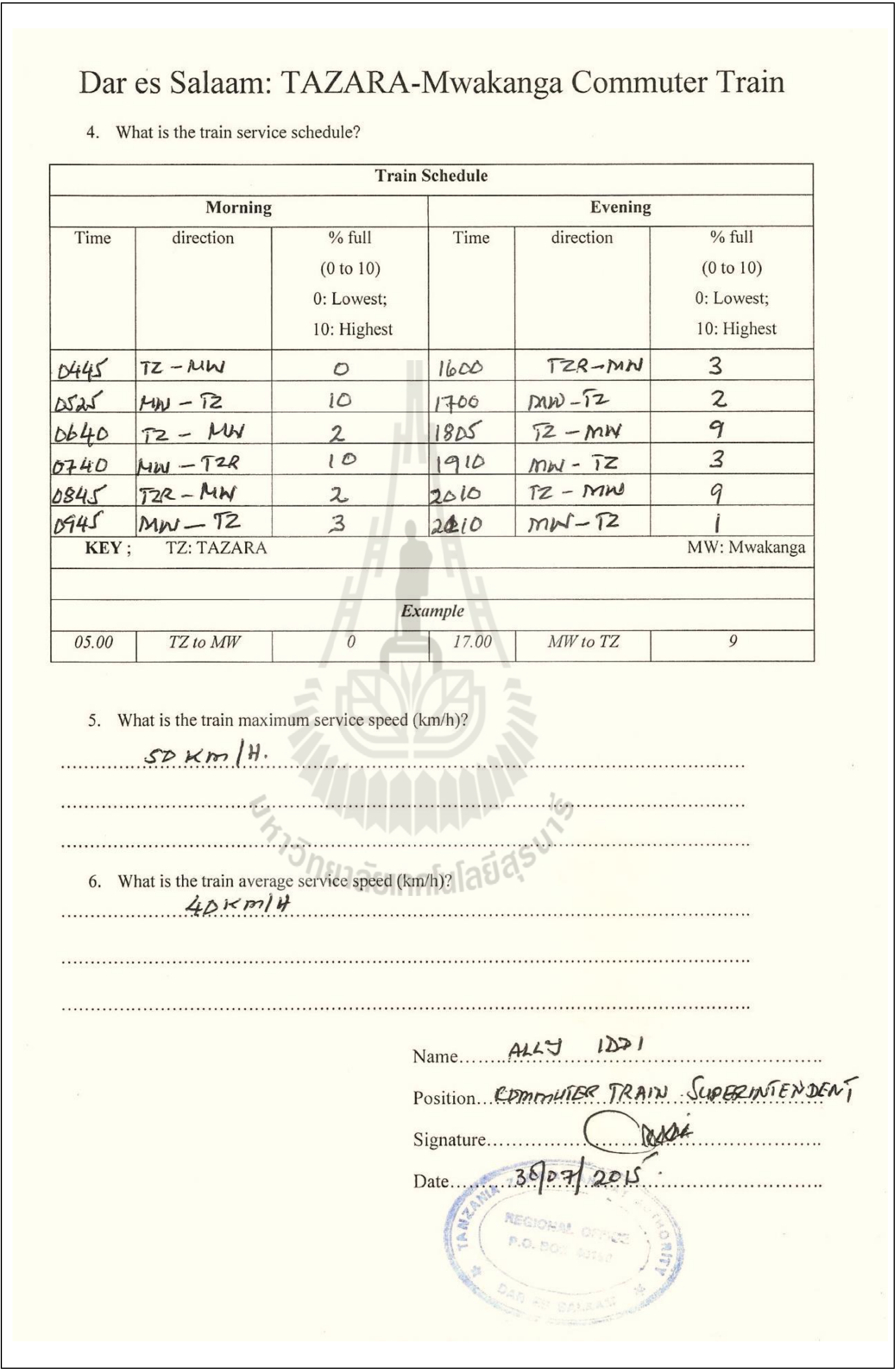
Name........ALLY IDDI

Position........COMMUTER TRAIN SUPERINTENDENT

Signature........[illegible]

Date........30/07/2015

TANZANIA ZAMBIA R... AUTHORITY
REGIONAL OFFICE
P.O. BOX ...
DAR ES SALAAM



Dar es Salaam: TAZARA-Mwakanga Commuter Train

4. What is the train service schedule?

Train Schedule					
Morning			**Evening**		
Time	direction	% full (0 to 10) 0: Lowest; 10: Highest	Time	direction	% full (0 to 10) 0: Lowest; 10: Highest
0445	TZ – MW	0	1600	TZR – MW	3
0525	MW – TZ	10	1700	MW – TZ	2
0640	TZ – MW	2	1805	TZ – MW	9
0740	MW – TZR	10	1910	MW – TZ	3
0845	TZR – MW	2	2010	TZ – MW	9
0945	MW – TZ	3	2110	MW – TZ	1
KEY ;	TZ: TAZARA				MW: Mwakanga
Example					
05.00	*TZ to MW*	*0*	*17.00*	*MW to TZ*	*9*

5. What is the train maximum service speed (km/h)?

50 Km/H.

6. What is the train average service speed (km/h)?

40 Km/H

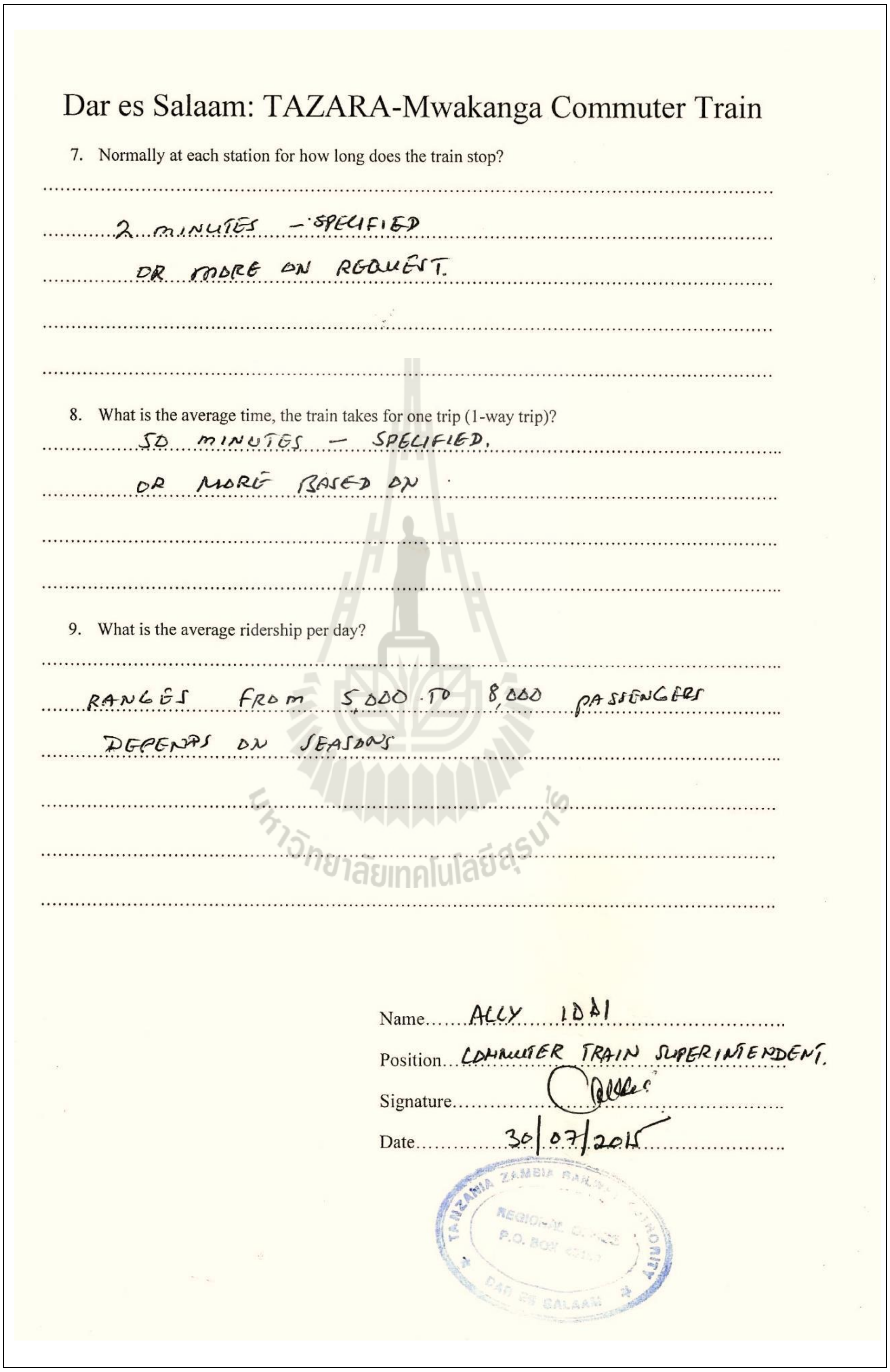
Name……ALLY IDDI……

Position…COMMUTER TRAIN SUPERINTENDENT……

Signature……[illegible]……

Date……30/07/2015……

TANZANIA [illegible] AUTHORITY
REGIONAL OFFICE
P.O. BOX [illegible]
DAR ES SALAAM



Dar es Salaam: TAZARA-Mwakanga Commuter Train

7. Normally at each station for how long does the train stop?

2 MINUTES - SPECIFIED
OR MORE ON REQUEST.

8. What is the average time, the train takes for one trip (1-way trip)?

50 MINUTES - SPECIFIED.
OR MORE BASED ON

9. What is the average ridership per day?

RANGES FROM 5,000 TO 8,000 PASSENGERS
DEPENDS ON SEASONS

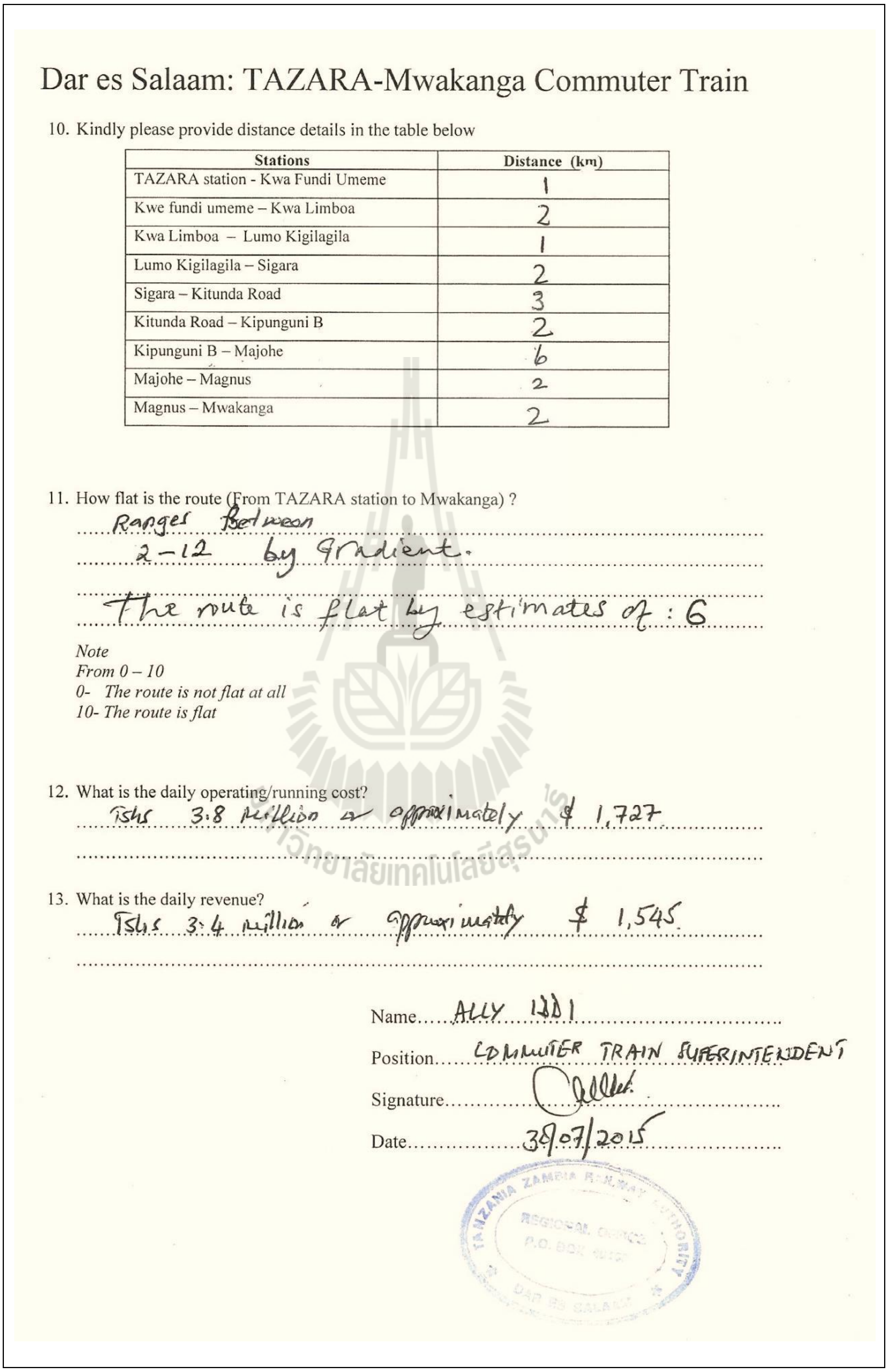
Name: ALLY IDDI

Position: COMMUTER TRAIN SUPERINTENDENT.

Signature: [illegible]

Date: 30/07/2015

TANZANIA ZAMBIA RAILWAY AUTHORITY
REGIONAL OFFICE
P.O. BOX [illegible]
DAR ES SALAAM



Dar es Salaam: TAZARA-Mwakanga Commuter Train

10. Kindly please provide distance details in the table below

Stations	Distance (km)
TAZARA station - Kwa Fundi Umeme	1
Kwe fundi umeme – Kwa Limboa	2
Kwa Limboa – Lumo Kigilagila	1
Lumo Kigilagila – Sigara	2
Sigara – Kitunda Road	3
Kitunda Road – Kipunguni B	2
Kipunguni B – Majohe	6
Majohe – Magnus	2
Magnus – Mwakanga	2

11. How flat is the route (From TAZARA station to Mwakanga) ?
Ranges between
2-12 by gradient.

The route is flat by estimates of : 6

Note
From 0 – 10
0- The route is not flat at all
10- The route is flat

12. What is the daily operating/running cost?
Tshs 3.8 Million or approximately $ 1,727

13. What is the daily revenue?
Tshs 3.4 million or approximately $ 1,545

Name.....ALLY IDDI.....
Position.....COMMUTER TRAIN SUPERINTENDENT
Signature.....[illegible].....
Date.....30/07/2015.....

TANZANIA ZAMBIA RAILWAY AUTHORITY
REGIONAL OFFICE
P.O. BOX [illegible]
DAR ES SALAAM

APPENDIX B

OPTIMIZATION ALGORITHMS TESTING RESULTS

B.1 Testing the Algorithms Using Ackely's Function

The Ackley's function is given as

$$f(x) = -Aexp\left(-B\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(Cx_i)\right) + A + exp(1) \quad \text{(B.1)}$$

where n is the number of variables and, A, B and C are constant numbers. Shown in Figure B.1 is the Ackley's function plotted in two dimensional space. The function has a global minimum at $x = 0$ where $f(x) = 0$.

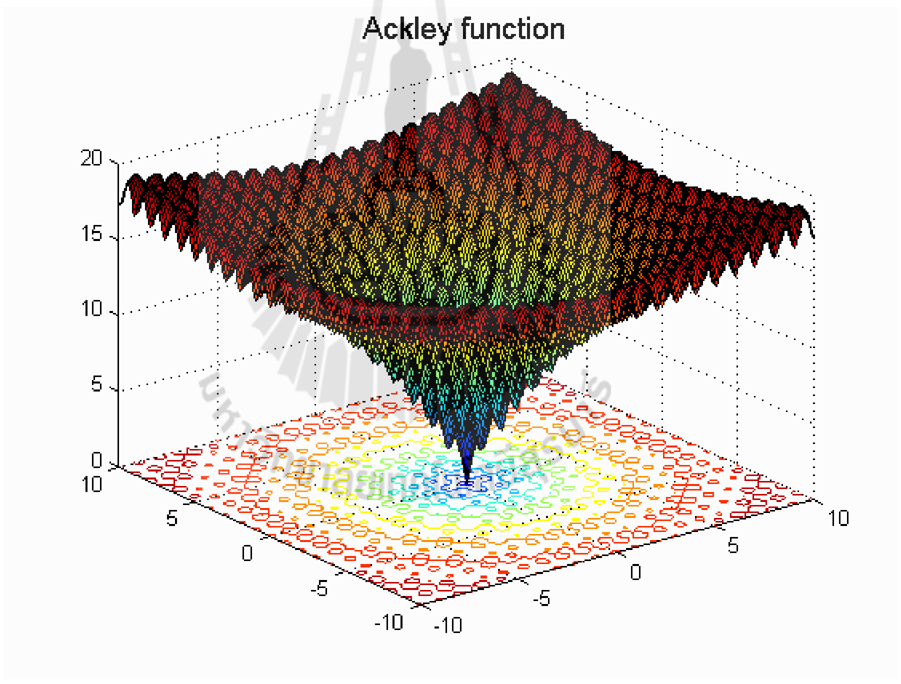
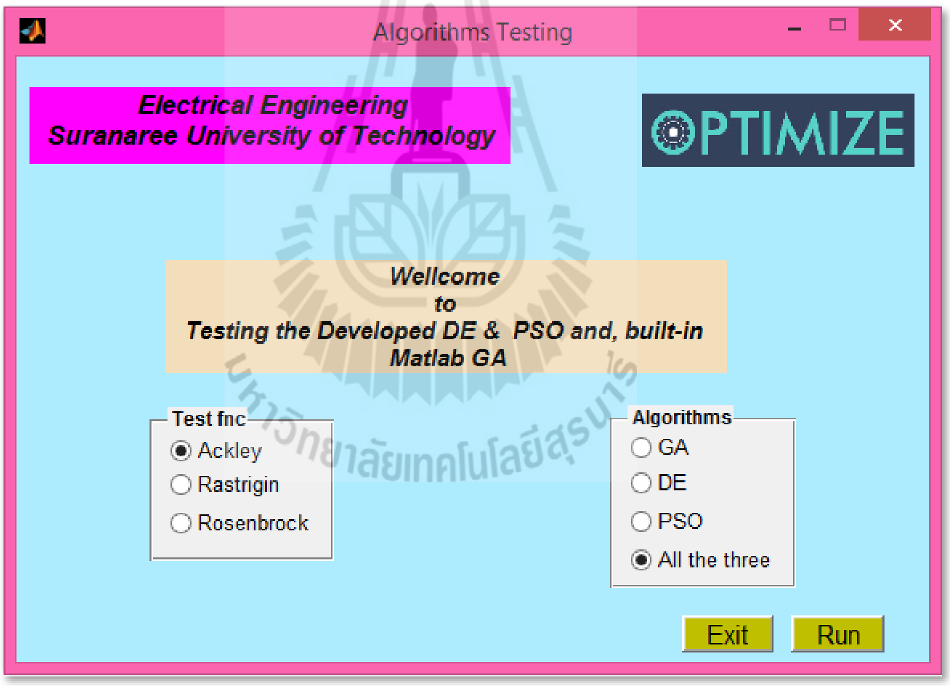


Figure B.1 Ackley's function plotted in two dimensional space

During the testing, values for n, A, B and C were set as $n = 3$, $A = 20$, $B = 0.2$, and $C = 2\pi$. And, $x_i \in [-5 \quad 10]$. Other settings related to the optimization algorithms can be found in the 'Options.m' function given in appendix D

As shown in Figure B.2, the Ackley's function and all the three optimization algorithms were selected, and when the program was executed, the optimization testing results were as presented in Figure B.3 through Figure B.6.

The MATLAB® built in GA plots the best and mean fitness values, and the developed DE and PSO plot the best, mean and the worst fitness values. The worst fitness value was added in the plot for the purpose of this thesis work so as to evaluate (i) what were the worst possible combination of the variables (in speed profile optimization) and (ii) the convergence of the individuals (or particles).


Algorithms Testing

Electrical Engineering
Suranaree University of Technology

OPTIMIZE

Wellcome
to
Testing the Developed DE & PSO and, built-in Matlab GA

Test fnc: [✓ Ackley] Rastrigin Rosenbrock

Algorithms: GA DE PSO [✓ All the three]

Figure B.2 Ackley's function and all the three optimization algorithms selected

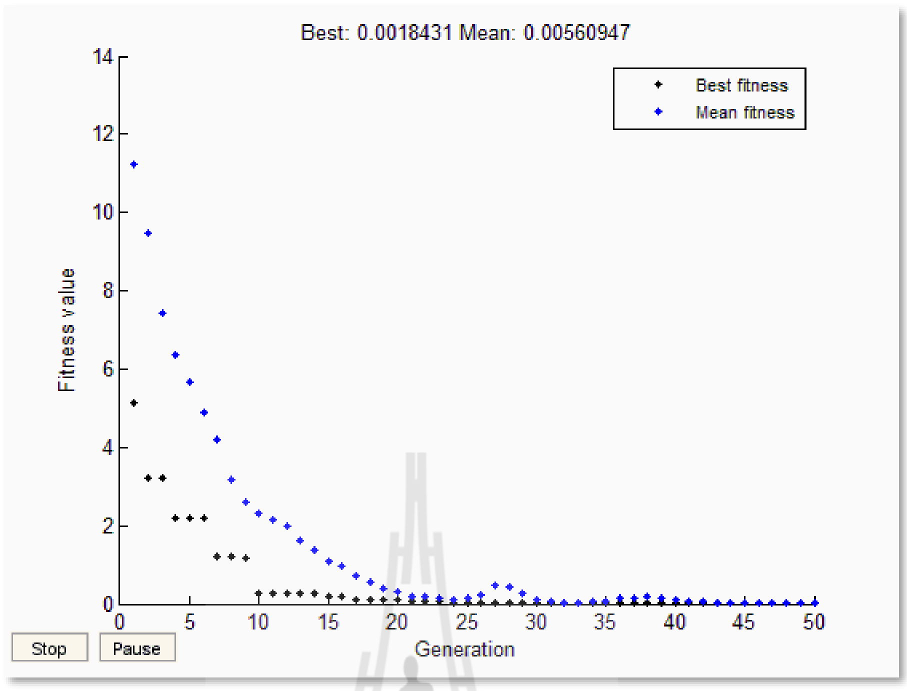


Figure B.3 Ackley's function, GA fitness value plot

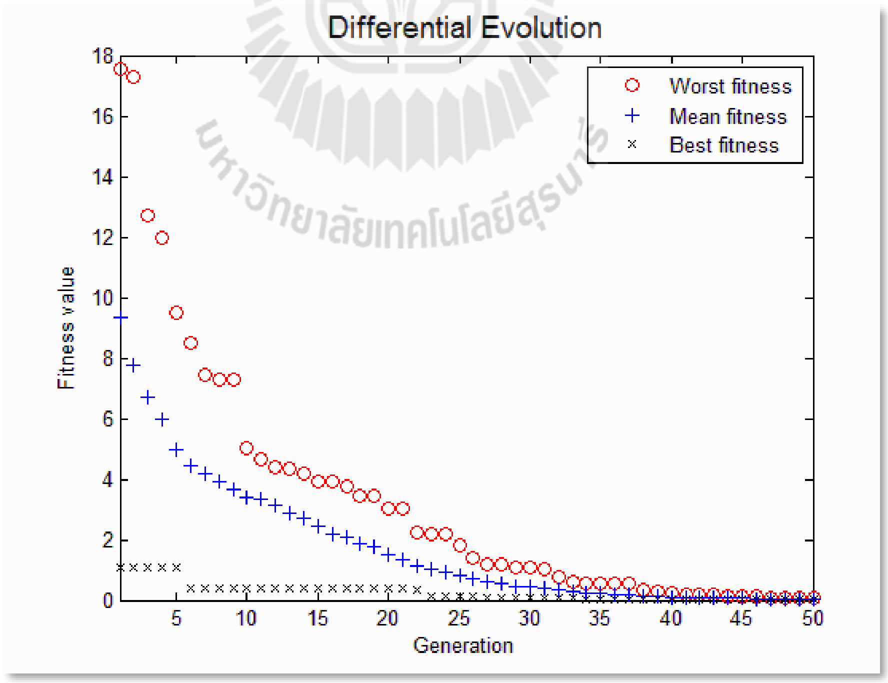


Figure B.4 Ackley's function, DE fitness value plot

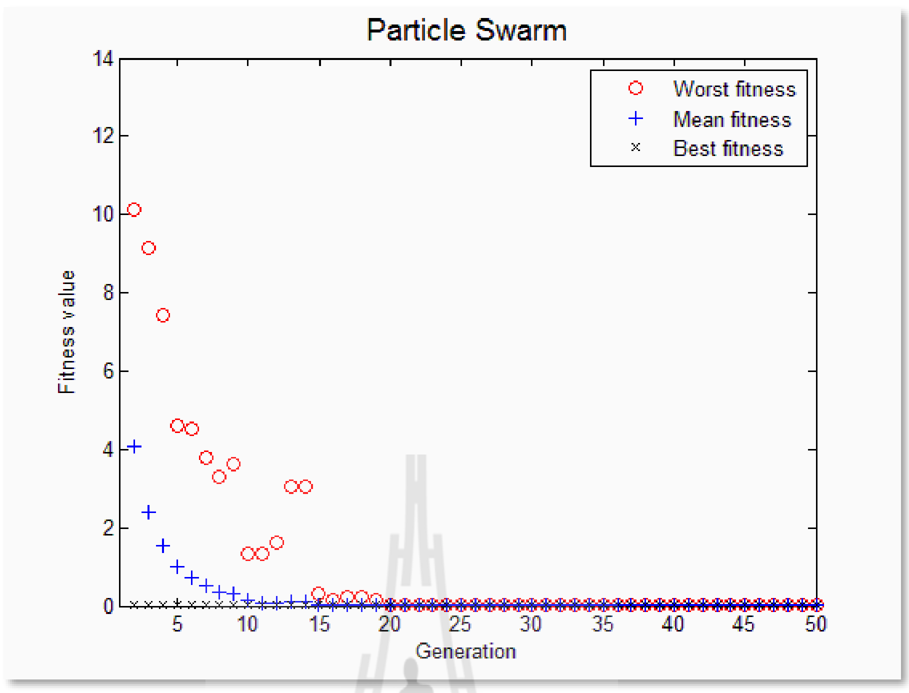
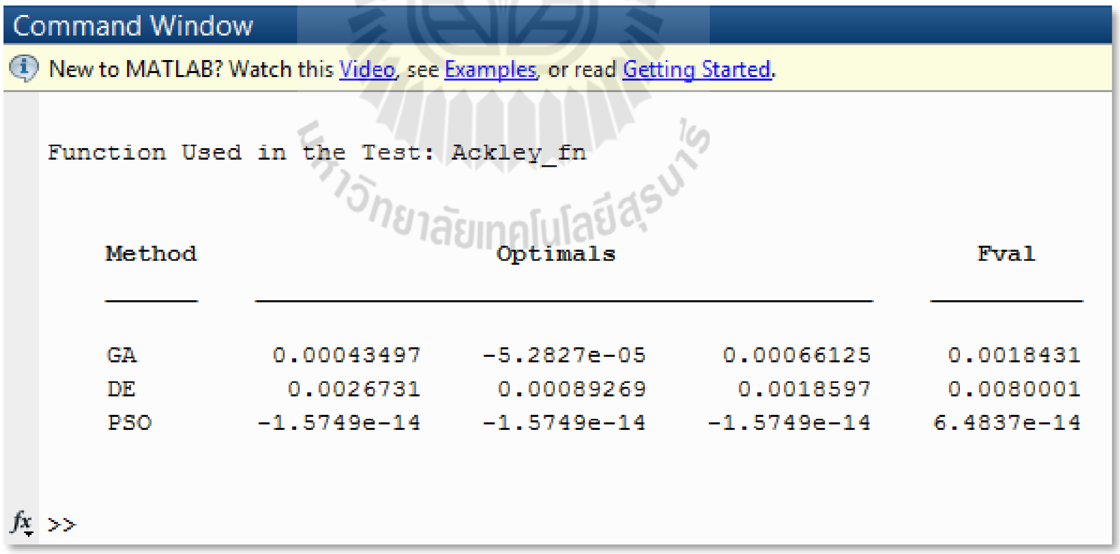


Figure B.5 Ackley's function, PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
Function Used in the Test: Ackley_fn

    Method                         Optimals                          Fval
    ______      ______________________________________________   __________

    GA          0.00043497     -5.2827e-05      0.00066125       0.0018431
    DE           0.0026731      0.00089269       0.0018597       0.0080001
    PSO        -1.5749e-14     -1.5749e-14     -1.5749e-14      6.4837e-14

fx >>
```

Figure B.6 Ackley's function; GA, DE and PSO optimal values

B.2 Testing the Algorithms Using Rastrigin's Function

The Rastrigin's function is defined as

$$f(x) = An + \sum_{i=1}^{n} [x_i^2 - A\cos(2\pi x_i)] \quad where\ x = [x_1, \dots, x_n] \in \mathbb{R}^n \tag{B.2}$$

where A is a constant number. Shown in Figure B.7 is the Rastrigin's function plotted in two dimensional space. The function has a global minimum at $x = 0$ where $f(x) = 0$.

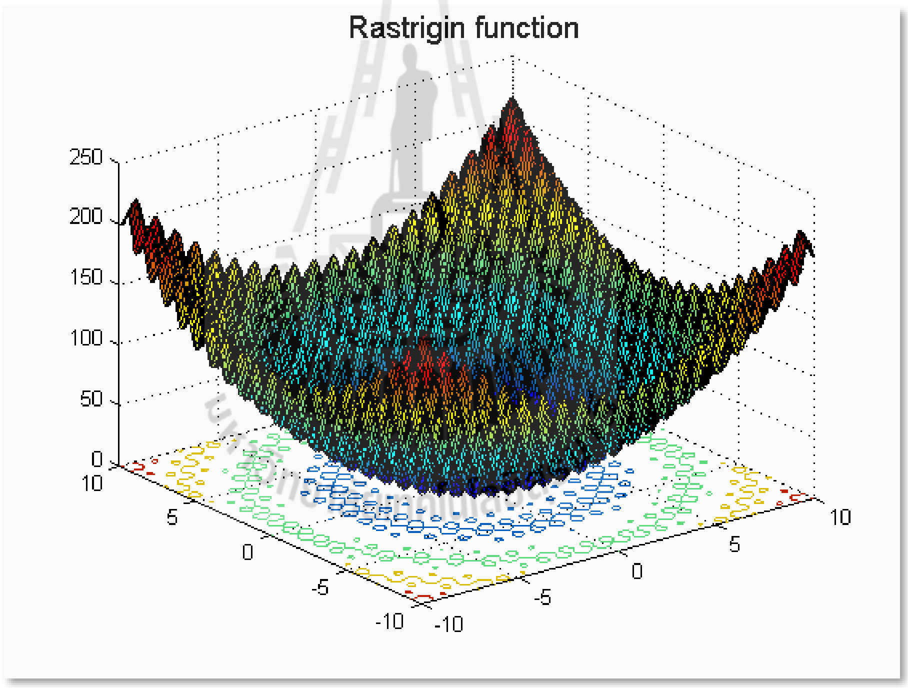
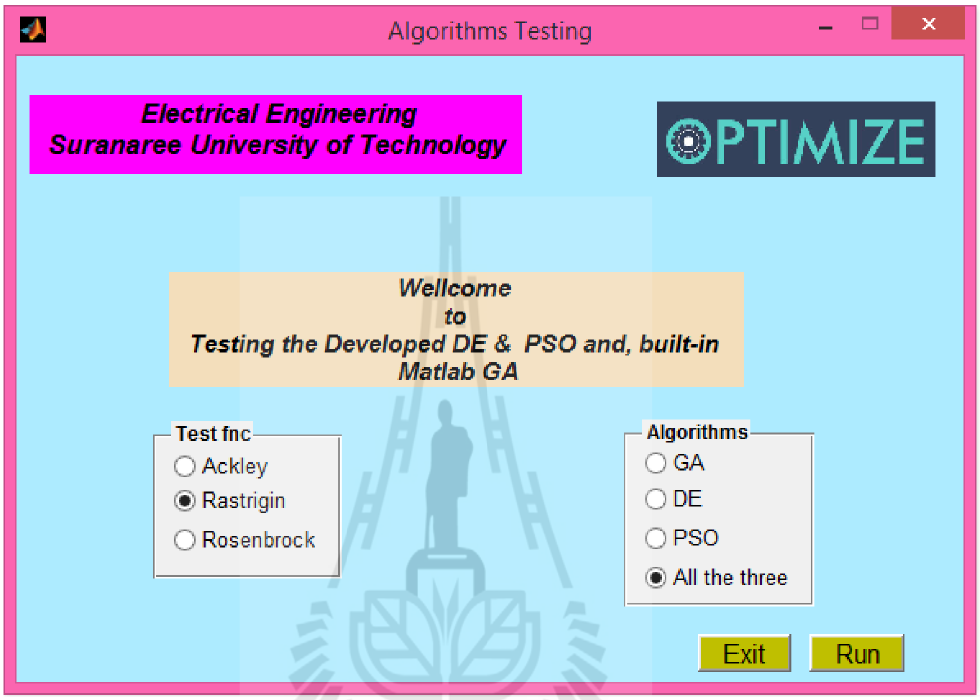


Figure B.7 Rastrigin's function plotted in two dimensional space

During the testing, values for n and A were set as $n = 3$ (testing with three variables) and $A = 10$. And, $x_i \in [-5 \quad 10]$. As mentioned previously, other settings related to the optimization algorithms can be found in the 'Options.m' function given in appendix D.

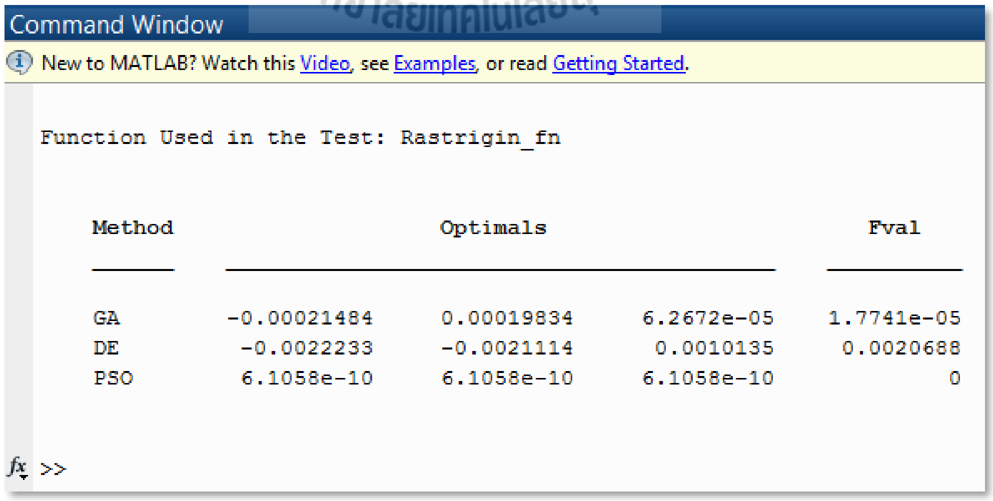
As shwon in Figure B.8, Rastrigin's function and all the three optimization algorithms were selected, and when the program was executed, the optimization testing results were as presented in Figure B.9 through Figure B.12.



Test fnc: Ackley [✓ Rastrigin] Rosenbrock

Algorithms: GA DE PSO [✓ All the three]

Figure B.8 Rastrigin's function and all the three optimization algorithms selected


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
Function Used in the Test: Rastrigin_fn


    Method                          Optimals                            Fval
    ______    ________________________________________________    __________

    GA        -0.00021484     0.00019834     6.2672e-05    1.7741e-05
    DE         -0.0022233     -0.0021114      0.0010135     0.0020688
    PSO        6.1058e-10     6.1058e-10     6.1058e-10             0

>>
```

Figure B.9 Rastrigin's function; GA, DE and PSO optimal values

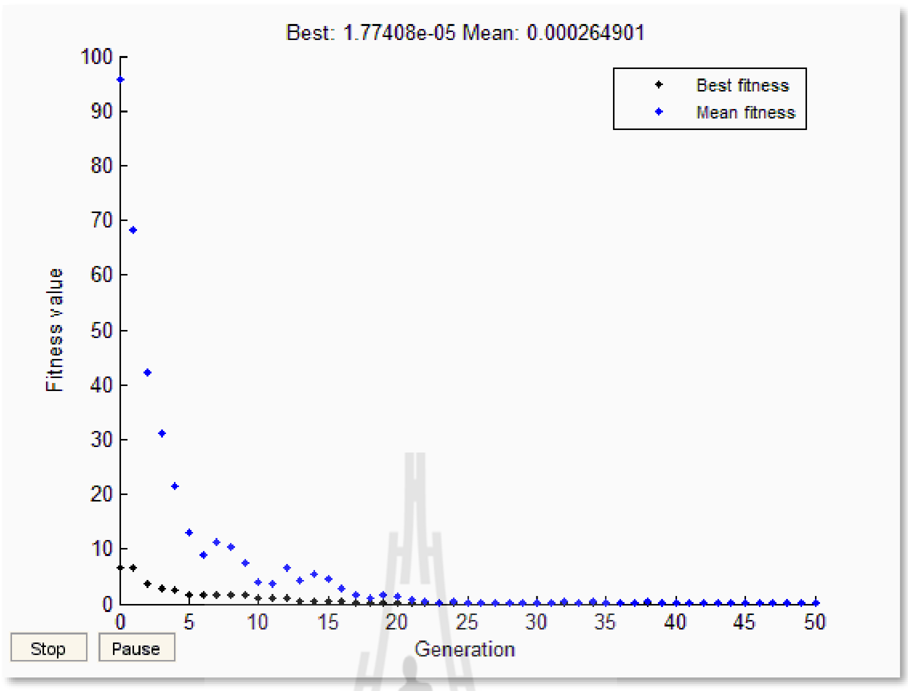


Figure B.10 Rastrigin's function, GA fitness value plot

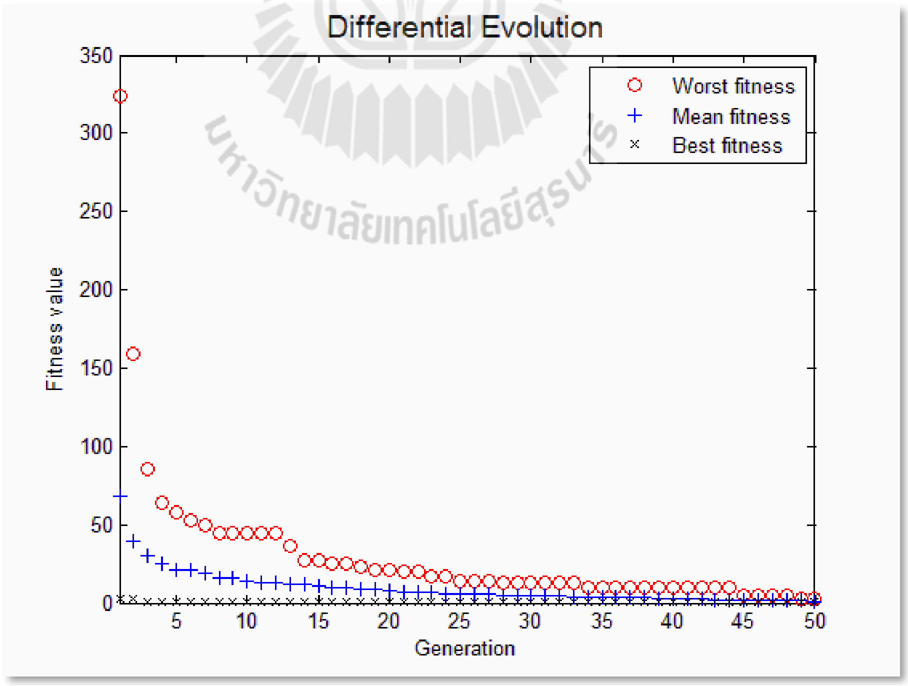


Figure B.11 Rastrigin's function, DE fitness value plot

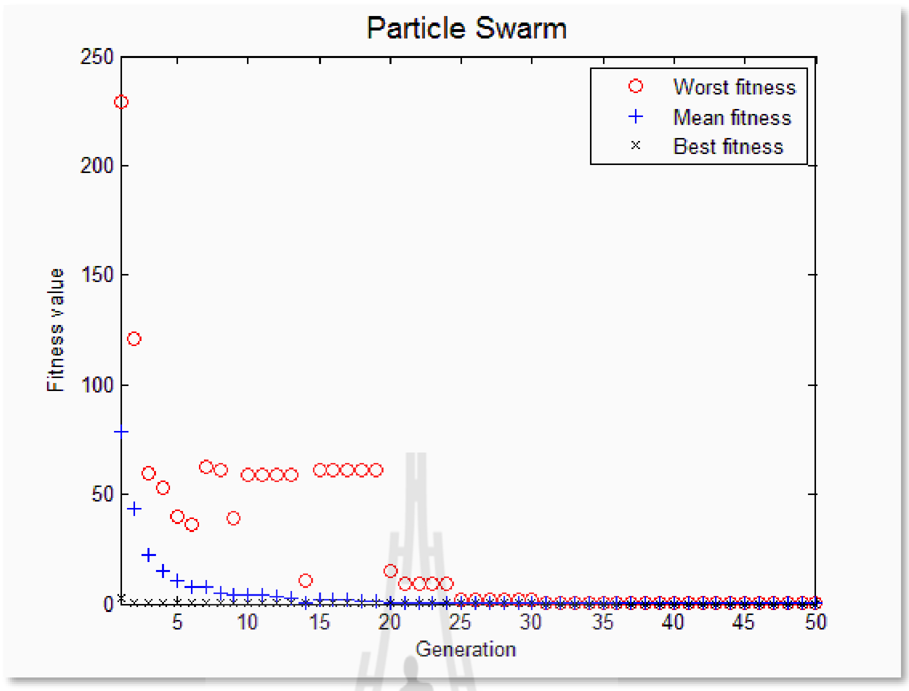


Figure B.12 Rastrigin's function, PSO fitness value plot

B.3 Testing the Algorithms Using Rosenbrock's Function

The Rosenbrock's function is defined as

$$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2] \; where \; x = [x_1, \dots, x_n] \in \mathbb{R}^n \qquad \text{(B.3)}$$

Shown in Figure B.13 is the Rosenbrock's function plotted in two dimensional space. The function has a global minimum at $x = 1$ where $f(x) = 0$.

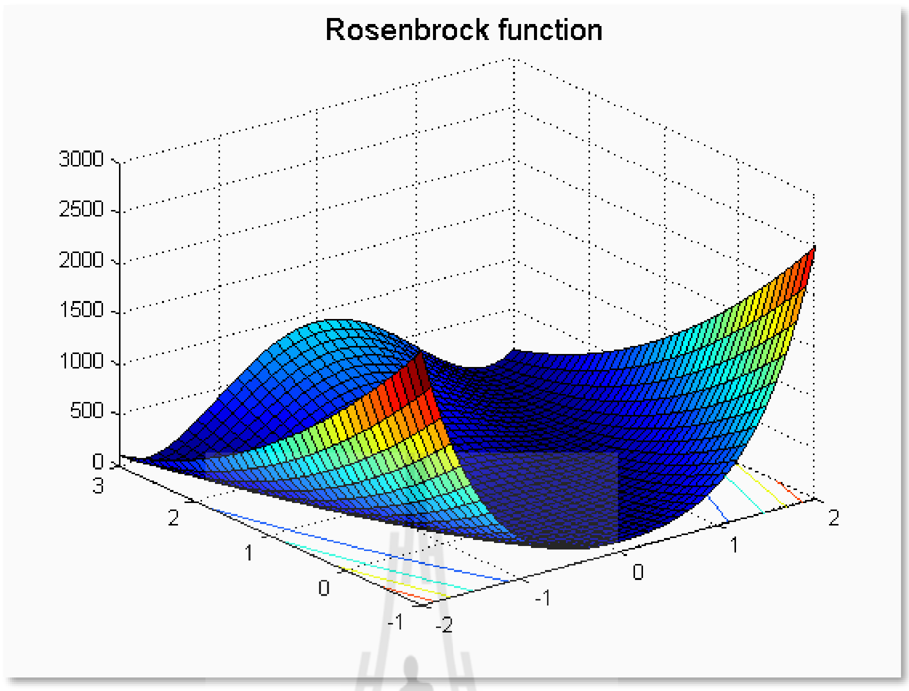
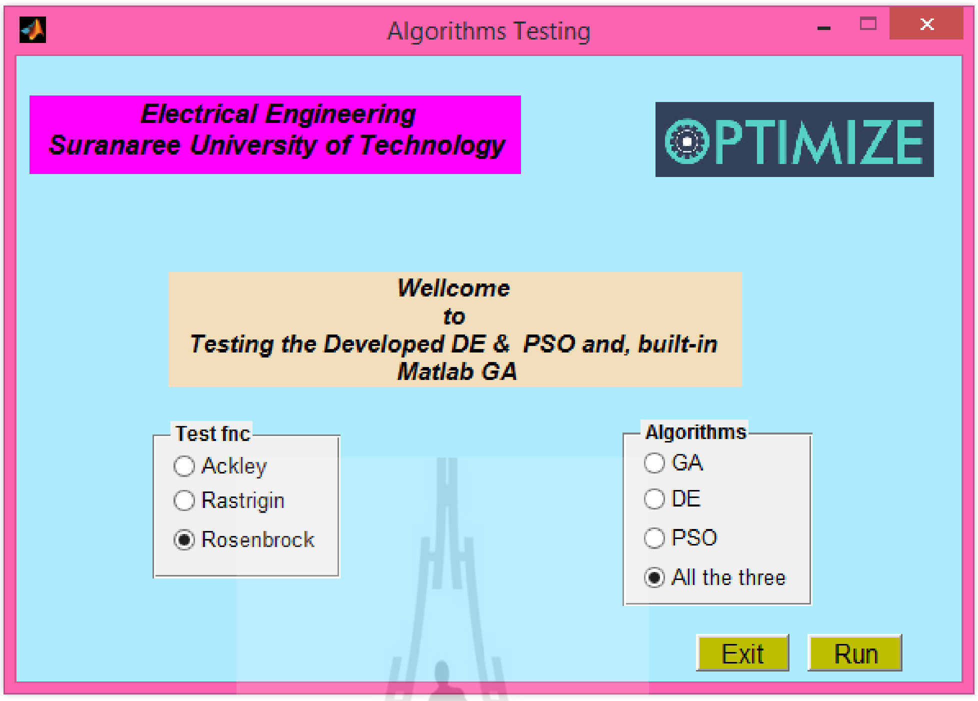


Figure B.13 Rosenbrock's function plotted in two dimensional space

During the testing, the values for n was set as $n = 3$ (testing with three variables) and, $x_i \in [-5 \quad 10]$. As mentioned previously, other option settings related to the optimization algorithms can be found in the 'Options.m' function given in appendix D.

As shwon in Figure B.14, Rosenbrock's function and all the three optimization algorithms were selected, and when the program was executed, the optimization testing results were as presented in Figure B.15 through Figure B.18.



Algorithms Testing

Electrical Engineering
Suranaree University of Technology

Wellcome
to
Testing the Developed DE & PSO and, built-in
Matlab GA

Test fnc
Ackley
Rastrigin
[✓ Rosenbrock]

Algorithms
GA
DE
PSO
[✓ All the three]

Exit
Run

Figure B.14 Rosenbrock's function and all the three optimization algorithms selected

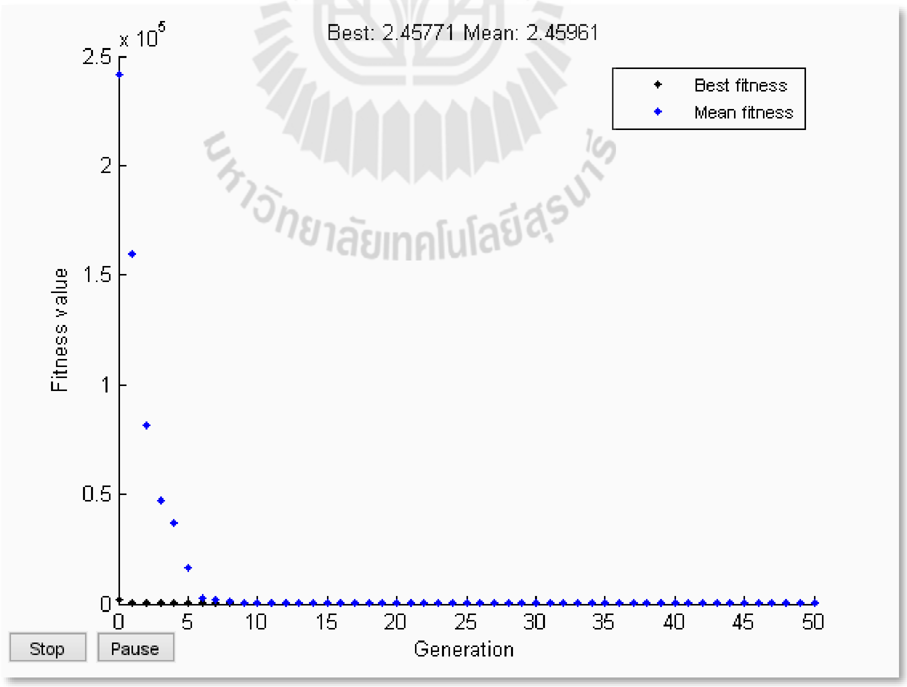


Figure B.15 Rosenbrock's function, GA fitness value plot

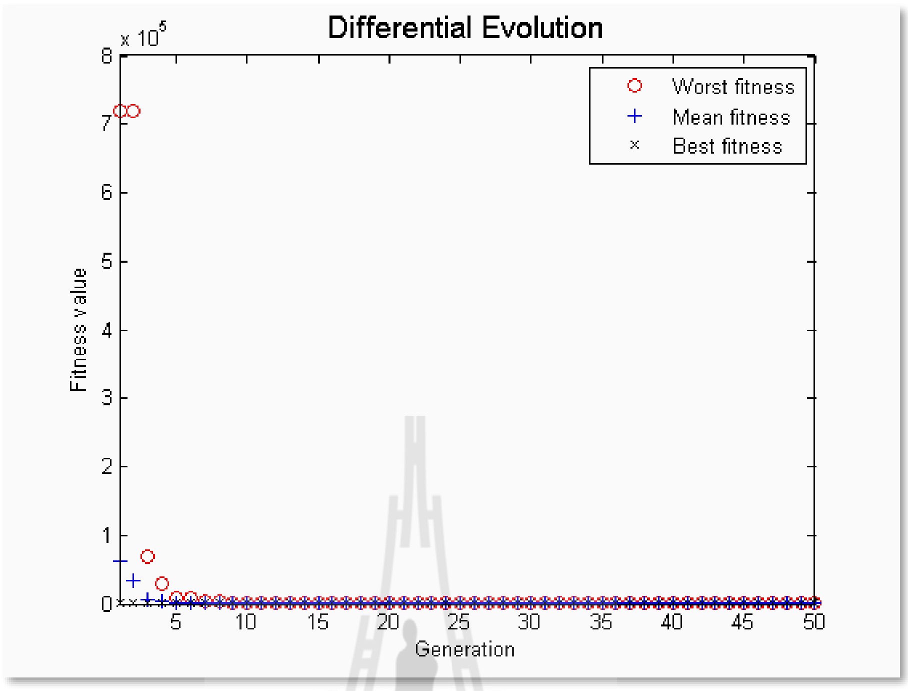


Figure B.16 Rosenbrock's function, DE fitness value plot

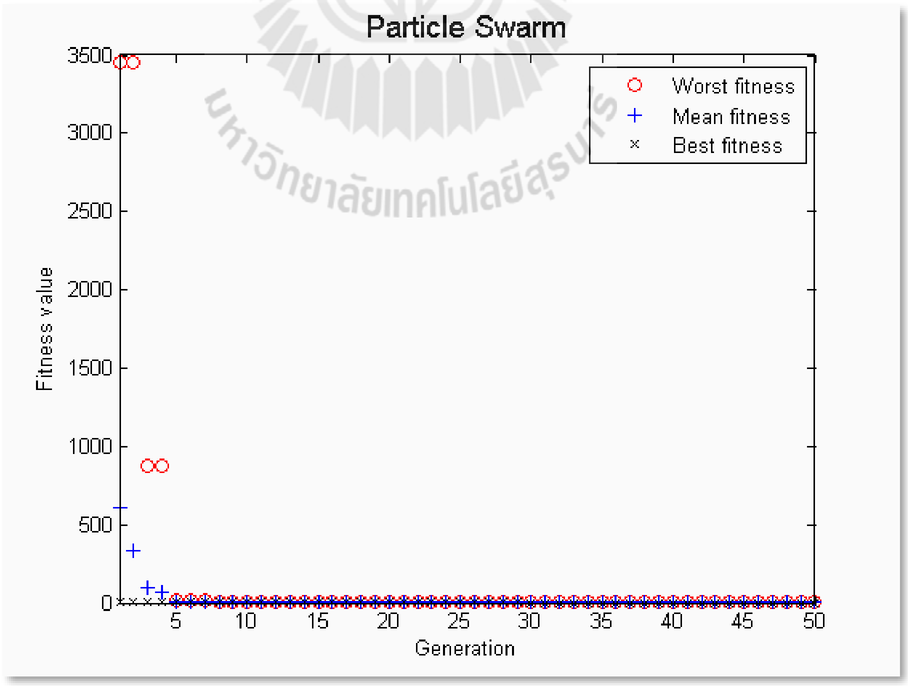
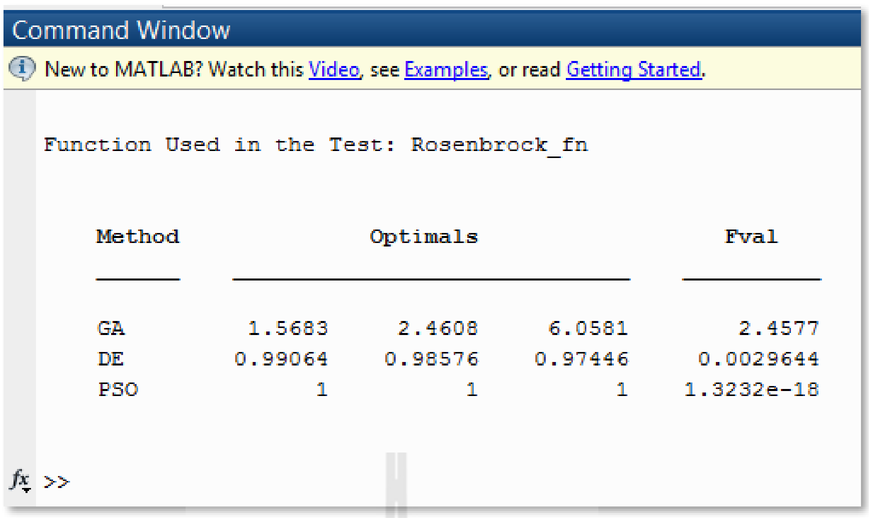


Figure B.17 Rosenbrock's function, PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
Function Used in the Test: Rosenbrock_fn


    Method              Optimals                        Fval
    ______    _____________________________    __________

