OPTIMISING AC ELECTRIC RAILWAY POWER FLOWS WITH POWER ELECTRONIC CONTROL

by

THANATCHAI KULWORAWANICHPONG

A thesis submitted to
The University of Birmingham
For the degree of
DOCTOR OF PHILOSOPHY

Department of Electronic, Electrical and Computer Engineering
School of Engineering
The University of Birmingham
November 2003

SYNOPSIS

The latest generation of AC-fed traction drives, employing high-speed switching devices, is able to control the reactive power drawn from the overhead line by each equipment. If the conditions at each locomotive or train could be fed back to a central control point, it is possible for a centrally located controller to calculate optimal values for the reactive power in each drive and to send those commands back to the individual equipment.

In this thesis, AC railway power flows are optimised in real time and the results are used to achieve some particular system objective via control of the PWM equipment as mobile reactive power compensators. The system voltage profile and the total power losses can be improved while the overall power factor at the feeder substation is also made nearer to unity.

For off-line simulation purposes, high execution speeds and low storage requirements are not generally significant with the latest computer hardware. However, this real-time control employs on-line optimising controllers, which need embedded power solvers running many times faster than real time. Thus, a fast and efficient algorithm for AC railway power flow calculation was developed. The proposed scheme is compared to a conventional reactive power compensation, e.g. SVC, and found to be less expensive to implement.

Several test cases for AC electric railway systems are examined. The centralised area control system leads to the best improvement where an existing fleet of diode or thyristor phase-angle controlled locomotives is partially replaced with PWM ones, compared to that obtained without compensation or to classical track-side Var compensation methods. From these results, the potential for PWM locomotives to improve overall system performance is confirmed.

This thesis is dedicated to my family, KULWORAWANICHPONG, and

the memory of my grand mother

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Dr Colin GOODMAN, for his guidance, and invaluable help and advice throughout the course of study and in writing this thesis.

I would like to acknowledge financial support from the Energy Policy and Planning Office, Ministry of Energy, The Royal Thai Government, Thailand, and Suranaree University of Technology, Thailand, during the period of study.

I am also grateful to Dr Sarawut SUJITJORN for his encouragement and sincere help for the last ten years. Thanks are also extended to my colleges in the Power & Control Research Group, The University of Birmingham, UK, for their help and interest in this work.

I am eternally grateful to my parents and my family for their continually sincere support. Finally, I would like to express my gratitude to all whom I have failed to mention here, but have generously supported me.

TABLE OF CONTENTS

PART II

AC Railway Power Flow Calculation – Exploitation of Sparsity

PART III

Optimal Area Control System

PART IV

Cost Estimation and Design

LIST OF ILLUSTRATIONS

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Chapter 6

LIST OF TABLES

Chapter 3

Chapter 4

Chapter 5

LIST OF ABBREVIATIONS AND SYMBOLS

a	Substation’s transformer ratio
AC	Alternating Current
AT	Autotransformer
ATO	Automatic Train Operation
[B]	Bus admittance matrix with neglecting the line resistance
B_1, B_2	Jacobian sub-matrices with neglecting the resistance
BBO	Banded Bus-ordering Algorithm
BMC	Band Matrix Computation
BPF	Band-pass Filter
BT	Booster Transformer
C	Overhead Catenary Contact Wire
C-bank	Capacitor bank
CC	Capital cost
C_F	Link capacitance
CGM	Conjugate Gradient Method
DC	Direct Current
DCC	Direct Current Control for PWM power converters
DVR	Dynamic Voltage Restorer
E_{Loss}^{New}	New-valued energy losses
E_{Loss}^{Old}	Old-valued energy losses
EMI	Electromagnetic Interference
E_t	Voltage source of the train model
F	Feeder wire
f	Operating frequency, normally 50 Hz
FD	Feeding distance (Between SS and MPTSC)
FDCPFM	Fast Decoupled Power Flow Method
F_{drag}	Drag force or train resistance
F_{grad}	Gradient force
$\mathbf{F}_i$	Current mismatches at bus i

FMC	Full Matrix Computation
FS	Feeder Substation
g	Acceleration due to gravity (9.81 m/s^2)
G	Generation or Generator
GS, GSPFM	Gauss-Seidel and Gauss-Seidel Power Flow Method
HVDC	High Voltage Direct Current
Hz	Hertz
$\mathbf{I}$	Identity matrix
I_C, V_C	Current and voltage of the overhead catenary contact wire
I_F, V_F	Current and voltage of the feeder wire
I_R, V_R	Current and voltage of the rails
i_S	Converter current as a function of time
$\hat{I}_S$	Peak value of the converter current
I_S, V_S	rms values of I_S and V_S
$I_S = \frac{\hat{I}_S}{\sqrt{2}} \angle \phi$	Phasor of the converter current
$I_{T,i}$, $S_{TR,i}$	Actual current and power drawn by a train at bus i
ITSC	Intermediate Track-Sectioning Cabin
J_1, J_2, J_3, J_4	Jacobian sub-matrices for single-phase power systems
$\mathbf{J}_i$	Current vector at bus i
J_S	Norton current of the substation
$\mathbf{J}_{SS}$	Current matrix representing the substation
$\mathbf{J}_{SS,i}$	Current source vector of the substation (if i = SS)
J_t	Current source of the train model
$\mathbf{J}_{TR}$	Current matrix representing the train
$\mathbf{J}_{TR,i}$	Current load vector of a train connected at bus i
k_E	Cost of per-unit energy charge
km	Kilometre
k_P	Cost of per-unit peak power loss
kph	Kilometre Per Hour
k_Q	Cost of per-unit reactive power source

k_S	is the incremental cost of per-unit power
kV	Kilo-volt
L, **U**	Lower and upper diagonal matrix for LU factorisation
LIFE	Lifetime of the reactive power source to be installed
LPF	Low-pass Filter
L_S	Input inductance of the PWM converter
m	Modulation Index
M	Annual maintenance cost
$M_1 – M_4$, $L_1 – L_4$	Jacobian sub-matrices for bi-phase power systems
Max. Grad.	Maximum absolute value of the gradient vector
MC	Total maintenance cost
M_{eff}	Effective mass
M_{frgt}	Freight load
m_{min}, m_{max}	Minimum and maximum limit of the modulation index
M_{psg}	Passenger mass
MPTSC	Mid-Point Track-Sectioning Cabin
M_{tare}	Tare mass
MVA	Mega Volt-Ampere
MVar	Mega Volt-Ampere Reactive
N_A	Total number of additional equipment in the system
NR, NRPFM	Newton-Raphson and Newton-Raphson Power Flow Method
p.u.	Per-unit
PAC	Phase-angle Compensator
pf	Power Factor
$P_{i,sch}$, $Q_{i,sch}$	Scheduled real and reactive power at bus i-th, respectively
$\mathbf{p}_k$	Search direction at iteration k
P_{Loss}^{Old}, P_{Loss}^{New}	Peak power losses before and after installation
$\tilde{P}_S$	Constant real power demand
P_S, Q_S	Real & reactive power flows through the PWM converter
PWM	Pulse-Width-Modulated
Q_C	Size of reactive power sources to be installed

R	Rails
RC	Return conductor
R_f, L_f, C_f	Filter resistance, inductance and capacitance
SBO	Simple Bus-ordering Algorithm
$S_{D,i} = P_{D,i} + jQ_{D,i}$	Complex power drawn by load or powering train i
$S_{G,i} = P_{G,i} + jQ_{G,i}$	Complex power supplied by generator or re-generative train i
SPS	Static Phase Shifter
SS	Substation
STATCOM	Static Synchronous Compensator
SVC + UPF	Compensation with the SVC together with PWM locomotives
SVC	Static Var Compensator
TCR	Thyristor-controlled Reactor
TCSC	Thyristor-controlled Series Compensator
TE	Tractive Effort (N)
TR	Train
Tx	Feeder section (Transmission line)
UPF	Unity Power Factor
UPFC	Unified Power Flow Controller
$V_C = \frac{mV_{DC}}{\sqrt{2}}\angle\delta$	Phasor voltage constructed by the PWM converter
V_{DC}	Average DC link voltage
v_{DC}	DC link voltage as a function of time
$V_{DC,ref}$, $Q_{S,ref}$	Reference value for V_{DC} and Q_S
V_{grid}	Nominal voltage of the utility supply grid
$\mathbf{V}_i$	Voltage vector at bus i
$\lvert V_i \rvert$	Magnitude of the voltage V_i
V_S	Thevenin voltage of the substation
$V_S = \frac{\hat{V}_S}{\sqrt{2}}\angle 0^\circ$	Phasor of the secondary-side pantograph voltage
$\hat{V}_S$	Peak value of the secondary-side pantograph voltage
$\hat{V}_{S,max}$	Maximum value of $\hat{V}_S$

VSI	Voltage Source Inverter
V_{SS}	Substation terminal voltage
VVC	Voltage Vector Control for PWM power converters
X_S	Input reactance of the PWM converter, $X_S = 2\pi f L_S$
Y_{i0}	Admittance connected between bus i to ground
Y, [**Y**]	Bus admittance matrix for single- and bi-phase systems
$\mathbf{Y}_{AT}$	Admittance matrix representing the AT
$\mathbf{Y}_{ij}$	Admittance matrix representing the feeder portion i – j
$\lvert Y_{ki} \rvert$	Magnitude of the admittance Y_{ki}
y_S	Norton admittance
$\mathbf{Y}_{SS}$	Admittance matrix representing the substation
z_1	Primary leakage impedance of the substation's transformer
$\mathbf{z}_2$	Primary leakage impedance of the substation's transformer
$z_{a,i}$	Impedance of the i^{th} additional equipment on the system
z_{CC}	Self-impedance of the overhead catenary contact wire
z_{CF}, z_{FC}	Mutual-impedance between conductor C and F
z_{CR}, z_{RC}	Mutual-impedance between conductor C and R
z_e	Earthing impedance
z_{FF}	Self-impedance of the feeder wire
z_g	Leakage impedance of the AT
z_{grid}	Short-circuit impedance of the utility supply grid
$\mathbf{Z}_{ij}$	Impedance matrix representing the feeder portion i – j
z_l	Average lumped impedance of the feeding system
z_m	Magnetising impedance of the AT
z_{OH}	Average impedance per length of C, R and F conductors
z_{RF}, z_{FR}	Mutual-impedance between conductor R and F
z_{RR}	Self-impedance of the rails
z_S	Thevenin impedance
z_t, y_t	Impedance/admittance of the train model
α	Linear acceleration
β	Diagonal bandwidth
δ	Phase angle of modulating signal
ΔE_{Loss}	Energy loss reduction

δ_i	Angle of the voltage V_i
δ_{min}, δ_{max}	Minimum and maximum limit of the phase angle δ
ΔP_i, ΔQ_i	Real and reactive power mismatches at bus i-th, respectively
ΔP_{Loss}	Power loss reduction
ΔS	Increased power capacity
ε	Maximum error allowance
ϕ	Phase angle of the converter current
Φ_E	Return of the electricity charge
Φ_P	Benefit of the peak power-loss reduction
τ	Rotary allowance
Γ	Incremental cost
λ	Annual percentage incremental of the maintenance cost
λ_k	Step length at iteration k
Θ	Construction cost and property cost
θ_{ki}	Angle of the complex number Y_{ki}
Ψ	Conductor sequence given by $\Psi = \{C, R, F\}$
$\Re$	Vector space of real numbers
*	Complex conjugation

This page is intentionally left blank

CHAPTER 1

INTRODUCTION

1.1 Background

Any transportation system needs a source of power to drive its vehicles carrying their passengers or goods from one place to another. From the early invention of steam engines to modern electric locomotives, railway transport systems have had a long history and huge developments to become one of the most popular modes of public transport over the last century. The demand of railway transportation systems has considerably increased every year by passenger journey and goods delivery in both short and long distance services. To provide power propulsion and the need of higher-speed, more luxurious and more reliable services, electrification has been the first choice and widely applied to modernise most of the railway transport systems across the world for several decades. However, not many decades after this modernisation, the demand continually grows and nearly reaches the limit of traction power supplies that were previously designed. This significantly affects unstable, unreliable, inefficient, ineffective and uneconomical operations. Thus existing electric railway systems need to be upgraded.

Since high performance computers have been developed, emerged and available for commercial purposes, planning a new rail route and extending or upgrading any existing railway can be designed with less costs and time consumed. Computer modelling and simulation, nowadays, have consequently played essential roles not only for system design but also for system operation and control. In the present, due to the greater complication of modern power supplies and traction drives, more accurate computer-based models are required. This

will allow engineers to analyse and predict the railway system performance precisely and efficiently. Therefore, the electric railway systems will be effectively designed and upgraded.

Nevertheless, to upgrade the railway system is obviously critical. Technically, many related approaches have been taken into account. The replacement of feeding cables or substation transformers is one of many possible solutions to increase power-feeding capacity but this inevitably interrupts the services. The other approach that is widely applied to typical railway systems is the use of additional reactive power sources, e.g. SVC. Although these installations can improve system performance, they require spacious installing areas and are also costly. The development of a new upgrading technique in order to minimise cost while maximising system performance therefore challenges railway system engineers for the 21st century.

1.2 Aim of Research and Structure of Thesis

Variations of pantograph voltage directly affect the traction performance of electric traction equipment on-board a train and also power quality, stability and security of a whole power feeding section. Computer-based simulation and the theoretical study of railway electrification influences a modern railway system in several aspects. In this thesis, it is principally mainline railways with an overhead AC catenary feeding system that are considered. The aim of the research was to carry out power supply analyses based on computer modelling and simulation. The first part of this thesis involves AC railway power flow solution methods for efficient application to AC railway power feeding systems. New fast and efficient simulation software was developed. Secondly, power flow optimisation through an overhead catenary feeding system was examined. A multi-train simulation of several hours of service has also been developed as a main simulation structure. The last section is concerned with cost estimation of

an optimal area control system in comparison with the system upgraded by the traditional approach.

The objective of this thesis is to optimise AC railway power flows by controlling power-electronic equipment on-board the trains rather than controlling a huge and expensive SVC. The thesis is divided into four main parts as given below.

1. AC railway power flow calculation – Theory and application
2. AC railway power flow calculation – Exploitation of matrix sparsity
3. Optimal area control system
4. Cost estimation and design

The first part describes work to determine a more accurate, more efficient and faster power flow solution method. The fundamental work was to derive mathematical models of an AC railway power flow problem. Feeder substations, overhead catenary feeding sections and electric locomotives were carefully examined to evolve models of AC railway power supply systems in both the single-phase and bi-phase AC railway power feeding configurations. Only a combination of nodal analysis and a simple iterative algorithm is required to solve such a problem. This method can simplify problem complexity and also reduces the memory requirement of the developed computer program.

The second part relates to program development of AC railway system simulation programs. Development of a fast and efficient algorithm was the main objective. Careful study and comparison among some powerful algorithms for solving a set of linear equations was carried out. Ordering a system matrix into an appropriate form of sparse matrix can achieve the best benefit from its computing efficiency. Thus, a fast and powerful AC railway system simulation was developed.

The third part is concerned with optimisation problems. This section introduces and develops an optimal area control in order to minimise some particular objective that gives an

improved overall system performance, e.g. lower power loss and less system voltage deviation, etc. A computer-based simulation for optimising AC railway power flows is used to evaluate the possibility of PWM locomotive control acting as mobile reactive power compensation. With railway signalling and control technology, the optimal area control system is first introduced and employed to upgrade AC railway systems.

The last section describes a design consideration for reactive power compensation. Cost estimation of the SVC installation and the special design of PWM locomotives to be able to compensate a considerable amount of reactive power is illustrated. Optimal rating and location of the SVC is determined in order to minimise the installation cost and maximise the energy loss reduction.

1.3 Arrangements of Chapters

As stated previously, the thesis consists of four main parts. It is also arranged into 8 chapters as follows.

- **Chapter 2** provides a literature survey of history and related research works on railway electrification, AC railway power feeding technology, conventional power flow solution methods, computer-based modelling and simulation, and power system optimisation and control.
- **Chapter 3** describes a new AC railway power flow solution method. It gives the basic concept, mathematical derivation, solution method, memory analysis and programming framework in both single-phase and bi-phase AC railway power feeding systems.
- **Chapter 4** focuses on the selection of a linear-solving system, which can effect considerable reduction of execution time and memory requirement. This chapter

mainly emphasises the use of sparse matrix computation to achieve the distinctive computing performance.

- **Chapter 5** starts with an introduction of optimal power flow problems. It also describes the reactive power compensation in the special case of AC railways. Then the use of modern electric locomotives acting as mobile reactive power compensation is investigated. Finally, an area control system is introduced and explained.
- **Chapter 6** discusses an AC railway system simulator, which consists of train movement simulation, network capture, AC railway power flow and optimisation to perform full-hour operation of typical train services.
- **Chapter 7** presents design considerations for reactive power compensation in AC railways. SVC installation represents a conventional compensation method for comparisons. Then, design and cost estimation of the proposed method are illustrated.
- **Chapter 8** ends the thesis with conclusions from the work done for the completion of the PhD study and also gives suggestions towards future work.

CHAPTER 2

GENERAL REVIEW

2.1 Introduction to Mainline Railway Electrification

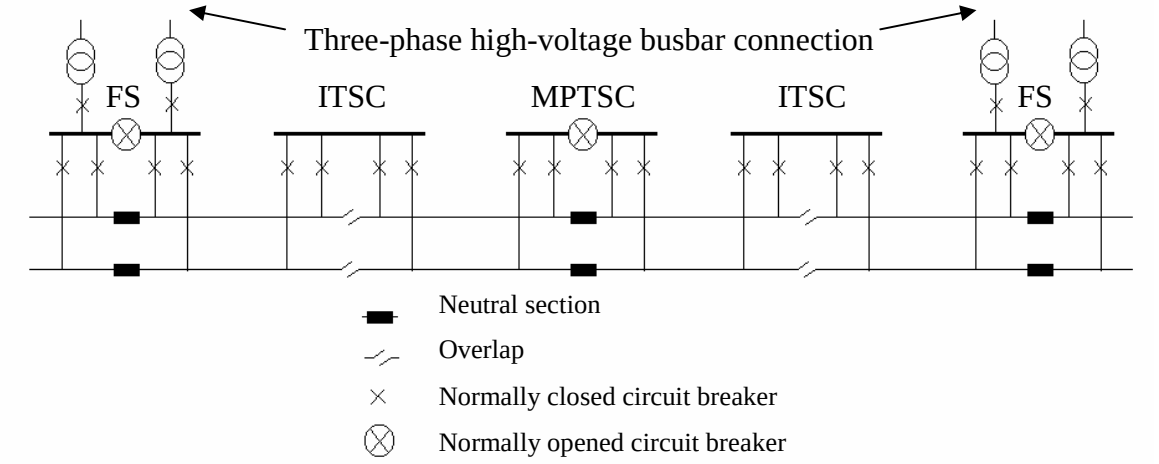
The knowledge of railway and steam engines has been around since the sixteenth century. Wagon roads for English coalmines using heavy planks were first designed and built in 1633. Mathew Murray of Leeds in England invented a steam locomotive that could run on timber rails in 1804 and this was probably the first railway engine [1]. Although railway and locomotive technologies were continually developed, the first electrified railway was introduced in the 1880s [2-4]. As a result of this revolution, the traction motor and the power supply system have become important parts of modern electrified railways.

In the past, DC power supplies (1.5 kV from the early 1900s and 3 kV from the 1930s) were mainly used because of ease of control. However, difficulties of DC motor commutation, limitations of feeding distance and expensive power supply equipment led to the restrictions on the use of DC mainline railways. Using a high-voltage AC power transmission system, long-distance feeders have become possible. Until recently, AC/DC converter-fed DC motors were used, with the converters fed from the AC supply through a traction transformer. The advantage of DC motors is simple torque-speed control. There exist different operating frequencies that are used in electrified railways in various parts of the world. Low-frequency high-voltage transmission networks, 15 kV at $16\frac{2}{3}$ Hz and 12 kV at 25 Hz, are used to feed AC commutator motors in some central European countries (Norway, Sweden, Switzerland, Austria and Germany) and in New York, USA, respectively [5,6]. Around the 1950s, electrified railways at the industrial frequency, 50 Hz, were established. The Valenciennes-Thionville line in France was the first 50 Hz railway electrification [7]. Thus far, a single-phase feeding system with 25

kV at 50 Hz has become the world standard of mainline railway electrification while a 50 kV feeding system has been used for railways with heavily loaded locomotives.

2.2 AC Railway Overhead Catenary Feeding System

An AC railway power supply system has several configuration features different to an industrial power system; notably, it is single phase. First, a typical AC railway feeder substation is directly connected to the three-phase high-voltage supply grid. The single-phase 25 kV is now the UK's standard overhead catenary feeding system [8]. Each feeder substation in the UK typically consists of two 132/25 kV (or nowadays at 275/25 kV or even 400/25 kV is the newest cases) power transformers. The high-voltage side is connected to the utility's three-phase busbar whereas its low-voltage side is connected to a single-phase busbar [9] as shown in Fig. 2.1.



Abbreviation notes:

FS	Feeder substation
MPTSC	Mid-point track sectioning cabin
ITSC	Intermediate track sectioning cabin

Fig. 2.1: Typical feeding diagram of a double-track 25 kV railway in UK

The feeding arrangement of the single-phase AC railway power supply requires neutral sections to separate two adjacent feeding networks at the feeder substation and the MPTSC under normal operation. The MPTSC is located approximately mid-way between feeder substations. Furthermore, there may be other sub-sectioning cabins on the trackside such as an ITSC. When

either a temporary fault or a permanent fault occurs on any feeding section, the corresponding sub-sectioning must isolate the fault instantly while other feeding sections continue to operate without any interruption. The ITSC is located between feeder substations and MPTSC. In addition, to provide some conventional equipment (e.g. SVC and/or power factor correction or power filter) there needs to be space at sub-sectioning cabins for installation.

There are some basic feeding configurations that are widely used for feeding electric energy to electric trains in mainline AC railways [7,8] as shown in the following description and Fig. 2.2.

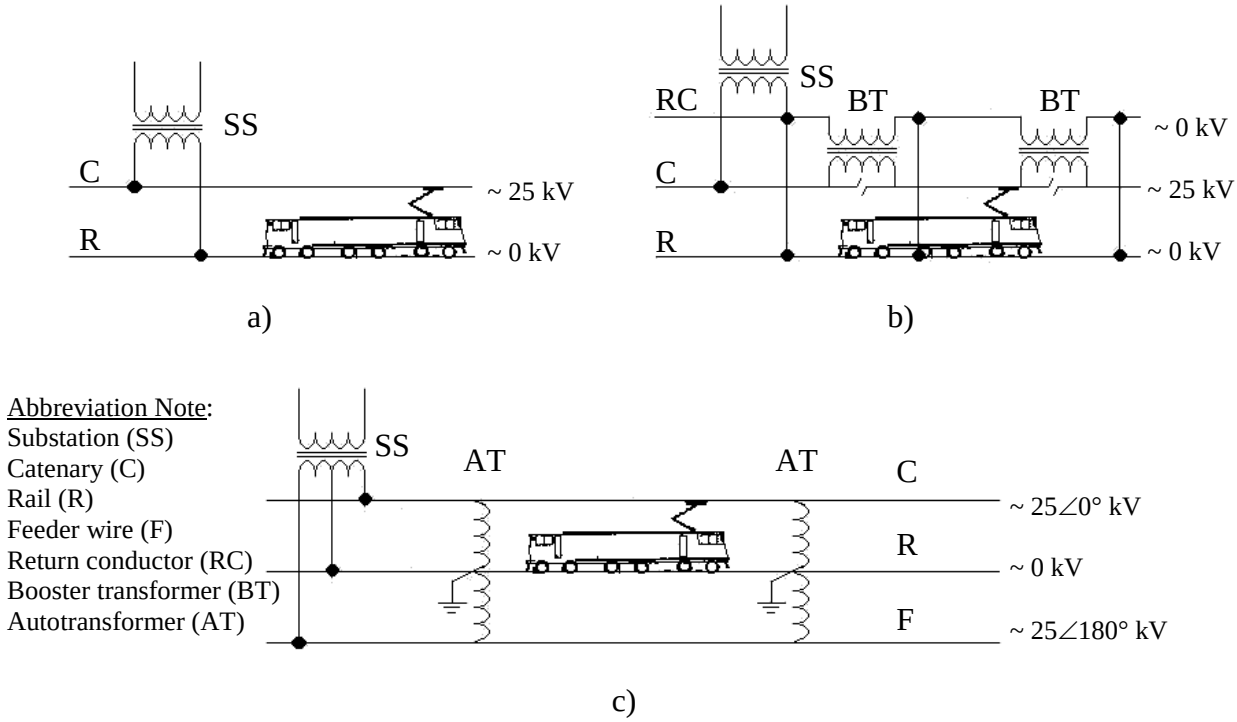


Fig. 2.2: Overhead catenary feeding systems for AC railways

1. Direct feeding configuration

Direct connection of the feeding transformer to the overhead catenary and the rails at each substation is quite simple and cheap. However, there are some disadvantages to this scheme (high impedance of feeders with large losses, high rail-to-earth voltage and the interference to neighbouring communication circuits). To reduce those effects, the addition of an extra conductor (Return Conductor) paralleled and tied to the rails at typically 5 or 6 km is

needed and this can reduce electromagnetic interference in parallel communication lines by 30% [7]. Fig. 2.2a shows the diagrams of the direct feeding configuration

2. Booster transformer feeding configuration

This method employs booster transformers (BT) added along the catenary at every 3-4 km. To force the return current to flow in the return conductor rather than in the rails to suppress the magneto-motive force resulting from the catenary current, the turn ratio needs to be unity [10]. Although this feeding reduces electromagnetic interference with about 0.025 screening factor (the ratio of the induced voltage in a disturbed circuit in a presence of compensation to the induced voltage obtained without compensation) at 50 Hz and 1 mile BT spacing, the leakage inductance of BTs with a return conductor increases the total feeding impedance by approximately 50% compared with the direct feeding. Thus, the distance of two adjacent feeder substations is reduced because of the voltage drop along the contact wire. Fig. 2.2b shows the configuration of the BT feeding system.

Furthermore, when a large-power train passes through a BT isolated overlap, severe arcing can occur. This arcing produces extra interference and damages the contact wire and the train pantograph.

3. Autotransformer feeding configuration

Adding 50 kV autotransformer (AT) at every 8-15 km intervals can increase substation distance up to 50-100 km. The AT has two equal-turn windings, whose middle tap is connected to the rails to provide earth potential for balancing a voltage between the contact wire and the return conductor [10-11]. The electromagnetic interference in an AT system is normally lower than that in the BT system. However, the size and MVA rating of the AT are much larger and more expensive than the BT. In addition, its protection equipment is more complicated and it needs more installation space. Fig. 2.2c shows the configuration of the AT feeding system.

2.3 Conventional Power Flow Solution Methods

A power flow problem is concerned with steady-state computation of power system performances, such as voltages, current or power flows through feeder portions, under normal operation of system loads and power generation. The main procedure is to formulate a set of equations, naturally non-linear, representing the entire system and then apply some solution technique to obtain its solution. Although there is a general form for power flow problems, in this thesis, only that with some modification for the special case of AC railway power feeding systems is reviewed.

Consider a typical bus of an AC power network as shown in Fig. 2.3. The connected generator at the bus can present the substation or a re-generative train, while the load represents a powering train.

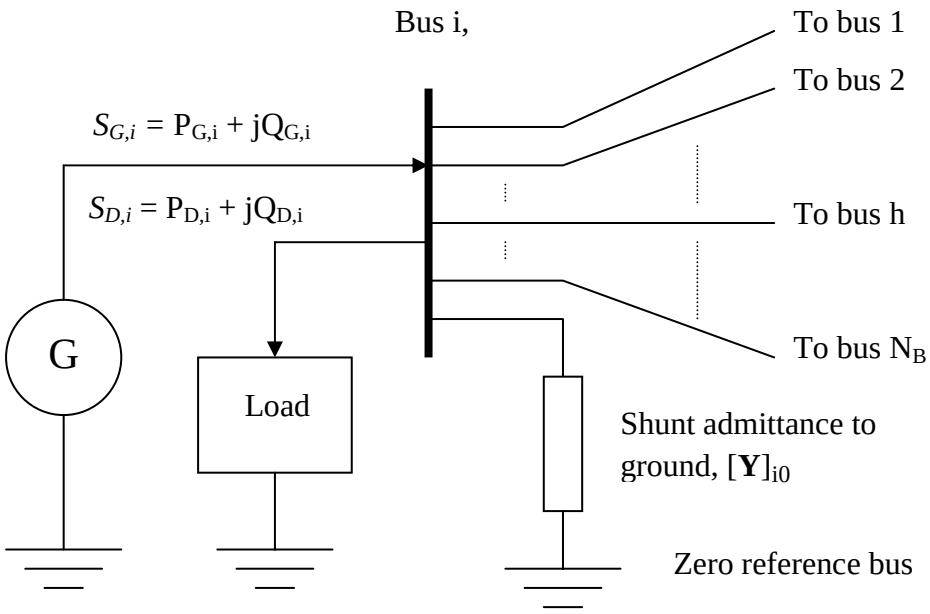


Fig. 2.3: Typical bus representation

From the figure, there is an N_B-bus power system and one of them is treated as a reference bus whose voltage magnitude and angle are both specified, normally at the substation. Conveniently, the bus labelled N_B will be selected as the reference bus. According to the Gauss-Seidel (GS) algorithm [12,13], bus voltages are updated repeatedly at each iteration. The

following is the expression to update bus voltages for the GS method, where the superscript k denotes the k^{th} iteration

$$V_i^{(k+1)} = \frac{1}{Y_{ii}} \left\{ \frac{P_i - jQ_i}{V_i^{(k)*}} - \sum_{m=1}^{i-1} Y_{im} V_m^{(k+1)} - \sum_{n=i+1}^{N_B} Y_{in} V_n^{(k)} \right\} \tag{2.1}$$

By applying the Taylor series expansion, the Newton-Raphson (NR) method approximates the real and reactive power flow equations by collecting the first two terms and neglecting other higher-order terms [14], but there is still some interaction between them. Briefly, equations 2.2 and 2.3 represent the real and reactive power mismatch equations after the approximation and equation 2.4 shows the compact form of the well-known NR power flow method.

$$\Delta P_i = P_{i,sch} - \sum_{\substack{k=1 \\ k \neq i}}^{N_B} |Y_{ki} V_k V_i| \cos(\theta_{ki} + \delta_k - \delta_i) = 0 \tag{2.2}$$

$$\Delta Q_i = Q_{i,sch} + \sum_{\substack{k=1 \\ k \neq i}}^{N_B} |Y_{ki} V_k V_i| \sin(\theta_{ki} + \delta_k - \delta_i) = 0 \tag{2.3}$$

$$\begin{bmatrix} \Delta\delta \\ \Delta|V| \end{bmatrix} = \begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix}^{-1} \begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} \tag{2.4}$$

The GS method proceeds by updating each unknown bus voltage separately, bus-by-bus, from the first to the last and thus converges slowly, and needs much more calculation time than the NR method, which converges quadratically [3]. Besides the GS and the NR methods, there are other modified methods such as Decoupled power flow or Fast Decoupled power flow methods.

In most AC transmission systems, the sub-matrices $[J_2]$ and $[J_3]$ are numerically less significant. So, the equation 2.4 can be decomposed to equations 2.5 and 2.6. This is the well-known Decoupled power flow method.

$$[\Delta\delta] = [J_1]^{-1} [\Delta P] \tag{2.5}$$

$$[\Delta|V|] = [J_4]^{-1} [\Delta Q] \tag{2.6}$$

These approximations require more iterations than the original does. However, with less memory requirement and simpler updating formulae for the Jacobian matrices, the overall execution time is shorter. Furthermore, when the ratio $\frac{R}{X}$ of the feeder lines is relatively small, $\frac{R}{X} << 1$, the resistance of the power lines can be neglected. Therefore, $[B_1]$ and $[B_2]$ of the following equations can adequately replace $[J_1]$ and $[J_4]$ of the equations 2.5 and 2.6. This leads to the Fast Decoupled power flow solution method consisting of two separate matrix equations as described in equations 2.7 – 2.8.

$$[\Delta\delta] = [B_1]^{-1}[\Delta P] \tag{2.7}$$

$$[\Delta|V|] = [B_2]^{-1}[\Delta Q] \tag{2.8}$$

2.4 Computer-based Modelling and Simulation in Railways

Planning and design of modern railway electrification demands a high quality computer-based simulator. There are many educational and commercial software packages developed for railway system simulation across the world, but each individual simulator has unique features for its own particular objective. Most of these are not only expensive but also complicated. A complete railway system simulation typically includes various features such as interlocking, signalling, power supply, train movement, etc. In many applications, many of these features are rarely, if ever used. In addition, modification and extension of the software sometimes becomes necessary for a particular study, but most software suppliers do not allow the end-user to do so. Design engineers wishing to improve the accuracy of the simulation need to develop an understanding of the programming and algorithms encapsulated in the software. This review therefore focuses on advanced modelling and simulation of a modern electric railway in order to explain the basis of an efficient railway system simulator, particularly for AC railway power

network systems. Before describing the proposed algorithm, some existing software for railway system simulation will be briefly reviewed.

- TTS/SIMON POWER LOG [15,16] has been developed and used by the Swedish National Rail Administration. This simulator is a combination of a Train Traffic Simulation (TSS) and a power calculation module called Power Log. However, it is primarily targeted on train traffic planning and operation, and only simple power supply assumptions are made such as constant train voltage throughout the operation.

- The rail transit Energy Management Model (EMM) [17] developed by Carnegie Mellon University, USA, consists of two major simulators: i) Train Performance Simulator (TPS) and ii) Electric Network Simulator (ENS). In this simulator, power network solutions are obtained based on the Gauss-Seidel power flow model. This model has widely been used in many North American Railway Systems such as WMATA and MARTA Metro-rails.

- The Multi-Train Simulator (MTS) [18,19] developed by University of Birmingham, UK, has full interfacing between the power network and the train movement simulators such that the train performances are updated according to their pantograph voltages, which are obtained by linearised nodal analysis, at least for DC. It has been used by several operators and manufacturers in the UK, Europe and the Far East.

- OSLO (Overhead System Loading), which can be added on to the VISION train movement simulator as the VISION/OSLO package [20], developed by British Rail Research, is an AC railway power network simulator. A lumped impedance model of the overhead catenary system and a load transfer technique are used to simplify the network equations. This simulator is used by Network Rail, for example for the West Coast Main Line and also in some other countries, and is currently available from AEA Technology Rail.

- The Computer Aided Design Tool [21] developed by a consortium of SIEMENS, ABB and the Universidade Tecnica de Lisboa, Portugal, (CAUTL). This software package

consists of four main modules that are: i) Database Management, ii) Train Movement Simulation, iii) Traffic Simulation and Load Flow, and iv) Harmonic Analysis. This software uses the well-known Newton-Raphson load flow as the main power network solver.

- SIMSPOG simulation tool [22], developed by Holland Railconsult, Netherlands, is a mathematical model of AC railway power supply systems based on the MATHCAD software package. This simulator can apply equally to the 1×25 kV (Simple feeding system) and the 2×25 kV (Autotransformer feeding system) AC power supplies. To obtain voltage solutions, a set of simple nodal equations are formed and then solved. It has been used to design Dutch AC railway lines, such as Betuweroute and Havenspoorlijn, and to analyse the Ligne du Nord of the Luxembourg Railways (CFL).

2.5 Power System Control and Reactive Power Compensation

To supply electric energy to loads in AC power transmission or distribution systems, the voltage waveform is regulated at the supply point to have nearly constant voltage magnitude, nearly constant operating frequency and no harmful harmonics. Also, power factor at the supply point is ideally expected to be unity. In fact, voltage drop, harmonic interference and poor power-factor operation are unavoidable due to non-linearity of loads and imperfection of power systems. Therefore, to minimise those effects, power filters or compensators have to be designed and installed. With the availability of high power semiconductor devices, power-electronic-controlled power compensators were invented and have been successfully used for this purpose [23,24]. Self-commutated devices such as GTOs or IGBTs have been developed with high-speed switching ability and have characteristics close to ideal switches. Therefore, more effective PWM techniques for controlling output magnitude and harmonic distortion are possible. By controlling shunt or series compensators with such switching devices, the power

flow through transmission lines can considerably be improved in both real and reactive powers [25,26].

2.5.1 Principles of AC Power Transmission and Power Compensation

To draw out the main features, the following simple power circuit is considered as in Fig. 2.4.

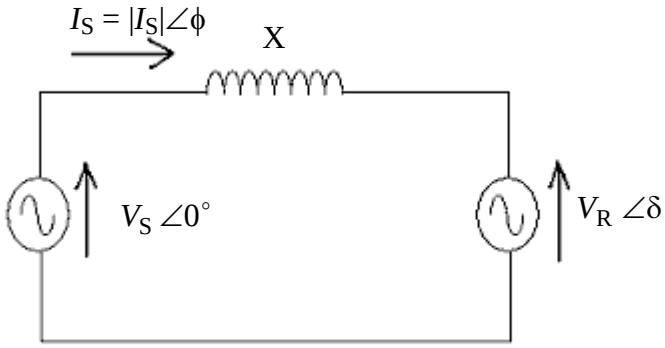


Fig. 2.4: A two-machine model for an AC transmission system

Neglecting line resistance, the power flow through the line can be expressed by the following equations [26].

$$P_S = \frac{|V_S||V_R|}{X} sin\,\delta \tag{2.9}$$

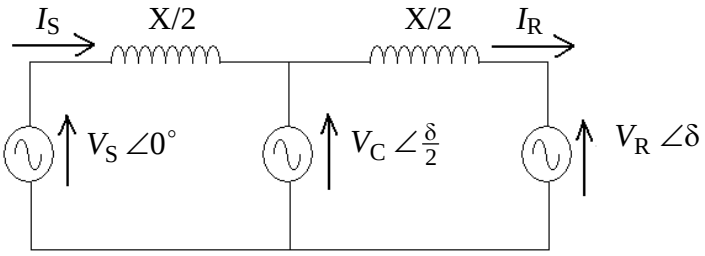
$$Q_S = \frac{|V_S|^2}{X} - \frac{|V_S||V_R|}{X} cos\,\delta \tag{2.10}$$

With specified X, $|V_S|$ and $|V_R|$, real and reactive power are both controlled by adjusting the variable δ. This conventional adjustment limits real power flow from $-\frac{|V_S||V_R|}{X}$ to $+\frac{|V_S||V_R|}{X}$.

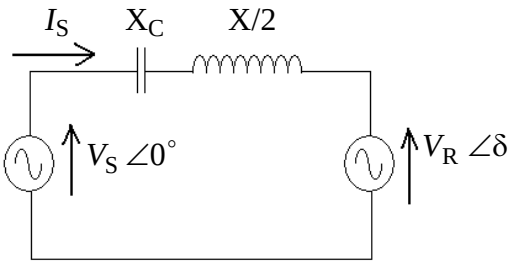
In the same manner, reactive power is also restricted from $\frac{|V_S|^2 - |V_S||V_R|}{X}$ *to* $\frac{|V_S|^2 + |V_S||V_R|}{X}$.

These limitations are the so-called steady-state power transmission limits [26]. To improve power flow capability, power system compensation with modern power-electronic control has been introduced and developed.

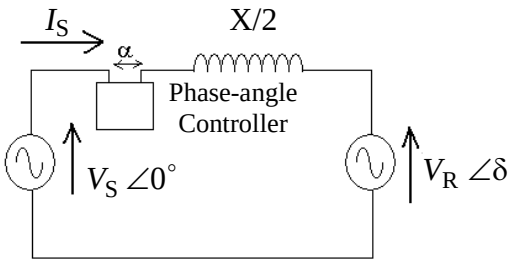
According to equations 2.9 and 2.10, power flows depend on the three basic parameters of voltage magnitude, line reactance and phase angle. Therefore, shunt, series and phase-angle compensation are used to control power flows as shown in Fig. 2.5. The power flow equations are thus modified to equations 2.11 to 2.13 for shunt, series and phase-angle compensation, respectively.



a) Shunt-connected compensation



b) Series compensation



c) Phase-angle compensation

Fig. 2.5: Three basic compensation methods

$$P_S + jQ_S = 2\frac{|V_S||V_C|}{X}\sin\delta + j\left(2\frac{|V_C|^2}{X} - 2\frac{|V_S||V_C|}{X}\cos\frac{\delta}{2}\right) \tag{2.11}$$

$$P_S + jQ_S = \frac{|V_S||V_R|}{(X - X_C)}\sin\delta + j\left(\frac{|V_C|^2}{(X - X_C)} - \frac{|V_S||V_C|}{(X - X_C)}\cos\frac{\delta}{2}\right) \tag{2.12}$$

$$P_S + jQ_S = \frac{|V_S||V_R|}{X}\sin(\delta - \alpha) + j\left(\frac{|V_S|^2}{X} - \frac{|V_S||V_R|}{X}\cos(\delta - \alpha)\right) \tag{2.13}$$

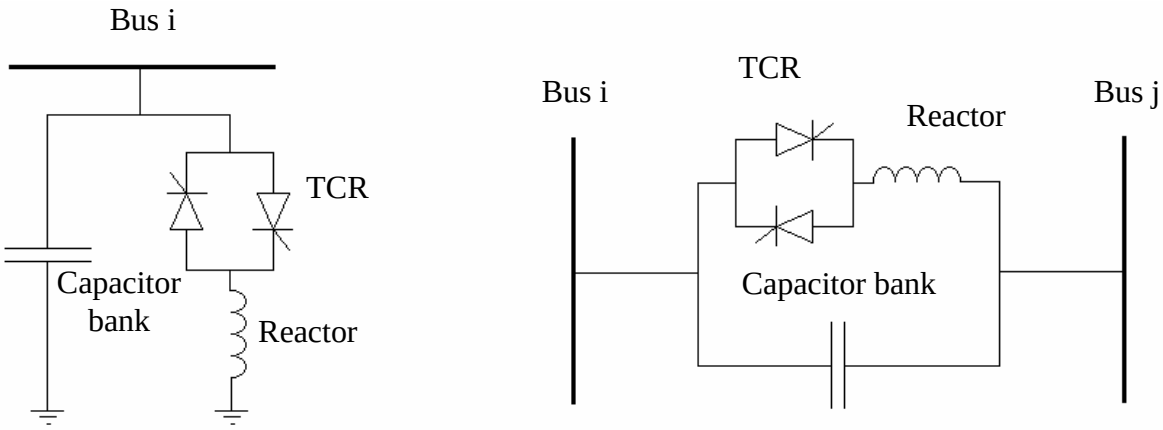
2.5.2 Applications of Power Electronics to Power Flow Control

Inductors and capacitors are passive elements. They can inject reactive power to help reactive power shortage situation. Combining them with power electronic switches enables reactive power-flow control that can minimise power losses, voltage drops and even harmonic interference in the system. The following describes the use of power electronics for power system compensation [25,26].

1. Static Var Compensator (SVC): It acts like a variable shunt reactor, which either injects or absorbs reactive power in order to regulate voltage level at the connected point [27]. The SVC circuit is shown in Fig. 2.6a.
2. Thyristor-controlled Series Compensator (TCSC): The TCSC is based on a non-linear series reactance [28]. Fig. 2.6b presents its simplified circuit diagram.
3. Static Phase Shifter (SPS): This equipment adjusts the phase angle of the receiving-end voltage with little delay [29]. It employs a lossless transformer with taps. To perform phase-angle regulation, the phase shifter consumes reactive power from the network. Fig. 2.7a shows its diagram.
4. Static Synchronous Compensator (STATCOM): In principle, it is similar to the SVC to perform the voltage regulation but it operates with an advanced energy storage element [30] as shown in Fig. 2.7b.
5. Dynamic Voltage Restorer (DVR): In steady-state condition, the DVR function is similar to the SPS but more gives versatile compensation because of an energy storage device [31]. The DVR is shown in Fig. 2.7c.
6. HVDC: The main function is to transmit constant DC power from the rectifier to the inverter station with high controllability [32-33]. Fig. 2.8a describes the HVDC circuit.

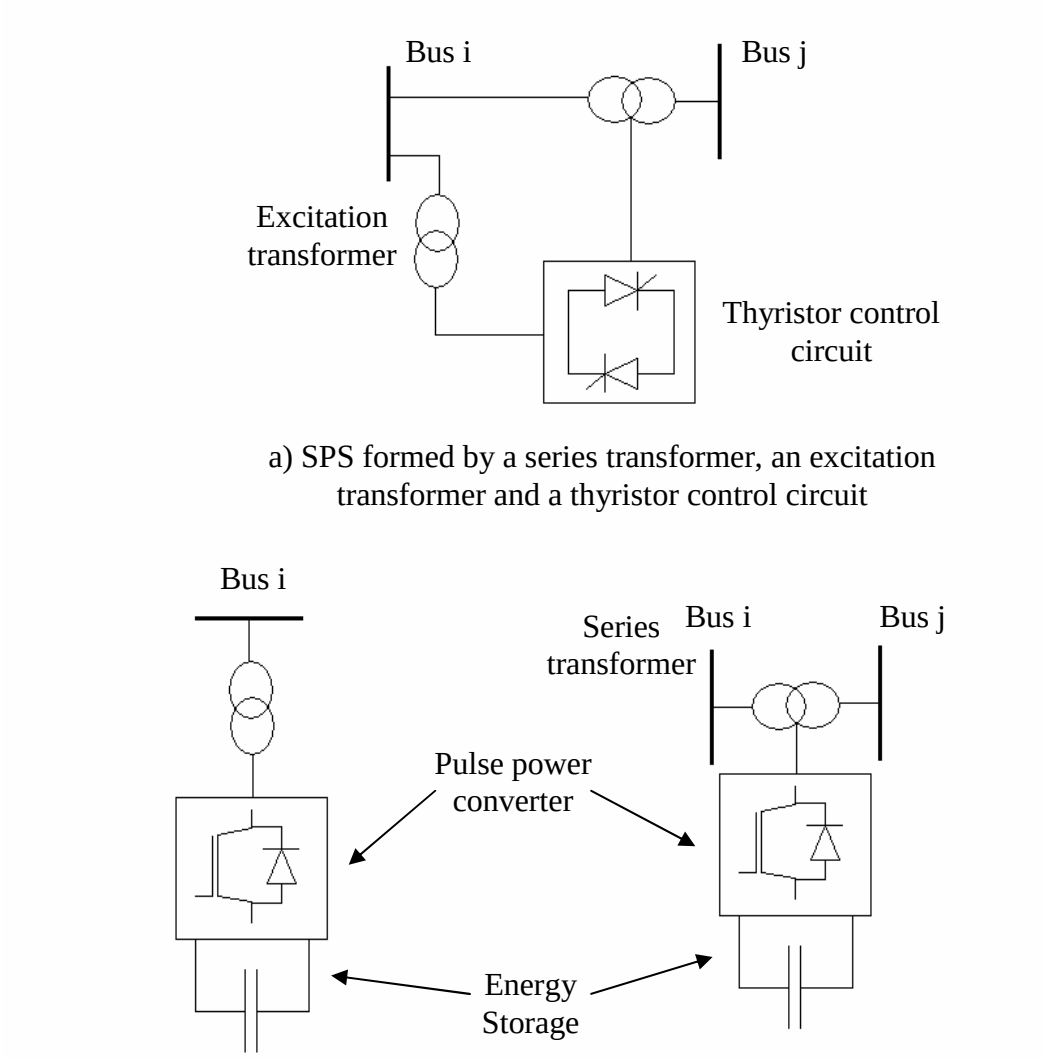
7. Unified Power Flow Controller (UPFC): The UPFC may be seem to consist of one STATCOM and one DVR sharing the DC link capacitor and their control system. It allows control of real power, reactive power and voltage magnitude at the UPFC terminals [34,35]. Fig. 2.8b shows the UPFC.

Power flow through the static power converter is controlled via the gating of switches and the DC bus voltage. The output power can be varied by changing the DC link voltage while the current being consumed constant and regulated. So the basic requirement of power converters for reactive power compensators is the control of either the voltage or current generated by the converter. Although there are two types of PWM power converters, the voltage source converter is widely used in high power applications including AC railways. Alternatively, the similar concept of reactive power control by power electronic equipment on-board a train can be further developed as a novel mobile reactive power source to compensate excessive reactive power required by the whole system. This leads to the reduction in size or elimination of the bulky and expensive SVC installed, that has become a possible solution for upgrading practical railway systems.



a) SVC circuit b) TCSC circuit

Fig. 2.6: SVC and TCSC circuits



a) SPS formed by a series transformer, an excitation transformer and a thyristor control circuit

b) STATCOM formed by SVC connected to the network via a shunt transformer

c) DVR formed by SVC connected to the network via a series transformer

Fig. 2.7: SPS, STATCOM and DVC circuits

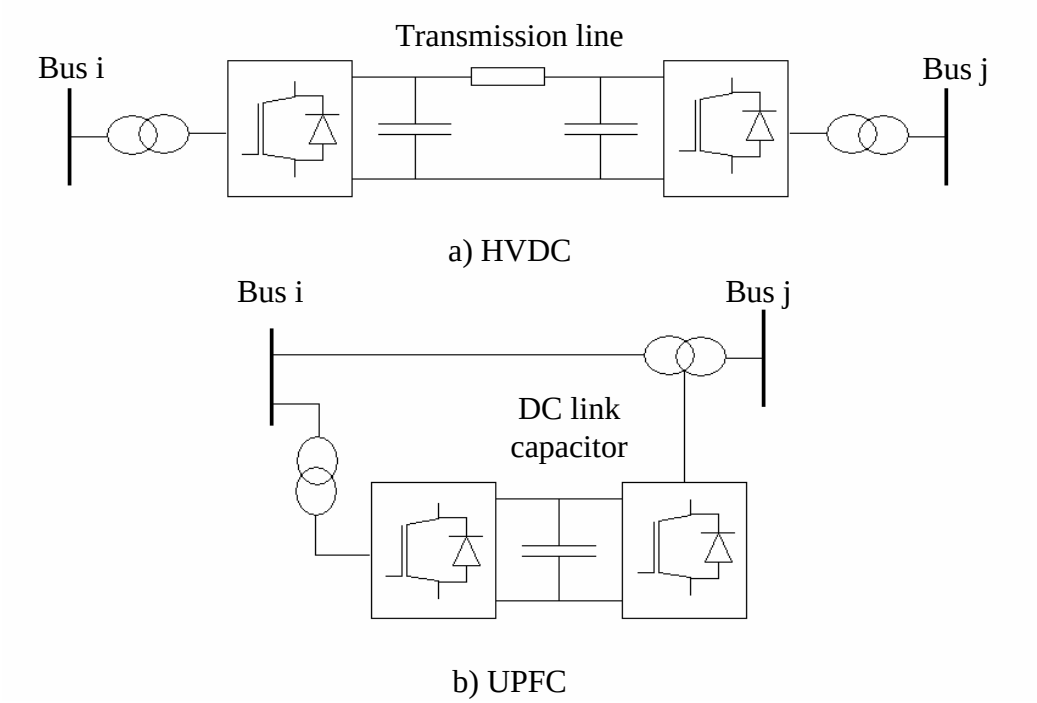


Fig. 2.8: HVDC and UPFC circuits

2.6 Summary

This chapter presents literature surveys for the development of AC railway power feeding systems, conventional power flow solution methods, computer-based modelling and simulation in railways, and power system control and reactive power compensation. There are three main feeding arrangements for mainline AC railways. Each has its individual loading characteristics. In the past, reactive power compensators, such as the SVC, have been used. Although these can compensate the system reactive power shortage, they are costly and space hungry.

With the successful development of high-performance computers, railway system modelling and simulation can be an alternative approach to plan, analyse, and design a modern AC railway system with a new upgrading method to be able to fully replace the older one. Methods of calculating power flows and software development were briefly reviewed. Many of them are not particularly efficient or effective. Demand for railway transport increases the system complication every year, while the existing railway simulators have some limitations.

This leads to a need for improved railway power supply calculations. In this thesis, the power network simulation is the principal topic.

PART I

AC Railway Power Flow Calculation
(Theory and Application)

CHAPTER 3

AC RAILWAY POWER FLOW ANALYSIS

3.1 Introduction

The main function of AC railway power supply systems is to deliver electric energy to the electric locomotives that are connected to the system, and to do so, effectively and economically. Calculations related to AC railway power supply systems need basic tools such as power flow algorithms for obtaining voltage, current or power flows through each feeder section. As long as the AC railway power systems can be described by the conventional power supply analysis [12], classical power flow methods are applicable. The Newton-Raphson (NR) and the Gauss-Seidel (GS) methods are two well-known iterative techniques for solving power flow problems. With quadratic convergence [14], the NR method has been successfully developed and broadly accepted as the most powerful algorithm for several decades. Although these methods were originally developed for solving industrial power systems, they can be adapted to railway power distribution systems. However, they are not necessarily the best approach for the railway systems. Due to moving loads and some special feeding configurations, these methods are inefficient in some circumstances. Therefore, modelling developed for the special case of AC railway power systems is needed to handle the system complexity and can then be embedded into the railway system simulator to calculate power supply conditions in real-time or at least at small discrete-time updates. So, a new power-flow algorithm that gives a reduction of execution time and memory use is required. In this chapter, analysis, modelling and simulation of the old fashion AC railway power flow analysis and the new developed power flow method are explained in more detail.

3.2 Modelling of AC Railway Power Feeding Systems

The overhead catenary feeding system is complicated as described in the literature survey. To formulate power flow equations, transmission lines and other network components requires sufficiently accurate modelling. In this thesis, the AC railway power feeding systems can be divided into two categories according to the feeding arrangement: i) Single-phase AC railway power feeding systems and ii) Bi-phase AC railway feeding systems (Autotransformer scheme). The modelling approach for each category is described as follows.

3.2.1 Single-phase AC railway power feeding system

Although the properties of the AC railway power feeding systems are intrinsically non-linear, there are some simplifications of the power network modelling. Impedances of the overhead catenary wire, rails, return conductor and other equipment such as BTs are all added in series. Thus, a single-phase AC equivalent circuit consisting of an average lumped conductor on one side (Line or positive conductor) and a zero-impedance conductor on the other side (Neutral or negative conductor) is formed as shown in Fig. 3.1. This approach is adequate to calculate voltages across trains and phase-to-ground substation and MPTSC voltages. However, rail potential, rail-to-earth leakage current, EMI effect, etc cannot be determined using this simplified model.

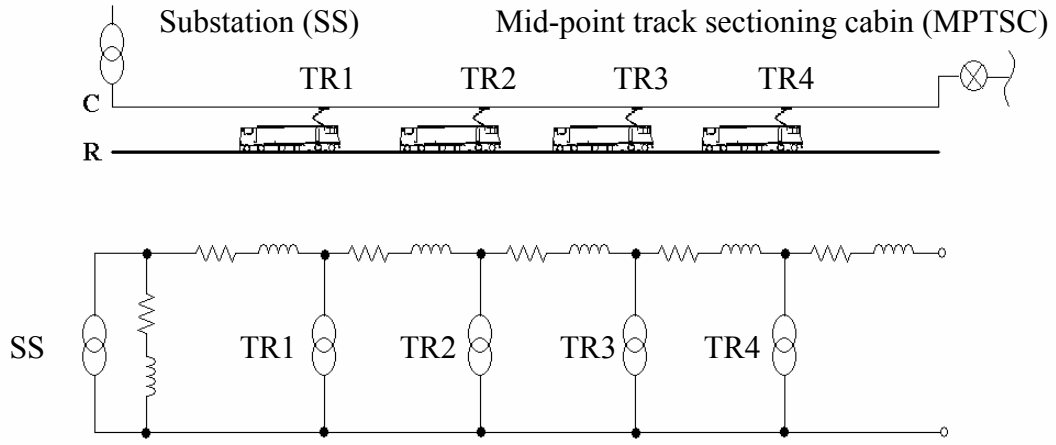
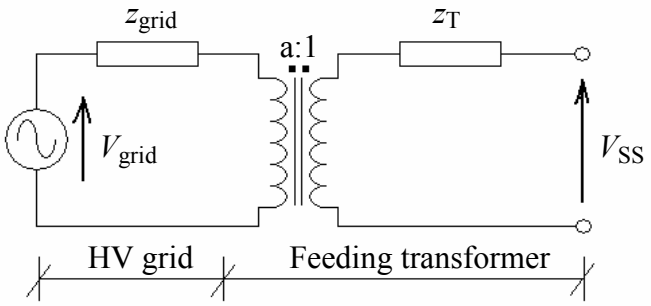


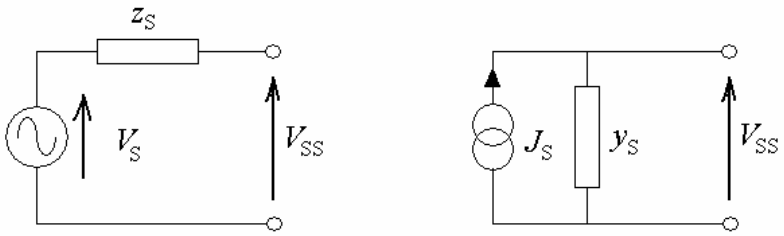
Fig. 3.1: Single-phase AC railway power feeding model

To present the lumped impedance model for this system, the simplest way is the addition of any transformer's or other equipment's impedances on the trackside all together with the overhead catenary, rails and return-conductor impedances (if used). Therefore, the average lumped impedance per unit length can be defined by the following equation [36].

$$z_1 = \frac{z_{OH} \cdot FD + \sum_{i=1}^{N_A} z_{a,i}}{FD}, FD = \text{Feeding distance} \tag{3.1}$$



a) Simplified power substation model



b) Thevenin equivalent circuit c) Norton equivalent circuit

Fig. 3.2: Power substation model

A typical AC railway power feeding system receives the electric energy at the substation. For some technical reasons, i.e. feeding reliability, protection, rotation of phases, etc, any feeding section is isolated from the others with only one power substation. Conventionally, the power substation is modelled by a combination of an infinite busbar (ideal voltage source) in series with equivalent high-voltage grid impedance connected to a feeder transformer with on-load tap changer as shown in Fig. 3.2a. This circuit is then simplified to either Thevenin or Norton equivalent circuits as shown in Figs 3.2b and Fig. 3.2c, respectively. The parameters of these circuits can be easily computed by the following relations.

$$V_S = \frac{V_{grid}}{a}, \qquad z_S = z_T + \frac{z_{grid}}{a^2}, \; y_S = \frac{1}{z_S} \tag{3.2}$$

The other key power system component is the electric train. It requires a simple model to reduce problem complexity and overall execution time; directly applying differential equations to obtain a model like that used in traction drive analysis is not appropriate for steady-state power flow calculations [13].

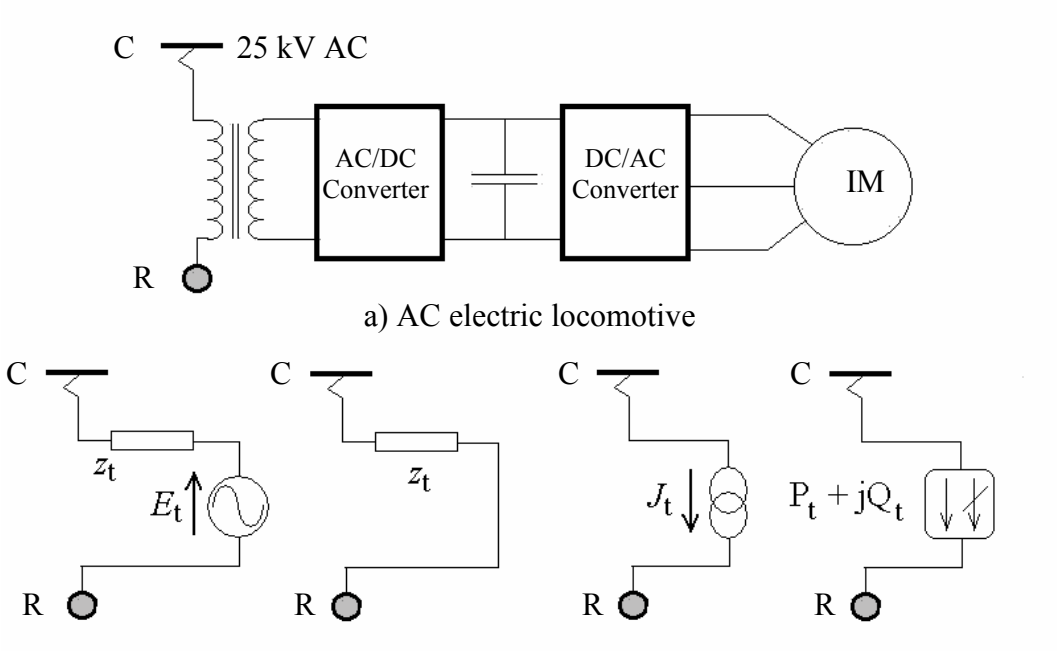


Fig. 3.3: Four locomotive models for power flow calculation

In some particular conditions, for example with a current control loop embedded in the on-board power converter [37], the current load model is reasonable. However, during simulation, the pantograph voltage is an unknown and needs to be solved for by a power flow program. The phase angle of the locomotive current must be specified with respect to the phase angle of the substation voltage. It cannot be specified unless the power flow solution has been successfully obtained. In practice, only four electrical quantities (Pantograph voltage-magnitude, input current-magnitude, power and power factor) can be measured on-board a train. Any model that

uses a well chosen sub-set of these quantities is satisfaction. The following presents four locomotive models that are used by at least some multi-train simulators [2].

- **Linear model**

The voltage source (E_t) and the series impedance (z_t) representing this model can be determined as a function of speed, pantograph voltage and tractive effort over the whole range of operation. The direct calculation cannot be done because of the phase angle of the internal voltage source, which must be referenced to the substation. Thus, a look-up table is used to define the model parameters at any instance according to train speed, voltage and tractive effort. When the locomotive model is needed, the relevant values can be loaded and used in the power flow program. The inevitable disadvantage of this model is that a substantial database for each locomotive must be prepared for the simulation program. The linear model is shown in Fig. 3.3b.

- **Impedance or admittance model**

The effective model is the one that uses only the four measured quantities. The impedance magnitude is calculated by the ratio between the voltage and current magnitudes whereas the phase angle is simply determined from the power factor interpretation. The impedance model is given in Fig. 3.3c.

- **Constant current model**

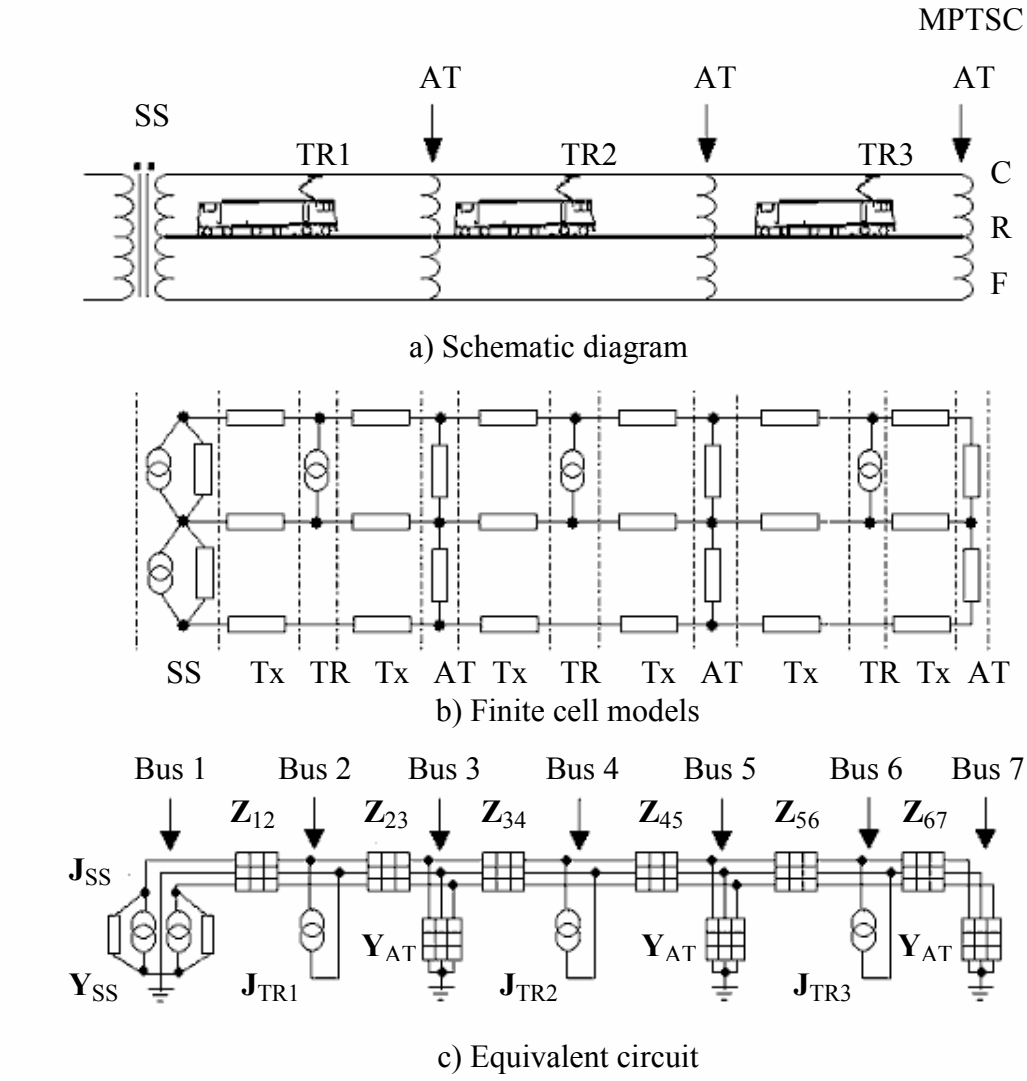
Although, as previously described, the current model is not appropriate due to the unknown phase angle, a numerical database stored for this model with respect to train speed, pantograph voltage and tractive effort like the linear model is another effective way. The look-up table is required to obtain its parameters at any simulated time step. Fig. 3.3d presents the current model.

- **Constant power model**

This model is widely used in three-phase power flow problems. Also, it is mainly used in this research because powers and power factor are the two quantities that can be measured by the on-board traction controller. Furthermore, applying tractive effort vs speed and train-running information with the Newton's second law of motion can alternatively calculate the model parameters [38,39]. Fig. 3.3e shows the power model.

Both direct and BT feeding configurations are in this category.

3.2.2 Bi-phase AC railway power feeding system



a) Schematic diagram

b) Finite cell models

c) Equivalent circuit

Fig. 3.4: Model representation of the AT feeding system

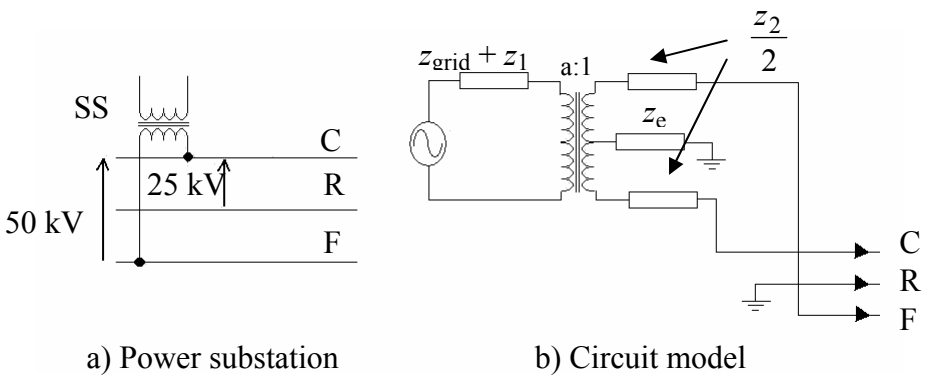


Fig. 3.5: Power substation model for the AT system

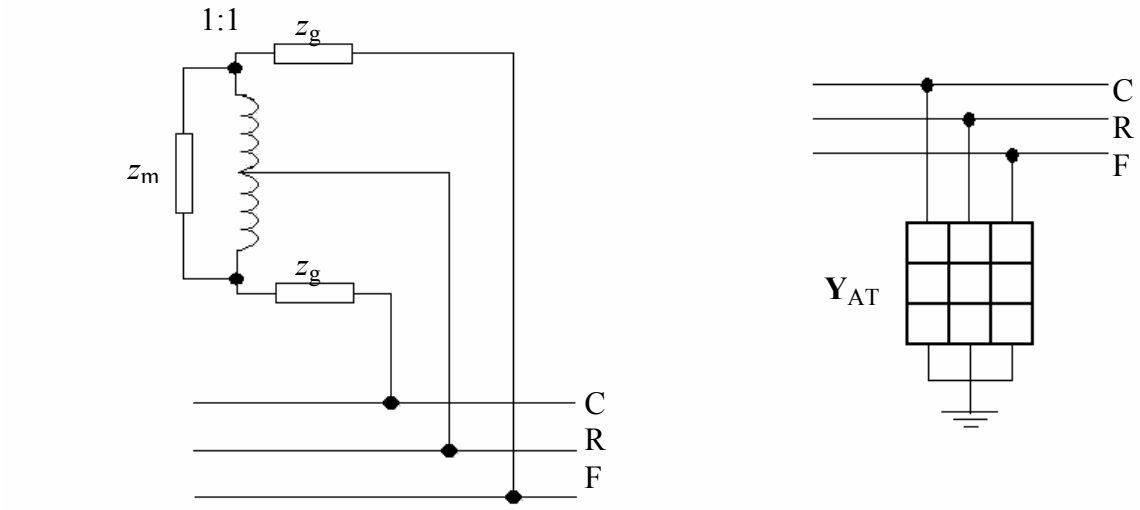


Fig. 3.6: Autotransformer model

For the AT system, the feeding configuration is more complicated. There exist many different approaches to describe such a power network [2,3,7,10,40,41]. Based on a linear AC power circuit, it is adequate to use models where the line is divided into relatively short sections (say 1 km) and the properties within each section are represented by lumped parameters. Each short section is thus modelled by a simple combination of impedances as shown in Fig. 3.4, the parallel impedance i.e. the power supply insulation usually being so high as not to need including. Parallel resistance will need to be included if the model is extended to the ground underneath the track as the rails are generally poorly insulated from the ground. This representative is sometimes known as the "finite cell" approach. Each individual element has its own model. Around 1980 – 1990, models for all the components of an AT power network were

proposed [2,3,10]. Figs 3.5 and 3.6 give the model of the power substation and the AT suggested by [10] with the multi-conductor approach. In addition, the equations 3.3 – 3.5 describe the substation and AT model parameters.

$$\mathbf{J}_{\mathrm{SS}} = \frac{V_0'}{z_{\mathrm{A}}}\begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}\begin{matrix} \mathrm{C} \\ \mathrm{R} \\ \mathrm{F} \end{matrix} \tag{3.3}$$

$$\mathbf{Y}_{\mathrm{SS}} = \begin{matrix} \\ \mathrm{C} \\ \mathrm{R} \\ \mathrm{F} \end{matrix}\begin{bmatrix} \mathrm{C} & \mathrm{R} & \mathrm{F} \\ \frac{1}{z_{\mathrm{A}}}+\frac{1}{z_{\mathrm{B}}} & \frac{2}{z_{\mathrm{B}}} & \frac{1}{z_{\mathrm{A}}}-\frac{1}{z_{\mathrm{B}}} \\ \frac{2}{z_{\mathrm{B}}} & -\frac{4}{z_{\mathrm{B}}} & \frac{2}{z_{\mathrm{B}}} \\ \frac{1}{z_{\mathrm{A}}}-\frac{1}{z_{\mathrm{B}}} & \frac{2}{z_{\mathrm{B}}} & \frac{1}{z_{\mathrm{A}}}+\frac{1}{z_{\mathrm{B}}} \end{bmatrix} \tag{3.4}$$

$$\mathbf{Y}_{\mathrm{AT}} = \begin{matrix} \\ \mathrm{C} \\ \mathrm{R} \\ \mathrm{F} \end{matrix}\begin{bmatrix} \mathrm{C} & \mathrm{R} & \mathrm{F} \\ \frac{1}{2z_{\mathrm{g}}}-\frac{1}{z_{\mathrm{m}}+2z_g} & -\frac{1}{z_g} & \frac{1}{2z_{\mathrm{g}}}+\frac{1}{z_{\mathrm{m}}+2z_g} \\ -\frac{1}{z_g} & \frac{2}{z_g} & -\frac{1}{z_g} \\ \frac{1}{2z_{\mathrm{g}}}-\frac{1}{z_{\mathrm{m}}+2z_g} & -\frac{1}{z_g} & \frac{1}{2z_{\mathrm{g}}}+\frac{1}{z_{\mathrm{m}}+2z_g} \end{bmatrix} \tag{3.5}$$

The impedance matrix representing the CRF conductor set connected between two adjacent buses (i and j) is described by equation 3.6.

$$\mathbf{Z}_{\mathrm{Tx,ij}} = \begin{matrix} \\ \mathrm{C} \\ \mathrm{R} \\ \mathrm{F} \end{matrix}\begin{bmatrix} \mathrm{C} & \mathrm{R} & \mathrm{F} \\ z_{CC} & z_{CR} & z_{CF} \\ z_{RC} & z_{RR} & z_{RF} \\ z_{FC} & z_{FR} & z_{FF} \end{bmatrix} \tag{3.6}$$

With these finite-cell models, a bus admittance matrix, [**Y**], characterising the whole feeding system is formed.

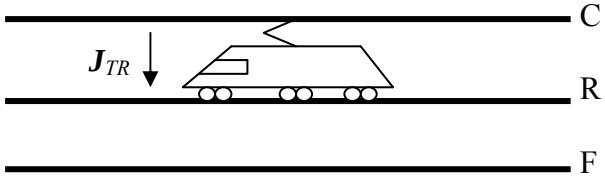


Fig. 3.7: Train model

For the electric train as shown in Fig. 3.7, it can be represented by various types of models such as current, admittance etc. If the admittance model is used, the train will be considered as an admittance to form the bus admittance matrix. When the current model is chosen, it will appear as a part of the bus current matrix [**J**]. The admittance model and the current model of a train can be expressed by equations 3.7 and 3.8. For more details, Appendix A presents mathematical derivations for all the models reviewed in this section.

$$\mathbf{Y}_{\mathrm{TR}} = \frac{1}{Y_{TR}} \begin{array}{c} \begin{matrix} \mathrm{C} & \mathrm{R} & \mathrm{F} \end{matrix} \\ \begin{bmatrix} 1 & -1 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{array} \begin{matrix} \mathrm{C} \\ \mathrm{R} \\ \mathrm{F} \end{matrix} \tag{3.7}$$

$$\mathbf{J}_{\mathrm{TR}} = J_{TR} \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix} \begin{matrix} \mathrm{C} \\ \mathrm{R} \\ \mathrm{F} \end{matrix} \tag{3.8}$$

After modelling each individual system component, the next step is to assemble all such models to form the system equation. This part is widely known as the power flow calculation. Using the phrase "power flow calculation" normally indicates that the system under consideration is non-linear. Over the long history of solving the AC railway power flow problems over half a century, an exact method of solution has not been found. Only numerical solutions have been developed and accepted as efficient algorithms for this purpose. In this chapter, the power flow calculation is investigated and then developed to improve the computing efficacy, i.e. reducing the execution time and memory requirement. For comparison, the well-known NR power flow method is developed as the standard method and explained in the next section.

3.3 Newton-Raphson AC Railway Power Flow Method

There are many kinds of the NR power flow method depending upon which circuit analysis technique is applied, Nodal or Mesh analyses [42]. This leads to the use of either bus admittance or bus impedance matrices. In this thesis, only power flow methods based on the

bus admittance matrix is determined. Before describing any power flow algorithm, the bus admittance matrix formulation is vital and needs to be reviewed.

3.3.1 Formulation of the bus admittance matrix

The bus admittance matrix, $[\mathbf{Y}]$, depends on the interconnection between bus pairs through a feeder portion. This is the essential characteristic of the power feeding system and it is usually independent from the load unless the train is described by the admittance model. In this thesis, the trains will be modelled by the power model only. This endows a great advantage to the power flow solution and will be discussed later. In addition, many other network components are also modelled by the admittance. Each of them can be either a series admittance connected between two buses or a shunt admittance to ground. Therefore, the bus admittance matrix is built up by the collection of all series and shunt components connected to the system.

Assume that $\mathbf{Y}_{ij}$ is an admittance of a feeder portion connected between bus i and j while $\mathbf{Y}_{i0}$ is an admittance to ground at bus i, e.g. $\mathbf{Y}_{AT}$ for an AT bus or $\mathbf{Y}_{SS}$ for the substation. The size of [**Y**] is equal to the total number, N_B, of buses in the system. The i^{th} row, j^{th} column of [**Y**] is called an off-diagonal element, $[\mathbf{Y}_{ij}]$ where $i \neq j$, and it represents the admittance value linking between bus i and j. The diagonal element, $[\mathbf{Y}_{ii}]$, is the collection of all the admittances connected to bus i, including the shunt admittances to ground. Each element of [**Y**] can be calculated by the following formula.

$$[\mathbf{Y}_{ii}] = \sum_{h=1}^{N_B} \mathbf{Y}_{ih} \quad \text{for diagonal elements} \tag{3.9}$$

$$[\mathbf{Y}_{ij}] = -\mathbf{Y}_{ij} \quad \text{for off-diagonal elements} \tag{3.10}$$

3.3.2 **Newton-Raphson power flow method for single-phase AC railway power feeding systems**

In a balanced three-phase industrial power system, the current and voltage magnitude of each phase component is equal at any part of the system but the phase difference between each phase pair is 120 electrical degrees. So, this leads to the use of a per-phase equivalent circuit for power flow calculations. As reviewed in Chapter 2, the power mismatch equations calculated for bus i are shown in equation 2.2 – 2.3. Thus, $N_B - 1$ independently non-linear equations of $N_B - 1$ variables are collected as equation 2.4. The bus voltage is updated by equation 3.11.

$$\mathbf{V}^{(k+1)} = \mathbf{V}^{(k)} + \mathrm{diag}\left(\Delta|V|\right)\cdot e^{j\Delta\delta} \tag{3.11}$$

Elements of the Jacobian sub-matrices, $[J_1]$ – $[J_4]$, in equation 2.4 can be computed by the following expressions [12,13].

Sub-matrix $[J_1]$:

$$J_1^{(k,i)} = -\left|V_k V_i Y_{ki}\right| sin\left(\theta_{ki} + \delta_i - \delta_k\right) \tag{3.12}$$

$$J_1^{(k,k)} = \sum_{\substack{h=1 \\ h\neq k}}^{N_B} \left|V_k V_h Y_{kh}\right| sin\left(\theta_{kh} + \delta_h - \delta_k\right) \tag{3.13}$$

Sub-matrix $[J_2]$:

$$J_2^{(k,i)} = -\left|V_k Y_{ki}\right| cos\left(\theta_{ki} + \delta_i - \delta_k\right) \tag{3.14}$$

$$J_2^{(k,k)} = 2\left|V_k Y_{kk}\right| cos\,\theta_{kk} + \sum_{\substack{h=1 \\ h\neq k}}^{N_B} \left|V_k Y_{kh}\right| cos\left(\theta_{kh} + \delta_h - \delta_k\right) \tag{3.15}$$

Sub-matrix $[J_3]$:

$$J_3^{(k,i)} = -\left|V_k V_i Y_{ki}\right| cos\left(\theta_{ki} + \delta_i - \delta_k\right) \tag{3.16}$$

$$J_3^{(k,k)} = \sum_{\substack{h=1 \\ h\neq k}}^{N_B} \left|V_k V_h Y_{kh}\right| cos\left(\theta_{kh} + \delta_h - \delta_k\right) \tag{3.17}$$

Sub-matrix $[J_4]$:

$$J_4^{(k,i)} = -|V_k Y_{ki}| \sin(\theta_{ki} + \delta_i - \delta_k) \tag{3.18}$$

$$J_4^{(k,k)} = -2|V_k Y_{kk}| \sin\theta_{kk} - \sum_{\substack{h=1 \\ h\neq k}}^{N_B} |V_k Y_{kh}| \sin(\theta_{kh} + \delta_h - \delta_k) \tag{3.19}$$

Furthermore, the sub-matrices $[J_2]$ and $[J_3]$ are numerically less significant. So, the equation 2.4 can be approximated and decomposed to equations 3.20 and 3.21. This is the well-known Decoupled power flow method.

$$[\Delta\delta] = [J_1]^{-1}[\Delta P] \tag{3.20}$$

$$[\Delta|V|] = [J_4]^{-1}[\Delta Q] \tag{3.21}$$

When the ratio $\frac{R}{X}$ of the feeder lines is relatively small, the resistance of the feeder portion can be neglected. Thus, $[B_1]$ and $[B_2]$ of the following equations can replace $[J_1]$ and $[J_4]$ of the equations 3.20 and 3.21. This results in reducing the execution time.

$$[\Delta\delta] = [B_1]^{-1}[\Delta P] \tag{3.22}$$

$$[\Delta|V|] = [B_2]^{-1}[\Delta Q] \tag{3.23}$$

Where

Sub-matrix $[B_1]$:

$$B_1^{(k,i)} = -|V_k V_i| B_{ki} \tag{3.24}$$

$$B_1^{(k,k)} = -|V_k|^2 B_{kk} \tag{3.25}$$

Sub-matrix $[B_2]$:

$$B_2^{(k,i)} = -|V_k| B_{ki} \tag{3.26}$$

$$B_2^{(k,k)} = -|V_k| B_{kk} \tag{3.27}$$

It notes that $[B_1]$ and $[B_2]$ are two Jacobian sub-matrices of the Fast Decoupled power flow method.

3.3.3 **Newton-Raphson power flow method for bi-phase AC railway power feeding systems**

Although the AT system consists of three main conductors, it is a bi-phase power system due to the rails acting like a neutral conductor. In some railway systems where rail potentials are determined, the rail impedance and mutual coupling between the rails and other conductors cannot be ignored. In many previous works [2,3,10,41], the multi-conductor approach has been used to model the AT power system as described in Section 3.2.2. However, the normal NR power flow methods for per-phase or three-phase power systems are not applicable due to the multiple conductors and the need for rail potential calculation. In this thesis, the NR power flow method is developed as the benchmark power flow calculation for the bi-phase AC railway power feeding system. It is described as follows.

Each bus of the AT system is defined as a collection of three separate conductor nodes. Current or voltage at each bus is a complex vector consisting of three components (C, R, F) as given in equation 3.28 – 3.29. Also, bus power is expressed in the similar manner.

$$\mathbf{V}_{\mathrm{i}} = \begin{bmatrix} V_i^{(C)} \\ V_i^{(R)} \\ V_i^{(F)} \end{bmatrix} \quad ; \text{for bus i} \tag{3.28}$$

$$\mathbf{J}_{\mathrm{i}} = \begin{bmatrix} J_i^{(C)} \\ J_i^{(R)} \\ J_i^{(F)} \end{bmatrix} \quad ; \text{for bus i} \tag{3.29}$$

Therefore, with applying KCL, the current-balance equation of bus i can be obtained by equation 3.30.

$$\sum_{\mathrm{h=1}}^{\mathrm{N_B}} \mathbf{Y}_{\mathrm{ih}} \mathbf{V}_{\mathrm{h}} = \mathbf{J}_{\mathrm{i}} \tag{3.30}$$

The current $\mathbf{J}_i$ depends on the current drawn by a train connected to bus i and also the current injected by the substation if i = SS, as described in equation 3.31.

$$\mathbf{J}_i = \mathbf{J}_{SS,i} - \mathbf{J}_{TR,i} \tag{3.31}$$

Assume that the power model is used to describe all trains in the system. Thus, the current-mismatch equation of the bus i can be expressed by the following relation.

$$\mathbf{F}_i = \sum_{h=1}^{N_B} \mathbf{Y}_{ih}\mathbf{V}_h - \mathbf{J}_{SS,i} + \mathbf{J}_{TR,i} = \mathbf{G}_i + j\mathbf{H}_i \tag{3.32}$$

$\mathbf{F}_i$ also consists of three components and can be described as follows.

$$F_i^{(\gamma)} = \sum_{h=1}^{N_B} \sum_{m\in\Psi} Y_{ih}^{(\gamma,m)} V_h^{(m)} - J_{SS,i}^{(\gamma)} + J_{TR,i}^{(\gamma)} \qquad ; \gamma = C, R, F \tag{3.33}$$

Rewriting the relevant variables into the complex rectangular form,

$$F_i^{(\gamma)} = G_i^{(\gamma)} + jH_i^{(\gamma)} \tag{3.34}$$

$$Y_{ih}^{(\gamma,m)} = g_{ih}^{(\gamma,m)} - jb_{ih}^{(\gamma,m)} \tag{3.35}$$

$$V_i^{(\gamma)} = u_i^{(\gamma)} + jv_i^{(\gamma)} \tag{3.36}$$

$$J_{SS,i}^{(\gamma)} = s_i^{(\gamma)} + jt_i^{(\gamma)} \tag{3.37}$$

$$J_{TR,i}^{(\gamma)} = q_i^{(\gamma)} + jr_i^{(\gamma)} \tag{3.38}$$

Therefore, the equation 3.33 can be transformed into the following equations.

$$F_i^{(\gamma)} = \sum_{h=1}^{N_B} \sum_{m\in\Psi} \left(g_{ih}^{(\gamma,m)} - jb_{ih}^{(\gamma,m)}\right)\left(u_h^{(m)} + jv_h^{(m)}\right) - \left(s_i^{(\gamma)} + jt_i^{(\gamma)}\right) + \left(q_i^{(\gamma)} + jr_i^{(\gamma)}\right) \tag{3.39}$$

$$G_i^{(\gamma)} = \sum_{h=1}^{N_B} \sum_{m\in\Psi} \left(g_{ih}^{(\gamma,m)} u_h^{(m)} + b_{ih}^{(\gamma,m)} v_h^{(m)}\right) - s_i^{(\gamma)} + q_i^{(\gamma)} \tag{3.40}$$

$$H_i^{(\gamma)} = \sum_{h=1}^{N_B} \sum_{m\in\Psi} \left(g_{ih}^{(\gamma,m)} v_h^{(m)} - b_{ih}^{(\gamma,m)} u_h^{(m)}\right) - t_i^{(\gamma)} + r_i^{(\gamma)} \tag{3.41}$$

Consider the current $\mathbf{J}_{TR,i}$ drawn by a train at bus i, due to the difficulty of assigning the current phase angle with respect to the substation voltage angle as the reference (zero degree), it can be rewritten in the form of the train power instead as follows.

$$\mathbf{J}_{\mathrm{TR,i}} = I_{T,i}\begin{bmatrix}1\\-1\\0\end{bmatrix} = \begin{bmatrix}J_{TR,i}^{(C)}\\J_{TR,i}^{(R)}\\J_{TR,i}^{(F)}\end{bmatrix} = \begin{bmatrix}\mathrm{q_i^{(C)}} + \mathrm{jr_i^{(C)}}\\ \mathrm{q_i^{(R)}} + \mathrm{jr_i^{(R)}}\\ \mathrm{q_i^{(F)}} + \mathrm{jr_i^{(F)}}\end{bmatrix} \tag{3.42}$$

$$\begin{aligned} I_{T,i} &= \frac{S_{TR,i}^{*}}{\left(V_i^{(C)} - V_i^{(R)}\right)^{*}} = \frac{\mathrm{w_i} - \mathrm{jz_i}}{\left(\mathrm{u_i^{(C)}} - \mathrm{u_i^{(R)}}\right) - \mathrm{j}\left(\mathrm{v_i^{(C)}} - \mathrm{v_i^{(R)}}\right)} \\ &= \frac{\left\{\mathrm{w_i}\left(\mathrm{u_i^{(C)}} - \mathrm{u_i^{(R)}}\right) + \mathrm{z_i}\left(\mathrm{v_i^{(C)}} - \mathrm{v_i^{(R)}}\right)\right\}}{\left(\mathrm{u_i^{(C)}} - \mathrm{u_i^{(R)}}\right)^2 + \left(\mathrm{v_i^{(C)}} - \mathrm{v_i^{(R)}}\right)^2} + \mathrm{j}\frac{\left\{\mathrm{w_i}\left(\mathrm{v_i^{(C)}} - \mathrm{v_i^{(R)}}\right) - \mathrm{z_i}\left(\mathrm{u_i^{(C)}} - \mathrm{u_i^{(R)}}\right)\right\}}{\left(\mathrm{u_i^{(C)}} - \mathrm{u_i^{(R)}}\right)^2 + \left(\mathrm{v_i^{(C)}} - \mathrm{v_i^{(R)}}\right)^2} \end{aligned} \tag{3.43}$$

Substituting equation 3.43 into equation 3.42, therefore

$$\mathrm{q_i^{(C)}} = \frac{\alpha_\mathrm{i}}{\Phi_\mathrm{i}} \tag{3.44}$$

$$\mathrm{q_i^{(R)}} = -\mathrm{q_i^{(C)}} \tag{3.45}$$

$$\mathrm{q_i^{(F)}} = 0 \tag{3.46}$$

$$\mathrm{r_i^{(C)}} = \frac{\beta_\mathrm{i}}{\Phi_\mathrm{i}} \tag{3.47}$$

$$\mathrm{r_i^{(R)}} = -\mathrm{r_i^{(C)}} \tag{3.48}$$

$$\mathrm{r_i^{(F)}} = 0 \tag{3.49}$$

$$\Delta\mathrm{u_i} = \mathrm{u_i^{(C)}} - \mathrm{u_i^{(R)}} \tag{3.50}$$

$$\Delta\mathrm{v_i} = \mathrm{v_i^{(C)}} - \mathrm{v_i^{(R)}} \tag{3.51}$$

$$\alpha_\mathrm{i} = \mathrm{w_i}\Delta\mathrm{u_i} + \mathrm{z_i}\Delta\mathrm{v_i} \tag{3.52}$$

$$\beta_\mathrm{i} = \mathrm{w_i}\Delta\mathrm{v_i} - \mathrm{z_i}\Delta\mathrm{u_i} \tag{3.53}$$

$$\Phi_i = (\Delta u_i)^2 + (\Delta v_i)^2 \tag{3.54}$$

Now let $G_i^{(\gamma)} = a_i^{(\gamma)} + q_i^{(\gamma)}$ and $H_i^{(\gamma)} = d_i^{(\gamma)} + r_i^{(\gamma)}$, that is

$$a_i^{(\gamma)} = \sum_{h=1}^{N_B} \sum_{m\in\Psi} \left(g_{ih}^{(\gamma,m)} u_h^{(m)} + b_{ih}^{(\gamma,m)} v_h^{(m)} \right) - s_i^{(\gamma)} \tag{3.55}$$

$$d_i^{(h)} = \sum_{h=1}^{N} \sum_{m\in\Psi} \left(g_{ih}^{(\gamma,m)} v_h^{(m)} - b_{ih}^{(\gamma,m)} u_h^{(m)} \right) - t_i^{(\gamma)} \tag{3.56}$$

Approximate the current-mismatch equations by Taylor series expansion and collect only the first two terms of the series, thus

$$G_i^{(\gamma)} = \sum_{h=1}^{N_B} \sum_{m\in\Psi} \frac{\partial G_i^{(\gamma)}}{\partial u_h^{(m)}} \cdot \Delta u_h^{(m)} + \sum_{h=1}^{N_B} \sum_{m\in\Psi} \frac{\partial G_i^{(\gamma)}}{\partial v_h^{(m)}} \cdot \Delta v_h^{(m)} \tag{3.57}$$

$$H_i^{(\gamma)} = \sum_{h=1}^{N_B} \sum_{m\in\Psi} \frac{\partial H_i^{(\gamma)}}{\partial u_h^{(m)}} \cdot \Delta u_h^{(m)} + \sum_{h=1}^{N_B} \sum_{m\in\Psi} \frac{\partial H_i^{(\gamma)}}{\partial v_h^{(m)}} \cdot \Delta v_h^{(m)} \tag{3.58}$$

By collecting all the equations from bus 1 to bus N_B, the NR power flow formula of the AT power feeding system can be obtained by equation 3.59.

$$\begin{aligned} \begin{bmatrix} G \\ H \end{bmatrix} &= \begin{bmatrix} \frac{\partial G}{\partial u} & \frac{\partial G}{\partial v} \\ \frac{\partial H}{\partial u} & \frac{\partial H}{\partial v} \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \begin{bmatrix} \frac{\partial a}{\partial u} + \frac{\partial q}{\partial u} & \frac{\partial a}{\partial v} + \frac{\partial q}{\partial v} \\ \frac{\partial d}{\partial u} + \frac{\partial r}{\partial u} & \frac{\partial d}{\partial v} + \frac{\partial r}{\partial v} \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} \\ &= \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} = \begin{bmatrix} M_1 + L_1 & M_2 + L_2 \\ M_3 + L_3 & M_4 + L_4 \end{bmatrix} \begin{bmatrix} \Delta u \\ \Delta v \end{bmatrix} \end{aligned} \tag{3.59}$$

The bus voltage of the k^{th} iteration is thus updated by the following equation.

$$[\mathbf{V}]^{(k+1)} = [\mathbf{V}]^{(k)} + ([\Delta u] + j[\Delta v]) \tag{3.60}$$

However, the equation 3.59 cannot be solved unless all elements of theJacobian matrices are calculated. With equations 3.44 – 3.58 and their derivatives with respect to u and v, the Jacobian sub-matrices can be determined by the following expressions.

- $[M_1]$ to $[M_4]$:
  - *Matrix* $[M_1]$:

$$\frac{\partial a_i^{(\gamma)}}{\partial u_h^{(m)}} = g_{ih}^{(\gamma,m)} \tag{3.61a}$$

  - *Matrix* $[M_2]$:

$$\frac{\partial a_i^{(\gamma)}}{\partial v_h^{(m)}} = b_{ih}^{(\gamma,m)} \tag{3.61b}$$

  - *Matrix* $[M_3]$:

$$\frac{\partial d_i^{(\gamma)}}{\partial u_h^{(m)}} = -b_{ih}^{(\gamma,m)} \tag{3.61c}$$

  - *Matrix* $[M_4]$:

$$\frac{\partial d_i^{(\gamma)}}{\partial v_h^{(m)}} = g_{ih}^{(\gamma,m)} \tag{3.61d}$$

- $[L_1]$ to $[L_4]$:
  - *Matrix* $[L_1]$:

For $i \neq h$

$$\frac{\partial q_i^{(\gamma)}}{\partial u_h^{(m)}} = 0 \qquad \text{for } \forall i \text{ and } \forall h \tag{3.62a}$$

For $i = h$

(a) $\gamma = C$ and $m = C$

$$\frac{\partial q_i^{(C)}}{\partial u_h^{(C)}} = \frac{w_i \Phi_i - 2\Delta u_i \alpha_i}{\Phi_i^2} \tag{3.62b}$$

(b) $\gamma = C$ and $m = R$

$$\frac{\partial q_i^{(C)}}{\partial u_h^{(R)}} = \frac{-w_i \Phi_i + 2\Delta u_i \alpha_i}{\Phi_i^2} \tag{3.62c}$$

(c) γ = R and m = C

$$\frac{\partial q_i^{(R)}}{\partial u_h^{(C)}} = -\frac{w_i \Phi_i - 2\Delta u_i \alpha_i}{\Phi_i^2} \tag{3.62d}$$

(d) γ = R and m = R

$$\frac{\partial q_i^{(R)}}{\partial u_h^{(R)}} = -\frac{-w_i \Phi_i + 2\Delta u_i \alpha_i}{\Phi_i^2} \tag{3.62e}$$

(e) Others,

$$\frac{\partial q_i^{(\gamma)}}{\partial u_h^{(m)}} = 0 \tag{3.62f}$$

- *Matrix* $[L_2]$:

For $i \neq h$

$$\frac{\partial q_i^{(\gamma)}}{\partial v_h^{(m)}} = 0 \qquad \text{for } \forall i \text{ and } \forall h \tag{3.63a}$$

For $i = h$

(a) γ = C and m = C

$$\frac{\partial q_i^{(C)}}{\partial v_h^{(C)}} = \frac{z_i \Phi_i - 2\Delta v_i \alpha_i}{\Phi_i^2} \tag{3.63b}$$

(b) γ = C and m = R

$$\frac{\partial q_i^{(C)}}{\partial v_h^{(R)}} = \frac{-z_i \Phi_i + 2\Delta v_i \alpha_i}{\Phi_i^2} \tag{3.63c}$$

(c) γ = R and m = C

$$\frac{\partial q_i^{(R)}}{\partial v_h^{(C)}} = -\frac{z_i \Phi_i - 2\Delta v_i \alpha_i}{\Phi_i^2} \tag{3.63d}$$

(d) γ = R and m = R

$$\frac{\partial q_i^{(R)}}{\partial v_h^{(R)}} = -\frac{-z_i\Phi_i + 2\Delta v_i\alpha_i}{\Phi_i^2} \tag{3.63e}$$

(e) Others,

$$\frac{\partial q_i^{(\gamma)}}{\partial v_h^{(m)}} = 0 \tag{3.63f}$$

- *Matrix* $[L_3]$:

For $i \neq h$

$$\frac{\partial r_i^{(\gamma)}}{\partial u_h^{(m)}} = 0 \qquad \text{for } \forall i \text{ and } \forall h \tag{3.64a}$$

For $i = h$

(a) $\gamma = C$ and $m = C$

$$\frac{\partial r_i^{(C)}}{\partial u_h^{(C)}} = \frac{-z_i\Phi_i - 2\Delta u_i\beta_i}{\Phi_i^2} \tag{3.64b}$$

(b) $\gamma = C$ and $m = R$

$$\frac{\partial r_k^{(C)}}{\partial x_k^{(R)}} = \frac{z_k\Phi_k + 2\Delta x_k\beta_k}{\Phi_k^2} \tag{3.64c}$$

(c) $\gamma = R$ and $m = C$

$$\frac{\partial r_i^{(R)}}{\partial u_h^{(C)}} = -\frac{-z_i\Phi_i - 2\Delta u_i\beta_i}{\Phi_i^2} \tag{3.64d}$$

(d) $\gamma = R$ and $m = R$

$$\frac{\partial q_i^{(R)}}{\partial u_h^{(R)}} = -\frac{z_i\Phi_i + 2\Delta u_i\beta_i}{\Phi_i^2} \tag{3.64e}$$

(e) Others,

$$\frac{\partial r_i^{(\gamma)}}{\partial u_h^{(m)}} = 0 \tag{3.64f}$$

- *Matrix* $[L_4]$:

  For $i \neq h$

$$\frac{\partial r_i^{(\gamma)}}{\partial v_h^{(m)}} = 0 \qquad \text{for } \forall i \text{ and } \forall h \tag{3.65a}$$

  For $i = h$

  (a) $\gamma = C$ and $m = C$

$$\frac{\partial r_i^{(C)}}{\partial v_h^{(C)}} = \frac{w_i \Phi_i - 2\Delta v_i \beta_i}{\Phi_i^2} \tag{3.65b}$$

  (b) $\gamma = C$ and $m = R$

$$\frac{\partial r_i^{(C)}}{\partial v_h^{(R)}} = \frac{-w_i \Phi_i + 2\Delta v_i \beta_i}{\Phi_i^2} \tag{3.65c}$$

  (c) $\gamma = R$ and $m = C$

$$\frac{\partial r_i^{(R)}}{\partial v_h^{(C)}} = -\frac{w_i \Phi_i - 2\Delta v_i \beta_i}{\Phi_i^2} \tag{3.65d}$$

  (d) $\gamma = R$ and $m = R$

$$\frac{\partial r_i^{(R)}}{\partial v_h^{(R)}} = -\frac{-w_i \Phi_i + 2\Delta v_i \beta_i}{\Phi_i^2} \tag{3.65e}$$

  (e) Others,

$$\frac{\partial r_i^{(\gamma)}}{\partial v_h^{(m)}} = 0 \tag{3.65f}$$

3.4 Sequential Linear Power Flow Method

The NR method requires complicated computations as described in the previous section. Substantial memory is needed to store the Jacobian matrix. So, most previous works reject the use of the NR method for the AC railway power-flow calculation, and simple iterative methods

to solve the equation of [**Y**][**V**] = [**J**] are widely used [2,3,10,36,41]. In this thesis, a new power-flow solution method is developed. It is different from the other methods. Most methods require linearisation of the load, but this can cause inaccurate results. The newly developed method uses the original non-linear model, but linearises non-linear power flow equations instead. The proposed power flow method is named "Sequential Linear Power Flow Method: SLPFM".

3.4.1 Sequential linear power flow method for a single-phase AC railway power feeding system

Consider an N_B-bus AC railway power system. As described previously, the power flow equation obtained from bus i can be expressed as follows.

$$\sum_{h=1}^{N_B} Y_{ih} V_h = J_i = J_{G,i} - J_{D,i} \tag{3.66}$$

Assume that all loads are modelled by a constant power. If the actual bus voltages are known, the equation above is numerically balanced. In fact, the bus voltages are unknowns unless the power flow problem has already been solved. To perform the power flow calculation, all the unknown bus voltages need to be assigned initial guesses. With initial guess bus voltages, the bus current J_i can be computed. In equation 3.66, $J_{G,i}$ represents the current injected by a connected generator. In the railway system, it will be non-zero for only the substation bus having a current source in parallel with a shunt admittance connected to the bus. Otherwise, $J_{G,i} = 0$. The second term, $J_{D,i}$ represents the calculated load current. Let $S_{TR,i}$ be the train power, the train current is evaluated by equation 3.67.

$$J_{D,i} = \left(\frac{S_{TR,i}}{V_i} \right)^* \tag{3.67}$$

With this calculation, the system matrix equation is therefore formed.

$$\begin{bmatrix} Y_{11} & Y_{12} & \cdots & Y_{1N_B} \\ Y_{21} & Y_{22} & \cdots & Y_{2N_B} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{N_B1} & Y_{N_B2} & \cdots & Y_{N_BN_B} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_{N_B} \end{bmatrix} = \begin{bmatrix} J_1 \\ J_2 \\ \vdots \\ J_{N_B} \end{bmatrix} \tag{3.68}$$

or in the compact form,

$$\mathbf{YV} = \mathbf{J} \tag{3.69}$$

Equation 3.69 is linear and can be solved by some appropriate method. Given that the voltage vector of the k^{th} iteration is already known as $\mathbf{V}^{(k)}$, the corresponding bus current, $\mathbf{J}^{(k)}$, is then obtained. Therefore, the updated voltage solution for iteration k+1 can be found by solving the following linear equation.

$$\mathbf{V}^{(k+1)} = \mathbf{Y}^{-1}\mathbf{J}^{(k)} \tag{3.70}$$

Equation 3.70 is quite simple but direct inversion method for solving a linear system like this is inefficient and not often used. Efficient methods for solving this linear system will be carefully investigated in the next chapter.