    GA         1.5683       2.4608      6.0581       2.4577
    DE        0.99064      0.98576     0.97446    0.0029644
    PSO             1            1           1   1.3232e-18

fx >>
```

Figure B.18 Rosenbrock's function; GA, DE and PSO optimal values

NOTE:

From these testing results, it can be seen that the three optimization algorithms were in most cases able to find a global minimum. The algorithms were therefore found to be sufficient enough to be used in the research

APPENDIX C

SPEED PROFILE OPTIMIZATION RESULTS

C.1 TRL Route

The MATLAB® built in GA, developed DE, and PSO algorithms were used to search for optimal speed profile for TRL route. The optimization results for each section (interstation distance) of the route are presented in this section.

C.1.1 TRL Route; S1-S2

Speed profile optimization results for TRL route, from S1 to S2 were as presented in Figure C.1 through Figure C.7.

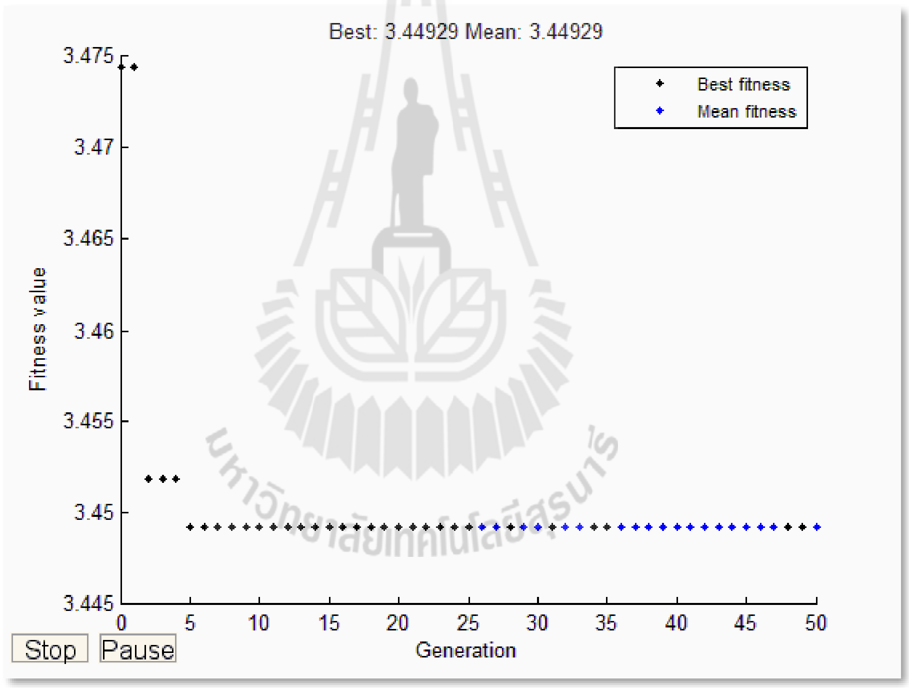


Figure C.1 TRL route, S1 to S2 GA fitness value plot

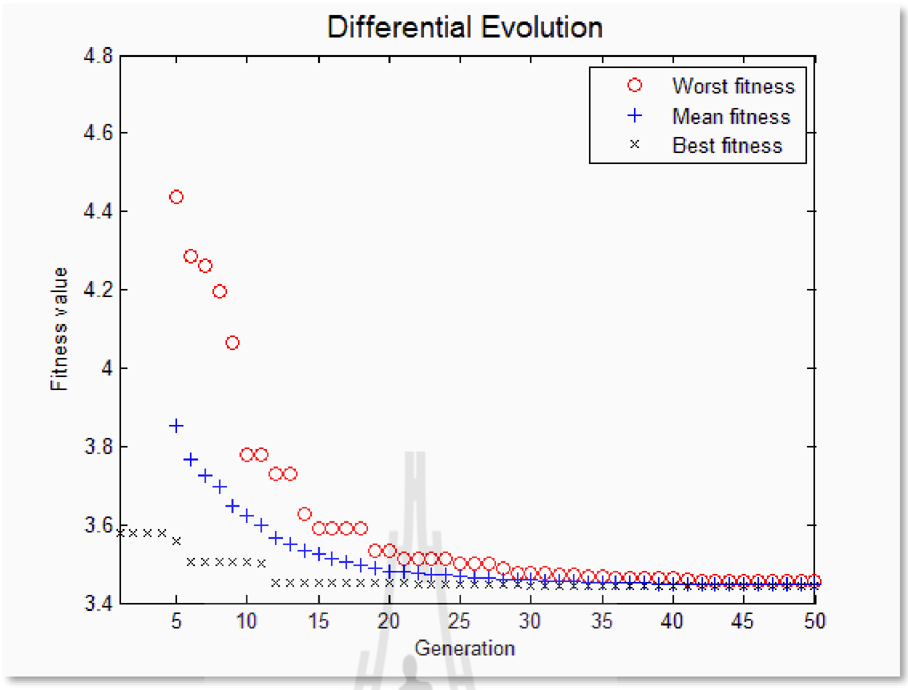


Figure C.2 TRL route, S1 to S2 DE fitness value plot

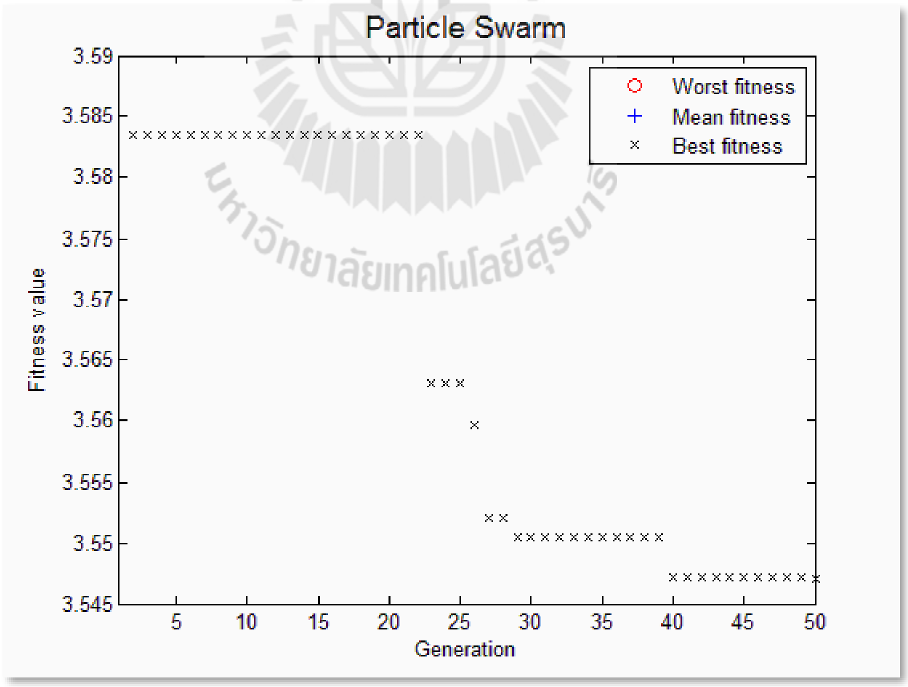
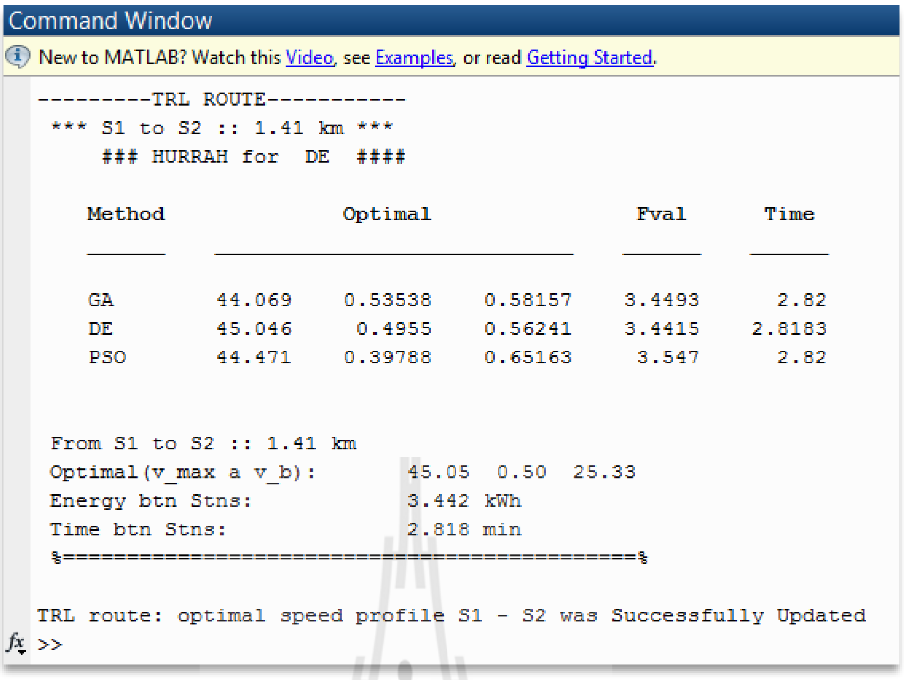


Figure C.3 TRL route, S1 to S2 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
---------TRL ROUTE-----------
 *** S1 to S2 :: 1.41 km ***
     ### HURRAH for  DE  ####

    Method              Optimal                   Fval      Time
    ______    ____________________________      ______    ______

    GA        44.069     0.53538     0.58157    3.4493      2.82
    DE        45.046      0.4955     0.56241    3.4415    2.8183
    PSO       44.471     0.39788     0.65163     3.547      2.82


 From S1 to S2 :: 1.41 km
 Optimal(v_max a v_b):        45.05  0.50  25.33
 Energy btn Stns:             3.442 kWh
 Time btn Stns:               2.818 min
 %==============================================%

TRL route: optimal speed profile S1 - S2 was Successfully Updated
>>
```

Figure C.4 TRL route, S1 to S2 GA, DE and PSO optimal values

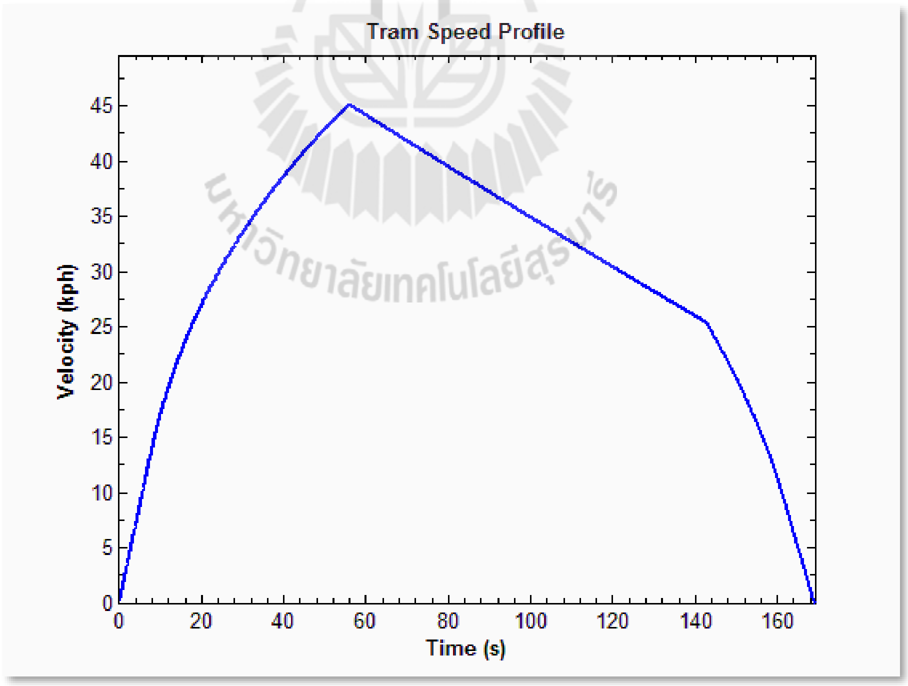


Figure C.5 TRL route, S1 to S2 the best speed profile, velocity against time

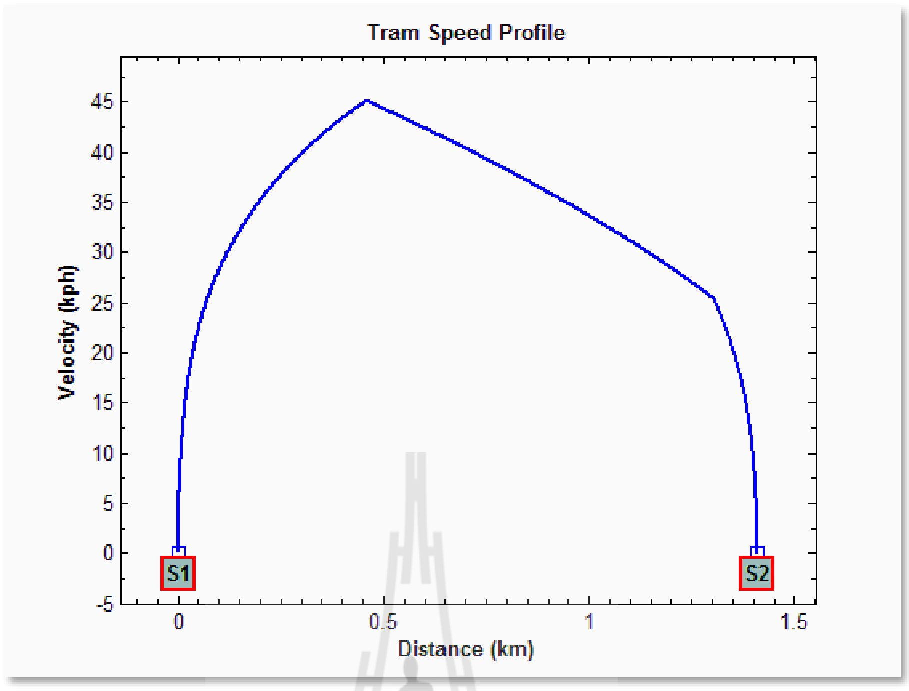


Figure C.6 TRL route, S1 to S2 the best speed profile, velocity against distance

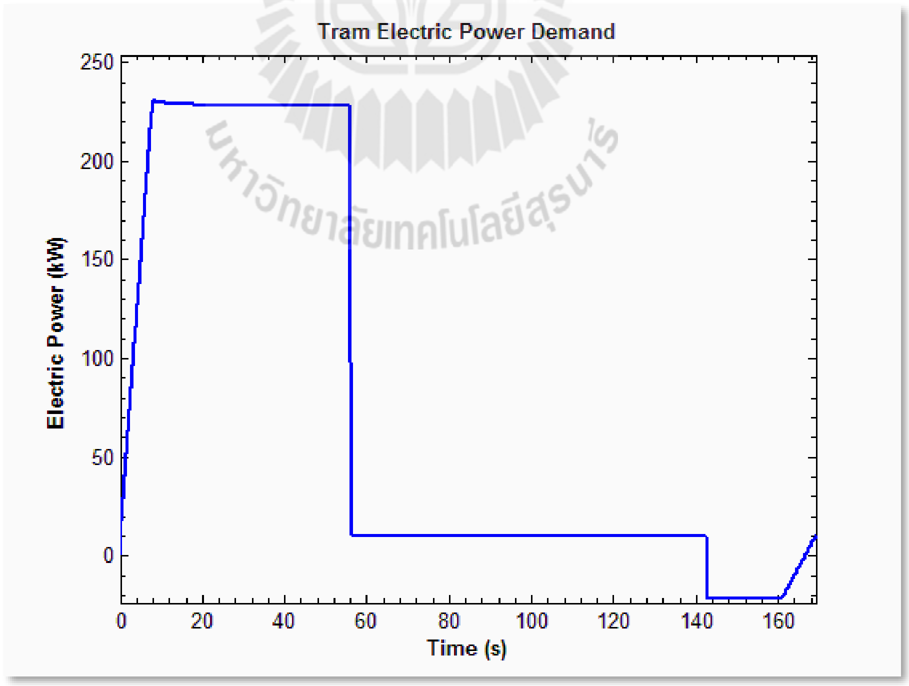


Figure C.7 TRL route, S1 to S2 power demand corresponding to the best speed profile

C.1.2 TRL Route; S2-S3

Speed profile optimization results for TRL route, from S2 to S3 were as presented in Figure C.8 through Figure C.14.

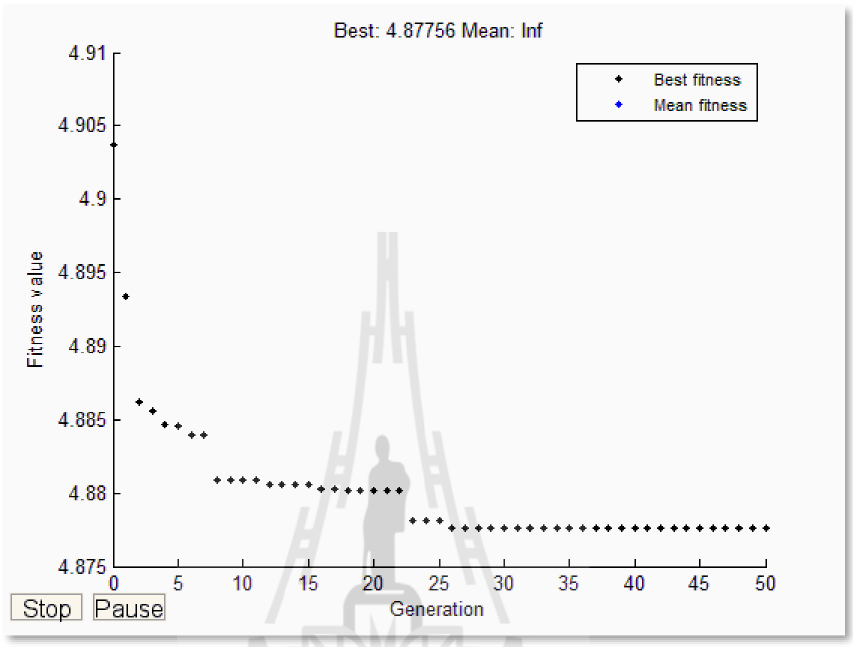


Figure C.8 TRL route, S2 to S3 GA fitness value plot

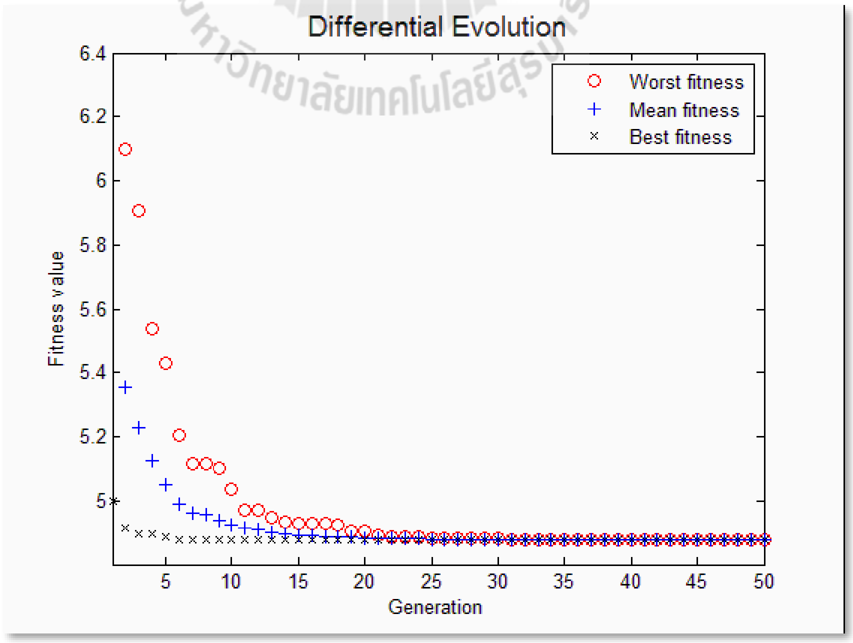


Figure C.9 TRL route, S2 to S3 DE fitness value plot

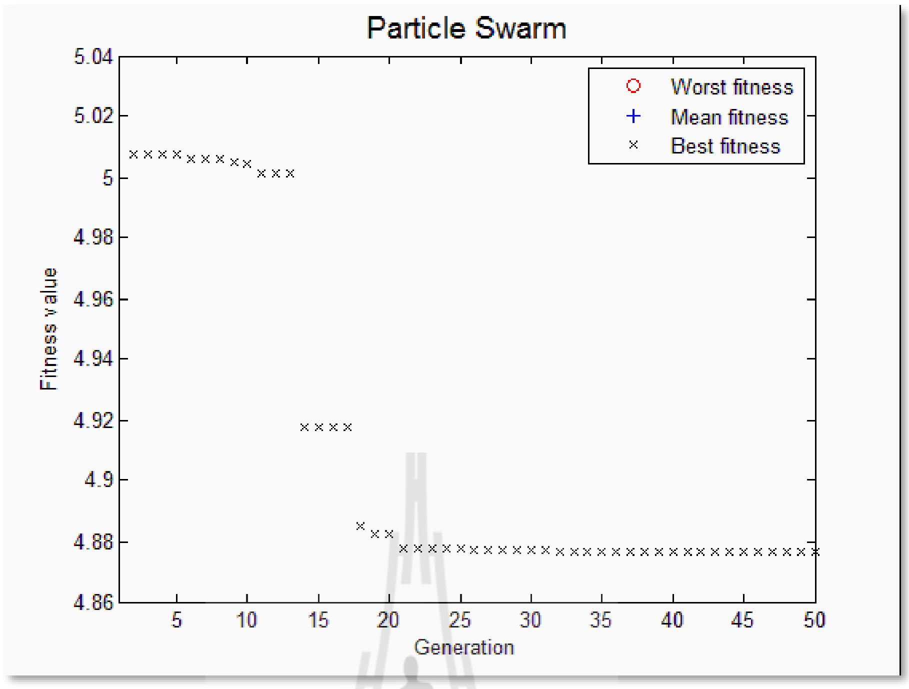
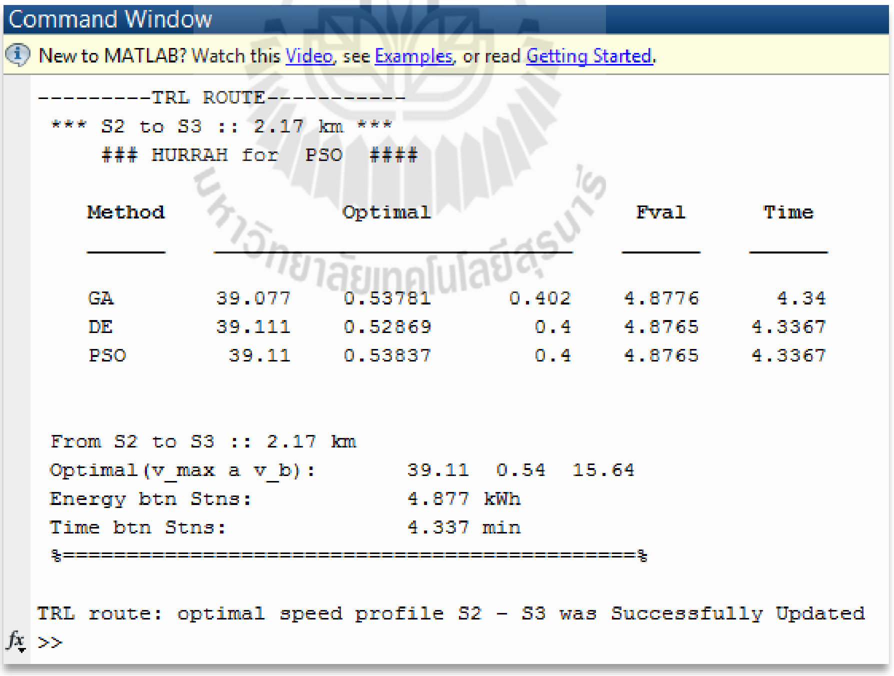


Figure C.10 TRL route, S2 to S3 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
---------TRL ROUTE-----------
 *** S2 to S3 :: 2.17 km ***
     ### HURRAH for  PSO  ####

    Method          Optimal                   Fval      Time
    ______    ____________________________    ______    ______

    GA        39.077    0.53781     0.402    4.8776      4.34
    DE        39.111    0.52869       0.4    4.8765    4.3367
    PSO        39.11    0.53837       0.4    4.8765    4.3367


 From S2 to S3 :: 2.17 km
 Optimal(v_max a v_b):         39.11  0.54  15.64
 Energy btn Stns:              4.877 kWh
 Time btn Stns:                4.337 min
 %==============================================%

TRL route: optimal speed profile S2 - S3 was Successfully Updated
>>
```

Figure C.11 TRL route, S2 to S3 GA, DE and PSO optimal values

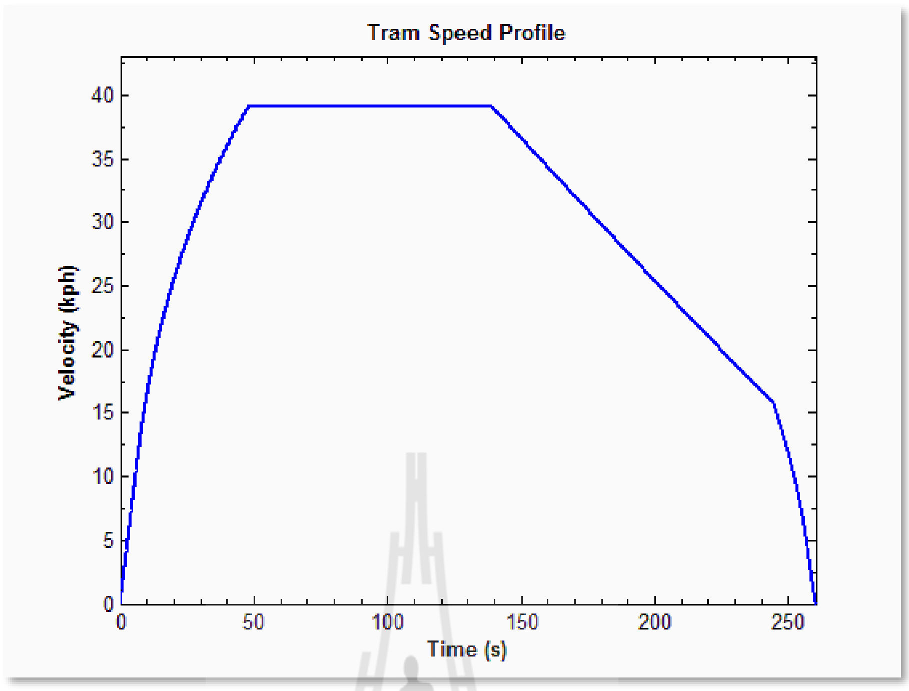


Figure C.12 TRL route, S2 to S3 the best speed profile, velocity against time

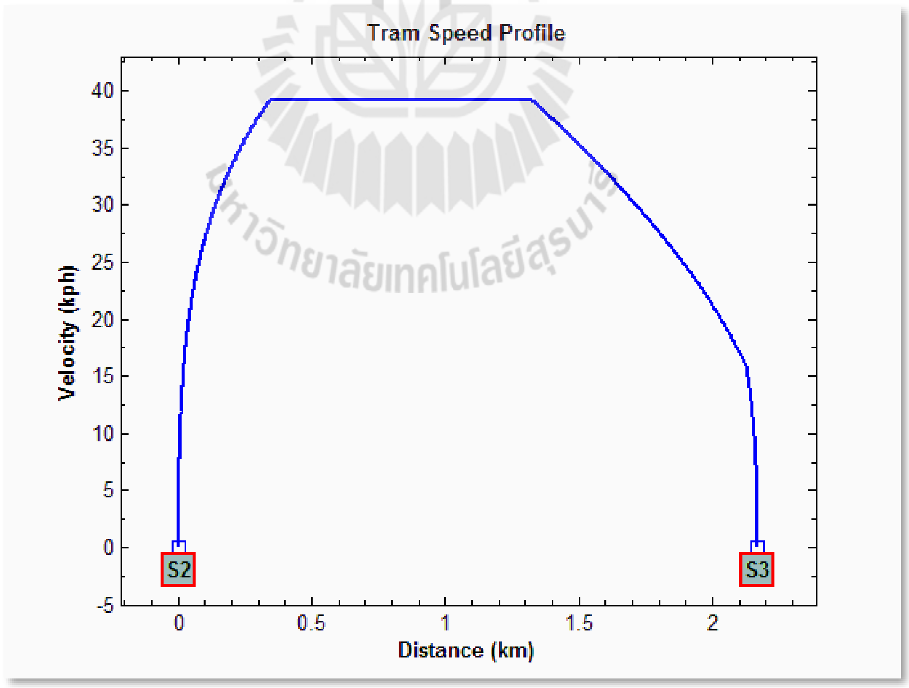


Figure C.13 TRL route, S2 to S3 the best speed profile, velocity against distance

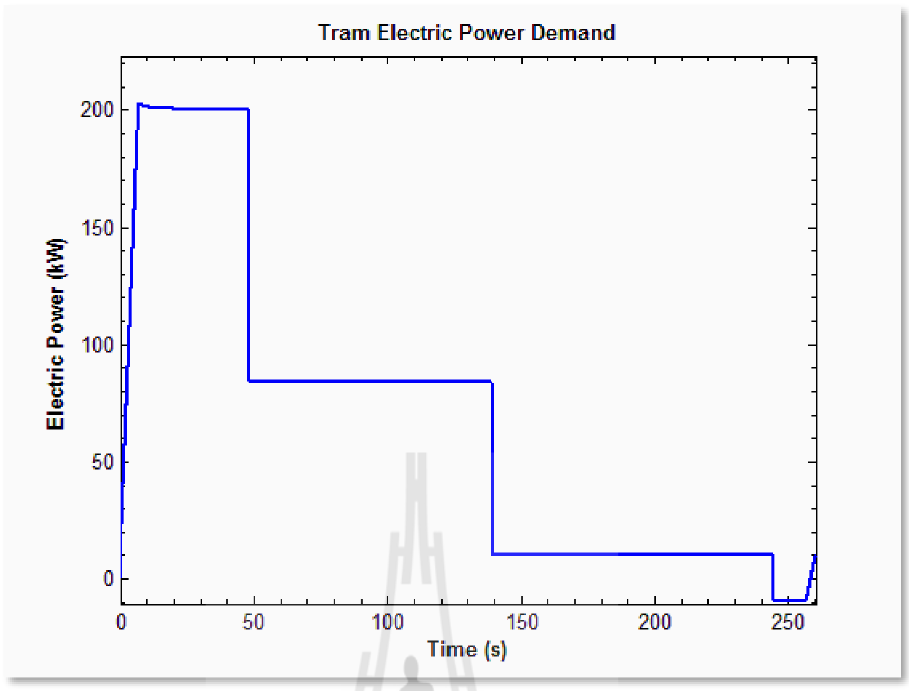


Figure C.14 TRL route, S2 to S3 power demand corresponding to the best speed profile

C.1.3 TRL Route; S3-S4

Speed profile optimization results for TRL route, from S3 to S4 were as presented in Figure C.15 through Figure C.21.

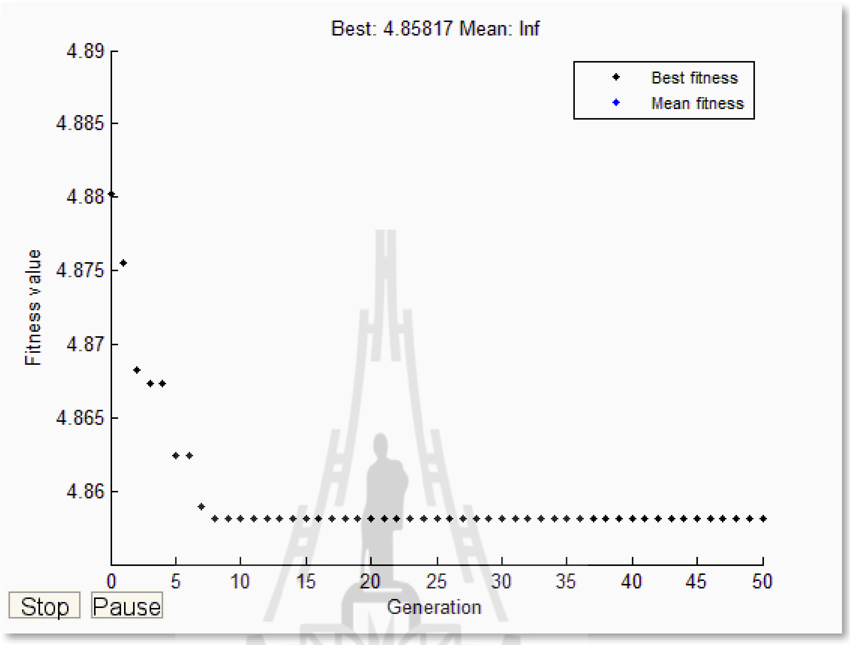


Figure C.15 TRL route, S3 to S4 GA fitness value plot

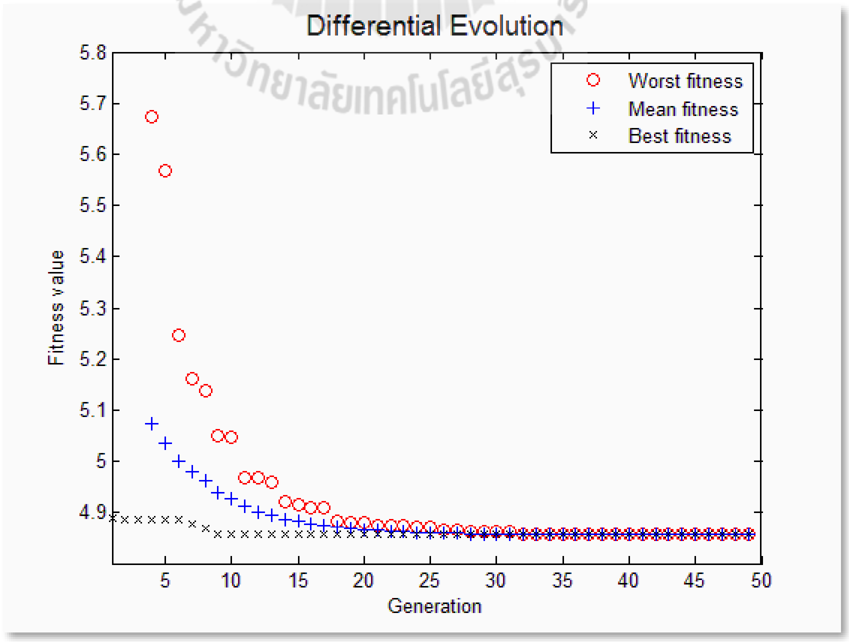


Figure C.16 TRL route, S3 to S4 DE fitness value plot

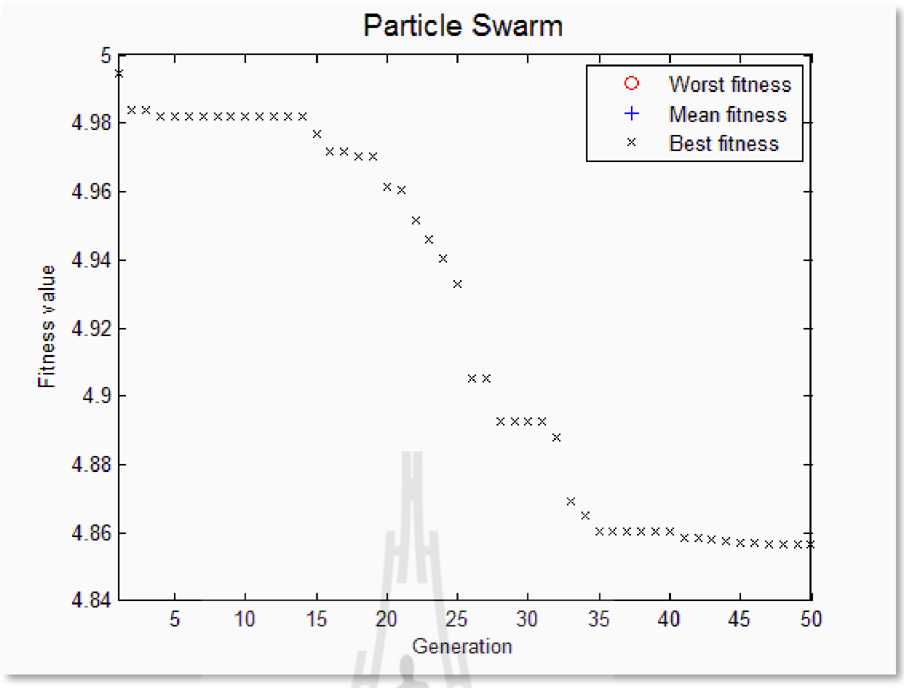
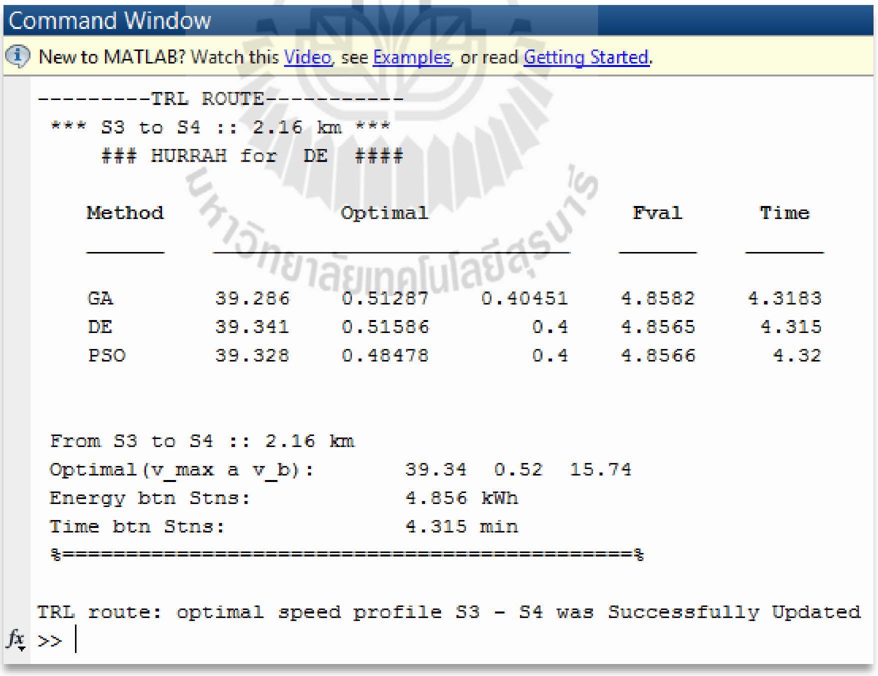


Figure C.17 TRL route, S3 to S4 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
---------TRL ROUTE-----------
 *** S3 to S4 :: 2.16 km ***
     ### HURRAH for  DE  ####

    Method          Optimal                     Fval      Time
    ______     ____________________________    ______    ______

    GA         39.286    0.51287    0.40451    4.8582    4.3183
    DE         39.341    0.51586        0.4    4.8565     4.315
    PSO        39.328    0.48478        0.4    4.8566      4.32


 From S3 to S4 :: 2.16 km
 Optimal(v_max a v_b):         39.34   0.52   15.74
 Energy btn Stns:              4.856 kWh
 Time btn Stns:                4.315 min
 %==============================================%

TRL route: optimal speed profile S3 - S4 was Successfully Updated
>>
```

Figure C.18 TRL route, S3 to S4 GA, DE and PSO optimal values

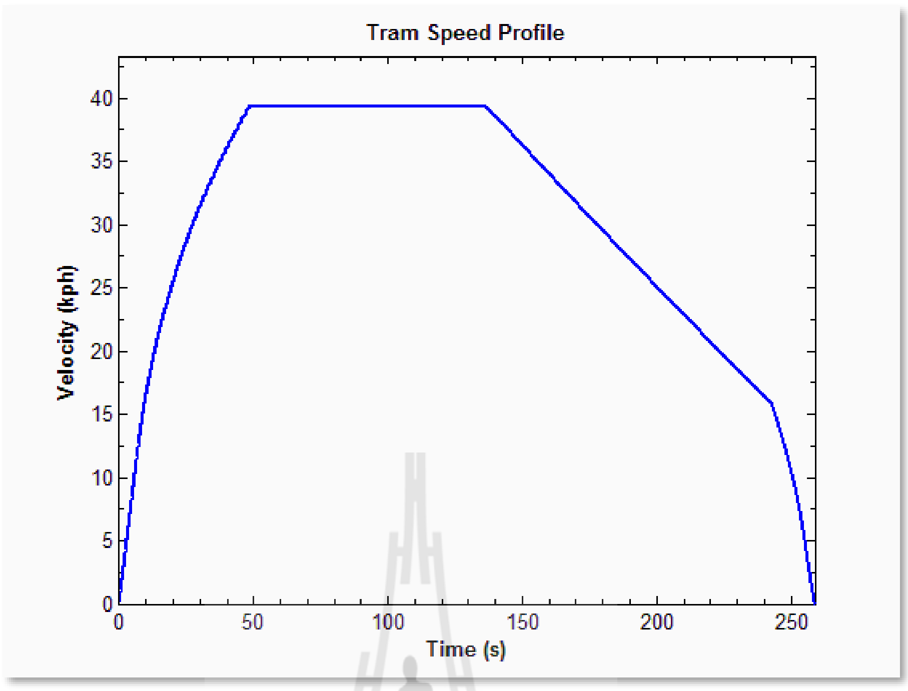


Figure C.19 TRL route, S3 to S4 the best speed profile, velocity against time

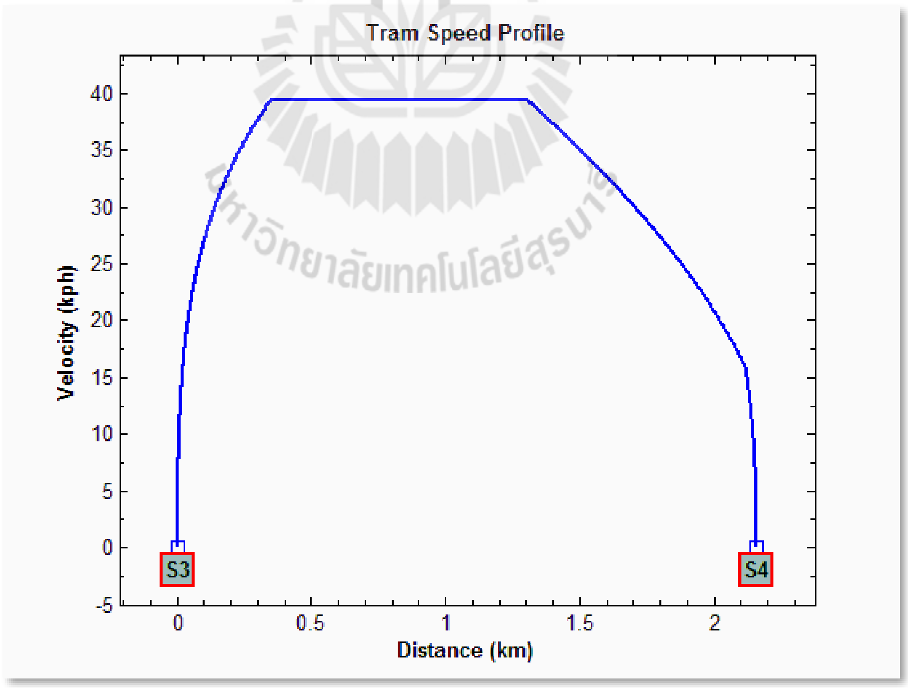


Figure C.20 TRL route, S3 to S4 the best speed profile, velocity against distance

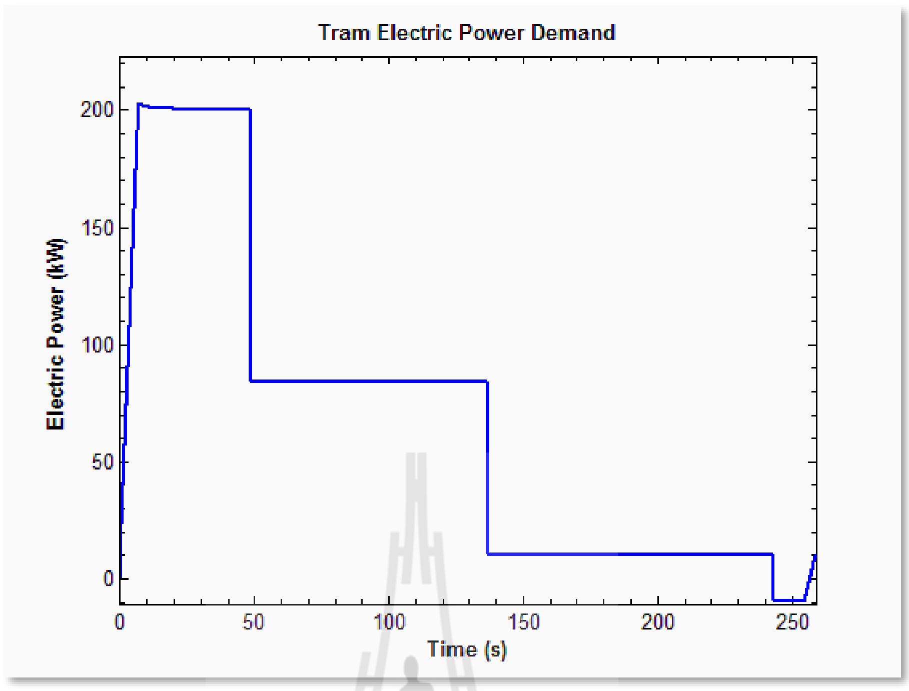


Figure C.21 TRL route, S3 to S4 power demand corresponding to the best speed profile

C.1.4 TRL Route; S4-S5

Speed profile optimization results for TRL route, from S4 to S5 were as presented in Figure C.22 through Figure C.28.

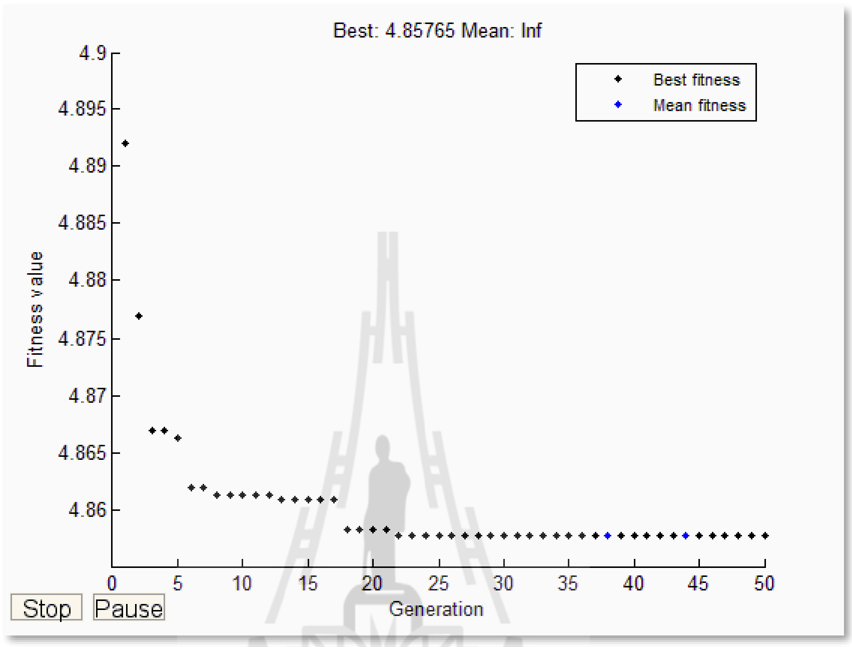


Figure C.22 TRL route, S4 to S5 GA fitness value plot

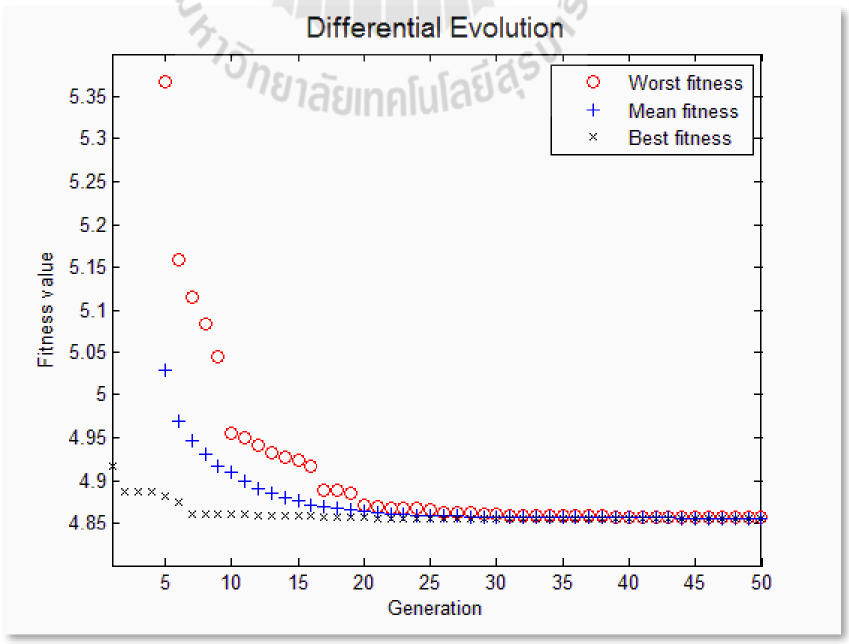


Figure C.23 TRL route, S4 to S5 DE fitness value plot

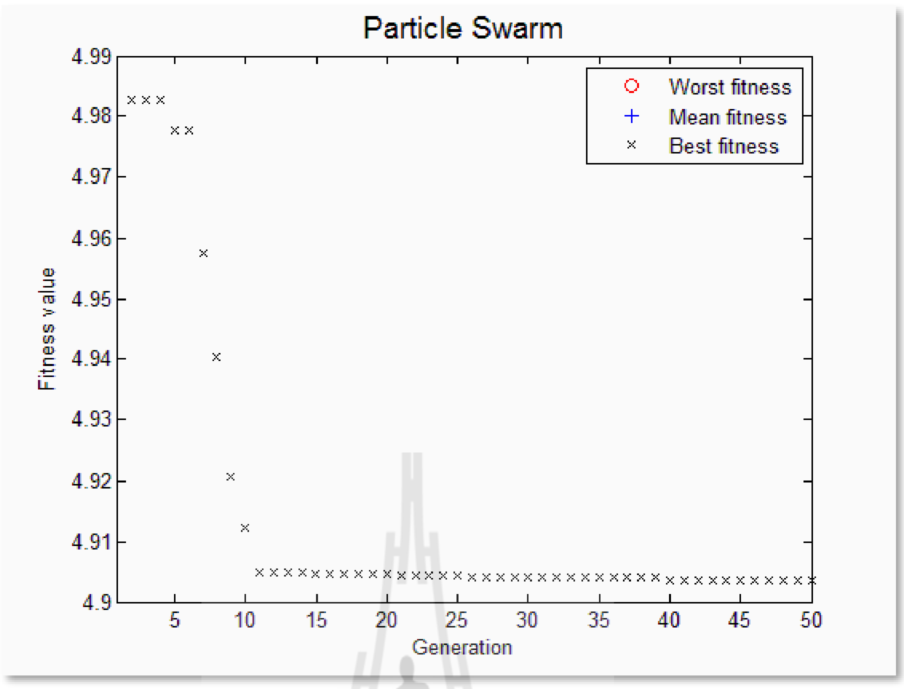
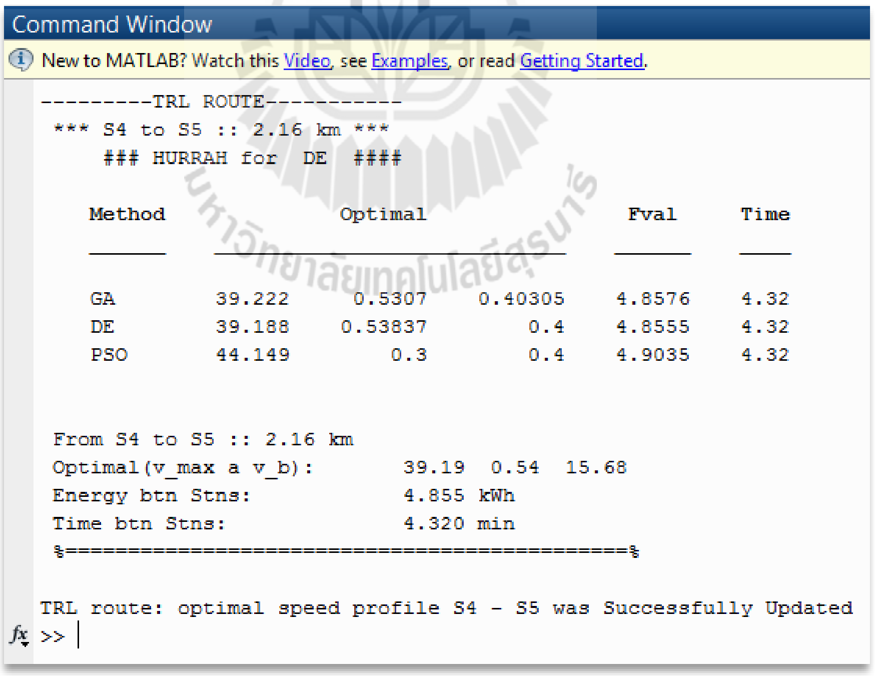


Figure C.24 TRL route, S4 to S5 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
---------TRL ROUTE-----------
 *** S4 to S5 :: 2.16 km ***
     ### HURRAH for  DE  ####

    Method              Optimal                   Fval      Time
    ______    ____________________________       ______    _____

    GA        39.222      0.5307     0.40305     4.8576     4.32
    DE        39.188     0.53837         0.4     4.8555     4.32
    PSO       44.149         0.3         0.4     4.9035     4.32


 From S4 to S5 :: 2.16 km
 Optimal(v_max a v_b):        39.19  0.54  15.68
 Energy btn Stns:             4.855 kWh
 Time btn Stns:               4.320 min
 %=============================================%

TRL route: optimal speed profile S4 - S5 was Successfully Updated
>>
```

Figure C.25 TRL route, S4 to S5 GA, DE and PSO optimal values

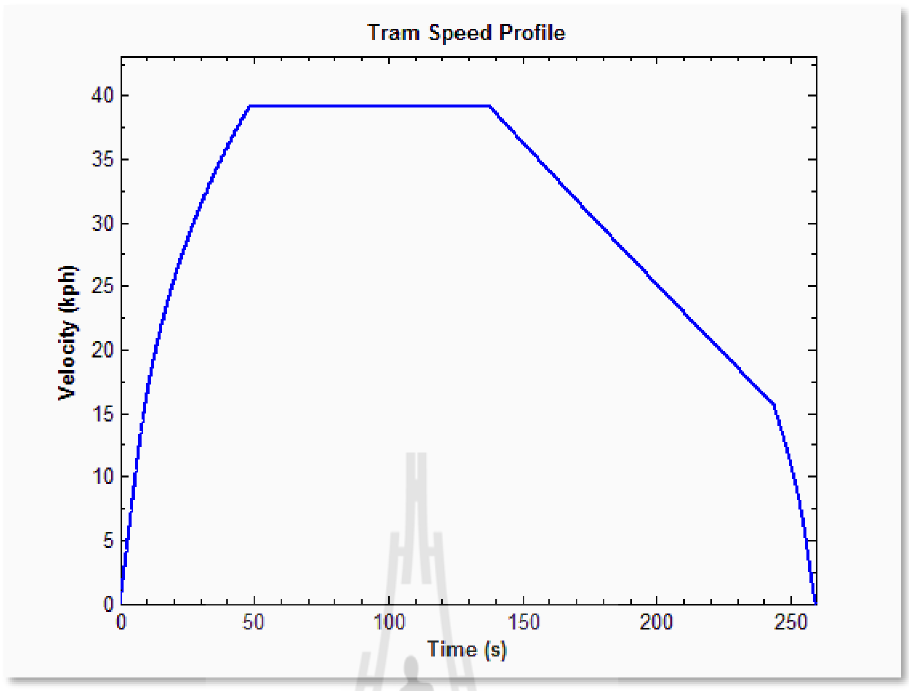


Figure C.26 TRL route, S4 to S5 the best speed profile, velocity against time

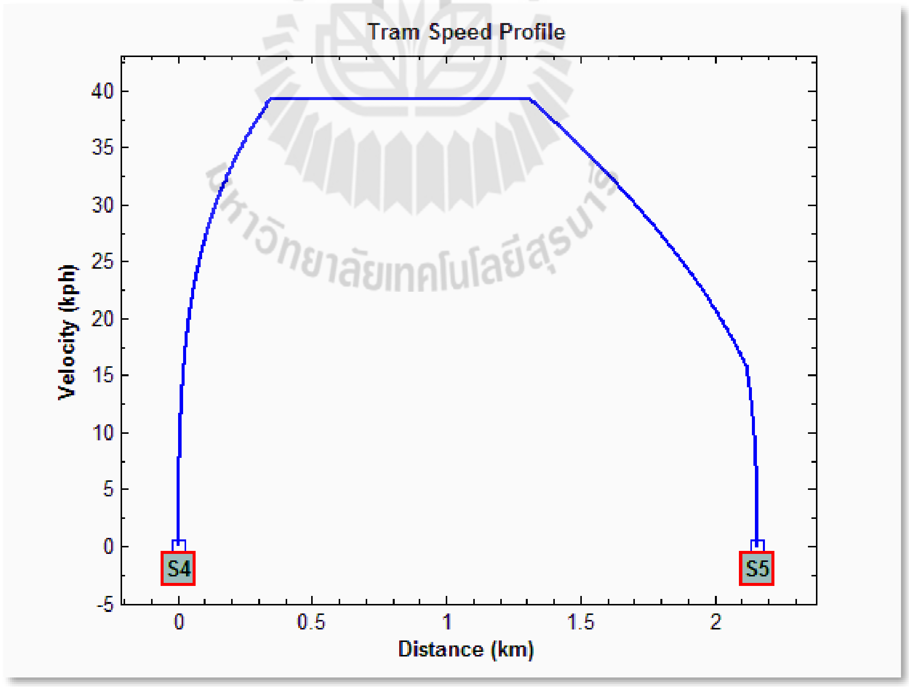


Figure C.27 TRL route, S4 to S5 the best speed profile, velocity against distance

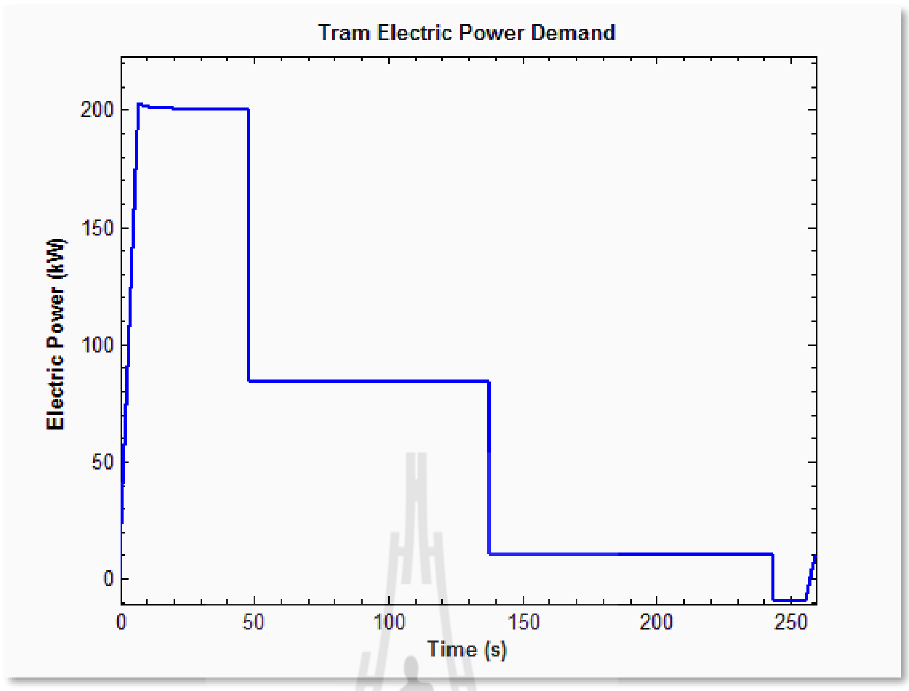


Figure C.28 TRL route, S4 to S5 power demand corresponding to the best speed profile

C.1.5 TRL Route; S5-S6

Speed profile optimization results for TRL route, from S5 to S6 were as presented in Figure C.29 through Figure C.35.

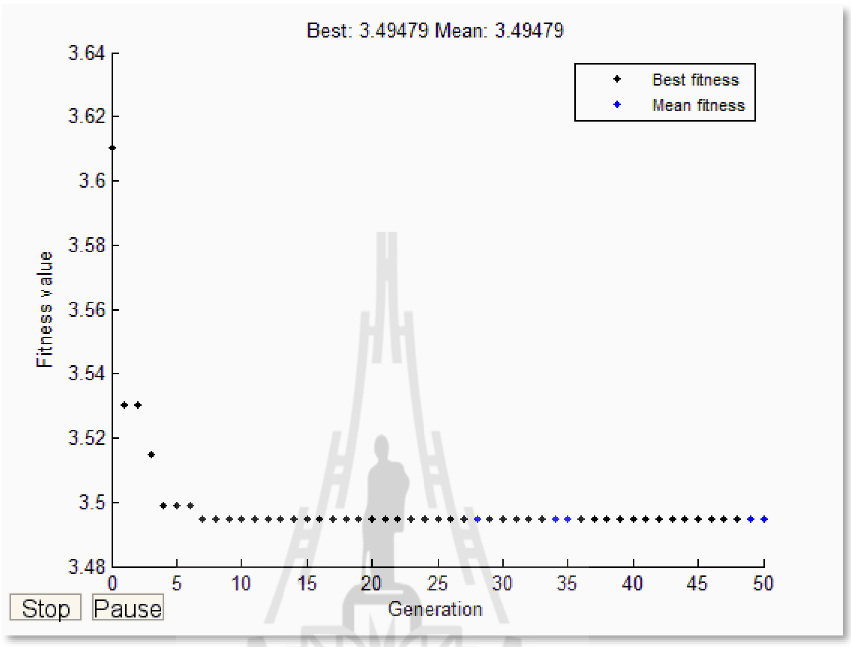


Figure C.29 TRL route, S5 to S6 GA fitness value plot

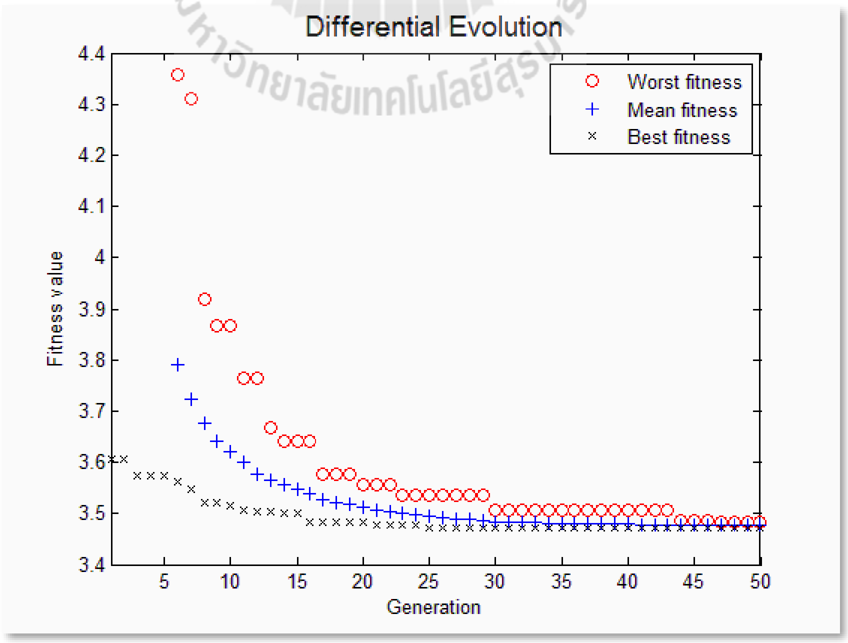


Figure C.30 TRL route, S5 to S6 DE fitness value plot

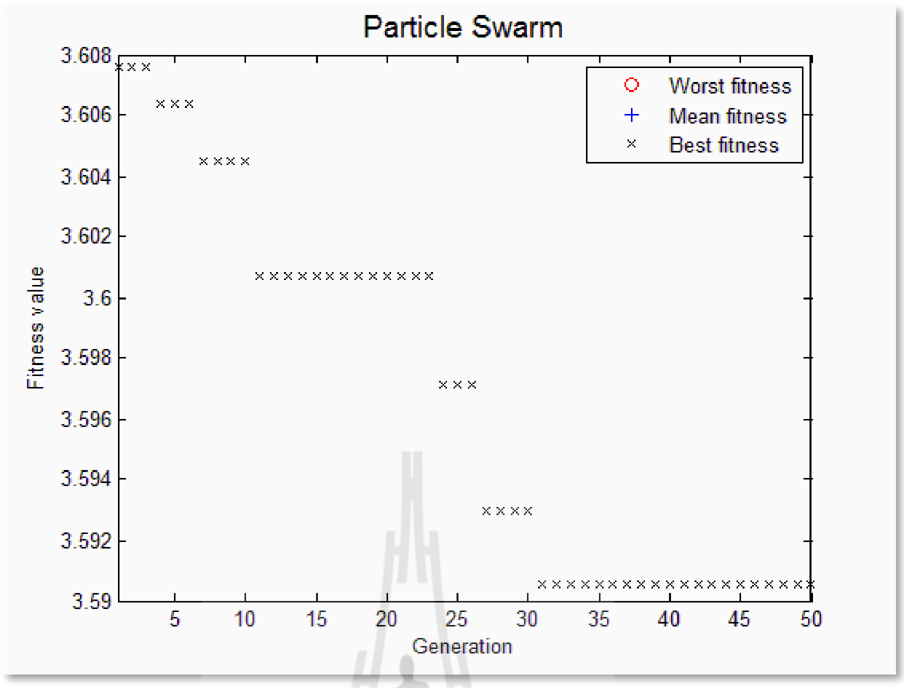
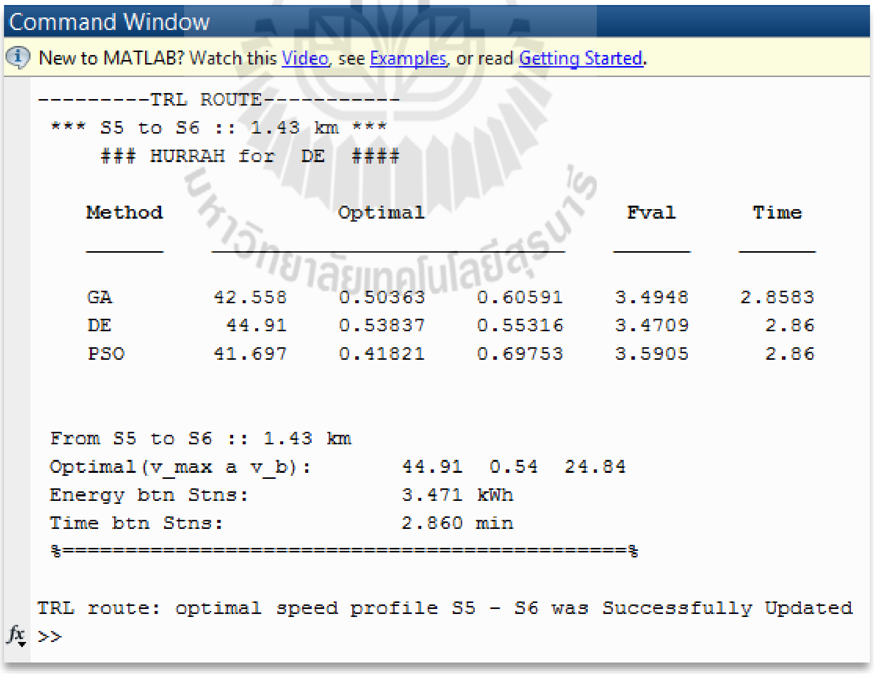


Figure C.31 TRL route, S5 to S6 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
 ---------TRL ROUTE-----------
  *** S5 to S6 :: 1.43 km ***
      ### HURRAH for  DE  ####

      Method          Optimal                       Fval       Time
      ______     ____________________________      ______     ______

      GA          42.558   0.50363   0.60591      3.4948     2.8583
      DE           44.91   0.53837   0.55316      3.4709       2.86
      PSO         41.697   0.41821   0.69753      3.5905       2.86


  From S5 to S6 :: 1.43 km
  Optimal(v_max a v_b):       44.91   0.54   24.84
  Energy btn Stns:            3.471 kWh
  Time btn Stns:              2.860 min
  %==============================================%

 TRL route: optimal speed profile S5 - S6 was Successfully Updated
fx >>
```