The iterative process to update the bus voltage will be carried on until some particular termination criteria is met. In this thesis, maximum voltage error is used as follows.

$$\max_{i} \left(\frac{\left| V_i^{(k+1)} - V_i^{(k)} \right|}{\left| V_i^{(k)} \right|} \right) < \varepsilon \tag{3.71}$$

Where i = 1, 2, …, N_B, ε is a small positive number and set at $1x10^{-8}$. This means that if the nominal voltage is 25 kV, the maximum voltage error must be less than $2.5x10^{-4}$ V for the iteration to terminate.

Furthermore, this method can be used for some form of load models such as the impedance, the current etc, as long as the equipment load current can be calculated. The following gives the calculation of the load current for some different kinds of the load models.

- Load impedance given by $z_{t,\mathrm{i}}$

$$J_{D,i} = \frac{V_i}{z_{t,i}} \qquad \text{; as shown in Fig. 3.3c} \tag{3.72}$$

- Load admittance given by $y_{t,i} = z_{t,i}^{-1}$

$$J_{D,i} = y_{t,i} V_{\mathrm{i}} \tag{3.73}$$

- Linear load as a voltage source, $E_{t,i}$, in series with an impedance, $z_{t,i}$

$$J_{D,i} = \frac{V_i - E_{t,i}}{z_{t,i}} \qquad \text{; as shown in Fig. 3.3b} \tag{3.74}$$

3.4.2 Sequential linear power flow method for bi-phase AC railway power feeding system

Each bus of the AT systems is defined as a collection of three separate conductor nodes. Currents or voltages at any bus are complex vectors consisting of three components representing C, R and F conductors as given in equation 3.28 – 3.29. The power flow equations for this case have already been described in equation 3.30. Instead of applying the NR method, a simple iterative method that does not require any derivative is used. The concept of this calculation is similar to that of the single-phase AC railway system. However, it needs some modifications due to the difference between the number of the conductors used. Consider the following equation.

$$\sum_{\mathrm{h=1}}^{\mathrm{N_B}} \mathbf{Y}_{\mathrm{ih}} \mathbf{V}_{\mathrm{h}} = \mathbf{J}_{\mathrm{i}} = \mathbf{J}_{\mathrm{SS,i}} - \mathbf{J}_{\mathrm{TR,i}} \tag{3.75}$$

$[\mathbf{J}]_{\mathrm{SS,i}}$ is the current supplied by the substation if bus i is the substation bus (as described in equation 3.3), otherwise it is set as a zero vector. The second term is the train current also having three conductor components. $\mathbf{J}_{\mathrm{TR,i}}$ can be computed by the following expression: -

$$\mathbf{J}_{\mathrm{TR,i}} = \begin{bmatrix} J_{TR,i}^{(C)} \\ J_{TR,i}^{(R)} \\ J_{TR,i}^{(F)} \end{bmatrix} = \left(\frac{S_{TR,i}}{V_i^{(C)} - V_i^{(R)}} \right)^* \cdot \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} \tag{3.76}$$

With this calculation, the system matrix equation is generated as shown in equation 3.77.

$$\begin{bmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} & \cdots & \mathbf{Y}_{1\mathrm{N_B}} \\ \mathbf{Y}_{21} & \mathbf{Y}_{22} & \cdots & \mathbf{Y}_{2\mathrm{N_B}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{Y}_{\mathrm{N_B}1} & \mathbf{Y}_{\mathrm{N_B}2} & \cdots & \mathbf{Y}_{\mathrm{N_B}\mathrm{N_B}} \end{bmatrix} \begin{bmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \\ \vdots \\ \mathbf{V}_{\mathrm{N_B}} \end{bmatrix} = \begin{bmatrix} \mathbf{J}_1 \\ \mathbf{J}_2 \\ \vdots \\ \mathbf{J}_{\mathrm{N_B}} \end{bmatrix} \tag{3.77}$$

or in the compact form,

$$[\mathbf{Y}][\mathbf{V}] = [\mathbf{J}] \tag{3.78}$$

Thus, the updated bus voltage for iteration k+1 can be found by solving the following linear equation.

$$[\mathbf{V}]^{(\mathrm{k+1})} = [\mathbf{Y}]^{-1}[\mathbf{J}]^{(\mathrm{k})} \tag{3.79}$$

The iterative process to update the bus voltage will be performed repeatedly until one of termination criteria is met. In this thesis, maximum voltage error is used as follows.

$$\max_{\mathrm{i}} \left\{ \max_{\gamma} \left(\frac{\left| \left[V^{(\gamma)} \right]_{\mathrm{i}}^{(\mathrm{k+1})} - \left[V^{(\gamma)} \right]_{\mathrm{i}}^{(\mathrm{k})} \right|}{\left| \left[V^{(\gamma)} \right]_{\mathrm{i}}^{(\mathrm{k})} \right|} \right) \right\} < \varepsilon \tag{3.80}$$

Where i = 1, 2, …, N_B, $\gamma \in \Psi$, and ε is a small positive number

3.5 Convergence, Memory Analysis and Computer Programming

Section 3.3 and 3.4 describe two efficient algorithms to solve the power flow problems for the AC railway power feeding systems. Before developing a simulation program, the convergence of the proposed method needs to be studied and also the memory requirement for each method.

3.5.1 Proof of convergence for the SLPFM

Given a system of N non-linear equations as follows: -

$$f_i(\mathbf{x}) = 0 \qquad ; i = 1, 2, \ldots, N \tag{3.81}$$

where $f_i(\mathbf{x})$ is assumed to be a continuous, real-valued function and differentiable. Most numerical methods for solving such a system are based on determining an initial approximation, $\mathbf{x}_0$, of the root. Starting with $\mathbf{x}_0$, a solution sequence $\{\mathbf{x}_j\}$ is constructed which then converges on the root, $\tilde{\mathbf{x}}$, as equation 3.82.

$$\lim_{j\to\infty} \mathbf{x}_j = \tilde{\mathbf{x}} \tag{3.82}$$

From the approximation $\mathbf{x}_k$ for the k^{th} iteration, a further approximation, $\mathbf{x}_{k+1}$, can be obtained as a function of $\mathbf{x}_k$ as follows.

$$\mathbf{x}_{k+1} = \varphi(\mathbf{x}_k) \tag{3.83}$$

This recursive process requires that $\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_\infty$ decreases as k increases. Note that the ∞-norm of any vector $\mathbf{x}$ is used to obtain the maximum absolute value of $\mathbf{x}$. If $\mathbf{x} = \begin{bmatrix} x^{(1)} & x^{(2)} & \cdots & x^{(N)} \end{bmatrix}^T$, then $\|\mathbf{x}\|_\infty = \max_i \left|x^{(i)}\right|$. Conveniently, in this section, $\|\cdot\|$ is given as the ∞-norm rather than the two-norm (Euclidean-norm) as usual. Thus, the approximated solution moves towards the root $\tilde{\mathbf{x}} = \varphi(\tilde{\mathbf{x}})$ with a relatively large integer of k. With the help of the Taylor series expansion [14], thus

$$\varphi(\tilde{\mathbf{x}}) = \varphi(\mathbf{x}_k) + \nabla\varphi(\xi_k)\cdot(\tilde{\mathbf{x}} - \mathbf{x}_k) \tag{3.84}$$

where ξ_k is a point between the root and the approximation $\mathbf{x}_k$. However, $\xi_k = \mathbf{x}_k$ in the case of approximating with the first two-terms. For every point in the solution sequence, distances between the root and any approximated solution are evaluated.

$$\tilde{\mathbf{x}} - \mathbf{x}_1 = \varphi(\tilde{\mathbf{x}}) - \varphi(\mathbf{x}_0) = \nabla\varphi(\xi_0)\cdot(\tilde{\mathbf{x}} - \mathbf{x}_0) \tag{3.85.1}$$

$$\tilde{\mathbf{x}} - \mathbf{x}_2 = \varphi(\tilde{\mathbf{x}}) - \varphi(\mathbf{x}_1) = \nabla\varphi(\xi_1)\cdot(\tilde{\mathbf{x}} - \mathbf{x}_1) \tag{3.85.2}$$

$$\vdots \qquad \vdots \qquad \vdots \qquad \vdots$$

$$\tilde{\mathbf{x}} - \mathbf{x}_n = \varphi(\tilde{\mathbf{x}}) - \varphi(\mathbf{x}_{n-1}) = \nabla\varphi(\xi_{n-1})\cdot(\tilde{\mathbf{x}} - \mathbf{x}_{n-1}) \tag{3.85.n}$$

Multiplying all n equations above altogether, so that.

$$\tilde{\mathbf{x}} - \mathbf{x}_n = \nabla\varphi(\xi_0)\cdot\nabla\varphi(\xi_1)\cdot\cdots\cdot\nabla\varphi(\xi_{n-1})\cdot(\tilde{\mathbf{x}} - \mathbf{x}_0) \tag{3.86}$$

Consequently,

$$\left\|\tilde{\mathbf{x}} - \mathbf{x}_n\right\| = \left\|\prod_{k=0}^{n-1}\nabla\varphi(\xi_k)\cdot(\tilde{\mathbf{x}} - \mathbf{x}_0)\right\| \tag{3.87}$$

The error $\left\|\tilde{\mathbf{x}} - \mathbf{x}_n\right\|$ can be made as small as we please as long as $\left\|\nabla\varphi(\xi_k)\right\| < 1$ for $\forall$k.

Note that $\nabla\varphi(\xi_k)$ is a square matrix. In this case, the ∞-norm of the matrix, $\|.\|_\infty$ or conveniently as $\|.\|$, is given by $\|\mathbf{A}\| = \underset{i=1,2,\ldots,N}{\text{Max}} \sum_{j=1}^{N}\left|\mathbf{A}_{ij}\right|$.

The SLPFM approximates the root by using the expression [**V**] = [**Y**]$^{-1}$ [**J**], where [**J**] is a function of [**V**]. This is rewritten into a general form of $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}(\mathbf{x})$, where **x**, **A** and $\mathrm{b}(\mathbf{x})$ represent [**V**], [**Y**] and [**J**], respectively . Thus,

$$\mathbf{x}_{k+1} = \varphi(\mathbf{x}_k) = \mathbf{A}^{-1}\mathbf{b}(\mathbf{x}_k) \tag{3.88}$$

As described previously, this method converges when $\left\|\nabla\varphi(\mathbf{x}_k)\right\| < 1$ for $\forall$k. Therefore, the criterion for the convergence is found by.

$$\left\|\nabla\varphi(\mathbf{x}_k)\right\| = \left\|\mathbf{A}^{-1}\nabla\mathbf{b}(\mathbf{x}_k)\right\| \le \left\|\mathbf{A}^{-1}\right\|\cdot\left\|\nabla\mathbf{b}(\mathbf{x}_k)\right\| < 1 \qquad ; \text{for } \forall k \tag{3.89}$$

Hence,

$$\begin{aligned}\left\|\tilde{\mathbf{x}} - \mathbf{x}_k\right\| &= \left\|\tilde{\mathbf{x}} - \mathbf{A}^{-1}\mathrm{b}(\mathbf{x}_{k-1})\right\| = \left\|\mathbf{A}^{-1}(\tilde{\mathbf{x}} - \mathbf{x}_{k-1})\right\| \\ &\le \left\|\mathbf{A}^{-1}\right\|\cdot\left\|\tilde{\mathbf{x}} - \mathbf{x}_{k-1}\right\|\end{aligned} \tag{3.90}$$

As can be seen, the error at iteration k is linearly dependent on the error of the previous iteration, k-1. By starting with a good approximation, $\mathbf{x}_0$, sufficiently close to the root, this method converges within a few iterations.

3.5.2 Memory analysis of the power flow algorithms

Memory requirement is a big problem to develop a computer program for solving a power flow problem where memory resources are limited. Different power flow algorithms require significantly different memory spaces. In C/C++, real number variables can be declared as one of three variable types (*float*, *double* and *long double*, which correspond to a single precision, a double precision and an extended double precision real number, respectively) [44]. Normally, they occupy 4, 8 and 12 bytes, respectively. *float*, *double* and *long double* store about 6, 15 and 18 decimal places, respectively. In some compilers, a *long double* requires 16 bytes that can store data up to 33 decimal places [45]. In MATLAB [46], single-precision, double-precision or even sparse double-precision arrays are provided.

Flow charts for the NRPFM and SLPFM are presented in Fig. 3.8 and 3.9.

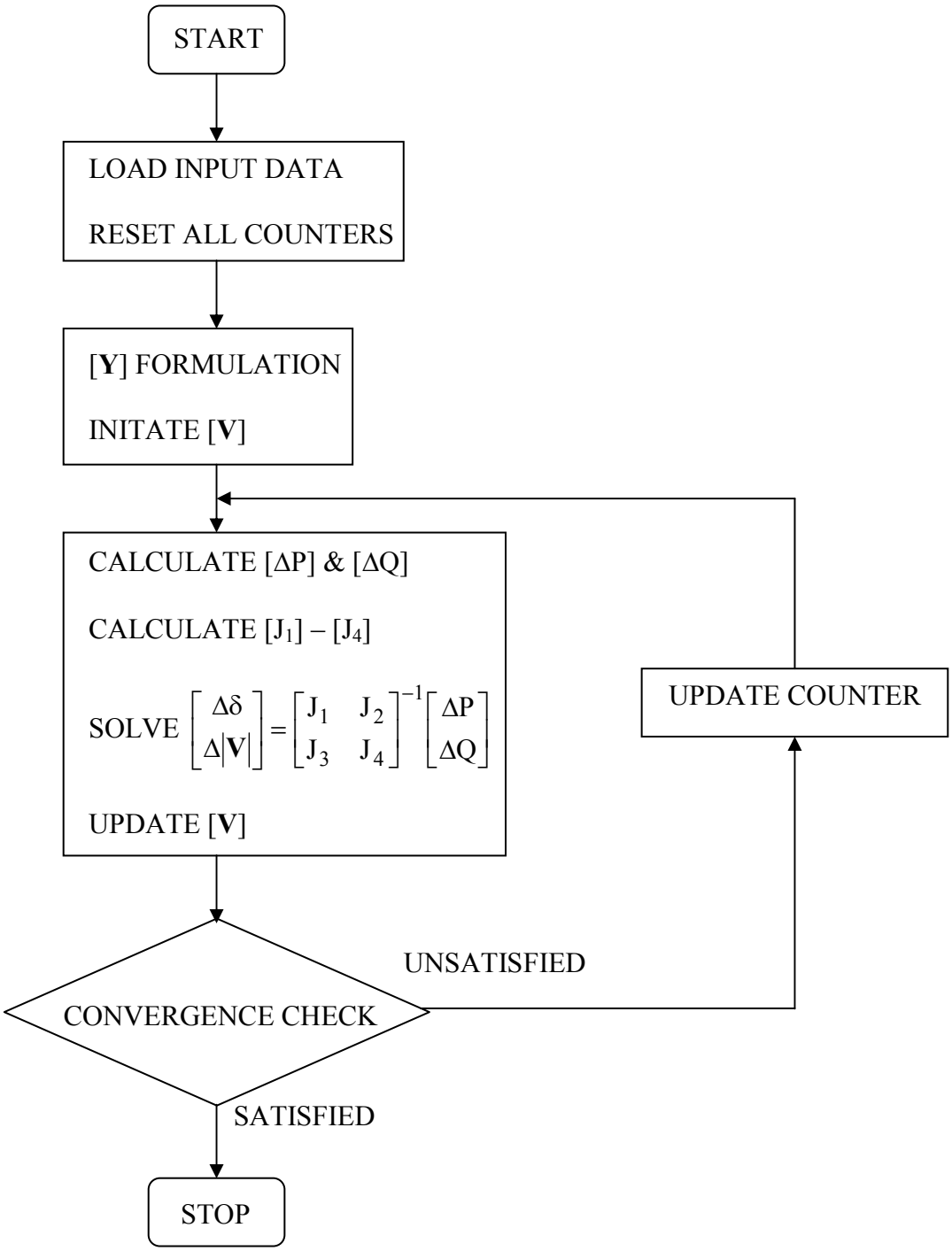


Fig. 3.8: Flow diagram of the NRPFM

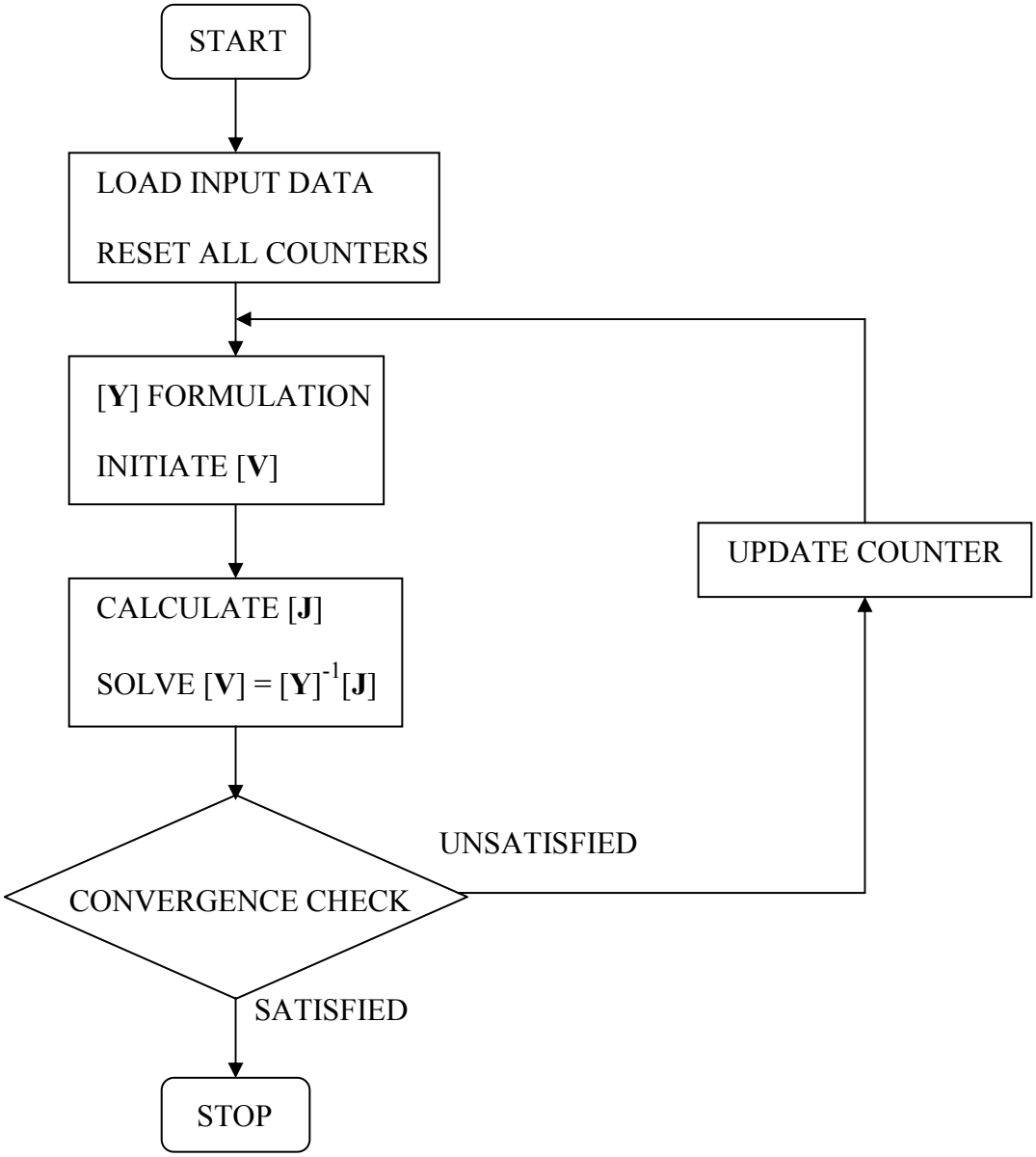


Fig. 3.9: Flow diagram of the SLPFM

In addition, Tables 3.1 and 3.2 list the use of standard routines by both methods.

Table 3.1: Program initialisation

Standard routine	NRPFM	SLPFM
1. Bus admittance matrix, [**Y**], formulation	Y	Y
2. Initiate bus voltage, [**V**]	Y	Y

Table 3.2: Iterative processes

Standard routine	NRPFM	SLPFM
1. Power-mismatch,[ΔP] & [ΔQ], calculation	Y	N
2. Jacobian matrix, $[J_1] – [J_4]$, calculation	Y	N
3. Current matrix, [**J**], calculation	N	Y
4. Solve a set of linear equations	Y	Y

- **Memory allocation**

Given that all variables used by a computer program are *double* and N is a total number of buses in the system. The memory requirement of each power flow method is presented in Table 3.3.

Table 3.3: Memory requirement of NRPFM and SLPFM

Variable	Memory Size (Byte)	Need updated?	Used by	
			NRPFM	SLPFM
[**Y**], Bus admittance matrix[+]	$16N^2$	N	Y	Y
[J], Jacobian matrix	$32N^2$	Y	Y	N
[ΔP] & [ΔQ], Power mismatch vectors	16N	Y	Y	N
[**J**], Bus current vector[+]	16N	Y	N	Y
[**V**], Bus voltage vector[+]	16N	Y	Y	Y

[+] A *double* complex-valued variable occupies twice the memory space of that required by a *double* real-valued variable

- Total memory space (TMS_{NR}) required by the NRPFM to store all the variables in Table 3.3 can be assessed by the following expression.

$$TMS_{NR} = Mem([\mathbf{Y}]) + Mem([J]) + Mem([\Delta P]) + Mem([\Delta Q]) + Mem([\mathbf{V}])$$

$$= 48N^2 + 32N \quad \text{Bytes}$$

- Total memory space (TMS_{SL}) required by the SLPFM to store all the variables in Table 3.3 can be assessed by the following expression.

$$TMS_{SL} = Mem([\mathbf{Y}]) + Mem([\mathbf{J}]) + Mem([\mathbf{V}])$$

$$= 16N^2 + 32N \quad \text{Bytes}$$

Where Mem(x) denotes the memory space required by the variable x.

Another key factor is to evaluate how much memory needs to be updated per iteration during the process.

- Total updated memory per iteration (TUM_{NR}) for the NRPFM is defined as follows.

$$TUM_{NR} = Mem([J]) + Mem([\Delta P]) + Mem([\Delta Q]) + Mem([\mathbf{V}])$$

$$= 32N^2 + 32N \qquad \text{Bytes}$$

- Total updated memory per iteration (TUM_{SL}) for the SLPFM is defined as follows.

$$TUM_{SL} = Mem([\mathbf{J}]) + Mem([\mathbf{V}])$$

$$= 32N \qquad \text{Bytes}$$

To compare memory requirements between the NRPFM and the SLPFM, TMS and TUM for both cases are graphically presented in Fig. 3.10.

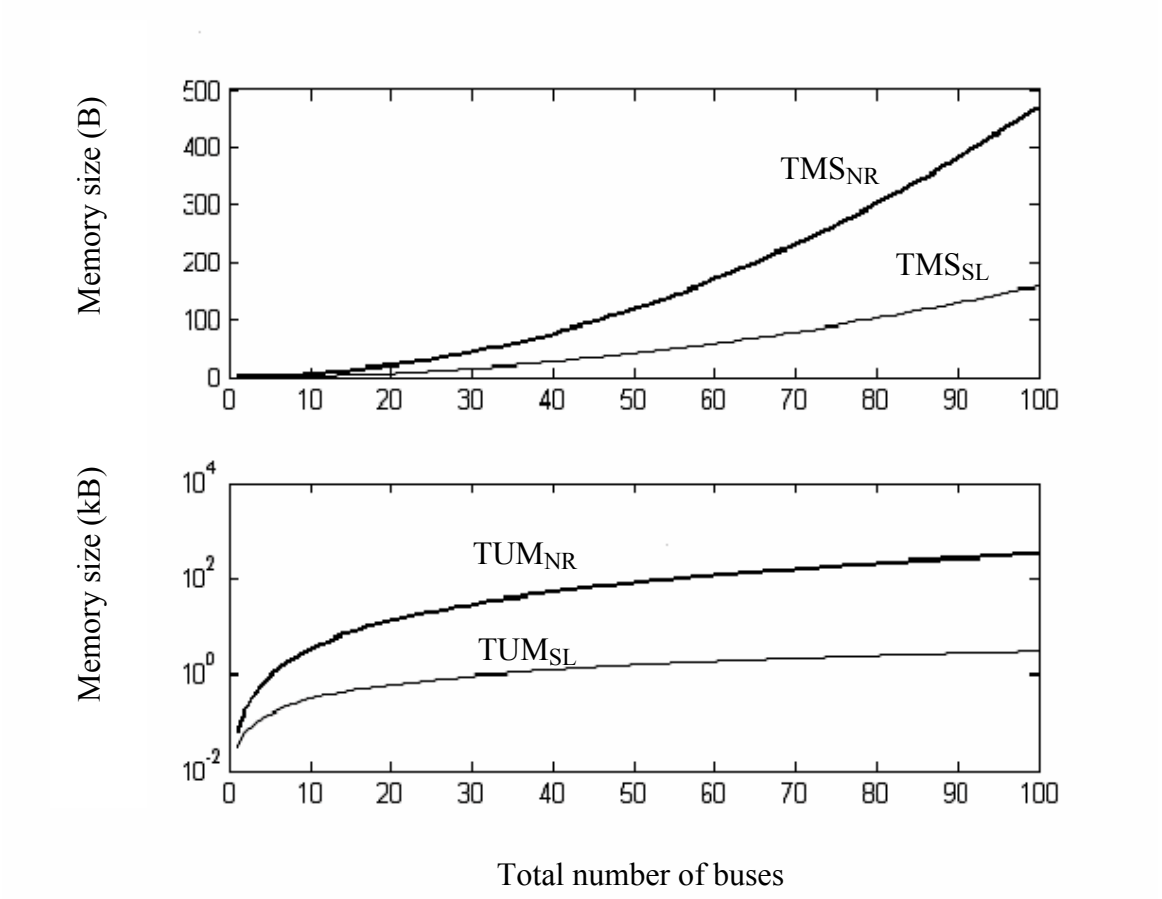


Fig. 3.10: Memory requirement of the NRPFM and SLPFM

The TMS comparison shows that the SLPFM requires less total memory space than that required by the NRPFM. However, this index does not indicate which method is faster. TUM is the amount of memory per iteration that is repetitively updated during the process. The smaller TUM, the faster is the update at each iteration. The convergence rate is another factor. The NRPFM has quadratic convergence [14] while the SLPFM has only linear convergence. This

means that the NRPFM uses less iterations than the SLPFM to find the solution starting from the same initial point.

3.6 Simulation Results and Discussions

The tests for AC railway power flow calculations are divided into two parts to cover both single- and bi-phase railway power feeding systems.

3.6.1 Single-phase AC railway power systems

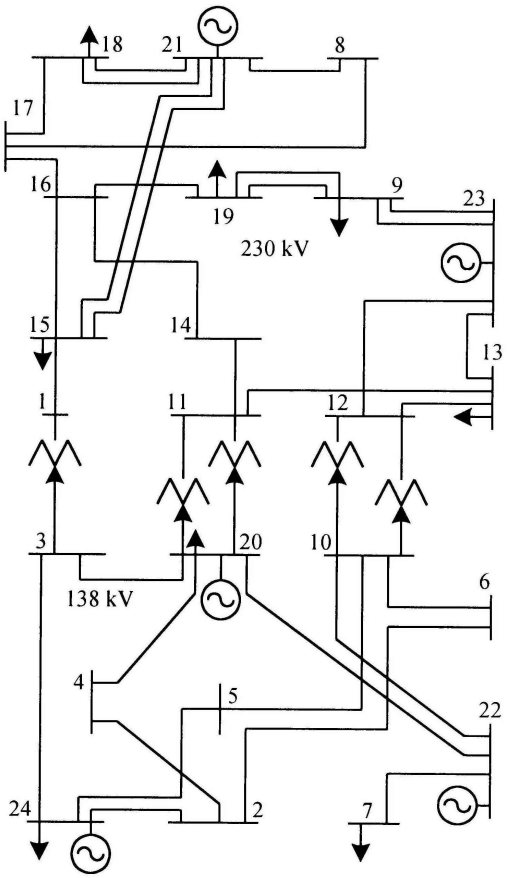


Fig. 3.11: The modified standard IEEE 24-bus test system

This first part is concerned with single-phase AC railway system. By using the latest version of MATPOWER [47], the well-known Newton-Raphson and Fast Decoupled power flow programs for the MATLAB environment developed by the school of Electrical Engineering, Cornell University, USA, to solve the modified standard IEEE 24- and 57-bus systems, and a 10-train railway test system as shown in Fig. 3.11 – 3.13 respectively, the effectiveness of the

SLPFM compared with the NRPFM and FDCPFM has been examined. The information of these three test systems can be found in Appendix E.1.

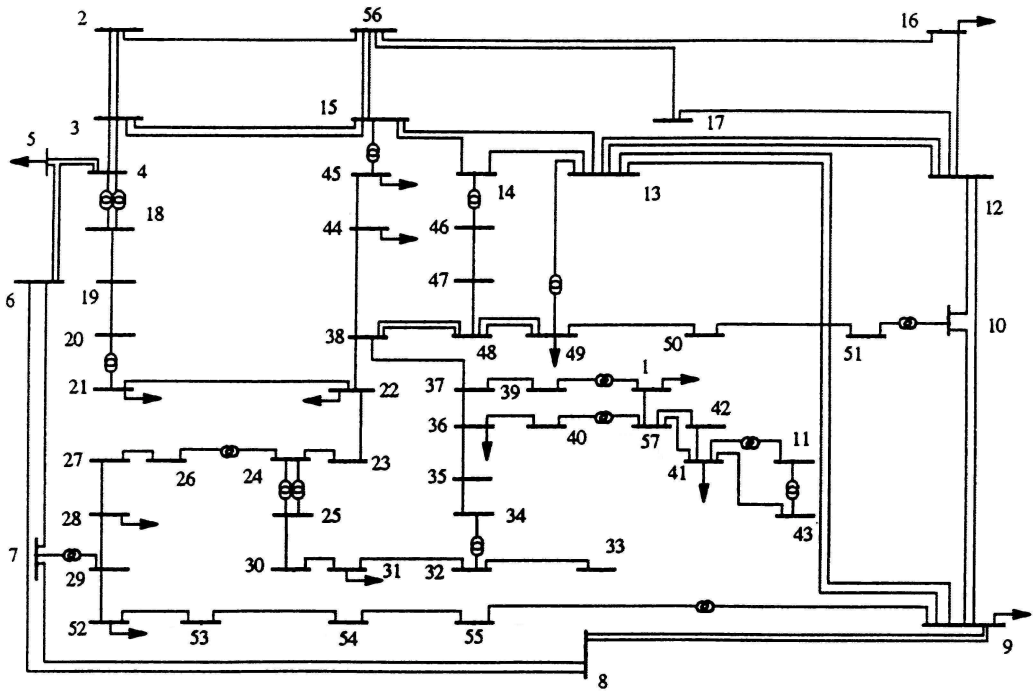
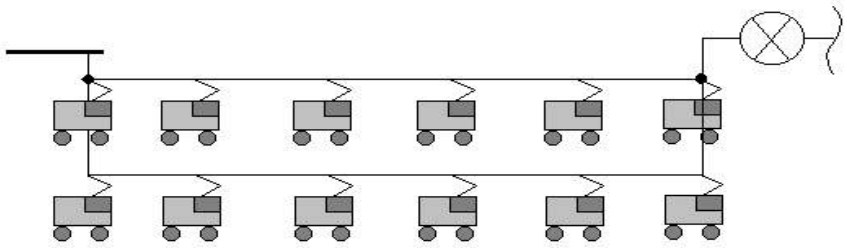
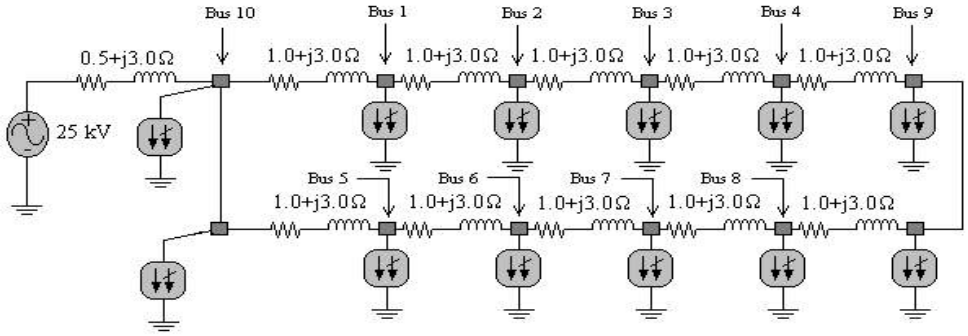


Fig. 3.12: The modified standard IEEE 57-bus test system



a) Simplified diagram of the 10-train AC railway system



b) Equivalent circuit

Fig. 3.13: The 10-train AC railway test system

These tests were performed on a Pentium 133 MHz, 48 MB-SDRAM with MATLAB 5.3 student version. It reveals that the results obtained by the three power flow methods are exactly the same, but the total number of iterations required and execution times are different depending on their individual performances. Fig. 3.14 shows the voltage profile of the 24-bus test system.

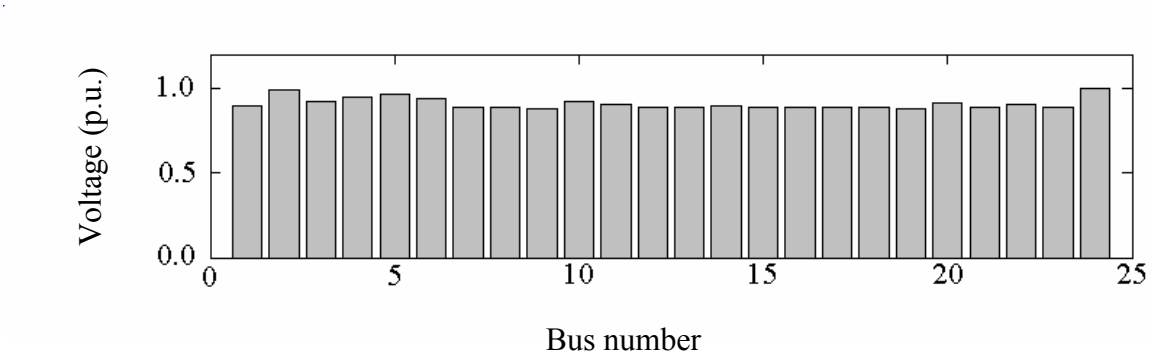


Fig. 3.14: Voltage profile for the modified standard IEEE-24 bus test system

With $1x10^{-8}$ equally applied to the relative termination criterion (maximum power mismatch for the NRPFM and FDCPFM, maximum voltage error for the SLPFM), their power flow solutions are compared. Fig. 3.15 and 3.16 present voltage magnitudes obtained for the IEEE 57-bus and the 10-train AC railway test systems. Note that all the numerical results for Section 3.6.1 are shown in the appendix F.1.

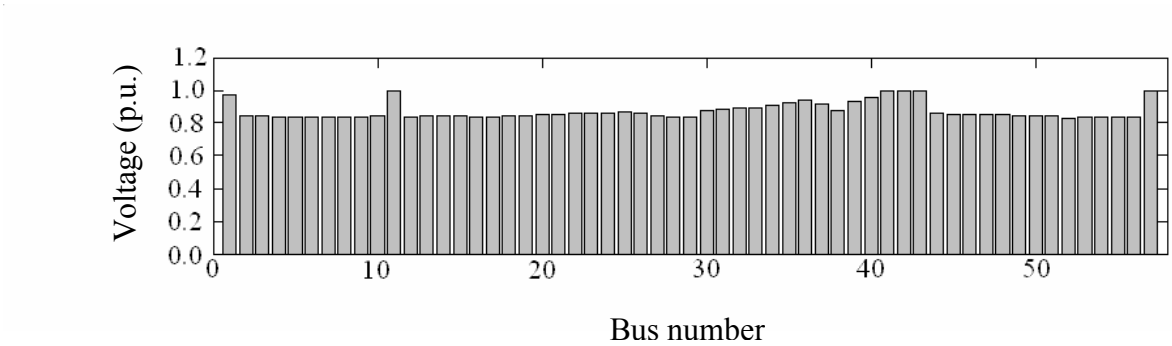


Fig. 3.15: Voltage profile of the modified standard IEEE-57 bus test system

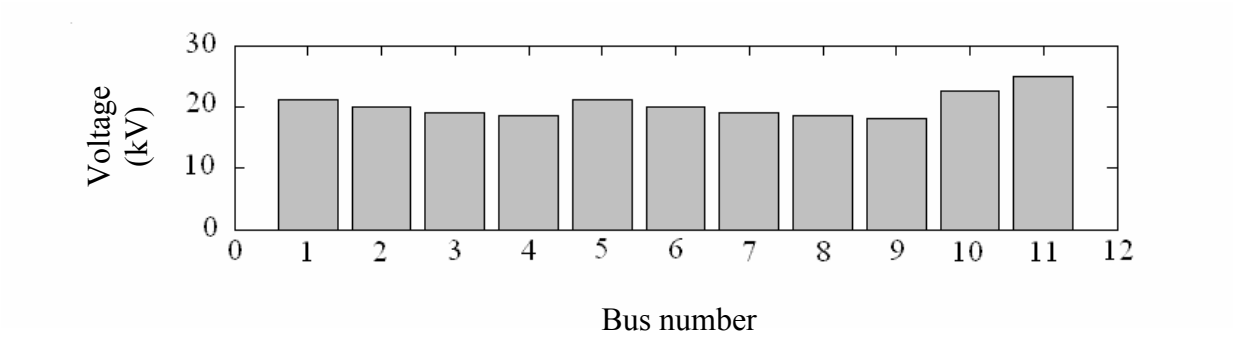


Fig. 3.16: Voltage profile of the 10-train AC railway test system

Key for diagrams 3.17 – 3.19:

SLPFM	Sequential Linear Power Flow Method
NRPFM	Newton-Raphson Power Flow Method
PDCPFM	Fast-Decouple Power Flow Method

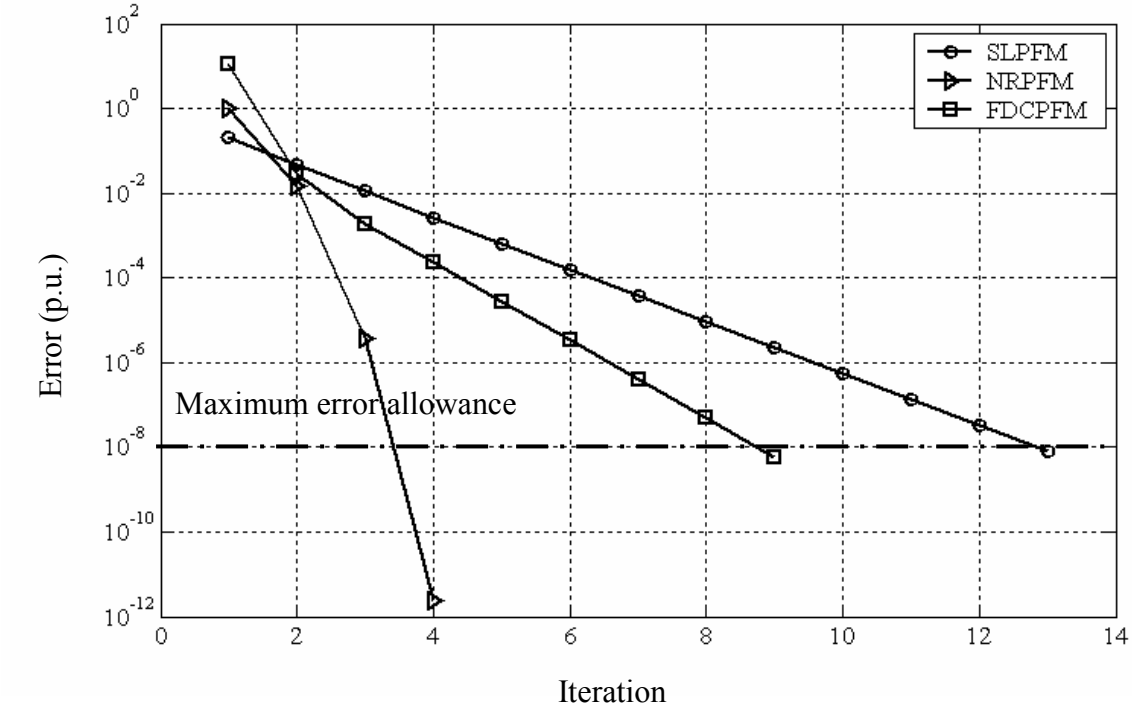


Fig. 3.17: Convergence for the modified standard IEEE 24-bus test system

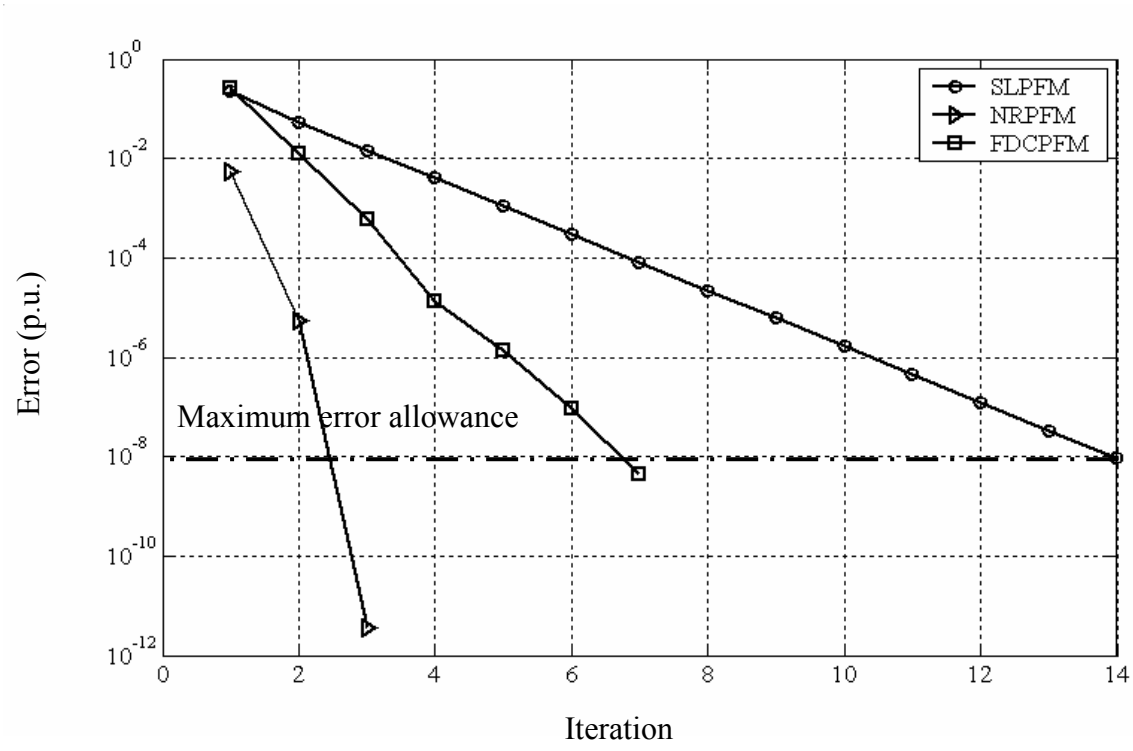


Fig. 3.18: Convergence for the modified standard IEEE 57-bus test system

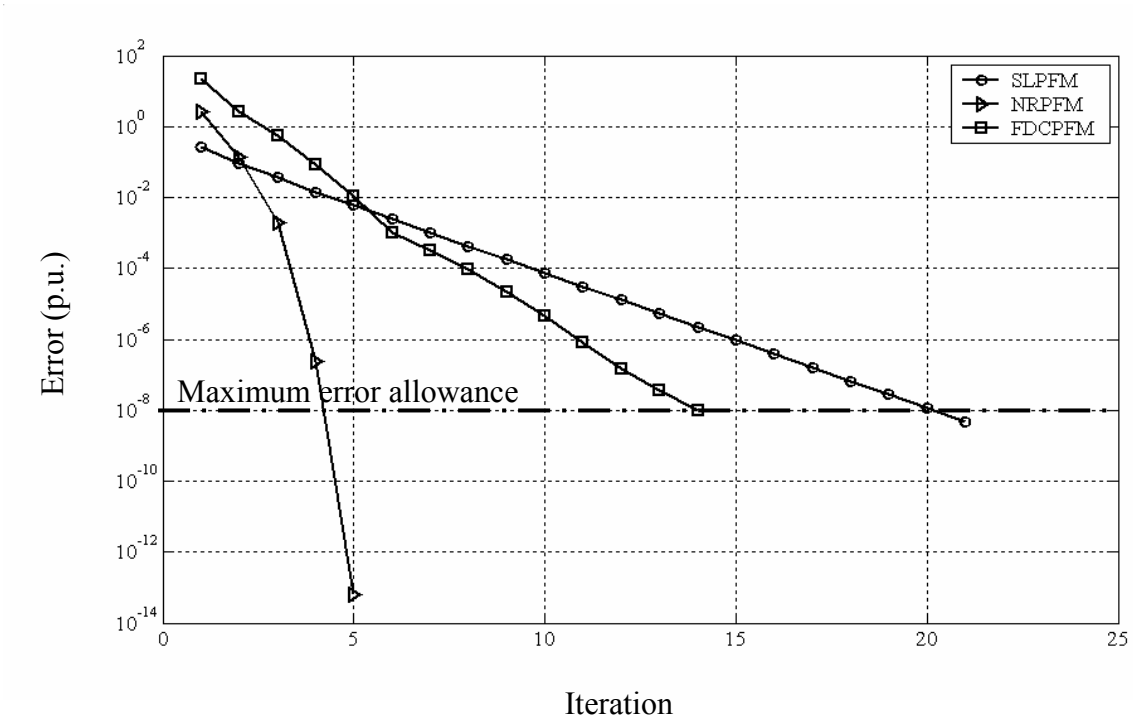


Fig. 3.19: Convergence for the 10-train AC railway test system

The convergence of the power flow methods for each test system is an essential indication to examine how the solution sequence moves towards the true solution and to show that the generated sequence is bounded. This roughly describes the rate of error reduction only and

cannot be used to judge the computational speed of the calculation. Thus, the execution times need to be observed carefully and also to be compared. Fig. 3.17 – 3.19 present the convergence property of the three respective test systems. In addition, the execution times for the three power flow methods applied to the three test systems are recorded and presented in Table 3.4 together with the number of iterations required as follows.

Table 3.4: Comparisons of execution times consumed and iterations required

Test case	Required iterations			Execution time (s)		
	SLPFM	NRPFM	FDCPFM	SLPFM	NRPFM	FDCPFM
24-bus	13	4	9	0.16	0.38	0.22
57-bus	14	3	7	0.24	0.44	0.33
10-train	21	5	14	0.22	0.44	0.32

With the quadratic convergence, the NRPFM requires the least number of iterations (4, 3 and 5 for the three respective test systems). The SLPFM has the linear convergence, on the other hand and therefore required the largest number of iterations, that is 13, 14 and 21 for the 24-, 57-bus and 10-train test systems, respectively. The convergence of the FDCPFM is in between (slower than quadratic but faster than linear); it used 9, 7 and 14 iterations, respectively. However, on the assessment of the overall execution time the SLPFM is the fastest method while the FDCPFM is the second and the NRPFM comes last. For the IEEE 24-bus system, the execution times are 0.16, 0.38 and 0.22 second for the SLPFM, NRPFM and FDCPFM, respectively. For the IEEE 57-bus system, they are 0.24, 0.44 and 0.32 second, respectively. Those of the 10-train test system are 0.22, 0.44 and 0.32 second, respectively.

3.6.2 Bi-phase AC railway power systems

With the special arrangement of the AT feeding system, the feeding circuit is more complicated and rather different from the industrial power system. However, with the significant effects of AT magnetisation and rail-to-earth leakage current, the resistance of a feeder portion cannot be ignored. This implies that the FDCPFM cannot be applied to this simulation. Also, from the work reported in [2] the mutual and capacitive coupling effects between tracks in multi-track

AT power feeding systems cause only minor changes in the overall system performances and thus are all neglected for this test.

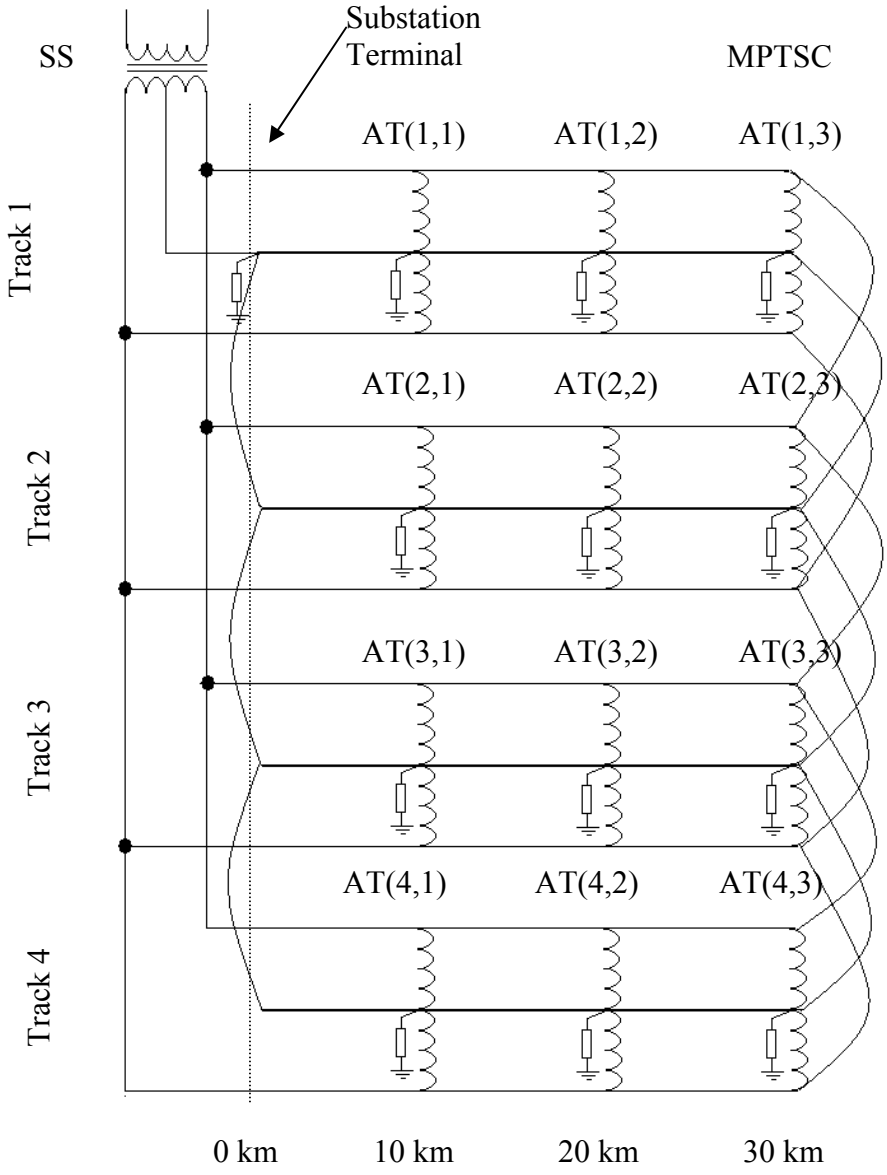


Fig. 3.20: Quad-track AT railway test system

Table 3.5: Train positions and powers for the double-track AT railway system

Track	Train number							
	1		2		3		4	
	D	*S*	D	*S*	D	*S*	D	*S*
1	8.0	1.0 - j0.8	12.0	1.0 + j0.8	19.0	1.0 - j0.8	25.0	1.0 + j0.8
2	6.0	1.0 - j0.8	12.0	1.0 + j0.8	18.0	1.0 + j0.8	26.0	1.0 - j0.8

Table 3.6: Train positions and powers for the quad-track AT railway system

Track	Train number							
	1		2		3		4	
	D	*S*	D	*S*	D	*S*	D	*S*
1	8.0	1.0 + j0.8	12.0	1.0 - j0.8	19.0	1.0 + j0.8	25.0	1.0 - j0.8
2	6.0	1.0 + j0.8	12.0	1.0 - j0.8	18.0	1.0 + j0.8	26.0	1.0 - j0.8
3	2.0	1.0 + j0.8	13.0	1.0 - j0.8	21.0	1.0 + j0.8	29.0	1.0 - j0.8
4	5.0	1.0 + j0.8	11.0	1.0 - j0.8	20.0	1.0 + j0.8	27.0	1.0 - j0.8

Note for Table 3.5 and 3.6, D is distance from the substation (km) and *S* is complex power (MVA)

Table 3.5 and 3.6 present train positions and powers of the double- and quad-track test systems, respectively. More information on these test systems can be found in Appendix E.2. The power flow solutions for these two test cases are shown in Table 3.7 and Table 3.8, while Fig. 3.21 and 3.22 show the convergence comparison between the two power flow methods for this test system. In addition, the execution times of both cases are presented in Table 3.9 and 3.10, respectively.