Figure C.32 TRL route, S5 to S6 GA, DE and PSO optimal values

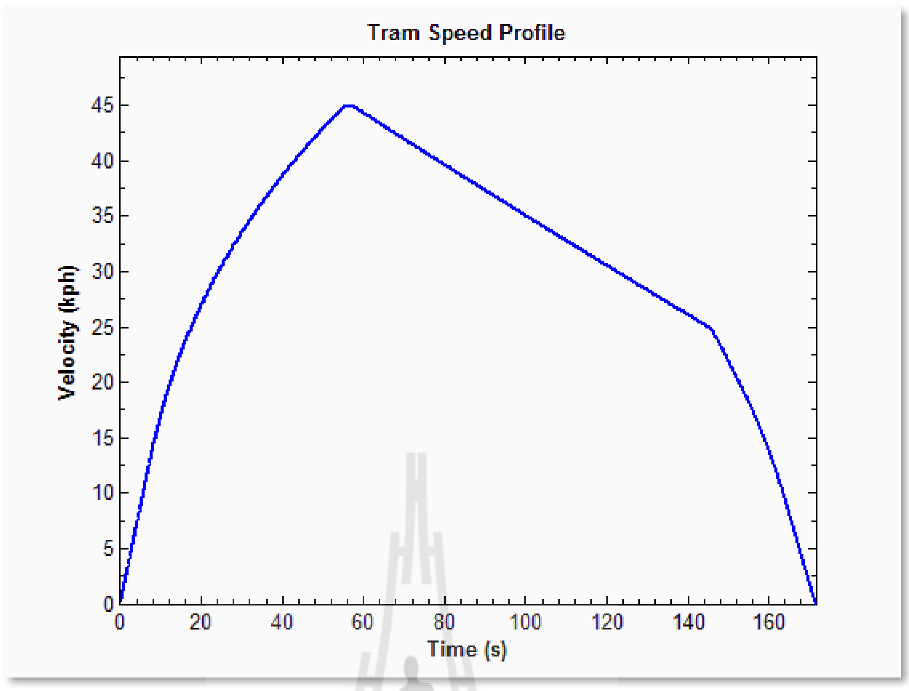


Figure C.33 TRL route, S5 to S6 the best speed profile, velocity against time

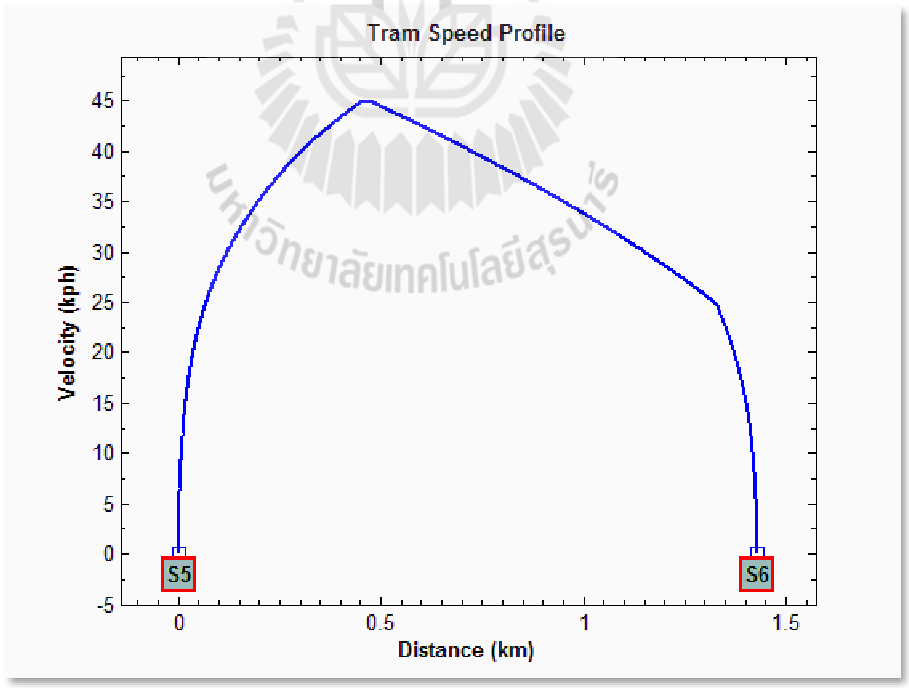


Figure C.34 TRL route, S5 to S6 the best speed profile, velocity against distance

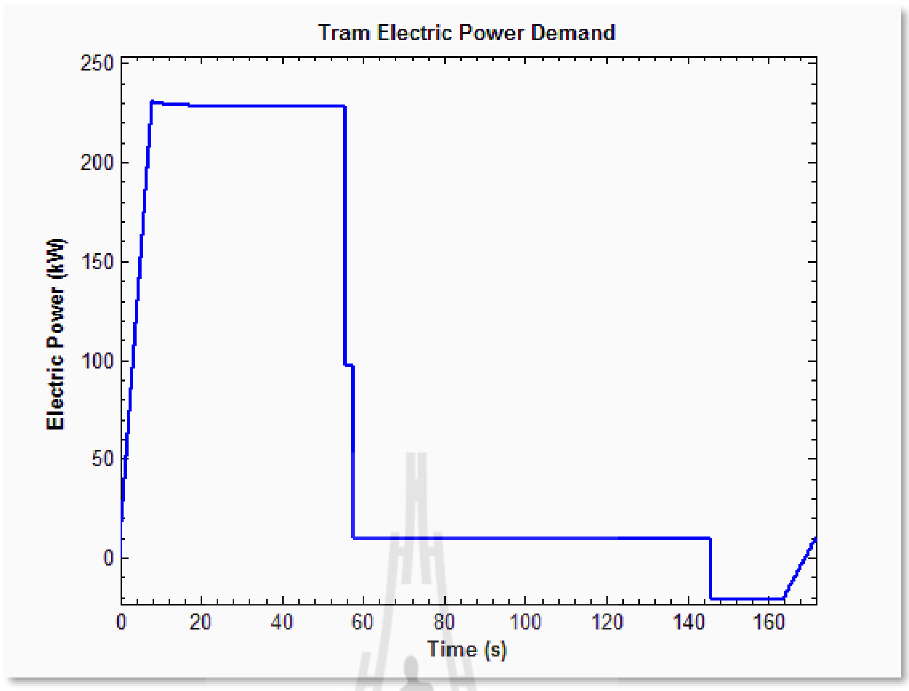


Figure C.35 TRL route, S4 to S5 power demand corresponding to the best speed profile

C.1.6 TRL Route; S6-S7

Speed profile optimization results for TRL route, from S6 to S7 were as presented in Figure C.36 through Figure C.42.

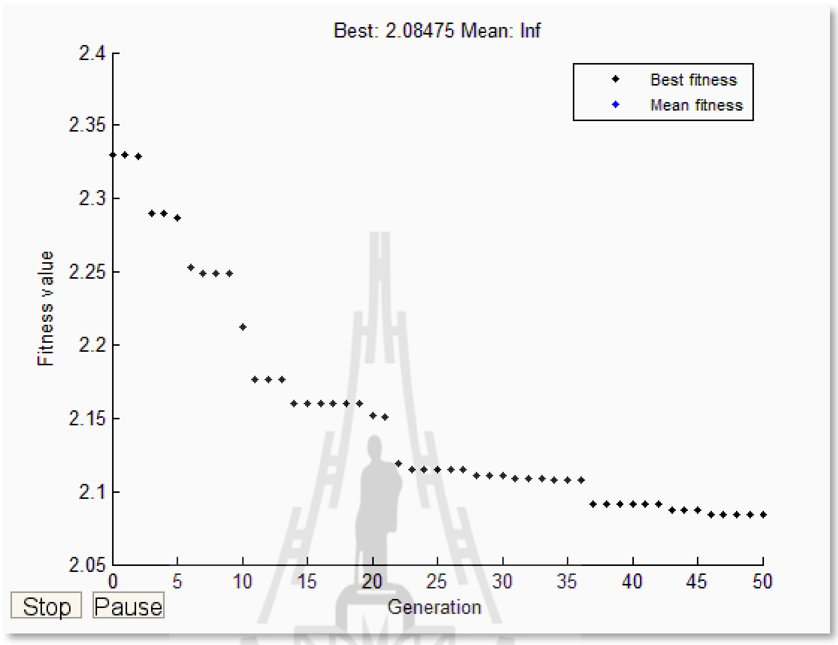


Figure C.36 TRL route, S6 to S7 GA fitness value plot

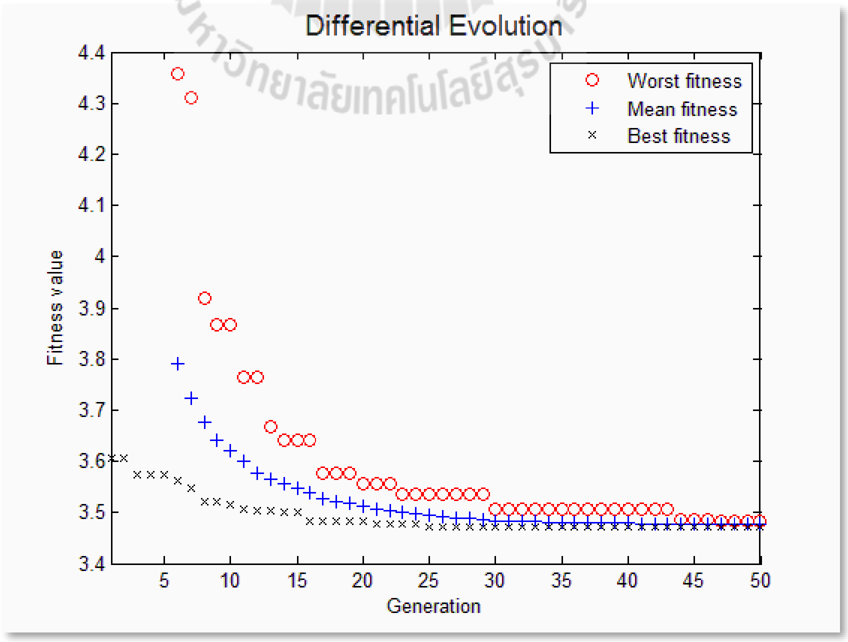


Figure C.37 TRL route, S6 to S7 DE fitness value plot

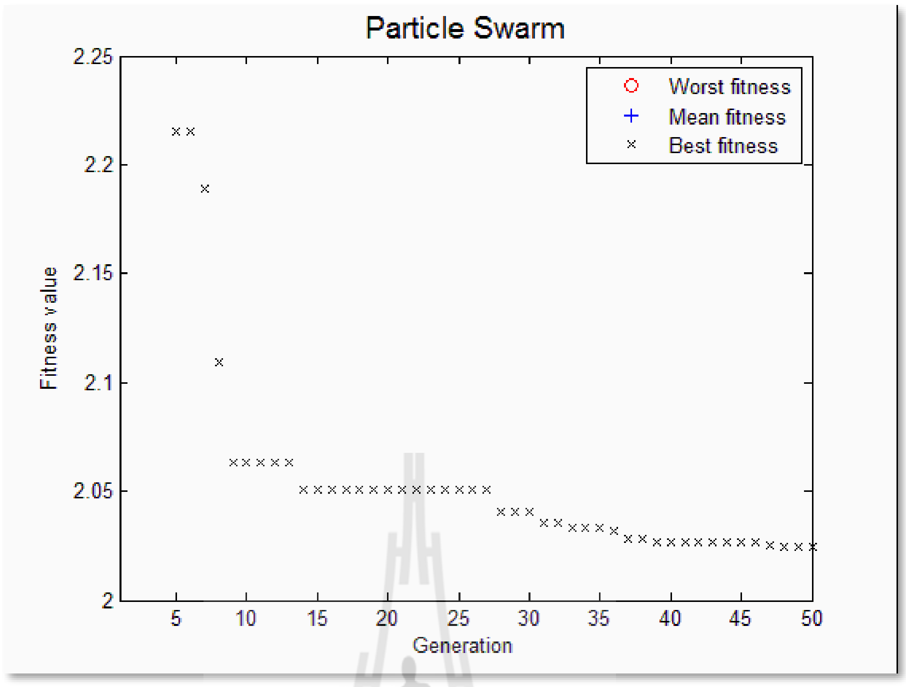
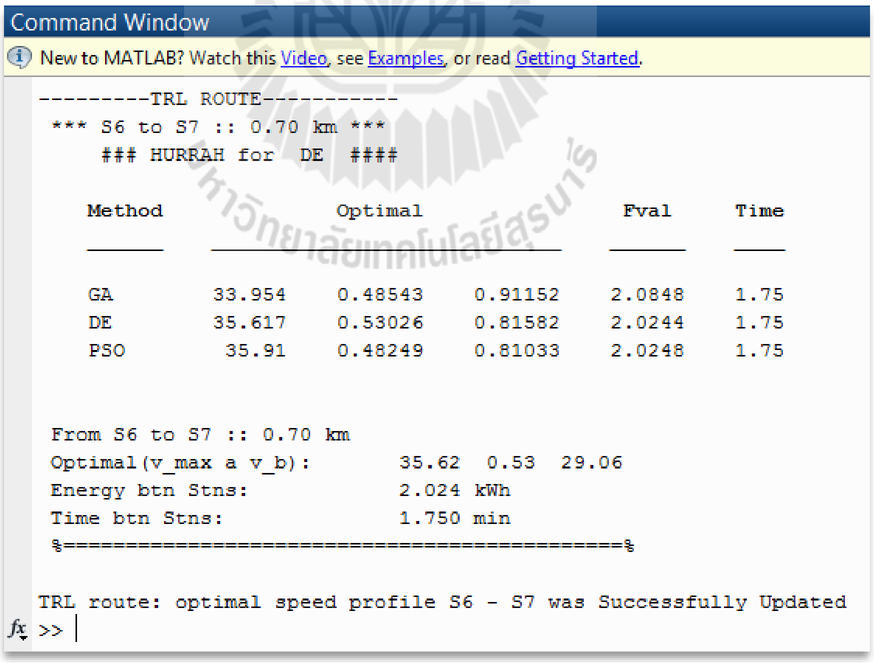


Figure C.38 TRL route, S6 to S7 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
---------TRL ROUTE-----------
 *** S6 to S7 :: 0.70 km ***
     ### HURRAH for  DE  ####

    Method           Optimal                      Fval      Time
    ______     ____________________________      ______    ____

    GA         33.954     0.48543    0.91152    2.0848    1.75
    DE         35.617     0.53026    0.81582    2.0244    1.75
    PSO         35.91     0.48249    0.81033    2.0248    1.75


 From S6 to S7 :: 0.70 km
 Optimal(v_max a v_b):         35.62   0.53   29.06
 Energy btn Stns:              2.024 kWh
 Time btn Stns:                1.750 min
 %=============================================%

TRL route: optimal speed profile S6 - S7 was Successfully Updated
>>
```

Figure C.39 TRL route, S6 to S7 GA, DE and PSO optimal values

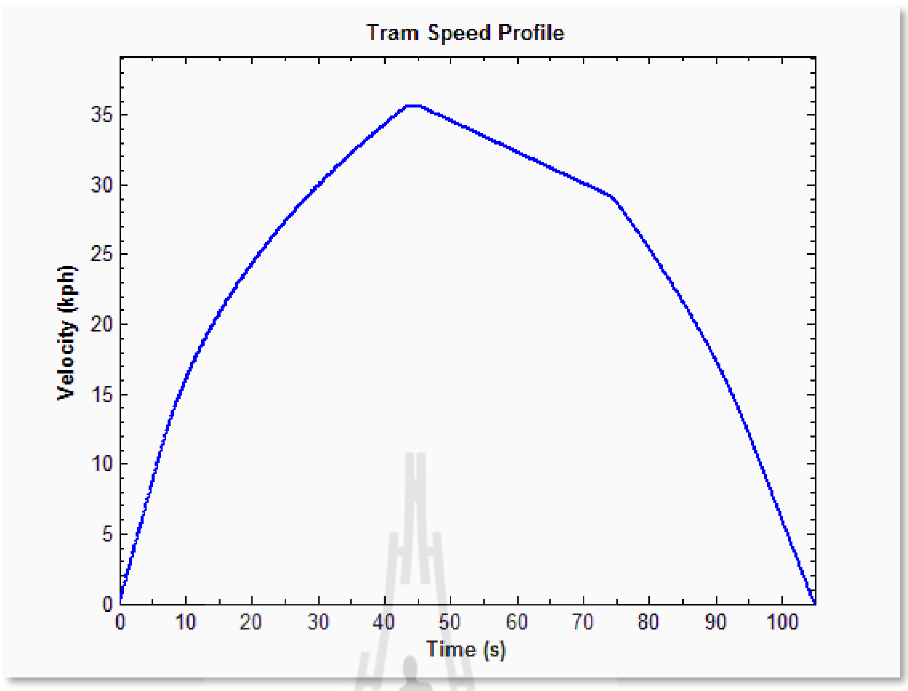


Figure C.40 TRL route, S6 to S7 the best speed profile, velocity against time

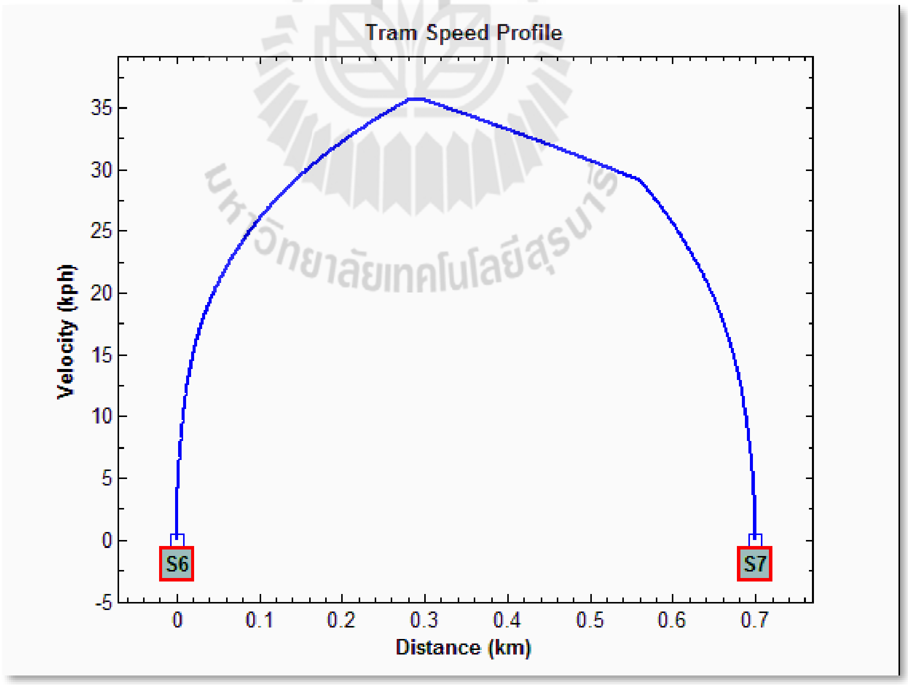


Figure C.41 TRL route, S6 to S7 the best speed profile, velocity against distance

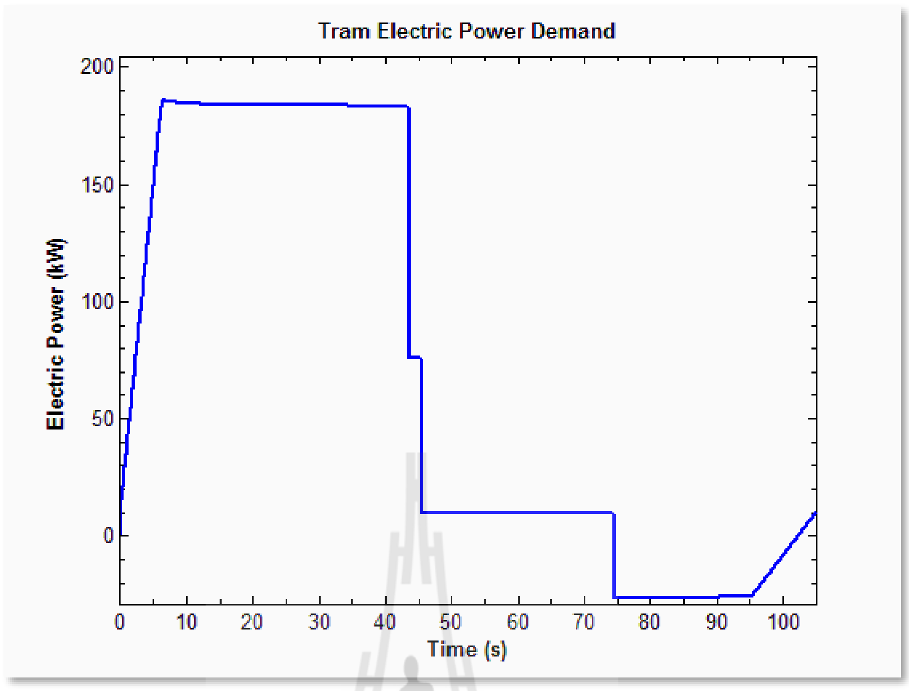


Figure C.42 TRL route, S6 to S7 power demand corresponding to the best speed profile

C.1.7 TRL Route; S7-S8

Speed profile optimization results for TRL route, from S7 to S8 were as presented in Figure C.43 through Figure C.49.

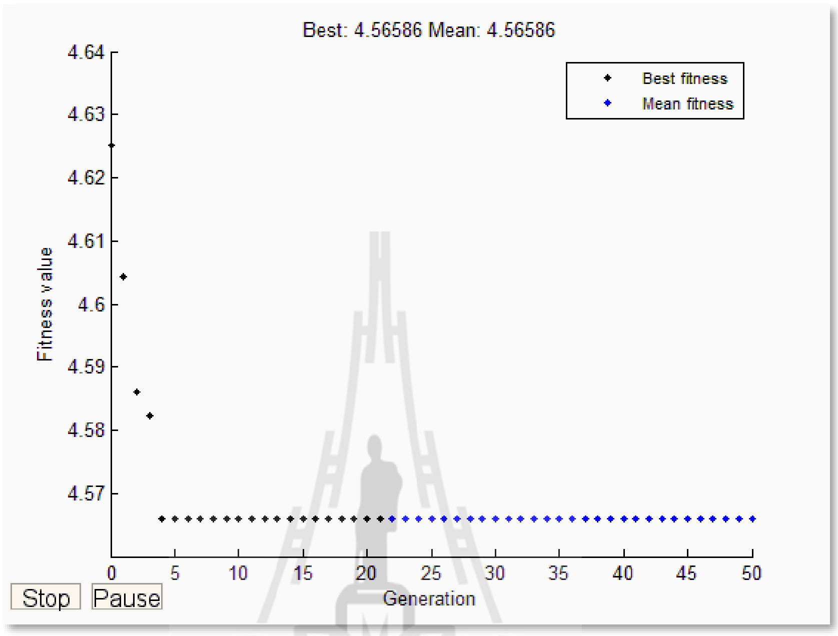


Figure C.43 TRL route, S7 to S8 GA fitness value plot

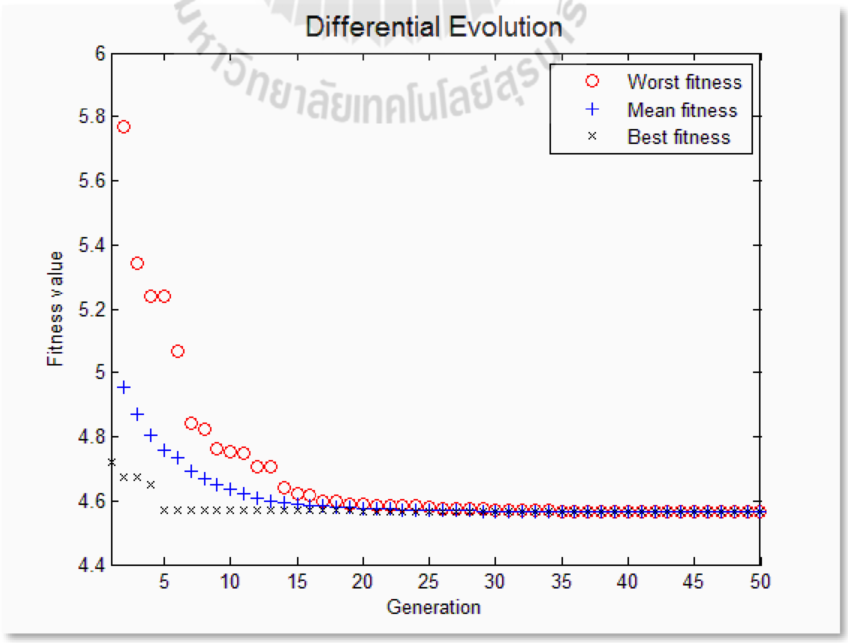


Figure C.44 TRL route, S7 to S8 DE fitness value plot

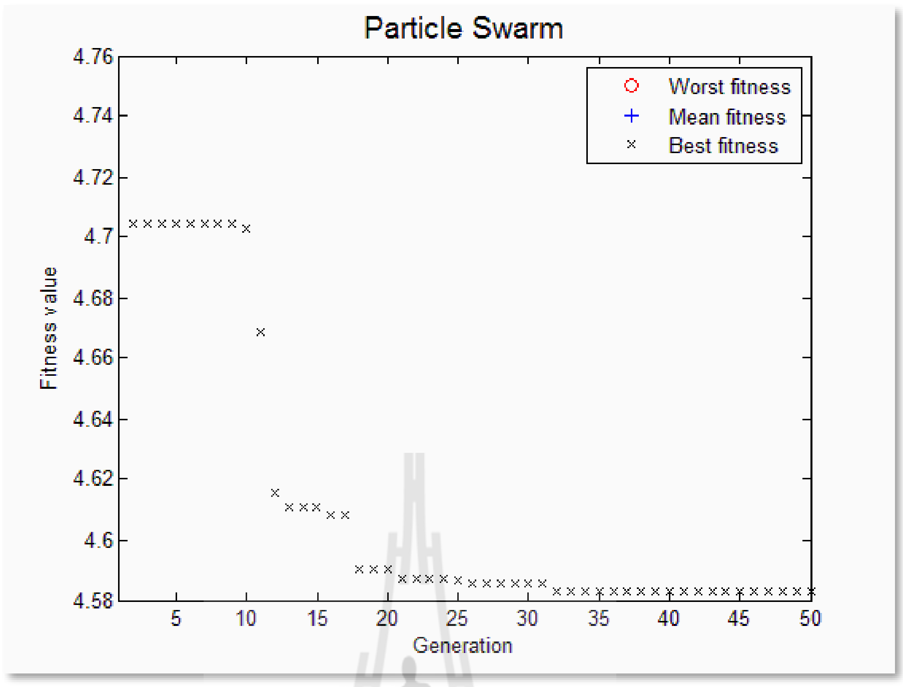
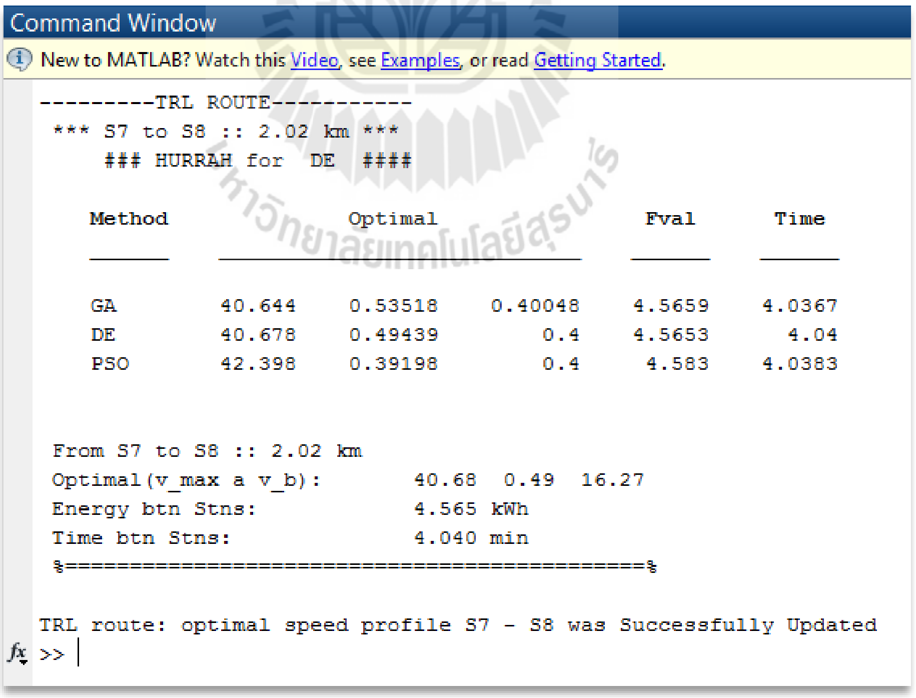


Figure C.45 TRL route, S7 to S8 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
---------TRL ROUTE-----------
 *** S7 to S8 :: 2.02 km ***
     ### HURRAH for  DE  ####

    Method          Optimal                   Fval       Time
    ______    ____________________________  ______     ______

    GA        40.644    0.53518   0.40048   4.5659     4.0367
    DE        40.678    0.49439       0.4   4.5653       4.04
    PSO       42.398    0.39198       0.4    4.583     4.0383


 From S7 to S8 :: 2.02 km
 Optimal(v_max a v_b):        40.68   0.49   16.27
 Energy btn Stns:             4.565 kWh
 Time btn Stns:               4.040 min
 %==============================================%

TRL route: optimal speed profile S7 - S8 was Successfully Updated
>> 
```

Figure C.46 TRL route, S7 to S8 GA, DE and PSO optimal values

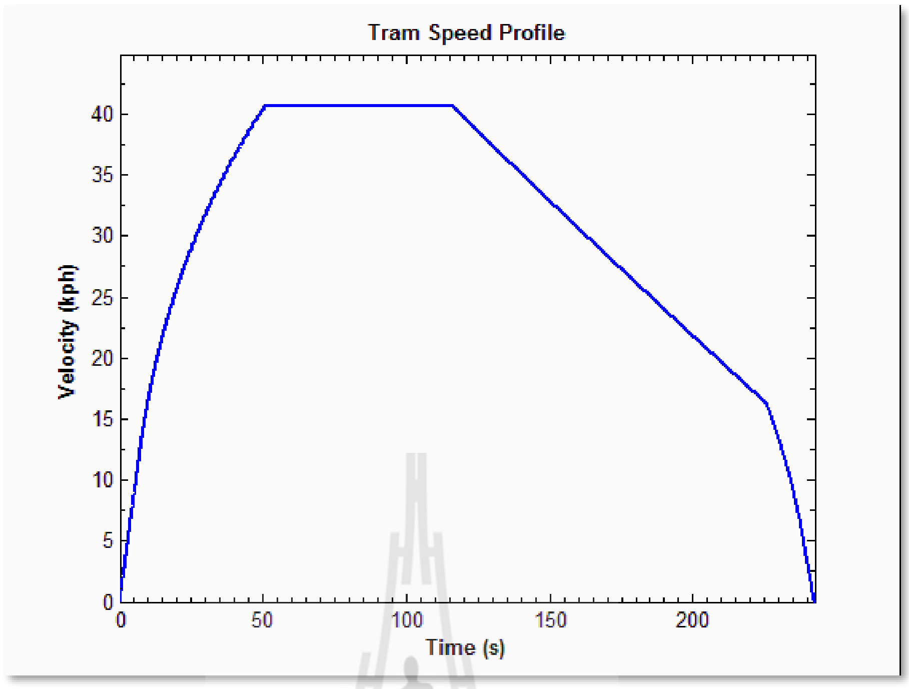


Figure C.47 TRL route, S7 to S8 the best speed profile, velocity against time

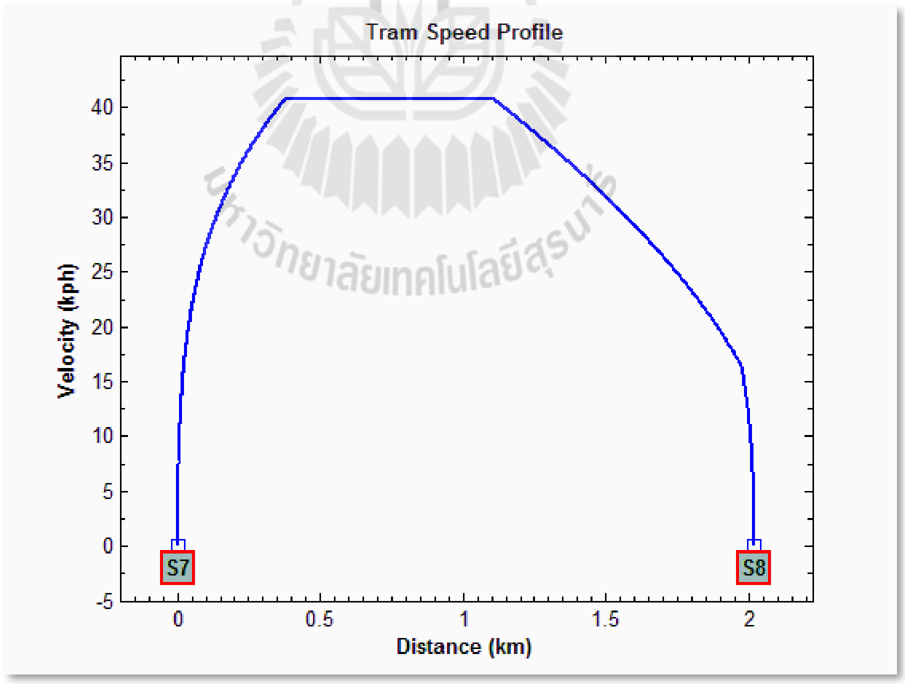


Figure C.48 TRL route, S7 to S8 the best speed profile, velocity against distance

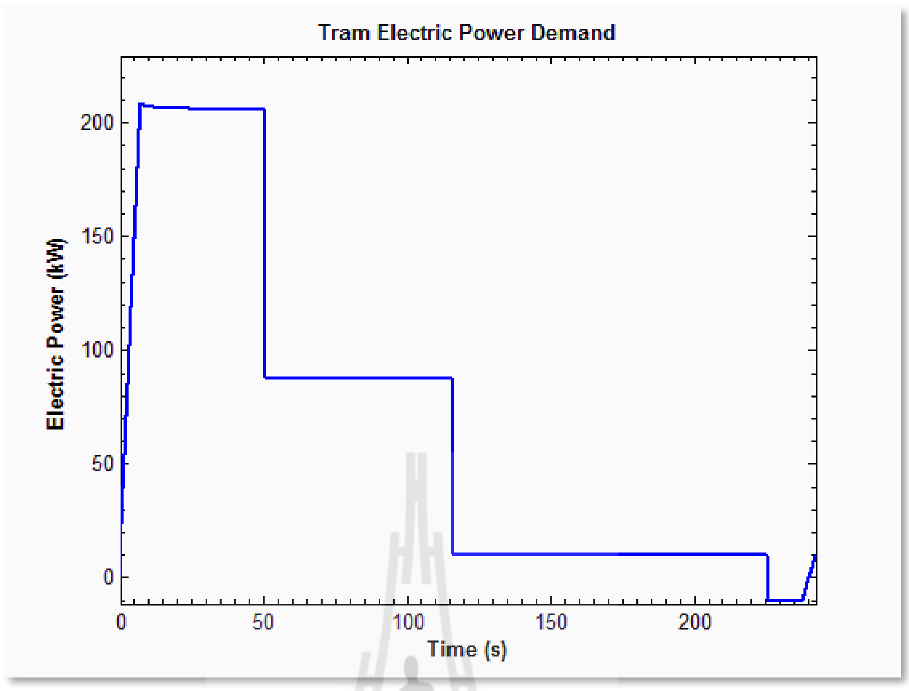


Figure C.49 TRL route, S7 to S8 power demand corresponding to the best speed profile

C.2 TAZARA Route

The MATLAB® built in GA, developed DE, and PSO algorithms were used to search for optimal speed profile for TAZARA route. The optimization results for each section (interstation distance) of the route are presented in this section.

C.2.1 TAZARA Route; S1-S2

Speed profile optimization results for TAZARA route, from S1 to S2 were as presented in Figure C.50 through Figure C.57.

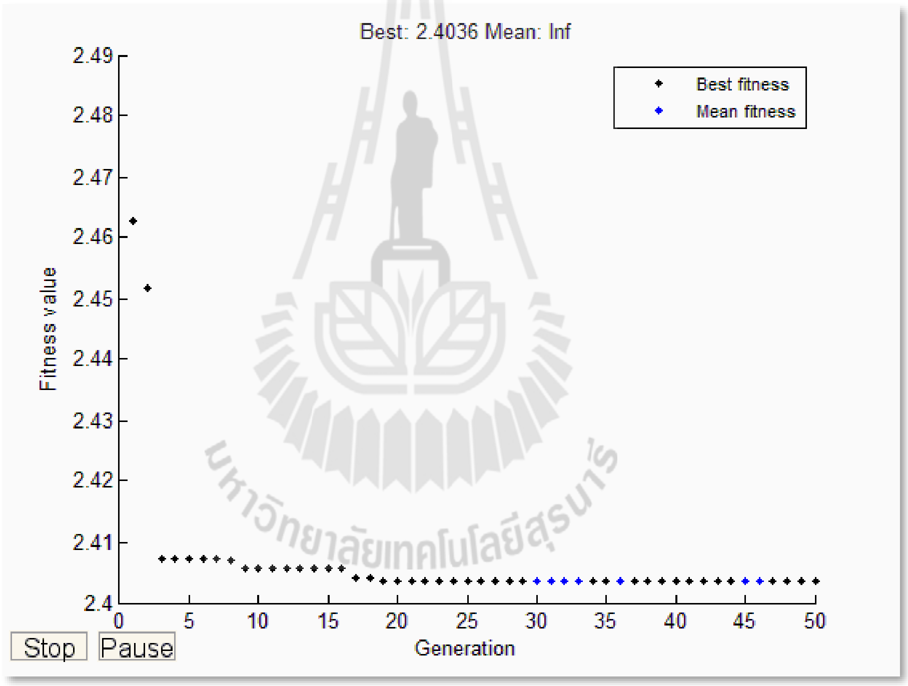


Figure C.50 TAZARA route, S1 to S2 GA fitness value plot

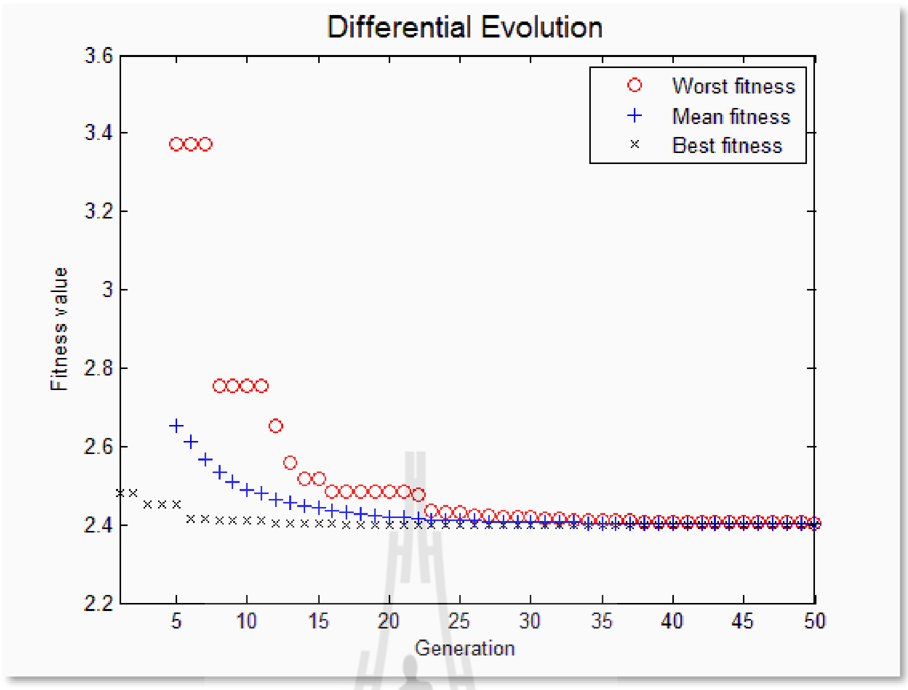


Figure C.51 TAZARA route, S1 to S2 DE fitness value plot

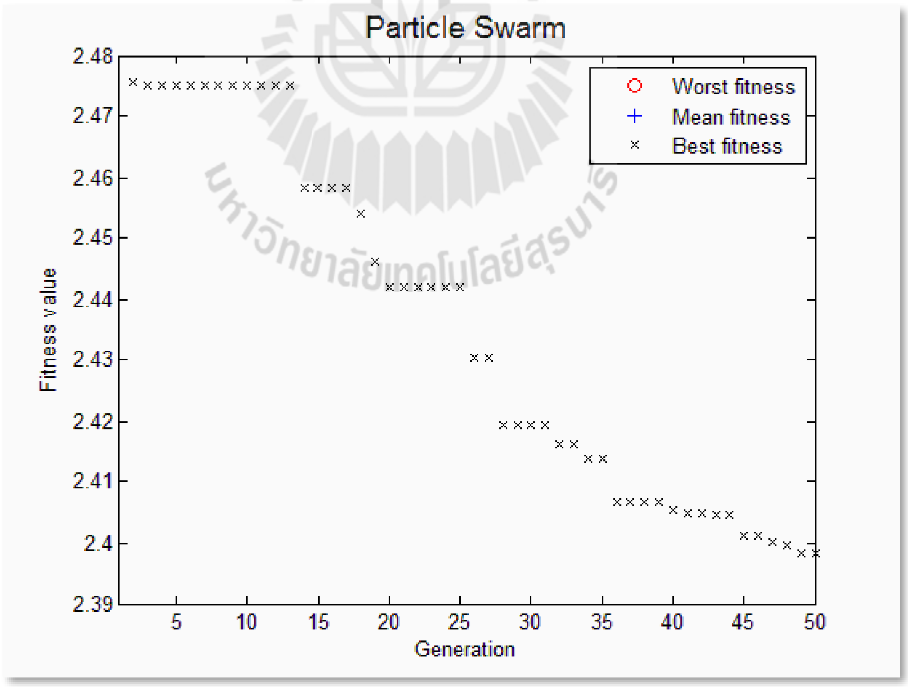
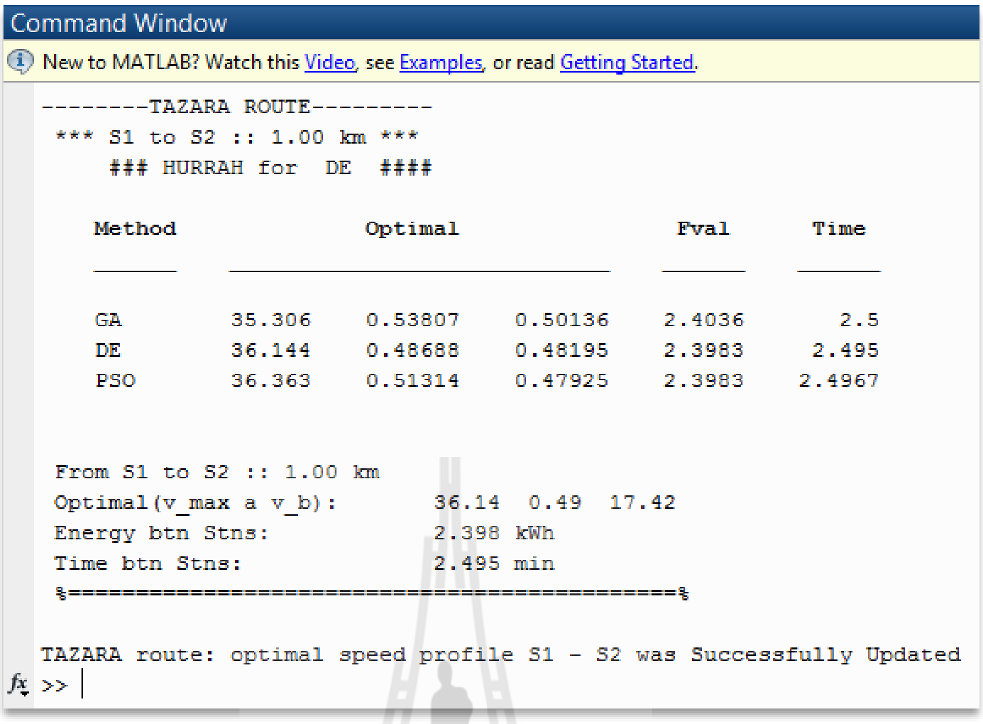


Figure C.52 TAZARA route, S1 to S2 PSO fitness value plot



Command Window

New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
--------TAZARA ROUTE---------
 *** S1 to S2 :: 1.00 km ***
     ### HURRAH for  DE  ####

    Method              Optimal                    Fval      Time

    ______    ____________________________    ______    ______

    GA        35.306     0.53807     0.50136    2.4036       2.5
    DE        36.144     0.48688     0.48195    2.3983     2.495
    PSO       36.363     0.51314     0.47925    2.3983    2.4967


 From S1 to S2 :: 1.00 km
 Optimal(v_max a v_b):        36.14   0.49  17.42
 Energy btn Stns:             2.398 kWh
 Time btn Stns:               2.495 min
 %=============================================%

TAZARA route: optimal speed profile S1 - S2 was Successfully Updated
>> |
```

Figure C.53 TAZARA route, S1 to S2 GA, DE and PSO optimal values

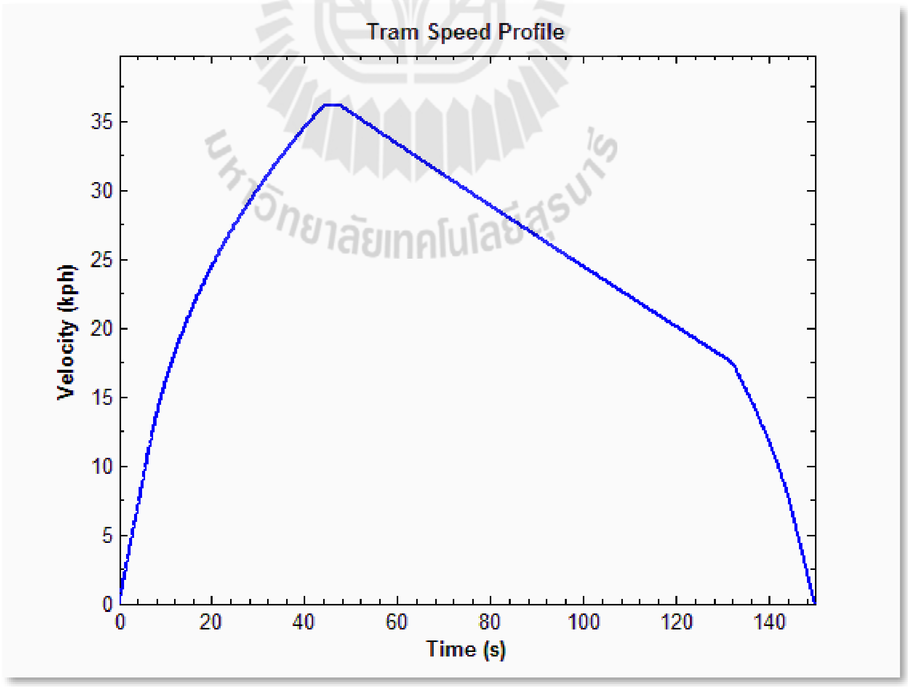


Figure C.54 TAZARA route, S1 to S2 the best speed profile, velocity against time

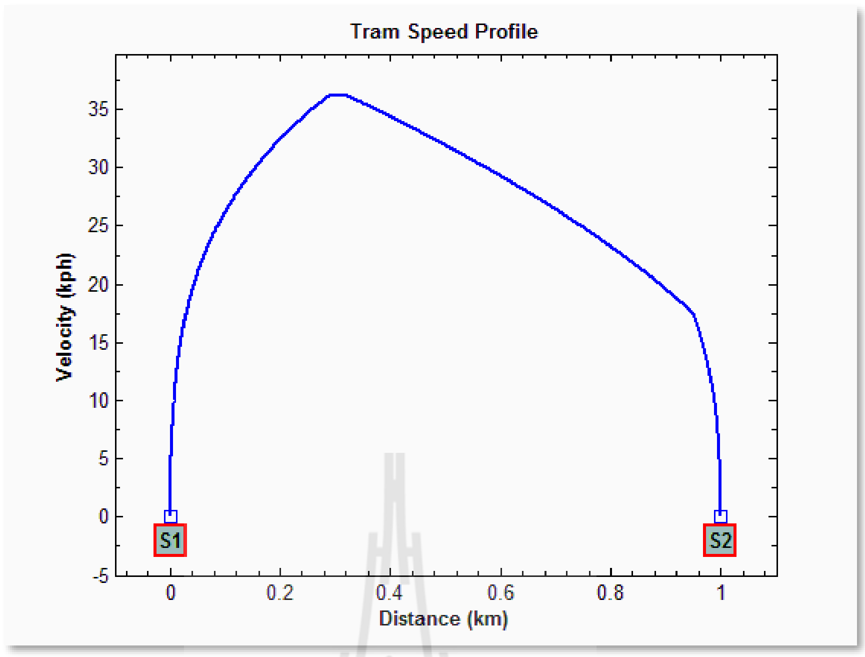


Figure C.55 TAZARA route, S1 to S2 the best speed profile, velocity against distance

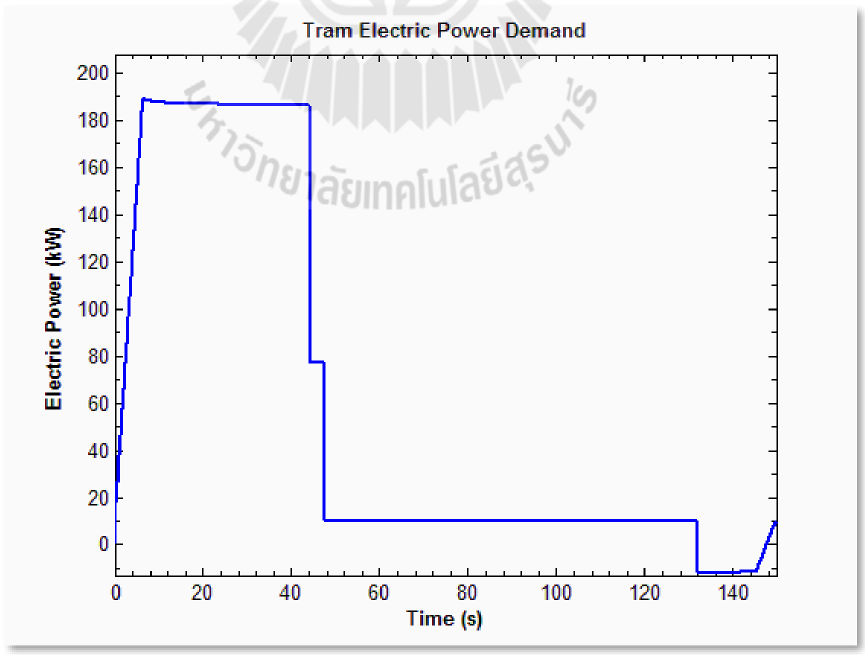


Figure C.56 TAZARA route, S1 to S2 power demand corresponding to the best speed profile

C.2.2 TAZARA Route; S2-S3

Speed profile optimization results for TAZARA route, from S2 to S3 were as presented in Figure C.57 through Figure C.63.

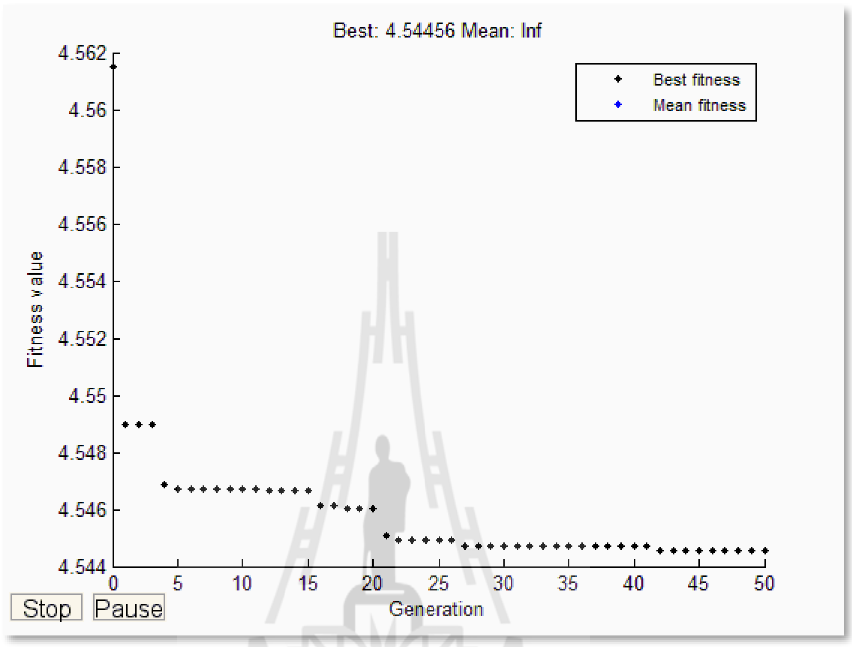


Figure C.57 TAZARA route, S2 to S3 GA fitness value plot

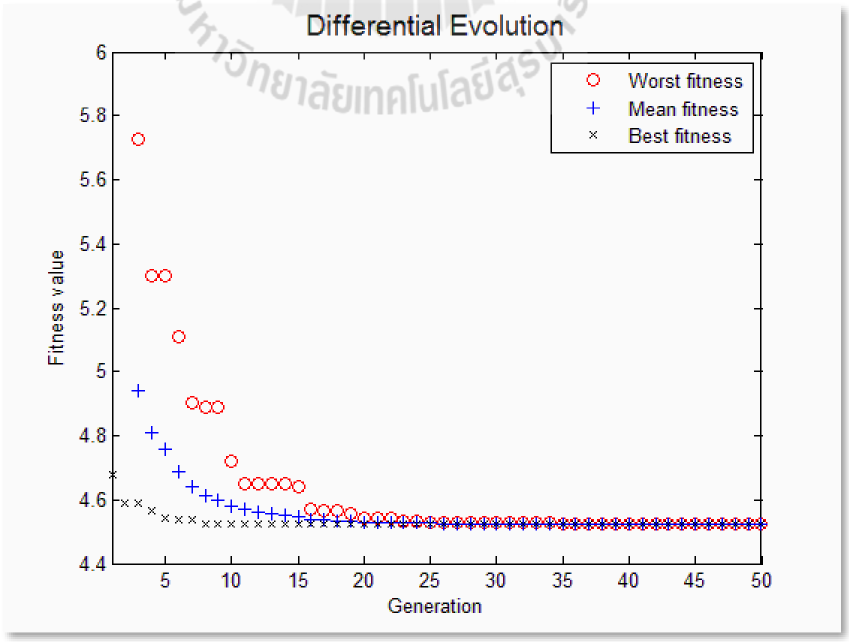


Figure C.58 TAZARA route, S2 to S3 DE fitness value plot

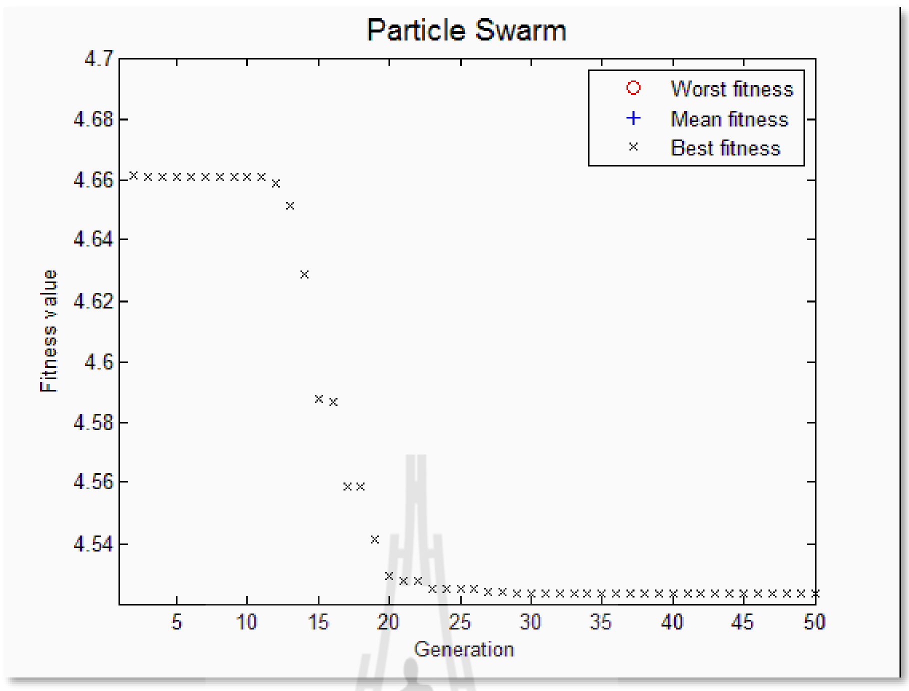
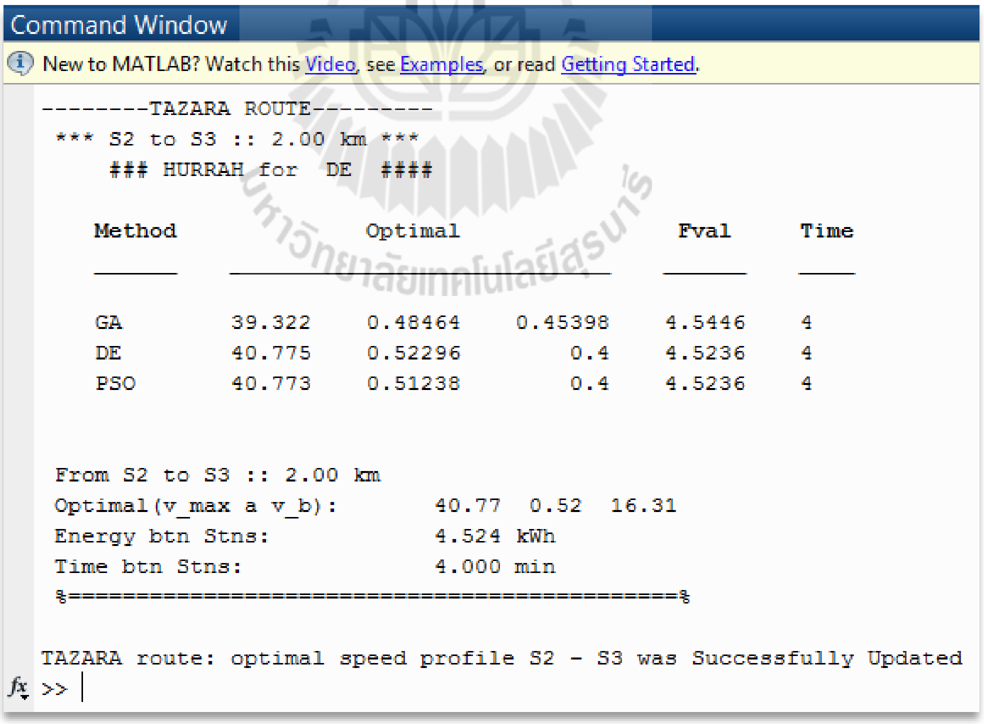


Figure C.59 TAZARA route, S2 to S3 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
--------TAZARA ROUTE---------
 *** S2 to S3 :: 2.00 km ***
     ### HURRAH for  DE  ####

    Method              Optimal                 Fval      Time
    ______    ____________________________    ______    ____

    GA        39.322    0.48464    0.45398    4.5446    4
    DE        40.775    0.52296        0.4    4.5236    4
    PSO       40.773    0.51238        0.4    4.5236    4


 From S2 to S3 :: 2.00 km
 Optimal(v_max a v_b):        40.77  0.52  16.31
 Energy btn Stns:             4.524 kWh
 Time btn Stns:               4.000 min
 %==============================================%

TAZARA route: optimal speed profile S2 - S3 was Successfully Updated
>>
```

Figure C.60 TAZARA route, S2 to S3 GA, DE and PSO optimal values

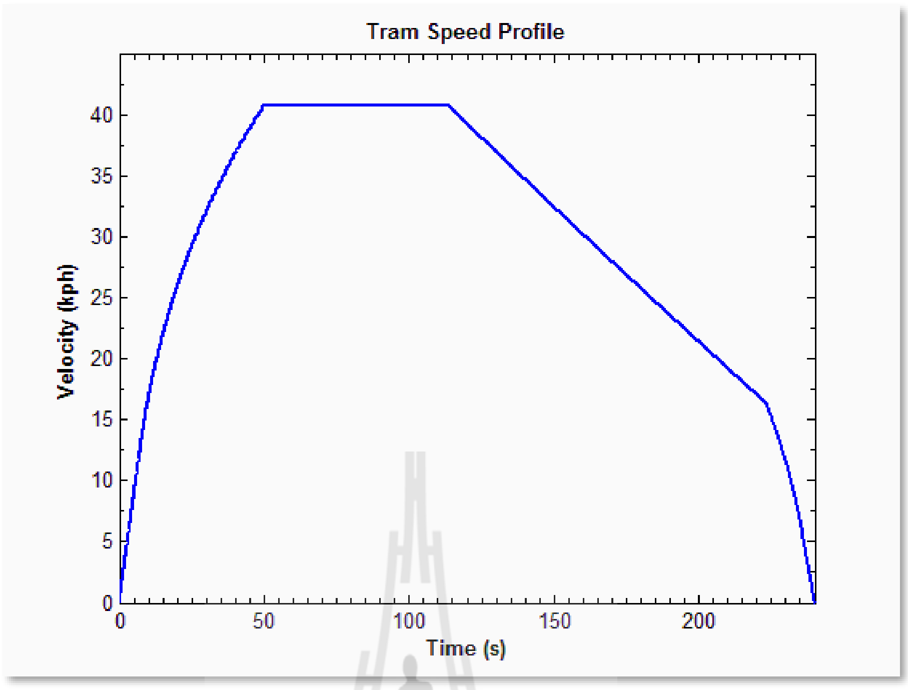


Figure C.61 TAZARA route, S2 to S3 the best speed profile, velocity against time

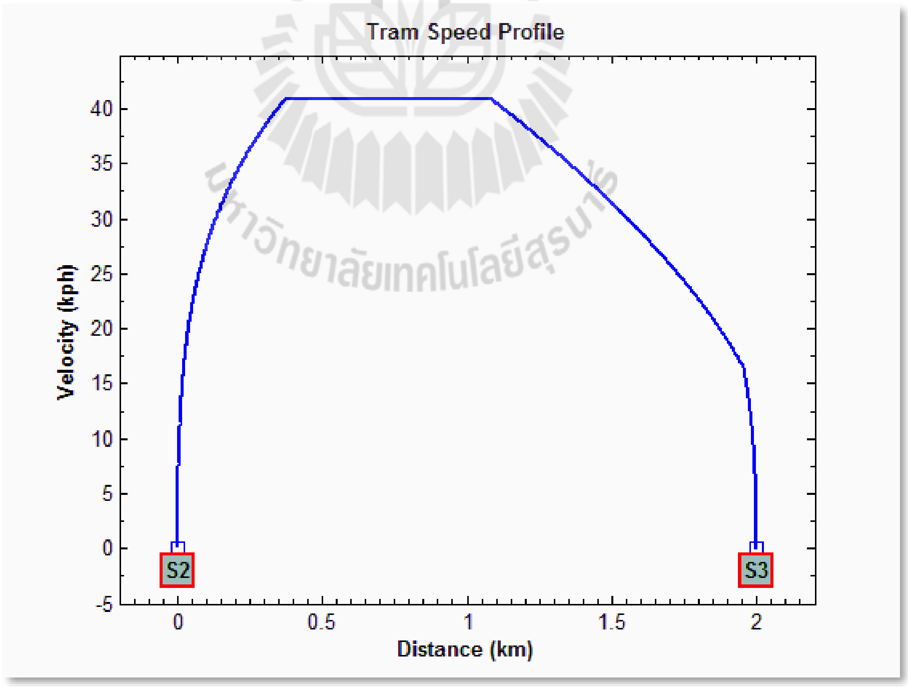


Figure C.62 TAZARA route, S2 to S3 the best speed profile, velocity against distance

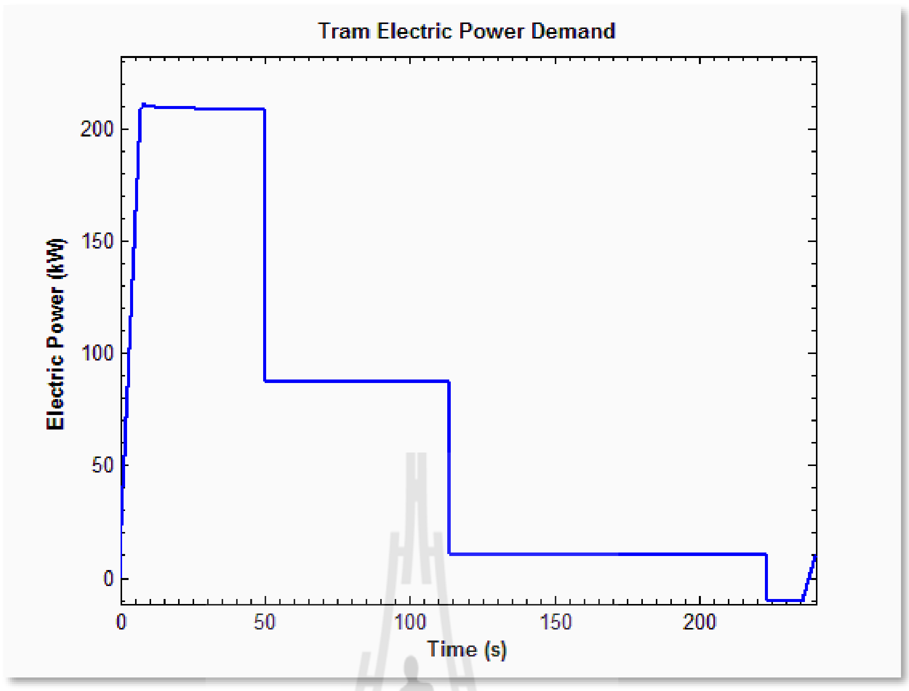


Figure C.63 TAZARA route, S2 to S3 power demand corresponding to the best speed profile

C.2.3 TAZARA Route; S3-S4

Speed profile optimization results for TAZARA route, from S3 to S4 were as presented in Figure C.64 through Figure C.70.

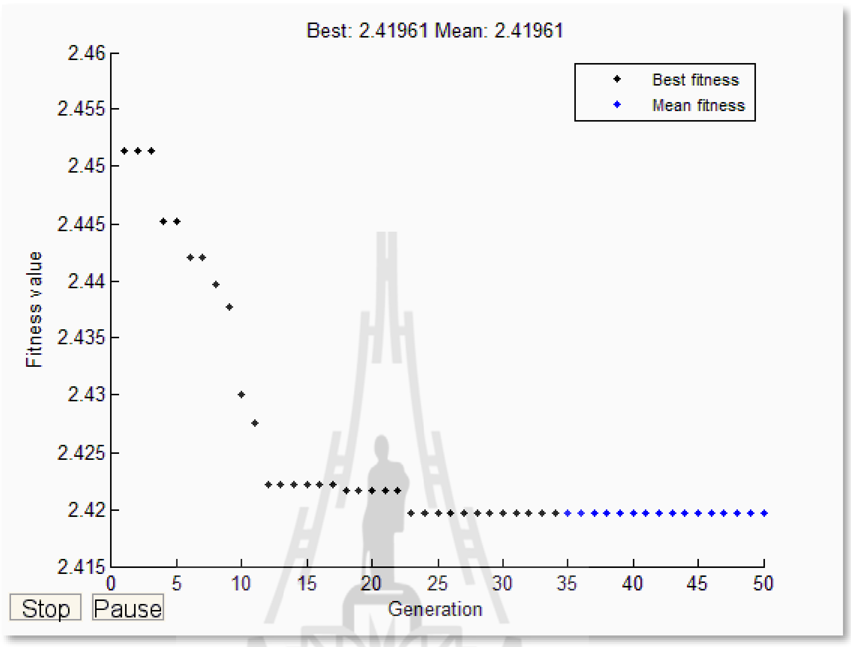


Figure C.64 TAZARA route, S3 to S4 GA fitness value plot

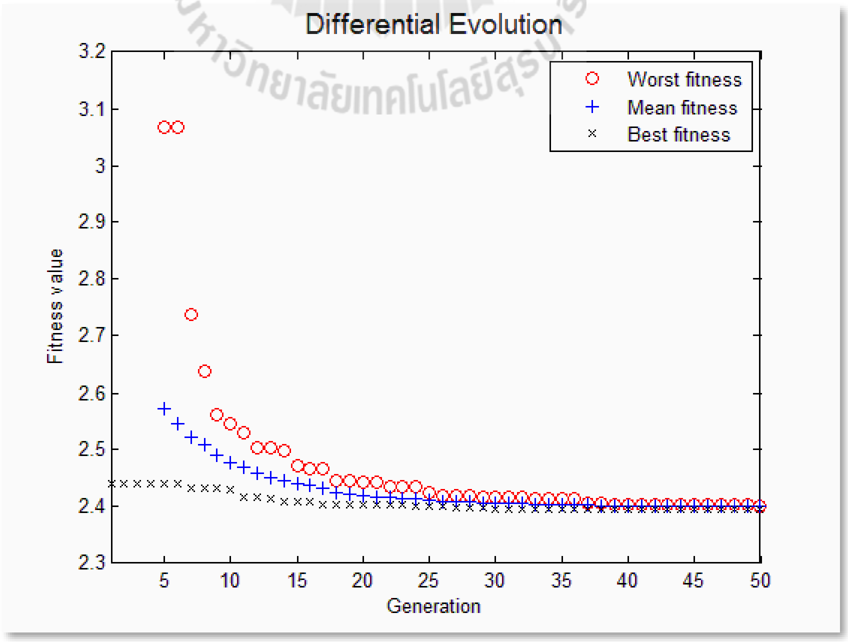


Figure C.65 TAZARA route, S3 to S4 DE fitness value plot

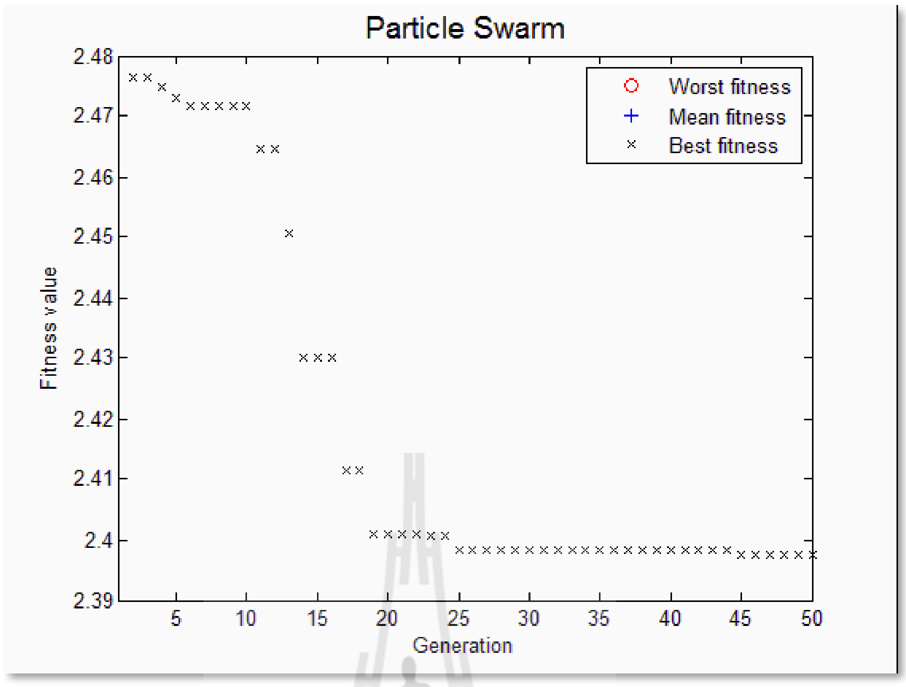
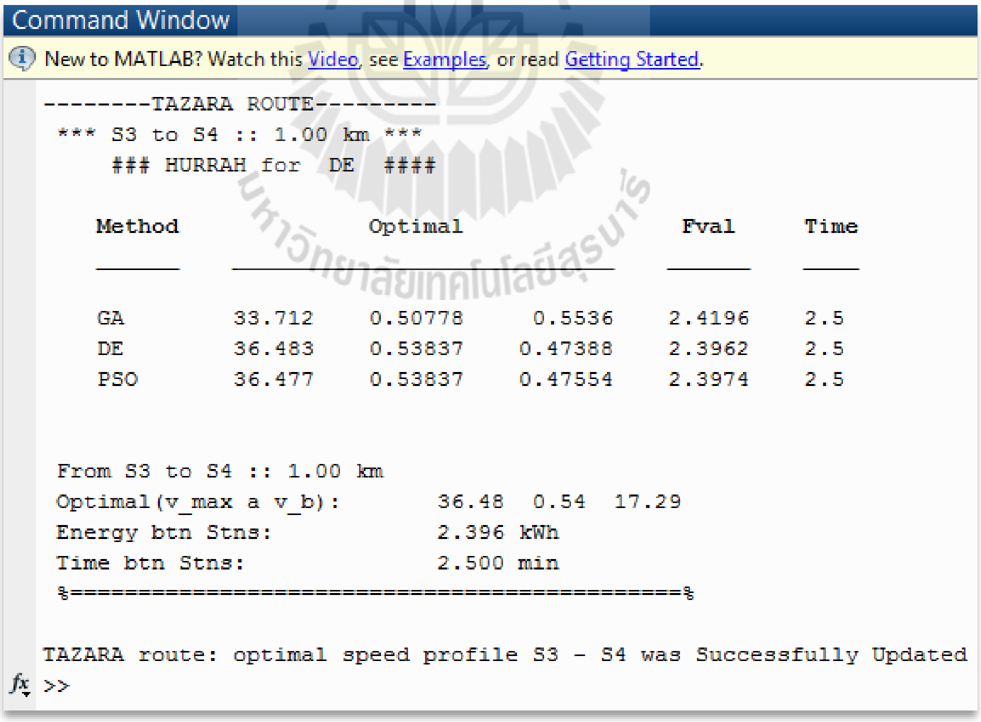


Figure C.66 TAZARA route, S3 to S4 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
--------TAZARA ROUTE---------
 *** S3 to S4 :: 1.00 km ***
     ### HURRAH for  DE  ####

    Method          Optimal                    Fval      Time
    ______    ____________________________    ______    ____

    GA        33.712    0.50778     0.5536    2.4196     2.5
    DE        36.483    0.53837    0.47388    2.3962     2.5
    PSO       36.477    0.53837    0.47554    2.3974     2.5


 From S3 to S4 :: 1.00 km
 Optimal(v_max a v_b):         36.48  0.54  17.29
 Energy btn Stns:              2.396 kWh
 Time btn Stns:                2.500 min
 %=============================================%

TAZARA route: optimal speed profile S3 - S4 was Successfully Updated
>>
```

Figure C.67 TAZARA route, S3 to S4 GA, DE and PSO optimal values

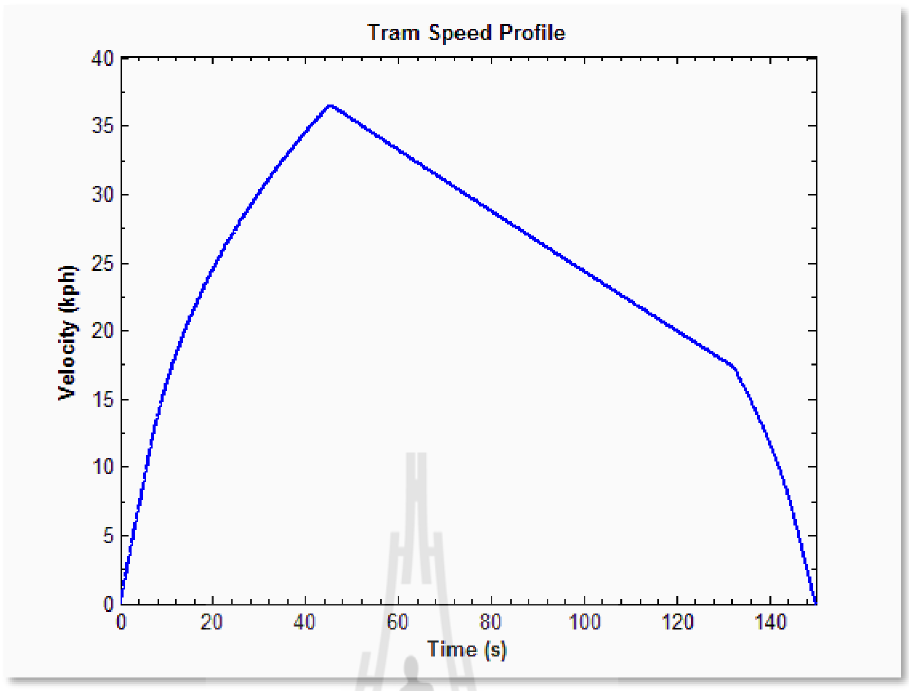


Figure C.68 TAZARA route, S3 to S4 the best speed profile, velocity against time

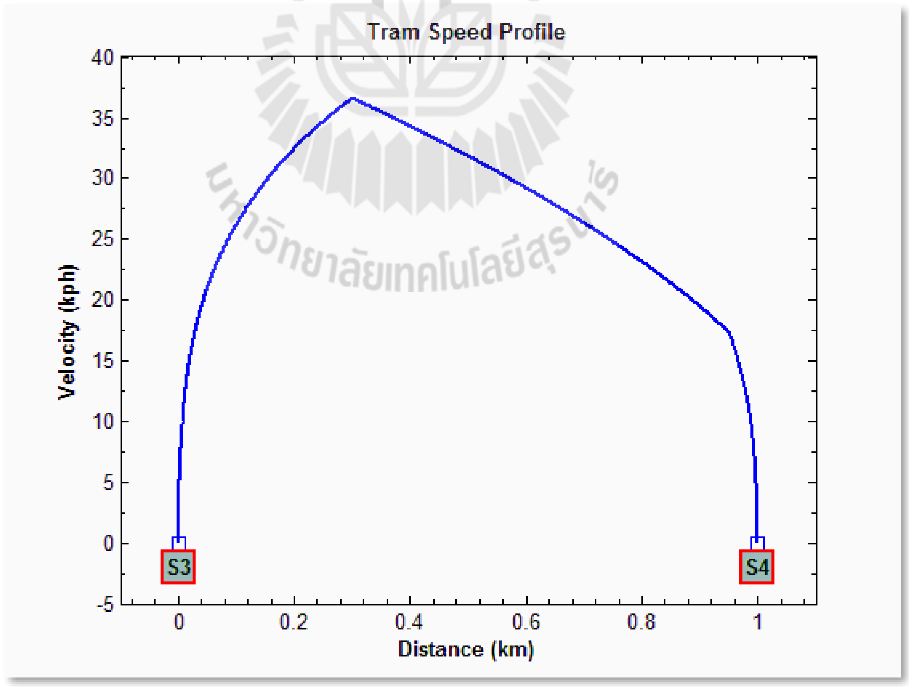


Figure C.69 TAZARA route, S3 to S4 the best speed profile, velocity against distance

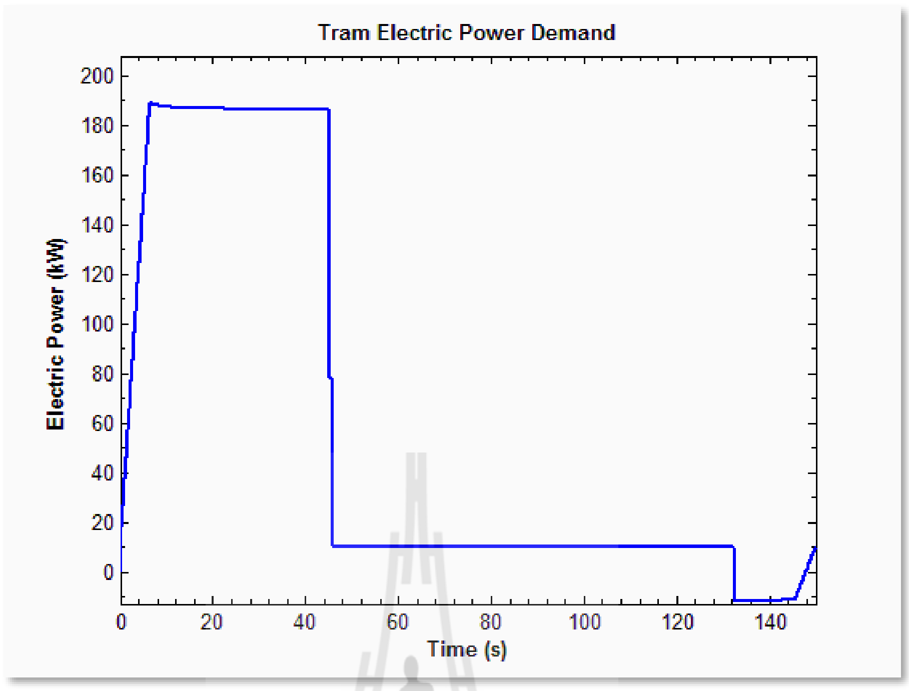


Figure C.70 TAZARA route, S3 to S4 power demand corresponding to the best speed profile

C.2.4 TAZARA Route; S4-S5

Speed profile optimization results for TAZARA route, from S4 to S5 were as presented in Figure C.71 through Figure C.77.

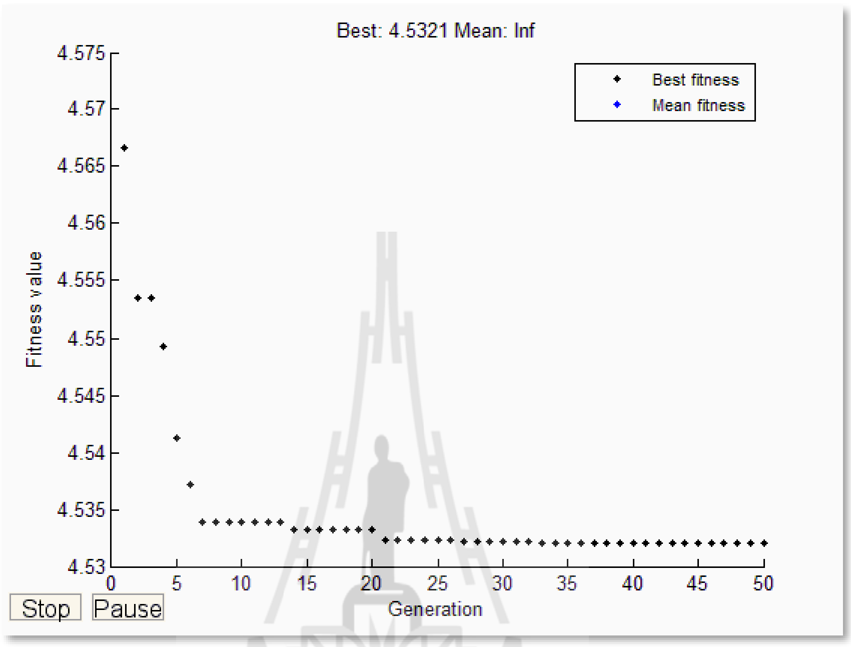


Figure C.71 TAZARA route, S4 to S5 GA fitness value plot

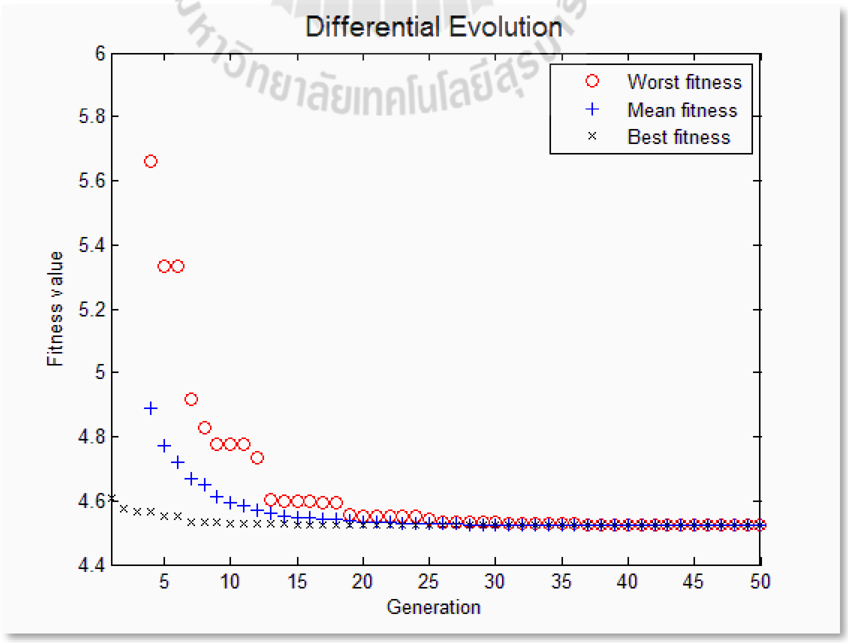


Figure C.72 TAZARA route, S4 to S5 DE fitness value plot

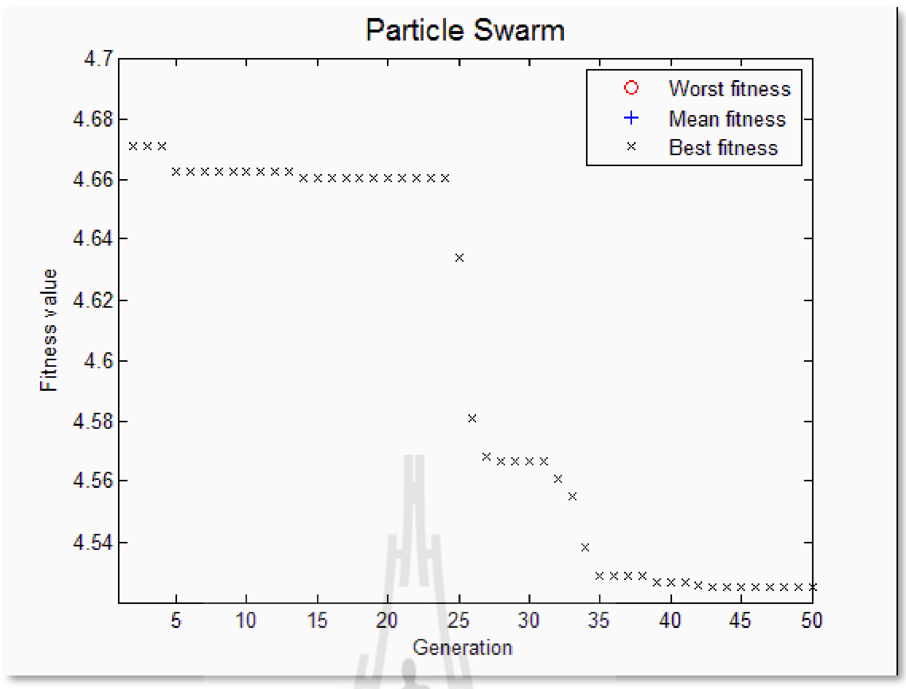
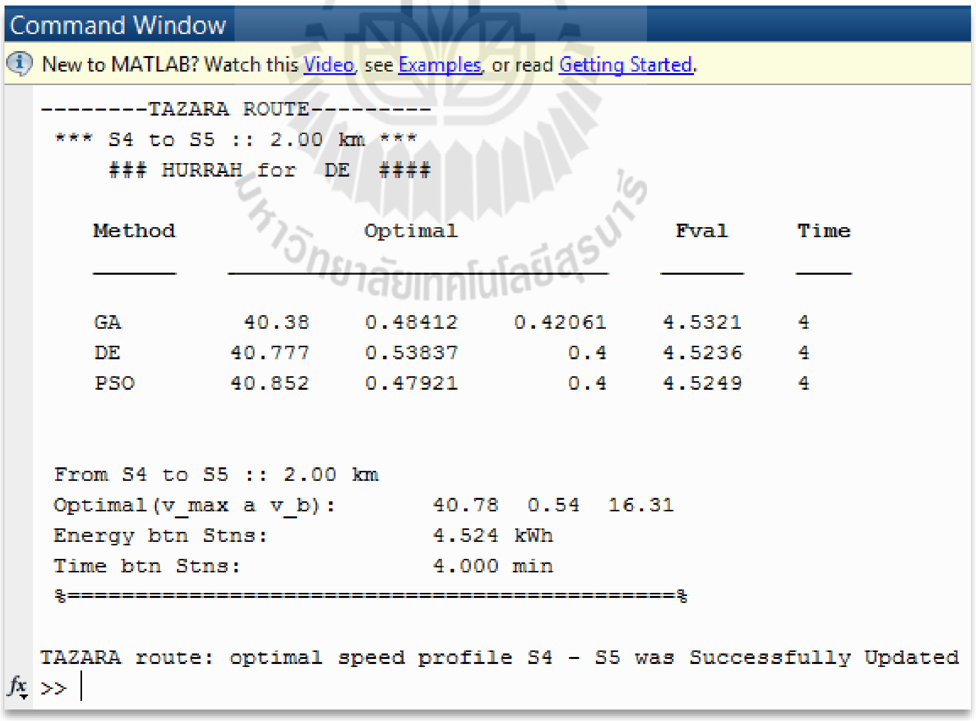


Figure C.73 TAZARA route, S4 to S5 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
--------TAZARA ROUTE---------
 *** S4 to S5 :: 2.00 km ***
     ### HURRAH for  DE   ####

    Method          Optimal                 Fval      Time

    ______     ____________________________      ______    ____

    GA          40.38    0.48412    0.42061    4.5321    4
    DE         40.777    0.53837        0.4    4.5236    4
    PSO        40.852    0.47921        0.4    4.5249    4


 From S4 to S5 :: 2.00 km
 Optimal(v_max a v_b):        40.78  0.54  16.31
 Energy btn Stns:             4.524 kWh
 Time btn Stns:               4.000 min
 %==============================================%

TAZARA route: optimal speed profile S4 - S5 was Successfully Updated
>> 
```

Figure C.74 TAZARA route, S4 to S5 GA, DE and PSO optimal values

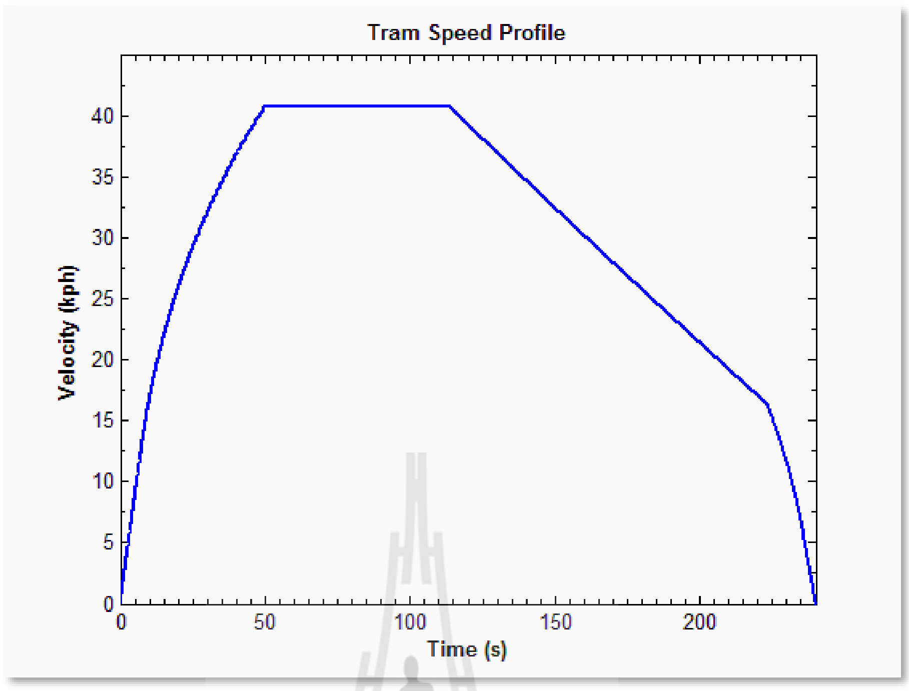


Figure C.75 TAZARA route, S4 to S5 the best speed profile, velocity against time

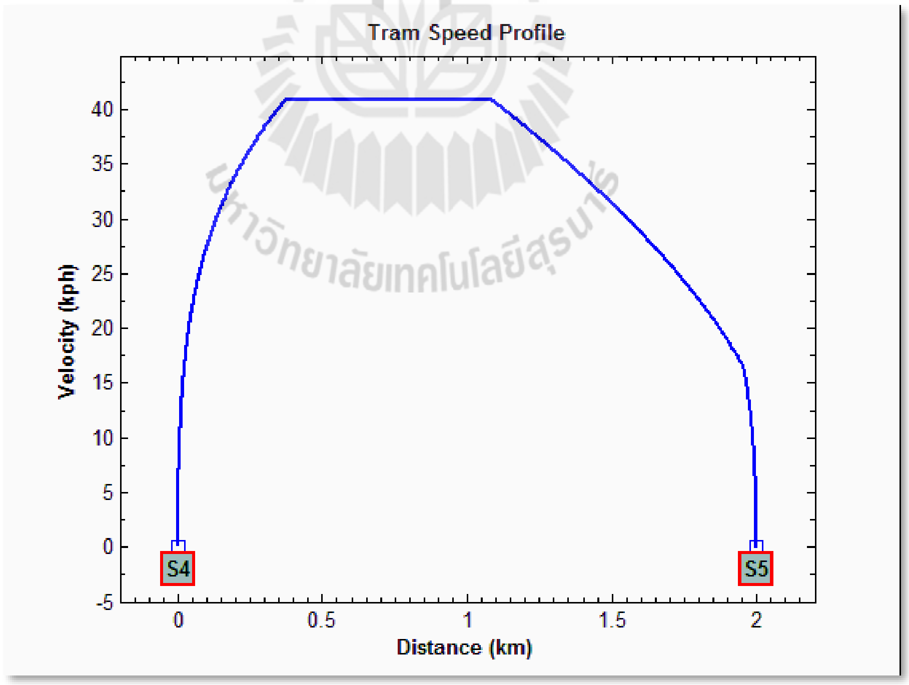


Figure C.76 TAZARA route, S4 to S5 the best speed profile, velocity against distance

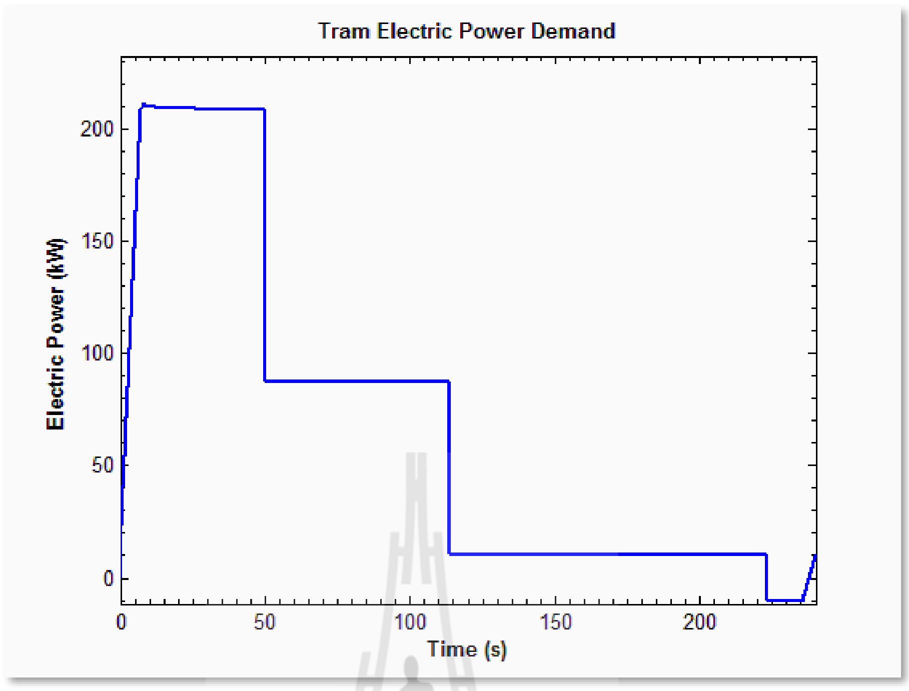


Figure C.77 TAZARA route, S4 to S5 power demand corresponding to the best speed profile

C.2.5 TAZARA Route; S5-S6

Speed profile optimization results for TAZARA route, from S5 to S6 were as presented in Figure C.78 through Figure C.84.

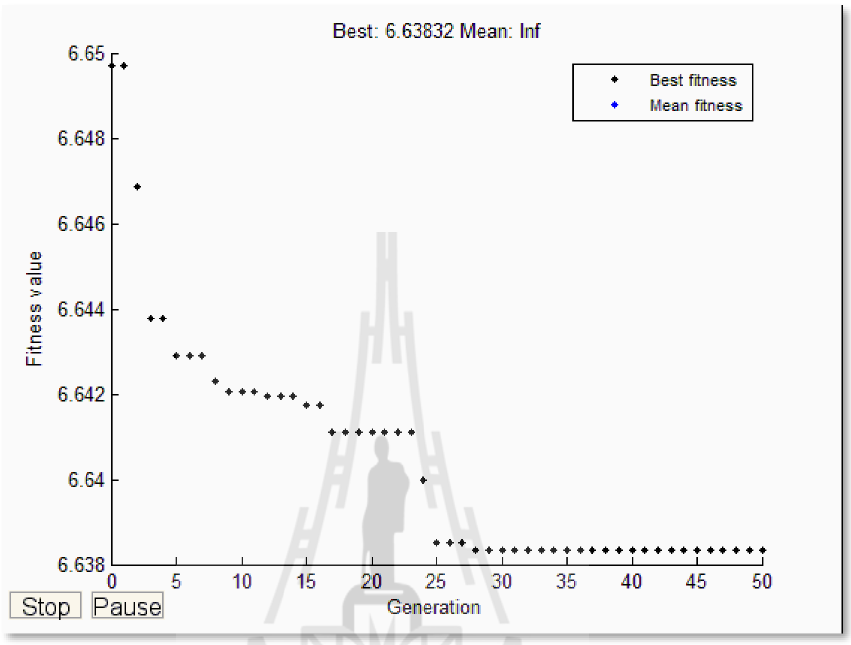


Figure C.78 TAZARA route, S5 to S6 GA fitness value plot

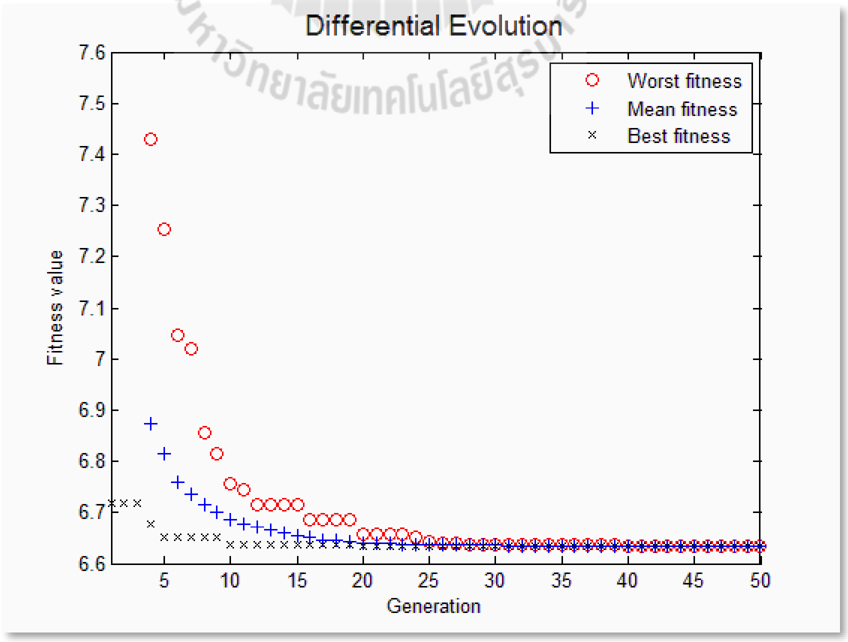


Figure C.79 TAZARA route, S5 to S6 DE fitness value plot

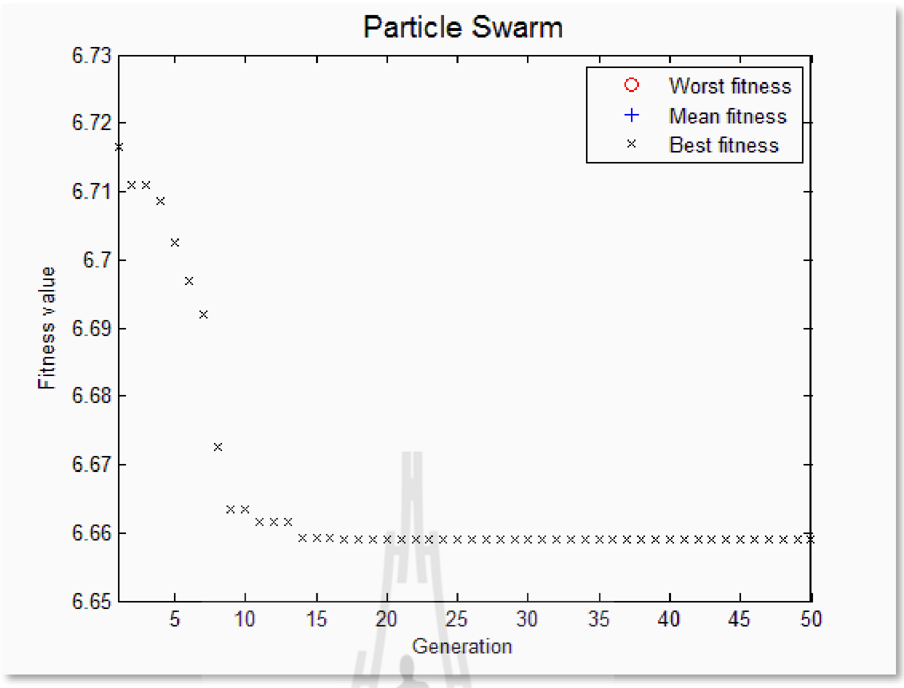
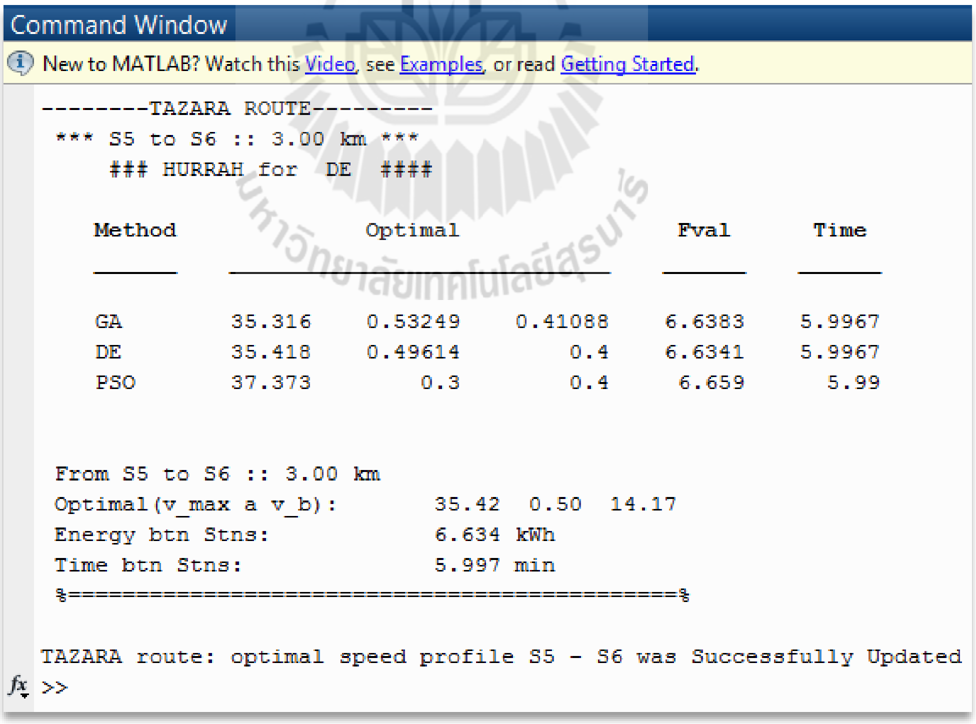


Figure C.80 TAZARA route, S5 to S6 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
--------TAZARA ROUTE---------
 *** S5 to S6 :: 3.00 km ***
     ### HURRAH for  DE  ####

    Method           Optimal                 Fval      Time
    ______     ____________________________   ______    ______

    GA         35.316    0.53249    0.41088    6.6383    5.9967
    DE         35.418    0.49614        0.4    6.6341    5.9967
    PSO        37.373        0.3        0.4     6.659      5.99


 From S5 to S6 :: 3.00 km
 Optimal(v_max a v_b):          35.42   0.50   14.17
 Energy btn Stns:               6.634 kWh
 Time btn Stns:                 5.997 min
 %==============================================%

TAZARA route: optimal speed profile S5 - S6 was Successfully Updated
>>
```

Figure C.81 TAZARA route, S5 to S6 GA, DE and PSO optimal values

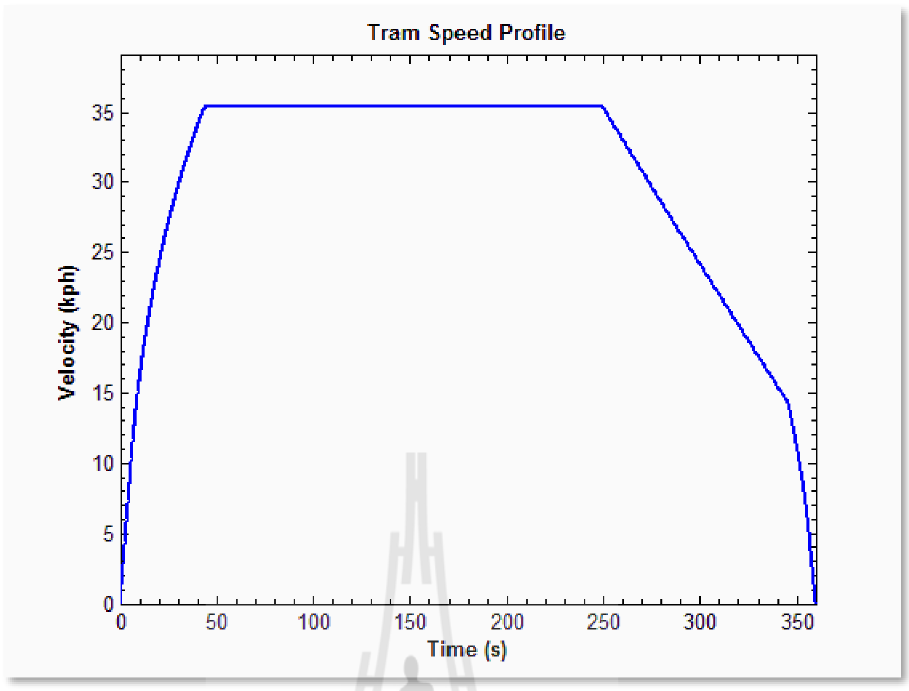


Figure C.82 TAZARA route, S5 to S6 the best speed profile, velocity against time

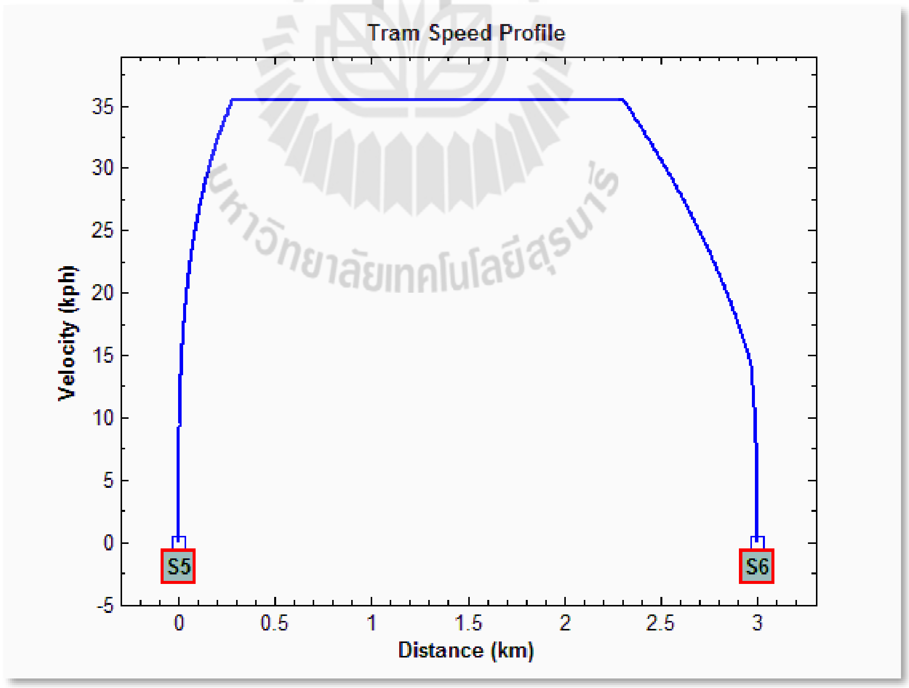


Figure C.83 TAZARA route, S5 to S6 the best speed profile, velocity against distance

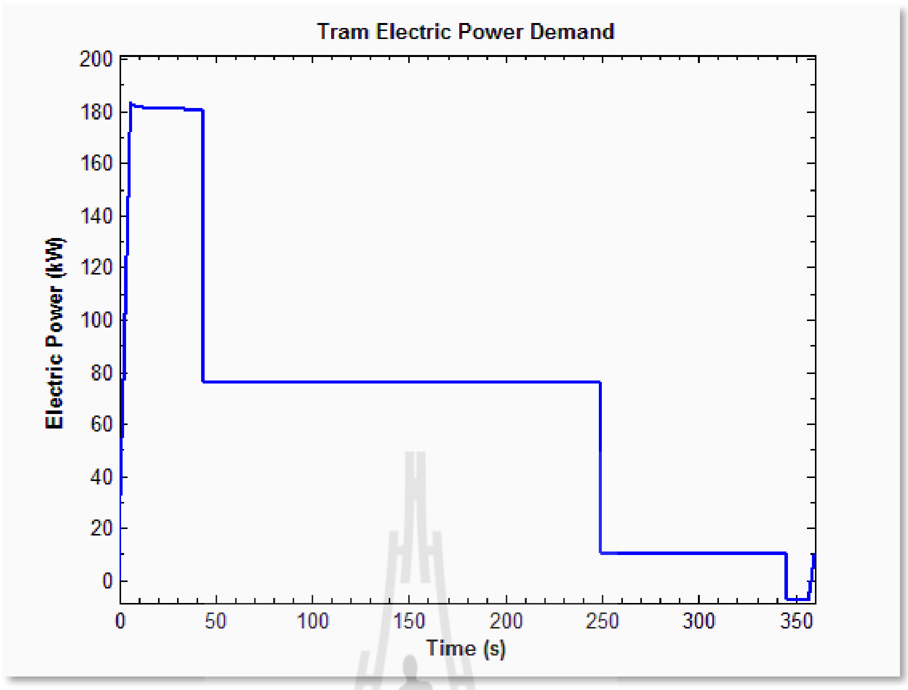


Figure C.84 TAZARA route, S4 to S5 power demand corresponding to the best speed profile

C.2.6 TAZARA Route; S6-S7

Speed profile optimization results for TAZARA route, from S6 to S7 were as presented in Figure C.85 through Figure C.91.

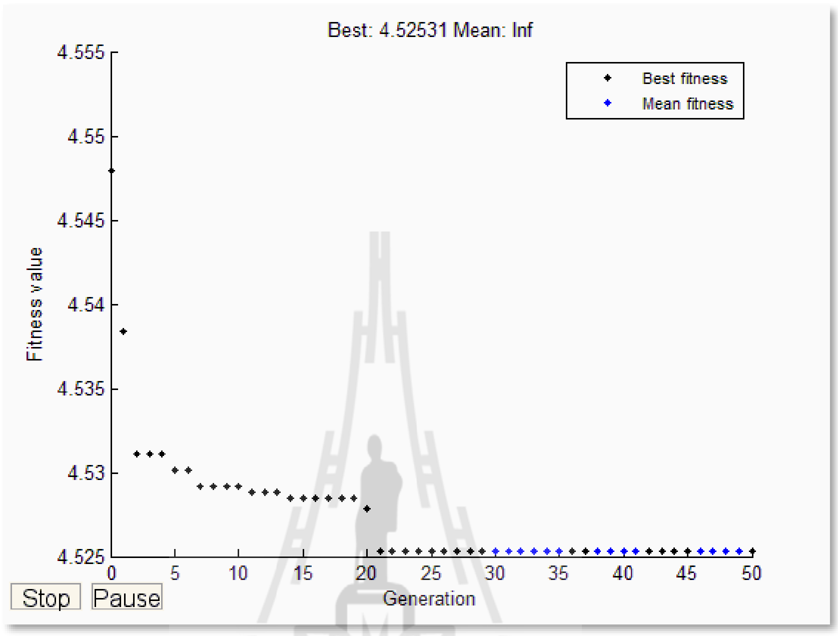


Figure C.85 TAZARA route, S6 to S7 GA fitness value plot

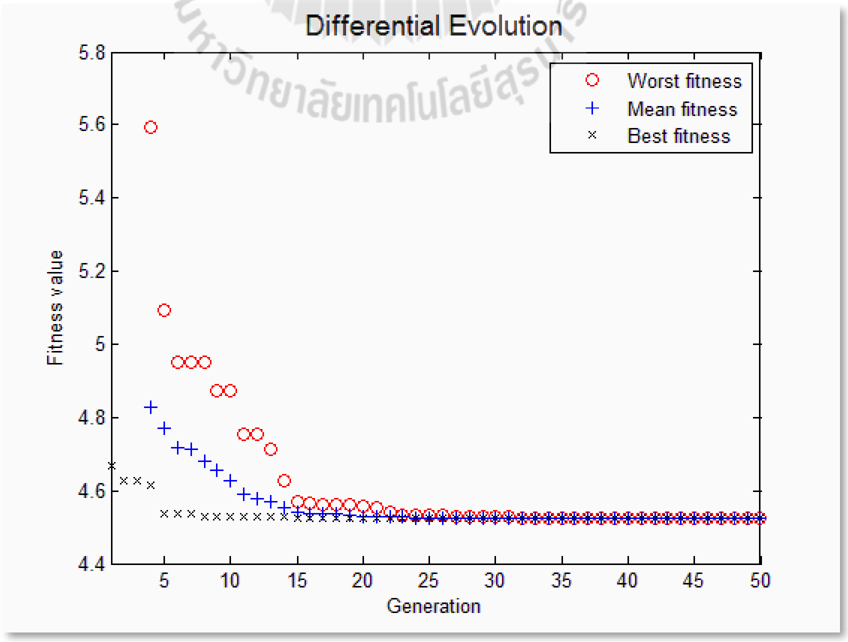


Figure C.86 TAZARA route, S6 to S7 DE fitness value plot

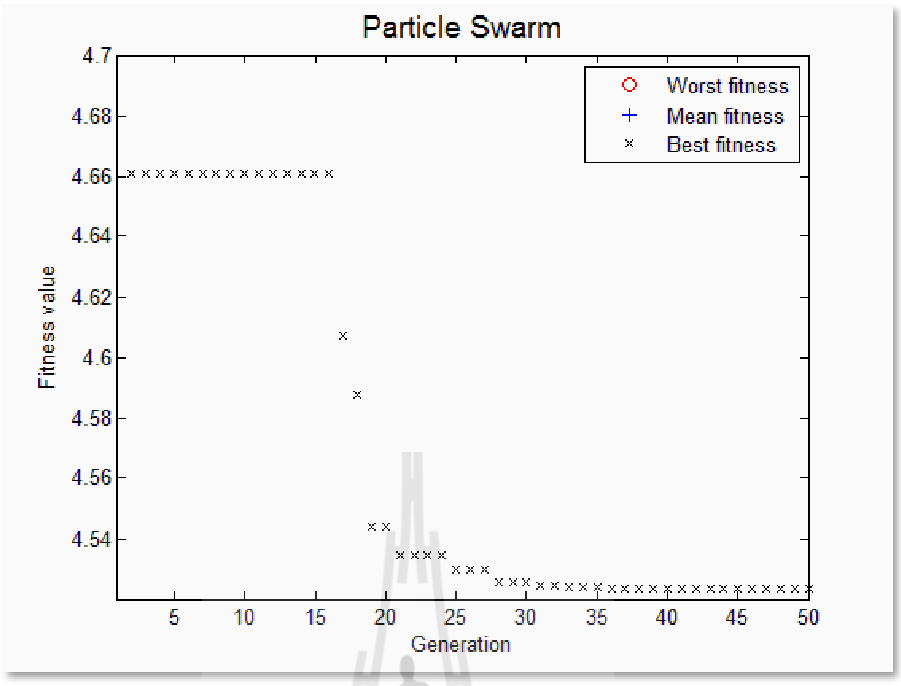
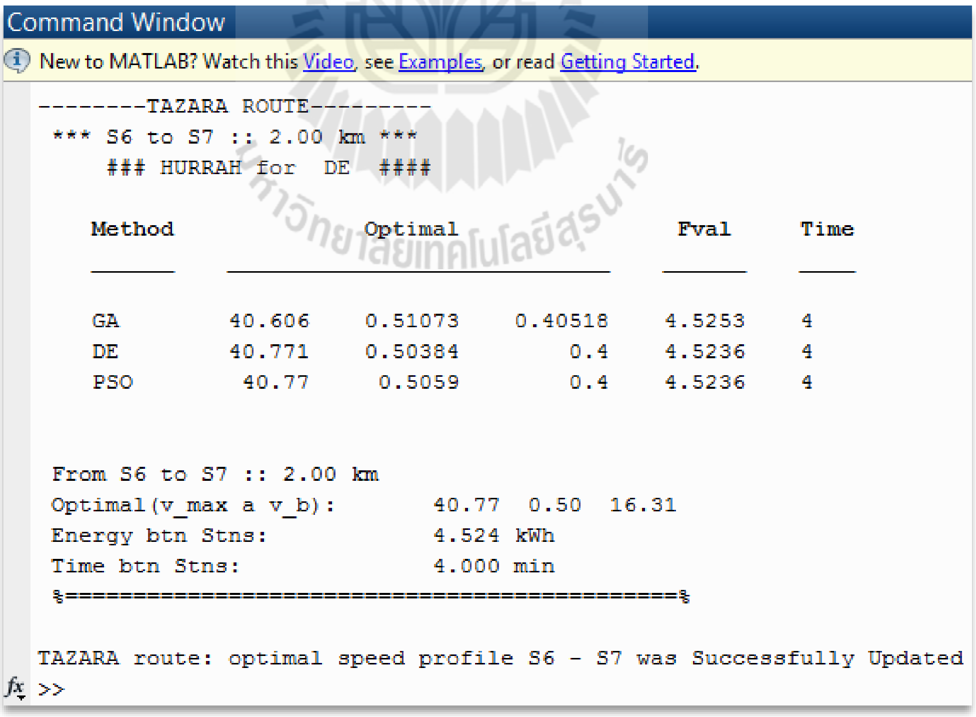


Figure C.87 TAZARA route, S6 to S7 PSO fitness value plot



Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
--------TAZARA ROUTE---------
 *** S6 to S7 :: 2.00 km ***
     ### HURRAH for  DE  ####

    Method          Optimal                    Fval      Time
    ______    ____________________________    ______    ____

    GA        40.606    0.51073    0.40518    4.5253     4
    DE        40.771    0.50384        0.4    4.5236     4
    PSO        40.77     0.5059        0.4    4.5236     4


 From S6 to S7 :: 2.00 km
 Optimal(v_max a v_b):       40.77  0.50  16.31
 Energy btn Stns:            4.524 kWh
 Time btn Stns:              4.000 min
 %==============================================%

TAZARA route: optimal speed profile S6 - S7 was Successfully Updated
>>
```