Table 3.7: Simulation results for the double-track AT railway system

Position	SLPFM			NRPFM		
	C (kV)	R (V)	F (kV)	C (kV)	R (V)	F (kV)
SS	24.2875	104.3	24.6508	24.875	104.3	24.6508
AT(1,1)	24.0592	39.6	24.7947	24.0592	39.3	24.7947
AT(2,1)	23.9568	39.2	24.8323	23.9568	39.2	24.8323
AT(1,2)	23.9343	22.4	24.8724	23.9343	22.4	24.8724
AT(2,2)	23.8176	27.4	24.9135	23.8176	27.4	24.9135
MPTSC	23.8563	17.6	24.9100	23.8563	17.6	24.9100
TR(1,1)	24.1158	37.2	24.7664	24.1158	37.2	24.7764
TR(2,1)	24.1058	39.8	24.7603	24.1058	39.8	24.7603
TR(1,2)	24.0061	46.2	24.8200	24.0061	46.2	24.8200
TR(2,2)	23.8898	46.8	24.8620	23.8898	46.8	24.8620
TR(1,3)	23.9518	21.7	24.8647	23.9518	21.7	24.8647
TR(2,3)	23.8072	40.4	24.9099	23.8072	40.4	24.9099
TR(1,4)	23.8497	38.5	24.9067	23.8497	38.5	24.9067
TR(2,4)	23.8685	18.1	24.9049	23.8685	18.1	24.9049

Table 3.8: Simulation results for the quad-track AT railway system

Position	SLPFM			NRPFM		
	C (kV)	R (V)	F (kV)	C (kV)	R (V)	F (kV)
SS	22.9008	150.5	23.9394	22.9008	150.5	23.9394
AT(1,1)	22.6959	46.0	24.1283	22.6959	46.0	24.1283
AT(2,1)	22.7113	38.6	24.1220	22.7113	38.6	24.1220
AT(3,1)	22.7749	30.1	24.0907	22.7749	30.1	24.0907
AT(4,1)	22.7404	30.5	22.1096	22.7404	30.5	22.1096
AT(1,2)	22.6667	36.4	24.1922	22.6667	36.4	24.1922
AT(2,2)	22.6774	30.0	24.1863	22.6774	30.0	24.1863
AT(3,2)	22.6852	29.9	24.176	22.6852	29.9	24.176
AT(4,2)$^{+}$	22.6586	34.8	24.1896	22.6586	34.8	24.1896
MPTSC	22.7547	25.0	24.1754	22.7547	25.0	24.1754
TR(1,1)	22.6980	68.5	24.1075	22.6980	68.5	24.1075
TR(2,1)	22.7261	80.5	24.0777	22.7261	80.5	24.0777
TR(3,1)	22.8361	120.0	23.9908	22.8361	120.0	23.9908
TR(4,1)	22.7573	87.3	24.0552	22.7573	87.3	24.0552
TR(1,2)	22.6989	27.7	24.1443	22.6989	27.7	24.1443
TR(2,2)	22.7109	23.8	24.1379	22.7109	23.8	24.1379
TR(3,2)	22.7692	16.2	24.1139	22.7692	16.2	24.1139
TR(4,2)	22.7396	21.3	24.1183	22.7396	21.3	24.1183
TR(1,3)	22.6540	42.5	24.1920	22.6540	42.5	24.1920
TR(2,3)	22.6576	41.0	24.1836	22.6576	41.0	24.1836
TR(3,3)	22.6765	35.2	24.1815	22.6765	35.2	24.1815
TR(1,4)	22.7416	28.1	24.1793	22.7416	28.1	24.1793
TR(2,4)	22.7546	28.2	24.1742	22.7546	28.2	24.1742
TR(3,4)	22.7566	26.4	24.1745	22.7566	26.4	24.1745
TR(4,4)	22.7524	28.4	24.1752	22.7524	28.4	24.1752

Note for Table 3.7 and 3.8, T(i,j) is train j on track i, AT(i,j) is autotransformer j on track i, and $^{+}$ indicates that train TR(4,3), which is not shown in Table 3.8, is at the same position of AT(4,3)

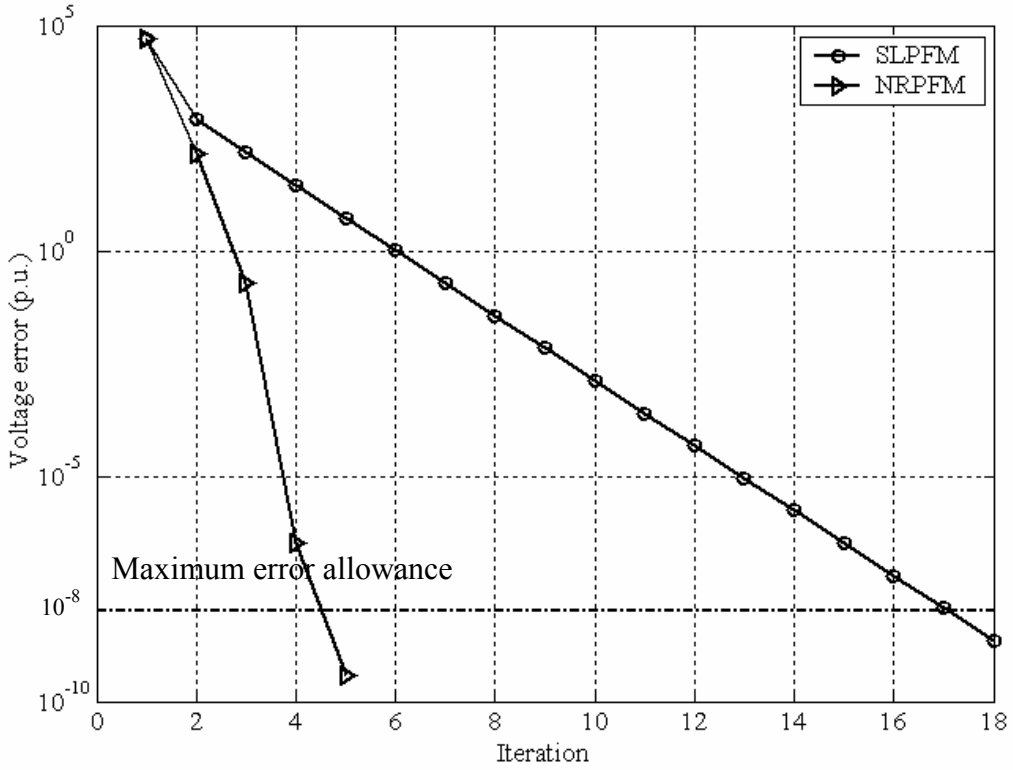


Fig. 3.21: Convergence for the double-track AT railway test system

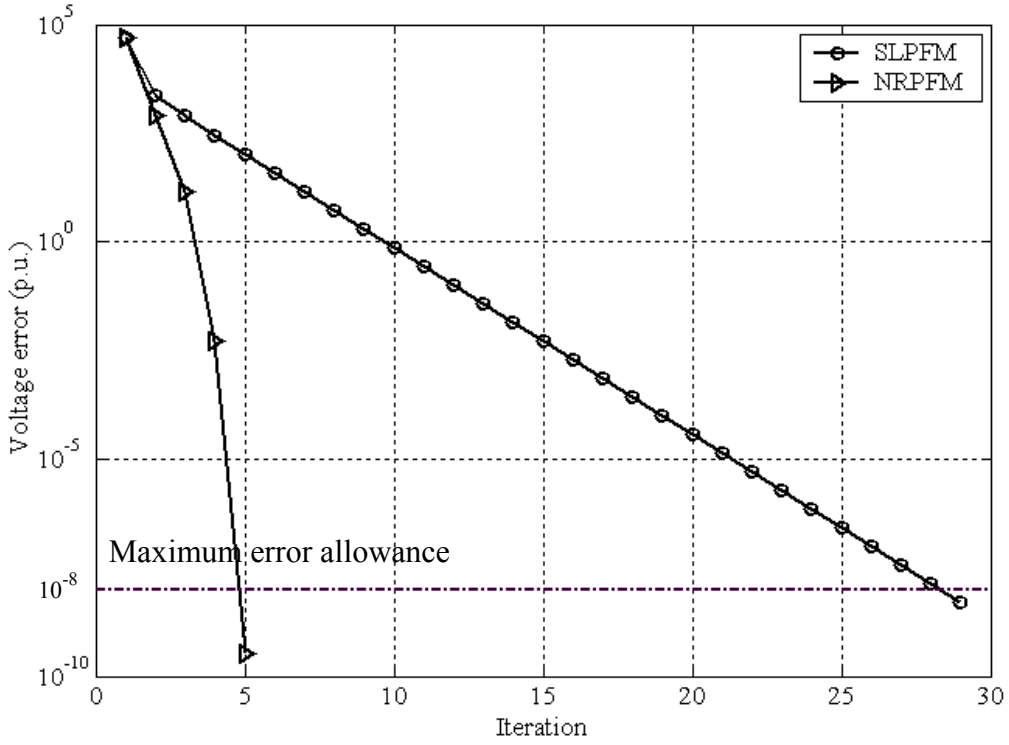


Fig. 3.22: Convergence for the quad-track AT railway test system

Table 3.9: Execution time and iteration used for the double-track AT railway system

Method	Total number of iteration used	Execution time (s)
SLPPFM	18	0.047
NRPFM	5	0.797

Table 3.10: Execution time and iteration used for the quad-track AT railway system

Method	Total number of iteration used	Execution time (s)
SLPFM	29	0.156
NRPFM	5	2.688

For the double-track system, 18 iterations and 0.047 s were consumed for obtaining the solution by the SLPFM. In contrast, the NRPFM used only 5 iterations but took the considerable amount of 0.797 s to execute. For the quad-track system, the solution obtained by the SLPFM converged within 29 iterations and cost 0.156 s of execution time, while the figures for the NRPFM were 5 iterations and 2.688 s, respectively. The MATLAB codes used for all the tests in this chapter can be found in Appendix G.1.

3.6.3 Discussions for AC railway power-flow solution techniques

The NRPFM has quadratic convergence. Only a few iterations, 3–5, is required to obtain the solution. However, the execution time is comparatively high. Also, the updating process of the NR method contains numerically heavy calculations such as power mismatch and Jacobian matrix formulae as described in Section 3.3.2 and 3.3.3 for the single- and bi-phase AC railway systems, respectively. Moreover, the size of the Jacobian matrix is 2N×2N, where N is the total number of buses; it is four times larger than the size of that used by the SLPFM. The number of operations required to obtain a solution of a linear system is the order of N^3 [48]. Thus, the number of operations required by the NRPFM is 8 times larger than that of the SLPFM. However, the SLPFM involves a complex linear system that requires four real multiplies and two real additions for a single complex multiplication. This indicates that if the required iteration and the calculation process within the iteration for both methods were the same, the NRPFM would spend a slightly shorter execution time than that consumed by the SLPFM. In

fact, although the required iteration for the SLPFM is larger than that of the NRPFM, the complicated formulae and the substantial memory updated during the iterative process cause the slower execution time of the NRPFM.

There is another significant factor that causes slow computation. It is the complexity of the updating formula, see equations 3.12 – 3.19 for the single-phase system and equations 3.44 – 3.65 for the bi-phase system. The formulae used by the SLPFM to update the bus current vector is noticeably simpler and is only an N×1 complex matrix compared with the huge 2N×2N Jacobian matrix. Table 3.11 and 3.12 summarise the size and total number of elements updated for the corresponding matrices.

Table 3.11: Matrix size and total number of elements for all the test cases

Test case			Matrix size or total number of elements				
			[**Y**]	[J]	[B_1], [B_2]	[**J**]	[ΔP],[ΔQ]
Single-phase tests system	24	SLPFM	24×24	-	-	24×1	-
		NRPFM	24×24	48×48	-	-	2×(24×1)
		FDCPFM	24×24	-	2×(24×24)	-	2×(24×1)
		Total Elements	576	2304	1152	24	48
		Total Updated Elements	0	2304	1152	24	48
	57	SLPFM	57×57	-	-	57×1	-
		NRPFM	57×57	114×114	-	-	2×(57×1)
		FDCPFM	57×57	-	2×(57×57)	-	2×(57×1)
		Total Elements	3249	12996	6498	57	114
		Total Updated Elements	0	12996	6498	57	114
	10	SLPFM	12×12	-	-	12×1	-
		NRPFM	12×12	24×24	-	-	2×(12×1)
		FDCPFM	12×12	-	2×(12×12)	-	2×(12×1)
		Total Elements	144	576	288	12	24
		Total Updated Elements	0	576	288	12	24
Bi-phase test system	D	SLPFM	42×42	-	-	42×1	-
		NRPFM	42×42	84×84	-	-	2×(42×1)
		Total Elements	1764	7056	-	42	84
		Total Updated Elements	0	7056	-	42	84
	Q	SLPFM	78×78	-	-	78×1	-
		NRPFM	78×78	156×156	-	-	2×(78×1)
		Total Elements	6084	24336	-	78	156
		Total Updated Elements	0	24336	-	78	156

D and Q denotes the double- and quad-track test systems, respectively.

Table 3.12: Summaries for the total number of elements updated for all the test cases

Test case		SLPFM	NRPFM	FDCPFM	μ_1	μ_2
1. Single-phase test system	IEEE 24-bus	24	2352	1200	98.0	1.96
	IEEE 57-bus	57	13110	6612	230.0	1.98
	10-train	12	600	312	50.0	1.92
2. Bi-phase test system	Double-track	42	7140	-	170.0	-
	Quad-track	78	24492	-	314.0	-

Where $\mu_1 = \dfrac{\text{Total number of elements updated for the NRPFM}}{\text{Total number of elements updated for the SLPFM}}$

$$\mu_2 = \frac{\text{Total number of elements updated for the NRPFM}}{\text{Total number of elements updated for the FDCPFM}}$$

The total number of elements updated at each iteration is a key indicator that strongly confirms the effectiveness of the SLPFM. During the iterative process if the ratio between the total number of elements updated for the SLPFM or the FDCPFM and the NRPFM $(\mu_1 \text{ or } \mu_2)$ is relatively large, the overall execution time tends to be a lot faster than that of the NRPFM. By using the FDCPFM, μ_2 is about $1.92 - 1.98$ for all test systems, so it could be said that the execution time obtained by using the FDCPFM might be approximately twice as fast as that of the NRPFM. The simulation results show that the FDCPFM is faster by a factor of 1.72, 1.33 and 1.38 for the respective three test systems of the single-phase case. The SLPFM is very much different in the algorithm, data type, etc. Each updated element involves complex number computation, therefore this can lead to reduction of the computational speed ratio. But the SLPFM uses simpler updating formula as mentioned previously. It is expected that the execution time consumed for solving the power flow problem will be greatly reduced by using the proposed method compared with the NRPFM and FDCPFM.

3.7 Summary

In this chapter, solution methods of the AC railway power-flow problem have been investigated, developed and tested. From the algorithms and the updating formulae, the SLPFM

is the simplest. The memory analysis also shows that the SLPFM requires less memory space than that required by the others. As a result, the solution obtained by the SLPFM gives the fastest execution time even though it only has linear convergence. Thus, the SLPFM will be used as the main power-flow algorithm to develop AC railway system simulators later in this thesis.

PART II

AC Railway Power Flow Calculation
(Exploitation of Sparsity)

CHAPTER 4

EFFICIENT ALGORITHM FOR AC RAILWAY SYSTEM SIMULATION

4.1 Introduction

The heart of the power flow calculation is to solve a set of non-linear equations. Most power flow algorithms require successive approximation to simplify the problem complexity and therefore correct the most recent approximated solution until the solution converges. The most commonly used technique is to repeatedly solve a linear system, which is generated during the iterative process. In Chapter 3, efficient power flow algorithms for AC railway systems have been proposed. Most iterative processes require an embedded sub-routine to handle such a linear system; at each iteration at least one linear system equation has to be solved. Convergence, memory allocation and simplification for the power flow algorithms have already been discussed in the previous chapter. In this chapter, solution methods for the linear system will be carefully investigated and thus exploited to develop the most efficient AC railway system simulation in all aspects.

4.2 Methods of Solving a Linear System [47,48]

Consider a system of linear equations

$$\mathbf{Ax} = \mathbf{b} \tag{4.1}$$

where n is the number of unknowns, m is the number of equations, $\mathbf{A} \in \Re^{m \times n}$, $\mathbf{b} \in \Re^{m}$, $m \geq n$. If n = m, the number of equations is as many as the number of unknowns. Usually, there is a unique solution. However, it can fail to have a unique solution if one or more of the m equations is a linear combination of the others. It is said to be degenerate. In this case, **A** will be

a singular matrix. In this chapter, the linear systems resulting from railway modelling are assumed to be non-singular to ensure the existence of the solution.

Let **A** be the coefficient matrix of a non-singular linear system as follows.

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}, \ \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \tag{4.2}$$

Although a direct inversion of the matrix **A** can be used to solve this system properly, $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$, Gaussian elimination with back-substitution is more efficient. It is in between the full elimination of the Gauss-Jordan technique and triangular decomposition. The disadvantage of this technique, however, is in the right-hand-side of the equation; vector **b** must be known in advance. When solving a linear system using a given matrix **A** with different values of **b**, the elimination must be performed repeatedly. This causes inefficient computations. In contrast, a direct method based on matrix factorisation, known as an LU factorisation, is more efficient. This method factorises the matrix A into the product of two triangular matrices **L** and **U**, where **A** = **LU**. With these two matrices, the forward and backward substitutions will be applied to obtain the solution for whatever the vector **b** is. Thus, such a disadvantage is eliminated.

Suppose that **A** can be factorised into the product of **L** and **U** as follows.

$$\mathbf{Ax} = (\mathbf{LU})\mathbf{x} = \mathbf{L}(\mathbf{Ux}) = \mathbf{Ly} = \mathbf{b} \tag{4.3}$$

where

$$\mathbf{L} = \begin{bmatrix} \ell_{11} & 0 & \cdots & 0 \\ \ell_{21} & \ell_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \ell_{n1} & \ell_{n2} & \cdots & \ell_{nn} \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ 0 & u_{22} & \cdots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & u_{nn} \end{bmatrix} \tag{4.4}$$

To obtain the solution **x**, firstly **Ly** = **b** must be solved by the forward substitution. Secondly, the backward substitution is used to solve **Ux** = **y**, these two procedures are the heart of the LU factorisation technique for solving a linear system.

There are two common methods for performing the LU factorisation: i) Gaussian elimination and ii) Direct computation. The Gaussian elimination starts with forming an augmented matrix, [**A**|**I**] by inserting an identity matrix **I** of the same dimension of **A** into the right-hand-side. The elimination is then performed by row operations until the augmented matrix is reduced to the form of [**U**|**L**]. Alternatively, the direct computation consists of three main methods: i) Doolittle method, ii) Crout method and iii) Cholesky method. First, the Doolittle form is to assign 1 to all diagonal elements of the **L** matrix. The **L** and **U** matrices obtained by this method, however, are the same as those obtained by using the Gaussian elimination. The second form is the Crout. It places 1 on all diagonal elements of the **U** matrix instead. The third is the Cholesky form that gives the values of the corresponding diagonal elements of **L** and **U** equally. When the matrix **A** is a positive definite matrix, there is the great advantage to use such a factorisation. The Cholesky factorisation is in the form of $\mathbf{L} = \mathbf{U}^T$. As can be seen, this method can reduce the memory space for storing **L** and **U** by a factor of two.

As mentioned above, the matrix factorisation is not unique. Although the Cholesky's factorisation has a good characteristic and has been successfully applied to several applications including DC railway power network calculation [50], it is limited to solving a linear system of real numbers; not complex numbers. There is no strong evidence to confirm the success of using the Cholesky method for any complex linear system. In the AC railway power systems, a complex linear system is normally generated. The simple LU factorisation based on the Gaussian elimination is more stable and more reliable in this situation. In this thesis, any complex linear system generated during the iterative process will be solved by LU factorisation with Gaussian elimination. Briefly, the proposed method can be summarised, step-by-step, as follows.

- **LU factorisation by Gaussian Elimination**

Step 1: Set **U** =**A** and **L** as an n×n identity matrix

Step 2: For j = 1 to j = n, Do

For i = j+1 to i = n, Do

$$\ell_{ij} = \frac{u_{ij}}{u_{jj}}$$

$$u_{ij} = u_{ij} - \sum_{k=1}^{n} \ell_{ij} u_{jk}$$

End

End

Step 3: The factorisation is completed. **L** and **U** are successfully found.

With the advantage of the matrix factorisation, the solution **x** can be obtained by using the forward and backward substitutions consecutively. Assume that the matrix **A** is already factorised into **L** and **U** and the vector **b** is given. The forward substitution for solving **Ly** = **b** and the backward substitution for solving **Ux** = **y** are described below.

- **Forward substitution**

Step 1: Compute $y_1 = b_1$

Step 2: For i = 2 to i = n, Do

$$y_i = \frac{1}{\ell_{ii}} \left(b_i - \sum_{j=1}^{i-1} \ell_{ij} b_j \right)$$

End

Step 3: The forward substitution is completed. **y** is successfully solved.

- **Backward substitution**

Step 1: Compute $x_n = \frac{y_n}{u_{nn}}$

Step 2: For i = n-1 to i = 1, Do

$$x_i = \frac{1}{u_{ii}} \left(y_i - \sum_{j=i+1}^{n} u_{ij} x_j \right)$$

End

Step 3: The backward substitution is completed. **x** is successfully solved.

Note that this algorithm can be equally well applied to both real and complex linear systems.

Alternatively, a large, sparse, positive-definite linear system can be efficiently solved by some iterative methods. There are four commonly used iterative methods: i) Jacobi method, ii) GS method, iii) Successive over-relaxation method and iv) Conjugate Gradient method (CGM) [50]. In most relevant textbooks, the CGM is widely accepted and broadly used as the most powerful iterative method among them. It is strongly recommended for especially large, unstructured and positive-definite sparse linear systems [49–53]. Unfortunately, the CGM can be used for the real linear system only. Even though the complex system can be decomposed and then re-formed to a quadruple-sized real linear system, using the CGM is still not as effectively as applying the LU factorisation especially with some special structures, such as the band diagonal matrix, as will be explained in the next section.

4.3 Sparse Linear System and Solution Method

A sparse linear system is a system whose matrix contains only a relatively small number of non-zero elements. General methods of linear algebra for solving the set of equations or inverting the matrix are wasteful because they involve zero operands. Although applying the CGM can effectively solve large, unstructured and positive-definite sparse linear systems, it is limited to real systems only. Thus, it is not recommended for use in many previous works [48,52,54]. With some special structured sparse matrices like tri-diagonal, band diagonal, block diagonal etc, the LU factorisation method is more powerful but wise handling of zero elements is required. In typical AC railways, one feeding section has only one feeding point at the substation. One or more overhead catenary feeding circuits are directly connected to this point

and then led towards the next station. One feeding circuit is for one track. All the circuits are in parallel and are all tied together at the MPTSC. With this special feeding configuration, the development of a bus-ordering algorithm to arrange all non-zero elements in such a way as to result in a band diagonal matrix is quite simple.

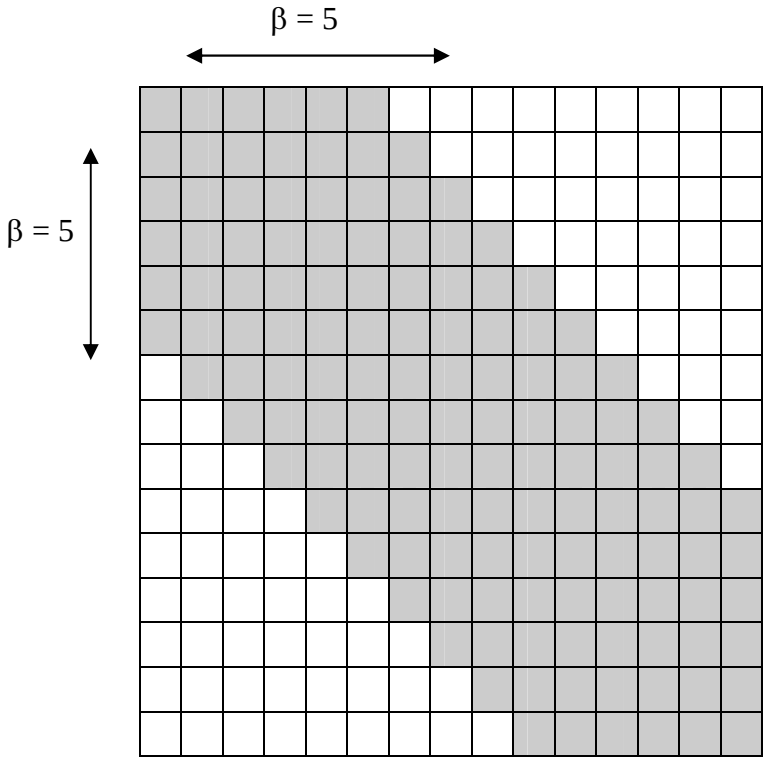


Fig. 4.1: Typical 15×15 band diagonal matrix with bandwidth 5

Assume that **A** is an n×n band diagonal matrix with bandwidth β as shown in Fig. 4.1 for the special case of the 15×15 band diagonal matrix with bandwidth 5. Clearly, all zero elements outside the band can be ignored. The execution time resulting from the matrix factorisation, forward and backward substitutions of the band diagonal matrix will be greatly reduced [52] if the direct method described in Section 4.2 is modified carefully only to operate on all elements inside the band as follows.

- **LU factorisation for a band diagonal matrix with bandwidth β**

Step 1: Set **U** =**A** and **L** as an n×n identity matrix

Step 2: For j = 1 to j = n, Do

For i = j+1 to i = j+β, Do *// modified*

$$\ell_{ij} = \frac{u_{ij}}{u_{jj}}$$

$$u_{ij} = u_{ij} - \sum_{k=j-\beta}^{j+\beta} \ell_{ij} u_{jk} \qquad \text{// modified}$$

End

End

Step 3: The factorisation is completed. **L** and **U** are successfully found.

- **Forward substitution**

Step 1: Compute $y_1 = b_1$

Step 2: For i = 2 to i = n, Do

$$y_i = \frac{1}{\ell_{ii}} \left(b_i - \sum_{j=i-\beta}^{i-1} \ell_{ij} b_j \right) \qquad \text{// modified}$$

End

Step 3: The forward substitution is completed. **y** is successfully solved.

- **Backward substitution**

Step 1: Compute $x_n = \frac{y_n}{u_{nn}}$

Step 2: For i = n-1 to i = 1, Do

$$x_i = \frac{1}{u_{ii}} \left(y_i - \sum_{j=i+1}^{i+\beta} u_{ij} x_j \right) \qquad \text{// modified}$$

End

Step 3: The backward substitution is completed. **x** is successfully solved.

4.4 Exploitation of Sparsity for AC Railway Simulation

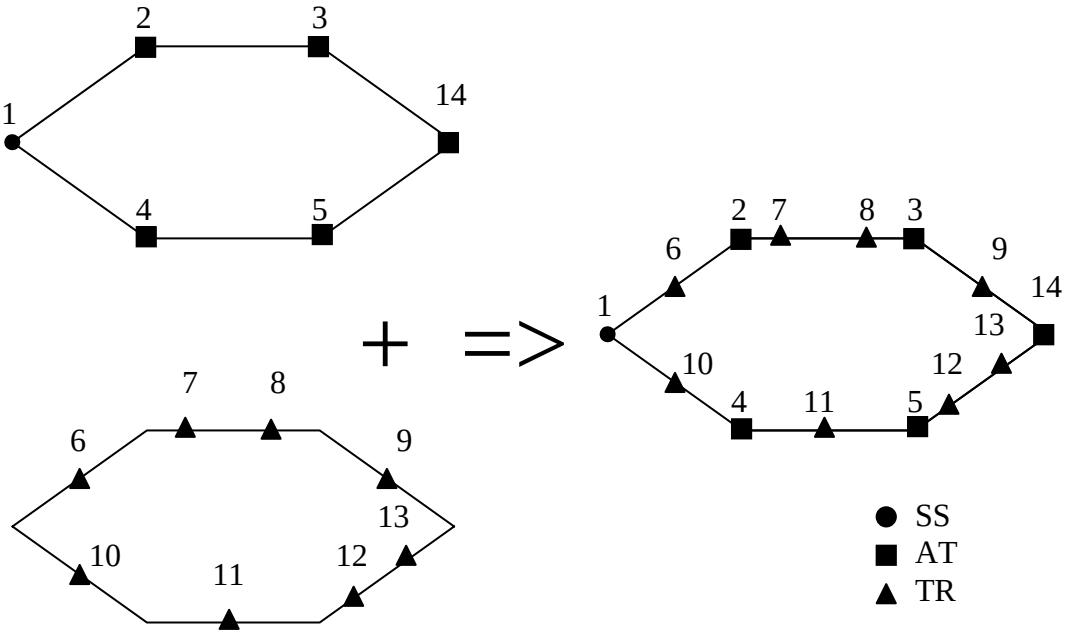


Fig. 4.2: SBO algorithm leading to a non-banded sparse matrix

The system equation, [**Y**][**V**] = [**I**], after the linearisation referred to above is both linear and sparse but needs repetitive solution within the iterative process. With only a few non-zero elements stored in the matrix, appropriately organising the bus numbers can effectively pack all those elements together to form some type of sparse matrix, e.g. tridiagonal, band, block diagonal, singly/doubly bordered block/ band diagonal, or other [48]. Using a normal matrix computation for solving a sparse linear system involves mostly zero operands and thus wastes execution time. Although there are various kinds of sparsity, some of which are very powerful for reducing solution times, such as the doubly bordered block diagonal matrix [48-50], it may be necessary to perform pre-conditioning operations which form the matrix into the optimal order. This optimum ordering sub-routine adds complication and also costs time. In this thesis, only the band matrix approach with a certain diagonal bandwidth [54] is used. This structure can be generated by a quite simple bus-numbering algorithm, which can be heuristically based. The following sections describe i) a simple bus ordering leading to a non-banded sparse matrix,

ii) a banded bus ordering, iii) a banded matrix storage and iv) a whole system simulator for AC railways.

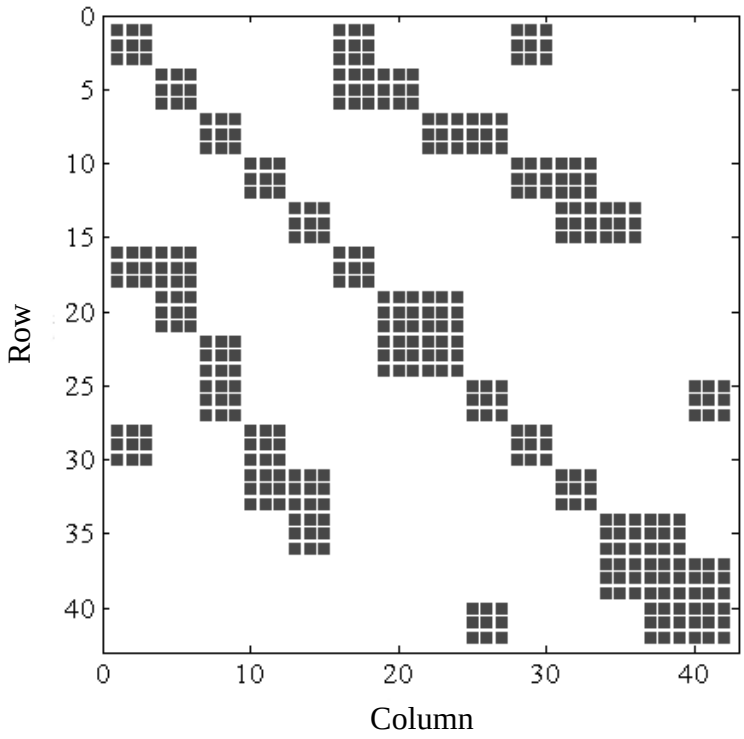


Fig. 4.3: Bus admittance matrix for the example given in Fig. 4.2

4.4.1 Simple bus-ordering algorithm (SBO)

In general, bus admittance matrices formulated during a simulation are naturally sparse, but their non-zero elements can appear in any position. Furthermore, when not well organised, the program execution to manipulate these separate non-zeros is still expensive. To represent an unorganised sparse matrix for comparison, Simple Bus Ordering (SBO) is now defined. First, the number 1 is assigned to the substation bus and then all fixed nodes are numbered, starting with the up-track from the AT node nearest the substation and proceeding to the farthest AT node on the same track. Next, all the AT nodes on the down-track are assigned numbers in sequential order, similarly to those on the up-track. Afterwards, numbering of the moving nodes is carried out starting with the nearest node to the substation on the up-track until reaching the farthest node on the same track. The same rule is then applied to the down-track. Finally, the

last bus number is given to the MPTSC. The above description is summarised diagrammatically in Fig. 4.2 for a simple two-track railway between a substation and MPTSC, while Fig. 4.3 shows the corresponding bus admittance matrix.

4.4.2 Banded bus-ordering algorithm (BBO)

When all the positions of the trains and transformers (if any) are known, a banded bus ordering (BBO) sub-routine can be performed. Suppose there is a total of M tracks. Heuristically, a bus ordering that can simply organise an AC-railway representative matrix into a band diagonal matrix is given by the following steps.

1. Set counter i = 1, j = 1 and assign the bus number 1 to the substation bus
2. For k = 1 to M, do

   2.1 The j^{th} bus on the k^{th} track is numbered as i+1

   2.2 Update counter i, i = i+1
3. Update counter j, repeat Step 2 if all the buses are not entirely numbered
4. The last bus number, i+1, is given to the MPTSC bus and terminates the process

An example of this algorithm is given in Fig. 4.4. The bus admittance matrix of the example given in Fig. 4.4 after applying the BBO is shown in Fig. 4.5.

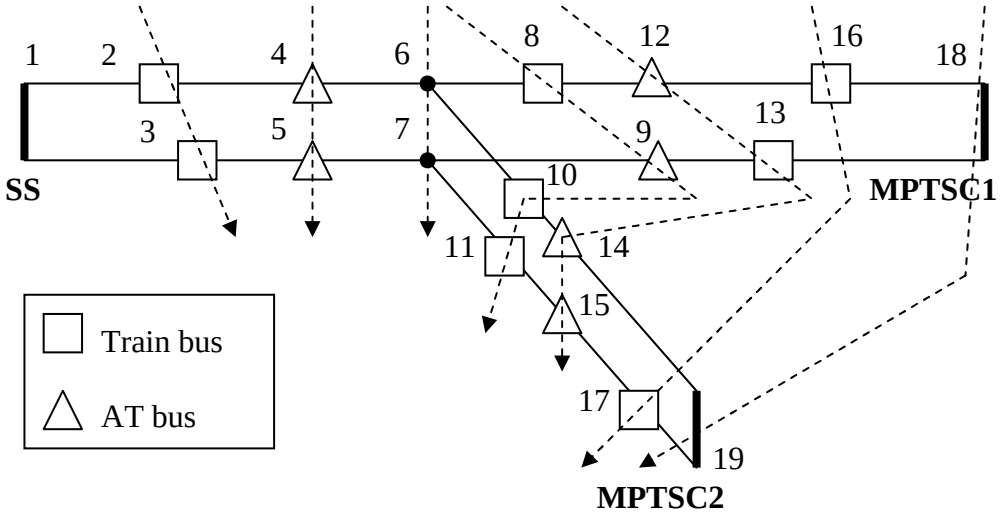


Fig. 4.4: BBO leading to a banded matrix

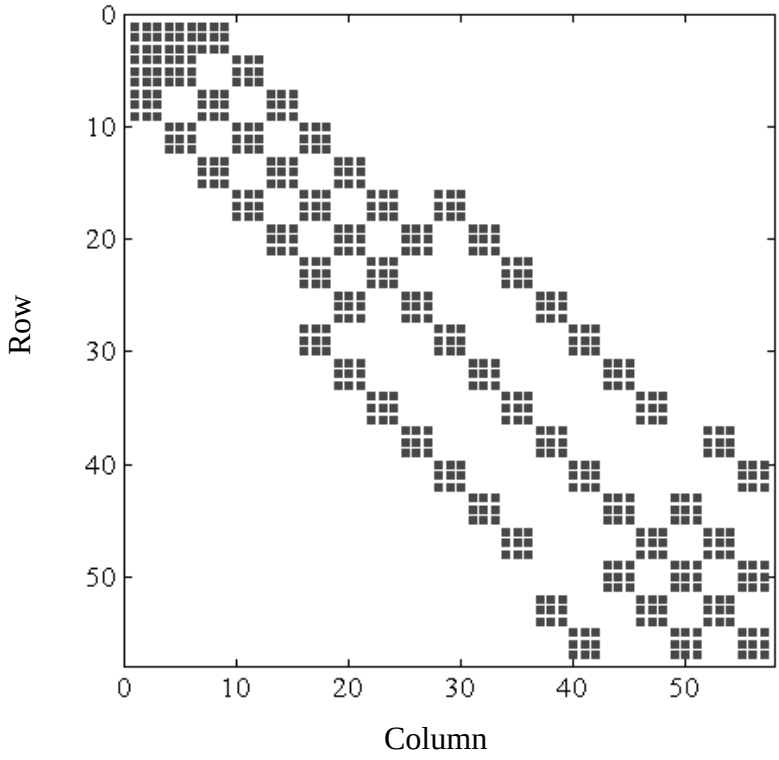


Fig. 4.5: Bus admittance matrix for the example given in Fig. 4.4

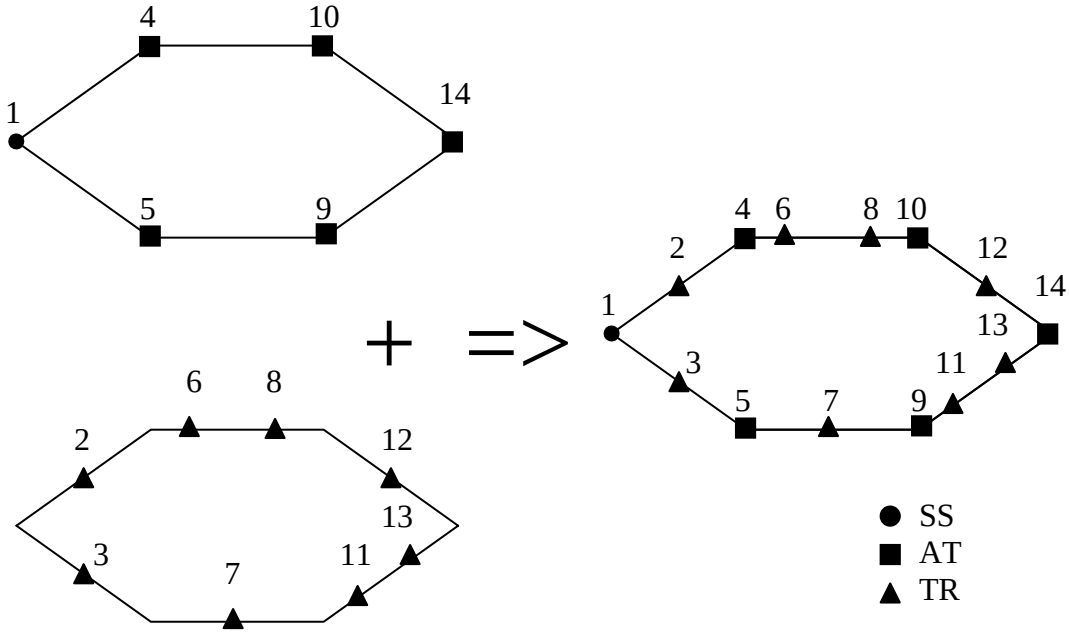


Fig. 4.6: BBO algorithm leading to a banded matrix for the example shown in Fig. 4.2

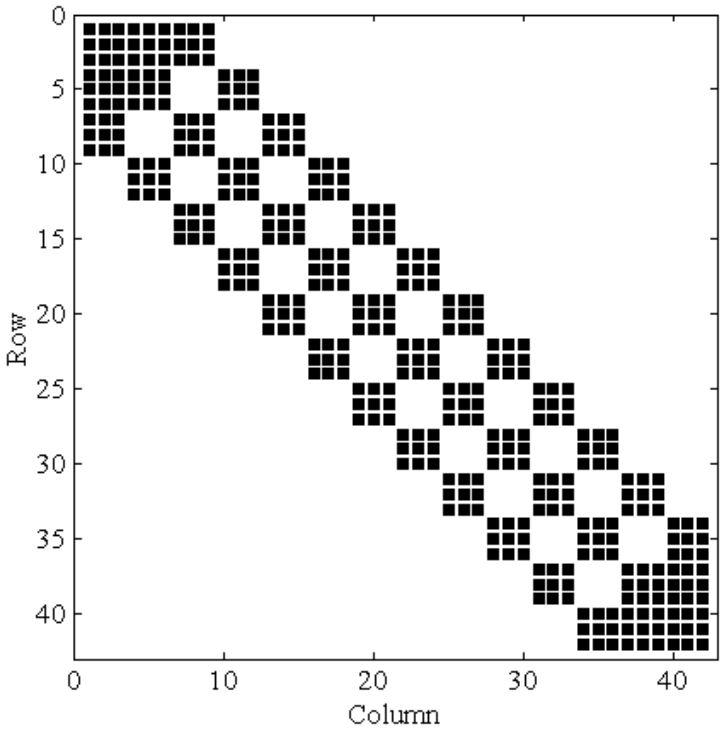


Fig. 4.7: Bus admittance matrix for the example given in Fig. 4.6

For a fair comparison, all the bus numbers of the example given in Fig. 4.2 are now re-ordered by using the BBO. Fig. 4.6 shows the bus numbers after re-ordering while Fig. 4.7 presents the sparsity of the bus admittance matrix. It can be seen that all non-zero elements are re-located within the diagonal band, and thus an efficient banded matrix algorithm to perform the computation can be applied.

4.4.3 Memory storage requirement for a band diagonal matrix

The most compact format in which to store an N-by-N matrix is to record only its non-zero elements. However, to avoid the need for a non-zero-searching algorithm, the method described here uses a storage arrangement based on recording all the elements, including zeros, that lie within the diagonal band in an N-by-(2×β+1) matrix as shown in Fig. 4.8. Although this storage method is not as efficient as the method that stores only non-zero elements, it is simple to program, and when the matrix size is very large while the band is comparatively small, the difference is insignificant. The total number of elements inside the band is $n\times(2\beta+1) - \beta(\beta+1)$.

For example, given a 100×100 matrix with $\beta = 15$, the space required by the proposed storage is 3100 elements, whilst the total number of elements inside the band is 2860. This saves about 7.7%.

x_{11}	x_{12}	x_{13}	0	0	0	0
x_{21}	x_{22}	x_{23}	x_{33}	0	0	0
x_{31}	x_{32}	x_{33}	x_{34}	x_{35}	0	0
0	x_{42}	x_{43}	x_{44}	x_{45}	x_{46}	0
0	0	x_{53}	x_{54}	x_{55}	x_{56}	x_{57}
0	0	0	x_{64}	x_{65}	x_{66}	x_{67}
0	0	0	0	x_{75}	x_{76}	x_{77}

a) Full storage

#	#	x_{11}	x_{12}	x_{13}
#	x_{21}	x_{22}	x_{23}	x_{33}
x_{31}	x_{32}	x_{33}	x_{34}	x_{35}
x_{42}	x_{43}	x_{44}	x_{45}	x_{46}
x_{53}	x_{54}	x_{55}	x_{56}	x_{57}
x_{64}	x_{65}	x_{66}	x_{67}	#
x_{75}	x_{76}	x_{77}	#	#

b) Band diagonal storage

represents an element that is not referenced by the program

Fig. 4.8: Band diagonal matrix storage

4.4.4 Whole system simulator

The simulator used in this thesis to test the network solution method is divided into three main modules: i) train movement simulation, ii) network capture, and iii) power network simulation. These can be briefly described as follows.

- Train movement simulation: This module performs the dynamic movement calculations of trains to update speed, position, tractive effort, power consumption, etc [55-57].

- Network capture: To solve the power supply network during simulated train running, the electrical nodes must be located. For a typical AC railway power feeding system, the substation terminal, MPTSC and transformers (if any) are in fixed positions. Thus, when the train positions have all been updated, the power network equations have to be re-formed. Locating all the trains, numbering all the buses and formulating the interconnection equations for every pair of connected nodes are thus the essential tasks of this sub-routine.

- Power network solution: This module, as described previously, formulates the set of power network equations and then solves for the voltages. Linearisation and sparsity are both applied to maximise the overall computing efficiency.

Note that all the three modules of the system simulator will be explained in more detail in Chapter 6.

4.5 Simulation Results

The first part of this section is to examine the BBO algorithm used to transform the system matrix of various AC railway power-feeding shapes that have many branches leading to different directions into a band diagonal form. Five different feeding figures are given in Fig. 4.9, 4.11, 4.13, 4.15 and 4.17. Their corresponding bus admittance matrices are shown in Fig. 4.10, 4.12, 4.14, 4.16 and 4.18, respectively.