Figure C.88 TRL route, S6 to S7 GA, DE and PSO optimal values

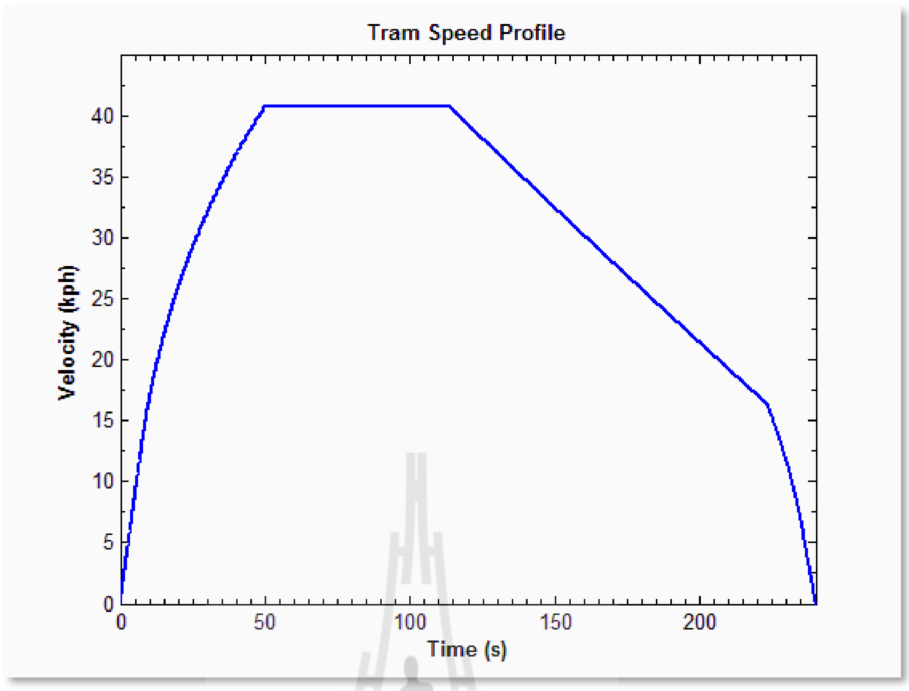


Figure C.89 TAZARA route, S6 to S7 the best speed profile, velocity against time

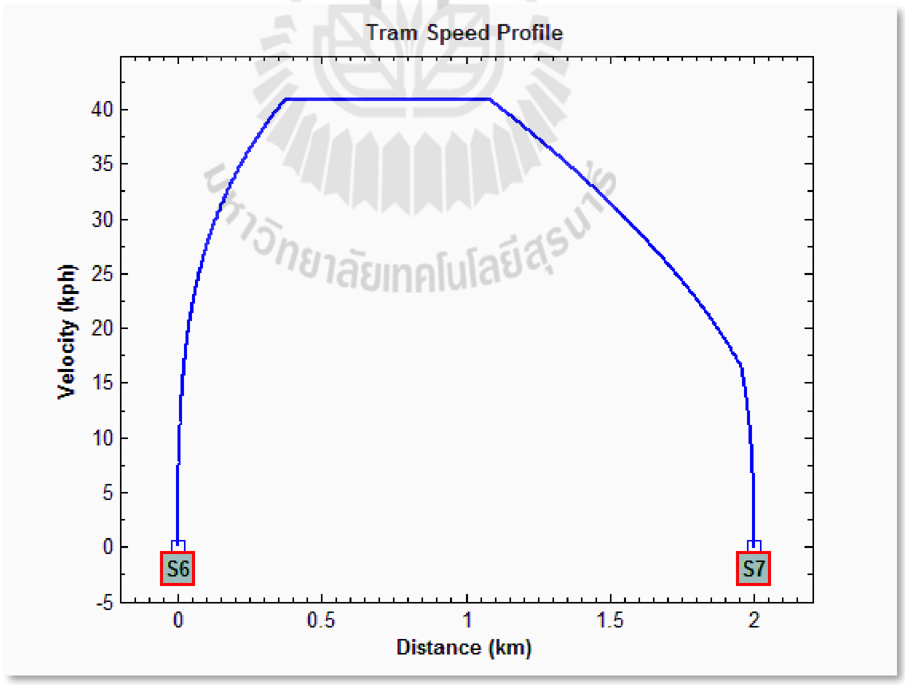


Figure C.90 TAZARA route, S6 to S7 the best speed profile, velocity against distance

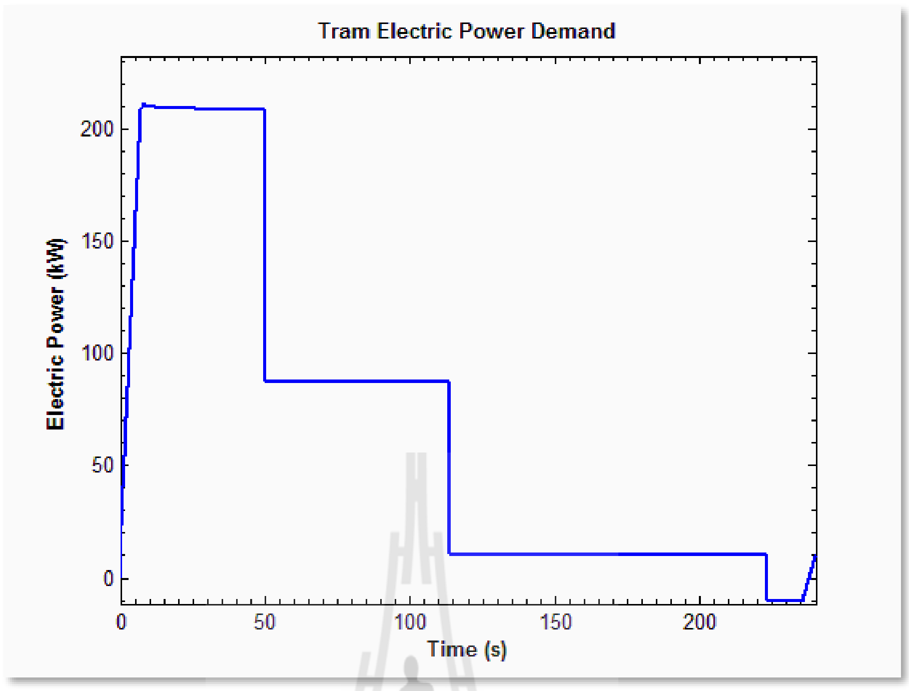


Figure C.91 TAZARA route, S6 to S7 power demand corresponding to the best speed profile

C.2.7 TAZARA Route; S7-S8

Speed profile optimization results for TAZARA route, from S7 to S8 were as presented in Figure C.92 through Figure C.98.

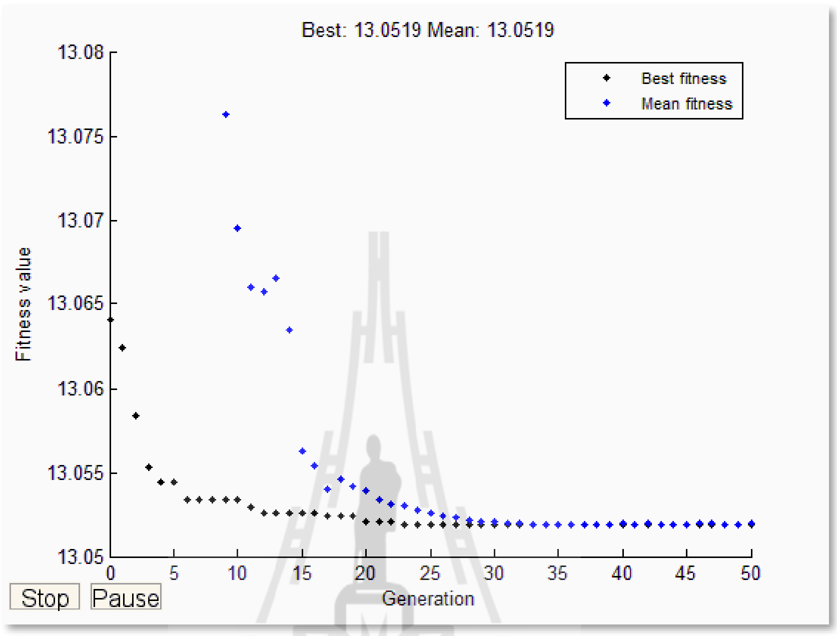


Figure C.92 TAZARA route, S7 to S8 GA fitness value plot

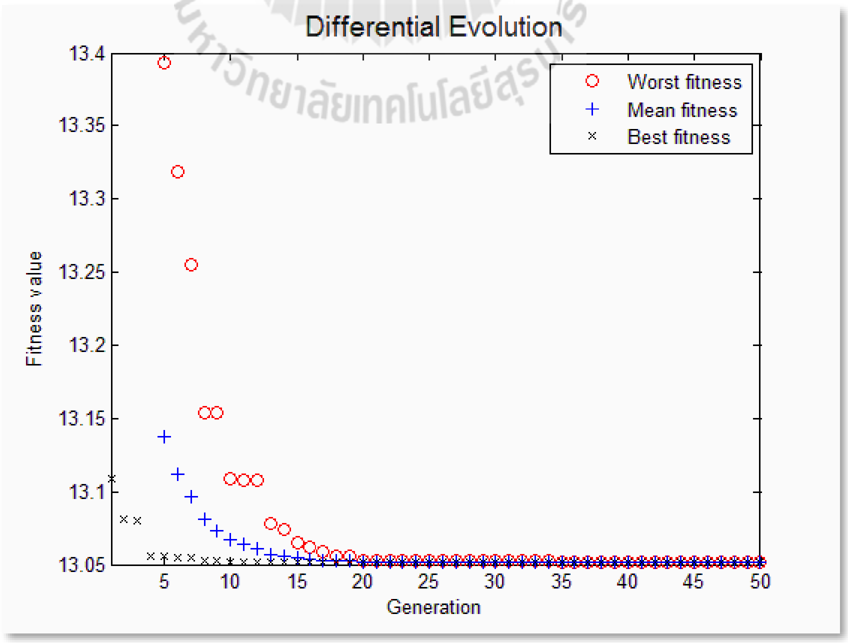


Figure C.93 TAZARA route, S7 to S8 DE fitness value plot

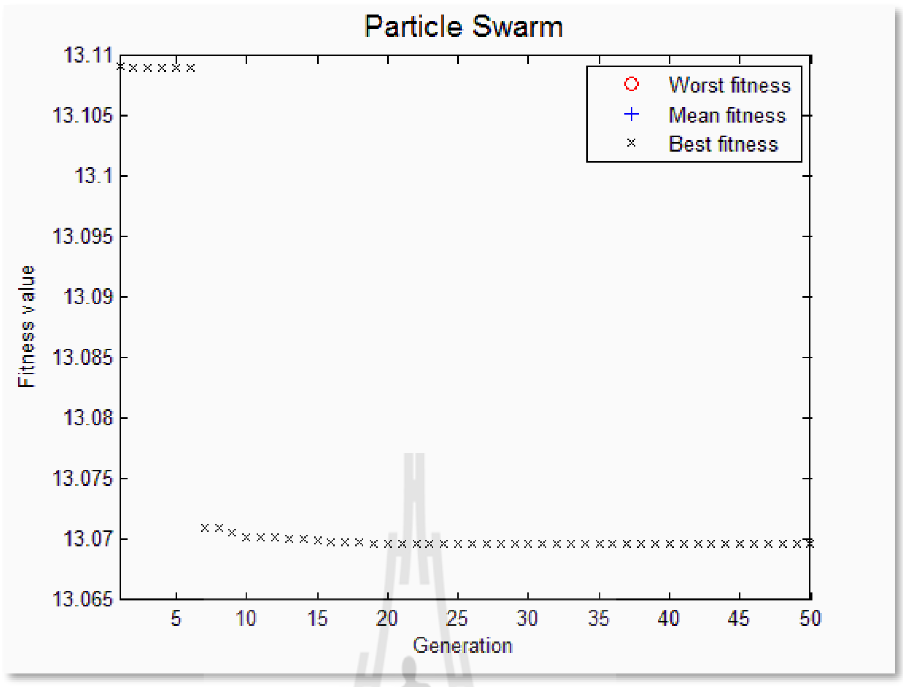
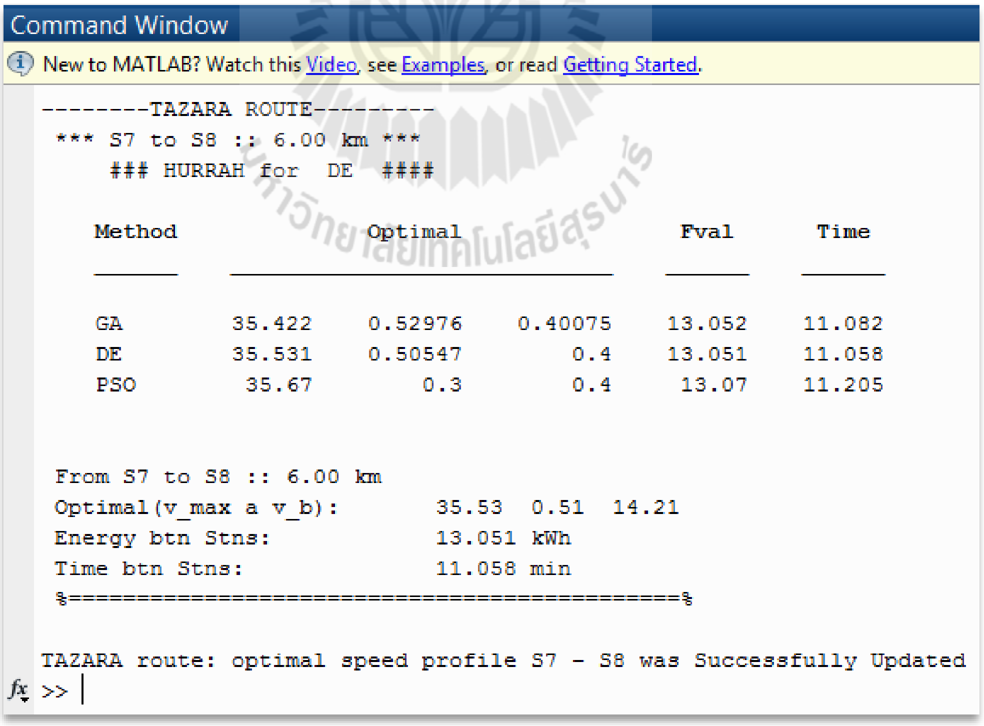


Figure C.94 TAZARA route, S7 to S8 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
--------TAZARA ROUTE---------
 *** S7 to S8 :: 6.00 km ***
     ### HURRAH for  DE   ####

    Method              Optimal                    Fval       Time
    ______    ____________________________       ______     ______

    GA          35.422     0.52976     0.40075     13.052     11.082
    DE          35.531     0.50547         0.4     13.051     11.058
    PSO          35.67         0.3         0.4      13.07     11.205


 From S7 to S8 :: 6.00 km
 Optimal(v_max a v_b):        35.53  0.51  14.21
 Energy btn Stns:             13.051 kWh
 Time btn Stns:               11.058 min
 %==============================================%

TAZARA route: optimal speed profile S7 - S8 was Successfully Updated
>> |
```

Figure C.95 TAZARA route, S7 to S8 GA, DE and PSO optimal values

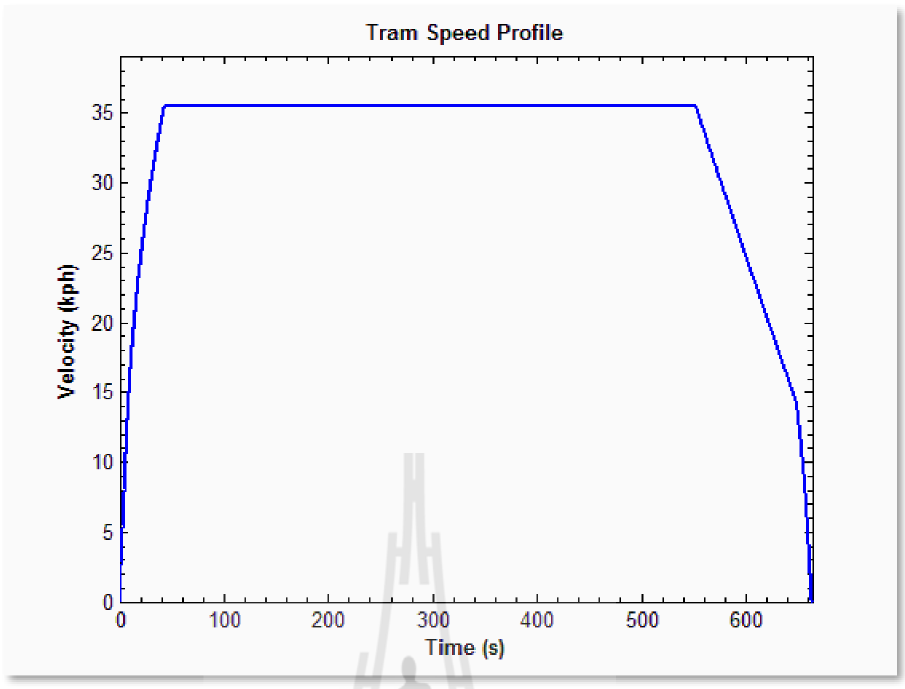


Figure C.96 TAZARA route, S7 to S8 the best speed profile, velocity against time

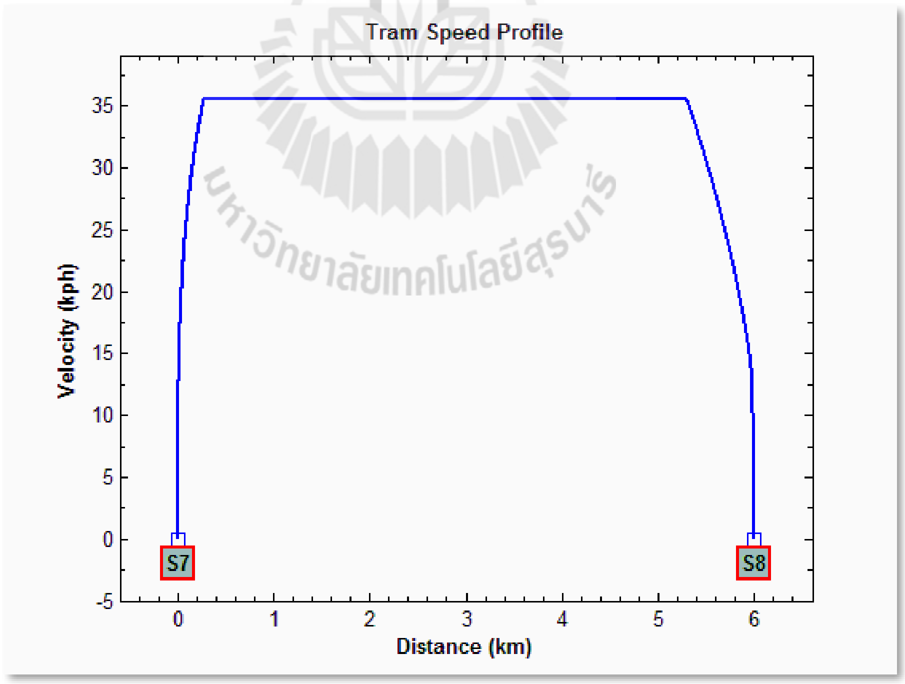


Figure C.97 TAZARA route, S7 to S8 the best speed profile, velocity against distance

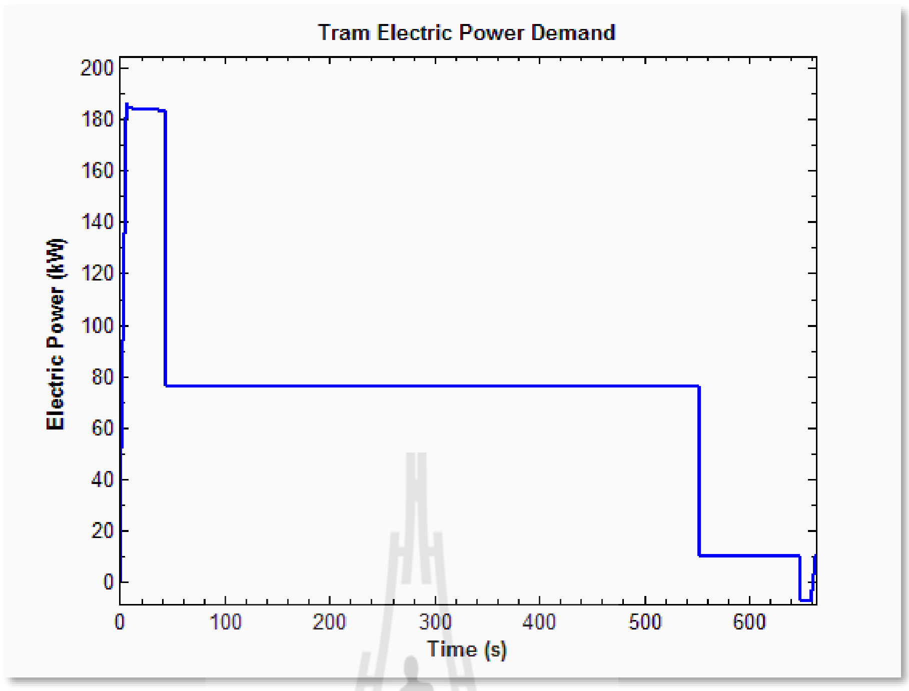


Figure C.98 TAZARA route, S7 to S8 power demand corresponding to the best speed profile

C.2.8 TAZARA Route; S8-S9

Speed profile optimization results for TAZARA route, from S8 to S9 were as presented in Figure C.99 through Figure C.105.

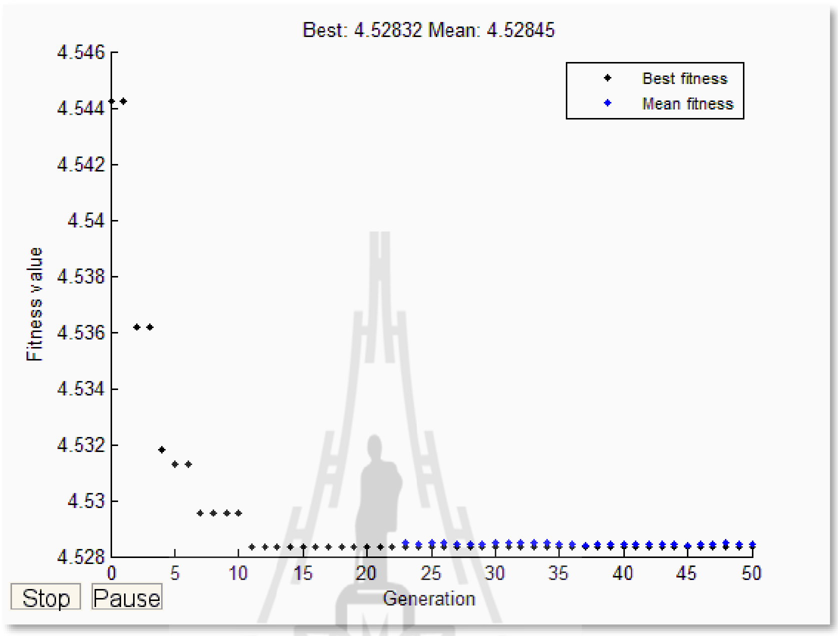


Figure C.99 TAZARA route, S8 to S9 GA fitness value plot

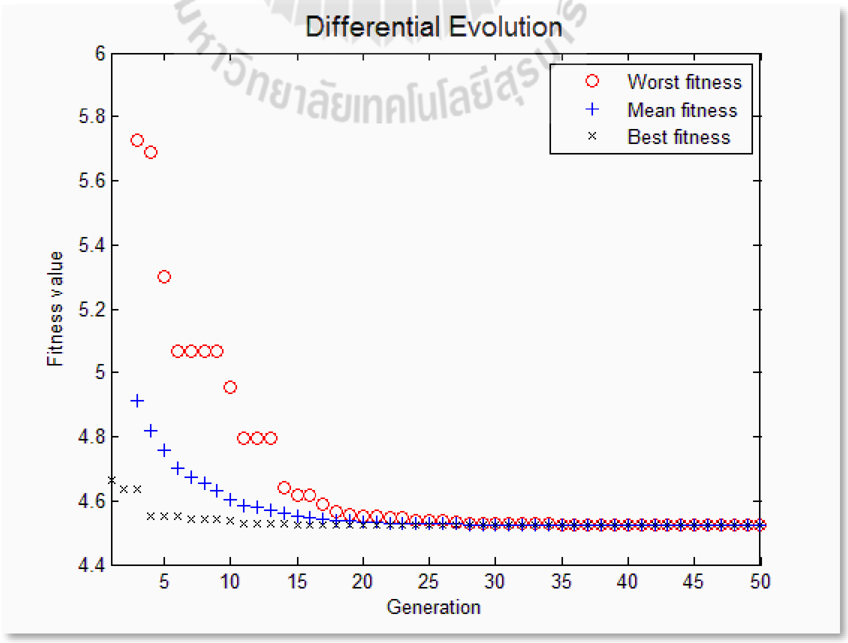


Figure C.100 TAZARA route, S8 to S9 DE fitness value plot

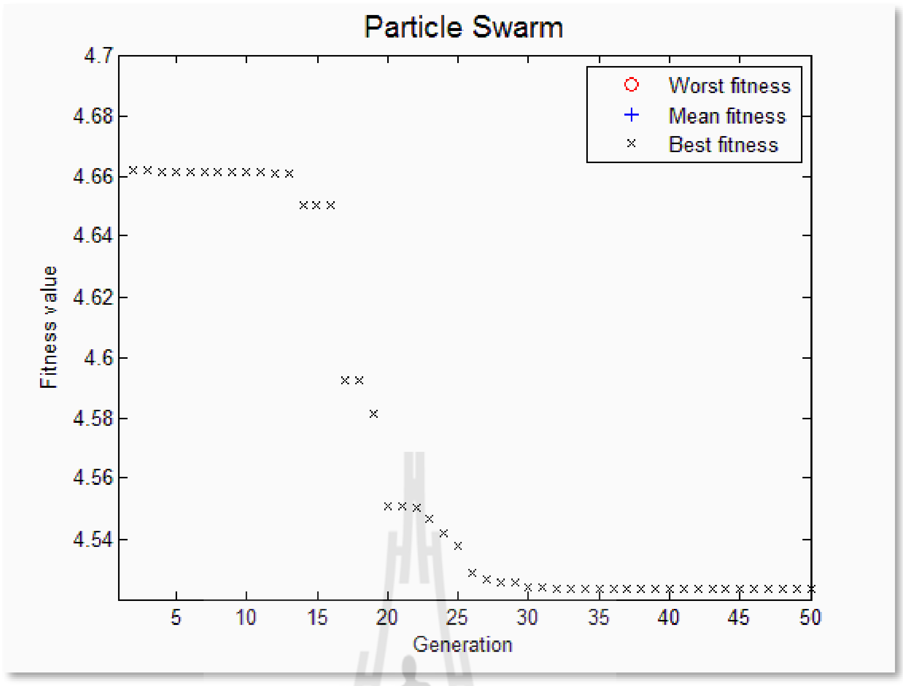
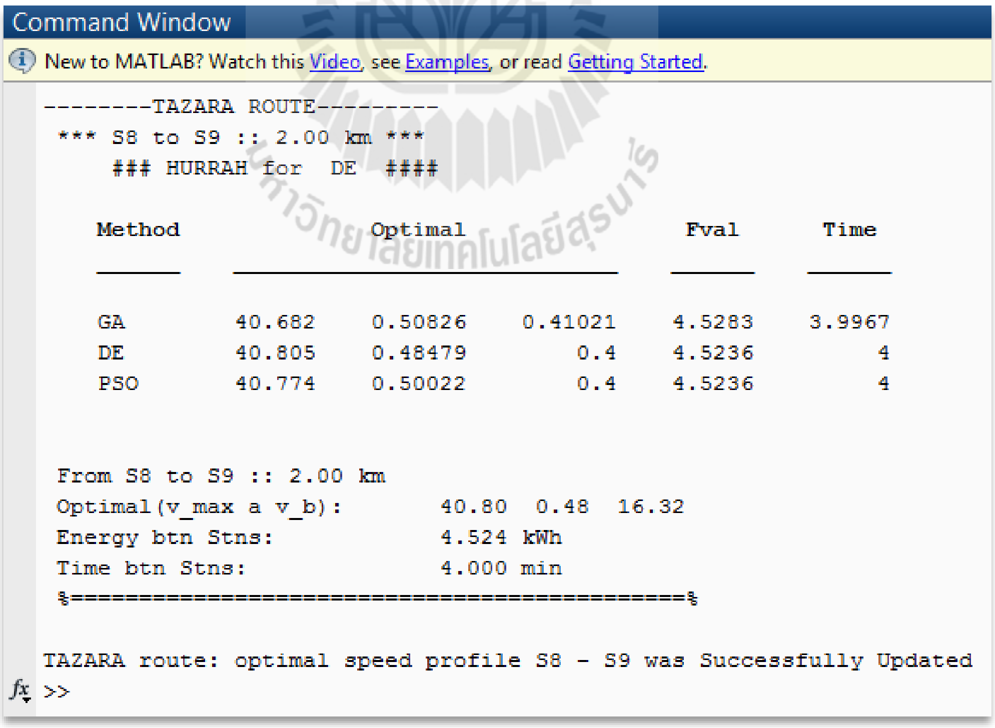


Figure C.101 TAZARA route, S8 to S9 PSO fitness value plot



Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
--------TAZARA ROUTE---------
 *** S8 to S9 :: 2.00 km ***
     ### HURRAH for  DE  ####

    Method         Optimal                     Fval      Time
    ______    ____________________________    ______    ______

    GA        40.682    0.50826    0.41021    4.5283    3.9967
    DE        40.805    0.48479        0.4    4.5236         4
    PSO       40.774    0.50022        0.4    4.5236         4


 From S8 to S9 :: 2.00 km
 Optimal(v_max a v_b):        40.80  0.48  16.32
 Energy btn Stns:             4.524 kWh
 Time btn Stns:               4.000 min
 %=============================================%

TAZARA route: optimal speed profile S8 - S9 was Successfully Updated
>>
```

Figure C.102 TAZARA route, S8 to S9 GA, DE and PSO optimal values

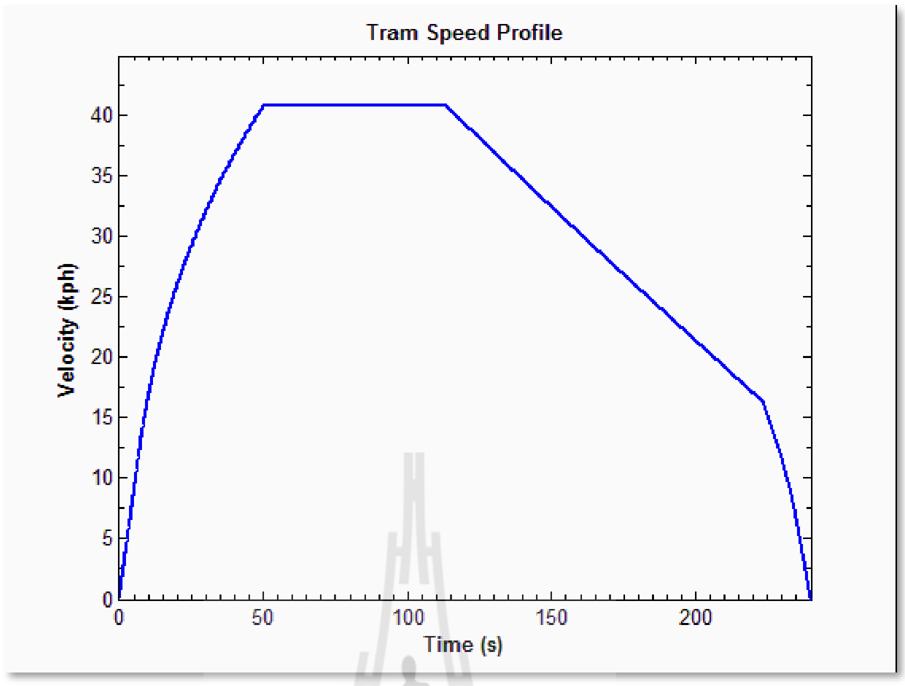


Figure C.103 TAZARA route, S8 to S9 the best speed profile, velocity against time

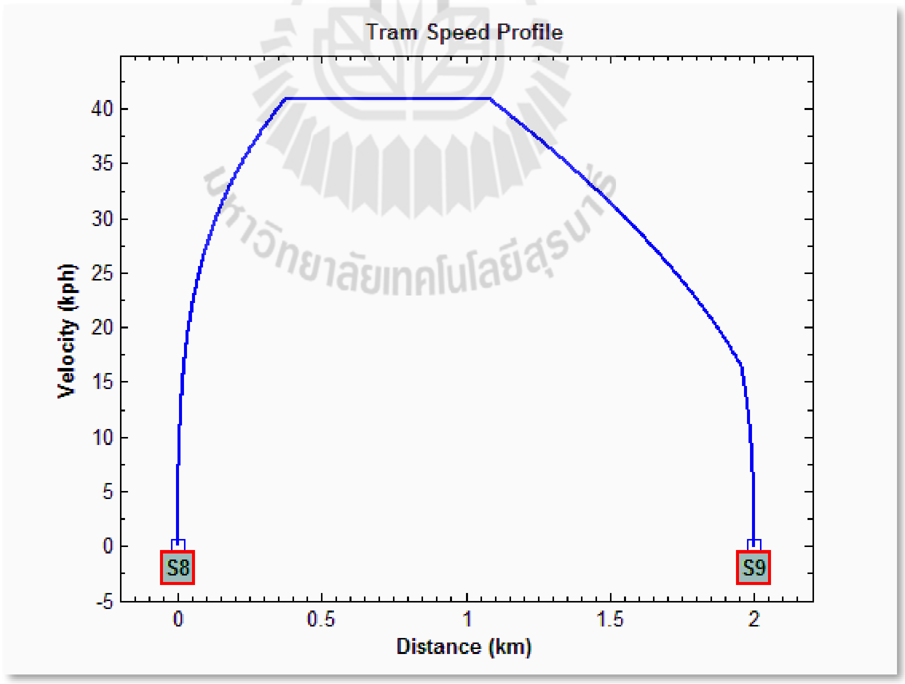


Figure C.104 TAZARA route, S8 to S9 the best speed profile, velocity against distance

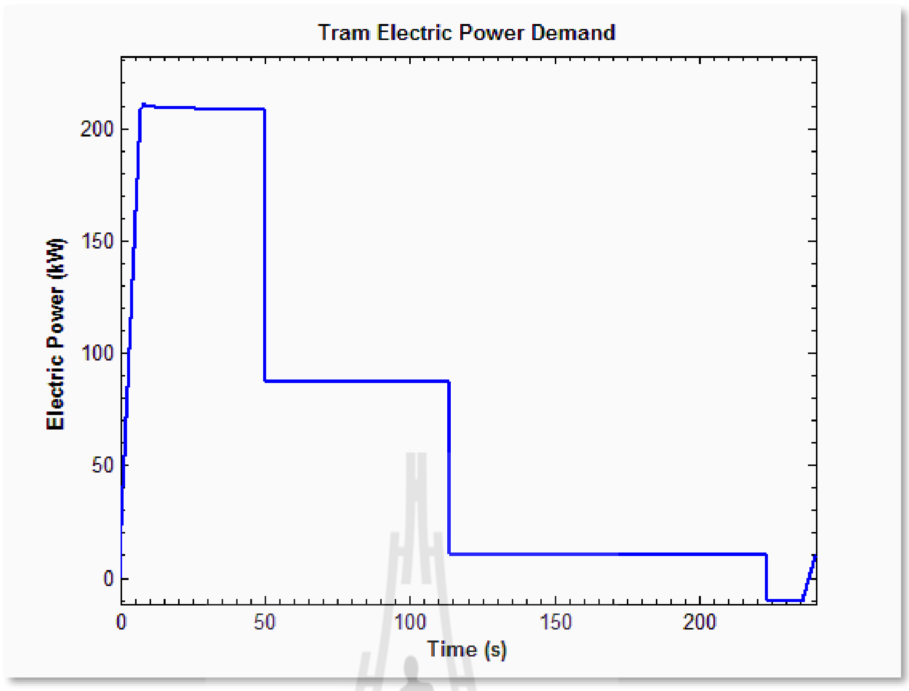


Figure C.105 TAZARA route, S8 to S9 power demand corresponding to the best speed profile

C.2.9 TAZARA Route; S9-S10

Speed profile optimization results for TAZARA route, from S9 to S10 were as presented in Figure C.106 through Figure C.112.

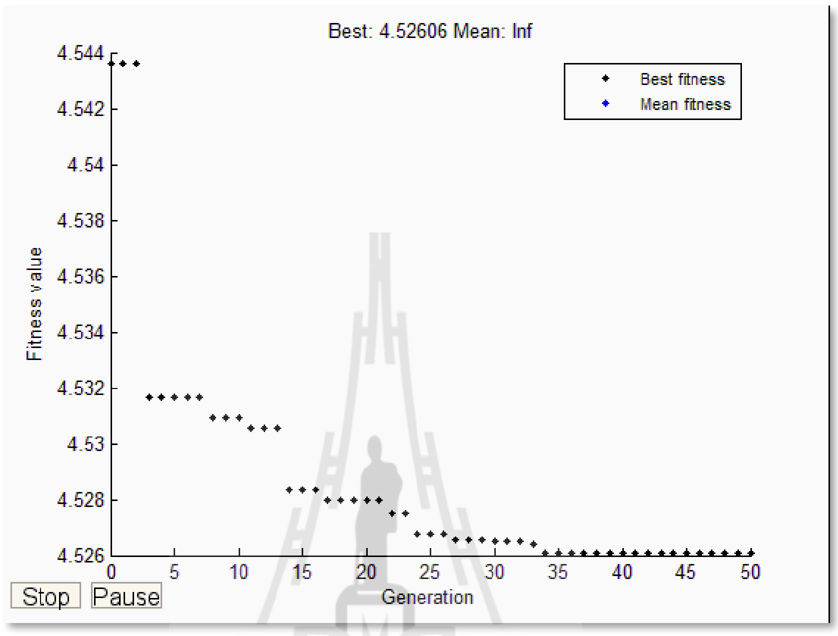


Figure C.106 TAZARA route, S9 to S10 GA fitness value plot

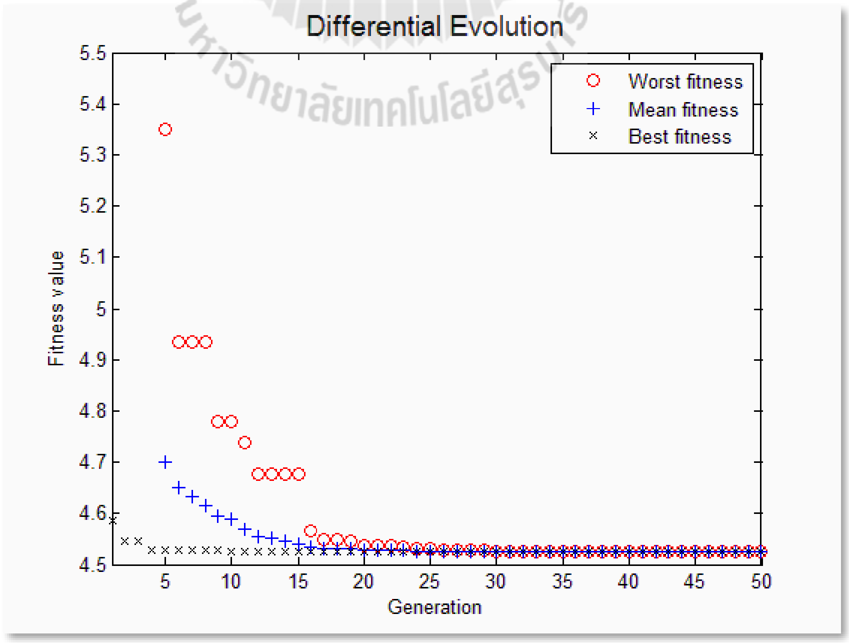


Figure C.107 TAZARA route, S9 to S10 DE fitness value plot

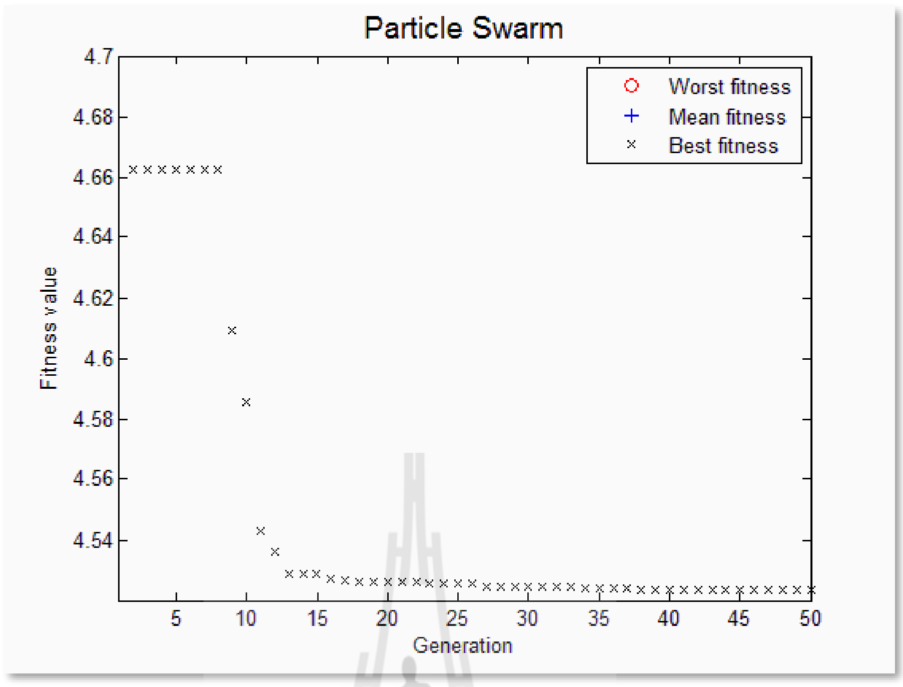
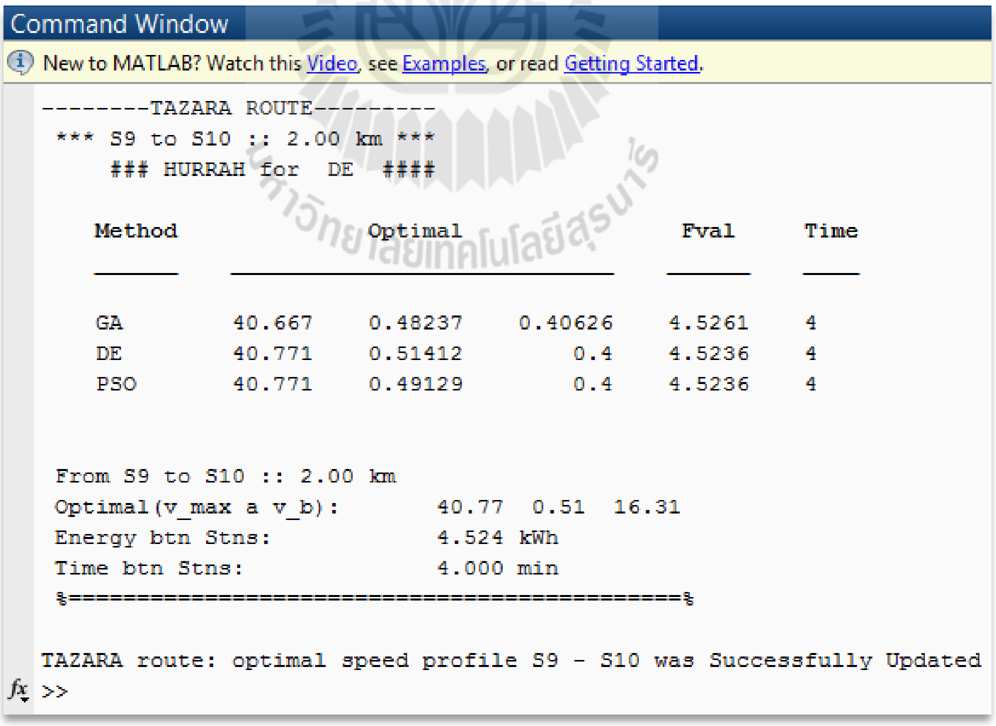


Figure C.108 TAZARA route, S9 to S10 PSO fitness value plot


Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
--------TAZARA ROUTE---------
 *** S9 to S10 :: 2.00 km ***
     ### HURRAH for  DE  ####

    Method          Optimal                     Fval      Time
    ______    ____________________________    ______    ____

    GA        40.667     0.48237    0.40626    4.5261     4
    DE        40.771     0.51412        0.4    4.5236     4
    PSO       40.771     0.49129        0.4    4.5236     4


 From S9 to S10 :: 2.00 km
 Optimal(v_max a v_b):        40.77  0.51  16.31
 Energy btn Stns:             4.524 kWh
 Time btn Stns:               4.000 min
 %===============================================%

TAZARA route: optimal speed profile S9 - S10 was Successfully Updated
>>
```

Figure C.109 TAZARA route, S9 to S10 GA, DE and PSO optimal values

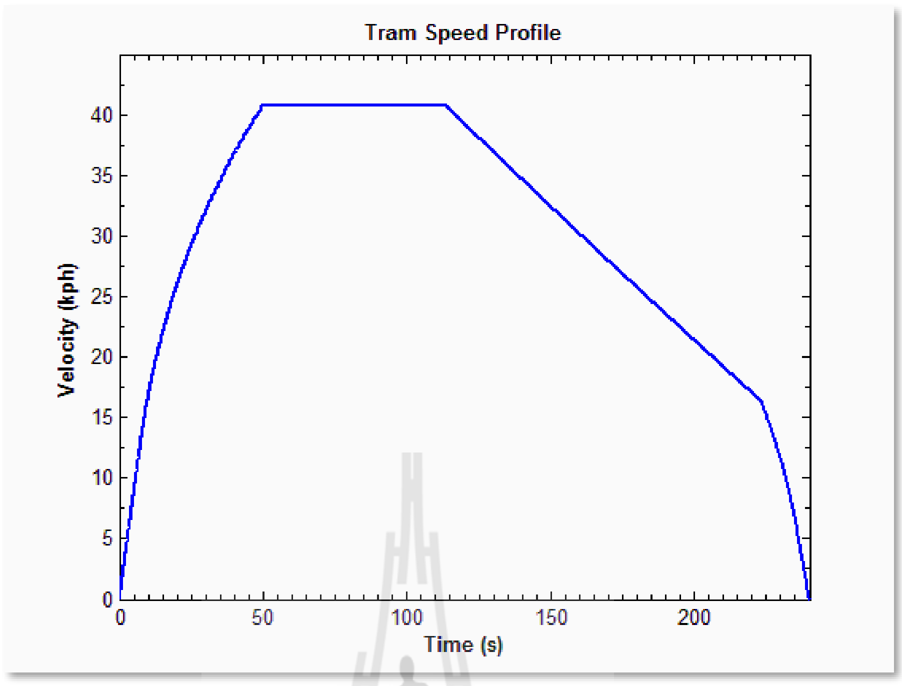


Figure C.110 TAZARA route, S9 to S10 the best speed profile, velocity against time

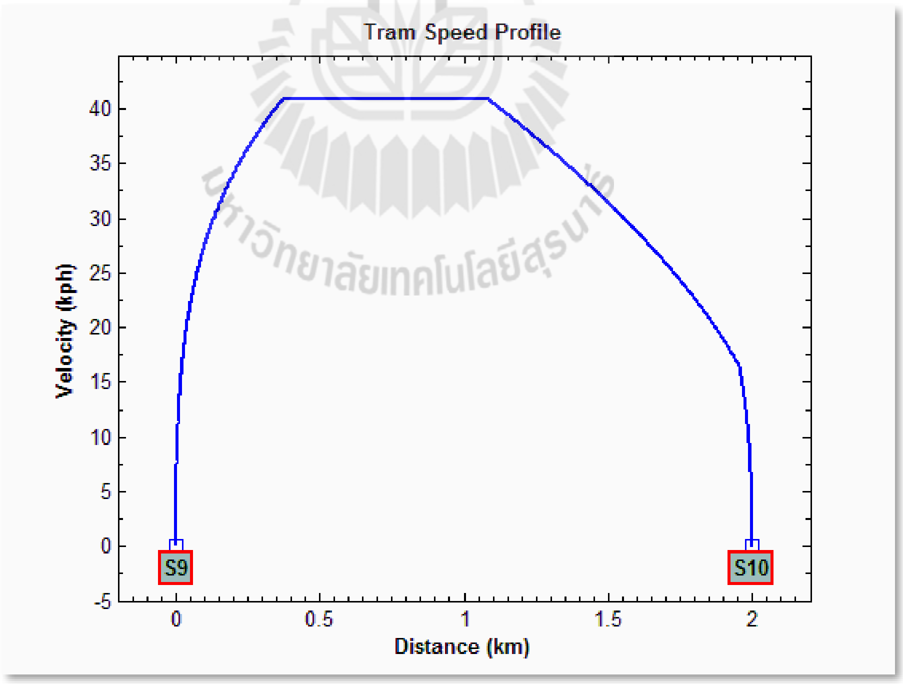


Figure C.111 TAZARA route, S9 to S10 the best speed profile, velocity against distance

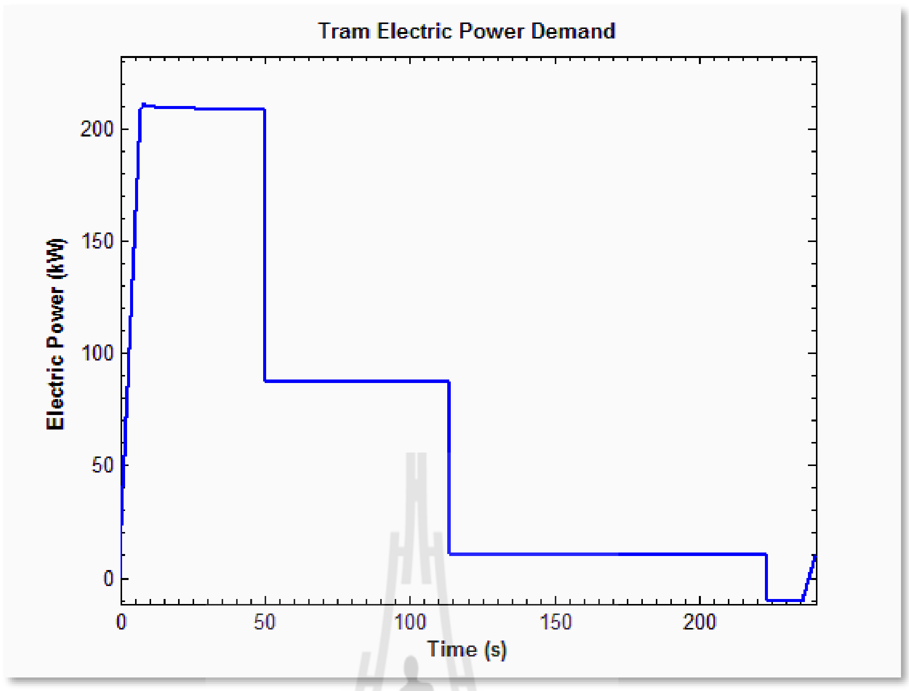


Figure C.112 TAZARA route, S9 to S10 power demand corresponding to the best speed profile

APPENDIX D

MATLAB CODE

D.1 Optimization

The MATLAB® scripts presented in this section were used to test the optimization algorithms and search for optimal speed profiles.

D.1.1 OptimizationTest.m

This script starts up the GUI from which the function to be used in the testing and the optimization algorithm (s) to be tested can be selected. The '***MainTest.m***' file is then called to proceed with the algorithm (s) testing.