- **Tree-shape AC railway feeding system**

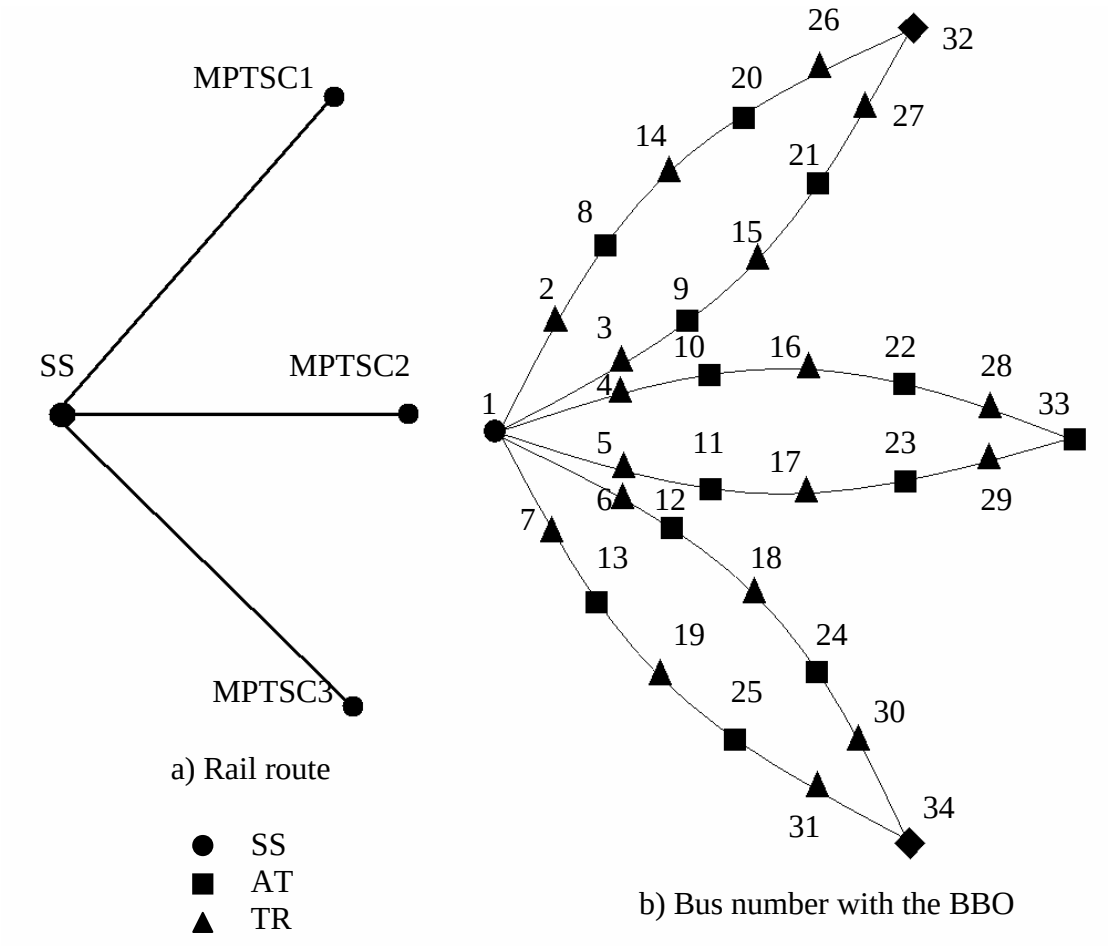


Fig. 4.9: Tree-shape AC railway power feeding system

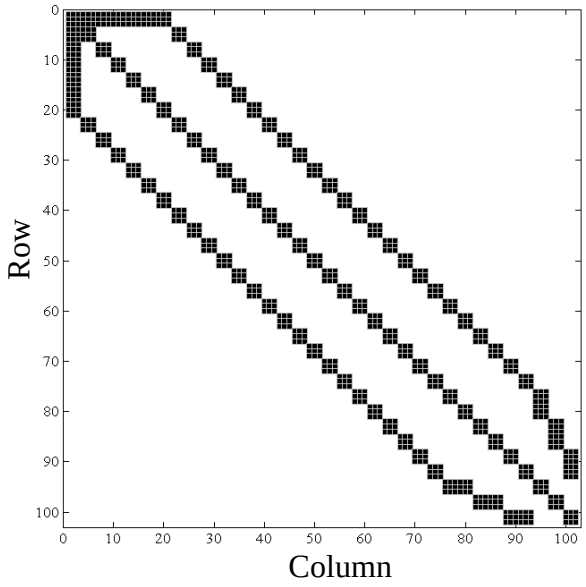


Fig. 4.10: Bus admittance matrix of the tree-shape feeding system

- **Parallel AC railway feeding system**

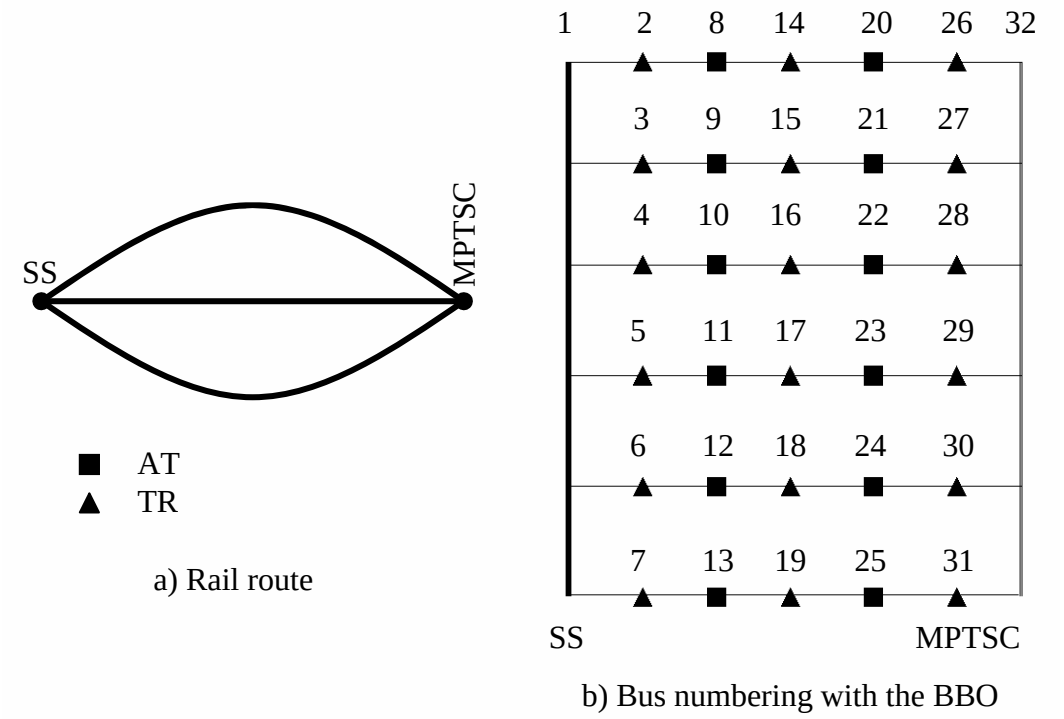


Fig. 4.11: Parallel AC railway feeding system with 6 tracks

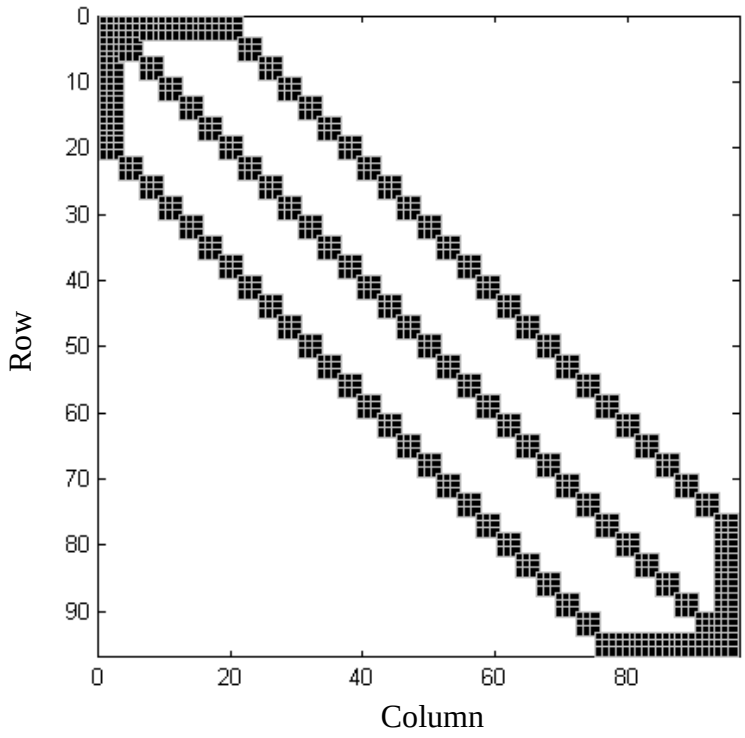
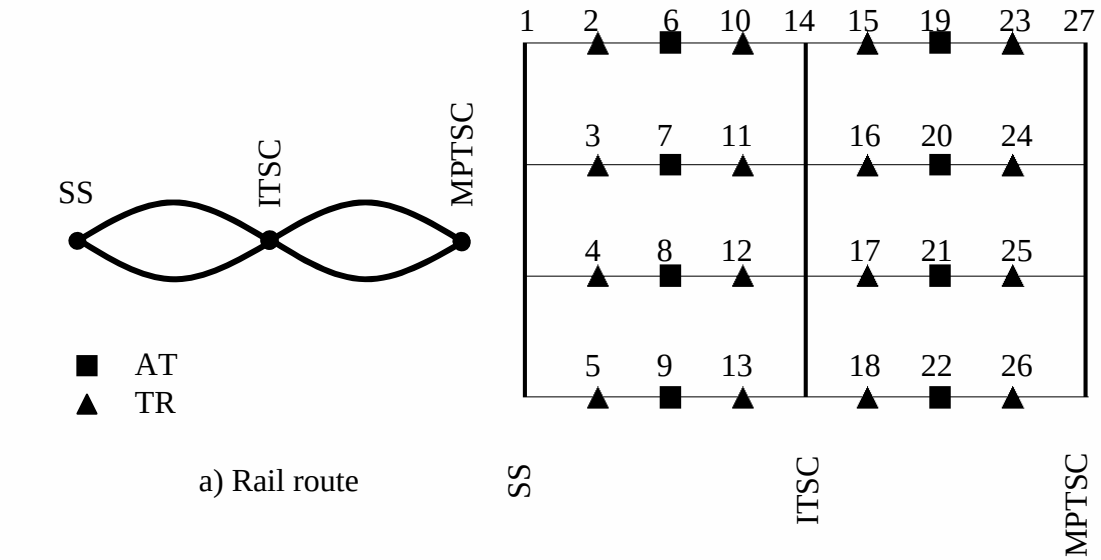


Fig. 4.12: Bus admittance matrix for the parallel AC railway feeding system

- **Parallel AC railway feeding system with the ITSC**



a) Rail route

b) Bus numbering with the BBO

Fig. 4.13: Parallel AC railway feeding system with the ITSC

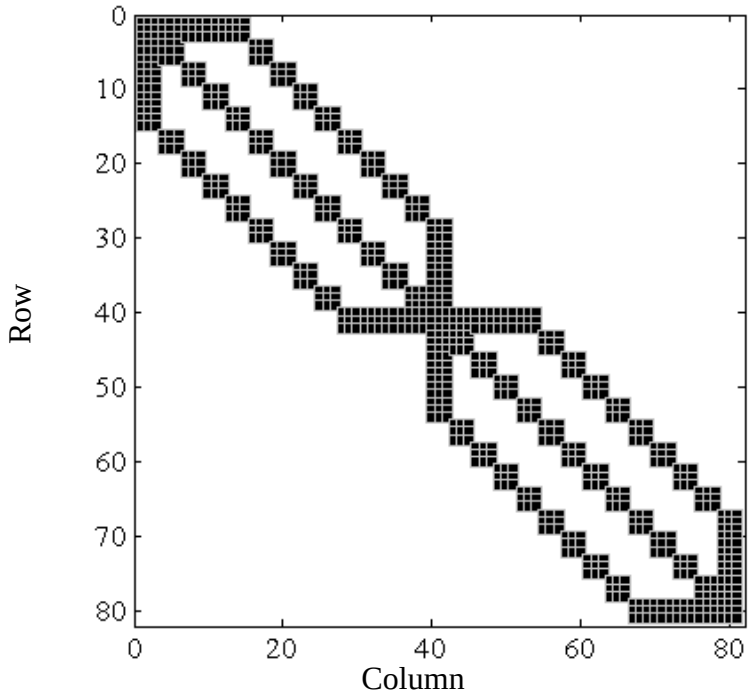
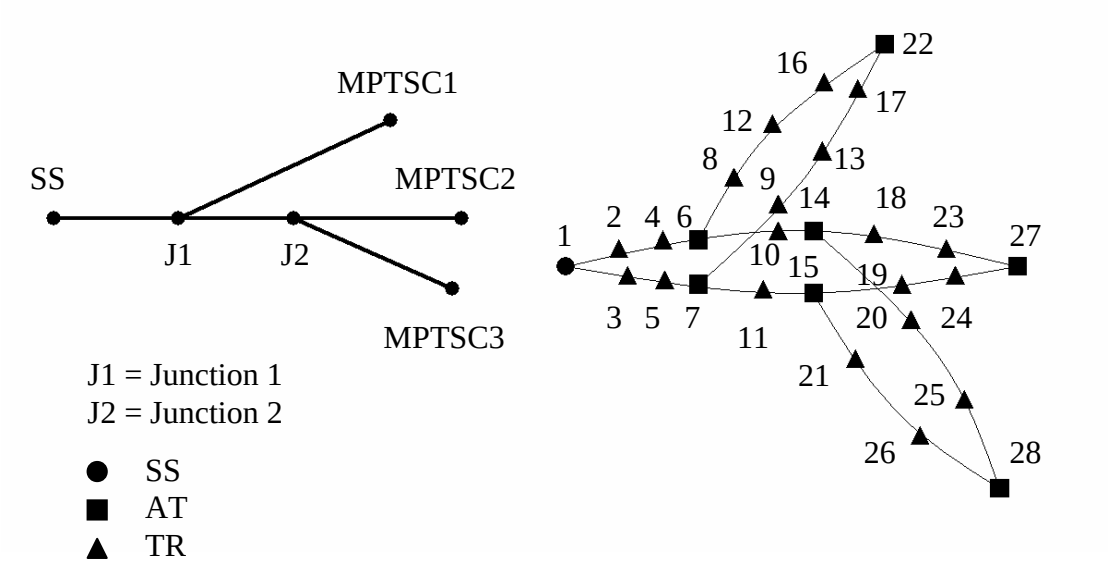


Fig. 4.14: Bus admittance matrix for the parallel AC railway feeding system (ITSC)

- **Multi-branch AC railway feeding system**



a) Rail route b) Bus numbering with the BBO

Fig. 4.15: Multi-branch AC railway feeding system

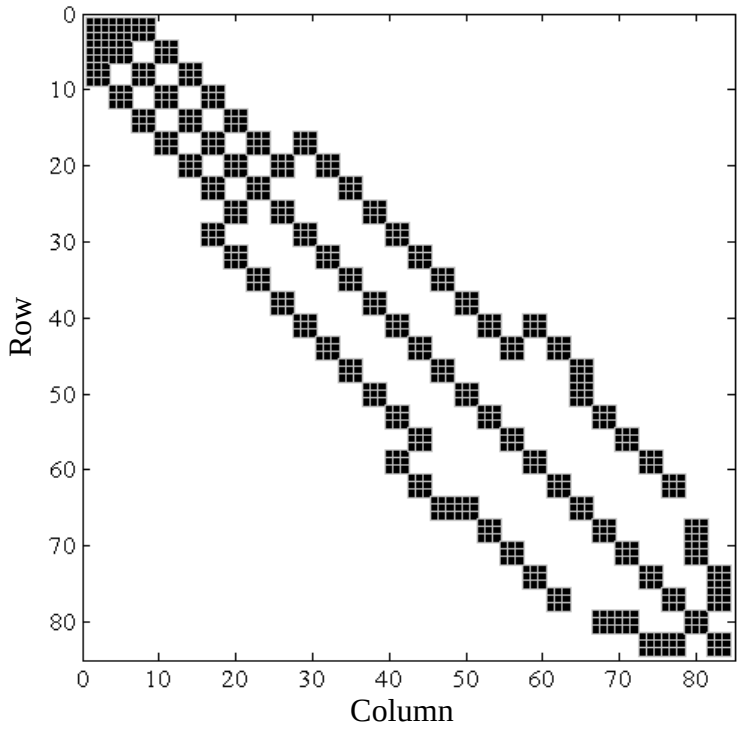
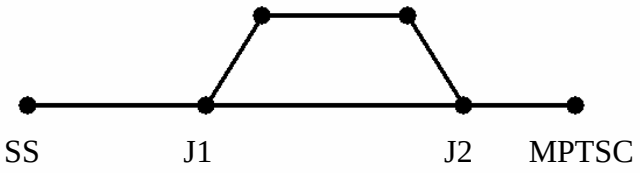


Fig. 4.16: Bus admittance matrix for the multi-branch AC railway feeding system

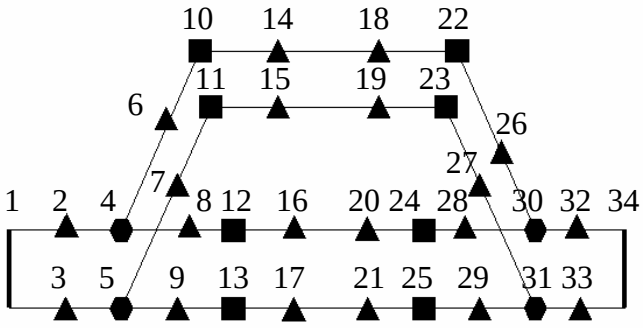
- **Loop AC railway feeding system**



a) Rail route

J1 = Junction 1
J2 = Junction 2

AT
TR
Junction



b) Bus numbering with the BBO

Fig.4.17: Loop AC railway feeding system

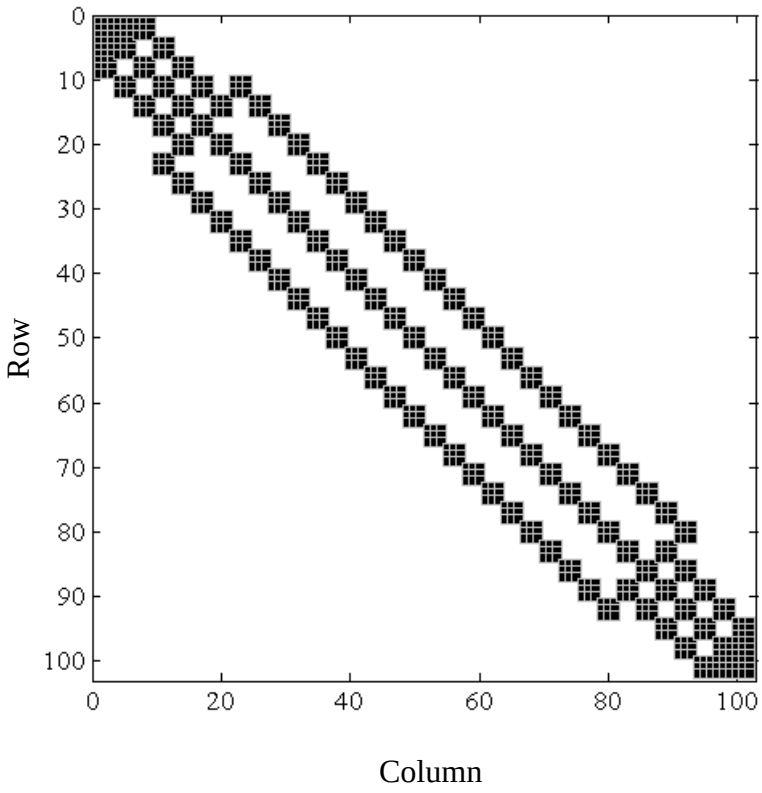


Fig. 4.18: Bus admittance matrix for the loop AC railway feeding system

The results show the effectiveness of the BBO algorithm to transform the system into the band diagonal system for various AC railway power-feeding shapes. Furthermore, the bus admittance matrices resulting from this algorithm are not only diagonally banded but also diagonally dominant. This confirms that the LU factorisation based on the Gaussian elimination for solving linear systems can be profitably used for the AC railway power feeding systems.

The second part is to apply the sparsity technique for the AC railway system simulation. The computer-based real-time AC railway system simulation with 0.5-s update has been developed. A double-track AT railway power feeding system as shown in Fig. 4.19 is used for the tests. The system has 10-minute headway of train operation. Full information for this test system, such as train tractive-effort models, train characteristics, power feeding parameters, power substation parameters, track gradient profile etc, can be found in Appendix E.3.

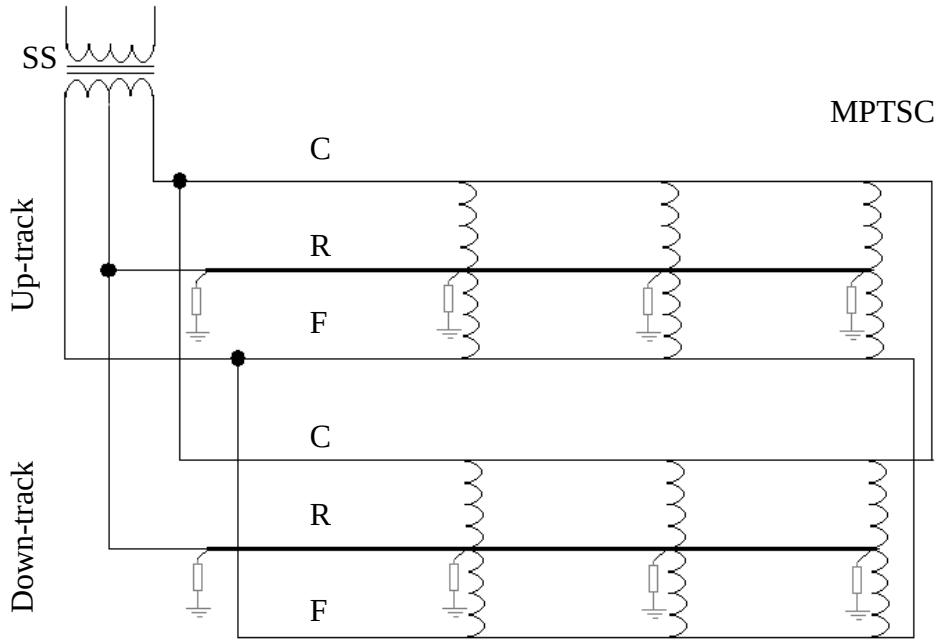


Fig. 4.19: Double-track AT railway power feeding system

The simulation is performed representing two hours of operation. Two main test cases: i) Nominal start and ii) Recursive start, are examined to evaluate the performance of the band diagonal system. The nominal start means initialising the voltage vector with the nominal voltage, 25 kV, at every time step, while the recursive start uses the voltage solution obtained from the previous time step as the initial voltage. Each case also consists of three sub-cases. The first is to simulate the test system by applying the SBO with the full matrix computation (FMC). The second sub-case is to use the BBO together with the band matrix computation (BMC) given in Section 4.3.

From the computer simulation, the train and power supply key variables are calculated during a two-hour period of service. Train position versus time curves and the train trajectories for the first trains on both tracks are given in Fig. 4.20 and 4.21, respectively. Fig. 4.22 shows the catenary voltage, rail voltage and apparent power drawn from the substation. In addition, train power, catenary and rail voltages for the selected train are shown in Fig. 4.23.

The simulation programs were written in C++. It consists of three modules as described in Section 4.4.4. The tests were performed on three different computers: i) AMD Athlon, 1.11 GHz, 256 MB-SDRAM, ii) Pentium IV, 1.7 GHz, 512 MB-SDRAM and iii) Pentium II, 133 MHz, 32 MB-SDRAM. Programming codes for this simulation can be found in Appendix G.2.

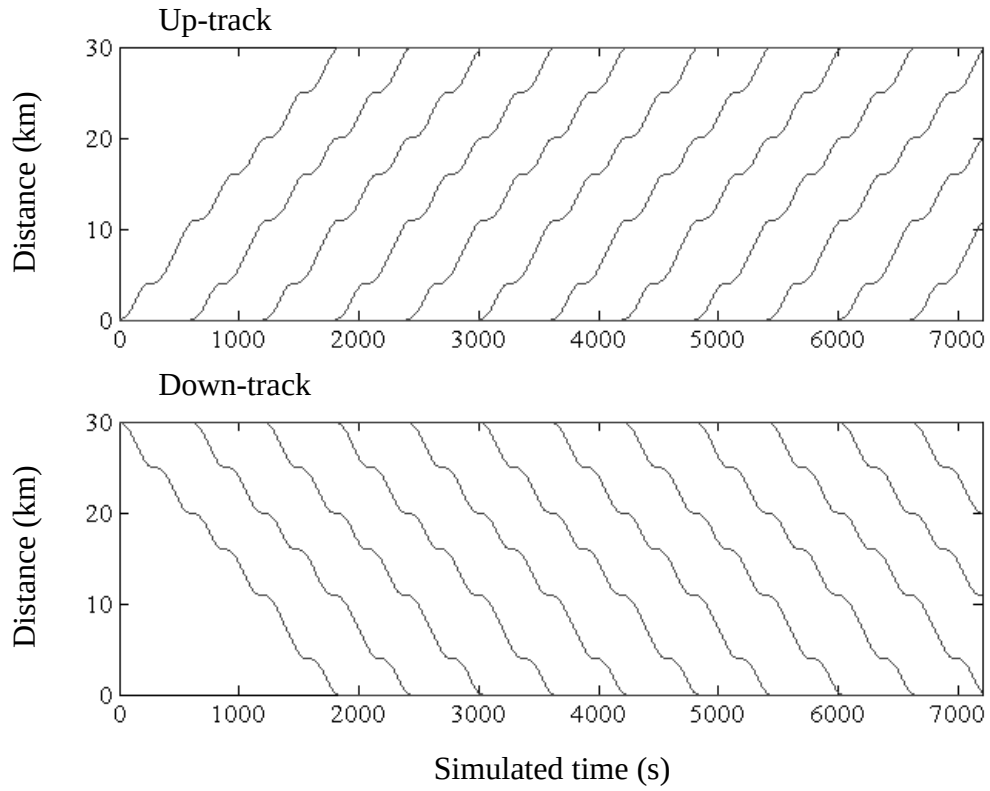


Fig. 4.20: Distance – time curves for two hours of operation

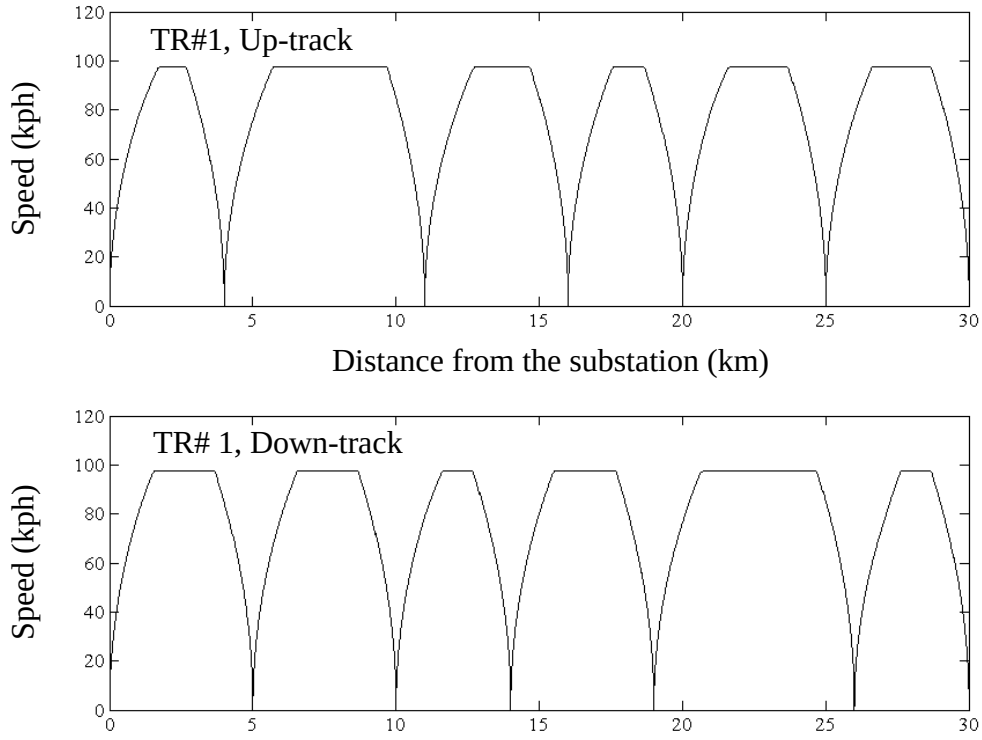


Fig. 4.21: Train speed-time trajectories for the first six trains on the up-track

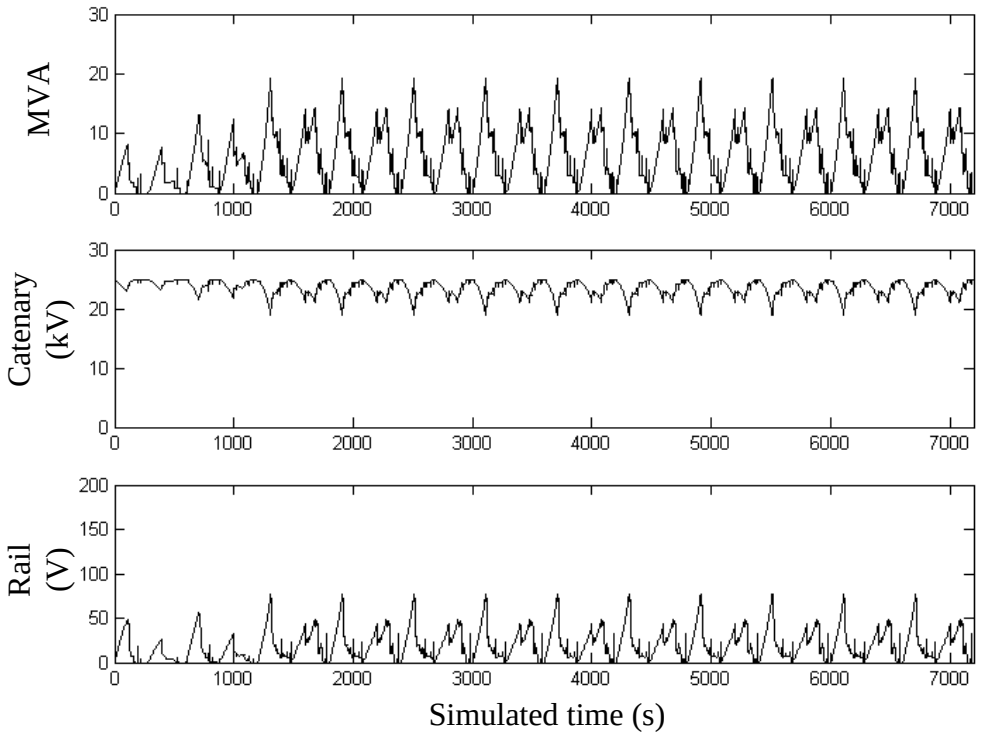


Fig. 4.22: Substation terminal conditions (Upper – Substation power, Middle – Catenary voltage, Lower – Rail voltage)

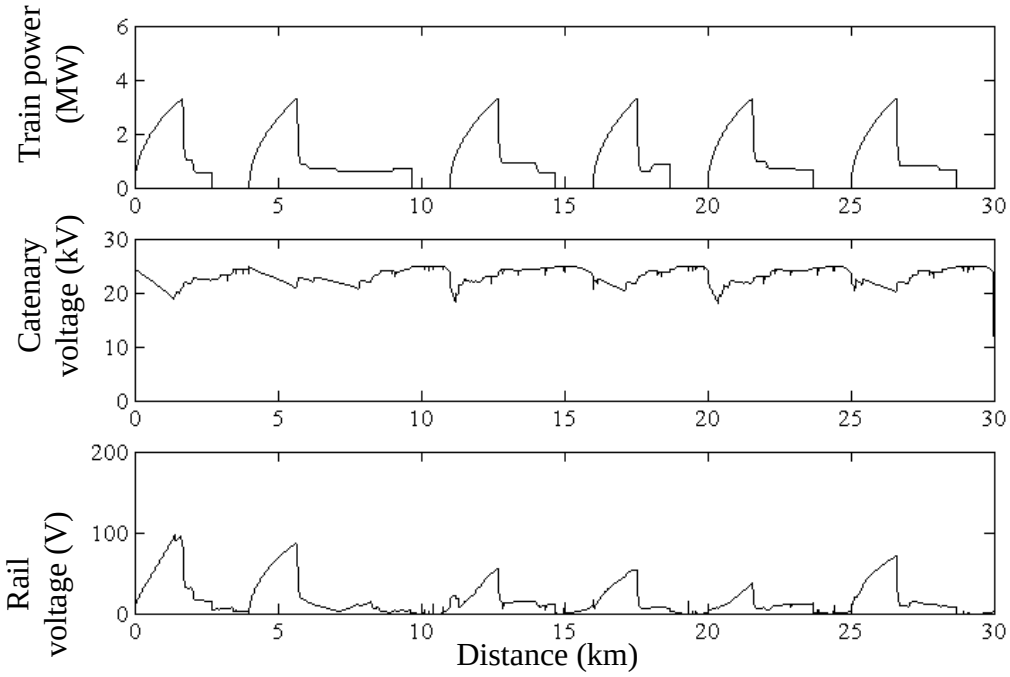


Fig. 4.23: Voltages of the fourth train on the up-track (Upper – Train power, Middle – Catenary voltage, Lower – Rail voltage)

Table 4.1: Execution times performed by the AMD Athlon computer

Test case	Nominal Start		Recursive Start	
	Execution time*	% Reduction	Execution time*	% Reduction
1. SBO + FMC	8.32	-	3.67	-
2. BBO + BMC	4.03	51.6	1.88	48.8

* Execution time in minute

Table 4.2: Execution times performed by the Pentium IV computer (P4)

Test case	Nominal Start		Recursive Start	
	Execution time*	% Reduction	Execution time*	% Reduction
1. SBO + FMC	9.72	-	4.38	-
2. BBO + BMC	4.57	53.0	2.20	49.8

* Execution time in minute

Table 4.3: Execution times performed by the Pentium II computer (P2)

Test case	Nominal Start		Recursive Start	
	Execution time*	% Reduction	Execution time*	% Reduction
1. SBO + FMC	91.63	-	42.95	-
2. BBO + BMC	53.80	41.3	27.15	36.8

* Execution time in minute

Table 4.1 presents the test results. Firstly, consider the effect of the two different computing methods on the execution time. With the nominal start, the time consumed by the SBO + FMC, say the base case, is 8.32, 9.72 and 91.63 minutes for the AMD, P4 and P2, respectively, while that of the BBO + BMC consumes 4.03, 4.57 and 53.80 minutes. This shows that for this test case scenario the Banded method can yield 51.6%, 53.0% and 41.3% reduction of the execution time for these three respective computers. With the recursive start, the SBO + FMC spends 3.67, 4.38 and 42.95 minutes for the AMD, P4 and P2, while 1.88, 2.20 and 27.15 minutes is for the BBO + BMC used in the AMD, P4 and P2, respectively. This can reduce the execution time by 48.8%, 49.8% and 36.8%.

Secondly, consider the effect of initialising conditions on the execution time consumed. The first comparison is the use of the SBO + FMC. With the recursive start, the execution time is only 3.67, 4.38 and 42.95 minutes, i.e. 55.9%, 54.9% and 53.1% reductions for the AMD, P4 and P2 when compared to their nominal start. For the BBO + FMC cases, the execution times consumed by the recursive start are 1.88, 2.20, 27.15 minutes (53.3%, 51.9% and 49.5% reduction) for these three respective computers. When applying the BBO + BMC together with the recursive start, the execution times are greatly reduced from 8.32 minutes to 1.88 minutes (77.4%) for the AMD computer, from 9.72 minutes to 2.20 minutes (77.4%) for the P4 computer and from 91.63 minutes to 27.15 minutes (70.4%) for the P2 computer.

Furthermore, to confirm the correctness of the power system solution, the current distribution in a double-track AT power feeding section with a single train located at 26.5 km from the substation is shown in Fig. 4.24. Also the voltage at each bus, the cumulative earth-leakage current at each bus and the train current are presented in Table 4.4.

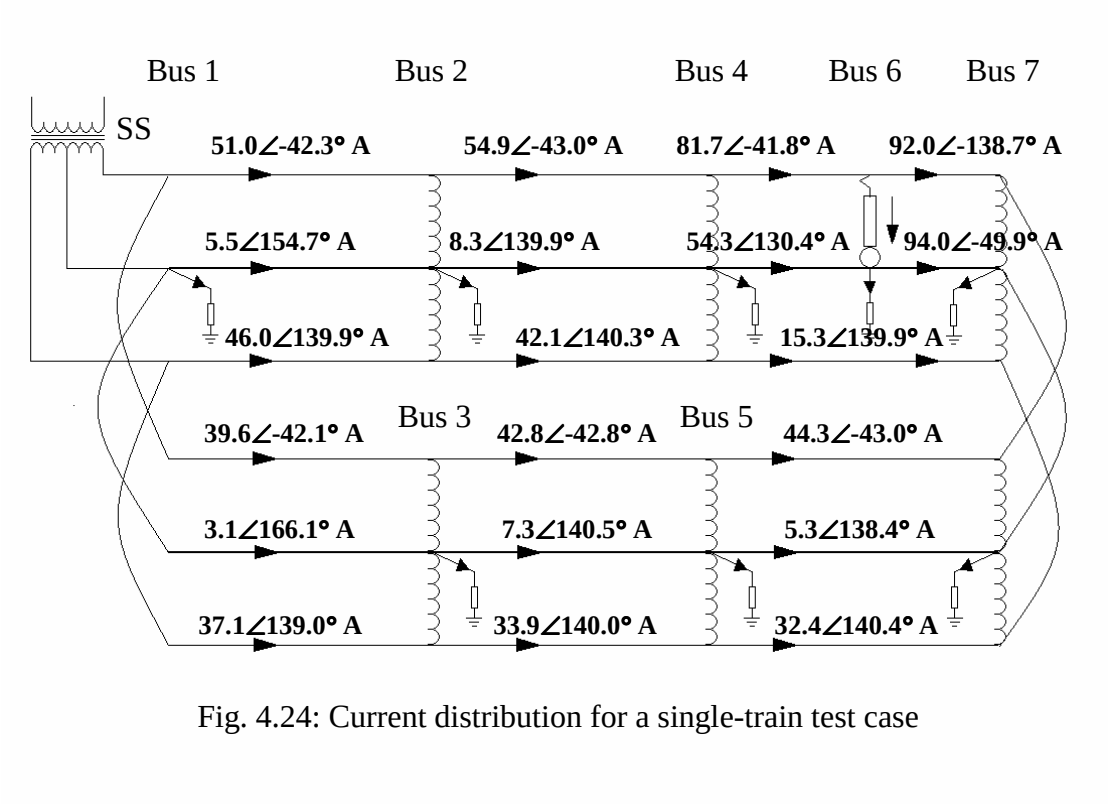


Fig. 4.24: Current distribution for a single-train test case

Table 4.4: Bus voltages, earth-leakage currents and train current for the test case

Bus	Catenary voltage (kV)	Rail voltage (V)	Feeder voltage (kV)	Cumulative earth leakage current (A)
1	24.11∠-1.9°	5.16∠-21.3°	24.15∠178.1°	5.16∠-21.3°
2	23.96∠-2.2°	4.88∠138.8°	23.97∠177.8°	4.88∠138.8°
3	24.00∠-2.1°	3.25∠138.4°	24.00∠177.9°	3.25∠138.4°
4	23.79∠-2.5°	15.42∠-171.7°	23.82∠177.5°	15.42∠-171.7°
5	23.87∠-2.4°	5.03∠137.8°	23.87∠177.6°	5.03∠137.8°
6	23.63∠-2.9°	68.77∠-3.3°	23.80∠177.6°	68.77∠-3.3°
7	23.72∠-2.6°	27.14∠-156.2°	23.77∠177.4°	27.14∠-156.2°
Train current (A)			172.94∠-39.7°	

4.6 Summary

This chapter discusses and exploits the band diagonal system for solving the AC railway power feeding systems. A heuristic bus ordering technique leading to the banded matrix called "Banded Bus-Ordering" or BBO is defined and thus enables the efficient and fast algorithm to obtain the power system solution.

The advantages of using the band matrix computation and the recursive start can be summarised as follows. Note that the percentages shown below are the average value of all relevant test cases in Tables 4.1 – 4.3.

- Recursive start can reduce the execution time by 53.1%
- BBO algorithm can reduce the execution time by 46.9%
- Combining the BBO and the recursive start can greatly reduce the execution time by 75.1%

The band diagonal technique is simple and distinctive from other sparsity techniques. Unlike other techniques, it has significant effects in reducing the execution time, not only for a large sparse system but also for a small sparse system that has a matrix dimension of less than 100, for example.

PART III

Optimal Area Control System

CHAPTER 5

OPTIMAL AC RAILWAY POWER FLOW

5.1 Introduction

The impedance of the overhead catenary feeding system and the train loading cause imperfections in the AC railway power supply, e.g. voltage drop along the feeder length, severe harmonic currents caused by powerful trains operating in poor power-factor conditions (0.5 – 0.85 typically), etc. Although most electric trains can operate under a wide range of pantograph voltage (16.5 kV – 27.5 kV is specified for normal operation in the UK's 25 kV system [9]), the traction performances and the power supply conditions are better at nominal voltage. In general, operating at low voltage level leads to a considerable amount of reactive power being drawn from the supply. This can lead to the system being overloaded with reactive power and thus insecure power supply operation in the feeding system. Thus, the compensation of excessive reactive power consumption by installing external reactive power sources has become one possible solution to power supply instability.

The main purpose of this chapter is to introduce and illustrate a new, alternative method of reactive power compensation. However, for comparison, the standard track-side reactive power compensation techniques are reviewed and discussed first.

5.2 Practical Use of the SVC in AC Railway Electrification

The SVC has become practically standard for upgrading electrified railway systems for several decades, especially in cases of weak supply. Reactive power consumed by loads or caused by line reactance can be adequately compensated by the SVC. It physically consists of a Thyristor controlled reactor working together with a capacitor bank as previously described in Chapter 2.

No more explanation about the SVC structure is given here, this section only presents the practical use of the SVC in the AC railway industry.

In east Central Queensland, Australia, a coal-haulage electrified railway network is supplied at 132 kV from the local high voltage grid. There are thirteen power substations with the autotransformer feeding system to provide a 25 kV overhead catenary voltage as shown in Fig. 5.1. In 1987, a tenth SVC rated 340 MVar was installed at Nebo substation as one of ten substations in an SVC installation plan undertaken by ABB [58,59].

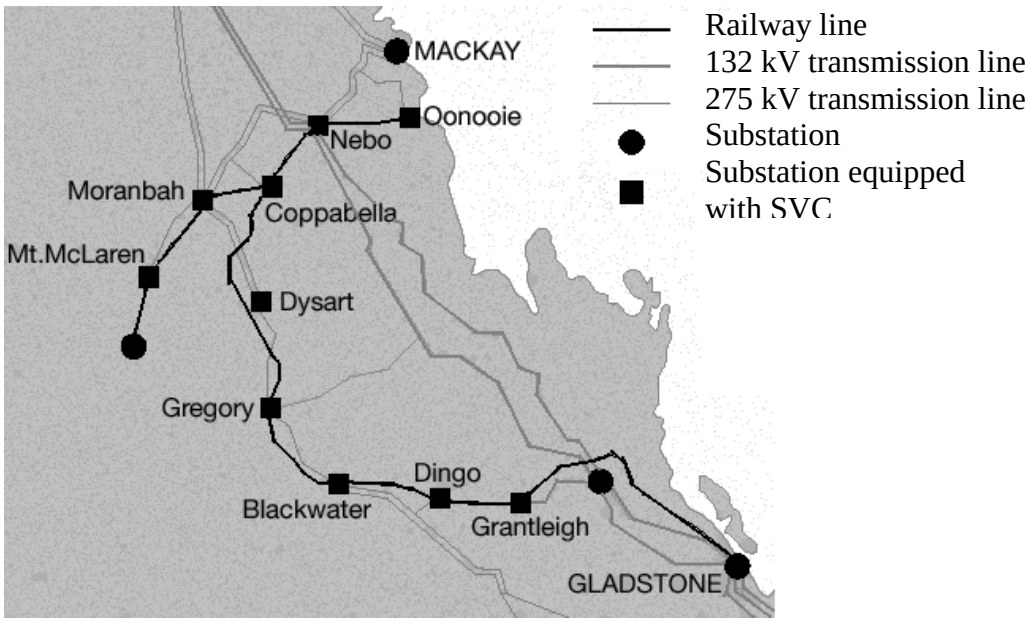


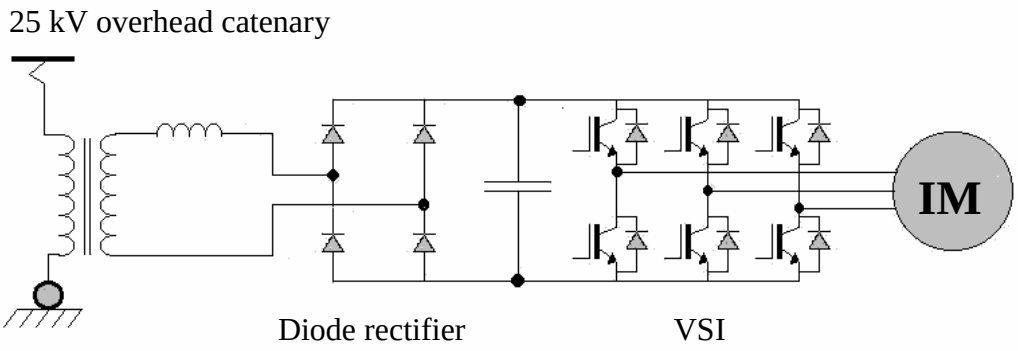
Fig. 5.1: Coal-haulage electrified railway network in Central Queensland, Australia

There are two disadvantages of using the SVC. It is costly and needs significant space for installation. An SVC cost approximation from the CANMET Energy Technology Centre (CETC) [60] gives the SVC cost per kVar as about £86.4 in 1992. The calculation shows that, based on this estimation, the SVC installation at the Nebo substation costs the considerable sum of about £29m. In addition, Fig. 5.2 shows the aerial view of the large SVC installation yard at Oonooie substation, illustrating the large nature of the equipment.

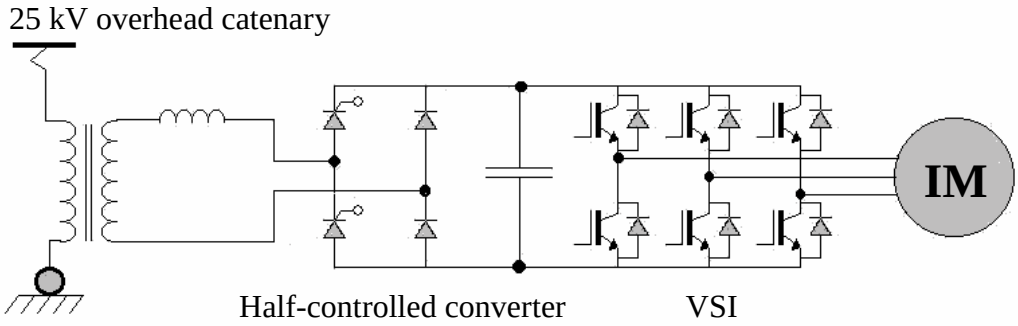
Fig. 5.2: Aerial view of the SVC installation yard along side the rail-track [58]

5.3 PWM Locomotive as a Mobile Reactive Power Compensator

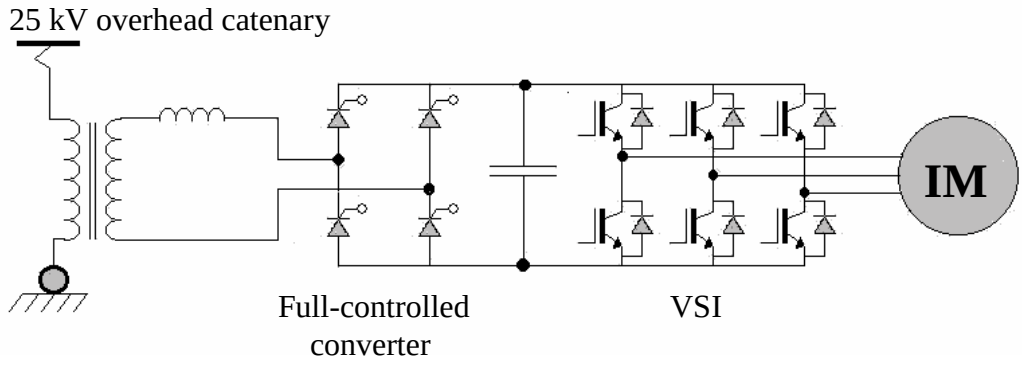
In this thesis, AC locomotives are classified into two main categories: i) Diode or Thyristor phase-angle controlled Locomotives (PAC locomotives) and ii) PWM locomotives. The on-board input converter of PAC locomotives is either a Diode or a Thyristor phase-angle controlled rectifier equipped as shown in Fig. 5.3. Power factor during service is typically between 0.5 and 0.85 [9]. This leads to a significant amount of reactive power being drawn from the power supply. For modern electric locomotives, a PWM converter as shown in Fig. 5.4 replaces the phase-angle controlled converter. It can operate at high power factors, such as 0.95 – 1.00 [58]. Notably, with a suitable PWM converter design, it can also operate at leading power factor.



a) Diode locomotive



b) Thyristor locomotive with a half-controlled converter



c) Thyristor locomotive with a full-controlled converter

Fig. 5.3: Diode and Thyristor phase-angle controlled locomotives

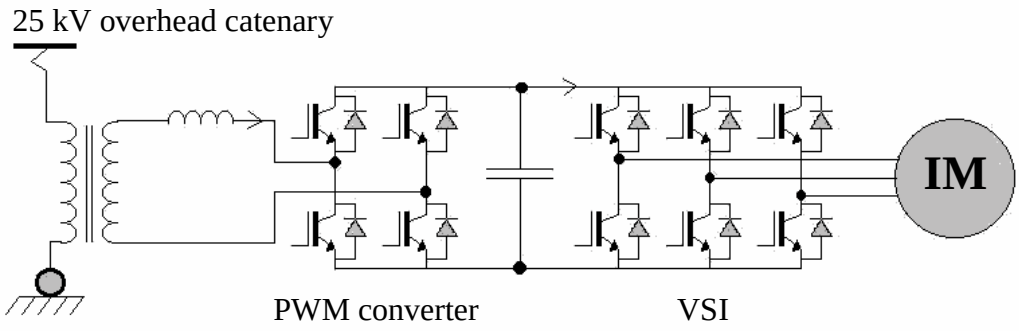
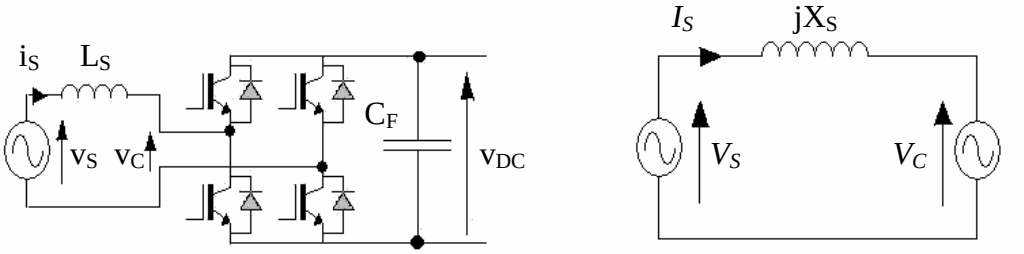


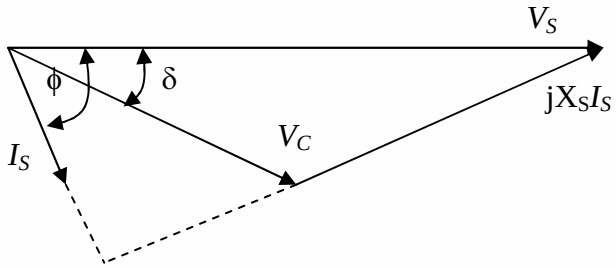
Fig. 5.4: PWM locomotive

As shown in Figs 5.3–5.4, a typical 25 kV, 50 Hz locomotive has a traction transformer to feed an input converter. With a DC link capacitor and a VSI mounted on-board, the induction motors can be fully controlled. When a PWM converter is used to replace an old fashioned diode or phase-angle controlled converter, current drawn from an overhead contact wire can be controlled in-phase with the pantograph voltage (unity power factor condition) or even to a leading phase angle. This means that the PWM locomotive is able to control reactive power drawn from the overhead catenary. An on-board controller for PWM locomotives is normally designed to operate at as high power factor as possible. Although this strategy performs very well in not drawing reactive power from the power supply, it does nothing to compensate the entire system where a large amount of reactive power is still being drawn by other PAC locomotives. Therefore, the expensive SVC is required. However, the SVC sometimes fails to perform compensation efficiently when the reactive power is caused by a poor power-factor locomotive far from the SVC position. Voltage drops and power losses are worse in this circumstance. With control of the PWM locomotive reactive powers, the PWM locomotives can help compensate the excess reactive power for the entire feeding section. This leads to more efficient reactive power compensation.

The SVC is special equipment designed primarily to supply reactive power to the system. It has a wider range of reactive power compensation than that of a typical PWM locomotive. However, when the on-board PWM power converters are specially designed for compensation, such a disadvantage will be eliminated. Unlike the SVC, PWM locomotives can move along the line and be closer to a load point so as to have significant effects on the entire power feeding system.



a) Schematic diagram b) Steady-state equivalent circuit



Where

$$V_S = \frac{\hat{V}_S}{\sqrt{2}} \angle 0^\circ$$

$$V_C = \frac{mV_{DC}}{\sqrt{2}} \angle -\delta$$

c) Phasor diagram

Fig. 5.5: PWM power converter

Consider a single-phase lossless PWM power converter as shown in Fig. 5.5a. Although the PWM converter involves non-linear switched-mode operation, the equivalent circuit written for the fundamental frequency as shown in Fig. 5.5b can adequately describe its performance in steady-state conditions [61,62]. The voltage source, V_S, represents the pantograph voltage. Therefore, real and reactive power flows through the PWM converter can be obtained from the following expressions.

$$S_S^* = V_S^* I_S = \frac{V_S^*(V_S - V_C)}{jX_S} = \frac{mV_{DC}\hat{V}_S}{2X_S} sin\delta - j\left(\frac{\hat{V}_S^2 - mV_{DC}\hat{V}_S cos\delta}{2X_S}\right)$$

$$= P_S - jQ_S$$

Thus,

$$P_S = \frac{mV_{DC}\hat{V}_S}{2X_S} sin\delta \tag{5.1}$$

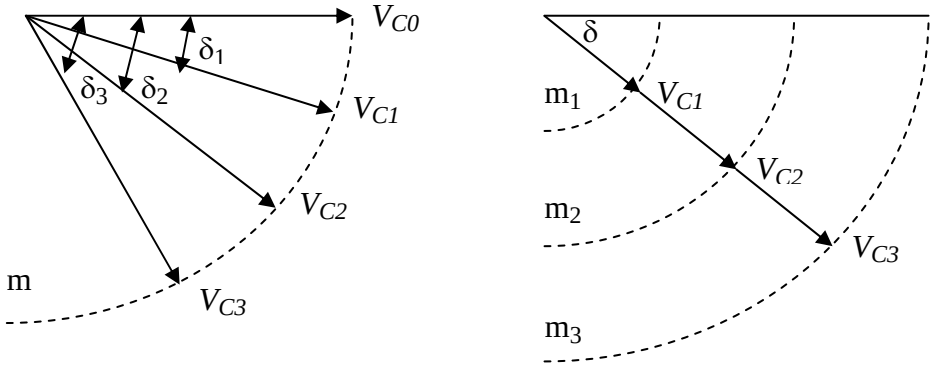
$$Q_S = \frac{\hat{V}_S^2 - mV_{DC}\hat{V}_S cos\delta}{2X_S} \tag{5.2}$$

To act as a mobile reactive power compensator, the reactive power flow through the PWM power converter must be adjustable while separately regulating the real power demand. If we assume that the real power demand of the converter is constant as described in equation 5.3, the reactive power can be re-written as in equation 5.4.

$$P_S = \frac{mV_{DC}\hat{V}_S}{2X_S} sin\,\delta = \tilde{P}_S \tag{5.3}$$

$$Q_S = \frac{\hat{V}_S^2}{2X_S} - \frac{\tilde{P}_S}{tan\,\delta} \tag{5.4}$$

Clearly, the reactive power depends on four key factors: i) pantograph voltage-magnitude transferred to the secondary side of the on-board traction transformer, ii) input reactance, iii) real power demand and iv) phase angle δ of the modulating waveform with respect to the carrier waveform. The real power demand is assumed to be constant while the pantograph voltage and the input reactance vary, depending on the train position, etc. Thus, only δ is available for the reactive power control.



a) Phasor V_C with varying δ b) Phasor V_C with varying m

Fig. 5.6: Phasor diagram for VC resulting from m and δ control

When $\hat{V}_S$, X_S and V_{DC} are specified, only m and δ control the converter operating conditions. Assume that the modulation index and the phase angle can be adjusted within the ranges of $m \in [m_{min}, m_{max}]$ and $\delta \in [\delta_{min}, \delta_{max}]$. The V_C phasor resulting from these two parameters is shown in Fig. 5.6. For constant real power operation when the pantograph receives power from

the overhead catenary at a certain voltage level of V_S and the DC link voltage, V_{DC}, is well regulated, the product of m and $sin\delta$ is constant, that is $m \cdot sin\delta = \frac{2X_S\tilde{P}_S}{V_{DC}\hat{V}_S}$. Consider equation 5.5,

$$V_C = \frac{mV_{DC}\hat{V}_S}{\sqrt{2}}\angle\delta = \frac{mV_{DC}\hat{V}_S}{\sqrt{2}}(cos\delta + jsin\delta) \tag{5.5}$$

Therefore, the vertical component of V_C is constant. With varying m and δ subject to constant real power operation, the phasor diagram for all feasible operating regions consisting of lagging, unity and leading power factors is formed in Fig.5.7. In addition, Fig. 5.8 shows the generated operating region for all feasible real power operations.

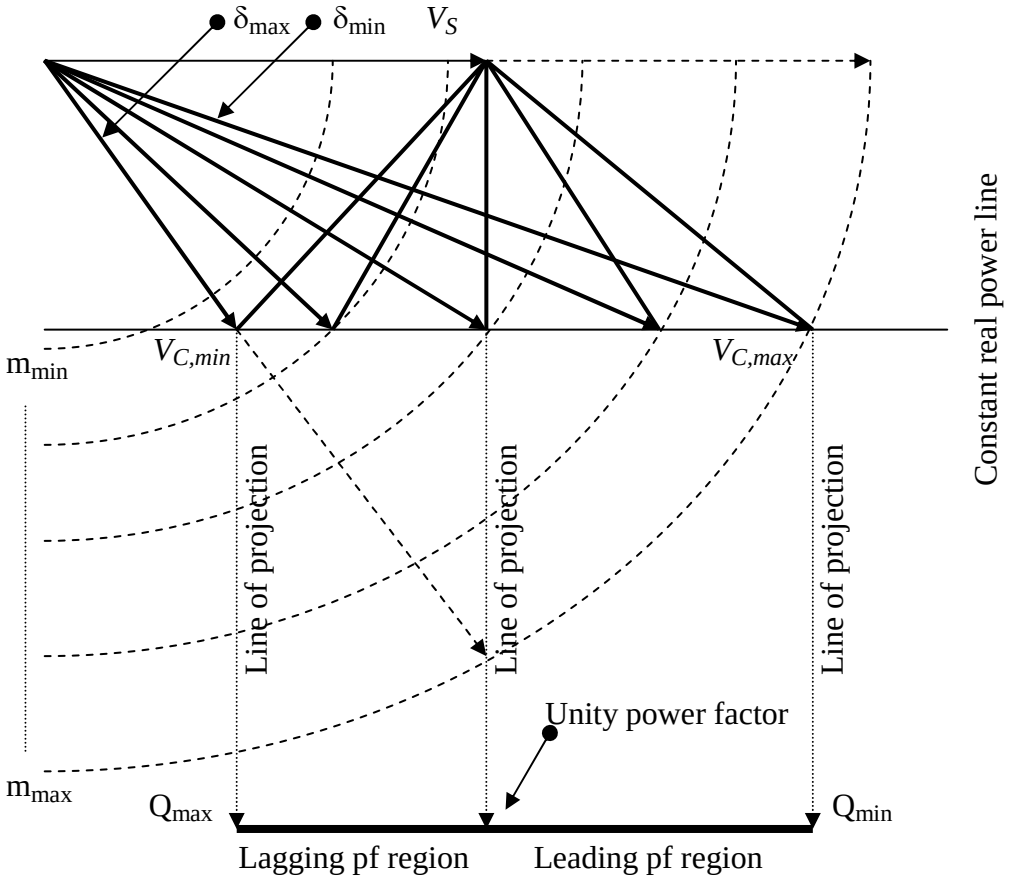


Fig. 5.7: Feasible operating region for constant real power operation

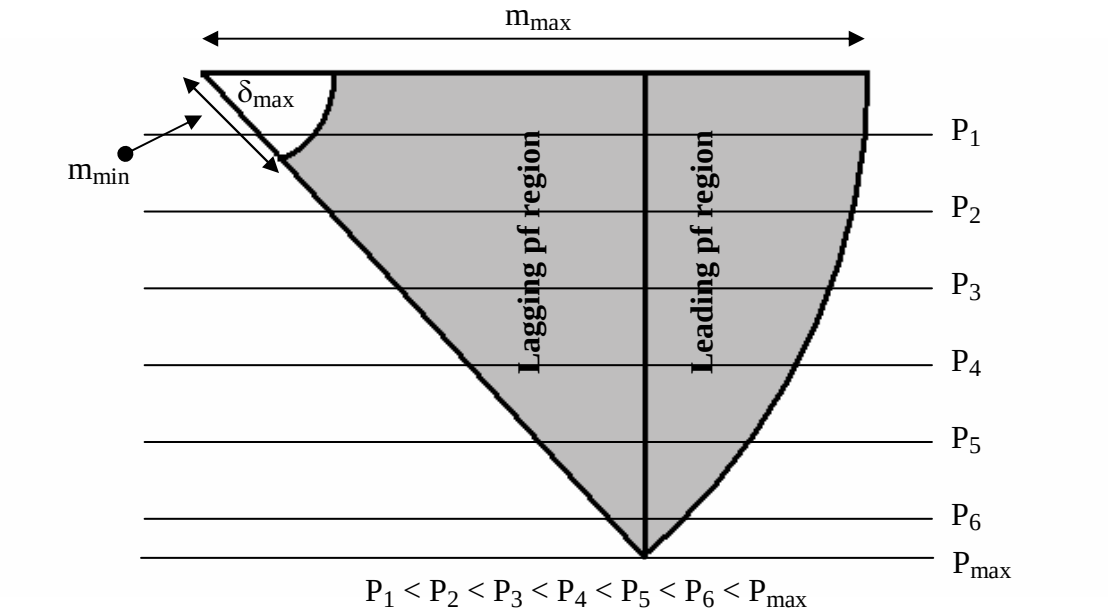


Fig. 5.8: Feasible operating region for the PWM power converter where $\hat{V}_S$ is fixed

The feasible region in Fig. 5.7 is governed by the limits of m and δ. The DC link voltage is normally regulated by an on-board traction controller, this parameter is also constant. From all these assumptions, minimum and maximum reactive powers drawn by the PWM power converter at the specified real power, $\tilde{P}_S$, are expressed in equations 5.6 and 5.7.

$$Q_{S,min} = \frac{\hat{V}_S^2}{2X_S} - \frac{\tilde{P}_S}{tan\ \tilde{\delta}_{min}} \tag{5.6}$$

$$Q_{S,max} = \frac{\hat{V}_S^2}{2X_S} - \frac{\tilde{P}_S}{tan\ \tilde{\delta}_{max}} \tag{5.7}$$

Consider the horizontal lines $P_1 - P_6$ in Fig. 5.7, the minimum reactive power is limited by m_{max}, together with the constant real power constraint, therefore

$$\tilde{\delta}_{min} = sin^{-1}\left(\frac{2X_S\tilde{P}_S}{m_{max}V_{DC}\hat{V}_S}\right) \tag{5.8}$$

For the maximum reactive power, the minimum phase angle, δ_{max}, will limit the upper bound of the reactive power. However, with a small amount of real power demand where the minimum

modulation index, m_{min}, is violated regarding the constraint of $m \cdot sin\delta = \frac{2X_S\tilde{P}_S}{V_{DC}\hat{V}_S}$, δ cannot reach its maximum value. In this case, the maximum phase angle is reduced from δ_{max} to $\tilde{\delta}_{max}$ as shown in equation 5.9.

$$\tilde{\delta}_{max} = sin^{-1}\left(\frac{2X_S\tilde{P}_S}{m_{min}V_{DC}\hat{V}_S}\right) \tag{5.9}$$

Example 1: Suppose that a PWM locomotive has the following parameters:

L_S 0.981 mH

V_{DC} 1800 V

$\hat{V}_{S,max}$ 2160 V

$m \in [0.45, 0.95]$

$P_{S,max}$ 2.5 MW

Find the maximum phase angle, δ_{max}, $\delta_{max} = sin^{-1}\left(\frac{2X_SP_{max}}{m_{max}V_{DC}\hat{V}_S}\right) = 24.7^\circ$

Thus, $\delta \in [0^\circ, 24.7^\circ]$

This example discusses the reactive power flow capacity resulting from the two key factors (P_S and $\hat{V}_S$) that are usually changed in each time update.

i) Keep $\hat{V}_S$ constant at 1250 V while varying P_S

ii) Keep P_S constant at 1.2 MW while varying $\hat{V}_S$

Figs 5.9 and 5.10 show families of curves generated for cases i and ii, respectively.

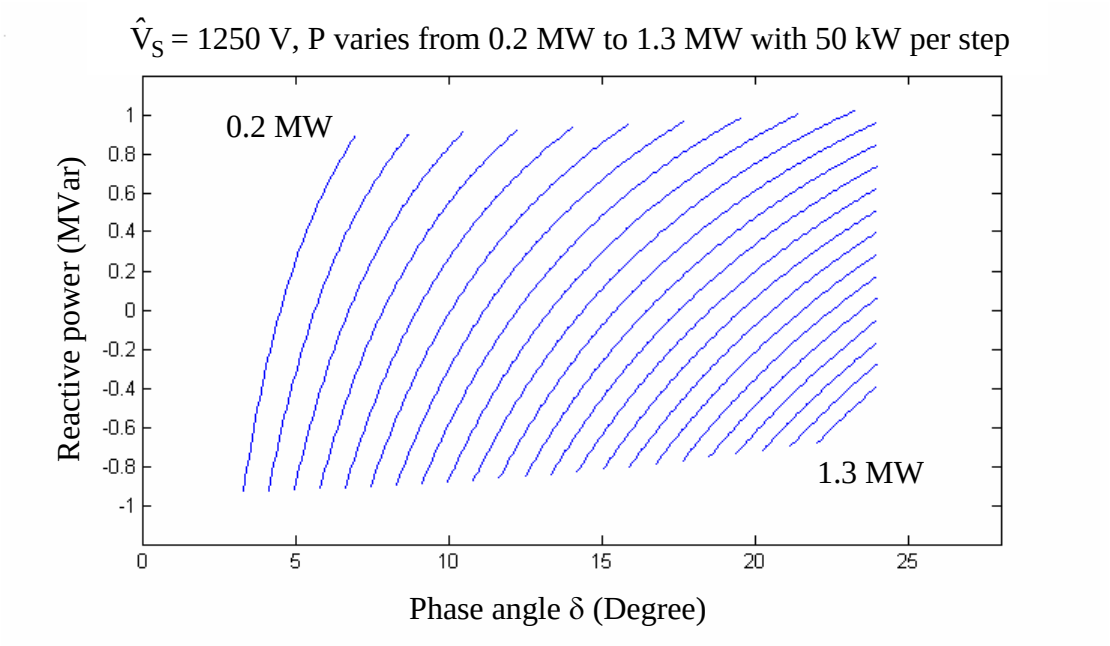


Fig. 5.9: Reactive power resulting from the real power demand

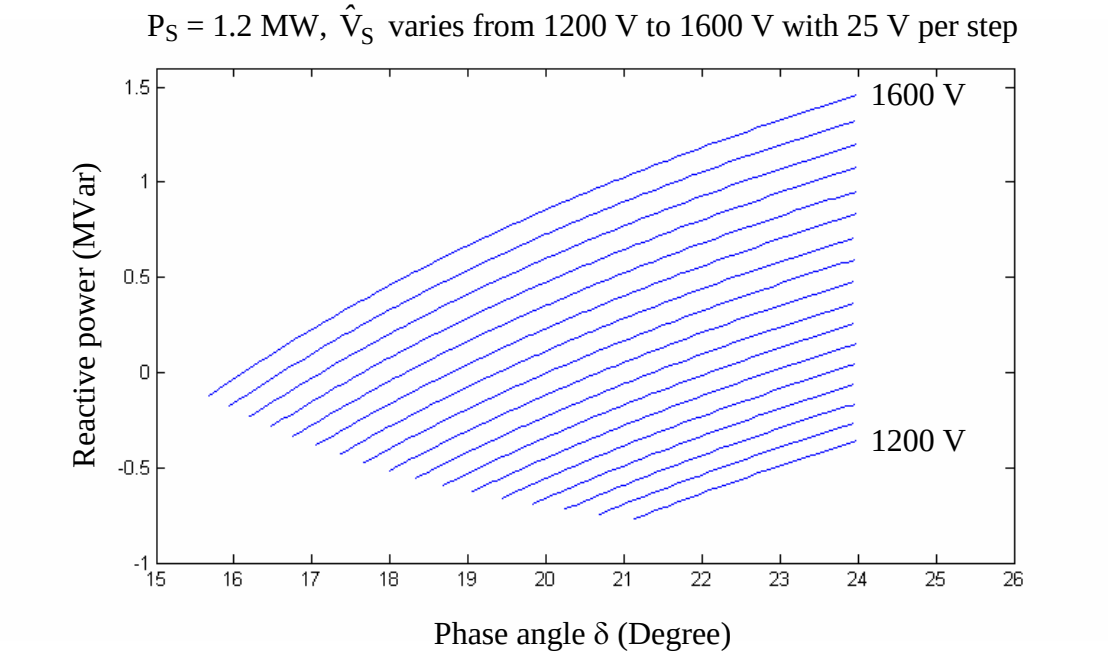


Fig. 5.10: Reactive power resulting from $\hat{V}_S$

As can be seen when the PWM locomotive is running along the line, the maximum and minimum reactive power limits depend upon many factors. This illustrates the possibility of using the PWM locomotive as a mobile reactive power compensator but a special converter design may be required to enhance the reactive power capacity.

In the implementation of this compensation, the response time to reactive power command is as important as the reactive power limits. The PWM converter control for the electric traction is not new; there are several related examples especially for railway applications. DC link voltage control has been carefully investigated and well designed for the PWM locomotive [63]. The input current and reactive power control techniques have also been developed, for example the input voltage vector control (VVC), the input direct current control (DCC), etc [37]. Although leading power factor operation can be performed, most PWM locomotives normally operate at as high a power factor as possible [37,61–64].

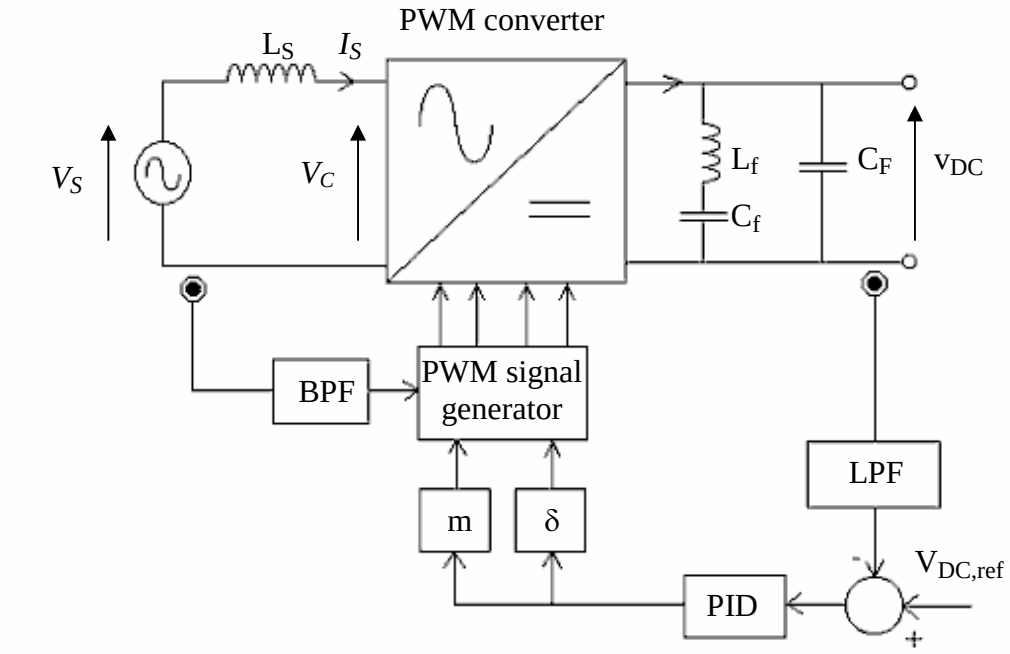


Fig. 5.11: Single-loop voltage vector control for PWM converter

In following, the response time of the reactive power flow through the PWM converter is investigated. The reactive power control technique used here is based on the technique described in [37] but with slight modification to adjust the reactive power flow rather than to regulate unity power-factor operation. It has been reported that the DCC schemes have some advantages over the VVC schemes in transient conditions [37]. This implies that the response time of the DCC tends to be faster. To specify the time step to update the reactive power

command, the controller that gives the slowest response time is more appropriate. The time step must be sufficiently greater than the response time of the reactive power controller to ensure that the locomotive reactive power can reach the steady-state value before receiving a new command. Thus, the reactive power flow control through the PWM power converter will be studied by using the VVC schemes. In any event, it is highly unlikely that DCC could be used in a railway because of its potential for generating unpredictable signal interference.

The control diagram for the VCC is shown in Fig. 5.11. The DC link voltage is fed back through the LPF and compared with the reference value. The PID controller is used to perform an error-less and fast-response regulation for the DC link voltage. The output of the PID controller is regarded as being proportional to the load current and is used to calculate the modulation index and phase angle with equations 5.10 and 5.11 [37].

$$\delta = tan^{-1}\left(\frac{\omega L_S I_S V_S \, sin\phi}{V_S \, cos\phi - R_S I_S}\right) \tag{5.10}$$

$$m = \frac{V_S \, cos(\phi + \theta)}{V_{DC} \, cos(\delta - \phi - \theta)}, \qquad \text{where } \theta = tan^{-1}\left(\frac{R_S}{\omega L_S}\right) \tag{5.11}$$

Note that R_S represents the transformer equivalent resistance and switching losses of the power converter, and is not shown in Fig. 5.11.

The VCC is to control the single-phase PWM power converter operating at a specified power factor as explained above. In general, such a converter can be fully controlled with adjustable reactive power while keeping the DC link voltage constant. To replace the power factor command by the reactive power command needs mathematical models of the reactive power and DC link voltage relating to m and δ as follows.

First, a load connected across the DC link terminal is modelled as a resistance R_M. Modifying the equations 5.1 and 5.2 with this assumption, the DC link voltage and reactive power can be simply obtained by equations 5.12 and 5.13.

$$V_{DC} = \frac{m\hat{V}_S R_M}{2X_S} sin\,\delta \tag{5.12}$$

$$Q_S = \frac{\hat{V}_S^2 - mV_{DC}\hat{V}_S\, cos\,\delta}{2X_S} \tag{5.13}$$

In this simplification, if the DC link voltage is adequately regulated as the set value ($V_{DC,ref}$) and the power (P_M) required by motors (loads connected across the DC link) is assumed to be constant at that instance, R_M can be simply calculated by $R_M = \frac{V_{DC,ref}^2}{P_M}$. Using Taylor series expansion, the changes in V_{DC} and Q_S can be approximated by the following expressions.

$$\Delta V_{DC} = \frac{\partial V_{DC}}{\partial m}\Delta m + \frac{\partial V_{DC}}{\partial \delta}\Delta\delta \tag{5.14}$$

$$\Delta Q_S = \frac{\partial Q_S}{\partial m}\Delta m + \frac{\partial Q_S}{\partial \delta}\Delta\delta \tag{5.15}$$

where

$$\frac{\partial V_{DC}}{\partial m} = \frac{\hat{V}_S R_M}{2X_S} sin\,\delta; \qquad \frac{\partial V_{DC}}{\partial \delta} = \frac{m\hat{V}_S R_M}{2X_S} cos\,\delta$$

$$\frac{\partial Q_S}{\partial m} = -\frac{mV_{DC}\hat{V}_S}{2X_S} cos\,\delta; \qquad \frac{\partial Q_S}{\partial \delta} = \frac{mV_{DC}\hat{V}_S}{2X_S} sin\,\delta = P_S$$

With these formulae, the modulation index and the phase angle can be repeatedly updated until V_{DC} and Q_S converge. In addition, to implement this control scheme, PID controllers can be used to gain better responses. The closed-loop control system for this operation is shown in Fig. 5.12. Full detail for analysis of the PWM power converter operation is found in Appendix C while programming code for this simulation is written in Appendix G.3.

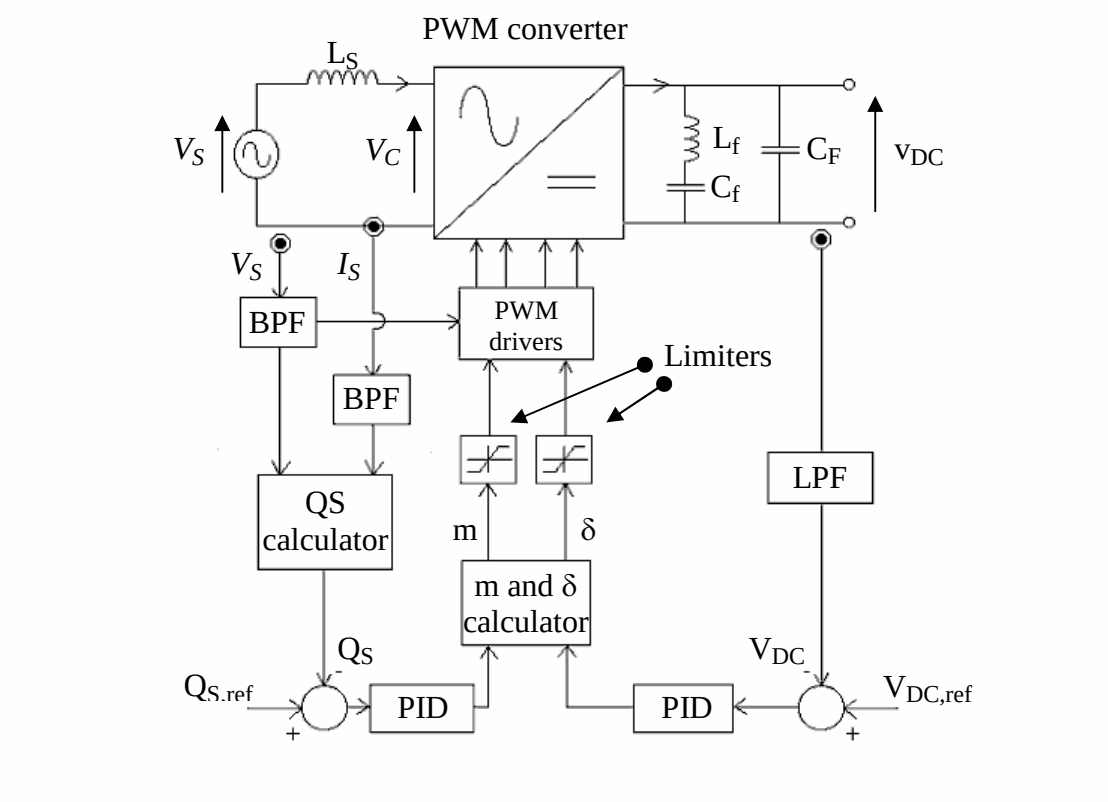


Fig. 5.12: Adjustable reactive power control and DC link voltage regulation

Example 2: Suppose that a PWM locomotive has the following parameters:

R_S	0.05 Ω	L_S	0.981 mH
Rf	0.01265 Ω	Lf	0.253 mH
Cf	10 mF	C_F	10 mF
$V_{DC,ref}$	1800 V	$\hat{V}_S$	1500 V
P_M	1.0 MW	$m \in [0.45,0.95]$	
$\delta \in [10^\circ, 30^\circ]$		Modulation ratio = 12	

Due to the definition for a steady-state power as the integration of a product of v_S and i_S over a full period, a 20 ms sampling period or 50 Hz sampling frequency is used to simulate this control operation. There are three reactive power commands, 0, -0.5 and 0.2 MVar. They are applied to the control system at time 0, 800 and 1600 ms, respectively. The DC link voltage is kept constant at 1800 V for all time. The simulation results, i) reactive power, ii) DC link

voltage, iii) input current, iv) modulation index, v) phase angle and vi) phase differences for each reactive power demand are shown in Figs 5.13 – 5.17.

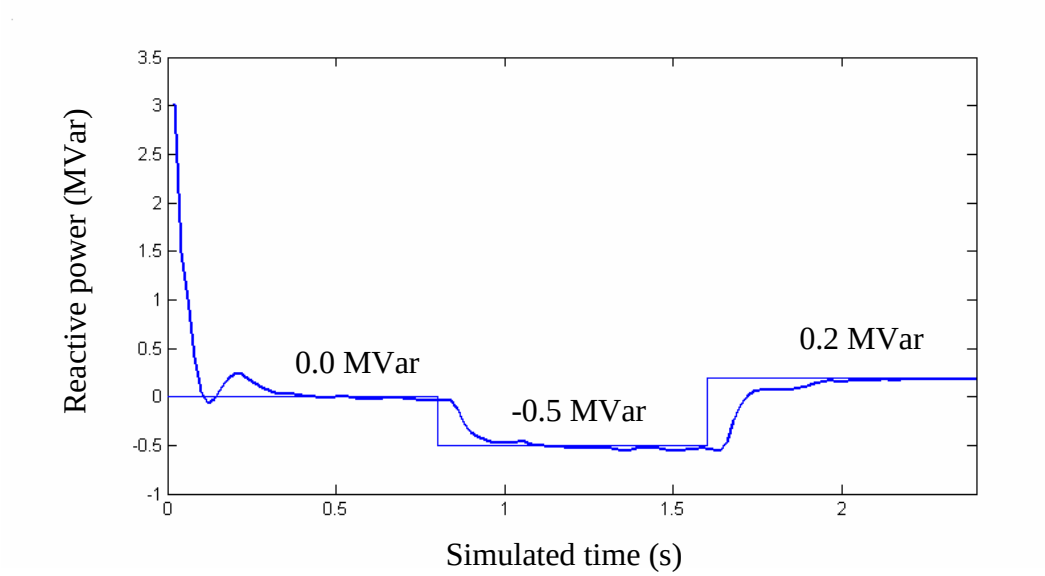


Fig. 5.13: Reactive power response

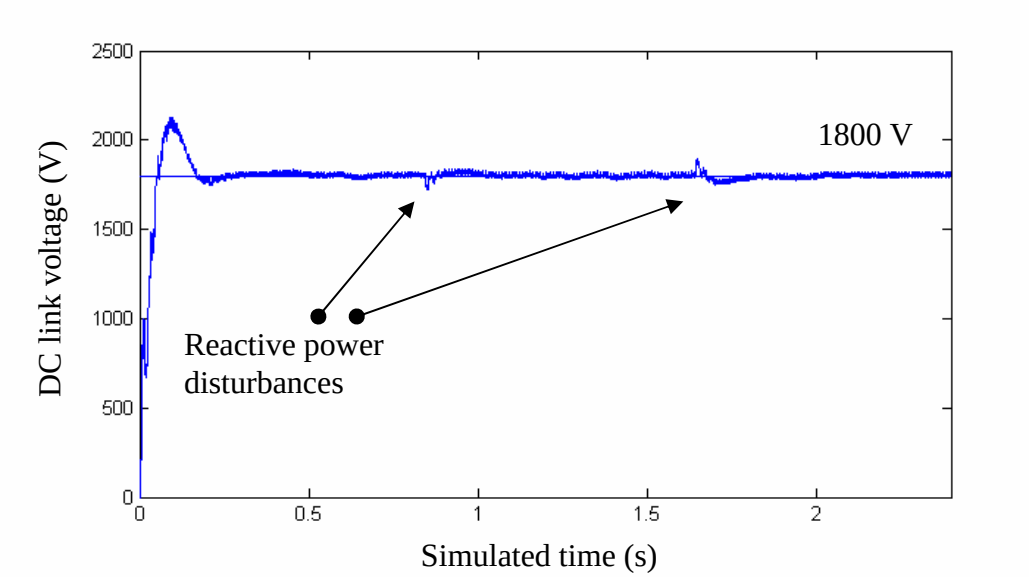


Fig. 5.14: DC link voltage response

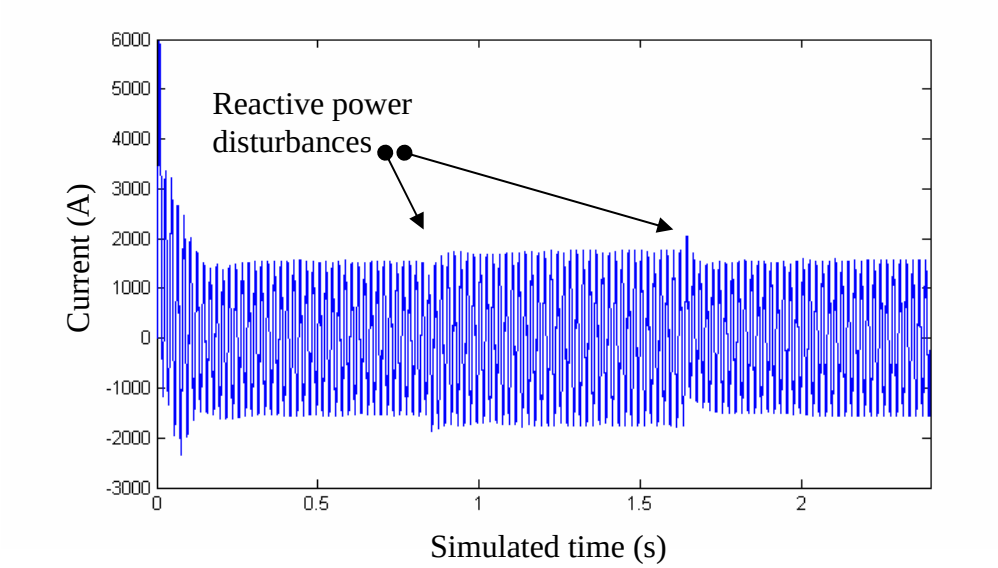


Fig. 5.15: Input current response

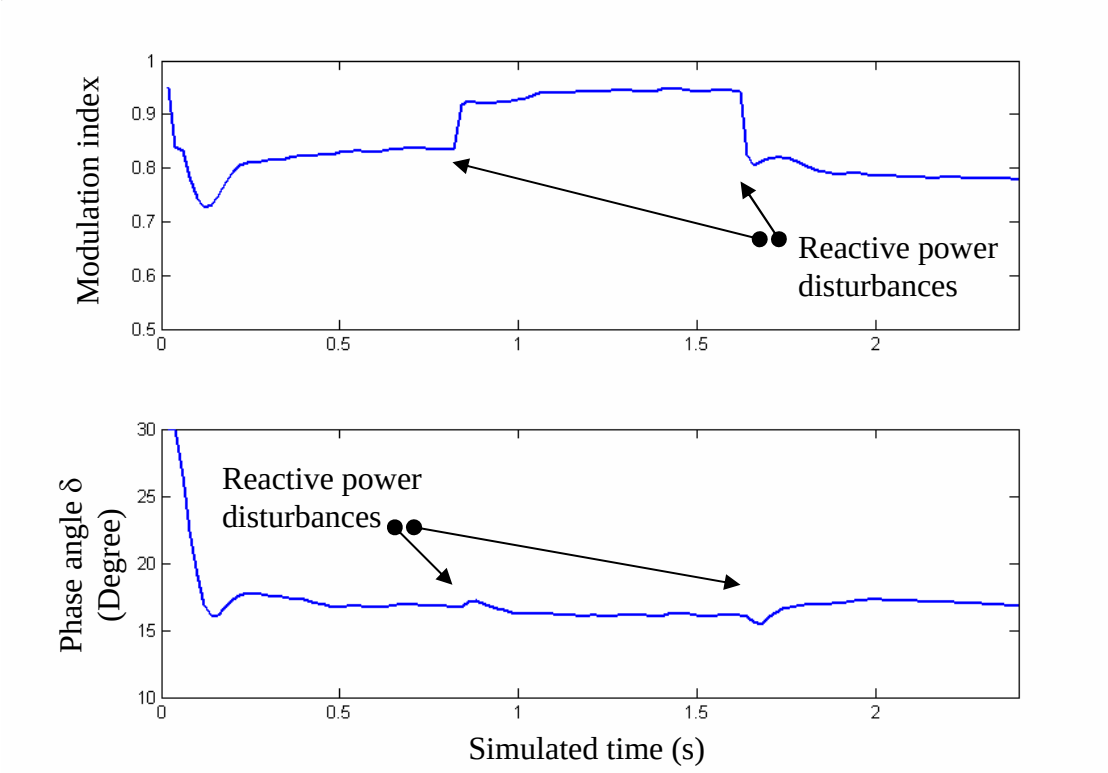


Fig. 5.16: Modulation index and phase angle

As shown in Fig. 5.16, the modulation index mainly causes the change in the reactive power while the phase angle δ is nearly constant. This gives a suggestion to a further development of decoupling control for real and reactive power flows through single-phase PWM power converters as the decouple power flow analysis [12,13].

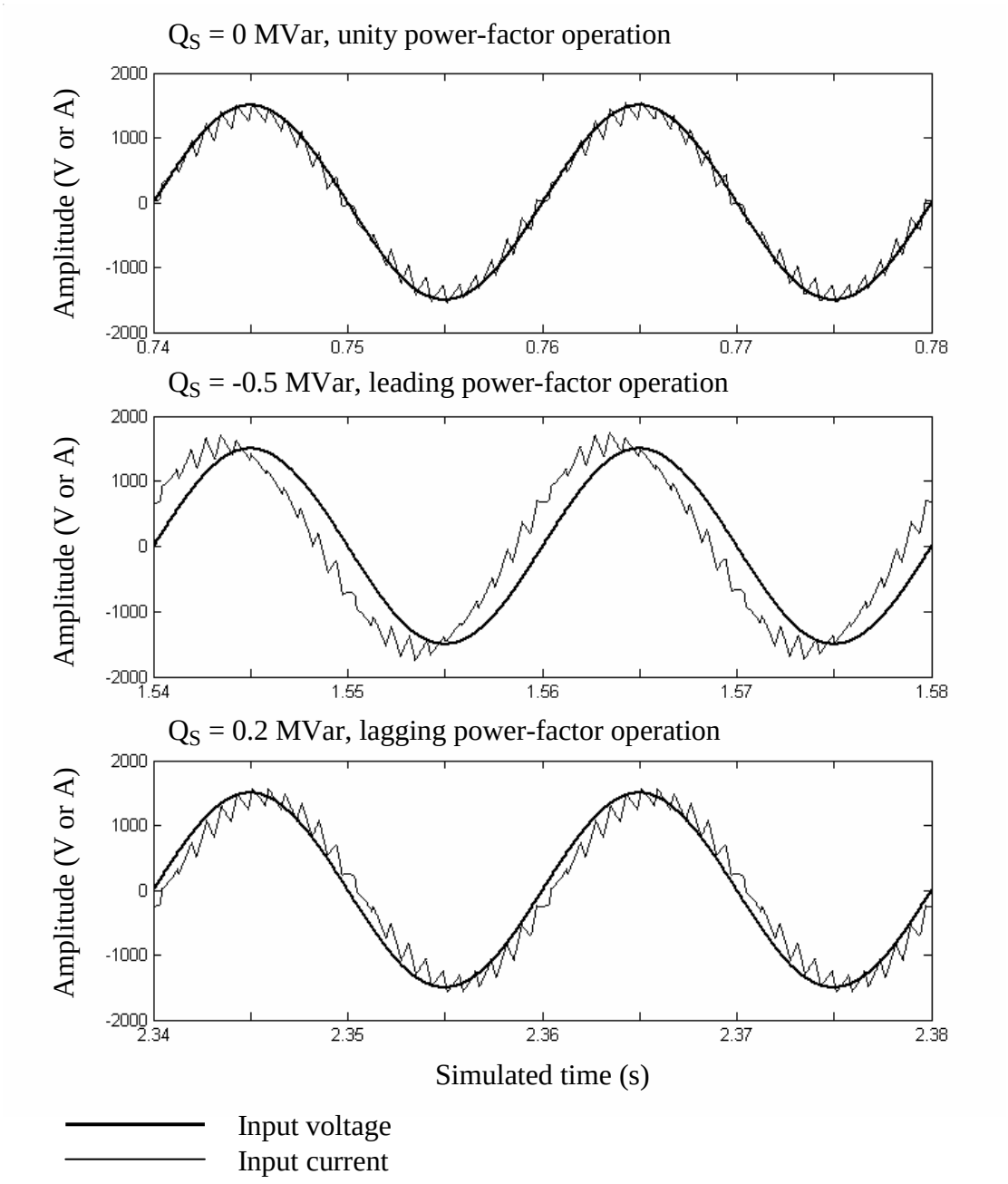


Fig. 5.17: Input voltage vs input current for each case

This simulation was performed with the PID parameters selected as $k_P = 0.8$, $k_I = 0.01$ and $k_D = 0.02$ for the DC link voltage control, and $k_P = 0.2$, $k_I = 0.05$ and $k_D = 0.01$ for the reactive power control. These PID parameters give a typical response for the reactive power controller. It is possible to achieve slightly faster responses, but there lead to exhibit excessive overshoot. From the simulation, the response time is about 200 – 300 ms. Thus, it is reasonable to update

the reactive power command from the area controller at every 500 ms. 500 ms will thus be the time step for all the simulation programs developed in this thesis.

5.4 Optimal AC railway power flow problem and its solution

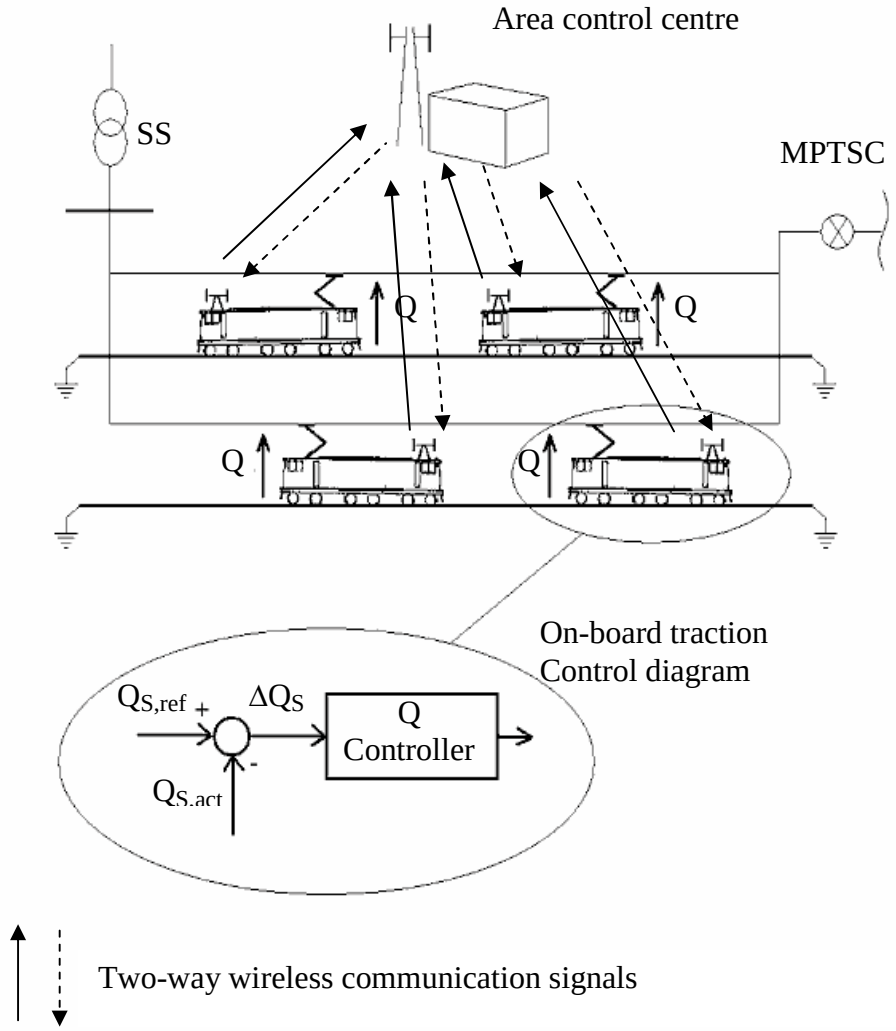


Fig. 5.18: Area control system of AC locomotive powers

A centralised area control of the locomotive reactive powers can improve system performance. One possible area control system is as indicated in Fig. 5.18. Individual on-board controllers are co-ordinated in such a way as to achieve overall performance objectives for the complete system of substations, overhead supply lines and trains. Suitable objectives are reduced transmission losses or improved voltage profile. These objectives are not in fact independent

and either may be formed into an objective function which can then be optimised subject to control variable constraints such as the limit on available reactive power at each train.

Some essential assumptions are required to implement this control system,. For example, it is necessary to have two-way communication between the trains and the control centre as illustrated in Fig. 5.17. The instantaneous values of the train positions and power demands are fed back to the central controller. At each update, the central controller performs the optimisation and thus reactive power commands are sent out to each PWM locomotive and used as the reference signal to an on-board closed-loop controller. All the assumptions are summarised as follows.

1. Area control system for one feeding section is developed
2. Only steady-state operation is taken into account
3. High-frequency harmonics drawn by the all locomotives are neglected
4. Two-way wireless communication between any locomotive and the control centre is available
5. All PWM locomotives are specially designed to supply a sufficient amount of reactive power as required by the central controller within certain reactive power limits.

- **Problem Formulation**

The main concept of this system is to minimise power losses by appropriately adjusting the reactive power of controlled PWM locomotives acting as mobile reactive power compensators. Assume that a given AC railway system contains N buses and M PWM locomotives. In addition, any PWM locomotive is able to provide reactive power only within the range of given reactive power limits. Thus, the system formulation can be expressed in the general form of an optimisation problem as follows.

$$\underset{\mathbf{q}^{min} \le \mathbf{q} \le \mathbf{q}^{max}}{\text{Minimise}} \quad f(\mathbf{q})$$

where

q denotes the reactive power of the PWM locomotives

$\mathbf{q}^{min}$ denotes the lower bound of the PWM locomotive reactive powers

$\mathbf{q}^{max}$ denotes the upper bound of the PWM locomotive reactive powers

f(**q**) denotes the total power losses

Since the power loss function is not a simple function of reactive power, an indirect calculation of power losses must be determined from the voltage solution [12,23]. As shown in Fig. 5.19, the power losses caused by power flowing through feeder sub-sections can be calculated in the following steps. Firstly, compute the complex conjugate of power flowing through a feeder sub-sections connected between bus i and bus j, S_{ij}^{*}, and similarly in the reverse direction, S_{ji}^{*}. Secondly, the power loss, P_{LOSS}, of any feeder sub-section is defined by the real part of the summation of these two complex powers as shown in equations 5.16–5.18.

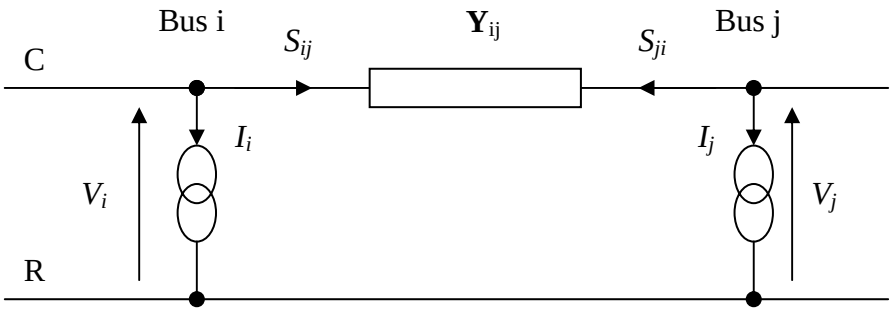


Fig. 5.19: Circuit representation of a feeder portion

$$S_{ij}^{*} = \mathbf{Y}_{ij} V_i^{*} \left(V_i - V_j \right) \tag{5.16}$$

$$S_{ji}^{*} = \mathbf{Y}_{ij} V_j^{*} \left(V_j - V_i \right) \tag{5.17}$$

$$P_{LOSS} = \mathrm{Re}\left\{ S_{ij}^{*} + S_{ji}^{*} \right\} = \mathrm{Re}\left\{ \mathbf{Y}_{ij} \left(\left| V_i \right|^2 + \left| V_j \right|^2 - V_i^{*} V_j - V_j^{*} V_i \right) \right\} \tag{5.18}$$

With a total of L feeder portions, the total power loss function, f(**q**), of the entire system can be expressed in equation 5.19.

$$f(\mathbf{q}) = \sum_{k=1}^{L} P_{LOSS,k} \tag{5.19}$$

- **Steepest descent method for solving optimisation problems**

To find an optimal solution for the centralised reactive power control problem, an appropriate method has to be chosen. Although there are many techniques available, in this thesis only the method of the Steepest Descent Algorithm (SDA) has been applied.

The general form of an unconstrained optimisation problem [65-66] can be expressed initially as follows.

$$\underset{\mathbf{x} \in \Re^n}{\text{Minimise}} \qquad f(\mathbf{x})$$

where

x is a vector of control variables

f(**x**) is a continuously differentiable objective function on bounded sets.

$\Re^n$ is the set of n-dimensional real numbers

The simplest gradient search technique is the steepest decent algorithm. It involves finding a minimum with the search direction defined by its gradient. Briefly, the following is a step-by-step description of this method [65].

Step 0: Initialisation. Given an initial $\mathbf{x}_0 \in \Re^n$ and the counter i = 0

Step 1: Compute the search direction $\mathbf{p}_i = -\nabla f(\mathbf{x}_i)$, Terminate if $\|\nabla f(\mathbf{x}_i)\| < \varepsilon$

Step 2: Compute the step-length λ_i by solving the following sub-problem

$$\underset{\lambda_i > 0}{\text{Minimise}} \qquad F(\lambda_i) \equiv f(\mathbf{x}_i + \lambda_i \mathbf{p}_i)$$

Step 3: Update the current solution $\mathbf{x}_{i+1} = \mathbf{x}_i + \lambda_i \mathbf{p}_i$ and the counter i = i +1

Step 4: Repeat step 1

Where

$\mathbf{p}_i$ is the search direction

λ_i is the step-length

ε is maximum error allowance of the objective function

Graphically, a typical solution sequence resulting from the steepest descent method as it moves towards the minimum is depicted in Fig. 5.20. In addition, a general mathematical description for the optimisation techniques especially for the steepest descent method is written in Appendix C in more detail.

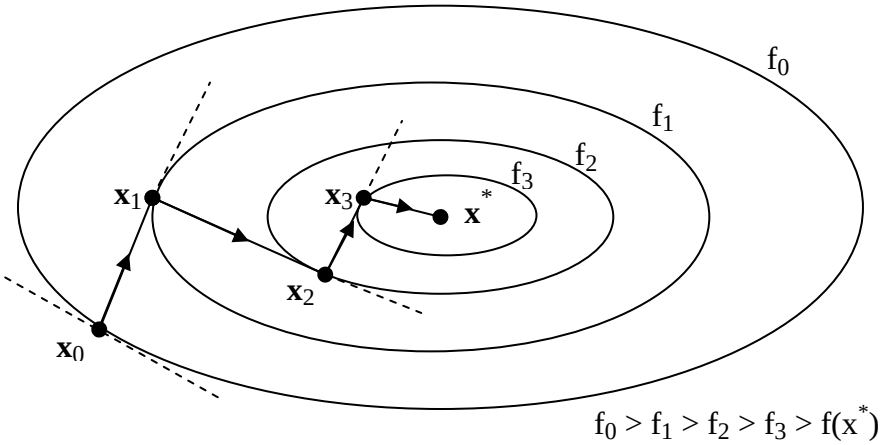


Fig. 5.20: Solution sequence resulting from the steepest descent algorithm [63]

- **Methodology**

At every time step, the optimisation problem is formed and solved for the reactive power commands to the PWM locomotives in order to minimise the power loss function. In the real application, the central computer would perform numerical calculations in real time based on input data received from all locomotives giving their current power consumption and positions. Thus, to evaluate the performance of the proposed real-time control of the locomotive reactive powers, a multi-train simulator is needed to simulate train movement and corresponding power consumption. This simulation is used to generate realistic real-time information (train positions and powers) acting as input signals to the control centre from the locomotives and thus tests the proposed control system. A general framework for the optimal area control can be summarised as shown in the flow diagram in Fig. 5.21.