```
function varargout = OptimizationTest(varargin)
% OPTIMIZATIONTEST MATLAB code for OptimizationTest.fig
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @OptimizationTest_OpeningFcn, ...
    'gui_OutputFcn',  @OptimizationTest_OutputFcn, ...
    'gui_LayoutFcn',  [] , ...
    'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

% --- Executes just before OptimizationTest is made visible.
function  OptimizationTest_OpeningFcn(hObject,  eventdata,  handles,
varargin)
% hObject    handle to figure

% Choose default command line output for OptimizationTest
handles.output = hObject;

axes(handles.axes_Optimize)     % Make current axes
imshow('A_photo_Optimize.png')
set(0, 'DefaultUIControlFontSize', 8);
% addpath([pwd '/DE_and_PSO_Testing']);
set(handles.radiobutton_All_3,'Value',1);
handles.Algorithm = 4;
set(handles.radiobutton_Ackley,'Value',1);
handles.Testfnc = 1;

% Update handles structure
guidata(hObject, handles);
```

```
% --- Outputs from this function are returned to the command line.
function varargout = OptimizationTest_OutputFcn(hObject, eventdata,
handles)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes when selected object is changed in uipanel_algorithms.
function uipanel_algorithms_SelectionChangeFcn(hObject, eventdata,
handles)

% disp(get(hObject,'Tag'));
AlgorithmTag = get(hObject,'Tag');
switch AlgorithmTag
    case 'radiobutton_GA'
        handles.Algorithm = 1;
    case 'radiobutton_DE'
        handles.Algorithm = 2;
    case 'radiobutton_PSO'
        handles.Algorithm = 3;
    case 'radiobutton_All_3'
        handles.Algorithm = 4;
end
% Update handles structure
guidata(hObject, handles);


% --- Executes when selected object is changed in uipanel_Testfcn.
function uipanel_Testfcn_SelectionChangeFcn(hObject, eventdata,
handles)

% disp(get(hObject,'Tag'));
TestfncTag = get(hObject,'Tag');
switch TestfncTag
    case 'radiobutton_Ackley'
        handles.Testfnc = 1;
    case 'radiobutton_Rastrigin'
        handles.Testfnc = 2;
    case 'radiobutton_Rosenbrock'
        handles.Testfnc = 3;
end
% Update handles structure
guidata(hObject, handles);


function pushbutton_Run_Callback(hObject, eventdata, handles)

Optm_Algothm2Test = handles.Algorithm;
Optm_Testingfnc = handles.Testfnc;
save ('VarStoreGui.mat', 'Optm_Algothm2Test','Optm_Testingfnc','-
append')
MainTest

% --- Executes on button press in pushbutton_Exit.
function pushbutton_Exit_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton_Exit (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
delete (gcf)
return
```

D.1.2 MainTest.m

This script calls the function to be used in the testing (that function may or may not be plotted, depending on the value of the Plotindex), it also calls the optimization algorithm(s) to be tested

```
%{
                  Written by Joachim J. Mwambeleko
                       *LU 25/Aug/2015*

 Testing the developed PSO and DE codes alongside the MATLAB built in
                                GA.

%}

clear all;clc; close all
global Obj
Plotindex = 2;
LV = load('VarStoreGui.mat', 'Optm_Algothm2Test','Optm_Testingfnc');
TestingFcn = LV.Optm_Testingfnc;
switch TestingFcn
    case 1
        Obj = @Ackley_fn; PlotAckley_fn(Plotindex)
    case 2
        Obj = @Rastrigin_fn; PlotRastrigin_fn(Plotindex)
    case 3
        Obj = @Rosenbrock_fn; PlotRosenbrock_fn(Plotindex)
end
nvars = 3;
LB = [-5 -5 -5];
UB = [10 10 10];

if length (LB)~= length (UB)
    error (' LB & UB should have equal length')
elseif nvars ~= length (LB)
    error (' nvars should be equal to the baundary size')
end
if min(UB-LB)<0
    error (' The UB is invalid, one or more UB(i) < LB(i)')
elseif min(UB-LB)== 0
    warning (' One or more UB(i) = LB(i)')
end
fprintf(['\nFunction Used in the Test: ' func2str(Obj),'\n\n'])

%------------ Select Which Algorithm to Test -----------------%
Optm_Algothm_2Test = LV.Optm_Algothm2Test;
switch Optm_Algothm_2Test
    case 1
```

```
        options = struct ;
        options = Options(options);
        np = options.PopSize; ng = options.Generations;
        opts = gaoptimset('Display','off','Generations',ng,...
            'PopulationSize',np,'PlotFcns',@gaplotbestf);

        [xop_ga,fop_ga,flag]                                           =
ga(Obj,nvars,[],[],[],[],LB,UB,[],opts);
        fprintf ('       Method:  GA \n'),...
        fprintf ('Optimal postn:  %2.3f %2.3f %2.3f \n',xop_ga);
        fprintf ('Fitness Value:  %4.4f \n\n',fop_ga);
        msg = deoutputmsg(flag);
        disp (msg)

    case 2
        [xop_dev, fop_dev, flag] = dev(Obj,nvars,LB,UB);
        fprintf ('       Method:  DE \n'),...
        fprintf ('Optimal postn:  %2.3f %2.3f %2.3f \n',xop_dev);
        fprintf ('Fitness Value:  %4.4f \n\n',fop_dev);
        msg = deoutputmsg(flag);
        disp (msg)

    case 3
        [xop_pso, fop_pso, flag] = pso(Obj,nvars,LB,UB);
        fprintf ('       Method:  PSO \n'),...
        fprintf ('Optimal postn:  %2.3f %2.3f %2.3f \n',xop_pso);
        fprintf ('Fitness Value:  %4.4f \n\n',fop_pso);
        msg = deoutputmsg(flag);
        disp (msg)

    case 4
        %----------------------------- GA ------------------
        options = struct ;
        options = Options(options);
        np = options.PopSize; ng = options.Generations;
        opts = gaoptimset('Display','off','Generations',ng,...
            'PopulationSize',np,'PlotFcns',@gaplotbestf);

        [xop_ga,fop_ga,flag]                                           =
ga(Obj,nvars,[],[],[],[],LB,UB,[],opts);
        msg = deoutputmsg(flag);
%         fprintf ('GA: %s \n',msg)

        %----------------------- DE --------------------------
        [xop_dev, fop_dev, flag] = dev(Obj,nvars,LB,UB);
        msg = deoutputmsg(flag);
%         fprintf ('DE: %s \n',msg)

        %----------------------- PSO ------------------------
        [xop_pso, fop_pso, flag] = pso(Obj,nvars,LB,UB);

        msg = deoutputmsg(flag);
%         fprintf ('PSO: %s \n',msg)

        %-----------Table Summary--------------%
        Summary   =   table(['GA   ';'DE   ';'PSO'],[xop_ga;   xop_dev;
xop_pso],...
            [fop_ga; fop_dev; fop_pso],...
            'VariableNames',{'Method' 'Optimals' 'Fval'});
        fprintf ('\n'); disp(Summary); fprintf ('\n');
end
```

Ackley_fn.m

This is the Ackley's function

```
function y = Ackley_fn(x)

%   min f(0,0,...n) = 0;  U can use n-varaibles
a = 20; b = 0.2; c = 2*pi;
s1 = 0; s2 = 0;
n = length(x);
for i=1:n;
    s1 = s1+x(i)^2;
    s2 = s2+cos(c*x(i));
end
y = -a*exp(-b*sqrt(1/n*s1))-exp(1/n*s2)+a+exp(1);
end
```

PlotAckley_fn.m

This function plots the Ackley's function in two dimensional space if the input is greater than one

```
function  PlotAckley_fn(Plotindex)

if Plotindex < 1
    return
end

xp= (-10:0.1:10); yp=xp;
[X,Y]=meshgrid(xp,yp);
[row,col]=size(X); Z = zeros (col,row);

for l=1:col
    for h=1:row
        Z(h,l)=Ackley([X(h,l),Y(h,l)]);
    end
end
figure (94)
surfc(X,Y,Z);
% shading interp
title('Ackley function','FontSize',14)
```

```
    function fx = Ackley(xyp)
        n = 2;
        a = 20; b = 0.2; c = 2*pi;
        s1 = 0; s2 = 0;
        for i=1:n;
            s1 = s1+xyp(i)^2;
            s2 = s2+cos(c*xyp(i));
        end
        fx = -a*exp(-b*sqrt(1/n*s1))-exp(1/n*s2)+a+exp(1);
    end
end
```

Rastrigin_fn.m

This is the Rastrigin function

```
function y = Rastrigin_fn(x)
%   min f(0,0,...n)  = 0;

n = length(x);
s1 = 0; a = 10;
for i = 1:n
    s1 = s1 + (x(i)^2 - a*cos(2*pi*x(i)));
end

y = a*n + s1;

end
```

PlotRastrigin_fn.m

This function plots the Rastrigin function in two dimensional space, if its input is greater than one.

```
function  PlotRastrigin_fn(Plotindex)
if Plotindex < 1
    return
end
xp= (-10:0.1:10); yp=xp;
[X,Y]=meshgrid(xp,yp);
[row,col]=size(X); Z = zeros (col,row);
for l=1:col
    for h=1:row
        Z(h,l)=Rastrigin([X(h,l),Y(h,l)]);
    end
end
figure (95)
surfc(X,Y,Z);
title('Rastrigin function','FontSize',14)
```

```
    function fx = Rastrigin(xyp)
        n = 2;
        s1 = 0;
        for i = 1:n
            s1 = s1 + (xyp(i)^2 - 10*cos(2*pi*xyp(i)));
        end
        fx = 10*n + s1;
    end
end
```

Rosenbrock_fn.m

This is the Rosenbrock function

```
function y = Rosenbrock_fn(x)

%   min f(a,a^2,...n) = 0;

n = length(x); a = 1; b = 100;
s1 = 0; s2 = 0;
for i = 1:n-1
    s1 = s1 + (a-x(i))^2;
    s2 = s2 + ((x(i+1)-x(i)^2))^2;
end

y = s1 + b*s2;

end
```

PlotRosenbrock_fn.m

This function plots the Rosenbrock function in two dimensional space, if its input is greater than one.

```
function  PlotRosenbrock_fn(Plotindex)

if Plotindex < 1
    return
end

xp= (-2:0.1:2); yp = (-1:0.1:3) ;
[X,Y]=meshgrid(xp,yp);
[row,col]=size(X); Z = zeros (col,row);

for l=1:col
    for h=1:row
```

```
        Z(h,l)= Rosenbrock([X(h,l),Y(h,l)]);
    end
end
figure (95)
surfc(X,Y,Z);
% shading interp
title('Rosenbrock function','FontSize',14)

    function fx = Rosenbrock(xyp)
        n = 2; a = 1; b = 100;
        s1 = 0; s2 = 0;
        for i = 1:n-1
            s1 = s1 + (a-xyp(i))^2;
            s2 = s2 + ((xyp(i+1)-xyp(i)^2))^2;
        end
          fx = s1 + b*s2;
         nd
end
```

dev.m

This is the developed DE algorithm script.

```
function [bestmem,bestfval,exitflag] = dev(Obj,nvars,LB,UB)
tic
%{
                  *Written by Joachim J. Mwambeleko*
                     ** Last Update 22/ Aug / 15 **
Comments are warmly welcome via (mwambejoachim@hotmail.com)
%}

if ~exist('LB','var'), LB = [] ; end
if ~exist('UB','var'), UB = [] ; end

if isempty(LB) || isempty(UB)
    error ('This programs needs finite LB and UB\n');
end

options = struct ;
options = Options(options);
np = options.PopSize; Cr = options.CrossoverFactor;
F = options.ScallingFactor;
if (np < 5)
    np = 5; options.PopSize = np;
    fprintf(['\n NP increased to minimal value ' num2str(np) '\n']);
end
if options.Generations < 10
    options.Generations = 10;
    options.StallGenLimit = 0.8 * options.Generations ;
    fprintf(['\n NG increased to minimal value ',...
        num2str(options.Generations) '\n']);
end
```

```
if ((Cr < 0) || (Cr > 1))
    Cr = 0.5;
    fprintf('\n Cr [0,1]; set to default: 0.5');
end
if ((F < 0) || (F > 1))
    F = 0.5;
    fprintf('\n Scalling-factor[0,1]; set to default: 0.7');
end

%-------------- Initiate the population & other Vectors -------------
-
[x, fx, ~, ~] = psointpop(nvars,LB,UB,options); % x: target vectors
v = zeros (np ,nvars);                %mutant vectors
u = zeros (np ,nvars);                %children vectors
x_nextg = zeros (np ,nvars);          %Initiate Next generation
fu = ones(np, 1)* inf;                %Initiate fit_val for children
ftval = ones(np, 1)* inf;        %Initiate fit_val for valid gen_members
idmut = zeros (1,3);                  % index for mutation [k,l,m]
[bestfval, idbestfval]= min(fx);
bestmem = x(idbestfval,:);
StallCounter = 0; exitflag = nan;

% --------------------------- Start Iterating ------------------
for ng = 1: options.Generations
    % Pick indexes for random vectors & generate mutant vector matrix
    for i = 1:np
        idmut(1)  = randi(np);
        while (idmut(1)==i)
            idmut(1)  = randi(np);              % get k ~= i
        end
        idmut(2)  = randi(np);
        while ((idmut(2)==idmut(1))||(idmut(2)==i))
            idmut(2)  = randi(np);              % get l ~= k, ~= i
        end
        idmut(3)  = randi(np);
        while
((idmut(3)==idmut(2))||(idmut(3)==idmut(1))||(idmut(3)==i))
            idmut(3)  = randi(np);              % get m ~= l, ~= k, ~= i
        end
        v(i,:)  = x(idmut(1),:)+ F*(x(idmut(2),:)-x(idmut(3),:));
    end %---> for i = 1:np

    %--------------------------------------------------------------
    % CrossOver the mutant vectors and the target vectors
    for i = 1:np
        for j = 1:nvars
            if rand ()<= Cr || j == randi(nvars)
                u(i,j)  = v(i,j);
            else
                u(i,j)  = x(i,j);
            end
            % Children (u) bnd_concstrain
            if u(i,j) > UB (j)
                u(i,j)  = UB (j);
            elseif u(i,j) < LB (j)
                u(i,j)  = LB (j);
            end
        end
    end
    %-----------------------------------------------------------
    % Select individuals from U and X for the next generation
    for i = 1:np
        fx(i,1) = Obj(x(i,:));
        fu(i,1) = Obj (u(i,:));
        if fu(i,1)<= fx(i,1)
            x_nextg(i,:) = u(i,:);
            ftval(i,1) = fu(i,1);
        else
            x_nextg(i,:)= x(i,:);
            ftval (i,1) = fx(i,1);
        end
    end
    ftvalworst = max(ftval);     % The worst value
    ftvalmean = mean(ftval);    % The mean value
    [ftvalmin, idbestmem] = min(ftval); % The best value
    if min(ftvalmin) <= bestfval            % That'll b obvious
        bestmem = x_nextg(idbestmem,:);
        change_bestfval = bestfval - ftvalmin;
        bestfval = ftvalmin;
    end
    if change_bestfval <= options.TolFun
        StallCounter = StallCounter + 1;
    else
        StallCounter = 0;     % Accumulate only if they are consecutive
    end
    %-------------------- Update the population --------------------
    x = x_nextg;
    %-------------------- Intermidiate Display ---------------------
    if strcmpi(options.DispInterval,'final')
        % Display nothing, wait for o/p msgs
    elseif isfinite(options.DispInterval) && options.DispInterval > 0
        if (rem(ng,options.DispInterval) == 0)
            fprintf('\n Gen: %d, Bestmem: [%s]', ng,num2str(bestmem))
            fprintf(' Bestfval: %f, F: %1.1f, Cr: %1.1f \n',...
                bestfval,F,Cr);
        end
    end
    %-------------------- Plot gfbest -------------------------
    if rem (ng,options.PlotInterval) == 0
        figure (90)
        plot
(ng,ftvalworst,'ro',ng,ftvalmean,'b+',ng,bestfval,'kx'),hold on
        title ('Differential Evolution','FontSize',14)
        xlabel ('Generation')
        ylabel ('Fitness value')
        xlim([1  options.Generations])
        hold on
    end
    % ---------------------- ExitFlags ------------------------
    if  StallCounter  >=  options.StallGenLimit,  exitflag  =  1;  break,
end
    timer = toc;
    if timer > options.TimeLimit, exitflag = -5; break, end
    if bestfval <= options.FitnessLimit, exitflag = 5; break, end
    if isnan (exitflag), exitflag = -1; end
    if ng == options.Generations, exitflag = 0; end

end %--> for ng = 1: options.Generations

%---------------------- If hybrid activated ----------------------
if options.Hybrid == 1
    X0 = bestmem;
    OPT = optimoptions('fmincon','Display', 'off');
    [X,FVAL] = fmincon(Obj,X0,[],[],[],[],LB,UB,[],OPT);
    if FVAL < bestfval
        untuned = bestfval;
        bestfval = FVAL;
        bestmem = X;
        tuned = bestfval;
        fprintf ('\nFine-tuned from %7.9f to %7.9f',untuned,tuned)
        plot (ng,bestfval,'*c','LineWidth',6)
    else
        fprintf  ('\nHybrid  was  activated  but,  there  was  no
improvement\n')
    end
end %---> if options.Hybrid == 1
legend('Worst fitness','Mean fitness','Best fitness')
hold off  % At the end of plotting you would like to HOLD OFF
return
```

pso.m

This is the developed PSO algorithm script which was used in speed profile optimization.

```
function [gbest,gfbest,exitflag] = pso(Obj,nvars,LB,UB)
tic
%{
             *written by Joachim J. Mwambeleko*
                  ** LU 17/ Aug / 15 **
         ** Appreciations to Dr T.Kulworawanichpong **
NOTE:
Given that w = particle inertia
           C1 = SelfAttraction
           C2 = SolcialAttraction
   1) 0 < (C1 + C2) < 4
   2) ((C1 + C2)/2) - 1 < w < 1
If  conditions  1  and  2  are  satisfied,  the  pso  will  have  a  better
chance to converge.
%}

if ~exist('LB','var'), LB = [] ; end
if ~exist('UB','var'), UB = [] ; end

if isempty(LB) || isempty(UB)
    error ('Sorry this programs needs finite LB and UB\n');
end
```

```
options = struct ;
options = Options(options);  % Load options.

if options.PopSize < 5
    options.PopSize = 5;
    fprintf(['\n NP increased to minimal value ',...
        num2str(options.PopSize) '\n']);
end
if options.Generations < 10
    options.Generations = 10;
    options.StallGenLimit = 0.8 * options.Generations ;
    fprintf(['\n NG increased to minimal value ',...
        num2str(options.Generations) '\n']);
end

% --------------------- Check swarm stability ----------------
w = options.Inertia;
C1 = options.SelfAttractin; C2 = options.SocialAttraction;
if (C1 + C2 >= 4 || C1 + C2 <= 0)
    msg = 'The swarm may not converge: ' ;
    msg = [msg 'The condition "0 < (C1 + C2) < 4" was not met'] ;
    warning('%s :Consider adjusting C1 and/or C2',msg)
end

if ( w >= 1 || ((C1 + C2)/2) - 1 >= w )
    msg = 'The Swarm may not converge: ' ;
    msg = [msg 'The condition "((C1 + C2)/2) - 1 < w < 1" was not
met'];
    warning('%s: Consider adjusting w or C1 & C2',msg)
end
%---------------- Decide on the method -------------------------
LDIW_a = 1; LDIW_b = 1; DVlim = 0;
ImproveConvergency = 1;
if DVlim == 1
    LDIW_a = 0; LDIW_b = 0;
elseif (LDIW_a == 1 || LDIW_b == 1)
    DVlim = 0;
    j_max = options.Generations;
    w_initial = 0.9; w_final = 0.4;
    w = w_initial;
end
%---------------- Initiate the population ------------------------
[x, lfbest, gfbest, gbest] = psointpop(nvars,LB,UB,options);

StallCounter = 0; exitflag = nan;
%---------- Move particles checking Bounds & V_limit -----------
lbest = x;
fx = lfbest;
np = options.PopSize;
v = zeros(np,nvars);

for ng = 1: options.Generations
    R = rand(1,2);
    GBEST = ones(np,1)*gbest;
```

```
%---------Update velocities take all particles at once----------
    % v_update option 1
    v = w*v + C1*R(1)*(lbest-x)+ C2*R(2)*(GBEST-x);
    %{
$$$ Improvement by Li & Shi "An IPSO algorithm for pattern
synthesis of phased arrays" r1 & r2 may all be too low or
too high, so take only r1 randomly, and substitute r2 by (1- r1)
    %}
    % v_update option_2
    %      v = w*v + C1*R(1)*(lbest-x)+ C2*(1-R(1))*(GBEST-x);

    %--------- Check velocities then update position ---------------
    vl = options.VelocityLimit;
    v(v > vl)= vl;
    v(v < -vl) = -vl;
    %disp(v)
    x_prv = x;
    x = x + v;
    %-----------------   Check the xs agaist UB & LB ------------
    for col = 1:nvars
        x (:,col)= min(x(:,col),UB(col));  %### NOTE: min->UB
        x (:,col)= max(x(:,col),LB(col));  % max->LB
    end
    %disp(x)
    %----------------------- Get the scores --------------------
    fx_prv = fx;
    for p = 1:np
        fx (p,1) = Obj(x(p,:));

        %-------- Improvement update to better positions ------
        if ImproveConvergency ==1;
            if fx(p,1) > lfbest(p,1)
                maxrepeat = 1;
                for repeat = 1:maxrepeat
                    % x(p,:) = rand*rand* x(p,:);      % mutate
                    x(p,:) = rand*x(p,:);            % mutate
                    for col = 1:nvars
                        x (p,col)= min(x(p,col),UB(col));  % UB & LB
                        x (p,col)= max(x(p,col),LB(col));
                    end

                    %_#  Altenatively just through the particle into
                    % the sln space
                    % x(p,:)= rand*(UB-LB)+ LB;

                    fx (p,1) = Obj(x(p,:));          % re-calculate

                    if fx(p,1) < lfbest(p,1)
                        break
                    elseif repeat == maxrepeat
                        if rand > rand*(1-ng/options.Generations)
                            x(p,:) =  x_prv (p,:); % Return to...
                                                   % prvoius location
                            fx (p,1) = fx_prv(p,1);
                        end
                    end
                end
            end
        end
        %------------ Check if you can update lbest ------------

        if fx(p,1) < lfbest(p,1)
            lfbest(p,1) = fx(p,1);
            lbest(p,:) = x(p,:);
        end
        fxmean = mean(fx);
        fxworst = max(fx);
    end
    %---------------------- Get gbest --------------------------
    [gfbest_new, idgfbest] = min(lfbest);
    change_gfbest = gfbest - gfbest_new ;
    gfbest = gfbest_new;
    gbest = lbest(idgfbest,:);
%     gbest = x(idgfbest,:);

    if change_gfbest <= options.TolFun
        StallCounter = StallCounter + 1;
    else
        StallCounter = 0;    % Accumulate only if they are consecutive
    end

    %-------------------reduce vl as ng increases---------------
    %_# Altenatively one could reduce the inertia weight linearly
    if DVlim == 1
        if ng == round (0.5*options.Generations)
            options.VelocityLimit = 0.8 * options.VelocityLimit;
        elseif ng == round (0.7*options.Generations)
            options.VelocityLimit = 0.8 * options.VelocityLimit;
        elseif ng == round (0.9*options.Generations)
            options.VelocityLimit = 0.8 * options.VelocityLimit;
        end
    end
    %------------reduce ineria weight w as ng increases-----------
    %_# Altenatively one could reduce velocity
    if (LDIW_a == 1 || LDIW_b == 1)
        j = ng;     % Update j
        if LDIW_a == 1
            w_j= w_initial-(w_initial-w_final )*j/j_max ;
        elseif LDIW_b == 1
            w_j= (w_initial-w_final )*(j_max-j)/j_max + w_final;
        end
        w = w_j;  % Update w
    end
    %--------------- Intermidiate Display -----------------------
    if strcmpi(options.DispInterval,'final')
        % Display nothing, wait for o/p msgs
    elseif isfinite(options.DispInterval) && options.DispInterval > 0
        if (rem(ng,options.DispInterval) == 0)
            fprintf('\n Gen: %d, Bestmem: [%s]', ng,num2str(gbest))
            fprintf('Bestfval:  %f,  vel_lim:  %1.1f,  Inertia:  %1.1f 
\n',...
                gfbest,options.VelocityLimit,options.Inertia);
        end
    end
    %------------------- Plot gfbest ----------------------------
    if rem (ng,options.PlotInterval) == 0
        figure (91) %('Name','Particle Swarm','NumberTitle','off')
        plot (ng,fxworst,'ro',ng,fxmean,'b+',ng,gfbest,'kx')
        title ('Particle Swarm','FontSize',14)
        xlabel ('Generation')
        ylabel ('Fitness value')
        xlim([1  options.Generations])  % x-axis  limit.
        hold on
    end
    % ---------------------  ExitFlags ---------------------------
    if  StallCounter  >=  options.StallGenLimit,  exitflag  =  1;  break,
end
    timer = toc;
    if timer > options.TimeLimit, exitflag = -5; break, end
    if gfbest <= options.FitnessLimit, exitflag = 5; break, end
    if isnan (exitflag), exitflag = -1; end
    if ng == options.Generations, exitflag = 0; end

end %--->g = 1: options.Generaions

%---------------------- If hybrid activated ----------------------
if options.Hybrid == 1
    X0 = gbest;
    OPT = optimoptions('fmincon','Display', 'off');
    [X,FVAL] = fmincon(Obj,X0,[],[],[],[],LB,UB,[],OPT);
    if FVAL < gfbest
        untuned = gfbest;
        gfbest = FVAL;
        gbest = X;
        tuned = gfbest;
        fprintf ('\nFine-tuned from %7.7f to %7.7f',untuned,tuned)
        plot (ng,gfbest,'*c','LineWidth',6)
    else
        fprintf   ('\nHybrid   was   activated   but,   there   was   no
improvement')
    end
end %---> options.Hybrid == 1
legend('Worst fitness','Mean fitness','Best fitness')
hold off  % At the end of plotting you would like to HOLD OFF
return
```

psointpop.m

This function randomly initializes the population (for both DE and PSO) within the given solution space

```
function [x, lfbest, gfbest, gbest] = psointpop(nvars,LB,UB,options)
%{
                  *written by Jojo Pidosa Joachim*
                      ** LU 17/ Aug / 15 **
%}
%---------------------- Initialize particles -----------------------
global Obj
```

```
np = options.PopSize;
x = zeros (np,nvars);           % Initialize memory
lfbest = ones(np,1)*inf;        % Column vector, score of particles
for i = 1:np
    %_# x = rand * (UB-LB)+ LB
    x(i,:)= rand *(UB-LB)+ LB;          % x(i,:), raw vector,1_particle
    lfbest(i,1)= Obj(x(i,:));
end

[gfbest, idgfbest] = min(lfbest);
gbest = x(idgfbest,:);

end
```

deoutputmsg.m

This function gives the reason as to why the argorithm (DE or PSO) stopped

```
function msg = deoutputmsg(exitflag)

if exitflag == 0
    msg = sprintf('Generation limit reached.') ;
elseif exitflag == 1
    msg = sprintf('Change in value of the fitness function over') ;
    msg   =   sprintf('%s   options.StallGenLimit   generations   less
than',msg);
    msg = sprintf('%s options.TolFun.',msg) ;

elseif exitflag == 5
    msg = sprintf('Fitness limit reached.');

elseif exitflag == -1
    msg = sprintf('Optimization Interupted.') ;

elseif exitflag == -5
    msg = sprintf('Time limit reached.') ;
else
    msg = sprintf('Unrecognized exitflag value') ;
end % if exitflag
```

Options.m

This function sets the options for the optimization algorithms

```
function options = Options(varargin)
%{
                  Written by Jojo Pidosa Joachim
                * Last Update 15/DEC/2015*

    Any Method will take all the Option and use those it needs
%}
% For all the three algorithms
options.PopSize = 50;                                 % np
options.Generations = 50 ;                             % ng

% For DEV & PSO
options.PlotInterval = 1 ;
options.FitnessLimit = -inf ;
options.DispInterval = 'final' ;
% options.DispInterval = 10 ;
 options.StallGenLimit = 0.8 * options.Generations ;
options.TimeLimit = 5*60 ;
options.TolFun = 1e-6 ;
options.Hybrid = 0;

% Specific for DEV
options.ScallingFactor = 0.8;                      % F
options.CrossoverFactor = 0.8 ;                    % Cr

% Specific for PSO
options.Inertia = 0.6;                             % w (If it's constant)
options.Inertia_initial = 0.9;                     % w (for LDIW-PSO)
options.Inertia_final = 0.4;                       % w (for LDIW-PSO)
options.SelfAttractin = 1.75 ;                      % C1
options.SocialAttraction = 1.0 ;                   % C2
options.VelocityLimit = 10 ;                       % vl

end
```

Main_Optimization.m

This script specifies the route (TRL or TAZARA route) and starts speed profile optimization.

```
%{
                 **Written by Joachim Mwambeleko**
                     *Last update 27/DEC/2015*
%}
clear all;clc; close all; echo off
global SaveStationaly
global i indexP Obj route staname stapos Nstops s_stop  TimeCheck
global  GAXL PSOXL DEVXL XOPXL Method
SaveStationaly = 1;
Obj = @EMU_Compute;
nvars = 3;

%% PERMANENT CONSTANTS
% Train parameters used in optimization
global m_optmzn fc efc Cdf dt g Te_max Tb_max dec_max t_max
m_optmzn  = (38.8e3 * 1.11)+ (60.7 * 312) + (27.4 * 54);

fc = 0.006;                % Rolling friction coefficient
Cd = 0.6;                  % Drag coefficient
Af = 7.7; Ad = 1.225;      % Frontal area (m^2); Air density (kg/m^3)
```

```
efc = 0.6;                % Power conversion efficiency
g  = 9.81;  dt = 0.1;
Cdf = 0.5*Ad*Cd*Af;       % Fdrag_Coefficient
Pe_max = 8*45e3;  v_base_max = 0.45 * (50/3.6); %(6 m/s2)
Te_max = (Pe_max * efc)/v_base_max; Tb_max = Te_max;
acc_max = (Te_max - (m_optmzn*fc))/ m_optmzn ;   dec_max = acc_max;
%% ROUTES - Initialize a route
set(0, 'DefaultUIControlFontSize', 12.5);
Routing = questdlg({'Welcome to Optimization:',...
    'Which route are you going for?'}, ...
    'SPECIFY A ROUTE ','TRL','TAZARA','TRL');
switch Routing
    case 'TRL'    ;    route = 1;
    case 'TAZARA' ;    route = 2;
end

[x_Optimal]= InitializeRoute();
if isempty (x_Optimal) % No route was chosen
    fprintf (' Sorry !\n The route was not specified')
    fprintf (' Please re-run the program\n')
    return
end

s_stop   =   zeros(1,Nstops);   Method   =   cell(1,Nstops);   xop   =
zeros(Nstops,3);
GAXL = zeros (6,Nstops); PSOXL = GAXL; DEVXL = GAXL; XOPXL =  GAXL;

indexP = 1;
if SaveStationaly == 1;
    Nstops = indexP;            % To make only one loop in for
end
%Nstops = 1;
%%
for i = indexP:Nstops
    s_stop (i) = stapos (i+1)-stapos(i);

    v_max = 50;
    LV = load ('VarStore','T_max1','T_max2');
    t_max = LV.T_max1;       % Max time (minute) per km
    if (s_stop (i)/1e3) <= 1 ;
        t_max = LV.T_max2;
    end
    v_min = (s_stop (i)/(t_max*60*s_stop (i)*1e-3))*3.6;
    % v_max(km/h), acc, b:
    LB = [v_min   0.3        0.4];
    UB = [v_max   acc_max     1];
    fprintf([' *** ',staname{i},' to ',staname{i+1},...
        ' :: %2.2f km ***\n'],(s_stop(i)/1e3))

    % -------------- Check varaibles --------------
    if length (LB)~= length (UB)
        error (' LB & UB should have equal length')
    elseif nvars ~= length (LB)
        error (' nvars should be equal to the baundary size')
    end
    if min(UB-LB)<0
        error (' The UB is invalid, one or more UB(i) < LB(i)')
```

```
elseif min(UB-LB)== 0
    warning (' One or more UB(i)  = LB(i)')
end
%-------------------------------------------------

options = struct ;
options = Options(options);
np = options.PopSize; ng = options.Generations;
opts = gaoptimset('Display','off','Generations',ng,...
    'PopulationSize',np,'PlotFcns',@gaplotbestf);

[xop_ga,fop_ga] = ga(Obj,nvars,[],[],[],[],LB,UB,[],opts);
fop_ga2 = EMU_Compute(xop_ga); % To assign TimeCheck
if abs(fop_ga-fop_ga2)/fop_ga > 1e-2
    %     warning ('fop_ga ~= fop_ga2'); % Just for curiosity
end
Time_ga = TimeCheck; % Get the time & prepare xlx DATA
GAXL(:,i) = [s_stop(i)/1e3; xop_ga(1);xop_ga(2);xop_ga(3);...
    fop_ga;Time_ga];

[xop_dev, fop_dev] = dev(Obj,nvars,LB,UB);
fop_dev2 = EMU_Compute(xop_dev);
if abs(fop_dev-fop_dev2)/fop_dev > 1e-2
    %     warning ('fop_dev ~= fop_dev2');
end
Time_dev = TimeCheck;
DEVXL(:,i) = [s_stop(i)/1e3; xop_dev(1);xop_dev(2);...
    xop_dev(3);fop_dev;Time_dev];

[xop_pso,fop_pso] = pso(Obj,nvars,LB,UB);
fop_pso2 = EMU_Compute(xop_pso);
if abs(fop_pso-fop_pso2)/fop_pso > 1e-2
    %     warning ('fop_pso ~= fop_pso2');
end
Time_pso = TimeCheck;
PSOXL(:,i) = [s_stop(i)/1e3; xop_pso(1);xop_pso(2);...
    xop_pso(3);fop_pso;Time_pso];

[Best, IndexBest] = min ([fop_ga, fop_dev, fop_pso]);

if fop_dev == fop_pso
    if rand > 0.5
        Best = fop_pso; IndexBest = 3;
    end
end

if IndexBest == 1
    fprintf ('     ### HURRAH for  GA  ####\n')
    xop(i,:) = xop_ga;
    XOPXL(:,i) = GAXL(:,i); Method(i) = {'GA'};
elseif IndexBest == 2
    fprintf ('     ### HURRAH for  DE  ####\n')
    xop(i,:) = xop_dev;
    XOPXL(:,i) = DEVXL(:,i); Method(i) = {'DE'};
elseif IndexBest == 3
    fprintf ('     ### HURRAH for  PSO  ####\n')
    xop(i,:) = xop_pso;
```

```
            XOPXL(:,i) = PSOXL(:,i); Method(i) = {'PSO'};
        else
            fprintf ('Soory Choose the BEST RESULT by yourself\n')
            warning ('xop has not been assigned\n')
        end

        %-----------Table Summary--------------%
        Summary    =    table(['GA    ';'DE    ';'PSO'],[xop_ga;    xop_dev;
    xop_pso],...
            [fop_ga; fop_dev; fop_pso],[Time_ga; Time_dev; Time_pso],...
            'VariableNames',{'Method' 'Optimal' 'Fval' 'Time'});
        fprintf ('\n'); disp(Summary);

        %---------If save after every stn --------%
        if SaveStationaly == 1
            SaveOptimalStationaly (i,route,xop)
        end
    end

    if SaveStationaly == 1;
        % Calling initialize route will write the
        % same data for every interstation. So, return
        return;
    end
    %% ----- Plot the optimal speed prifile----------%
    % xop = load('TZR_Speed_Profile.txt');
    EMU_Compute(xop,'Plotoptimal');
    % % pause
    %---------- Update the  chooseRoute fnc --------%
    if Nstops == Nstops+1-indexP;
        set(0, 'DefaultUIControlFontSize', 12.5);
        msgbox({'Review   the   Speed   Profile   and','Press   any   key   to
    continue'},...
            'Review Speed profile');
        pause
        if route == 1
            Overwrite = questdlg({'Do you want to Update',...
                'Optimal Speed Profile for TRL Route?'}, ...
                'TRL: Speed Profile','YES','NO','NO');
            switch Overwrite
                case 'YES'
                    xop_trl = xop;
                    InitializeRoute(1,'TRL',xop_trl);
            end

        elseif route == 2
            Overwrite = questdlg({'Do you want to Update',...
                'Optimal Speed Profile for TAZARA Route?'}, ...
                'TAZARA: Speed Profile','YES','NO','NO');
            switch Overwrite
                case 'YES'
                    xop_tzr = xop;
                    InitializeRoute(1,'TZR',xop_tzr);
            end
        end % ---> If route == 1
    end % ---> if Nstops == length(stapos)-1
```

EMU_Compute.m

This function simulates train movement. It was also used as objective function during speed profile optimization.

```
function [y] = EMU_Compute(x,varargin)
%{
                 **Written by Joachim J Mwambeleko**
                      *Last update 20/Jan/2015*
NOTE:
   x is an Array When the fnc is called for Optimization, a Matrix
when
   the fnc is called for Compute OR Plotoptimal.
   Vararging should tell why calling, if it's Empty, called for
optimization
The Flow
1. Determine acc to move the train (Using current 4Cs))
2. Increment time (Implying the train has moved)
2. Update distance {Use previous v}:i.e. s = s + ds;
   ds = v*t + 0.5*a*dt^2
3. Update velocity: i.e v = v + a*dt
4. Calculate power
%}
if isempty (varargin)
    varargin = 'Objfnc' ;       % Sets the func 2b used as Obj fnc
end

global i indexP route stapos Nstops staname dec_max  Tramov
global TimeCheck t_max CallAnimation Batt SaveStationaly

%------------------- PERMANENT CONSTANTS---------------------
global m_optmzn m_compute fc Cdf      % mass, Frr_Coef, Fdrad_Coef
global dt efc g Te_max Tb_max

y = [];
if strcmp(varargin,'Compute')
    m = m_compute;
    N_STOPS = Nstops;
    indexI = 1;             % Start for loop from i = 1 to Nstops
    stt = 60;               % Station-stopping (Dwell) time (sec)
    bp_ah  = Batt.Pack.Ah;
    bp_c = 0;  bp_v = Batt.Pack.V_int;
    bp_soc = Batt.Pack.soc_int;
    bp_eng = Batt.Pack.E_int;
    bp_esr = Batt.Pack.ESR;

    % ------ For Commulative Storage (C,V & SOC) --------
    BP_V = bp_v ; BP_SOC = bp_soc;  BP_C = bp_c;
    powfm_bp = 0; powfm_cat = 0;
    Powfrm_BP = powfm_bp;  Powfm_CAT = powfm_cat;
elseif strcmp(varargin,'Plotoptimal')
    m = m_optmzn;
    N_STOPS =  Nstops;
    indexI = indexP;
    stt = -inf;                        % Not to Include Dwell time
elseif strcmp(varargin,'Objfnc')
    m = m_optmzn;
    indexI = i;                     % IndexI keeps gloal value of i
    N_STOPS = indexI;               % Avoid for loop
    stt = -inf;
end
%-------------------------- INITIALIZATION --------------------
hr = 3600;        % 1hr = 3600 sec
s_stop = zeros(1,N_STOPS); t_bs =  s_stop; E_bs =  s_stop;
mode =  s_stop;  t_Com =  s_stop;  E_Com =  s_stop;

t = 0; t_rsv = 0; s = 0; v = 0;
Te = 0;  Pe=0 ; Ee = 0;
Frr = fc*m*g;                              % Remains constant
Fdrag = 0; Fgrad = 0;
Fr = Frr + Fdrag + Fgrad; Pm = 0;

%-------------------- INITIATE GENERAL STORAGE ---------------
T = t;  V = v;  S = s; TE = Te;
PE = Pe;  EE = Ee; % FR = Fr; PM = Pm;
%--------------- INITIALIZE  IN EVERY ITERATION -------------
for i = indexI:N_STOPS
    Apu = 10;           % Aux load (kW),Set to Zero after 1 min D-time
    v = 0; s_trav = 0;
    s_stop (i) = stapos (i+1)-stapos(i);
    disTonext = s_stop(i);
    if strcmp(varargin,'Compute')
        LV = load ('VarStore','T_max1','T_max2');
        t_max = LV.T_max1;         % Max time (minute) per km
        if (s_stop (i)/1e3) <= 1 ;
            t_max = LV.T_max2;
        end
        mtt = t_max *60 * s_stop(i)/1e3;  % max.travel timein in Sec
        mtt = floor(mtt/dt); mtt = mtt*dt;
    end
    if strcmp(varargin,'Compute')||strcmp(varargin,'Plotoptimal')
        v_max = x(i,1)/3.6; v_base = 0.3*v_max;
        acc = x(i,2);
        v_b = x(i,3)*v_max;  v_base_b = 0.45*v_b;
    elseif strcmp(varargin,'Objfnc')
        v_max = x(1)/3.6;   v_base = 0.3*v_max;
        acc = x(2);
        v_b = x(3)*v_max; v_base_b = 0.45*v_b;
    end

    Frr = fc*m*g;
    Fdrag = Cdf*v^2; Fgrad = 0;
    Fr = Frr + Fdrag + Fgrad;
    Te = Frr + m*acc;              % Initial Te
    if Te >= Te_max
        Te = Te_max;
        a = (Te - Frr)/m;
    else
        a = acc;
```

```
end
%----------------------- GET THE UNKNOWNS ------------------
% S_1 : Determined by a & v_max
% S_3 : Determined by v_max & v_b
% S_4 : Determined by v_b ; as a result
% S_2 : Determined by S_1,S_3,& S_4

[S_4,  ~] = EMU_Compute_S_4(v_b, v_base_b, m);
[S_3, ~, ~] = EMU_Compute_S_3(v_max, v_b, m);
[S_1, ~, ~] = EMU_Compute_S_1(acc, v_max ,v_base, m);

if (strcmp(varargin,'Objfnc')&&...
        ((S_1 + S_4 + S_3)- s_stop(i))> 1e-3)
    y = inf; % Invalid, discard
    return
end
S_2 = s_stop(i)-(S_1 + S_3 + S_4);

%%----------------------- CONTROLLERS -------------------
once = zeros(6,5); plotInterval = 10; plotindex = plotInterval;
dontinterfere = zeros(6,5);

%% ------------ MOVE THE TRAIN TILL THE NEXT STOP---------------
tic
while (disTonext > 0)
    time_stop = toc;
    if time_stop > 120*(s_stop (i)/1e3)
       error ('Compute: Timed out');
    end

    %NOTE: t & t_Com  will include Dwell time outside the loop
    t = t + dt;
    t_bs (i) = t_bs (i) + dt;
    t_Com (i) = t_Com (i) + dt;

    %% ----------- MODE 1, Change creteria is V_max. -----------
    if(v < v_max && s_trav < S_1 && disTonext > S_4)
        mode(i) = 1;
        Fdrag = Cdf*v^2;  Fr = Frr + Fdrag + Fgrad;

        %------------- Check & Controll Te & a ---------
        if v >= v_base
            if once(1,1) < 1
                P_base = Te * v;     %Take the P_base, hold it
                once(1,1) = 2;
            end
            Te = P_base/v;              % Te starts Decreasing
            a =  (Te - Fr)/m;
        elseif Te >= Te_max
            Te = Te_max;
            a =  (Te - Fr)/m;
        else
            % Calculate Te, outside the 2 conditions
            Te = Fr + m*a;
        end
        %-------- Done Check & Controll Te & a -------
        ds = v*dt + 0.5*a*dt^2;
        s = s + ds;
        s_trav = s_trav + ds;
        disTonext = s_stop(i) - s_trav;
        v = v + a*dt;            % Current v

        % ------ MODE 2, Change creteria is Distance ------------

    elseif (v >= v_max && s_trav < (S_1 + S_2) && ...
           disTonext > S_4)
        mode(i) = 2;
        a = 0;
        Fdrag = Cdf*v^2; Fr = Frr + Fdrag; Te = Fr;

        ds = v*dt + 0.5*a*dt^2;
        s = s + ds;
        s_trav = s_trav + ds;
        disTonext = s_stop(i) - s_trav;
        v = v + a*dt;

        %% ------ MODE 3, Change creteria distance --------------
    elseif ( s_trav >= (S_1 + S_2) && disTonext > S_4)
        mode(i) = 3;
        if (once (3,1)< 1) && (strcmp(varargin,'Compute')||...
                strcmp(varargin,'Plotoptimal'))
            S_mismatch = S_3 -(disTonext - S_4);
            once (3,1) = 2;
        end
        Te = 0;
        Fdrag = Cdf*v^2; Fr = Frr + Fdrag;
        a = - Fr/m;

        ds = v*dt + 0.5*a*dt^2;
        s = s + ds;
        s_trav = s_trav + ds;
        disTonext = s_stop(i) - s_trav;
        v = v + a*dt;

        %% ------------ MODE 4,The braking --------------------
    elseif ((disTonext <= S_4) ||...
            ((v <= v_b && s_trav > S_1 + S_2)))
        mode(i) = 4;
        Fdrag = Cdf*v^2;  Fr = Frr + Fdrag;

        if (once (4,1)< 1),
            Te = -2*Frr;                 % Starting Braking 4c
            if abs (Te) >= Tb_max ; Te = -Tb_max; end
            Pconst = Te * v;           % Take the Pconst, Hold it
            once (4,1) = 5;
        end

        if abs (Te) > Tb_max
            Te = -Tb_max;
            a =  (Te - Fr)/m;
            if abs (Te) > 32e3
                disp (Te)
                warning ('(Te) > 32e3 ')
                pause
            end
        elseif (v^2/abs(2*0.9*dec_max) >= disTonext)
            a = -0.9*dec_max;
            Te = a*m + Fr;
            dontinterfere(1) = 5;
            %                   if abs (Te) > 32e3     %Check
            %                        disp (Te)
            %                       warning ('(Te) > 32e3 ')
            %                       pause
            %                   end
        elseif (v^2/abs(2*a) <= disTonext)
            dontinterfere(1) = 5;
            if (once (4,2)< 1)
                acrt = v^2/(2*disTonext);
                once (4,2) = 5;
            end
            a = -acrt;
            Te = a*m + Fr;
            %                   if abs (Te) > 32e3      % Check
            %                        disp (Te)
            %                       warning ('(Te) > 32e3 ')
            %                       pause
            %                    end

            %              elseif (v > v_base_b)
        elseif (dontinterfere(1) < 1  &&...
                v^2/abs(2*a) > disTonext)

            Te = Pconst/v;                 % |Te| increases
            a =  (Te - Fr)/m;              % |a| increase slowly
            if (v^2/abs(2*a) <= disTonext)
                a = v^2/(-2*disTonext);
            end
            Te = a*m + Fr;
            if abs (Te) > Tb_max
                Te = -Tb_max;
                a =  (Te - Fr)/m;
            end
            %                    if abs (Te) > 32e3      % Check
            %                        disp (Te)
            %                       warning ('(Te) > 32e3 ')
            %                       pause
            %                    end

        end

        ds = v*dt + 0.5*a*dt^2;
        s = s + ds;
        s_trav = s_trav + ds;
        disTonext = s_stop(i) - s_trav;
        v = v + a*dt;
    else
        %%----------- ASSES THE PERFOMANCE --------------------

        fprintf ('\nNeither of the conditions was met after');
        fprintf ('\ns = %3.3f ; v = %3.3f; a = %3.3f\n',s,v,a);
```

```
        fprintf ('Stopped at Mode[%1.0f,%1.0f]\n',i,mode(i));
        error('Mode Conditions not Met')
    end %---> If (v < v_max...)
    % -----Done with determining Mode of Operation ------------

    % -------------** VISUALIZE_1** ---------------------------
    if CallAnimation >= 1
        if (s_trav >= s_stop(i)) ; end %v = 0; end
        Visualize( mode(i), t ,v, s,once(5,1), plotindex,indexI)
        once(5,1) = 2;  % Controlls route plotting: Plot route x1
        plotindex = plotindex + 1;
    end
    %-------------------------------------------------------------

    %% ----------------- POWER & ENERGY COMPUTATION -------------
    Pm = (Te * v)/1e3;           % NOTE: Values In K
    Pe_t = Pm/efc;
    if mode(i) == 4
        Pe_t = Pm * efc ;
    end
    Pe = Pe_t + Apu;
    Ee_t = (Pe_t * dt)/hr;          %kWh
    Ee_Apu = (Apu*dt)/hr;           %kWh
    dEe = Ee_t + Ee_Apu;
    Ee = Ee + dEe;                     % Store Commulatively
    E_Com(i) = E_Com(i) + dEe;  E_bs(i) = E_bs(i) + dEe;

    if strcmp(varargin,'Compute')

        %--- Get engfm_bp & cp, then soc, v,& c
        powfm_bp = Pe;
        bp_c = powfm_bp*1e3/bp_v;
        bp_v  = Batt.Pack.V_nom - bp_c*bp_esr;
        engfm_bp = (powfm_bp*dt)/hr;      % kWh
        bp_eng = bp_eng - engfm_bp ;
        bp_soc = (bp_eng/Batt.Pack.E_int);

    end
    %-------------- STORE THE ACCEPTED VALUES ----------------
    T = [T t];  V = [V v];
    S = [S s];  TE = [TE Te];
    PE = [PE Pe]; EE = [EE Ee];
    %PM = [PM Pm];    FR = [FR Fr];
    if strcmp(varargin,'Compute')
        BP_C = [BP_C, bp_c] ; BP_V = [BP_V, bp_v];
        BP_SOC = [BP_SOC, bp_soc];
        Powfrm_BP = [Powfrm_BP, powfm_bp];
        Powfm_CAT = [Powfm_CAT, powfm_cat];
    end

    if                strcmp(varargin,'Compute')                ||
strcmp(varargin,'Plotoptimal')
        %------- STOP KNOWING WHAT CAUSED THE STOP ---------
        if ((s_trav >= s_stop(i)) ||(v <= 0))
            %  fprintf('Distance to S%2.0f = %3.3f m: V = %3.3f
m/s\n',...
            %  (i+1),disTonext,v)
```

```
                break
            end
        elseif strcmp(varargin,'Objfnc')
            if(s_trav >= s_stop(i))  || (v <= 0)
                break
            end
        end

    end % ---> while (disTonext > 0)

    %% If the functon was called just for optimization, return Don't
Dwell.
    if strcmp(varargin,'Objfnc')
        TimeCheck = (T(end))/60;
        if (TimeCheck/((s_stop (i))/1e3)) <= t_max
            y =  EE (end);
        else
%               y =  10 * (EE(end));       % 10x Penalty
                y =  inf;       % invalid, discard
        end

        if v >= 0.1                    % The program cheated on time
%               y =  10 * (EE(end));
            y =  inf;       % invalid, discard
        end

        if disTonext >= 2
%               y =  10 * (EE(end));

            y =  inf;       % invalid, discard
        end
        return % The optimization does not include Dwell time
    end

    %% ===================== DWELL TIME =========================
    if strcmp(varargin,'Compute') % @ During Dwell time
        % update the t_rsv (time reserved)
        t_rsv = t_rsv + (mtt - t_bs(i));
        % No Mechanical Power Issues here, Efficiencies
        cde = 0.9;                   % Batt Charge Discharge efficiency
        conve = 0.91;                % Converter efficiency

        mode(i) = 0;
        Ischargepoint = 0; STT = stt;
        if i < Nstops; TCT = 1; elseif i == Nstops;
        TCT = 10;CallAnimation = 0;end
        if (route == 1); chargepoint = {'S1','S8'};
        elseif(route   ==   2);   chargepoint   =   {'S1','S5'   'S7'
'S8','S10'}; end
        %         elseif(route == 2); chargepoint = {'S1','S10'}; end
        if ismember (staname{i+1},chargepoint)
            Ischargepoint = 1;
            if i < Nstops
                STT = stt + t_rsv;       % ### Rmb to Reset t_rsv
                                         % After chargng
            end
        end
```

```
for p = dt:dt:STT*TCT;
    v = 0; Te = 0;      % Reset the velocity
    % Brake to v=0;
    if once (6,1)< 1; Te = TE(end); once(6,1) = 2; end
    t = t + dt; t_Com (i) = t_Com (i) + dt;
    if (i == Nstops && p >= stt), Apu = 0; end
    Pe = Apu;                   % Equalize Arrays to Plot
    dEe = (Apu*dt)/hr;          %kWh
    Ee = Ee + dEe;
    E_Com(i) = E_Com(i) + dEe;

    % Calculate Batt current,voltage, soc, etc.
    % Knowing not whether the station is a charging point
    if Ischargepoint == 1
        powfrm_cat = Pe/conve;  %Rmb: Pe now is only Aux Load
        powfm_bp = 0;
    else
        powfm_bp = Pe/conve;
        powfrm_cat = 0;
    end

    bp_c = powfm_bp*1e3/bp_v;
    bp_v  = Batt.Pack.V_nom - bp_c*bp_esr;
    engfm_bp = (powfm_bp*dt)/hr;      % kWh
    bp_eng = bp_eng - engfm_bp ;  % de included within 0.6
    bp_soc = (bp_eng/Batt.Pack.E_int);

    if ((Ischargepoint == 1)&& (bp_soc < 1))
        % powfrm_cat supplies Pe(Aux load) and
        % charges Batteries
        if bp_soc < 0.8
            c_rate = 4.5;
            bp_soc = bp_soc +  (c_rate * cde * dt)/hr;
            powfm_cat = (Pe + (c_rate * bp_ah * bp_v))/conve;
        elseif bp_soc >= 0.8
            c_rate = 3;
            bp_soc = bp_soc +  (c_rate * cde * dt)/hr;
            powfm_cat = (Pe + (c_rate * bp_ah * bp_v))/conve;
        end
        if (bp_soc > 1);        % Limit max bp_soc
            bp_soc  = 1;
            c_rate = 0;
            powfm_cat = Pe/conve;    % Reset powefrm_cat
        end
        bp_eng = Batt.Pack.E_int * bp_soc;
        bp_c = - c_rate * bp_ah;
        bp_v  = Batt.Pack.V_nom - bp_c*bp_esr;
    end

    % -- Store variables
    T = [T t];  V = [V v];
    S = [S s];  TE = [TE Te];
    PE = [PE Pe]; EE = [EE Ee];
    %PM = [PM Pm];    FR = [FR Fr];

    BP_C = [BP_C, bp_c] ; BP_V = [BP_V, bp_v];
```

```
            BP_SOC = [BP_SOC, bp_soc];
            Powfrm_BP = [ Powfrm_BP, powfm_bp];
            Powfm_CAT = [Powfm_CAT, powfm_cat];

            % -----------** VISUALIZE_2** -----------------------
            if CallAnimation >= 1
                Visualize( mode(i), t ,v, s,once(5,1), ...
                plotindex,indexI)
                once(5,1) = 2; % Controlls route plotting:
                                 % Plot route once
                plotindex = plotindex + 1;
            end

        end %---> for p = 0:dt:STT*TCT;

        if (Ischargepoint == 1 && i < Nstops) % t_rsv was used
            t_rsv = 0;    % Reset time reserve
        end
    end   % --> if strcmp(varargin,'Compute') % @ During Dwell time
    %%-------- Formating Output btn Stations ---------------------
    fprintf (['\n From ', staname{i},' to ',...
        staname{i+1},' :: %2.2f km'],s_stop(i)/1e3)

    Opt = [x(i,1) x(i,2) x(i,3)*x(i,1)];
    if strcmp(varargin,'Compute')
        if Ischargepoint == 1; fprintf ([ ':: ^^^ @',staname{i+1}]);
        end;
        fprintf  ('\n  Optimal(v_max  a  v_b):          %4.2f   %4.2f
%4.2f', Opt)
        fprintf ('\n Energy btn Stns:           %4.3f kWh',E_bs(i))
        fprintf ('\n Energy btn Stns + D_time:  %4.3f kWh',E_Com(i))
        fprintf ('\n Energy Commulative:        %4.3f kWh',EE(end))
        fprintf  ('\n  Time  btn  Stns:                   %4.3f  min
',(t_bs(i)/60))
        fprintf   ('\n   Time   btn   Stns   +   Dwell   time:  %4.3f   min
',(t_Com(i)/60))
        fprintf  ('\n  Time  Commulative:                   %4.3f  min
',T(end)/60)
        fprintf                                                   ('\n
%%===============================================%% \n\n')
    end
    if strcmp(varargin,'Plotoptimal')
        fprintf  ('\n  Optimal(v_max  a  v_b):          %4.2f   %4.2f
%4.2f', Opt)
        fprintf ('\n Energy btn Stns:           %4.3f kWh',E_bs(i))
        fprintf  ('\n  Time  btn  Stns:                   %4.3f  min
',(t_bs(i)/60))
        fprintf                                                   ('\n
%%===============================================%% \n\n')
    end
    %----------------------------------------------------
end  % ---> for i = indexI:N_STOPS

% Structure & Globalize Tramov
Tramov.V = V; Tramov.S = S; Tramov.T = T; Tramov.PE = PE;
Tramov.TE = TE; Tramov.EE = EE;
if strcmp(varargin,'Compute')
```

```
    Tramov.Powfrm_BP = Powfrm_BP;
    Tramov.Powfm_CAT = Powfm_CAT;
    Batt.Pack_C = BP_C; Batt.Pack_V = BP_V; Batt.Pack_SOC = BP_SOC;
end
%% ###--------- PLOTTING OPTIMAL--------- ###
if strcmp(varargin,'Plotoptimal')
    Plotfigures('Plotoptimal',indexI)
end

%-------------------- Journey Summary -------------------------%
if (strcmp(varargin,'Compute')||...
        (strcmp(varargin,'Plotoptimal')&& SaveStationaly == 0))
    fprintf('\n ---------------- Journey Summary ------------------')
    fprintf('\n Journey distance (km): %4.2f :: Stops: %2.0f',...
        S(end)/1e3,Nstops+1-indexI)
    fprintf('\n Total Time(min) = %4.3f',T(end)/60)
    fprintf('\n Total Energy(kWh) = %4.3f',EE(end))
    fprintf('\n ------------------- THE END ---------------------\n')
end
end      %----------------THE END ----------------%
```

InitializeRoute.m

This function initializes a chosen route (during speed profile optimization or train movement simulation), saves the optimal speed profile values (during speed profile optimization) and load the optimal speed profile values (during train movement simulation).

One thing to note is that the function is used to save the optimal speed profile values if an entire route speed profile was optimize as a whole. That is, it cannot be used to save the optimal speed profile values if optimization was done in section wise.

```
function [x_Optimal]= InitializeRoute(varargin)
%{
                    Written by Joachim Mwambeleko
                   Last update **8/DEC/2015**

##### ====================  NOTE ========================== #####

1. Varagin 1-3 are just for updating speed profile
    2.1   New_x_Optimal: is the Best profile, and contains only
parameter-values for the speed profile, this is the data to write to
the txt file, if speed profile is to be updated.
    2.2  Individual speed profile e.g. GAXL, DEXL and PSOXL can be
local or global. If they are local they should be part of the
varagin, starting from varagin(4)

2. Profile data structure
```

```
    How profile data for xlx file is arranged is quite different to
that for the txt file.
    The xlx data is merely for report writing, when running the
program data from the txt file is loaded.