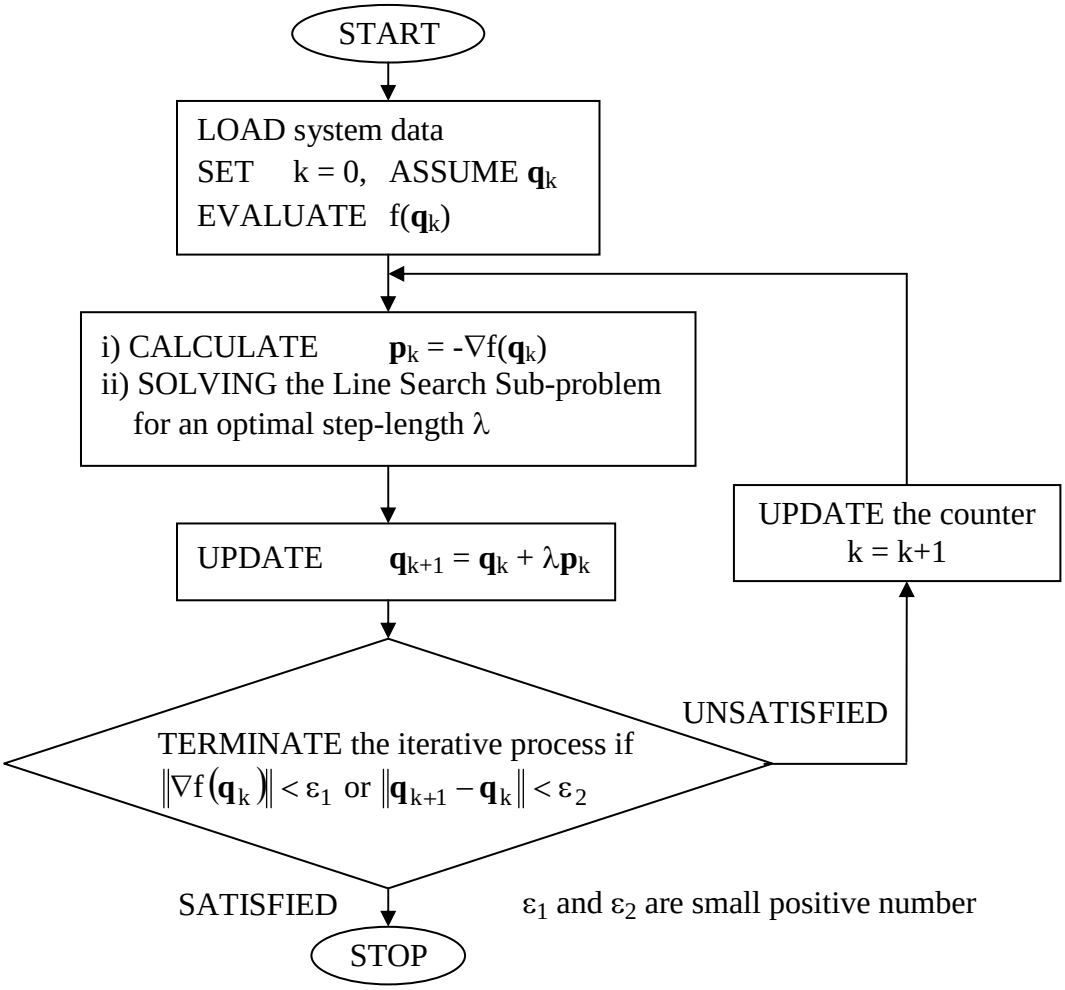


Fig. 5.21: Flow diagram for the AC railway power flow solution

In this chapter, the optimisation of the centralised reactive-power control problem for a particular time update is illustrated; discussion of tests with test data generated by the multi-train simulation is given in the next chapter.

5.5 Simulation Results and Discussions

Fig. 5.22 shows a four-train AC railway test system having 25-km feeding length. TR1 and TR4 are equipped with adequately designed PWM converters and can act as mobile reactive power compensators. By controlling these two trains the railway power flow optimisation problem can be solved. Full details for this test system can be found in Appendix E.4.

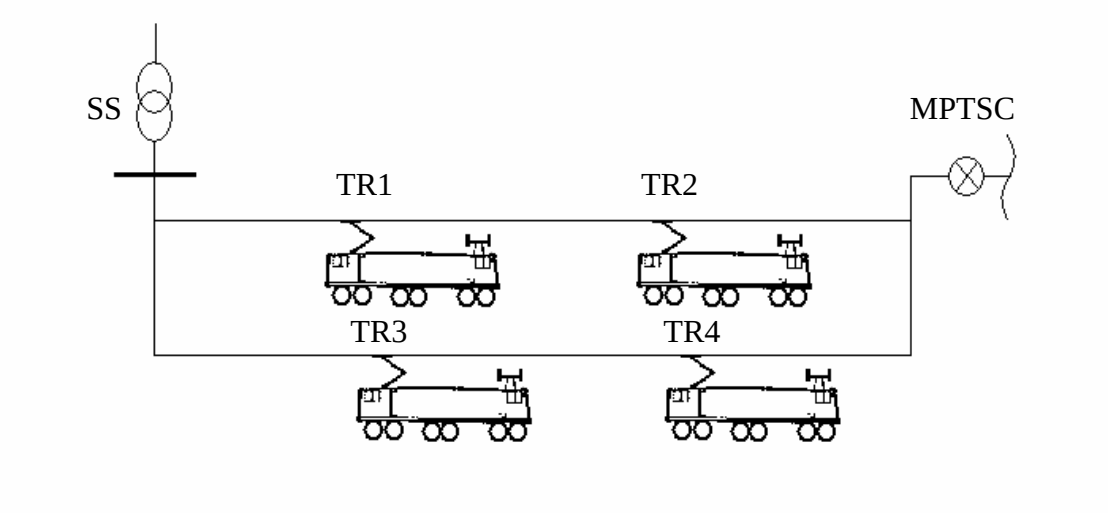


Fig. 5.22: Four-train AC railway test system

Applying the SDA, the optimal reactive power can obtained by a search through the reactive power space formed by the reactive power of the controlled trains (TR1 and TR4) as shown in Fig. 5.23. Starting with unity power-factor conditions, the search process is repeatedly performed until the solution converges. The present example only required 5 iterations to obtain the local minimum in which the maximum gradient magnitude is less than 1×10^{-4} W/Var as shown in Fig. 5.23. The results for this test are presented in Table 5.1.

Table 5.1: Test result for the four-train system by using the area control technique where 50% of all trains are PWM (50% test case)

Iteration	Reactive power (MVar)		Power losses (kW)	Max. Grad. (W/Var)
	TR1	TR4		
0	0.0000	0.0000	260.97	1.7847×10^{-2}
1	-1.6975	-2.0054	231.36	4.0784×10^{-4}
2	-1.9042	-1.8303	231.19	3.5505×10^{-5}

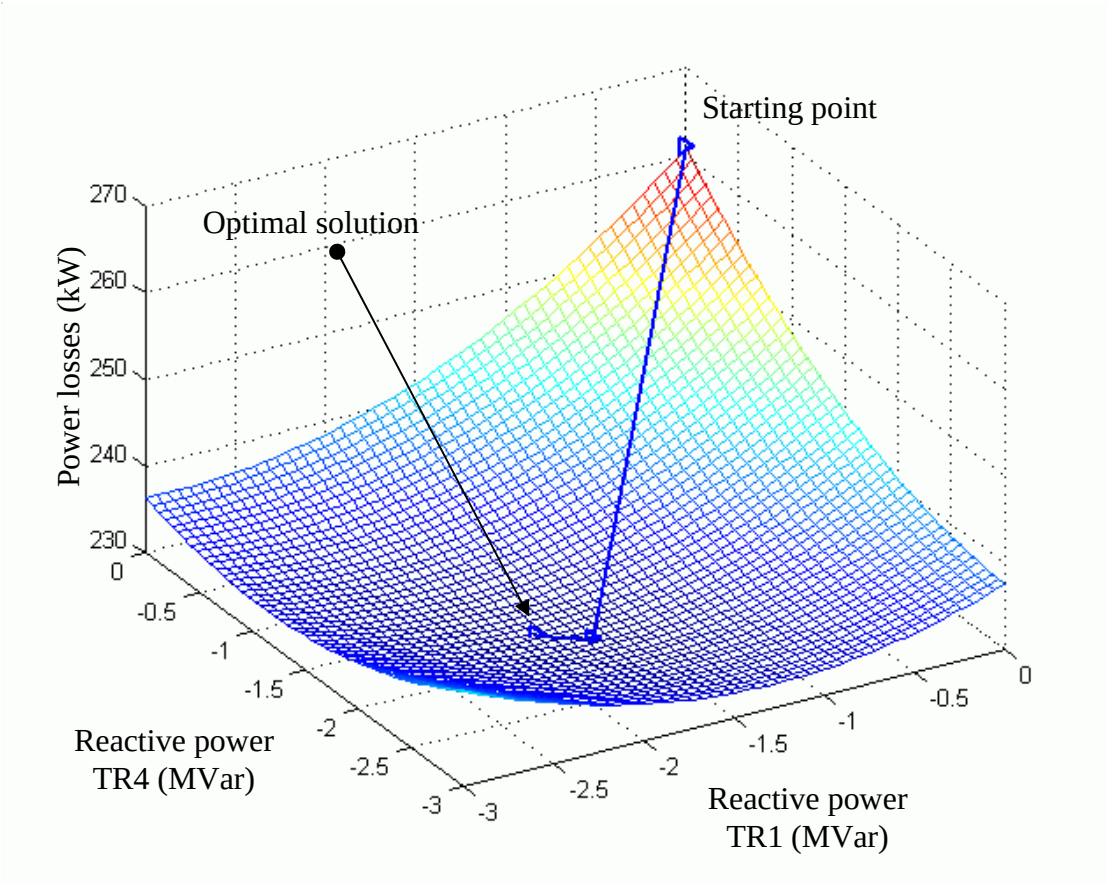


Fig. 5.23: Solution sequence towards the optimum (50% test case)

For comparative purposes, SVCs are considered installed in the system as means of compensating the reactive power. The substation and the MPTSC are the two locations where the SVCs are installed. For the tests, each PWM-converter-equipped train has -3 MVar as its maximum reactive power limit. The SVCs used for comparison are rated as 5 MVar for the first case (SVC1) and 3 MVar for the second (SVC2). The corresponding results for each case are shown in Tables 5.2 and 5.3. The maximum terminating gradient allowance for these two test cases is set as 1×10^{-4} W/Var.

Table 5.2: Test result for the four-train system (SVC1 test case)

Iteration	Reactive power (MVar)		Power losses (kW)	Max. Grad. (W/Var)
	SVC at the SS	SVC at the MPTSC		
0	0.0000	0.0000	344.55	3.7671×10^{-2}
1	-2.1212	-4.8838	242.45	2.3316×10^{-3}
2	-4.2052	-3.9786	239.57	7.9736×10^{-4}
3	-4.2594	-4.1032	239.51	5.1336×10^{-5}

Table 5.3: Test result for the four-train system (SVC2 test case)

Iteration	Reactive power (MVar)		Power losses (kW)	Max. Grad. (W/Var)
	SVC at the SS	SVC at the MPTSC		
0	0.0000	0.0000	344.55	3.7671×10^{-2}
1	-2.1212	-3.0000	251.21	1.0552×10^{-2}
2	-2.9752	-3.0000	247.02	8.9643×10^{-3}
3	-3.0000	-3.0000	246.92	8.9188×10^{-3}
4*	-3.0000	-3.0000	246.92	8.9188×10^{-3}

* Terminated because SVC limits reached.

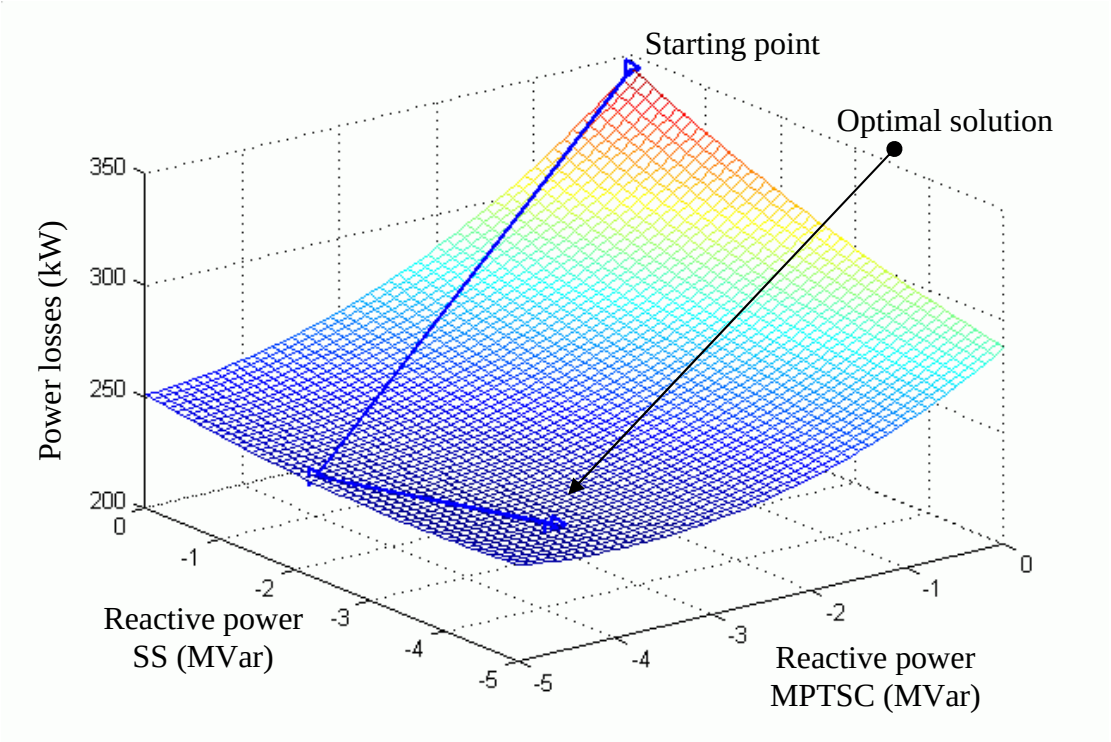


Fig. 5.24: Solution sequence towards the optimum (SVC1 test case)

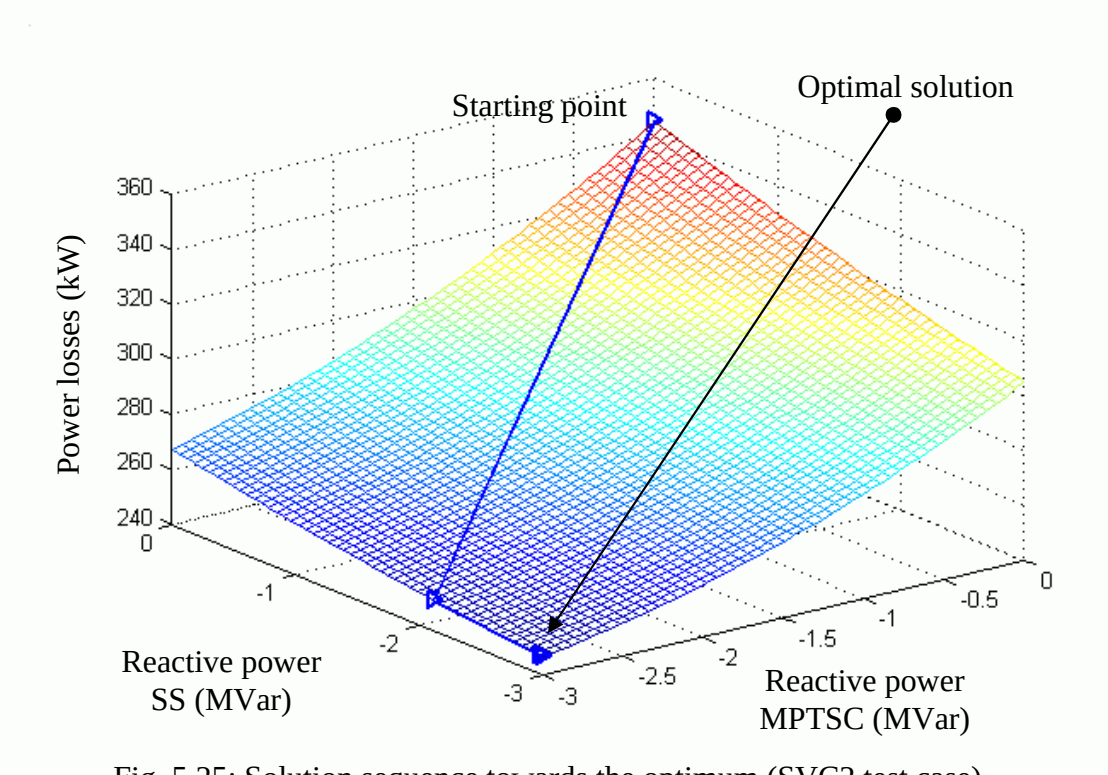


Fig. 5.25: Solution sequence towards the optimum (SVC2 test case)

With 3 MVar SVCs installed at the SS and the MPTSC, the generated solution sequence is limited by the reactive power constraints. Fig. 5.25 shows that the optimum is found as one of the extreme points. It means that the actual local minimum of the power loss function has not been found and the reduction in the power losses could be improved by increasing the reactive power rating for the SVC. As shown in Fig. 5.24, 5 MVar SVCs enlarge the solution space and the actual local minimum is founded without violating any constraint. This shows that installing any higher-rated SVCs, for example 10 MVar, will give no further improvement than installing the 5-MVar SVCs.

The fourth case presents the PWM locomotives operating at unity power factor (UPF) with SVCs at the Substation and the MPTSC. The test results are shown in Table 5.4 and Fig. 5.26, and show that the solution exists within the 3 MVar limits.

Table 5.4: Test result for the four-train system (SVC3 test case)

Iteration	Reactive power (MVar)		Power losses (kW)	Max. Grad. (W/Var)
	SVC at the SS	SVC at the MPTSC		
0	0.0000	0.0000	260.97	1.8624×10^{-2}
1	-1.2793	-2.6550	231.56	1.7812×10^{-3}
2	-2.8131	-1.9160	229.88	9.3453×10^{-4}
3	-2.8836	-2.0622	229.80	9.2518×10^{-5}

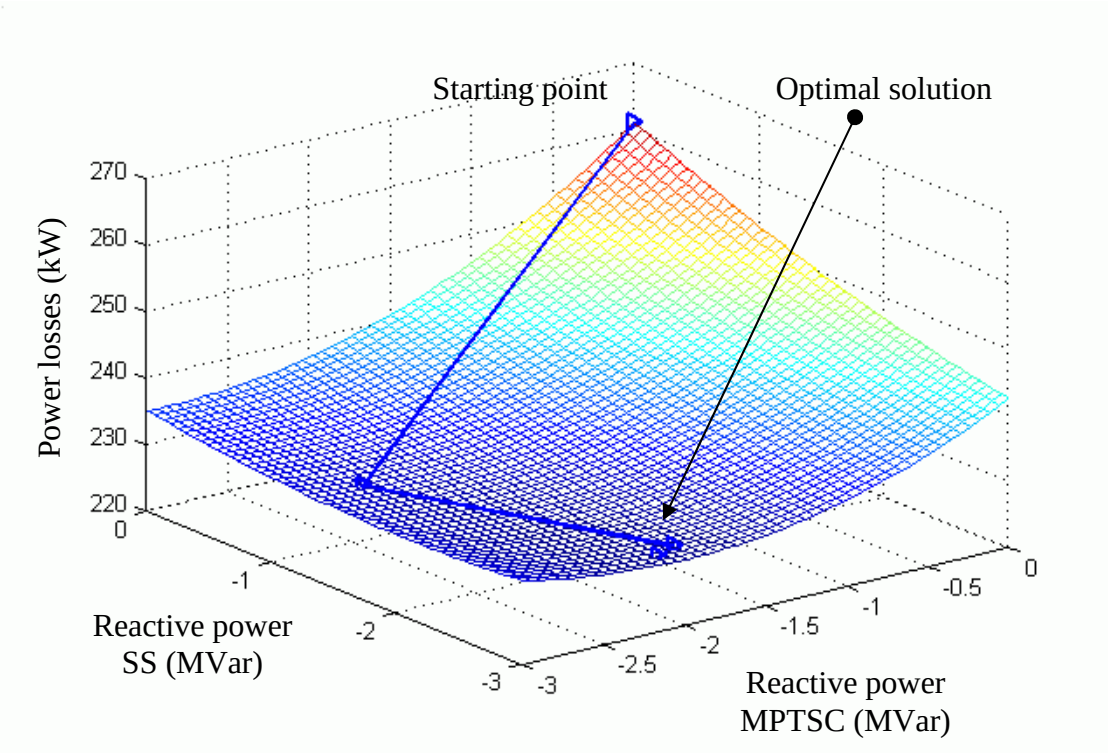


Fig. 5.26: Solution sequence towards the optimum (SVC3 test case)

At present, the older phase-angle controlled type locomotives are still in service with relatively few numbers of PWM locomotives yet. They will eventually be commissioned replaced by modern PWM locomotives. To complete this comparative study, the case where all trains are equipped with PWM converters is considered (100% test case). With control of up to four variables, the greatest reduction in power losses is obtained, as shown in Table 5.5.

Table 5.5: Test result for the four-train system (100% test case)

Iteration	Reactive power (MVar)				Power losses (kW)	Max. Grad. (W/Var)
	TR1	TR2	TR3	TR4		
0	0.0000	0.0000	0.0000	0.0000	226.14	3.6335×10^{-3}
1	-0.2360	-0.2434	-0.2306	-0.2432	224.46	4.1934×10^{-4}
2	-0.3962	-0.0472	-0.4871	-0.0409	224.32	3.4124×10^{-4}
3	-0.4198	-0.0679	-0.5035	-0.0602	224.31	6.5306×10^{-5}

Table 5.6 summarises the objective function values for each test case. In addition, two more test cases are added: i) all trains phase-angle controlled (Base case) and ii) all trains operating at UPF (UPF test case). The voltage profiles for all the test cases are depicted in Fig. 5.27.

Table 5.6: Summary of the power losses for each test case (Four-train test system)

	Test case						
	Base	SVC1	SVC2	SVC3	UPF	50%	100%
Power losses (kW)	344.55	246.92	239.51	229.80	226.14	231.19	224.31
% Reduction	-	28.3	30.5	33.3	34.4	32.9	34.9

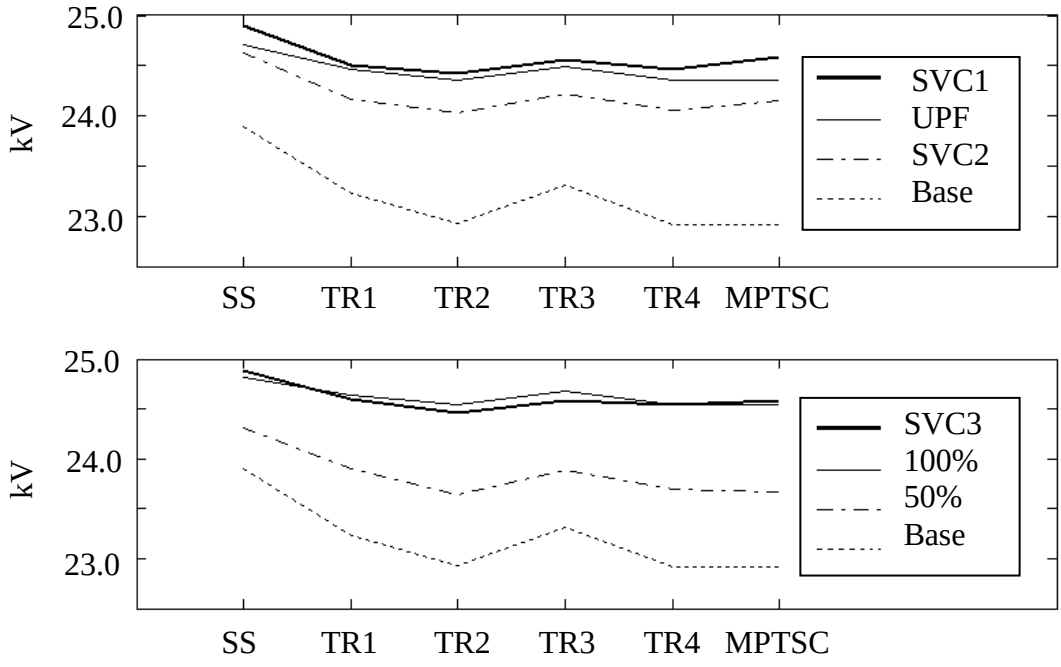


Fig. 5.27: Voltage profiles for all the test cases (Four-train test system)

The results show that the SVCs can help to reduce the power losses by 28% from the base case and that the voltage profile is also improved. The presence of the PWM locomotives can also

result in overall performance improvement. For the case of 50% of all the trains being PWM, the power losses are reduced by approximately 33%. The greatest reduction occurs when all the trains are PWM where 34% and 35% of the power losses are eliminated for the UPF and 100% test cases, respectively. The combination of the SVCs and 50% PWM trains reduces the power losses by just over 33%. This implies that the SVC may not be necessary when some trains are PWM and suitably controlled.

The four-train test system is rather small. Although it is not entirely representative of a practical system, it is quite appropriate to give an understanding of the area control concept and the optimal AC railway power flow problem. In practice, there will be several trains running in the same feeding section at the same instance. So the second test system is proposed to represent a denser AC railway system. The power feeding system is still the same as that shown in Fig. 5.22, but eight running trains are assumed. Full information for the second test system can be found in Appendix E.5. Six test case scenarios were simulated for this test. Note that the case of SVC1 is the use of the SVCs alone while the SVC2 is the use of the SVCs together with all PWM locomotives operating at unity power factor. For the 50% test case, TR1, TR3, TR6 and TR8 are PWM. Tables 5.7 – 5.12 present the test results. A summary is shown in Table 5.13. In addition, the voltage profile for each test case is presented in Fig. 5.28.

Table 5.7: Base case (Solving the power flow problem)

Power losses (kW)	817.16

Table 5.8: SVC1 (10-MVar SVC installed at the SS and the MPTSC)

Iteration	Reactive power (MVar)		Power losses (kW)	Max. Grad. (W/Var)
	SVC at the SS	SVC at the MPTSC		
0	0.0000	0.0000	817.16	7.6619×10^{-2}
1	-3.8805	-8.3101	478.78	5.7187×10^{-3}
2	-8.7427	-6.0406	461.99	2.3190×10^{-3}
3	-8.9094	-6.3976	461.49	1.9732×10^{-4}
4	-9.0832	-6.3164	461.46	9.4834×10^{-5}

Table 5.9: SVC2 (10-MVar SVC installed at the SS and the MPTSC with the help from the 50% of PWM trains)

Iteration	Reactive power (MVar)		Power losses (kW)	Max. Grad. (W/Var)
	SVC at the SS	SVC at the MPTSC		
0	0.0000	0.0000	533.77	3.5462×10^{-2}
1	-2.3305	-4.6065	437.79	3.7848×10^{-3}
2	-5.4437	-3.0314	430.42	2.1668×10^{-3}
3	-5.6125	-3.3650	429.97	2.4929×10^{-4}
4	-5.8220	-3.2590	429.93	1.5488×10^{-5}

Table 5.10: UPF (All PWM trains are operated at unity power factor)

Power losses (kW)	423.90

Table 5.11: 50% controlled case

Iteration	Reactive power (MVar)				Power losses (kW)	Max. Grad. (W/Var)
	TR1	TR3	TR6	TR8		
0	0.0000	0.0000	0.0000	0.0000	533.77	3.4777×10^{-2}
1	-1.4850	-2.0000	-1.8198	-2.0000	424.97	1.6780×10^{-3}
2	-1.8590	-2.0000	-2.0000	-1.8591	424.32	1.3637×10^{-3}
3	-1.9353	-1.9054	-2.0000	-1.6191	424.13	1.0784×10^{-3}
4	-2.0000	-1.9695	-2.0000	-1.6394	424.06	6.9786×10^{-4}

Table 5.12: 100% controlled case

Iteration		0	1	2	3
Reactive power (MVar)	TR1	0.0000	-0.2174	-0.6376	-0.6679
	TR2	0.0000	-0.2304	-0.3266	-0.3379
	TR3	0.0000	-0.2345	-0.0663	-0.0771
	TR4	0.0000	-0.2352	0.0000	-0.0117
	TR5	0.0000	-0.2174	-0.6376	-0.6679
	TR6	0.0000	-0.2304	-0.3266	-0.3379
	TR7	0.0000	-0.2345	-0.0663	-0.0771
	TR8	0.0000	-0.2352	0.0000	-0.0117
Power losses (kW)		423.90	418.19	417.24	417.21
Max. Grad. (W/Var)		6.4649×10^{-3}	1.1409×10^{-3}	5.5591×10^{-4}	2.3224×10^{-4}

Table 5.13: Summary of the power losses for each test case (Eight-train test system)

	Test case					
	Base	SVC1	SVC2	UPF	50%	100%
Power losses (kW)	817.16	461.46	429.93	423.90	424.06	417.24
% Reduction	-	43.5	47.4	48.1	48.1	48.9

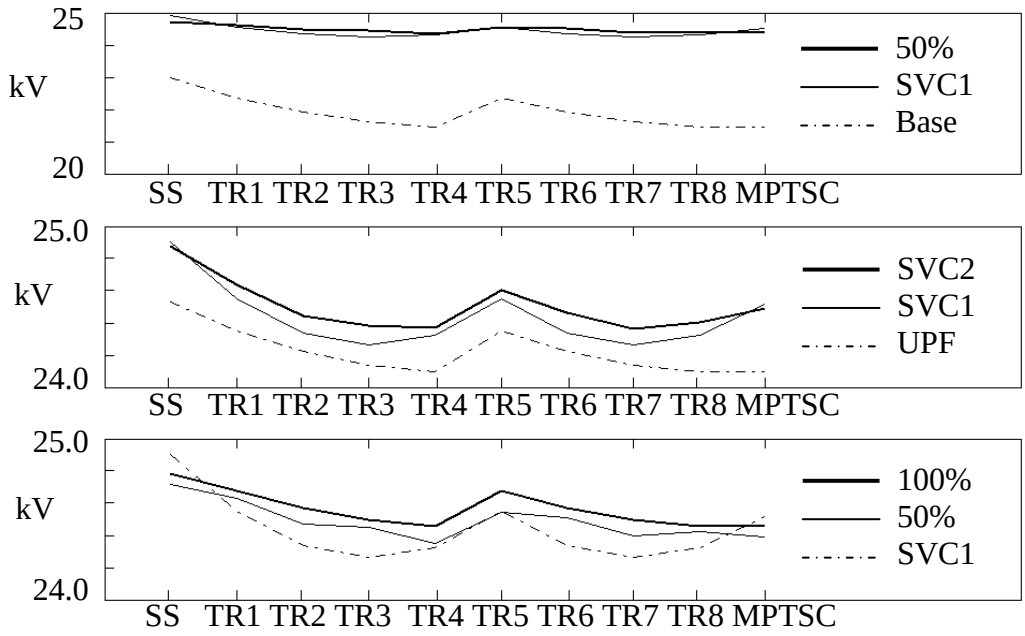


Fig. 5.28: Voltage profiles for all the test cases (Eight-train test system)

The area control technique works quite well even for the dense AC railway system. 48 – 49% of the power losses is eliminated for the cases of the area control system while the SVC cases can reduce the power losses by 44 – 47%. Noticeably, when the power losses is minimised, the voltage profile is improved.

This chapter has described reactive power compensation in AC railway electrification. Optimising reactive power flow in the entire feeding system can increase the power transfer capacity and the voltage magnitude [25,26]. This means that the optimised power feeding system can supply much more energy to the loads without any help from external reactive power sources. However, a new steady-state operating point resulting from the area controller needs a careful investigation of system stability to ensure that the system still operates within a safe region. A calculation for maximum power transfer and voltage collapse is thus now conducted. Consider the power feeding system as shown in Fig. 5.22. The worst case scenario will occur when all loads are connected to the system at the farthest point from the substation; that is at the MPTSC. Fig. 5.29 is a single-loop equivalent circuit representing the double-track,

25-km-length AC railway power feeding system. The investigation is performed by varying the real power and power factor of the load. The results, V-P curves, are shown in Fig. 5.30.

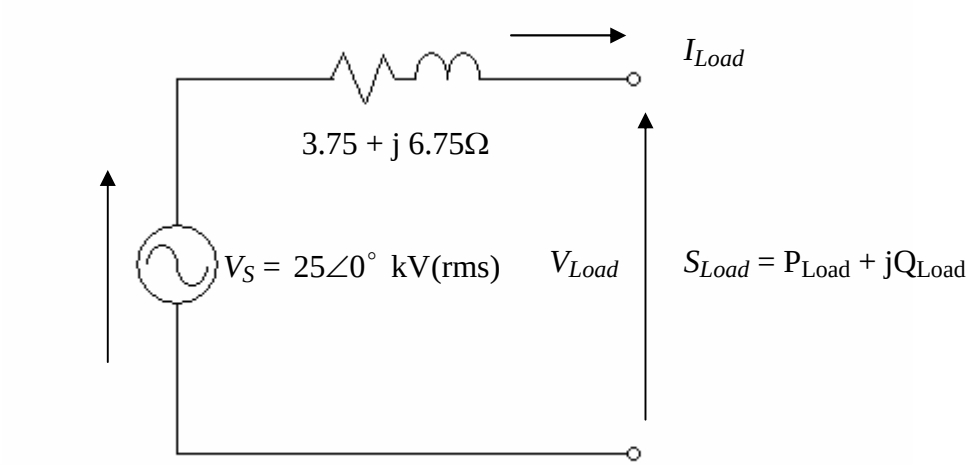


Fig. 5.29: Single-loop equivalent circuit

Each curve shows the voltage characteristic resulting from a constant-power-factor load. The number on each curve (0.6, 0.7, 0.8, 0.9, 1.0, -0.9 and -0.8) describes the corresponding power factor; a negative number refers to leading power factor. With poor power-factor operation such as 0.8 lagging, the maximum load that can be supplied by the power supply is just over 20 MW at the voltage magnitude of 13.3 kV. Compensating the excessive reactive power improves the overall power factor and, obviously, more power can be supplied. At unity power factor, the system can deliver a load power up to about 36 MW at a voltage magnitude of 15.8 kV. The line L is the division between stable and unstable operating regions, where the region above the line is declared stable while the region underneath the line is unstable. This confirms that the reactive power compensation improves the power transfer capability and the voltage profile.

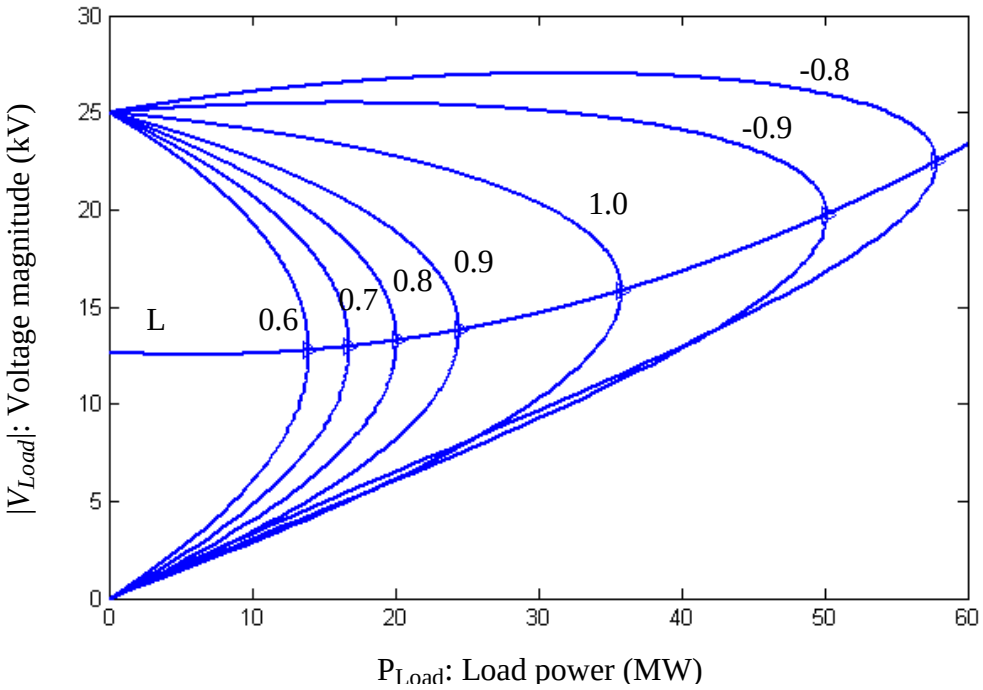


Fig. 5.30: V-P curves for the system shown in Fig. 5.29

5.6 Summary

This chapter presents the use of PWM locomotives as mobile reactive power sources to help the system reactive power shortage. The reactive power limit for the PWM locomotive with respect to various operating conditions is derived. Thus, the optimal AC railway power flow and the area control technique is developed and tested via two AC railway test systems. Furthermore, a calculation to evaluate the system stability is illustrated.

All moving loads in this chapter are assumed to be fixed in position. However, in the real world, the running trains change their position at every second. Hence, a full hour operation of train services is required to investigate the effectiveness of the optimal AC railway power flow control via the area control system. The simulation will be carried out in the next chapter.

CHAPTER 6

OPTIMAL AREA CONTROL SIMULATION

6.1 Introduction

The area control system was introduced in Chapter 5, using the concept of the locomotive reactive power co-ordination performing compensation of the surplus reactive power for the entire feeding section. Compared with the bulky and expensive SVC, it has been shown that the power loss reduction and the system voltage profile is noticeably improved. However, the tests in Chapter 5 consider only one particular time update. In fact, trains are moving loads and the train's power consumption from the supply conductor is dependent on the train position, mode of operation, etc. In the real world application, the developed area control system has to be able to perform compensation of the reactive power in any normal operating condition. Therefore, study of the real-time area control system operation is necessary to evaluate the effectiveness of the optimal area controller during typical train services. In this chapter, a multi-train simulator is developed and used to simulate realistic real-time information (train positions and powers) acting as input signals to the central controller from the locomotives or trains. With the combination of the multi-train simulator and the central controller, an overall area control system simulator is created.

6.2 Area Control Simulation with Discrete Time Update

The central controller in the control centre is a high-performance digital computer able, accurately and precisely, to perform the necessary simulation and optimisation within a particular time update. Although, in practice, communication hardware between the central computer and the on-board controllers is required, the multi-train simulator will be used to

model the trains and any such communication systems in this thesis. The interaction between the central controller and the multi-train simulator can be demonstrated as follows.

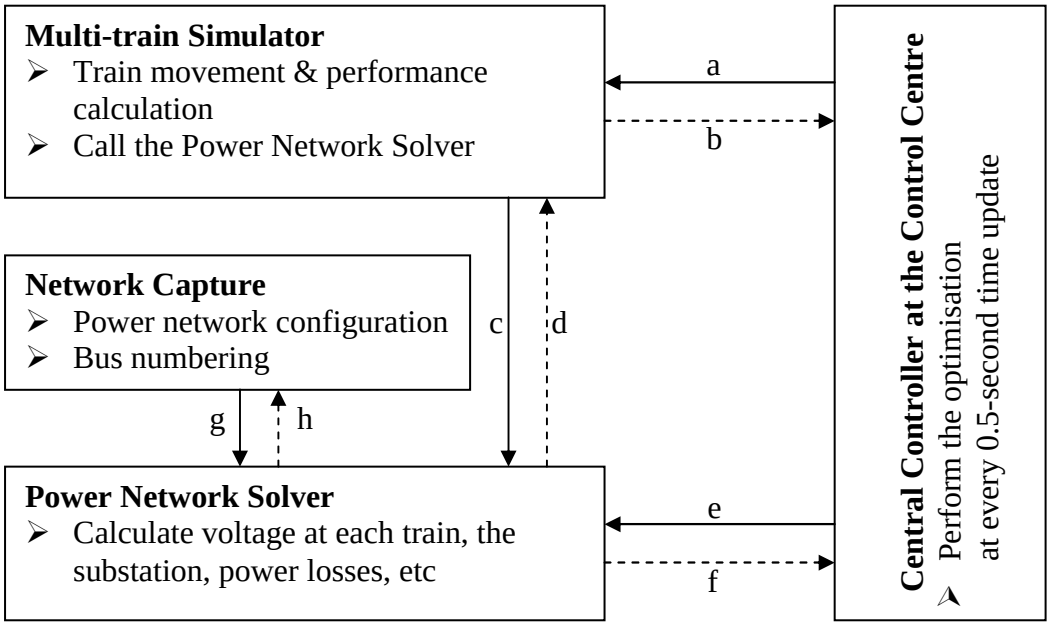


Fig. 6.1: Program structure of the optimal area control system simulator

In Fig. 6.1, there are four main sections: i) Central controller, ii) Multi-train simulator, iii) Network capture and iv) Power network solver. At each time update, the multi-train simulator is used to simulate position and power consumption of all trains. The change in train positions and powers causes voltage variation in the power feeding system. The multi-train simulator calls the power network solver (Solid line c) for voltage calculation. The voltage solution obtained by the power network solver is sent back to the multi-train simulator (Dashed line d) and will be used to evaluate the train performances for the next time update (if required). Then, all train positions and powers are sent to the central controller (Dashed line b). This information will be used in the optimisation process. While optimising the system objective, the power losses in the case under study, the power network solver will be called repeatedly to evaluate the objective function and voltage distribution (Solid line e). The power losses and voltage information is transferred back to the main processor (Dashed line f). When a local minimum is found by the main controller, the reactive power commands are sent out to the locomotives (Solid line a).

The network capture will be called every time the power network solver is processed (Lines g and h). This repetitive process will be performed until the stop time is reached.

In the present study, all the simulation modules described above have been developed for testing in a single computer. They are summarised in the following sections.

6.3 Multi-train Simulation

Train movement calculations are based on the well-known equations of motion, taking account of gradients, speed restriction, mode of operation, etc [68]. The net force applied to accelerate a train is $F = M_{eff}\alpha$, where F is the net force, M_{eff} is the effective mass and α is the train acceleration. With a small time update, e.g. 0.5 s, train performances can be calculated according to the mode of operation such as accelerating, coasting, braking etc. During a journey between two adjacent passenger stations, the operating mode is determined by reference to the speed control strategy, e.g. look-ahead speed control, ATO look-up table speed control, hysteresis speed control or so-called bang-bang type speed control, proportional speed control, etc [50]. The energy-minimisation principle, such that described in [69], is not implemented. In this section, the basic modules within the multi-train simulation are briefly reviewed.

6.3.1 Speed control strategies

The speed control model determines the modes of operation. Typically, there are three operating modes for the train: i) Accelerating mode, ii) Braking mode and iii) Coasting modes. The tractive effort provided is dependent on the speed control strategy applied. In this thesis, two types of speed controllers are reviewed.

- **Hysteresis speed control**

This model is quite simple. It corrects a train speed error by applying the full tractive effort when the speed is less than a set value and applying the full braking effort when the speed

is greater than the speed limit as shown in Fig. 6.2. The difference between the upper and lower bounds provides some hysteresis to prevent the train changing modes too frequently.

- **Proportional speed control**

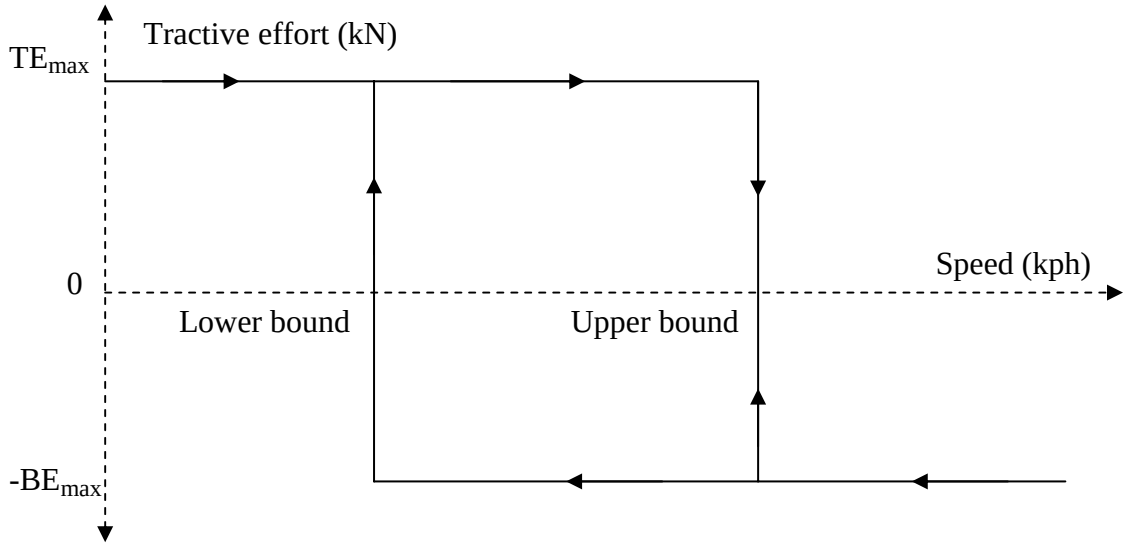


Fig. 6.2: Hysteresis speed control model

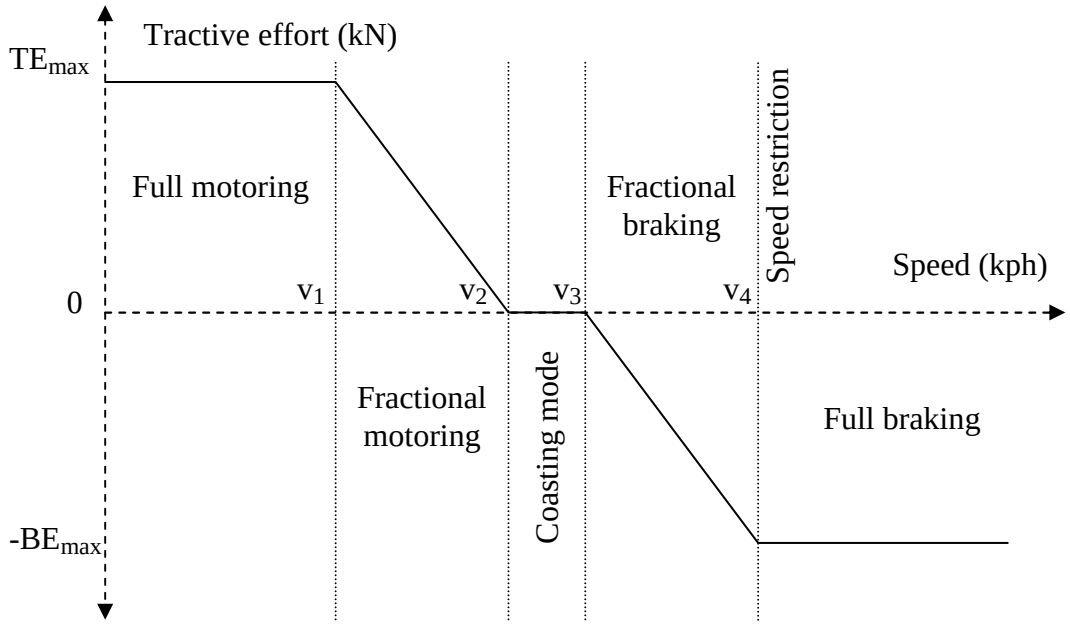


Fig. 6.3: Proportional speed control model

As shown in Fig. 6.3, full tractive effort is applied when the train speed is less than v_1. The fractional tractive or braking efforts will be calculated according to the speed is in the

motoring or braking regions ($v_1 - v_2$ or $v_3 - v_4$). The values of v_1 to v_4 can be defined by the operator.

6.3.2 Train movement and performance calculation

Suppose that a train of mass M_{eff} is on a slope making an angle θ to the horizontal as shown in Fig. 6.4, where M_{eff} is the effective mass [68]. The train motion is opposed by various forces, e.g. aerodynamic drag, track gradient force, etc. By applying Newton's second law, the train movement equation is expressed in equation 6.1, where α is the train acceleration.

$$TE - F_{grad} - Fdrag = M_{eff}\,\alpha \tag{6.1}$$

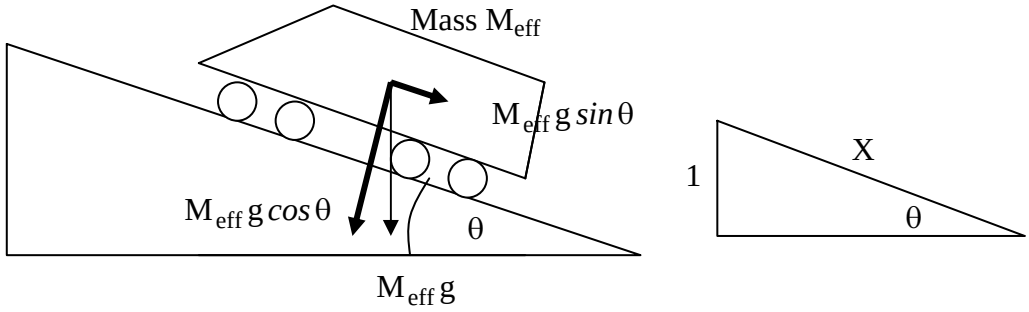


Fig. 6.4: Resolution of Mass force on a gradient

The aerodynamic drag is difficult to predict from calculations, however based on measurements from run-down tests where the natural deceleration of a train on straight, level track on a windless day is measured the drag force can be characterised [68,70,71]. Different operators have their own favourite equation to fit the train resistance. The Davis equation [71] as equation 6.2 is commonly used.

$$F_{drag} = a + bv + cv^2 \tag{6.2}$$

Where a, b and c are drag-force coefficients and v is the train speed. It is common to use different values for open or tunnel situations [70].

To push a heavy train up slopes requires substantial force. Gradients on railways are small and usually expressed in the form of 1 in X, where X is the horizontal distance moved to

rise 1 unit, see Fig. 6.4. With a small arbitrary angle θ the gradient force can be approximated by using equation 6.3.

$$F_{grad} = M_{eff}\, g\, sin\theta = \frac{M_{eff}\, g}{X} \tag{6.3}$$

When accelerating a train, there are both linear and rotational movements. The total mass (tare mass plus passenger or freight mass) is accelerated in linear sense while the wheel sets, gears and motors are involved rotationally. It is usual to express this rotational 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 as equation 6.4.

$$M_{eff} = M_{tare}(1+\tau) + M_{psg} + M_{frgt} \tag{6.4}$$

where

M_{tare} is the tare mass

M_{psg} is the passenger mass

M_{frgt} is the freight mass

τ is the rotary allowance

Traction drive equipment provides the tractive effort. The power being developed is the product of the tractive effort and speed. There are some key factors that result in variations of train performance. In the UK system, during train running, the pantograph voltage can vary from 16.5 kV to 27.5 kV with a maximum fault level at the pantograph of 330 MVA, 25 kV [9]. This variation causes a change of the tractive effort as a function of the line voltage applied to traction motors, especially in DC railways. The tractive effort curves resulting from line voltage variation are shown in Fig. 6.5, where three traction control regions [72-75]: i) Constant tractive effort, ii) Constant power and iii) Reduced power are evident. In AC railways where traction motors are fed through an AC/DC power converter with DC link voltage regulation, the train performance is less dependent on the pantograph voltage. Thus, in the present study, tractive

effort variation with line voltage has not been included in the modelling. This is reasonable as one of the main objectives of the central controller is precisely to maintain a good voltage profile. However, if it became apparent that voltages lower than 19.5 kV, say, were present, it would be necessary to refine the simulation to include voltage dependent tractive effort models as shown in Fig. 6.5.

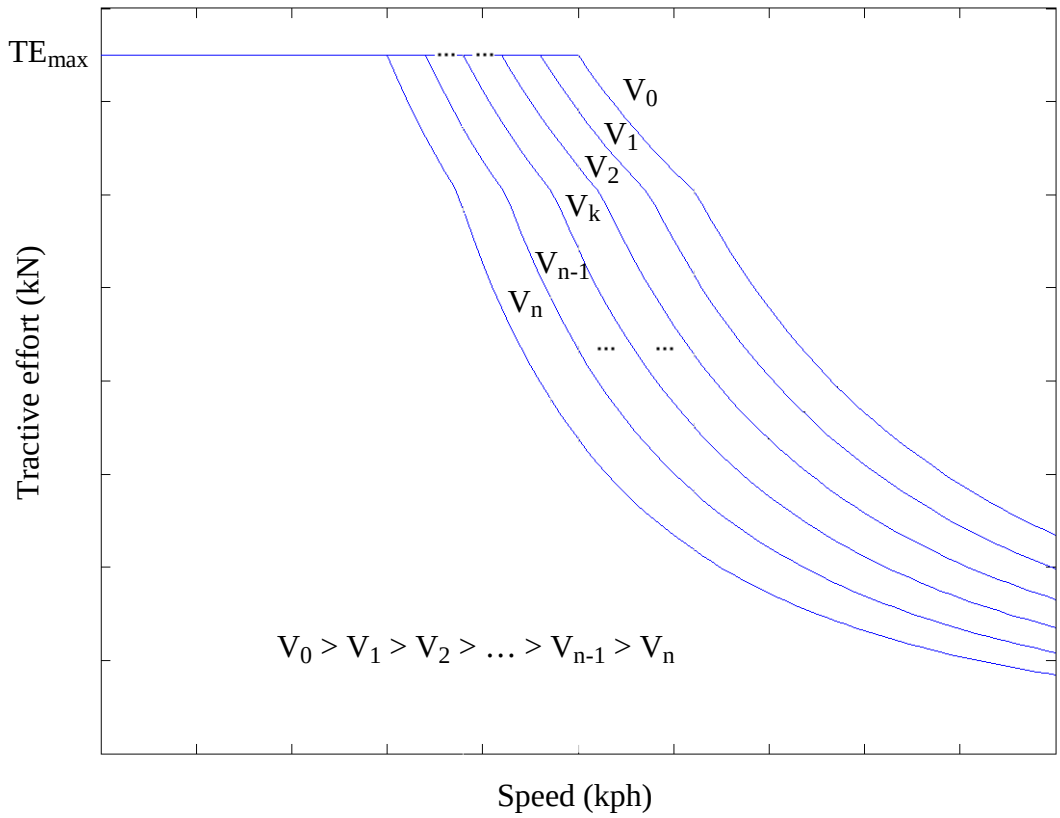


Fig. 6.5: Tractive efforts resulting from the line voltage variation

6.3.3 Power consumed by a train

Power consumed by a train corresponding to tractive effort TE and instantaneous speed v is given by the following expression

$$P = \frac{TE \times v}{\eta} \tag{6.5}$$

Where η denotes the efficiency of conversion of electrical input power to the mechanical output power at the wheels.