3. Profile data
    The beauty with the txt file, you can write & load even if it is
open You can NOT write to xlx file if it is open. Make sure the xlx
file is writable, before update the txt file, 'cos you wanna make
sure all the files are updated, with the same data

4. Route details, such as stapos, Nstops and journey_distare are
global. They will be defined only here, after the fnc call.
When the fnc is called only to update speed profile, that means it
was already called to initialize the route details.
  ***** --------------------------------------------------- *****
%}

global route direction Nstops stapos staname journey_dist
global GAXL PSOXL DEVXL XOPXL Method

% ---------------- Updating Speed Profile -------------------%
if ~isempty (varargin)
   JustUpdate = varargin {1};
   UpdateRoute = varargin {2};
   New_x_Optimal = varargin {3};

if JustUpdate == 1
   if strcmpi(UpdateRoute,'TRL')
  Test_Writing_xlx_File('TRL')
  xop_trl = New_x_Optimal;
  xlswrite('TRL_Spd_Profile.xlsx',GAXL,1,'E5')  % Sheet 1->GA
  xlswrite('TRL_Spd_Profile.xlsx',DEVXL,2,'E5')  % Sheet 2->DEV
  xlswrite('TRL_Spd_Profile.xlsx',PSOXL,3,'E5') % Sheet 3->PSO
  xlswrite('TRL_Spd_Profile.xlsx',XOPXL,4,'E5')    % Sheet 4->ZeBest
  xlswrite('TRL_Spd_Profile.xlsx',Method,4,'E11')
  save('TRL_Speed_Profile.txt','xop_trl','-ascii') % Save to txt file
  save('VarStore.mat','xop_trl','-append')        % Save to MAT file
 msgbox('Optimal  Speed  Profile  for  TRL  Route  was  successfully
Updated',...
      'TRL Speed Profile'); % Just let the user know
 fprintf('Optimal Speed Profile for TRL Route has been Updated\n')

   elseif strcmpi(UpdateRoute,'TZR')
   Test_Writing_xlx_File('TZR')
   xop_tzr = New_x_Optimal;
  xlswrite('TZR_Spd_Profile.xlsx',GAXL,1,'E5')  % Sheet 1->GA
  xlswrite('TZR_Spd_Profile.xlsx',DEVXL,2,'E5')  % Sheet 2->DEV
  xlswrite('TZR_Spd_Profile.xlsx',PSOXL,3,'E5') % Sheet 3->PSO
  xlswrite('TZR_Spd_Profile.xlsx',XOPXL,4,'E5')    % Sheet 4->ZeBest
  xlswrite('TZR_Spd_Profile.xlsx',Method,4,'E11')
    save('TZR_Speed_Profile.txt','xop_tzr','-ascii')  %  Save  to  txt
file
    save('VarStore.mat','xop_tzr','-append')           %  Save  to  MAT
file
 msgbox('Optimal  Speed  Profile  for  TAZARA  Route  was  successfully
Updated',...
```

```
      'TRL Speed Profile'); % Just let the user know
    fprintf('Optimal  Speed  Profile  for  TAZARA  Route  has  been
Updated\n')
   end
    x_Optimal = New_x_Optimal;
    return
end %---> if JustUpdate == 1
end  %---> if ~isempty (varargin)
 %%----------------------   CHOOSE A ROUTE   -------------------------

if route == 1
      fprintf ('---------TRL ROUTE-----------\n')
staname = {'S1', 'S2', 'S3',  'S4',  'S5'  'S6'   'S7'     'S8'};
stapos =   [0    1.41  3.58   5.74   7.9   9.33   10.03   12.05]*1e3;
x_Optimal =  load('TRL_Speed_Profile.txt');

  elseif route == 2
       fprintf ('--------TAZARA ROUTE---------\n')
staname = {'S1','S2','S3','S4','S5','S6','S7','S8','S9', 'S10'};
stapos =   [0    1    3    4     6    9   11   17   19     21]*1e3;

x_Optimal = load('TZR_Speed_Profile.txt');

else
     x_Optimal = [];
     return
end
if direction == 2
   staname = fliplr(staname);
   stapos = -(fliplr(stapos));
   x_Optimal = flipud(x_Optimal);
end

 Nstops = length(stapos)-1;
journey_dist = (stapos(Nstops + 1)-stapos(1))/1e3;   % km
  end
```

EMU_Compute_S_1.m

This function computes distance to be travelled in acceleration mode

```
function   [S_1,    t_1,  v_max_mode1]=  EMU_Compute_S_1(acc,  v_max
,v_base, m)
%{
NOTE
 Whenever calling Compute_S_1 make sure
 The velocities are in m/s
%------------------- PERMANENT CONSTANTS -------------------------
global fc Cdf  Te_max
global dt g
Frr = fc*m*g;
%-----------------Checking v_base Vs v_max------------------------
if v_base >= v_max
```

```
    error ('v_base should be less than  v_max')
end
if v_max > 50/3.6
    v_max = 50/3.6;
    warning ('\n Maximum velocity has been reset to 60 kph')
end
%------------------------------------------------------------------
tic
t = 0;s = 0; v = 0;   Fdrag = 0; Fgrad = 0;
Fr = Frr + Fdrag + Fgrad;
Te = Frr + m*acc;                  % Initial Te
if Te >= Te_max
    Te = Te_max;                   % Acceptable initial Te
    a = (Te - Frr)/m;              % Acceptable starting acc
else
    a = acc;
    Te = Fr + m*a;
end

once_S1 = 0;
S_1 = s;  t_1 = t;
T = t; A_1 = a; S = s; V = v; TE = Te;
%%-----------------------------------------------------------------

while (v < v_max)
    time_stop = toc;
    if time_stop >= 60
        error ('EMU_Compute_S_1: Timed out')
    end

    t = t + dt;
    Fdrag = Cdf*v^2;
    Fr = Frr + Fdrag;


    if v >= v_base
        if once_S1 < 1
            P_base = Te * v;           %Take the P_base, hold const
            once_S1 = 2;
        end
        Te = P_base/v;                    % Te starts decreasing
        a =  (Te - Fr)/m;

    elseif Te >= Te_max
        Te = Te_max;
        a =  (Te - Fr)/m;
    else
        % Calculate Te, outside the 3 conditions
        Te = Fr + m*a;
    end
    ds = v*dt + 0.5*a*dt^2;
    s = s + ds;
    v = v + a*dt;           % This 'a' has already been accepted

        S_1 = s;  t_1 = t;
        T = [T, t]; S = [S,s]; TE = [TE, Te]; V = [V,v];
        A_1 =[A_1, a];
        v_max_mode1 = V(end);
    end %---> if v >= v_max
end %---> while 1
```

EMU_Compute_S_3.m

This function computes the distance to be travelled in coasting mode (mode 3)

```
function [S_3,  t_3, v_b_mode3] = EMU_Compute_S_3(v, v_b, m)

% NOTE
% Whenever calling Compute_S_3 make sure
% The velocities are in m/s
% v = v_max; m = mass;

% PERMANENT CONSTANTS
global fc Cdf dt g   % Rolling friction coef, Fdrag Coef

Frr = fc*m*g;
Fdrag = Cdf*v^2;
Fr = Frr + Fdrag;
a = -(Fr/m);
v_check = v + a*dt;

if (v < v_b)
   fprintf ('v_b = %2.5f while v_max = %2.5f \n', v_b, v)
    error ('v_b can not be > v_max')

elseif (v_check <= v_b)
  S_3 = 0;   t_3 = 0; v_b_mode3 = v;
    return
elseif (v*0.97 <= v_b)
   S_3 = 0;   t_3 = 0; v_b_mode3 = v;   % Assume No Coasting
   return
end
dispTest = 1; tic
%==========================================================
s = 0; t = 0; a = 0; Te = 0;
S_3 = s;  t_3 = t;

T = t; A = a; S = s; V = v; TE = Te;

while 1
 time_stop = toc;
 if time_stop >= 10
      error ('IPEMU_Compute_S_3: Timed out')
 end

t = t + dt;
Fdrag = Cdf*v^2;
Fr = Frr + Fdrag;
%Te = m*a+ Fr = 0
%-------------------------TESTING-----------------------
```

```
 a = -(Fr/m);
%a = -1; if dispTest ==1 warning ('Testing dec'); dispTest = 0; end
                            % Testing
       % t_3 = (v_b - v_max )/a ; s_3 = (v_b)^2 - v_max^2)/(2*a)
%-------------------------------------------------------------------
ds = v*dt + 0.5*a*dt^2;    % This is correct, v is updated later
s = s + ds;
v = v + a*dt;

if (v <= v_b)                 % Break b4 storing, else.
         break
else
 S_3 = s;  t_3 = t;
 T = [T, t]; A =[A, a]; S = [S,s]; TE = [TE, Te]; V = [V,v];
 v_b_mode3 = V(end);
end
end
end
```

EMU_Compute_S_4.m

This function computes the distance to be traveled in braking mode (mode 4)

```
function [S_4,  t_4, A] = EMU_Compute_S_4(v_b, v_base_b, m)
%{
NOTE
Whenever calling this fnc make sure
The velocities are in m/s
%}
%--------------------- PERMANENT CONSTANTS-------------------------
global fc Cdf  Tb_max dec_max
global dt g
Frr = fc*m*g;
v = v_b;
Fdrag = Cdf*v^2; Fgrad = 0;
Fr = Frr + Fdrag + Fgrad;
Te = -2*Frr;                          % Starting Braking 4c
if abs (Te) >= Tb_max ; Te = -Tb_max; end
a = (Te-Fr)/m;
Pconst = Te * v_b;         % Hold this constant till at v_base_b

%------------------Checking v_base_b Vs v_b------------------------
 if v_base_b > v_b
     error ('v_base braking should be less than  v_b')
 elseif v_base_b >= 0.6*v_b
     %warning ('v_base braking is too close to v_b')
 end

%--------------------------------------------------------------------
tic
t = 0; s = 0;
S_4 = s;  t_4 = t;
```

```
T = t; A = a; S = s; V = v; TE = Te;
%%-------------------------------------------------------------------

while 1
  time_stop = toc;
 if time_stop >= 10
       error ('IPEMU_Compute_S_4: Timed out')
 end

 t = t + dt;
 ds = v*dt + 0.5*a*dt^2;      % Use previous velocity
 s = s + ds;                  % For IPEMU check the energy b4 moving
 v = v + a*dt;                % This 'a' has already been accepted
 Fdrag = Cdf*v^2;             % Use current velocity
 Fr = Frr + Fdrag;

if abs (Te) >= Tb_max         % if this is reached b4 v_base_b.
       Te = -Tb_max;              % U can't reach v_base_b any more
    a =  (Te - Fr)/m;             % NOTE: Fr is decreasing |a|decreases

    if abs (a)>= dec_max                    % Comfortability
      a = -dec_max;
     Te = a*m + Fr;
    end

elseif v > v_base_b
       Te = Pconst/v;            % Te is neg, |Te| increases
    a =  (Te - Fr)/m;            % |a| increases

    if abs (a)>= dec_max                    % Comfortability
      a = -dec_max;
     Te = a*m + Fr;
    end
else
    % Calculate Te, outside the 2 conditions
    % If u reach v_base_b b4 reaching Tb_max
    Te = Fr + m*a;
end

      if v <= 0
            break                             % Breake b4 storing
          else

         S_4 = s;  t_4 = t;
       T = [T, t]; A =[A, a]; S = [S,s]; TE = [TE, Te]; V = [V,v];
      end
end
end
```

Test_Writing_xlx_File.m

This function checks the xlx file if it is writable, if it is not; the user is given an opportunity to close the file to allow the optimal data to be saved, otherwise the MATLAB® program detect that as an error and the data is lost. Thus, speed profile optimization has to be repeated, which is very time consuming.

```
function Test_Writing_xlx_File(TestFile)

         %============= ###### NOTE ##### ==================
 %{
                  **Written by Joachim Mwambeleko**
                     *Last Update 10/DEC/ 2015*
1. Its no good at all to spend several hours searching for optimal
speed profile, and then at the end the program fails to save the
optimal values, just because the xlx file was not writable. With this
fucn the problem it taken care of, one will have the chance to close
the file and save the data
2. Test writing to the first sheet of a particular xlx file.
3. Rmb: Try writing OUTSIDE the data Table, So for convenience sake
we are going to to test writing at cell Q4 for both TRL and TZR files
4. Don't confuse or mix-up the names
  %}
 % ***** --------------------------------------------------- ***** %
 if strcmp(TestFile ,'TRL')
 try
    %Check if TRL_Spd_Profile is open
    Checkfileopenid = 0;
    xlswrite('TRL_Spd_Profile.xlsx',{'TEST_1'},1,'Q4')
catch Open_Test
    if (strcmp(Open_Test.identifier,'MATLAB:xlswrite:LockedFile'))
    Checkfileopenid = 1; % To tell the program that the file was open
% Call a user
 set(0, 'DefaultUIControlFontSize', 12);
 d = dialog('Position',[600 350 400 150],'Name','Attention!');

 uicontrol('Parent',d,'Style','text','Position',[20 80 350 60],...
    'String',{'TRL_Spd_Profile is open, overwriting failed !' ''...
        'Please Close the file then click Continue '});

 uicontrol('Parent',d,'Position',[300 20 90 35],...
            'String','Continue','Callback','delete(gcf)');
   uiwait(d);   % ### Wait for user responce (User input wait)
     end
 end    % ---> try (The 1st try)

% RE-Check
if (Checkfileopenid > 0) % Re-check only if the file WAS indeed open
try
    xlswrite('TRL_Spd_Profile.xlsx',{'TEST_2'},1,'Q4')
catch Open_Test
    switch Open_Test.identifier
        case ('MATLAB:xlswrite:LockedFile')
  % Call user
   uigetpref('Xlx','Chec','**Attention-Attention**!',...
    {'For the last time, TRL_Spd_Profile is open' '' ...
    'Please CLOSE the file, then click Continue'},...
    {'Go_ahead';'Continue'});         % Values and button strings
        otherwise
        disp('Thanks for your attention\n') %## Y not executed ???
    end
end
 xlswrite('TRL_Spd_Profile.xlsx',{'TEST_2B'},1,'Q4')
end % ---> if (Checkfileopenid > 1)
% *-----------------------------------------------------------* %
                            TAZARA
 % *----------------------------------------------------------* %
 elseif strcmp(TestFile ,'TZR')
  try
        Checkfileopenid = 0;
    xlswrite('TZR_Spd_Profile.xlsx',{'TEST_1'},1,'Q4')
catch Open_Test
    if (strcmp(Open_Test.identifier,'MATLAB:xlswrite:LockedFile'))
    Checkfileopenid = 1;
  d = dialog('Position',[600 350 400 150],'Name','Attention!');

 txt  =  uicontrol('Parent',d,'Style','text','Position',[20  80  350
60],...
    'String',{'TZR_Spd_Profile is open, overwriting failed !' ''...
        'Please Close the file then click Continue '});

    btn = uicontrol('Parent',d,'Position',[300 20 90 35],...
            'String','Continue','Callback','delete(gcf)');
   uiwait(d);   % ### Wait for user responce
    end
 end
% RE-Check
if (Checkfileopenid > 0) % Re-check only if the file was indeed open
try
    xlswrite('TZR_Spd_Profile.xlsx',{'TEST_2'},1,'Q4')
catch Open_Test
    switch Open_Test.identifier
        case ('MATLAB:xlswrite:LockedFile')
  % Call user
   uigetpref('Xlx','Chec','**Attention-Attention**!',...
    {'For the last time, TZR_Spd_Profile is open' '' ...
    'Please CLOSE the file, then click Continue'},...
    {'Go_ahead';'Continue'});         % Values and button strings
        otherwise
        disp('Thanks for your attention\n') %## Y not executed ???
    end
 end
end % ---> if (Checkid == 1)
xlswrite('TZR_Spd_Profile.xlsx',{'TEST_2B'},1,'Q4')
 end
```

SaveOptimalStationaly.m

Unlike the “**InitializeRoute.m**” this function allows optimizing and saving speed profile optimal values section wise

```
function SaveOptimalStationaly(varargin)
%{
                     Written by Joachim Mwambeleko
                    Last update **29/Feb/2015**

##### ===================  NOTE ====================== #####
    This is a new Technique, to save optimal values for every
intermediate station independently.
    Actually this was written so as graphs and snapshots for every
section can be saved.
    For convenient purposes, in this function the route is identified
by Number: 1-> TRL, 2-> TZR

    ***** --------------------------------------------------- *****
%}
global dt
global GAXL PSOXL DEVXL XOPXL Method

% ---- Updating Speed Profile for every stn independently ------%

stn = varargin {1};    % Value of i from Main_Optimization
UpdateRoute = varargin {2};
xopt_sctn = varargin {3};

if  (UpdateRoute == 1)
    Test_Writing_xlx_File('TRL')
    xop_trl_newsectn = xopt_sctn;

    %----View the fig b4 saving the data-----
    % Call EMU_Compute, it will then call the plotfigures
    %  Index & Nstops are global
    EMU_Compute(xopt_sctn,'Plotoptimal');

    hmsgbx = msgbox({'Review the Speed Profile and',...
        'Press any key to continue'},...
        'Review Speed profile');
    set(hmsgbx, 'position', [450 300 200 70]);
    ah = get( hmsgbx, 'CurrentAxes' );
    ch = get( ah, 'Children' );
    set( ch, 'FontSize', 12.5 );
    pause

    set(0, 'DefaultUIControlFontSize', 12.5);
    Overwrite  =  questdlg({'Do  you  want  to  Update  Optimal  Speed
Profile',...
        'for THIS SPECIFIC Section of the TRL Route?'}, ...
        'TRL: Speed Profile Sectionwise','YES','NO','NO');
    switch Overwrite
        case 'YES'
```

```
            % Keep going
        case 'NO'
            warning ('Unupdated Section')
            fprintf ('TRL route: optimal speed profile ')
            fprintf ('S%1.0f - S%1.0f ',stn,stn+1)
            fprintf ('Was NOT Updated\n')
            return
    end

    % rmb % Sheet 1->GA, 2->DE, 3->PSO , and 4->ZeBest
    % ### double ('A') = 65, and char(65)= A
    % Start writing col E row 5, then col F row 5 ...
    % The data 2b written has 2b a col vector
    % Rmb: GAXL,DEVXL, PSOXL XOPXL are written column wise
    % Method is a row vector
    % xop is written row wise

xlswrite('TRL_Spd_Profile.xlsx',GAXL(:,stn),1,[char(stn+68),'5'])
xlswrite('TRL_Spd_Profile.xlsx',DEVXL(:,stn),2,[char(stn+68),'5'])
xlswrite('TRL_Spd_Profile.xlsx',PSOXL(:,stn),3,[char(stn+68),'5'])
xlswrite('TRL_Spd_Profile.xlsx',XOPXL(:,stn),4,[char(stn+68),'5'])

    % The Methods are written horizontally

xlswrite('TRL_Spd_Profile.xlsx',Method(stn),4,[char(stn+68),'11'])
 %{
     --------------- Saving into text & mat file ---------------
    ## Careful, to change a specific variable there are two options
    1. Use load & save . i.e. load the file edit the specific
       variable and save. This is memory consuming
    2. use 'matfile' fnc . This has some limitations

    All of the two options require that the variable must
    have been created beforehand.
 %}
    % For text file
    xop_trl = load('TRL_Speed_Profile.txt');
    xop_trl(stn,:) = xop_trl_newsectn(stn,:);      % Edit that part
    % Save to txt file
    save('TRL_Speed_Profile.txt','xop_trl','-ascii')

    % Save to MAT file
    save('VarStore.mat','xop_trl','-append')

    fprintf ('TRL route: optimal speed profile ')
            fprintf ('S%1.0f - S%1.0f ',stn,stn+1)
            fprintf ('was Successfully Updated\n')

%==================================================================%
elseif (UpdateRoute == 2)
    Test_Writing_xlx_File('TZR')
    xop_tzr_newsectn = xopt_sctn;

    %----View the fig b4 saving the data-----
    % Call EMU_Compute, it will then call the plotfigures
    %  Index & Nstops are global
    EMU_Compute(xopt_sctn,'Plotoptimal');

    hmsgbx = msgbox({'Review the Speed Profile and',...
        'Press any key to continue'},...
        'Review Speed profile');
    set(hmsgbx, 'position', [450 300 200 70]);
    ah = get( hmsgbx, 'CurrentAxes' );
    ch = get( ah, 'Children' );
    set( ch, 'FontSize', 12.5 );

    pause

    set(0, 'DefaultUIControlFontSize', 12.5);
    Overwrite  =  questdlg({'Do  you  want  to  Update  Optimal  Speed
Profile',...
        'for THIS SPECIFIC Section of the TAZARA Route?'}, ...
        'TRL: Speed Profile Sectionwise','YES','NO','NO');
    switch Overwrite
        case 'YES'
            % Keep going
        case 'NO'
            warning ('Unupdated Section')
            fprintf ('TAZARA route: optimal speed profile ')
            fprintf ('S%1.0f - S%1.0f ',stn,stn+1)
            fprintf ('Was NOT Updated\n')
            return
    end

xlswrite('TZR_Spd_Profile.xlsx',GAXL(:,stn),1,[char(stn+68),'5'])
xlswrite('TZR_Spd_Profile.xlsx',DEVXL(:,stn),2,[char(stn+68),'5'])
xlswrite('TZR_Spd_Profile.xlsx',PSOXL(:,stn),3,[char(stn+68),'5'])
xlswrite('TZR_Spd_Profile.xlsx',XOPXL(:,stn),4,[char(stn+68),'5'])

    % The Methods are written horizontally

xlswrite('TZR_Spd_Profile.xlsx',Method(stn),4,[char(stn+68),'11'])

%     --------------- Saving into text & mat file --------------

    % For text file
    xop_tzr = load('TZR_Speed_Profile.txt');
    xop_tzr(stn,:) = xop_tzr_newsectn(stn,:);      % Edit that part
    % Save to txt file
    save('TZR_Speed_Profile.txt','xop_tzr','-ascii')

    % Save to MAT file
    save('VarStore.mat','xop_tzr','-append')

    fprintf ('TAZARA route: optimal speed profile ')
            fprintf ('S%1.0f - S%1.0f ',stn,stn+1)
            fprintf ('was Successfully Updated\n')
end
end
```

D.2 BEMU Movement Simulation

The MATLAB® scripts presented in this section were used to model and simulate the tram as a BEMU, plot graphs and, save results.

D.2.1 Agui_Initialization.m

This script starts up a GUI from which a route and direction can be selected. And, an option to view train movement animation can also be selected as either 'YES' or 'NO'. The **'BEMU()'** function is then called to proceed with BEMU movement simulation.

```
function varargout = Agui_Initialization(varargin)
%AGUI_INITIALIZATION M-file for Agui_Initialization.fig
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn',...
                   @Agui_Initialization_OpeningFcn, ...
                   'gui_OutputFcn',...
                   @Agui_Initialization_OutputFcn, ...
                   'gui_LayoutFcn',  [], ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
   gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

% --- Executes just before Agui_Initialization is made visible.
function Agui_Initialization_OpeningFcn(hObject, eventdata, handles,
varargin)
% Choose default command line output for Agui_Initialization
handles.output = hObject;

set(0, 'DefaultUIControlFontSize', 8);
axes(handles.SUT_logo)      % Make current axes
imshow('AA_IESUT.png')
axes(handles.Tram_photo)      % Make current axes
imshow('AA_Tram.png')

handles.rut = 1;         % Default Values
handles.dirctn = 1;
handles.animate = 0;

% Update handles structure
guidata(hObject, handles);
```

```
% --- Outputs from this function are returned to the command line.
function     varargout     =     Agui_Initialization_OutputFcn(hObject,
eventdata, handles)
% hObject    handle to figure
% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on selection change in RouteMenu_Tag.
function RouteMenu_Tag_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function RouteMenu_Tag_CreateFcn(hObject, eventdata, handles)
% hObject    handle to RouteMenu_Tag
% handles     empty - handles not created until after all CreateFcns
called
% Set white background for popupmenu controls
if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in DirectionMenu_Tag.
function DirectionMenu_Tag_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function DirectionMenu_Tag_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),...
        get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in Animation_Tag.
function Animation_Tag_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function Animation_Tag_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),...
    get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in Save_Run_PB.
function Save_Run_PB_Callback(hObject, eventdata, handles)

% Get the Chosen Route
str = get(handles.RouteMenu_Tag,'String');
val = get(handles.RouteMenu_Tag,'Value');
    % Pp Hint: String — a CELL ARRAY that contains the menu contents
    %         Value — the index of a menu selected content
  switch str{val}
      case 'TRL'
          handles.rut = 1;
      case 'TAZARA'
```

```
            handles.rut = 2;
    end

  % Get the Chosen Direction
    str = get(handles.DirectionMenu_Tag,'String');
    val = get(handles.DirectionMenu_Tag,'Value');
    switch str{val}
        case 'Odd'
            handles.dirctn = 1;
        case 'Even'
            handles.dirctn = 2;
    end

  % Get Decision on Animating
    str = get(handles.Animation_Tag,'String');
    val = get(handles.Animation_Tag,'Value');
    switch str{val}
        case 'NO'
            handles.animate = 0;
        case 'YES'
            handles.animate = 1;
    end

  guidata(hObject,handles); % Save/Update the handles
  global route direction CallAnimation
  route = handles.rut ;
  direction = handles.dirctn ;
  CallAnimation = handles.animate ;
   run BEMU()         % Call the EMU_Batt_SuperCap Fnc

  % --- Executes on button press in Exit_PB.
  function Exit_PB_Callback(hObject, eventdata, handles)
  % hObject    handle to Exit_PB (see GCBO)
  set(0, 'DefaultUIControlFontSize', 12);
  exit_now = questdlg('Are you sure you want to Exit?',...
      'Exit Program','Yes','No','No');
  switch exit_now
  case 'Yes'
  delete (gcf)
  case 'No'
  return
  end
```

BEMU.m

This function starts the BEMU movement simulation

```
function  BEMU()
%{
                  **Written by Joachim Mwambeleko**
                      *Last update 25/Jan/2016*
%}

%% ROUTE & PROFILES : Initialize a Route and Speed Profile
global direction CallAnimation Tramov route
PlotOtherfigs = CallAnimation;

[x_Optimal]= InitializeRoute();
if isempty (x_Optimal) % No route was chosen
    fprintf  ('  A  route  was  not  Initialized  Please  re-run  the
program\n')
    return
end
if direction == 1;
    fprintf(' ***** Direction: ODD *******\n')
elseif direction == 2;
    fprintf('   ***** Direction: EVEN *****\n')
end

%%  STORAGE CAPACITY & EFFECTIVE WEIGHT
global m_compute Batt

% Get battery details
Batt.weight = 27.4;       % kg
Batt.Ah = 60;
Batt.ESR = 4e-3;                 % Ohms
Batt.soc_int = 1;
Batt.V_nom = 24;
Batt.V_fullcharge = 27;
Batt.V_int = 24;            %   Under nominal area, (fcn of soc)
Batt.No_total = 54;
Batt.No_series = 27;
Batt.No_parallel = Batt.No_total/Batt.No_series;
if mod(Batt.No_parallel, 1)~= 0;
    error ('No series-parallel battery combination could be found')
end
Batt.Pack.ESR = Batt.ESR * Batt.No_series/Batt.No_parallel;
Batt.Pack.Ah = Batt.Ah * Batt.No_parallel;
Batt.Pack.V_int = Batt.V_int * Batt.No_series;
Batt.Pack.V_nom = Batt.V_nom * Batt.No_series;
Batt.Pack.E_int = (Batt.Pack.Ah * Batt.Pack.V_int)/1e3;  % kWh
Batt.Pack.soc_int = Batt.soc_int;
Batt.Pack.Plim = 86;    %### in kW
Batt.Pack.W = Batt.No_total * Batt.weight;
```

```
%--- Get Tram details
m_t = 38.8e3; RA = 1.11;                          % Tare-mass, rotary
allowance
PC  = 312; Wpp = 60.7; Pw = PC*Wpp;          % Passenger Weight
m_compute = m_t* RA + Pw + Batt.Pack.W;
% m_compute = 10;


%% PERMANENT CONSTANTS & TRAIN DETAILS
global fc Cdf Te_max Tb_max dec_max   Pe_max % Frr_Coef,Fdrag_Coef
global dt efc g
fc = 0.006;                % Rolling friction coefficient
Cd = 0.6;                  % Drag coefficient
Af = 7.7; Ad = 1.225;      % Frontal area (m^2);  Air density (kg/m^3)
efc = 0.6;                 % Power conversion efficiency
g  = 9.81;  dt = 0.1;
Cdf = 0.5*Ad*Cd*Af;        % Fdrag_Coefficient
Pe_max = 8*45e3;  v_base_max = 0.45 * (50/3.6); %(6 m/s2)
Te_max = (Pe_max * efc)/v_base_max; Tb_max = Te_max;
dec_max = abs((-Tb_max - (fc*m_compute*g))/m_compute);
%dec.crt1 = 0.25.

%% TRAIN MOVEMENT SIMULATION
fprintf (' Battery Pack:\n');
fprintf ('         V_nom: %3.2f V\n',Batt.Pack.V_nom);
fprintf ('            Ah: %3.2f Ah\n',Batt.Pack.Ah);
fprintf ('         E_int: %3.2f kWh\n',Batt.Pack.E_int);
EMU_Compute(x_Optimal,'Compute');

%-------------------- Plot the rest of the Figures -----------------
set(0, 'DefaultUIControlFontSize', 12);
if PlotOtherfigs == 0
    Plotfigures()
elseif PlotOtherfigs == 1
    Plotfigs = questdlg('Do you want to Plot Other Figures?',...
        'Figures-Plotting','YES','NO','NO');
    switch Plotfigs
        case 'YES'
            Plotfigures()
    end
end

SAVE    =    questdlg('Do    you    wan''t    to    Save    Results?','
','YES','NO','NO');
switch SAVE
    case 'YES'
        if route == 1
            TramEE_trl = Tramov.EE(end);
            save ('VarStore.mat', 'TramEE_trl','-append') %### Rmb -
append
        elseif route == 2
            TramEE_tzr = Tramov.EE(end);
            save ('VarStore.mat', 'TramEE_tzr','-append')
        end
end
set(0, 'DefaultUIControlFontSize', 8);
end
```

Visualize.m

If animation is selected as ‘YES’, this function animate tram movement

```
function []= Visualize(mode, t ,v, s,once_pr, plotindex,indexI)
%{
                    Written by Joachim Mwambeleko
                   Last update **2/Sept/2015**
                          *PLATFORM*
%}

global Nstops stapos staname
plotInterval = 25;
plotInterval_2 = 10*plotInterval;
plotInterval_3 = 5*plotInterval;
plotInterval_reset = plotInterval;
figure (100)
if mode == 2 || mode == 0 , plotInterval = plotInterval_2;
elseif mode == 3,  plotInterval = plotInterval_3;
elseif mode == 4,  plotInterval = 1.2* plotInterval ;
    if v <= 1; plotInterval = plotindex;end % Plot the last poin
else  plotInterval = plotInterval_reset;
end

if rem (plotindex,plotInterval)== 0             % Visualize
    subplot (2,1,1)
    plot(t, v.*3.6,'kx','LineWidth',1.5),grid off,hold on
    title ('Train Kinematics','FontWeight','bold')
    xlabel('Time (s)')
    ylabel ('Velocity (kph)')
end
 subplot (2,1,2)
PlotRange = (stapos(Nstops + 1)-stapos(indexI))/1e3;
PlotStart = stapos(indexI)/1e3;

if once_pr < 1
    for pr = indexI: Nstops + 1
        plot ((stapos(pr)/1e3 - PlotStart ),2,'s'), hold on
        xlabel({'Train Moving'; 'Distance in km'})
        set(gca,'YTick',[])
        line([(stapos(pr)/1e3    -    PlotStart)    (stapos(pr)/1e3    -
PlotStart)],[3 8])
        text((stapos(pr)/1e3 - PlotStart),8,staname(pr),...
            'EdgeColor','red','LineWidth',2,'FontWeight','bold',...
            'LineStyle','-','HorizontalAlignment','center',...
            'BackgroundColor',[.6 .9 .7])
        axis ([-0.5 PlotRange + 0.5  0  9] )
    end
end
if rem (plotindex,plotInterval)== 0
    get([]);
    plot (s/1e3, 2, '+')
end
```

Plotfigures.m

This function plots the figures related to BEMU movement simulation

```
function []= Plotfigures(varargin)
%{
                     Written by Joachim Mwambeleko
                    Last update **25/Jan/2016**
                           *PLATFORM*
The varagin
    varagin {1} = for 'Plotoptimal'
    varagin {2} = Index  (Plotting starting point), by default the
                  ending point is specified by "Nstops + 1"
%}
global Batt Tramov staname stapos Nstops
V = Tramov.V;
S = Tramov.S;
T = Tramov.T;
TE = Tramov.TE;
PE = Tramov.PE;   EE = Tramov.EE;

if ~isempty(varargin)
    if strcmp (varargin{1},'Plotoptimal')
        indexI = varargin{2};
        figure ('Name','Speed Vs Time Optimal', ...
         'NumberTitle','off', ...
         'units','normalized',...
         'DefaultAxesXMinorTick','on','DefaultAxesYminorTick','on')
        plot(T, V.*3.6,'LineWidth',1.5)
        title ('Tram Speed Profile','FontWeight','bold')
        xlabel('Time (s)','FontWeight','bold')
        ylabel ('Velocity (kph)','FontWeight','bold')
        axis ([0  inf   0  (max(V)*3.6)*1.1])

       figure ('Name','Speed Vs Distance Optimal', ...
         'NumberTitle','off', ...
         'units','normalized',...
         'DefaultAxesXMinorTick','on','DefaultAxesYminorTick','on')
        PlotRange = (stapos(Nstops + 1)-stapos(indexI))/1e3;
        PlotStart = stapos(indexI)/1e3;
        for pr = indexI: Nstops + 1
            plot ((stapos(pr)/1e3 - PlotStart),0,'s'), hold on
            text((stapos(pr)/1e3 - PlotStart),-2,staname(pr),...
                'EdgeColor','red','LineWidth',2,...
                'FontWeight','bold','LineStyle','-',...
                'HorizontalAlignment','center',...
                'BackgroundColor',[.6 .9 .7])
            axis     ([-0.1*PlotRange     1.1*PlotRange           -5
(max(V)*3.6)*1.1] )
        end
        plot(S./1e3, V.*3.6,'LineWidth',1.5)
        title ('Tram Speed Profile','FontWeight','bold')
        xlabel('Distance (km)','FontWeight','bold')
        ylabel ('Velocity (kph)','FontWeight','bold')
        hold off
```

```
          figure ('Name','Power Profile Clue','NumberTitle','off', ...
          'units','normalized',...
          'DefaultAxesXMinorTick','on','DefaultAxesYminorTick','on')
         plot(T, PE,'LineWidth',1.5)
         title ('Tram Electric Power Demand','FontWeight','bold')
         xlabel('Time (s)','FontWeight','bold')
         ylabel ('Electric Power (kW)','FontWeight','bold')
         axis ([0 inf  1.1*min(PE)  1.1*max(PE)])
         return
    end
end

BP = Tramov.Powfrm_BP;
BSOC = Batt.Pack_SOC; BV = Batt.Pack_V; BC = Batt.Pack_C;

figure ('Name','Velocity & Tractive Effort','NumberTitle','off', ...
         'units','normalized',...
         'DefaultAxesXMinorTick','on','DefaultAxesYminorTick','on')
[hax, hline1, hline2] = plotyy (T, V.*3.6, T,TE);
set (hline1,'LineWidth',2);
set (hline2,'LineStyle','-','LineWidth',1);
xlabel ('Time (s)','FontWeight','bold')
ylabel (hax(1), 'Velocity (km/h)','FontWeight','bold')        % Left
ylabel (hax(2), 'Tractive Force (kN)','FontWeight','bold')    % Right
xlim (hax(1),[0 inf]); xlim (hax(2),[0 inf]);
legend('Velocity','Tractive Force')

    ylabels  =  {'Distance  Travelled  (km)','Net  Energy  Consumed
(kWh)',...
        'Battery SoC (%)'};
    Xlabel = 'Time (s)';
    Title  = 'Distance, Energy & Battery SoC  Vs Time';
    Legend = {'Distance','Energy','Batt SoC','SouthEast'};
    Plot_yyy(T,S/1e3,                                          T,EE,
T,BSOC.*100,[],[],ylabels,Xlabel,Title,Legend);

    figure('Name','Batt         Current,Voltage         &         Tram
Power','NumberTitle','off', ...
         'units','normalized',...
         'DefaultAxesXMinorTick','on','DefaultAxesYminorTick','on')
    subplot(4,1,[2,3,4])
    [ax, hline1, hline2] = plotyy(T,BC, T, PE);   %T,BSOC*100
    set(hline1,'LineWidth',1.5);
    set (hline2,'LineStyle','-.','LineWidth',1.5);
    ylabel (ax(1), 'Battery Current (A)','FontWeight','bold')
    ylabel (ax(2), {'Tram Power','Demand (kW)'},'FontWeight','bold')
    set(gca,'xtick',[]);
    xlabel ('Time (s)','FontWeight','bold')
    hleg = legend('Batt current','Tram Power Demand',...
              'Location','SouthEast');
    set(hleg,'FontSize',9,'FontAngle','italic')

    xlim (ax(1),[0  inf]); xlim (ax(2),[0  inf])
    subplot (4,1,1)
    plot(T,BV, 'b','LineWidth',2)
    ylabel ({'Battery',' Voltage','(V)'},'FontWeight','bold')
    xlim ([0 inf]); set(gca,'xtick',[])
```

```
figure ('Name','TE Vs Velocity','NumberTitle','off')
plot (V*3.6 ,TE/1e3,'b','LineWidth',2),grid off
xlabel ('Velocity (kph)','FontWeight','bold')
ylabel ('Tractive Force (kN)','FontWeight','bold')

function [ax,hlines] = Plot_yyy(x1,y1,x2,y2,x3,y3,x4,y4,varargin)
    ylabels = varargin{1};
    Xlabel  = varargin{2};
    FigTitle   = varargin{3};
    LEgend = varargin {4};
    if isempty(varargin{1})
        ylabels{1,:}=' ';
    elseif (nargin - length(varargin)) > 8
        error('Too many input arguments')
    elseif (nargin-length(varargin)) < 6
        error('Not enough input arguments')
    end
     figure('units','normalized',...
        'DefaultAxesXMinorTick','on','DefaultAxesYminorTick','on',...
        'Name',FigTitle,'NumberTitle','off');
    %Plot first two lines with plotyy
    [ax,hlines(1),hlines(2)] = plotyy(x1,y1,x2,y2);
    cfig = get(gcf,'color');
    pos = [0.1  0.1  0.7  0.8];
    offset = pos(3)/5.5;
    set (hlines(1),'LineWidth',1);
    set (hlines(2),'LineStyle','--','LineWidth',1.5);
    xlabel (Xlabel,'FontWeight','bold')
    xlim (ax(1),[0  x1(end)]); xlim (ax(2),[0  x1(end)]);
    %Reduce width of the two axes generated by plotyy
    pos(3) = pos(3) - offset/2;
    set(ax,'position',pos);

    %Determine the position of the third axes
    pos3=[pos(1) pos(2) pos(3)+offset pos(4)];

    %Determine the proper x-limits for the third axes
    limx1=get(ax(1),'xlim');
    limx3=[limx1(1)   limx1(1) + 1.2*(limx1(2)-limx1(1))];

    ax(3)=axes('Position',pos3,'box','off',...
        'Color','none','XColor','k','YColor','r',...
        'xtick',[],'xlim',limx3,'yaxislocation','right');

    hlines(3) = line(x3,y3,'Color','r','Parent',ax(3), ...
        'LineStyle','-',...
        'LineWidth',1.5);
    if (~isempty(x4) && ~isempty(y4))
        hlines(4) = line(x4,y4,'Color','r', ...
            'Parent',ax(3),'LineStyle',':',...
            'LineWidth',1.5);   % Same axis (ax(3))
    end
    limy3=get(ax(3),'YLim');
    line([limx1(2) limx3(2)],[limy3(1) limy3(1)],...
        'Color',cfig,'Parent',ax(3),'Clipping','off');
    axes(ax(2))
    %Label all three y-axes
    set(get(ax(1),'ylabel'),'string',ylabels{1},'FontWeight','bold')
    set(get(ax(2),'ylabel'),'string',ylabels{2},'FontWeight','bold')
    set(get(ax(3),'ylabel'),'string',ylabels{3},'FontWeight','bold')

    hleg = legend (hlines,LEgend{1:end-1},'location',LEgend{end});
    set(hleg,'FontSize',9,'FontAngle','italic')
```

D.3 Passenger Travel Cost and Emission Analysis

The MATLAB® scripts presented in this section computes travel cost and emission per passenger kilometer, and plot

```
function  PTFCEA (Ttrlelco, Ttzrelco)

if ~exist('Ttrlelco','var'),
    LV = load ('VarStore.mat','TramEE_trl') ;
    Ttrlelco = LV.TramEE_trl;
end
if ~exist('Ttzrelco','var'),
    LV = load ('VarStore.mat','TramEE_tzr') ;
    Ttzrelco = LV.TramEE_tzr;
end
Charger_eff       =       0.9;
diesel.price      =       1   ;      % (US$/ liter)
diesel.emission   =       2.68;      % CO2 emission (kg/liter)
electr.price      =       0.082;     %  (US$/kWh)
electr.emission   =       0.52  ;     %  kg/kWh
routedis.trl      =       12.05;     % km
routedis.tzr      =       21;        % km
%% ---- TRL Commuter Train
TRL_train.pcc = 1440;    % pcc: passenger carrying capacity
Tram_trl.pcc = 312;
TRL_train.dico  = 100;    % discon: diesel consumption 1_way trip
Tram_trl.elco   = Ttrlelco/Charger_eff;   %  kWh from grid 1_way trip

TRL_train.perfull_mo = [0,  0.9 , 0.5, 1, 0.6, 0.8];
TRL_train.perfull_ev = [0.9, 0.6, 1, 0.5, 0.8, 0.4];
% TRL_train.avrfull_mo =  mean(TRL_train.perfull_mo);
% TRL_train.avrfull_ev =  mean(TRL_train.perfull_ev);
TRL_train.avrfull        =        mean([TRL_train.perfull_mo        ,
TRL_train.perfull_ev]);
fprintf   ('TRL_Train   Average   Passengers   Full   %5.0f      (%2.2f
percent)\n',...
    TRL_train.avrfull*TRL_train.pcc,TRL_train.avrfull*100);
Tram_trl.avrfull = TRL_train.avrfull;
% fcpp : fuel cost per passenger
TRL_train.fcppkm     =  (TRL_train.dico * diesel.price)/...
    (TRL_train.avrfull*TRL_train.pcc*routedis.trl);
Tram_trl.fcppkm      =  (Tram_trl.elco * electr.price)/ ...
    (Tram_trl.avrfull*Tram_trl.pcc*routedis.trl);
% epp : emission per passenger
TRL_train.eppkm     =  (TRL_train.dico * diesel.emission)/...
```

```
    (TRL_train.avrfull*TRL_train.pcc*routedis.trl) ;
Tram_trl.eppkm      =  (Tram_trl.elco * electr.emission)/ ...
    (Tram_trl.avrfull*Tram_trl.pcc*routedis.trl);

% eppred & fccpred
Tram_trl.fcppkmred = 1 - Tram_trl.fcppkm /TRL_train.fcppkm;
Tram_trl.eppkmred = 1 - Tram_trl.eppkm /TRL_train.eppkm;

fprintf ('Compared to TRL_train IPEMU reduces:\n')
fprintf ('   ptfc by %2.2f perceent\n',Tram_trl.fcppkmred*100)
fprintf ('   CO2 emission by %2.2f percent\n',Tram_trl.eppkmred*100)

%% ---- TAZARA Commuter Train

TZR_train.pcc = 1200;
Tram_tzr.pcc = 312;
TZR_train.dico  = 70;
Tram_tzr.elco   = Ttzrelco/Charger_eff;

TZR_train.perfull_mo = [0,  1 , 0.2, 1, 0.2, 0.3];
TZR_train.perfull_ev = [0.3, 0.2, 0.9, 0.3, 0.9, 0.1];

TZR_train.avrfull        =        mean([TZR_train.perfull_mo        ,
TZR_train.perfull_ev]);
fprintf('\n\nTAZARA_Train  Average  Passengers  Full  %5.0f    (%2.2f
percent)\n',...
    TZR_train.avrfull*TZR_train.pcc,TZR_train.avrfull*100);
Tram_tzr.avrfull = TZR_train.avrfull;
% fcpp : fuel cost per passenger
TZR_train.fcppkm     =  (TZR_train.dico * diesel.price)/...
    (TZR_train.avrfull*TZR_train.pcc*routedis.tzr);
Tram_tzr.fcppkm      =  (Tram_tzr.elco * electr.price)/ ...
    (Tram_tzr.avrfull*Tram_tzr.pcc*routedis.tzr);
% epp : emission per passenger
TZR_train.eppkm     =  (TZR_train.dico * diesel.emission)/...
    (TZR_train.avrfull*TZR_train.pcc*routedis.tzr) ;
Tram_tzr.eppkm      =  (Tram_tzr.elco * electr.emission)/ ...
    (Tram_tzr.avrfull*Tram_tzr.pcc*routedis.tzr);

% eppred & fccpred
Tram_tzr.fcppkmred = 1 - Tram_tzr.fcppkm /TZR_train.fcppkm;
Tram_tzr.eppkmred = 1 - Tram_tzr.eppkm /TZR_train.eppkm;

fprintf ('Compared to TAZARA_train IPEMU reduces:\n')
fprintf ('   ptfc by %2.2f perceent\n',Tram_tzr.fcppkmred*100)
fprintf ('   CO2 emission by %2.2f percent\n',Tram_tzr.eppkmred*100)

% plot fcppkm
Yfcppkm   =   [TRL_train.fcppkm,   TZR_train.fcppkm   Tram_trl.fcppkm
Tram_tzr.fcppkm];
figure ('Name','Passenger Travel Fuel Cost','NumberTitle','off')
width = 0.4;
bar (1,Yfcppkm(1),width,'r'),hold on
bar (2,Yfcppkm(2),width,'b'),hold on
bar (3,Yfcppkm(3),width,'g'),hold on
bar (4,Yfcppkm(4),width,'c'),hold off
```

```
set(gca,'xtick',[]);
ylabel ('US$ per passenger km','FontWeight','bold')
hleg = legend ('TRL train','TAZARA train','BEMU,TRL route',...
    'BEMU,TAZARA route');
% set(hleg,'FontAngle','italic','TextColor',[.3,.2,.1])
set(hleg,'FontSize',9)

% plot eppkm
Yeppkm    =    [TRL_train.eppkm,    TZR_train.eppkm    Tram_trl.eppkm
Tram_tzr.eppkm];
figure ('Name','Passenger Travel Emission','NumberTitle','off')
bar (1,Yeppkm(1),width,'r'),hold on
bar (2,Yeppkm(2),width,'b'),hold on
bar (3,Yeppkm(3),width,'g'),hold on
bar (4,Yeppkm(4),width,'c'),hold off
set(gca,'xtick',[]);
ylabel ('kg CO_2 per passenger km','FontWeight','bold')
hleg = legend ('TRL train','TAZARA train','BEMU,TRL route',...
    'BEMU,TAZARA route');
set(hleg,'FontSize',9)
```

APPENDIX E

PUBLICATIONS

List of Publications

Mwambeleko, J.J., Kulworawanichpong, T., and Greyson, K. A. (2015). Tram and Trolleybus Net Traction Energy Consumption Comparison. **Proceedings of the IEEE 18th International Conference on Electrical Machines and Systems (ICEMS2015)**. Pattaya.

Kulworawanichpong, T., and Mwambeleko, J.J. (2015). Design and costing of a stand-alone solar photovoltaic system for a Tanzanian rural household. **Sustainable Energy Technologies and Assessments,** Vol. 12: 53-59.

Note:

This publication does not directly relate to the research. It has been included here just because it was done during the period of the Master Degree study

2015 18th International Conference on Electrical Machines and Systems (ICEMS), Oct. 25-28, 2015, Pattaya City, Thailand

Tram and Trolleybus Net Traction Energy Consumption Comparison

Joachim J. Mwambeleko[1], Thanatchai Kulworawanichpong[2], Kenedy A. Greyson[3].
[1]School of Electrical Engineering, Institute of Engineering, Suranaree University of Technology, Nakhon Ratchasima, Thailand.
[2]School of Electrical Engineering, Institute of Engineering, Suranaree University of Technology, Nakhon Ratchasima, Thailand.
[3]Department of Electronics and Telecommunications, Dar es Salaam Institute of Technology, Dar es Salaam, Tanzania.
E-mail: mwambejoachim@hotmail.com

Abstract — The need for energy efficient and environmentally friendly intracity transportation has now become very critical. Electric driven mass transportation systems have shown to be one of the solutions to traffic congestion and its related pollution level in most of the cities. Additionally, when compared with the traditional internal combustion engine (ICE), the electric driven mode of transport is much more efficient especially when regenerative braking is used. This paper took a step farther and compared two most common electric public transportation systems in cities. Traction energy consumption per passenger of a trolleybus was compared to that of a tram, both using regenerative braking, travelling from point A to point B, assuming the same route. For the sake of energy consumption comparison an 11.95 km route with eight (8) stations from Ubungo maziwa to Central railway station in Dar es Salaam, Tanzania was chosen. With their particular specifications such as effective mass and coefficient of friction, the two vehicles were modeled and simulated using MATLAB, in such a way their acceleration, braking deceleration, maximum speed, and travelling time were all the same. Their speed profiles which were more or less the same are presented in this paper alongside their corresponding energy consumption.

Keywords: Tram, Trolleybus, Traction energy consumption per passenger

I. Introduction

Growing pollution levels and traffic congestion in major cities are becoming delicate issues that could be eased by more efficient public transportation systems [1]. The need for energy efficient and environmentally friendly intracity transportation has never been more critical than it is today. Electric driven mass transportation systems have shown to be one of the solutions to traffic congestion and its related pollution level in most of the cities. In comparison with the traditional internal combustion engine (ICE), the electric driven mode of transport is much more efficient especially when regenerative braking system is used; as part of energy is recovered during braking when motors are used as generators and the magnetic torque provides the necessary braking force.

Apart from regenerative breaking, the electric driven system has higher efficiency than that of the traditional ICE driven system due to the high efficiency of the electric traction motors compared to that of the ICE. Energy transmission system (how energy is transferred to a driving axle) also plays a big role; as in electric driven mode, the energy transmission system has higher efficiency than the one in the traditional ICE driven mode. Generally the superiority of the electric driven systems over the traditional ICE driven systems is undeniable, and most cities are now shifting to electric mass transport systems.

Going into electric mass transport systems, most of the cities often opt for either trams or trolleybuses. Trams and trolleybuses are thus, very popular mode of transport in most of the cities nowadays. The choice as to whether city planners should go for trams or trolleybuses as means of public transport depends on many factors such as capital and life time running cost of trams and trolleybuses. As far as running cost is concerned, energy consumed by a tram or a trolleybus per passenger kilometer is a major contributing factor.

Almost in all cases a tram is heavier than a trolleybus in terms of weight per passenger. This might tempt and mislead people to conclude that the tram just because is heavier, it then consumes more energy per passenger kilometer than the trolleybus. Despite the fact that the tram is heavier, it is not necessarily true that it consumes more energy per passenger kilometer than the trolleybus, due to the fact that, on the other hand the tram having steel wheels running on steel rail has a rolling resistance that is far much lower than that of the trolleybus having rubber tyres running on tarmac or concrete road.

This paper then compares the traction energy per passenger consumed by a tram to that consumed by a trolleybus, both using regenerative braking and travelling the same distance from point A to point B, assuming the same route.

The rest of the content in this paper has been structured as follows: The motive of the paper is explained in section II, vehicle kinematics and vehicle dynamics are presented in section III and section IV respectively, the two altogether give the general overview on vehicle speed profiles and associated forces. Details associated with the tram and trolleybus, alongside the selected route are described in section V, the tram and trolleybus simulation results are presented and discussed in section VI, and finally a conclusion is drawn in section VII.

978-1-4799-8805-1/15/$31.00 ©2015 IEEE

II. Motive of the Paper

Because of the rising energy prices and environmental concerns, the choice of an energy-efficient mode of transport is extremely important. Trams and trolleybuses are by far the most preferable modes of public transport in cities; where a tram is always heavier than a trolleybus in terms of weight per passenger but, on the other hand the tram has lower coefficient of friction than the trolleybus. This paper then compares the energy consumption per passenger of the two modes of transport to see which of them uses less energy per passenger, taking the same route.

III. Vehicle Kinematics

When a vehicle travels from point A (Starting point) to point B (Stopping point) it generally passes through four stages namely i) Accelerating mode, ii) Constant speed mode, iii) Coasting mode, and iv) braking mode; as shown in Fig. 1.

Acceleration mode: Under the acceleration mode a vehicle accelerates from rest to a maximum operating speed (V_m). The tractive force (F_T) needed is given by $F_T = M\,a + F_R$; $F_T \leq F_{T\text{-}max}$, where M is the effective vehicle mass, a the acceleration, and F_R the net force opposing the motion. The tractive power drawn is such that to overcome the resistive forces and accelerate the vehicle.

Constant speed mode: Under the constant speed mode, a vehicle maintains the speed reached by the end of the accelerating mode, that is to say a vehicle's acceleration is zero. The tractive force (F_T) needed is given by $F_T = F_R$ where F_R is the net force opposing the motion. The vehicle draws tractive power only to overcome the resistive forces.

Coasting mode: In coasting mode a vehicle draws no tractive power, it moves by its own momentum, that is tractive force is zero ($F_T = 0$); due to forces opposing the motion, the vehicle starts decelerating; the vehicle's acceleration is thus given by $a = -F_R/M$.

Braking mode: In the braking mode, brakes are applied to stop the vehicle; braking system can be regenerative or non-regenerative. This paper considers a regenerative braking system, with which part of energy is recovered during braking by using the traction motors as generators and the vehicle's kinetic energy is converted into electrical energy.

A vehicle speed profile will always consist of the accelerating and braking modes but not necessarily the constant speed and coasting modes. Depending on the distance between point A (Starting point) and point B (Stopping point), and time limit for a vehicle to travel from point A to Point B, the constant speed mode or coasting mode or both may not be included in the vehicle speed profile. Fig. 2 shows a vehicle speed profile with only accelerating, coasting and braking modes.

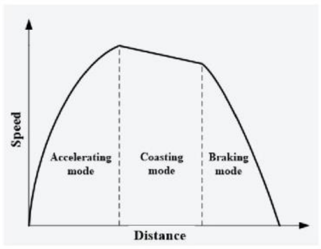


Fig. 2 Vehicle speed profile without a constant speed mode

IV. Vehicle Dynamics

Regardless of whether the vehicle is a simple private-car, bus or train, their movement calculations are basically the same, the differences between them will only be their respective parameters such as mass, maximum velocity, maximum tractive effort, coeficient of friction, aerodynamic drag coefficient and frontal area. A vehicle movement calculations are based on the Newton's laws of motion. A free body diagram showing the forces acting upon an uphill moving train is shown in Fig. 3.

The relationship between the forces is as expressed in (1) and (2)

$$F_T = ma + F_R \tag{1}$$

$$F_R = F_{RR} + F_{grad} + F_{drad} \tag{2}$$

where F_T is the vehicle tractive force (N), F_R is the net resistance force against the vehicle's motion (N), F_{RR} is the vehicle rolling resistance force (N), F_{drag} is the aerodynamic drag force (N), F_{grad} is the vehicle gravitational (gradient) force (N), it is positive when the vehicle is moving uphill and negative when the vehicle is moving downhill, m is the vehicle effective mass (kg), and a is the vehicle acceleration (ms^{-2}).

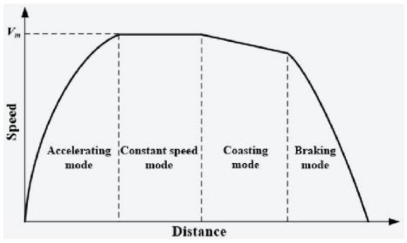


Fig. 1 General vehicle speed profile

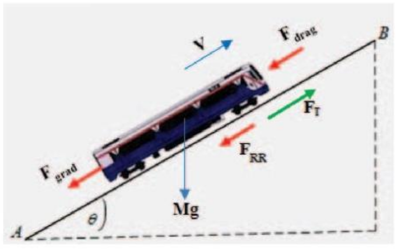


Fig. 3 Free body diagram of a vehicle going uphill

Tractive force, F_T: A vehicle accelerates through the application of tractive forces. The tractive force of a vehicle is produced at a tyre-road interface [2]; it is normally restricted in such a way $F_T \leq F_{T\_Max}$, where F_{T_Max} is the maximum tractive force; otherwise the vehicle wheels or tyres will be slipping instead of spinning. The maximum tractive force depends on the coefficient of adhesion between a wheel or tyre and a track or road respectively, the greater the coefficient of adhesion the greater the maximum tractive force.

Aerodynamic drag force, F_{drag}: The motion of a vehicle is taking place in the air and the force exerted by air on the vehicle will influence the motion. The aerodynamic resistance force results from three basic effects: i) the pressure different in front and behind the vehicle due to the separation of the air flow and the vortex creation behind the vehicle, ii) skin friction representing the surface roughness of the vehicle body and iii) internal flow of air entering the internal parts of the vehicle [3]. It is common to express the aerodynamic resistance force in the basic form as in (3).

$$F_{drag} = \frac{1}{2}\rho_{air} C_d A_f V_{air}^2 \tag{3}$$

where ρ_{air} is the air density (kgm^{-3}), C_d is an aerodynamic drag coefficient, A_f is the projected vehicle frontal area (m^2) and, V_{air} is the speed of air relative to the vehicle body (ms^{-1}) [3].

Gravitational (gradient) force, F_{grad}: The force depends on the inclination of the hill and weight of the vehicle, it is positive when the vehicle is moving uphill and negative when the vehicle is moving downhill, mathematically expressed as in (4)

$$F_{grad} = \pm mg\sin\theta \tag{4}$$

where m is the vehicle effective mass (kg), g is the acceleration due to gravity (9.81ms^{-2}) and, θ is the slope angle.

Rolling resistance, F_{RR}: is the resistance to motion of rotating parts. It can be categorized into two main resistances: i) frictional torques (bearing torques, gear teeth friction, brake pads) and ii) tyre deformation. At the most elementary level the rolling resistance of a moving vehicle is mathematically expressed as in (5) [4].

$$F_{RR} = \mu W \tag{5}$$

where W is the wheel load (N) and μ is the rolling resistance coefficient.

In some literatures, (3) and (5) are combined in one equation expressing the resistance to motion due to aerodynamics and coefficient of friction as a function of velocity, which is called Davis' equation as in [5].

Mathematically the approach to calculate the traction energy consumed by a vehicle moving from point A (Starting point) to point B (Stopping point) is as summarized in step (i) through step (vii) below.

i. Variables are initialized, e.g. $t = 0$; $v(t) = 0$; $s(t) = 0$.

ii. Vehicle acceleration (a (m/s^2)) is determined, which depends of the mode of operation.

iii. At every time step (dt), a value of velocity (v) and change in distance (ds) are calculated as $v(t) = v(t-dt) + adt$ (this shows that change in velocity is directly proportional to change in time when acceleration is constant) and $ds = v(t-dt) + 0.5adt^2$, where $v(t-dt)$ and $v(t)$ are previous and current velocities respectively.

iv. Resistance forces (rolling resistance force, aerodynamic drag force and, gradient force) are calculated, and then the tractive force (T_F) is obtained.

v. The output mechanical power (P_M), and input electrical power (P_E) are calculated as $P_M = T_F v$ and $P_E = \frac{P_M}{\eta}$ respectively and then electrical energy consumption (E) is computed as $E = P_E dt$; where η is the input electrical power (P_E) to the output mechanical power (P_M) conversion efficiency.

vi. The data are stored

vii. The values of time (t), velocity (v), and distance (s) are updated as $t = t + dt$, $v(t-dt) = v(t)$ and $s(t) = s(t-dt) + ds$ respectively.

viii. The cycle (step ii. to step vii.) repeats till a vehicle reaches a stopping point. The procedures are presented in a flow chart as shown in Fig. 4.

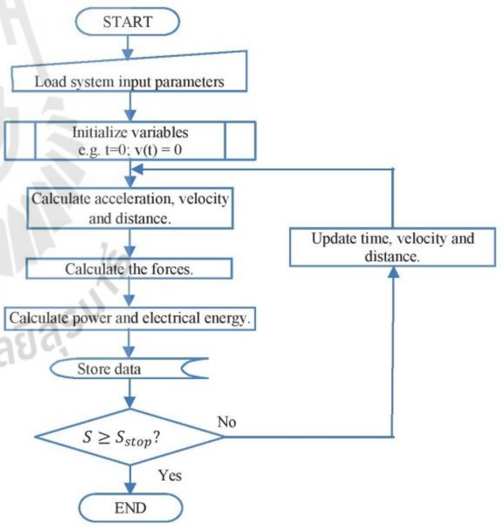

START
Load system input parameters
Initialize variables
e.g. t=0; v(t) = 0
Calculate acceleration, velocity and distance.
Calculate the forces.
Calculate power and electrical energy.
Store data
$S \geq S_{stop}$?
No
Yes
Update time, velocity and distance.
END

Fig.4 Vehicle energy consumption, calculation flow chart

V. Details Associated with the Tram and Trolleybus

For the purpose of traction energy per passenger comparison, Solaris trolleybus and Škoda tram as shown in Fig. 5 and Fig. 6 respectively, were chosen. The tram and trolleybus general details are as given in TABLE I.

Taking a person average weight as 60.7 kg [6], aerodynamic drag coefficient as 0.6 [1], a frontal area as 8.5 m^2 and 8.3 m^2 for a tram and trolleybus respectively, the simulation details associated with the tram and trolleybus were as given in TABLE I.

For the sake of traction energy consumption comparison, the tram route from Ubungo maziwa to Central railway station in Dar es Salaam Tanzania which is fairly flat was chosen for both tram and trolleybus (assuming the trolleybus has exactly similar route). As shown in Fig. 7, the route is 11.95 km long, having eight (8) stations.

Fig. 5 Solaris Trollino 18 trolleybus [11]

Fig. 6 Škoda 10 T tram [7]

TABLE I
TRAM AND TROLLEYBUS DETAILS [9], [10], [11]

Attribute	Value	
	Tram	Trolleybus
General details		
Maximum speed	70 km/h	70km/h
Length	20 m	18 m
width	2. 46 m	2.55 m
height	3. 44 m	3.24 m
Vehicle empty weight	28800 kg	17700 kg
Details used in the simulation		
Maximum operational speed	50 km/h	50 km/h
Acceleration	0.5 m/s^2	0.5 m/s^2
Braking deceleration	0.5 m/s^2	0.5 m/s^2
Passenger capacity @ 3p/m^2	Approx. 148	Approx.138
Vehicle effective weight @ 60.7kg/p	37784 kg	26077 kg
Rolling resistance coefficient	0.006	0.015
Aerodynamic drag coefficient	0.6	0.6
Frontal area	8.5 m^2	8.3 m^2
Air density	1.225kg/m^3	1.225kg/m^3

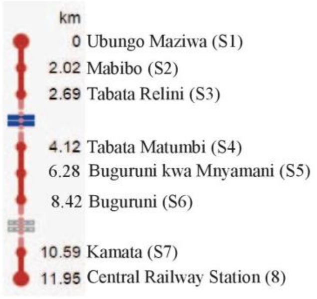


Fig. 7 Ubungo maziwa to Central railway station route

VI. Simulation Results and Discussion

When the two vehicles (i.e. Tram and Trolleybus) were modeled and simulated in such a way that their speed profiles were more or less the same, that is both vehicles accelerate at 0.5 ms^{-2} from rest to a maximum speed in acceleration mode, then they maintain the speed in constant speed mode. In the coasting mode the vehicles move by their own momenta, and lastly in braking mode brakes are applied and vehicles decelerate at 0.5 ms^{-2} to rest; such that the vehicles cover the same distance 11.95 km, in 1461 seconds (approx. 24 minutes) as shown in Fig. 8. At each station along the way, the vehicles stop for 15 seconds, i.e. the station stopping time (STT) was taken as 15 seconds.

With power profiles as shown in Fig. 9, the tram used approx. 11.21 kWh while the trolleybus used approx. 18.37 kWh as tabulated in TABLE 2.

Given passenger carrying capacities in TABLE I the tram used approx. 75.74 Wh per passenger while the trolleybus used approx. 133.12 Wh per passenger. That is to say the tram consumes 57 percent of the energy consumed by the trolleybus per passenger. Assuming the electricity cost is 0.15 \$/kWh, then the tram costs approx. 0.0114 \$ per passenger while the trolleybus cost approx. 0.02 \$ per passenger. That is, the tram costs 57 percent of the cost of the trolleybus per passenger.

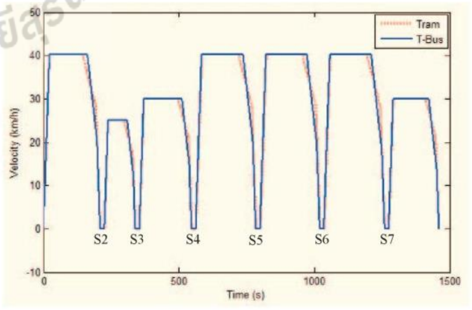


Fig. 8 Tram and Trolleybus speed profiles

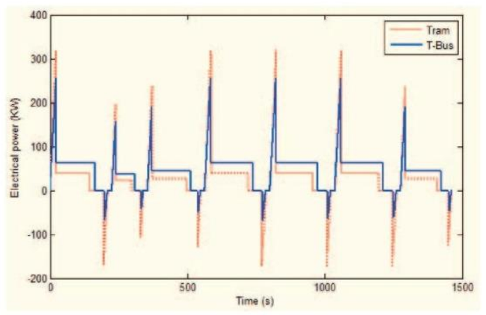


Fig. 9 Tram and Trolleybus power profile

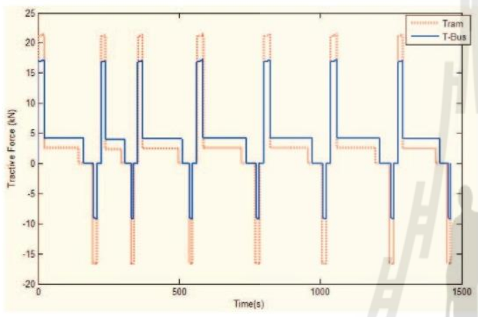


Fig. 10 Tram and Trolleybus Tractive Force Profile

TABLE 2
ENERGY CONSUMED BETWEEN STATIONS AND TIME TAKEN INCLUDING STT

Stations	Distance (km)	Energy Consumed (kWh)		Time (min)
		Tram	**Trolleybus**	
S1 – S2	2.02	1.93	3.14	3.75
S2 – S3	0.67	0.59	0.99	2.16
S3 – S4	1.43	1.29	2.14	3.47
S4 – S5	2.16	2.06	3.36	3.96
S5 – S6	2.14	2.05	3.33	3.93
S6 – S7	2.17	2.07	3.37	3.99
S7 – S8	1.36	1.22	2.04	3.09
Total	**11.95**	**11.21**	**18.37**	**24.35**

Though the tram has higher peak power than the trolleybus as shown in Fig. 9, yet it ends up consuming lesser net energy than the trolleybus. This is due to two main reasons:-

First; despite the tram being heavier yet it has lower rolling resistance than the trolleybus. This gives the tram an advantage over the trolleybus especially when the vehicles are not accelerating, this can be realized from (1). For example; when travelling form station 1 (S1) to station 2 (S2), when the vehicles are in a constant speed mode, the tractive force needed is only to counterbalance the forces opposing the motion, the tractive force (F_T) is thus given as $F_T = F_R$ where F_R is the net force opposing the motion as given in (2). As assumed the track is flat, with details as shown in TABLE I, the traction force needed by the tram is approx. 2.6 kN while that needed by the trolleybus is approx. 4.2 kN as shown in Fig. 10.

On the other hand when the vehicles are in acceleration mode, the tractive force (F_T) is as given in (1), and the tractive force needed by the tram is higher than that needed by the trolleybus at any given time with the acceleration of 0.5 ms^{-2} as shown in Fig. 9. The trolleybus will in turn consume more energy than the tram in the constant speed mode, and vice versa in the acceleration mode, the energy consumed has to do with not only the tractive force, but also velocity and time, as in (6).

$$dE = F_T(t) \times v(t) \times dt \qquad (6)$$

where dE and dt are changes in energy and time respectively, $F_T(t)$ and $v(t)$ are tractive force and velocity respectively. In this particular case the constant speed mode last longer, with tram tractive effort lower than that of the trolleybus. This shows that vehicle speed profile has a significant effect on energy consumption.

Second; the fact that the tram being heavier than the trolleybus, with the same maximum velocity, the tram has more kinetic energy than the trolleybus, thus in the coasting mode (when the power is disconnected and a vehicle is moving by its own momentum), the tram has higher kinetic energy and lower coefficient of friction than the trolleybus; it can then travel a longer distance in the coasting mode, and even when the brakes are applied, the tram recovers more energy than the trolleybus. The trolleybus on the other hand has to travel longer in the constant speed mode, and during braking it recovers lesser energy, these can be seen in Fig. 8 and Fig. 9.

VII. CONCLUSION

In this paper vehicle dynamics and mathematical approach in computing net traction energy consumed by a vehicle travelling from point A (starting point) to point B (stopping point) were highlighted. And traction energy per passenger consumed by the tram was compared to the traction energy per passenger consumed by the trolleybus, both using regenerative braking, and travelling the same distance, assuming the same route. It was found that with their respective details (such as effective mass, weight, and passenger carrying capacity); despite the tram being heavier than the trolleybus in terms of weight per passenger, yet by taking the advantage of low coefficient of rolling friction during constant speed mode, and high kinetic energy during coasting mode, the tram was found to consume only 57 percent of the energy consumed by the trolleybus per passenger. Assuming the electricity cost is 0.15 \$/kWh, then the tram costs approx. 0.0114 \$ per passenger while the trolleybus cost approx. 0.02 \$ per passenger. That is, the tram costs 57 percent of the cost of the trolleybus per passenger. In such a case then, the tram was a better choice.

REFERENCES

[1] R. Barrero, J. V. Mierlo and X. Tackoen, "Energy savings in Public Transport," *Vehicular Technology Magazine, IEEE,* vol. III, no. 3, pp. 26 - 36, 2008.

[2] B. Mashadi and D. Crolla, Vehicle Powertrain Systems, Wiley, 2012.

[3] S. Punpaisan and T. Kulworawanichpong, "Dynamic Simulation of Electric Bus Vehicle," *Standard International Journals (SIJ),* vol. II, pp. 99-104, MAy 2014.

[4] T. D. Gillespie, Fundametals of vehicle dynamics, Warrendale, PA : Society of Automotive Engineers, 1992.

[5] S. Lu, S. Hillmansen, T. k. Ho and C. Robert, "Single Train Trajectory Optimization," *IEEE Transactions On Intelligent Transportation Systems,* vol. XII, no. 2, pp. 743-750, June 2013.

[6] Wikipedia, October 2014. [Online]. Available: http://en.wikipedia.org/wiki/Body_weight.

[7] Wikipedia, "Škoda 10 T," Wikipedia, 10 March 2014. [Online]. Available: https://en.wikipedia.org/wiki/%C5%A0koda_10_T. [Accessed 15 April 2015].

[8] Škoda Transportation, "Tramcar Elektra Portland," 2004. [Online]. Available: http://www.skoda.cz/en/products/tramcars/elektra/tramcar-10-t/Contents.3/0/DE4BEDD0F49126E734D278A8BC2A1EFB/resource.pdf. [Accessed 10 April 2015].

[9] A. Bruce, "Trolleybuss to cut carbon emissions now," The Electric Tbus Group, 23 June 2013. [Online]. Available: http://www.tbus.org.uk/models.htm. [Accessed 13 June 2015].

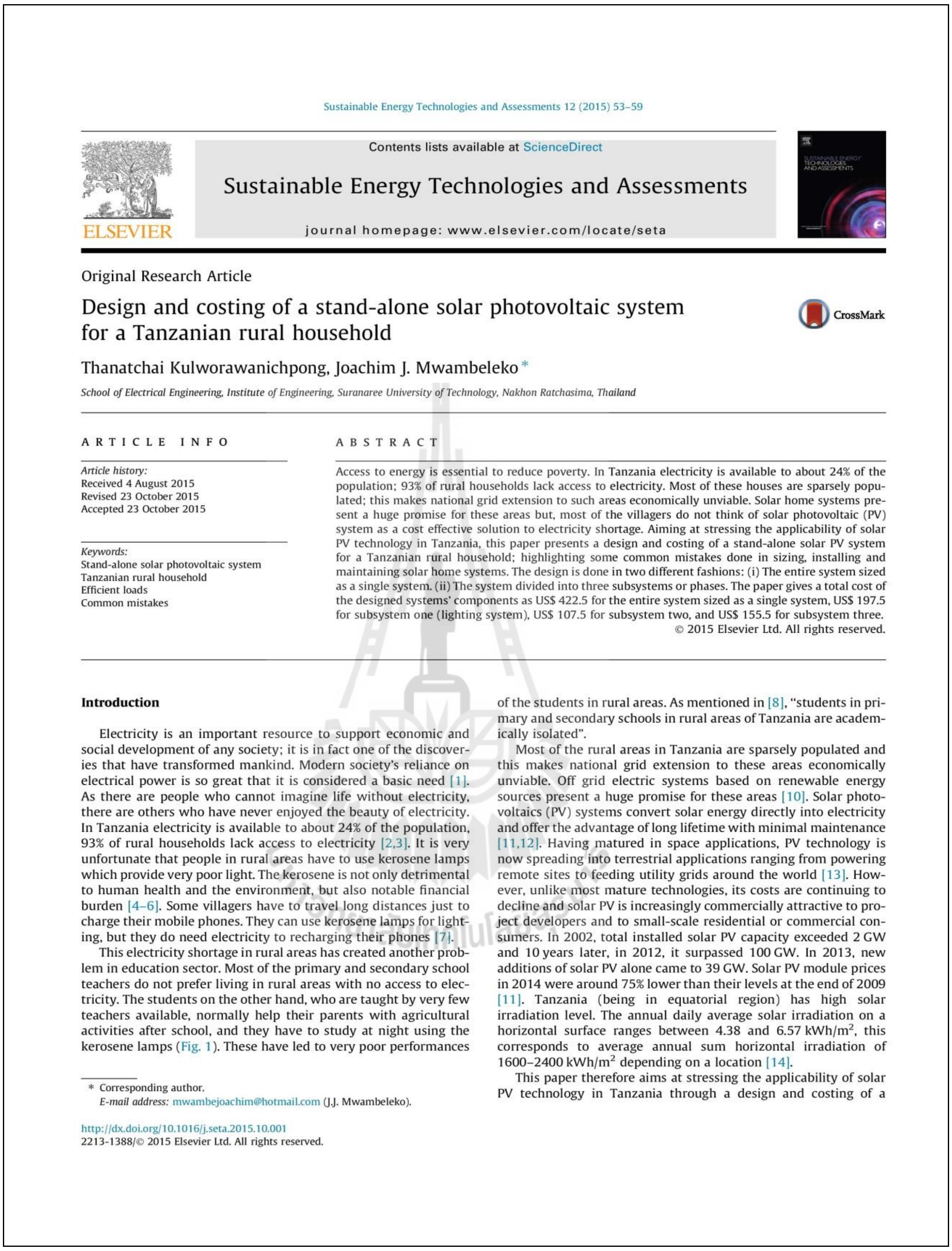
[10] Wikipedia, November 2014. [Online]. Available: http://en.wikipedia.org/wiki/Dar_es_Salaam_commuter_rail.

[11] Austria-In-Motion.net , "Solaris Trollino 18 MetroStyle," Austria-In-Motion.net , 2015. [Online]. Available: http://austria-in-motion.net/Austria/Salzburg/Salzburg-Stadt-und-Umgebung/Verkehrsmittel-Uebersicht/VKM-Obus/Obus-Salzburg/Fuhrpark/Solaris-Obusse-MetroStyle.html. [Accessed 5 July 2015].

[12] J. Larminie and J. Lowry, Electric Vehicle Technology Explained, West Sussex: John Wiley & Sons Ltd, 2003.

[13] R. Rajput, Utilisation of Electrical Power: Including Electrical Drives and Electric Traction, New Delhi: Laxmi Publications (P) ltd, 2006.

[14] A. Steimel, Electric Traction - Motive Power and Energy Supply, Munich: Oldenbourg Industrieverlag GmbH, 2008.



Sustainable Energy Technologies and Assessments 12 (2015) 53–59

Contents lists available at ScienceDirect

Sustainable Energy Technologies and Assessments

journal homepage: www.elsevier.com/locate/seta

Original Research Article

Design and costing of a stand-alone solar photovoltaic system for a Tanzanian rural household

Thanatchai Kulworawanichpong, Joachim J. Mwambeleko *

School of Electrical Engineering, Institute of Engineering, Suranaree University of Technology, Nakhon Ratchasima, Thailand

A R T I C L E I N F O

Article history:
Received 4 August 2015
Revised 23 October 2015
Accepted 23 October 2015

Keywords:
Stand-alone solar photovoltaic system
Tanzanian rural household
Efficient loads
Common mistakes

A B S T R A C T

Access to energy is essential to reduce poverty. In Tanzania electricity is available to about 24% of the population; 93% of rural households lack access to electricity. Most of these houses are sparsely populated; this makes national grid extension to such areas economically unviable. Solar home systems present a huge promise for these areas but, most of the villagers do not think of solar photovoltaic (PV) system as a cost effective solution to electricity shortage. Aiming at stressing the applicability of solar PV technology in Tanzania, this paper presents a design and costing of a stand-alone solar PV system for a Tanzanian rural household; highlighting some common mistakes done in sizing, installing and maintaining solar home systems. The design is done in two different fashions: (i) The entire system sized as a single system. (ii) The system divided into three subsystems or phases. The paper gives a total cost of the designed systems' components as US$ 422.5 for the entire system sized as a single system, US$ 197.5 for subsystem one (lighting system), US$ 107.5 for subsystem two, and US$ 155.5 for subsystem three.

© 2015 Elsevier Ltd. All rights reserved.

Introduction

Electricity is an important resource to support economic and social development of any society; it is in fact one of the discoveries that have transformed mankind. Modern society's reliance on electrical power is so great that it is considered a basic need [1]. As there are people who cannot imagine life without electricity, there are others who have never enjoyed the beauty of electricity. In Tanzania electricity is available to about 24% of the population, 93% of rural households lack access to electricity [2,3]. It is very unfortunate that people in rural areas have to use kerosene lamps which provide very poor light. The kerosene is not only detrimental to human health and the environment, but also notable financial burden [4–6]. Some villagers have to travel long distances just to charge their mobile phones. They can use kerosene lamps for lighting, but they do need electricity to recharging their phones [7].

This electricity shortage in rural areas has created another problem in education sector. Most of the primary and secondary school teachers do not prefer living in rural areas with no access to electricity. The students on the other hand, who are taught by very few teachers available, normally help their parents with agricultural activities after school, and they have to study at night using the kerosene lamps (Fig. 1). These have led to very poor performances of the students in rural areas. As mentioned in [8], "students in primary and secondary schools in rural areas of Tanzania are academically isolated".

Most of the rural areas in Tanzania are sparsely populated and this makes national grid extension to these areas economically unviable. Off grid electric systems based on renewable energy sources present a huge promise for these areas [10]. Solar photovoltaics (PV) systems convert solar energy directly into electricity and offer the advantage of long lifetime with minimal maintenance [11,12]. Having matured in space applications, PV technology is now spreading into terrestrial applications ranging from powering remote sites to feeding utility grids around the world [13]. However, unlike most mature technologies, its costs are continuing to decline and solar PV is increasingly commercially attractive to project developers and to small-scale residential or commercial consumers. In 2002, total installed solar PV capacity exceeded 2 GW and 10 years later, in 2012, it surpassed 100 GW. In 2013, new additions of solar PV alone came to 39 GW. Solar PV module prices in 2014 were around 75% lower than their levels at the end of 2009 [11]. Tanzania (being in equatorial region) has high solar irradiation level. The annual daily average solar irradiation on a horizontal surface ranges between 4.38 and 6.57 kWh/m^2, this corresponds to average annual sum horizontal irradiation of 1600–2400 kWh/m^2 depending on a location [14].

This paper therefore aims at stressing the applicability of solar PV technology in Tanzania through a design and costing of a

* Corresponding author.
E-mail address: mwambejoachim@hotmail.com (J.J. Mwambeleko).

http://dx.doi.org/10.1016/j.seta.2015.10.001
2213-1388/© 2015 Elsevier Ltd. All rights reserved.

Fig. 1. Students studying using a local kerosene lamp [9].

stand-alone solar PV system for a typical Tanzanian rural household. The house is assumed to be having DC lighting system with energy efficient light emitting diode (LED) lamps; and AC loads such as TV, DVD player, digital satellite receiver, small music system and fan.

A stand-alone solar PV system

As shown in Fig. 2 a stand-alone solar PV system for a typical rural household is expected to comprise the following:

(a) Solar module(s);
(b) charge controller;
(c) storage system (batteries);
(d) inverter (if the system includes ac load); and
(e) DC and/or AC load(s).

Solar module

Solar modules are made of several PV cells connected in series and parallel circuits. The solar array or panel is then a group of several modules electrically connected in a series–parallel combination to generate the required current and voltage [13,15]. The underlying operating principle of a PV cell is the photoelectric effect, by which radiation of photons of greater energy than the bandgap of the semiconductor material excite free electrons. When the PV circuit is closed, the freed electrons flow through an external circuit generating a DC current. Therefore the current generated is directly dependent on the number of incoming photons and thus, the solar irradiation [13].

It is perhaps worth mentioning that in a string of modules, a shaded module reduces the overall voltage of the string instead of adding to it. To overcome this effect, a bypass diode is usually added in parallel to each module. Similarly, in parallel-connected modules or strings, a diode is connected in series to each string to prevent reverse currents flowing into the lower voltage string [16]. Besides reducing shading losses the bypass diodes are also used to prevent the existence of hotspots [17].

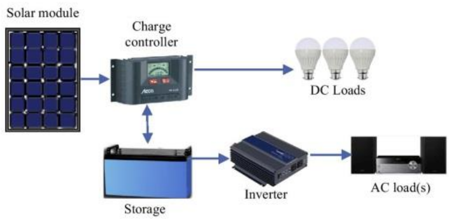


Fig. 2. A stand-alone solar PV system with dc and ac loads.

Charge controller

The success of any off-grid PV system depends to a large extent on the long-term performance of the batteries. For a system to operate well and have a long lifetime, the batteries must be charged properly and kept at high state of charge. A charge controller regulates power from a PV module to prevent batteries from overcharging and unacceptable voltage levels; it also functions as a low-voltage disconnect to disconnect DC load from the battery, preventing the batteries from over-discharge [18,19].

Given a solar irradiation level and PV cell temperature, maximum power from a PV module is harvested at a certain voltage known as maximum power voltage (V_{mp}). The V_{mp} is highly affected by changes in PV cell temperature. Normally, controllers allow the battery voltage to determine the operating voltage of a PV system. However, the battery voltage may not be the optimum PV operating voltage [19]. A maximum power point tracking (MPPT) charge controller can control PV output voltage such that maximum power is harvested.

Storage battery

Solar resource being intermittent in nature, stand-alone solar PV system is usually coupled with energy storage devices to ensure reliable supply. The most commonly used energy storage device is a battery, mainly lead acid type [12]. Lead-acid batteries are used almost exclusively in photovoltaic stand-alone systems. Although most other types of batteries have advantages such as high storage density or lower self-discharge, the decisive advantage of the lead battery is its lower price [17]. Batteries make up the largest component cost over the lifetime of a stand-alone solar PV system [18].

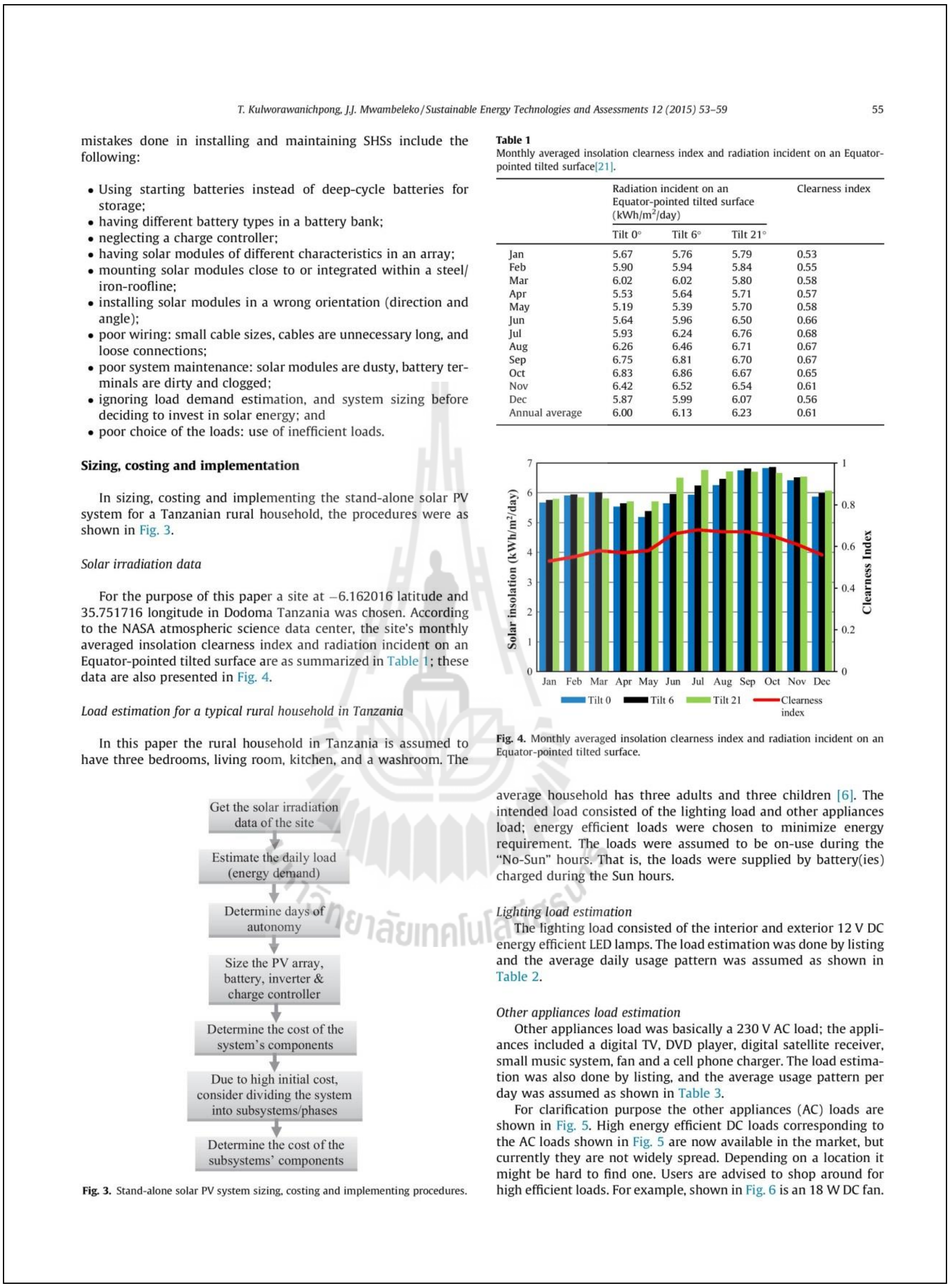
It might be worth mentioning at this point that, a normal car battery (starter battery) is not suitable as storage in a stand-alone solar plant as it would become defective in a short period due to the cycle operation. The battery is constructed to operate in buffer mode, for most of the time it is fully charged but occasionally must deliver short-term high currents to start the engine [17]. A battery to be used in a stand-alone solar PV system is charged during sun hours, and has to withstand deep discharge during the no-sun hours; such batteries are called deep cycle battery. Different requirements affect the construction of the two different types. In a battery bank, batteries should be of the same type and manufacturer, and about the same age. Old or poorly performing batteries decrease the performance of those to which they are connected [18].

Inverter

An inverter is included in the stand-alone solar PV system to convert the DC into AC electricity. The inverter needs to meet two needs: peak (or surge) power and continuous power. Some appliances, particularly those with electric motors, need a much higher power level at startup than they do when running [1]. Pumps, refrigerators, and blenders are common examples.

Common mistakes done in installing and maintaining solar home systems

In rural areas of Tanzania, it is common to see solar home systems (SHSs) that have been locally installed (not installed by a qualified technician), many of which are inefficient and some unsuccessful. As mentioned in [20], examples of the common



mistakes done in installing and maintaining SHSs include the following:

- Using starting batteries instead of deep-cycle batteries for storage;
- having different battery types in a battery bank;
- neglecting a charge controller;
- having solar modules of different characteristics in an array;
- mounting solar modules close to or integrated within a steel/iron-roofline;
- installing solar modules in a wrong orientation (direction and angle);
- poor wiring: small cable sizes, cables are unnecessary long, and loose connections;
- poor system maintenance: solar modules are dusty, battery terminals are dirty and clogged;
- ignoring load demand estimation, and system sizing before deciding to invest in solar energy; and
- poor choice of the loads: use of inefficient loads.

Sizing, costing and implementation

In sizing, costing and implementing the stand-alone solar PV system for a Tanzanian rural household, the procedures were as shown in Fig. 3.

Solar irradiation data

For the purpose of this paper a site at −6.162016 latitude and 35.751716 longitude in Dodoma Tanzania was chosen. According to the NASA atmospheric science data center, the site's monthly averaged insolation clearness index and radiation incident on an Equator-pointed tilted surface are as summarized in Table 1; these data are also presented in Fig. 4.

Load estimation for a typical rural household in Tanzania

In this paper the rural household in Tanzania is assumed to have three bedrooms, living room, kitchen, and a washroom. The average household has three adults and three children [6]. The intended load consisted of the lighting load and other appliances load; energy efficient loads were chosen to minimize energy requirement. The loads were assumed to be on-use during the "No-Sun" hours. That is, the loads were supplied by battery(ies) charged during the Sun hours.

Table 1
Monthly averaged insolation clearness index and radiation incident on an Equator-pointed tilted surface[21].

	Radiation incident on an Equator-pointed tilted surface ($kWh/m^2/day$)			Clearness index
	Tilt 0°	Tilt 6°	Tilt 21°	
Jan	5.67	5.76	5.79	0.53
Feb	5.90	5.94	5.84	0.55
Mar	6.02	6.02	5.80	0.58
Apr	5.53	5.64	5.71	0.57
May	5.19	5.39	5.70	0.58
Jun	5.64	5.96	6.50	0.66
Jul	5.93	6.24	6.76	0.68
Aug	6.26	6.46	6.71	0.67
Sep	6.75	6.81	6.70	0.67
Oct	6.83	6.86	6.67	0.65
Nov	6.42	6.52	6.54	0.61
Dec	5.87	5.99	6.07	0.56
Annual average	6.00	6.13	6.23	0.61

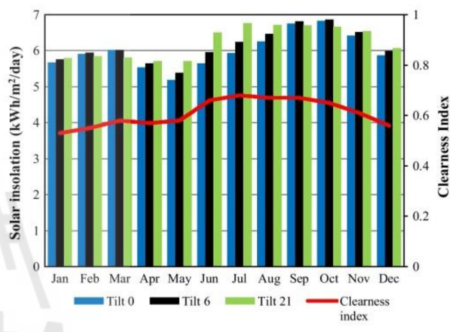


Fig. 4. Monthly averaged insolation clearness index and radiation incident on an Equator-pointed tilted surface.

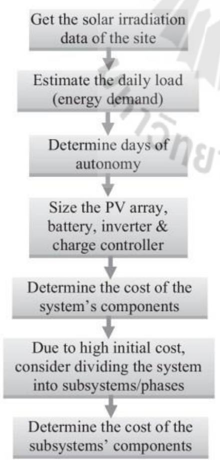

Get the solar irradiation data of the site
Estimate the daily load (energy demand)
Determine days of autonomy
Size the PV array, battery, inverter & charge controller
Determine the cost of the system's components
Due to high initial cost, consider dividing the system into subsystems/phases
Determine the cost of the subsystems' components

Fig. 3. Stand-alone solar PV system sizing, costing and implementing procedures.

Lighting load estimation

The lighting load consisted of the interior and exterior 12 V DC energy efficient LED lamps. The load estimation was done by listing and the average daily usage pattern was assumed as shown in Table 2.

Other appliances load estimation

Other appliances load was basically a 230 V AC load; the appliances included a digital TV, DVD player, digital satellite receiver, small music system, fan and a cell phone charger. The load estimation was also done by listing, and the average usage pattern per day was assumed as shown in Table 3.

For clarification purpose the other appliances (AC) loads are shown in Fig. 5. High energy efficient DC loads corresponding to the AC loads shown in Fig. 5 are now available in the market, but currently they are not widely spread. Depending on a location it might be hard to find one. Users are advised to shop around for high efficient loads. For example, shown in Fig. 6 is an 18 W DC fan.

Table 2
Rural household daily average lighting load with 12 V DC LED lamps.

	Room (location)	No. of lamps	Watt per lamp	Watt-lamp	Hours per day	Energy per day (Wh)
Interior lighting	Bedroom_1	2	3	6	2	12
	Bedroom_2	2	3	6	2	12
	Bedroom_3	2	3	6	2	12
	Kitchen	2	5	10	1.5	15
	Living room	2	5	10	4	40
	Wash room	1	3	3	2	6
	Corridor	1	3	3	4	12
Exterior lighting	Front	2	3	6	12	72
	Rear	2	3	6	12	72
Subtotal		16	31	56		253

Sizing the solar array

The solar array power rating (SAPR) was calculated using Eq. (1)

$$\text{SAPR (W)} = \frac{\text{DAD (Wh)}}{\text{PSH} * m_{\text{rf}} * \eta_{\text{cbc}} * \eta_{\text{inv}}} \tag{1}$$

where DAD is daily average demand, SAPR is solar array power rating, PSH is peak sun hours, m_{rf} is module output power reduction factor, η_{cbc} is the cables, battery, and charge controller efficiency, and η_{inv} is the inverter efficiency.

Sizing the battery pack

The battery pack capacity (BPC) was calculated using Eq. (2)

$$\text{BPC (Ah)} = \frac{\text{DAD (Wh)} * \text{AD}}{\text{BPNV} * \text{DOD}_a * \eta_{\text{cbc}} * \eta_{\text{inv}}} \tag{2}$$

where AD is autonomous days, BPC is the battery pack capacity, BPNV is the battery pack nominal voltage, and DOD_a is the allowable battery depth of discharge.

Sizing the inverter and charge controller

An inverter was included to converter 12 V DC to 230 V AC, so as to supply the AC loads. From Table 3, it is very likely all of the loads to be on at the same time, and a 1.25 safety factor was added, the inverter power rating was then obtained as a product of AC load and the safety factor. In a similar fashion was the charge controller sized, taking into account module short circuit current (Isc) and the maximum DC load current; in both cases a safety factor of 1.25 was included. The higher current between the two, determined the size of the controller.

Costing and implementation

Given the product list and the 2014 revised prices from Zara Solar shop at Nyerere-Sheikh Amin road, Mwanza-Tanzania; the appropriate sizes of the module(s), battery(ies), charge controller, and inverter(s) were chosen, and the system cost was computed as the total cost of the components.

According to the World Bank, the gross domestic product (GDP) per capita in Tanzania is approx. US$ 1000 [22]. It is likely to be lower than US$ 1000 for most of the people in rural areas. Realizing that, one of the main problems to the spread of SHSs in rural areas of Tanzania is the high initial cost; the system was sized and implemented in two different fashions. In the first scenario the entire system was sized as a single system, while in the second scenario the system was divided into three subsystems or phases; assuming a user can afford to purchase the subsystems in phases. The first subsystem was comprised of only the lighting load, as in most cases electricity for lighting is the number one priority for most of the people in rural areas. The second subsystem comprised some of the AC loads (music system, fan and cellphone charging), and the third subsystem was comprised of the TV, DVD player, and digital satellite receiver.

Results and discussion

System sizing and costing input data

The system sizing and costing was carried out under the assumptions/specifications as given in Table 4 and products list and prices as given in Table 5. Getting the daily average demand input from MS Excel, a program written in MATLAB was used to size the system, select the components and compute the system cost. For the purposes of this paper the given prices have been converted from TZS to US$ at a rate of US$ 1 = TZS 2000 and an extra column was added to compare the prices in a normalized fashion; hopefully the prices will keep going down as mentioned in Introduction.

Results

When the system was sized as a single system, the sizes of the components and system cost were as summarized in Table 6 and Fig. 7.

Table 3
Rural household daily average other appliances (AC) load.

Appliance	No. of units	Watt per unit	Watt-unit	Hours per day	Energy per day (Wh)
22″ LED TV	1	36	36	2.5	90
DVD player	1	15	15	0.5	7.5
Digital satellite receiver	1	10	10	2	20
Music system	1	35	35	1	35
Fan	1	45	45	0.5	22.5
Cell phone	2	3	6	0.5	3
Subtotal			147		178

Fig. 5. Chosen other appliances (AC) loads.

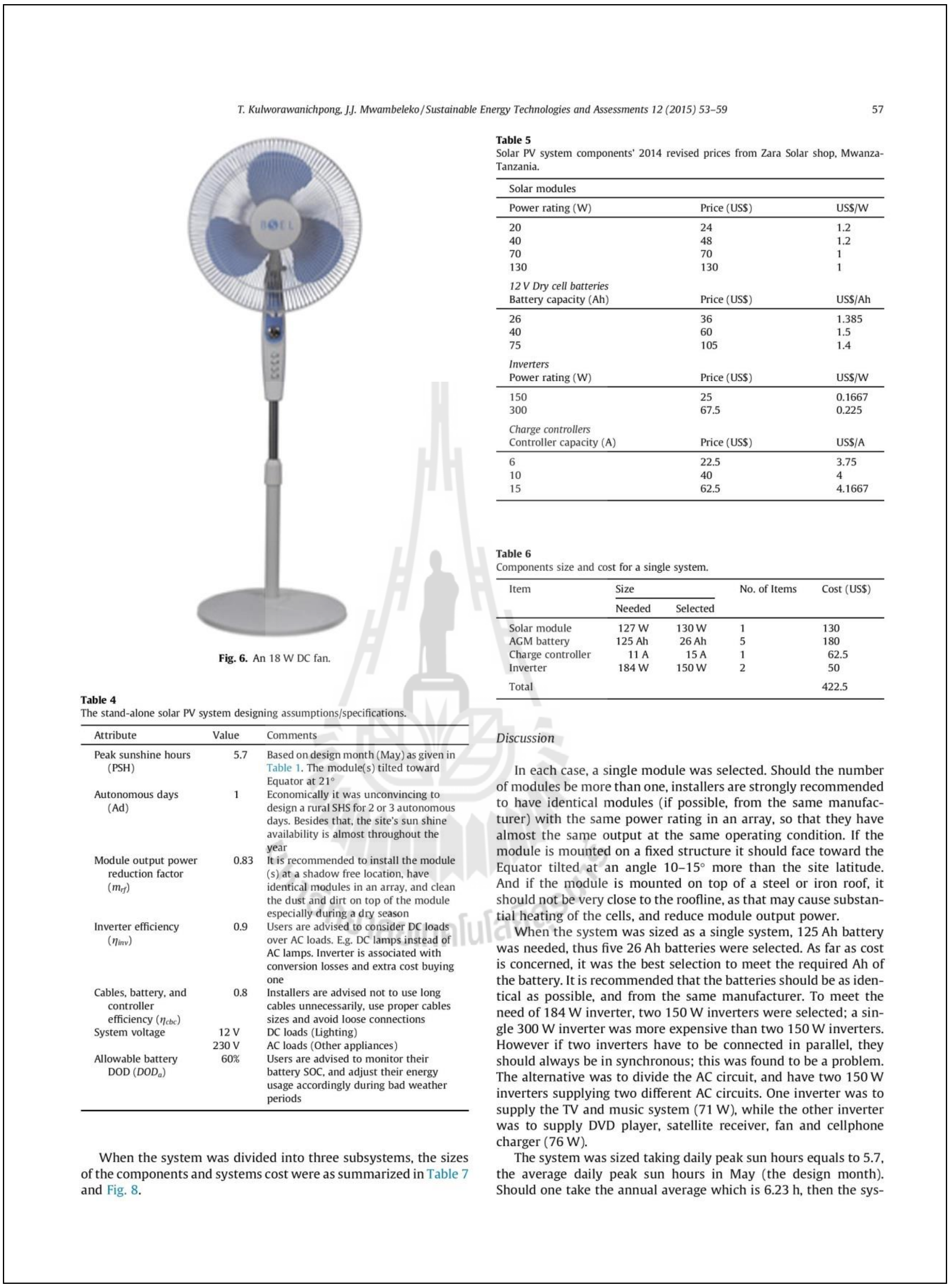


Fig. 6. An 18 W DC fan.

Table 4
The stand-alone solar PV system designing assumptions/specifications.

Attribute	Value	Comments
Peak sunshine hours (PSH)	5.7	Based on design month (May) as given in Table 1. The module(s) tilted toward Equator at 21°
Autonomous days (Ad)	1	Economically it was unconvincing to design a rural SHS for 2 or 3 autonomous days. Besides that, the site's sun shine availability is almost throughout the year
Module output power reduction factor (m_{rf})	0.83	It is recommended to install the module (s) at a shadow free location, have identical modules in an array, and clean the dust and dirt on top of the module especially during a dry season
Inverter efficiency (η_{inv})	0.9	Users are advised to consider DC loads over AC loads. E.g. DC lamps instead of AC lamps. Inverter is associated with conversion losses and extra cost buying one
Cables, battery, and controller efficiency (η_{cbc})	0.8	Installers are advised not to use long cables unnecessarily, use proper cables sizes and avoid loose connections
System voltage	12 V	DC loads (Lighting)
	230 V	AC loads (Other appliances)
Allowable battery DOD (DOD_a)	60%	Users are advised to monitor their battery SOC, and adjust their energy usage accordingly during bad weather periods

When the system was divided into three subsystems, the sizes of the components and systems cost were as summarized in Table 7 and Fig. 8.

Table 5
Solar PV system components' 2014 revised prices from Zara Solar shop, Mwanza-Tanzania.

Solar modules		
Power rating (W)	Price (US$)	US$/W
20	24	1.2
40	48	1.2
70	70	1
130	130	1
12 V Dry cell batteries		
Battery capacity (Ah)	Price (US$)	US$/Ah
26	36	1.385
40	60	1.5
75	105	1.4
Inverters		
Power rating (W)	Price (US$)	US$/W
150	25	0.1667
300	67.5	0.225
Charge controllers		
Controller capacity (A)	Price (US$)	US$/A
6	22.5	3.75
10	40	4
15	62.5	4.1667

Table 6
Components size and cost for a single system.

Item	Size		No. of Items	Cost (US$)
	Needed	Selected		
Solar module	127 W	130 W	1	130
AGM battery	125 Ah	26 Ah	5	180
Charge controller	11 A	15 A	1	62.5
Inverter	184 W	150 W	2	50
Total				422.5

Discussion

In each case, a single module was selected. Should the number of modules be more than one, installers are strongly recommended to have identical modules (if possible, from the same manufacturer) with the same power rating in an array, so that they have almost the same output at the same operating condition. If the module is mounted on a fixed structure it should face toward the Equator tilted at an angle 10–15° more than the site latitude. And if the module is mounted on top of a steel or iron roof, it should not be very close to the roofline, as that may cause substantial heating of the cells, and reduce module output power.

When the system was sized as a single system, 125 Ah battery was needed, thus five 26 Ah batteries were selected. As far as cost is concerned, it was the best selection to meet the required Ah of the battery. It is recommended that the batteries should be as identical as possible, and from the same manufacturer. To meet the need of 184 W inverter, two 150 W inverters were selected; a single 300 W inverter was more expensive than two 150 W inverters. However if two inverters have to be connected in parallel, they should always be in synchronous; this was found to be a problem. The alternative was to divide the AC circuit, and have two 150 W inverters supplying two different AC circuits. One inverter was to supply the TV and music system (71 W), while the other inverter was to supply DVD player, satellite receiver, fan and cellphone charger (76 W).

The system was sized taking daily peak sun hours equals to 5.7, the average daily peak sun hours in May (the design month). Should one take the annual average which is 6.23 h, then the sys-

Table 7
Size and cost of the subsystems components.

Item	Subsystem-1				Subsystem-2				Subsystem-3			
	Size		No. of Items	Cost (US $)	Size		No. of Items	Cost (US $)	Size		No. of Items	Cost (US $)
	Needed	Selected			Needed	Selected			Needed	Selected		
Solar module	67 W	70 W	1	70	18 W	20 W	1	24	35 W	40 W	1	48
AGM battery	66 Ah	75 Ah	1	105	18 Ah	26 Ah	1	36	34 Ah	40 Ah	1	60
Charge controller	6 A	6 A	1	22.5	2 A	6 A	1	22.5	4 A	6 A	1	22.5
Inverter	–		–	–	108 W	150 W	1	25	77 W	150	1	25
Total				197.5				107.5				155.5

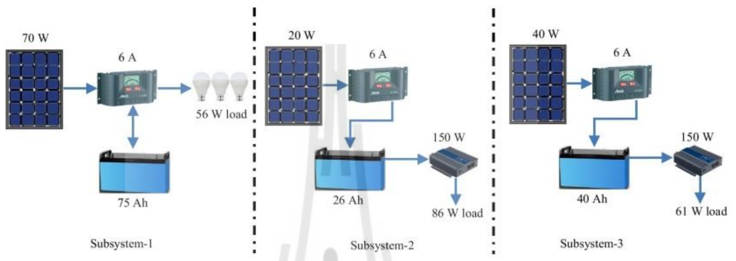


Fig. 8. Sizes of the three subsystems.

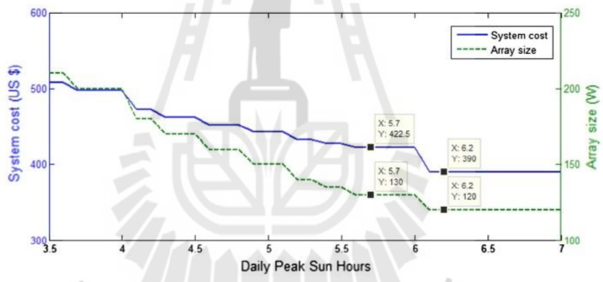


Fig. 9. System cost and array size variation with daily peak sun hours.

tem cost drops from US$ 422.5 to US$ 390. The peak sun hours affect the module size, which in turn affect the size of a charge controller (charge controller is sized taking into account the module short circuit current), all these effects are reflected in the system cost. With the given charge controllers and modules list and their prices, system cost and array size variation with daily peak sun hours is as shown in Fig. 9.

When the system was divided into three subsystems/phases, the sizes of modules and batteries of one subsystem were different from another subsystem, due to the mismatch, the subsystems should be independent. Though the overall cost of the subsystems is slightly higher than that of a single system (one complete system), dividing the system into phases is more convenient to individuals who cannot afford the complete system at once. In the first subsystem (lighting), energy needed for exterior lighting is more than half of the total energy demand. Installing the exterior lights with motion detectors can significantly reduce energy demand, and hence reduce system size.

As an inverter is associated with conversion losses and extra cost buying one, users are advised to consider DC loads over AC loads, such as DC lamps over AC lamps. And since motor loads require higher power level at startup than they do when running, turning off other loads when switching on a motor load is one tech-

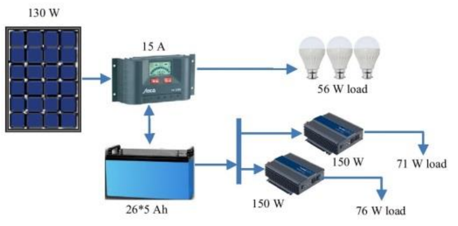
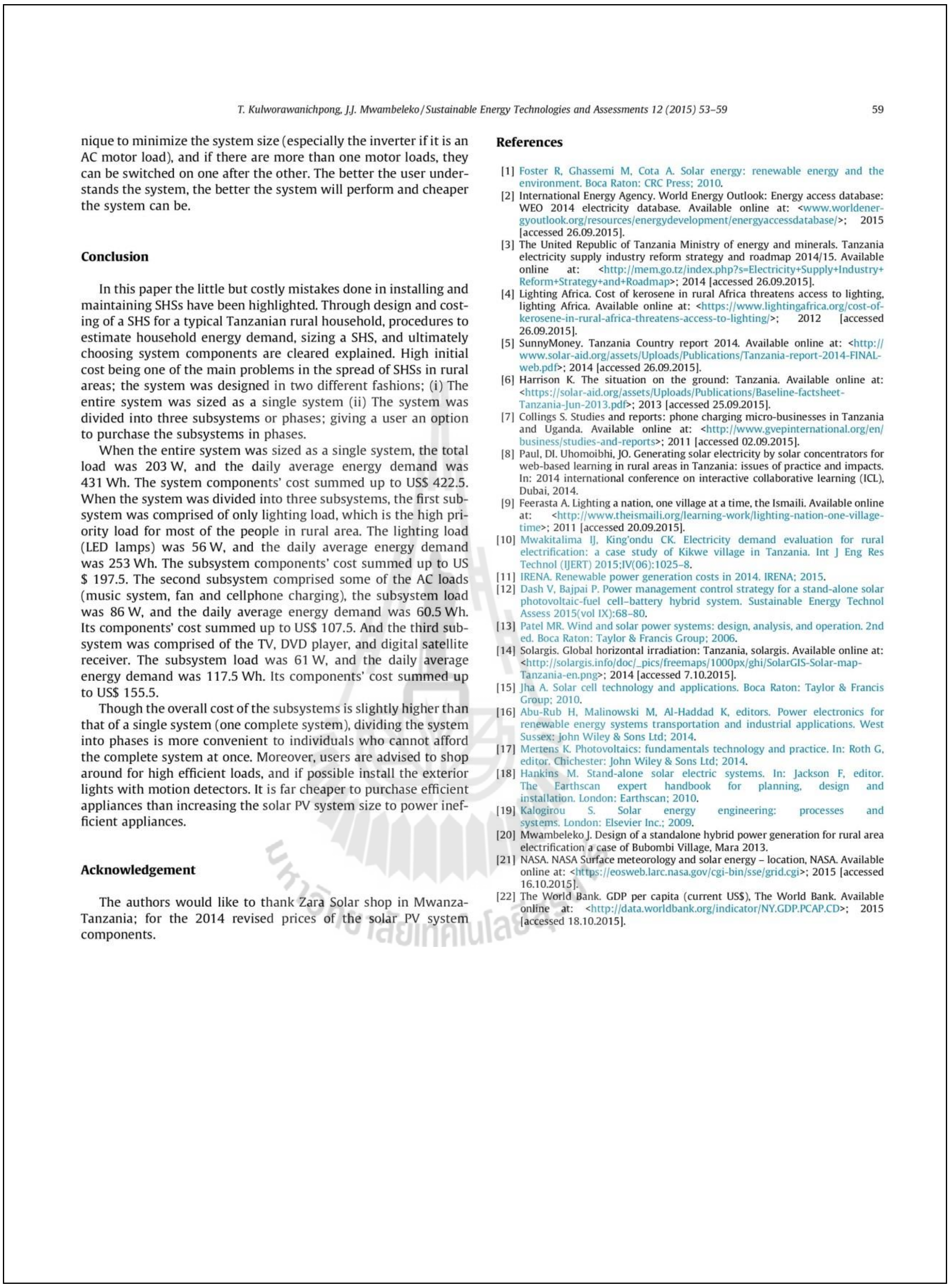


Fig. 7. Size of the system sized as a single system.



nique to minimize the system size (especially the inverter if it is an AC motor load), and if there are more than one motor loads, they can be switched on one after the other. The better the user understands the system, the better the system will perform and cheaper the system can be.

Conclusion

In this paper the little but costly mistakes done in installing and maintaining SHSs have been highlighted. Through design and costing of a SHS for a typical Tanzanian rural household, procedures to estimate household energy demand, sizing a SHS, and ultimately choosing system components are cleared explained. High initial cost being one of the main problems in the spread of SHSs in rural areas; the system was designed in two different fashions; (i) The entire system was sized as a single system (ii) The system was divided into three subsystems or phases; giving a user an option to purchase the subsystems in phases.

When the entire system was sized as a single system, the total load was 203 W, and the daily average energy demand was 431 Wh. The system components' cost summed up to US$ 422.5. When the system was divided into three subsystems, the first subsystem was comprised of only lighting load, which is the high priority load for most of the people in rural area. The lighting load (LED lamps) was 56 W, and the daily average energy demand was 253 Wh. The subsystem components' cost summed up to US $ 197.5. The second subsystem comprised some of the AC loads (music system, fan and cellphone charging), the subsystem load was 86 W, and the daily average energy demand was 60.5 Wh. Its components' cost summed up to US$ 107.5. And the third subsystem was comprised of the TV, DVD player, and digital satellite receiver. The subsystem load was 61 W, and the daily average energy demand was 117.5 Wh. Its components' cost summed up to US$ 155.5.

Though the overall cost of the subsystems is slightly higher than that of a single system (one complete system), dividing the system into phases is more convenient to individuals who cannot afford the complete system at once. Moreover, users are advised to shop around for high efficient loads, and if possible install the exterior lights with motion detectors. It is far cheaper to purchase efficient appliances than increasing the solar PV system size to power inefficient appliances.

Acknowledgement

The authors would like to thank Zara Solar shop in Mwanza-Tanzania; for the 2014 revised prices of the solar PV system components.

References

[1] Foster R, Ghassemi M, Cota A. Solar energy: renewable energy and the environment. Boca Raton: CRC Press; 2010.
[2] International Energy Agency. World Energy Outlook: Energy access database: WEO 2014 electricity database. Available online at: <www.worldenergyoutlook.org/resources/energydevelopment/energyaccessdatabase/>; 2015 [accessed 26.09.2015].
[3] The United Republic of Tanzania Ministry of energy and minerals. Tanzania electricity supply industry reform strategy and roadmap 2014/15. Available online at: <http://mem.go.tz/index.php?s=Electricity+Supply+Industry+Reform+Strategy+and+Roadmap>; 2014 [accessed 26.09.2015].
[4] Lighting Africa. Cost of kerosene in rural Africa threatens access to lighting, lighting Africa. Available online at: <https://www.lightingafrica.org/cost-of-kerosene-in-rural-africa-threatens-access-to-lighting/>; 2012 [accessed 26.09.2015].
[5] SunnyMoney. Tanzania Country report 2014. Available online at: <http://www.solar-aid.org/assets/Uploads/Publications/Tanzania-report-2014-FINAL-web.pdf>; 2014 [accessed 26.09.2015].
[6] Harrison K. The situation on the ground: Tanzania. Available online at: <https://solar-aid.org/assets/Uploads/Publications/Baseline-factsheet-Tanzania-Jun-2013.pdf>; 2013 [accessed 25.09.2015].
[7] Collings S. Studies and reports: phone charging micro-businesses in Tanzania and Uganda. Available online at: <http://www.gvepinternational.org/en/business/studies-and-reports>; 2011 [accessed 02.09.2015].
[8] Paul, DI. Uhomoibhi, JO. Generating solar electricity by solar concentrators for web-based learning in rural areas in Tanzania: issues of practice and impacts. In: 2014 international conference on interactive collaborative learning (ICL), Dubai, 2014.
[9] Feerasta A. Lighting a nation, one village at a time, the Ismaili. Available online at: <http://www.theismaili.org/learning-work/lighting-nation-one-village-time>; 2011 [accessed 20.09.2015].
[10] Mwakitalima IJ, King'ondu CK. Electricity demand evaluation for rural electrification: a case study of Kikwe village in Tanzania. Int J Eng Res Technol (IJERT) 2015;IV(06):1025–8.
[11] IRENA. Renewable power generation costs in 2014. IRENA; 2015.
[12] Dash V, Bajpai P. Power management control strategy for a stand-alone solar photovoltaic-fuel cell–battery hybrid system. Sustainable Energy Technol Assess 2015(vol IX):68–80.
[13] Patel MR. Wind and solar power systems: design, analysis, and operation. 2nd ed. Boca Raton: Taylor & Francis Group; 2006.
[14] Solargis. Global horizontal irradiation: Tanzania, solargis. Available online at: <http://solargis.info/doc/_pics/freemaps/1000px/ghi/SolarGIS-Solar-map-Tanzania-en.png>; 2014 [accessed 7.10.2015].
[15] Jha A. Solar cell technology and applications. Boca Raton: Taylor & Francis Group; 2010.
[16] Abu-Rub H, Malinowski M, Al-Haddad K, editors. Power electronics for renewable energy systems transportation and industrial applications. West Sussex: John Wiley & Sons Ltd; 2014.
[17] Mertens K. Photovoltaics: fundamentals technology and practice. In: Roth G, editor. Chichester: John Wiley & Sons Ltd; 2014.
[18] Hankins M. Stand-alone solar electric systems. In: Jackson F, editor. The Earthscan expert handbook for planning, design and installation. London: Earthscan; 2010.
[19] Kalogirou S. Solar energy engineering: processes and systems. London: Elsevier Inc.; 2009.
[20] Mwambeleko J. Design of a standalone hybrid power generation for rural area electrification a case of Bubombi Village, Mara 2013.
[21] NASA. NASA Surface meteorology and solar energy – location, NASA. Available online at: <https://eosweb.larc.nasa.gov/cgi-bin/sse/grid.cgi>; 2015 [accessed 16.10.2015].
[22] The World Bank. GDP per capita (current US$), The World Bank. Available online at: <http://data.worldbank.org/indicator/NY.GDP.PCAP.CD>; 2015 [accessed 18.10.2015].

BIOGRAPHY

Mr. Joachim Julius Mwambeleko was born on the 1st of March 1987 in Morogoro Region, Tanzania. He received his Bachelor of Science in Electrical Engineering from Saint Augustine University of Tanzania (SAUT), Mwanza, Tanzania in 2013. Currently he is a Master Degree student at the School of Electrical Engineering, Suranaree University of Technology (SUT), Nakhon Ratchasima, Thailand. He has published a technical paper related to electric traction, titled tram and trolleybus net traction energy consumption comparison in the Proceedings of the IEEE 18th International Conference on Electrical Machines and Systems (ICEMS2015), Pattaya, Thailand. He also has published a technical paper related to solar energy, titled design and costing of a stand-alone solar photovoltaic system for a Tanzanian rural household in the Sustainable Energy Technologies and Assessments journal.