6.3.4 Speed and position update

Once the train acceleration is obtained, the speed and position of the train is calculated by the following equations,

$$v_t = v_i + \alpha \Delta t \tag{6.6}$$

$$d_t = d_i + v_i + \frac{1}{2}\alpha \Delta t^2 \tag{6.7}$$

where

v_i and v_t are the initial and terminal speed

Δt is the time step, 0.5 s in this thesis

d_i and d_t are the position before and after updated

6.3.5 Summary of train movement calculation

The train movement calculation can be summarised as follows.

Called from the main program

For track = First_Track to track = Last_Track

For Train = First_Train_InService to Train = Last_Train_Inservice

i) Use the speed (and pantograph voltage if necessary) from the previous time update to evaluate the tractive effort (Section 6.3.2) and thus the power consumption (Section 6.3.3)

ii) Determine the gradient, drag forces, etc and then compute the train acceleration (Equations 6.1 – 6.4)

iii) Update the train speed and position (Section 6.3.4)

End

End

Call the power network solver to update the pantograph voltages

Return to the main program

6.3.6 Numerical examples

A typical EMU (Electric Multiple Unit) is the BR class 319. Its tractive effort characteristic is shown in Fig. 6.6 [71,76]. By applying the hysteresis or proportional speed controls as described in Section 6.3.1, the distance-time, speed-distance and power-distance curves for a single-train simulation is shown in Figs 6.7 and 6.8, respectively.

As can be seen from these two diagrams, the proportional speed controller gives a smoother response as the trains are near the speed limit. Thus, in all subsequent simulations, the proportional controller is used.

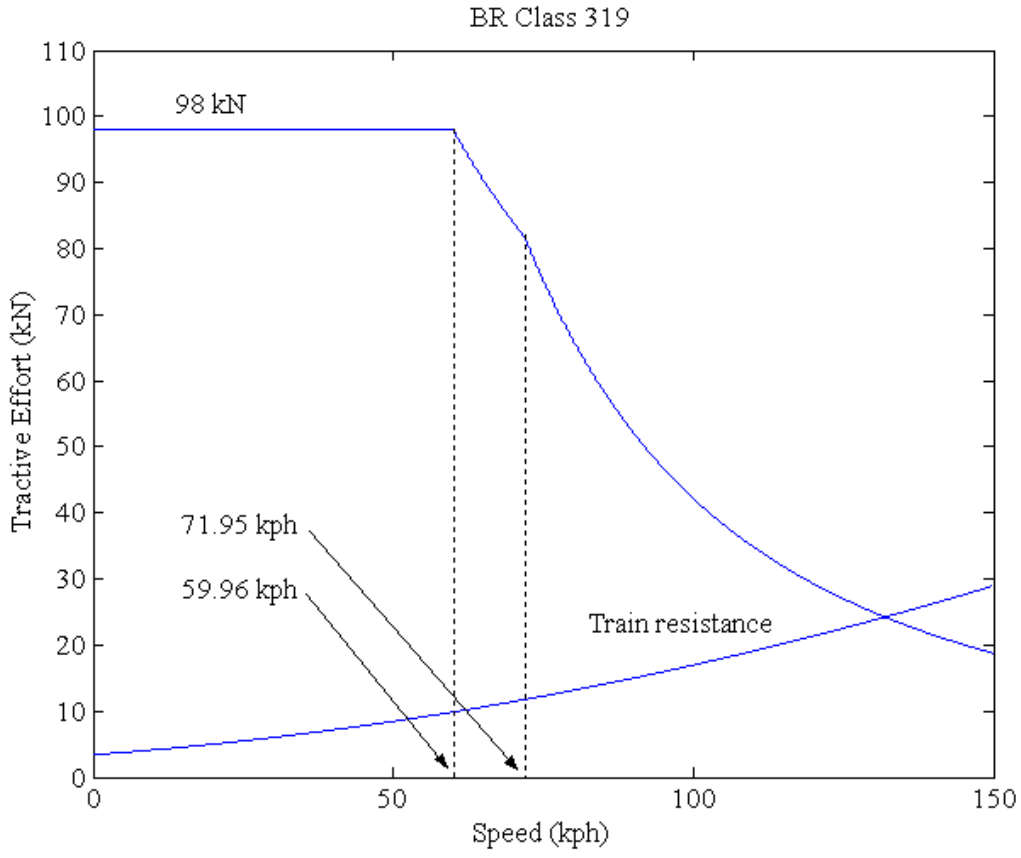


Fig. 6.6: Tractive effort and train resistance for the single-train simulation

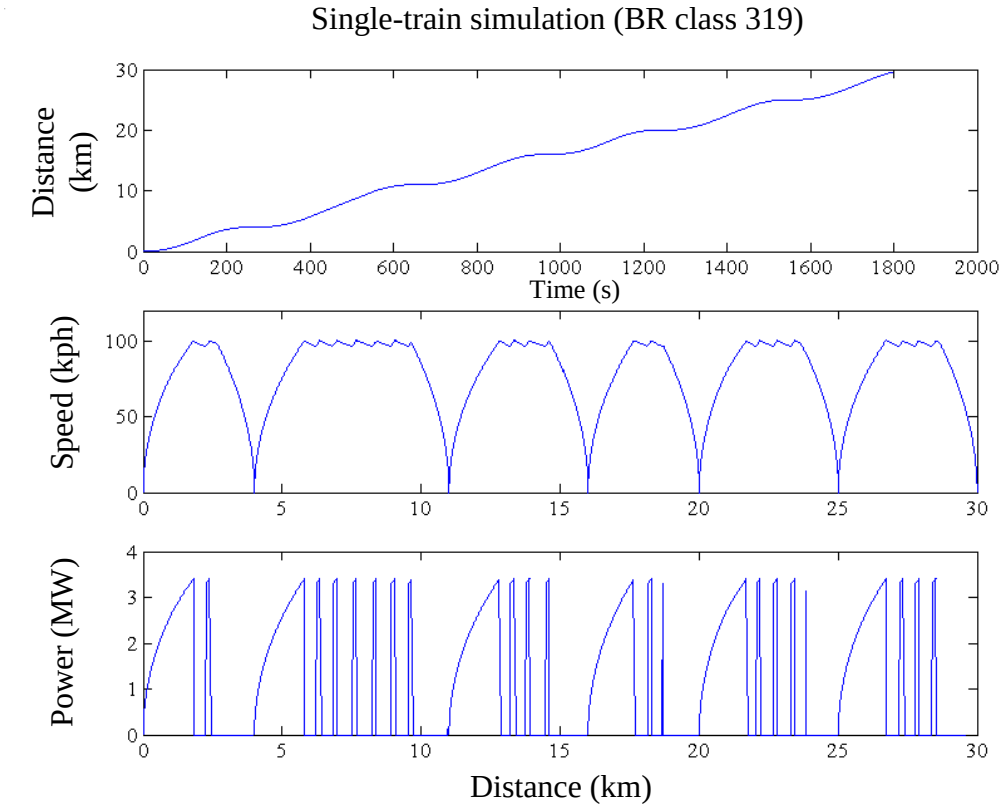


Fig. 6.7: Simulation results for the case of the hysteresis speed control

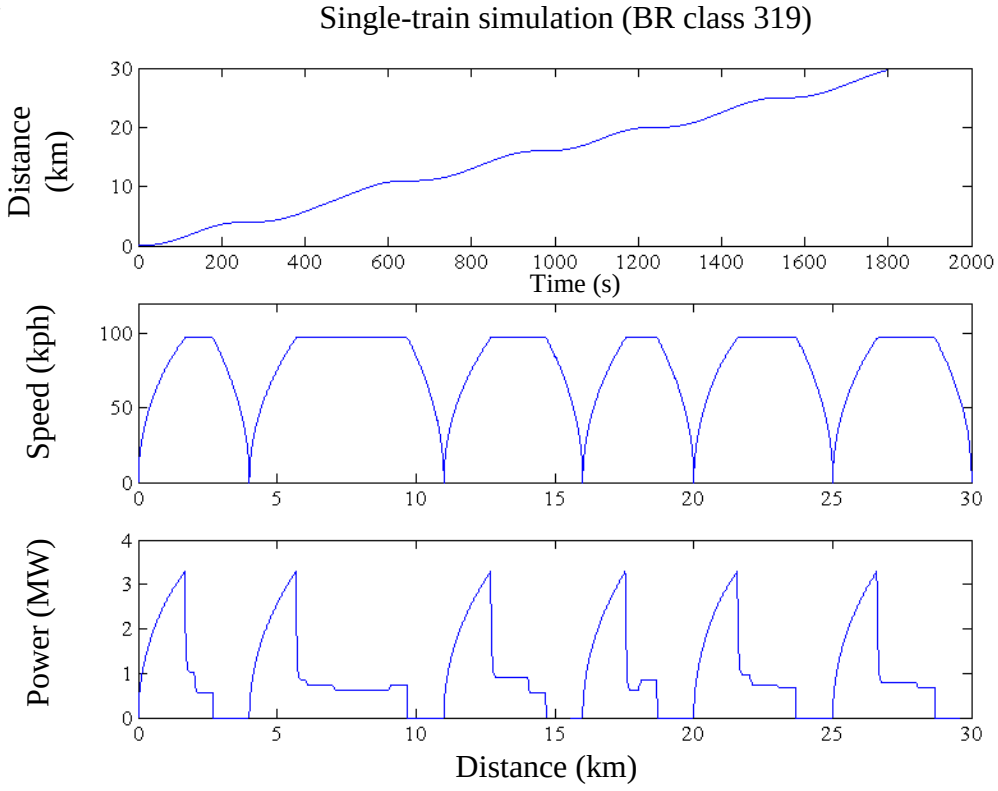


Fig. 6.8: Simulation results for the case of the proportional speed control

6.3 Power Network Solver and Network Capture

At each time update, the power network solver is called once by the multi-train simulator (lines c and d in Fig. 6.1) to calculate the voltage distribution of the entire power feeding section. This is normal procedure for the calculation of the pantograph voltage. However, to optimise the power losses, it needs to be called several times by the central controller (Lines e and f in Fig. 6.1). The power network algorithm and the method of solutions has been discussed already in Chapters 3 and 4. In this section, the interaction between the power flow module and the others is reviewed.

During the simulation, the multi-train simulator will simulate the train movement, e.g. train position, train power, etc. For example, Fig. 6.9 shows the numerical results of the multi-train simulator at one particular time update. As can be seen, six trains are running and each train power is calculated. This data, however, is not in a suitable form to be used by the power network solver. Bus numbers must be assigned to all the trains and substations. Thus, the power network solver transfers this data to the network capture module for defining bus numbers and making bus connection. Fig. 6.10 presents the printed on-screen results from the network capture. With this information, the power network is now ready to perform the power flow calculation. For clarity, all the numerical data in Fig. 6.10 is graphically presented by the equivalent circuit as shown in Fig. 6.11.

Train ID

1 2 3 Up-track
1 2 3 Down-track

Train position (m)

27321.72	17825.71	8541.86	Up-track
2387.36	11713.12	21165.15	Down-track

Train real power (W)

790134.65	614719.00	614372.81	Up-track
2901757.93	0.00	0.00	Down-track

Train reactive power (Var)

632107.72	491775.20	491498.25	Up-track
2321406.34	0.00	0.00	Down-track

Fig. 6.9: Results from the train movement calculation

Train bus information (Bus number)

```
10 6 2
3 7 11
```

Up-track
Down-track

Network data (Bus connection)

```
 1    1    2    8541.86
 2    2    4    1458.14
 3    4    6    7825.71
 4    6    8    2174.29
 5    8   10    7321.72
 6   10   12    2678.28
 7    1    3    2387.36
 8    3    5    7612.64
 9    5    7    1713.12
10    7    9    8286.88
11    9   11    1165.15
12   11   12    8834.85
```

// Column 1: Line number
// Column 2: From bus
// Column 3: To bus
// Column 4: Line length (m)

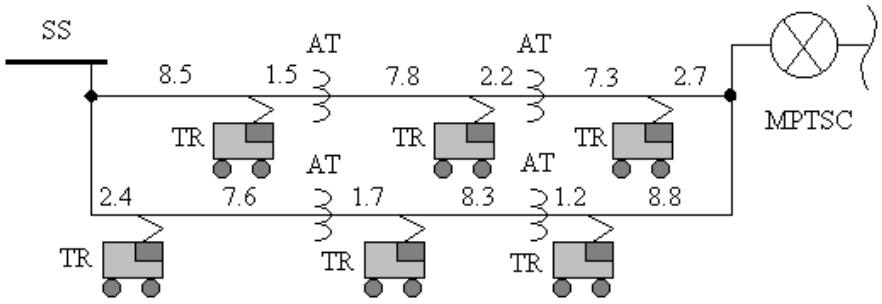
Total Number of buses = 12

Bus Information

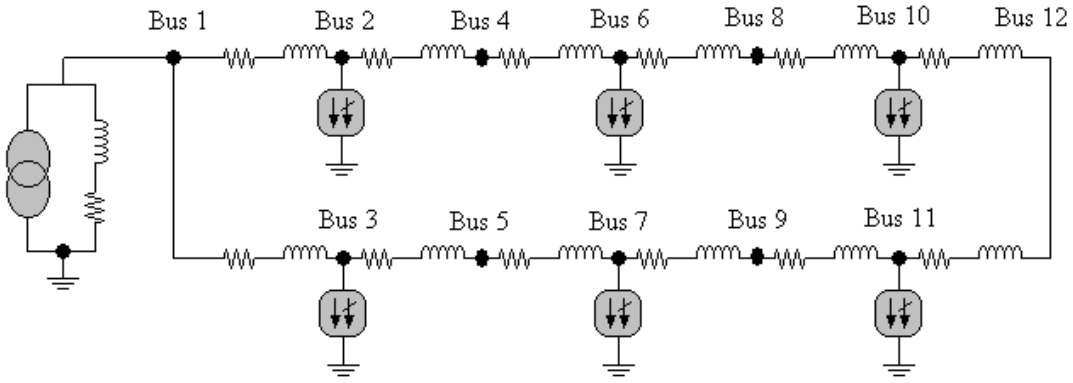
```
 1        0.00        0.00    0        0.00
 2   614372.81   491498.25    3     8541.86
 3  2901757.93  2321406.34    3     2387.36
 4        0.00        0.00    1    10000.00
 5        0.00        0.00    1    10000.00
 6   614719.00   491775.20    3    17825.71
 7        0.00        0.00    3    11713.12
 8        0.00        0.00    1    20000.00
 9        0.00        0.00    1    20000.00
10   790134.65   632107.72    3    27321.72
11        0.00        0.00    3    21165.15
12        0.00        0.00    2    30000.00
```

// Column 1: Bus number
// Column 2: Real power (W)
// Column 3: Reactive power (Var)
// Column 4: Bus type
0 is the substation bus
1 is an AT bus
2 is the MPTSC bus
3 is a train bus
// Column 5: Bus position (m)

Fig. 6.10: Results from the network capture



a) Schematic diagram with line lengths (in km)



b) Circuit diagram with bus numbers

Fig. 6.11: Graphical representation for the system given in Fig. 6.10

When the power flow optimisation is required, the central controller defines the optimum search steps and the power network solver is repeatedly called to evaluate the objective function at each iteration until the optimum is found.

6.5 Simulation Results and Discussions

A dense, suburban railway service with a 5-minute headway was modelled for the simulation tests as shown in Fig. 6.12. The trains were all assumed to be identical. The properties of the trains and the power supply used in this simulation are shown in Appendix E.6. Fig. 6.13 shows the distance vs time curves from these tests using the multi-train simulator described in Section 6.3. For more detail, the speed-time trajectory for train number 1 on the up-track is selected and depicted in Fig. 6.14.

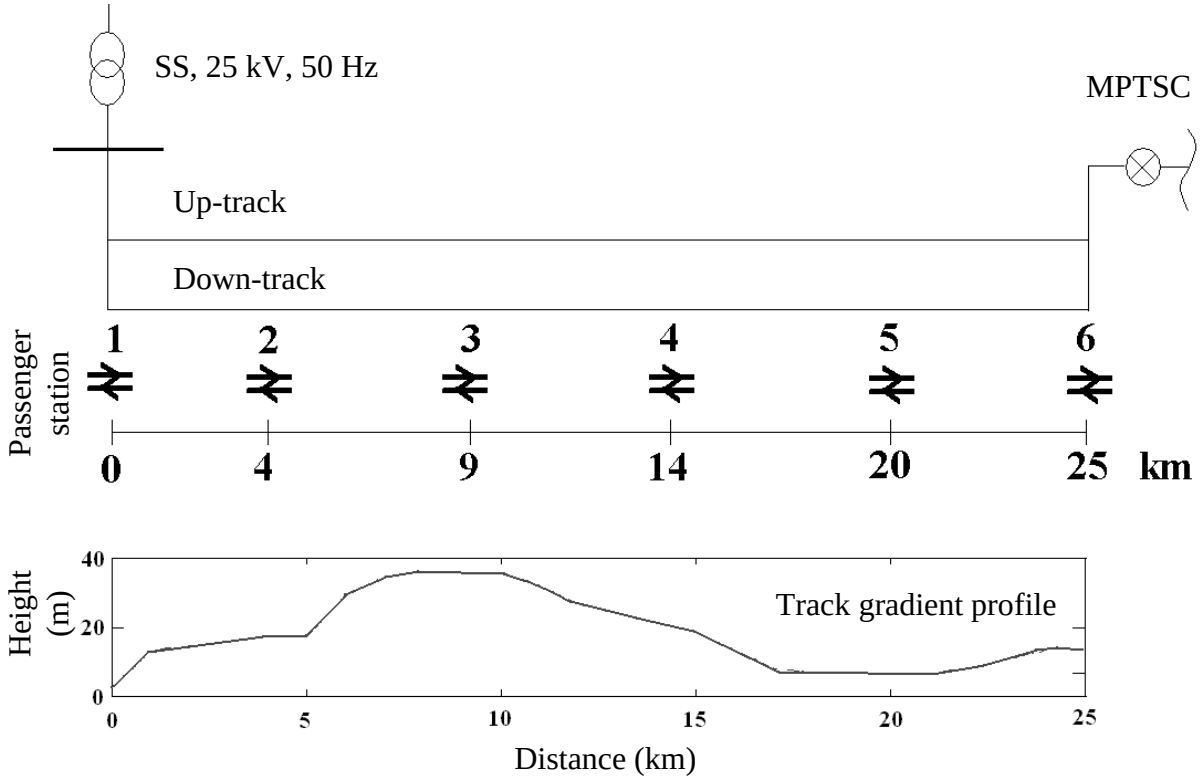


Fig. 6.12: Double-track AC railway test system

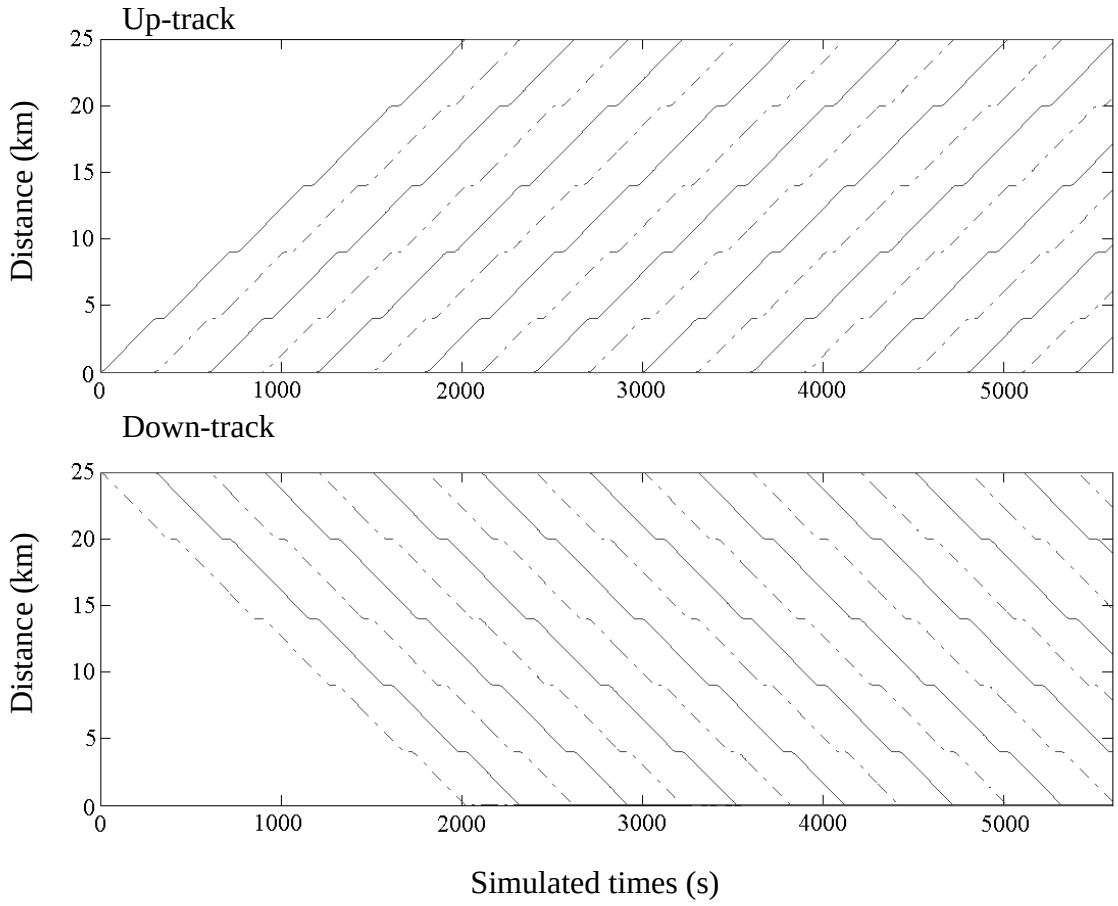


Fig. 6.13: Train position vs time curves (Solid line: Phase-controlled, Dash line: PWM)

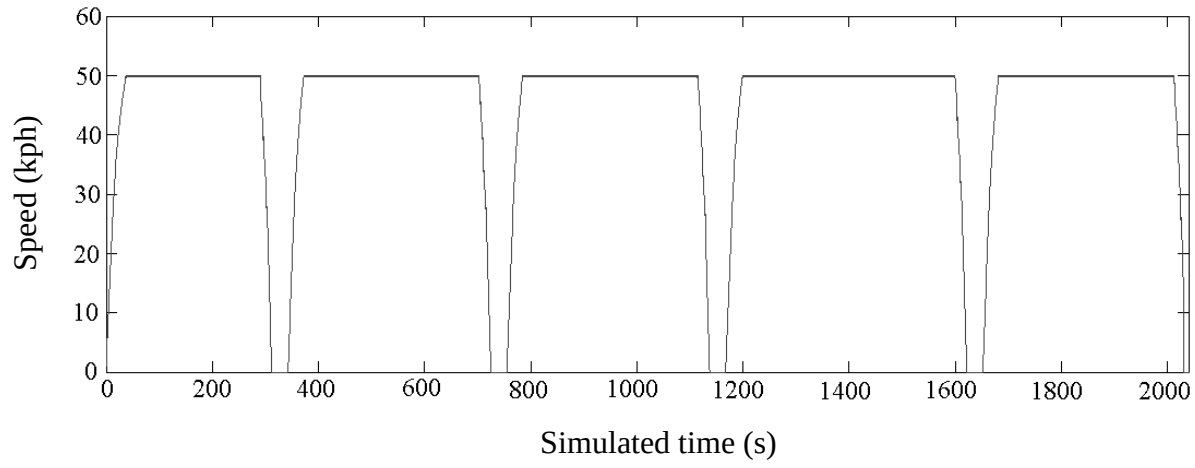


Fig. 6.14: Speed-time trajectory of train number 1 on the up-track

Table 6.1**:** Energy losses and RMSVD for an hour of typical operation

Case	Average RMSVD (kV)	Energy loss (MWh)	Energy loss reduction (kWh)	Energy loss reduction (%)	Maximum MVA drawn by the substation
1. Base	3.310	1.171	-	-	41.3
2. SVC	3.136	1.104	70.0	6.0	39.7
3. 50% UPF	1.931	0.734	436.9	37.3	33.2
4. 50% UPF + SVC	1.912	0.727	443.7	37.9	31.3
5. 100% UPF	0.742	0.560	610.5	52.1	27.5
6. 100% UPF + SVC	0.738	0.560	610.6	52.1	27.4
7. 50% OPF	0.429	0.557	614.0	52.4	27.5
8. 100% OPF	0.349	0.547	623.9	53.3	27.3
Total energy consumed by all trains				20.7 MWh	

The base case is where all trains in the system are phase-angle controlled. An alternative approach as mentioned previously is to use Static Var Compensators (SVCs), usually at the mid-section neutral gap (MPTSC). Such equipment is large and costly, but simulation results for this approach have been included here for comparison (see Table 6.1). The arrangement used is a single SVC at the MPTSC. The use of modern PWM converters at the front end to replace the older types offers the possibility of providing the reactive compensation with only a slight increase in the capital cost, and is therefore the case of principle interest in this study. Naturally, the introduction of equipment able to operate at unity power factor (UPF) improves the losses in itself, and thus the cases where 50% or 100% of the trains operate at UPF (50% UPF and 100% UPF)), but without the central optimal power flow control (OPF), are also included for comparison.

The main case of interest is that with 50% PWM locomotives and OPF; the case of 100% PWM with OPF is clearly relevant to railways where the whole fleet has been updated. There remains the possibility of combining UPF and SVC, although this is very unlikely to be economic. For completeness, therefore the cases of 50% and 100% UPF with SVC are included in Table 6.1. Results for the average root-mean-square voltage deviation (RMSVD), calculated

according to equation 6.8, are also given. Generally, the voltage profile improves as the losses reduce.

$$\text{RMSVD} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(V_{ref} - |V_i|\right)^2} \qquad \text{(kV)} \qquad (6.8)$$

where

V_{ref} is the nominal voltage, 25 kV for the test system

From Table 6.1, it can be seen that the mid-point SVC gives a modest reduction of 6% in the losses with the all phase-angle controlled fleet. Replacing 50% of the fleet with PWM trains operating at UPF gives a loss reduction of over 37% without the SVC; adding the SVC gives only a marginal further improvement. With 100% UPF equipment employed, the energy saving reaches over 52%. Adding the SVC gives no discernible improvement in energy saving, although the voltage profile does improve a little. In the case of principle interest, that with 50% PWM trains operating under the optimising central control, the energy saving is very similar at a little over 52%, but there is a further significant improvement in the voltage profile. This energy saving is noticeably better than the 50% UPF case. For completeness, the 100% OPF case is given, showing a small further improvement in both measures. However, the advantage is only marginal, so far as energy is concerned, compared to the 100% UPF case, i.e. without the central controller.

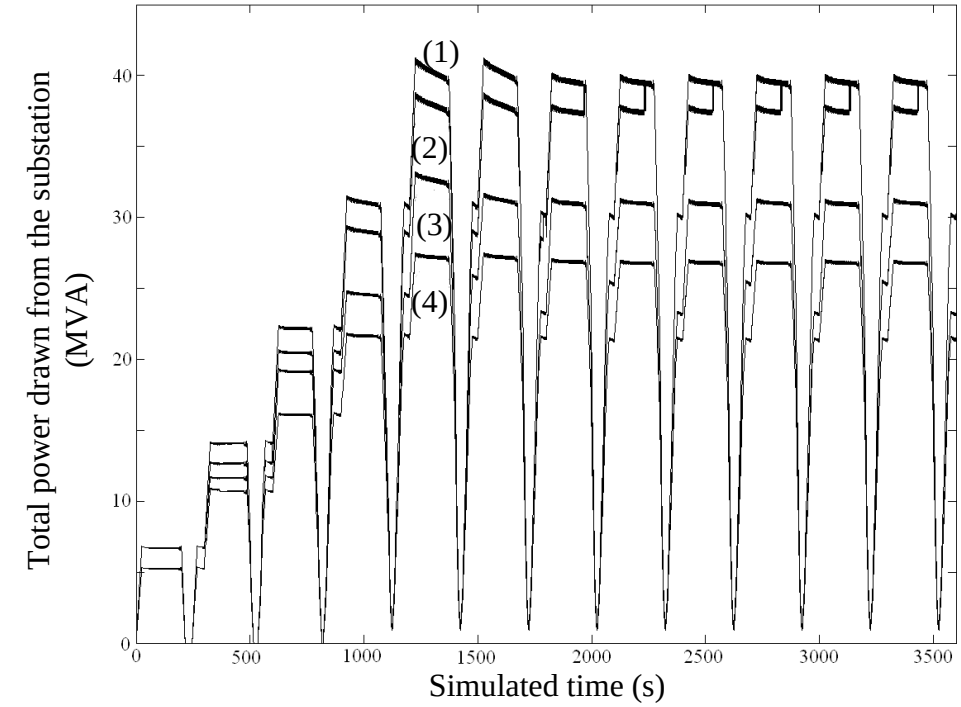


Fig. 6.15: MVA drawn from the substation (Cases 1 – 4)

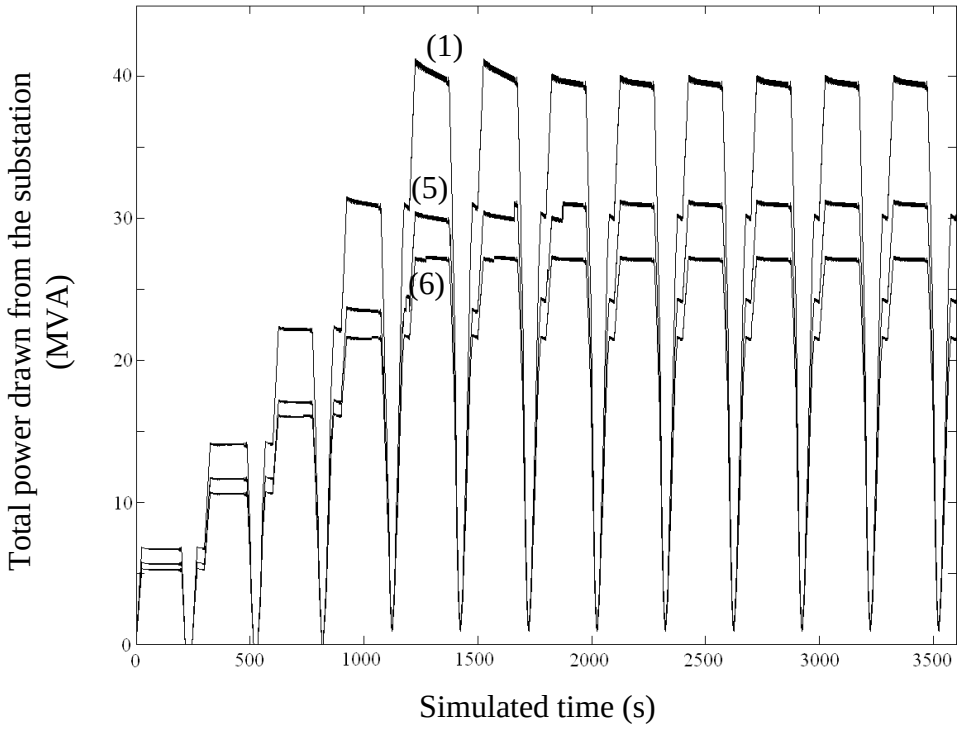


Fig. 6.16: MVA drawn from the substation (Cases 1, 5, 6)

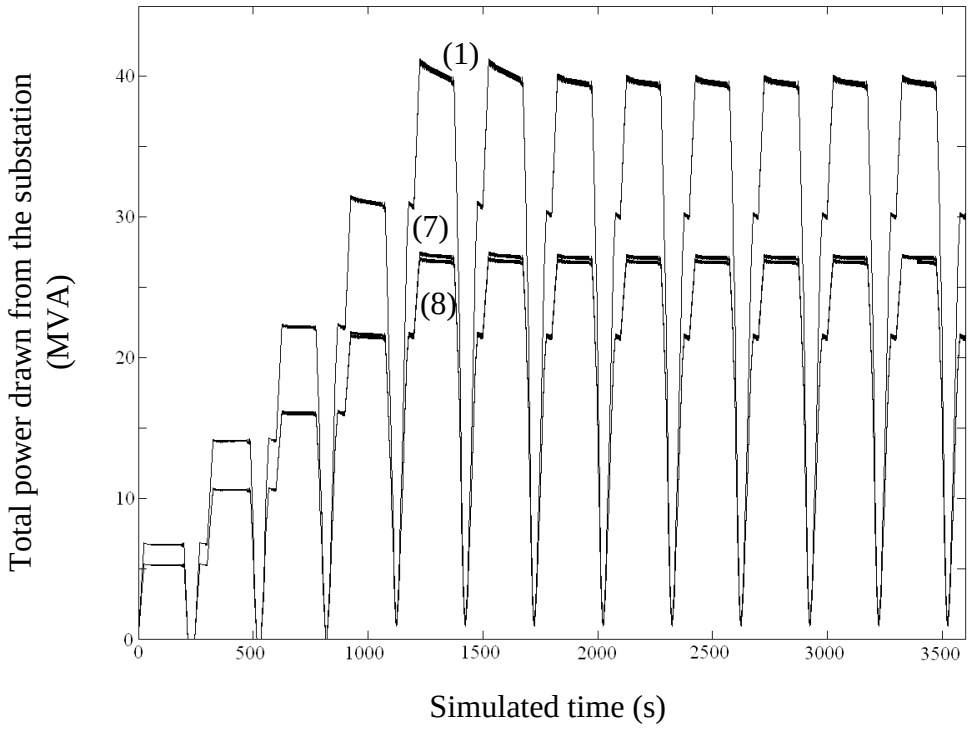


Fig. 6.17: MVA drawn from the substation (Cases 1, 7, 8)

The two key objectives of the optimal area control system are to increase power transfer capability and to improve the system voltage profile by control of the reactive powers on-board the trains themselves, obviating the need for extra reactive power sources to be installed elsewhere. In relation to the first of these objectives, the total MVA drawn from the substation for each case is presented in Figs 6.15–6.16 where the case numbers are as in Table 6.1. The maximum MVA drawn from the local supply grid by the substation are also shown in Table 6.1. This confirms that the system with the optimal area control is able to support the same train service as in the base case, but with a substantial reduction in the maximum MVA drawn from the supply grid. This would enable the power supply system to support denser train services without any upgrading. In relation to the second objective, the enhancement of system voltage profile, the pantograph voltages of train 1 on the up-track are compared in Fig. 6.18, again for the same cases as given in Table 6.1. The MPTSC voltages for each case are presented in Figs 6.19–6.20.

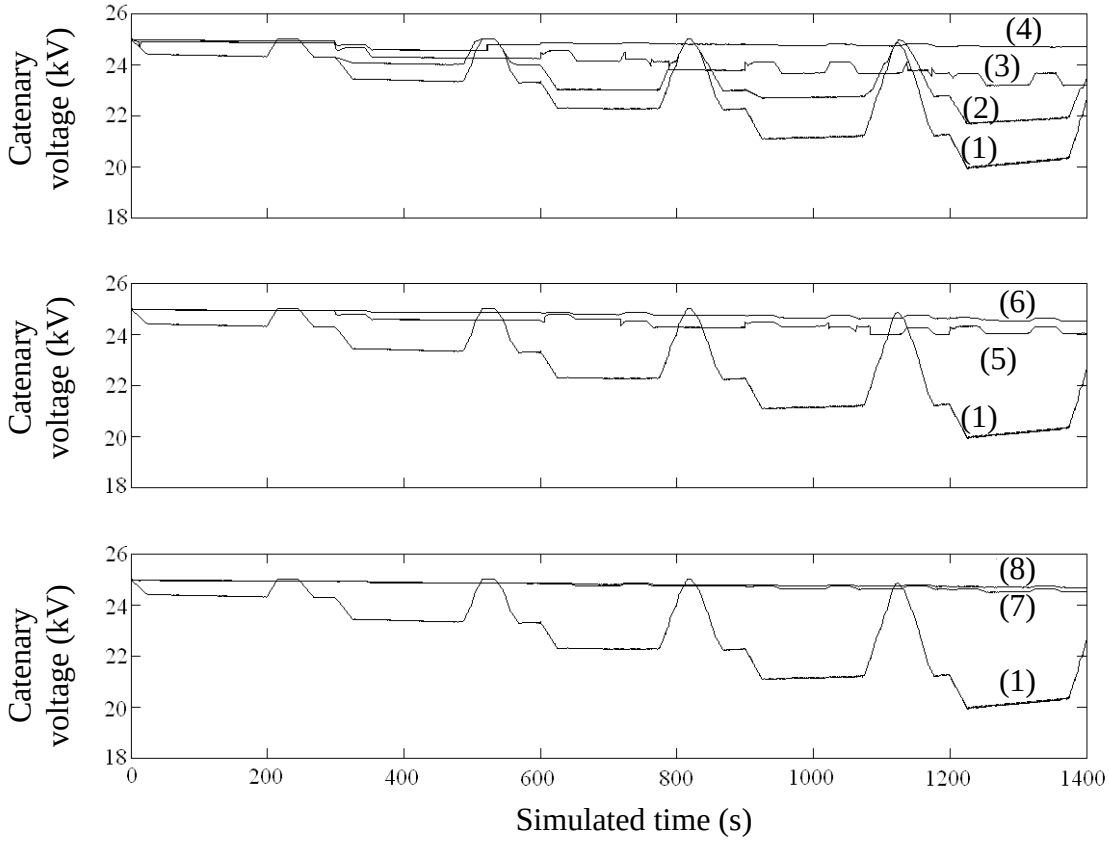


Fig. 6.18: Pantograph voltage of train no. 1 on the up-track

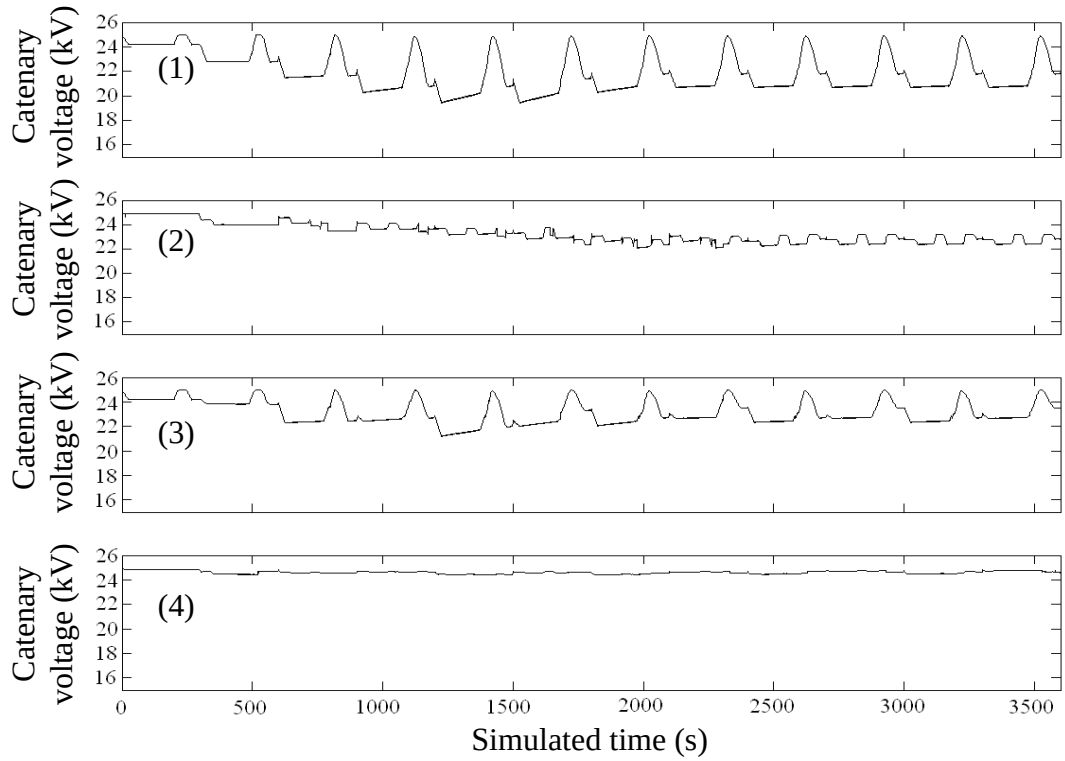


Fig. 6.19: Voltage at the MPTSC (Cases 1 – 4)

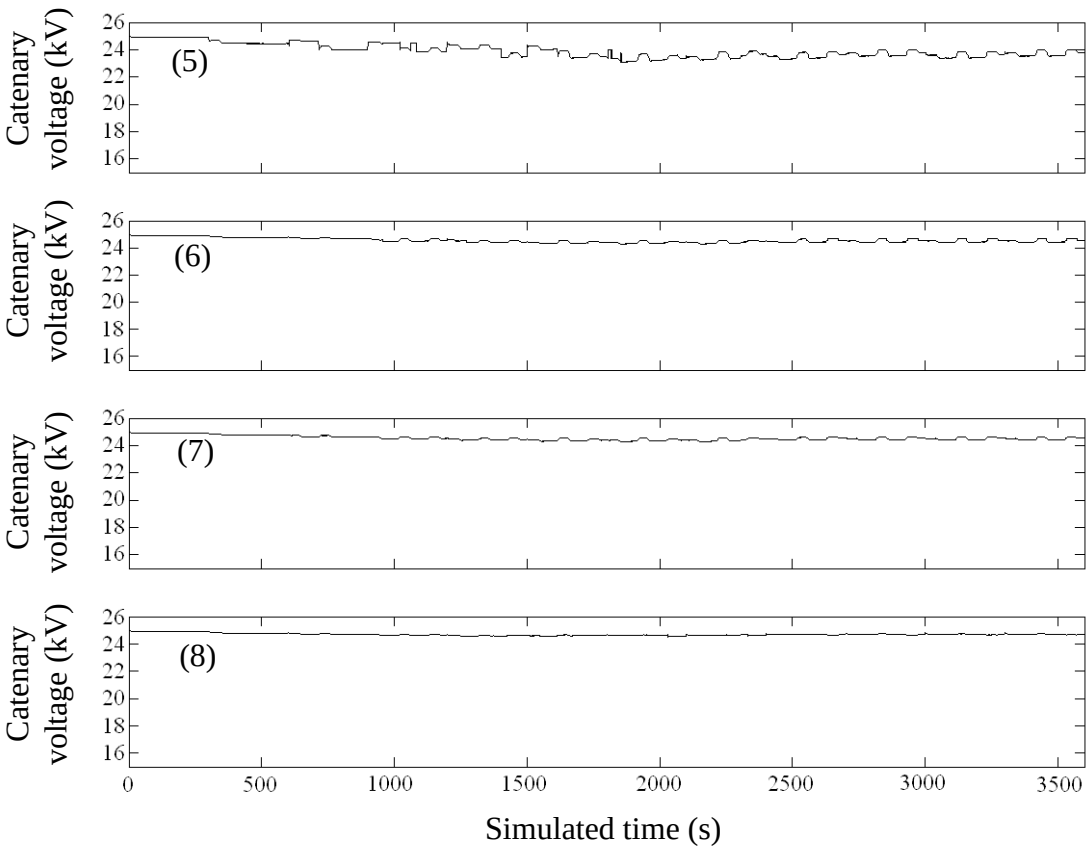


Fig. 6.20: Voltage at the MPTSC (Cases 5 – 8)

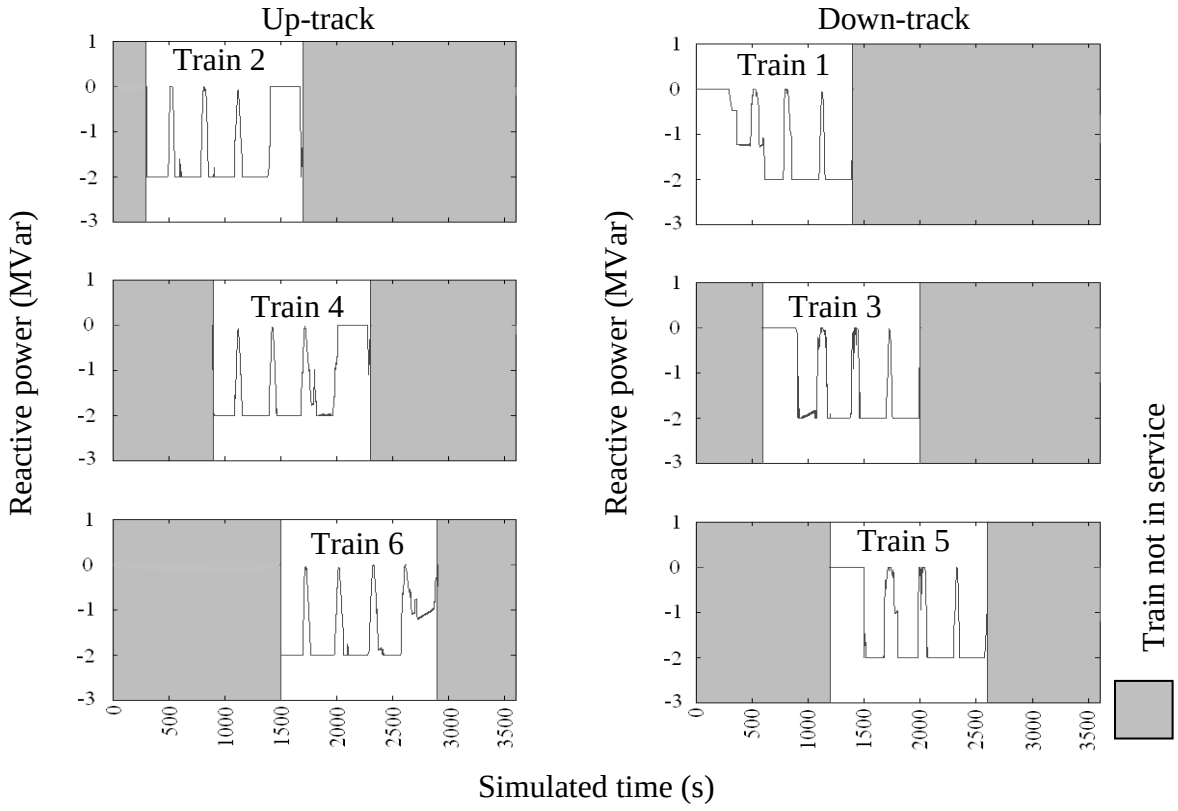


Fig. 6.21: Reactive power variations for selected trains; 50% OPF case

To illustrate the operation of the optimising controller and the reactive power commands, it transmits to the train, Fig. 6.21 shows the variations in reactive power aboard the first few PWM trains in each track. It can be seen that the trains spend a fair proportion of the time against the –2.0 MVar limit, suggesting that further improvements in both loss reduction and voltage profile would be attainable with greater MVar rating for the equipment. However, it was felt to be reasonable to limit the MVar rating to be the same as the MW rating, i.e. to limit the p.f. to 0.7 leading. To require significantly more MVar would impact significantly on the cost of the equipment.

6.5 Summary

An optimal area control system for an AC electrified railway system has been developed and discussed in this chapter. Appropriately controlling reactive power flows in the system leads to the reduction of power losses, the improvement of system voltage distribution and to the improvement of power transfer capability. The results confirm that useful improvements can be achieved by a central optimisation that utilises the ability of modern PWM locomotives to provide controlled reactive power. The greatest potential gains are where PWM trains are used to compensate for the relatively poor power factor of older equipment in a mixed fleet. Here, the area controller significantly out-performs the use of individual unity-power-factor controllers on each train or the use of the SVC. If, however, the entire fleet consists of PWM equipment operating at UPF, the further gains attainable with the area controller would not justify its use.

PART IV

Cost Estimation and Design

CHAPTER 7

COST ESTIMATION FOR REACTIVE POWER COMPENSATION

7.1 Introduction

Upgrading an electric railway system is to improve the system performance in some particular objectives, for example to increase the power transfer capability, to improve the system voltage profile, to increase the maximum speed limit, or to upgrade the signalling system, etc [11,77–80]. In this thesis, the power transfer and the voltage profile have been chosen as the main targets for investigation. The upgrade will enable the power feeding system to support heavier train services with stable, secure and reliable operation. In practice, there are several traditional methods to increase power supply capability and even the voltage profile, e.g.

- Replacement of power equipment such as conductors, transformers, etc [11,77,81]
- Installation of reactive power sources such as capacitor banks, SVCs, etc [82–84]

The area control concept introduced in Chapter 5 is an alternative to these traditional methods. Although the results in Chapters 5 and 6 confirm the excellent performance of the area control system, it would be less attractive in practice if there is no economic profit from its use. This chapter therefore discusses the use of this method from the economic perspective and any cost benefit is evaluated and compared.

The investigation involves cost estimation for upgrading an existing AC electric railway system by the use of the SVC or the area control system. This comparative study will demonstrate substantial benefits and will give basic arguments for justifying the methods of reactive power compensation.

7.2 Cost Estimation of Reactive Power Compensation for AC Railways

In the daily operation of an AC railway power supply operation, trains draw a considerable amount of reactive power from the power feeding system. Significantly, this causes insecure, unreliable and possibly unstable power supply operation due to the voltage drop and the power losses in the system. One possible solution to this problem is to introduce reactive power sources to minimise the power losses [85,86]. However, cost is a major factor in limiting the use of such compensating sources.

7.2.1 Benefit due to energy loss reduction

The power losses of the power feeding system can be reduced by the installation of the reactive power sources such as a capacitor bank, an SVC, etc. The saved power losses may be interpreted as a reduction of the electricity charge from the power supply company. Thus, the reduction of the electric energy charge can be expressed [87,88] as follows.

$$\Phi_E = k_E \times \Delta E_{Loss} \tag{7.1}$$

where

$\Delta E_{Loss} = E_{Loss}^{Old} - E_{Loss}^{New}$ in kWh

$E_{Loss}^{Old}, E_{Loss}^{New}$ are the energy losses before and after installation, $E_{Loss} = \int P_{Loss} dt$

k_E is the cost of per-unit energy charge (£/kWh)

Φ_E is the remission of the electricity charge (£)

7.2.2 Benefit due to peak power loss reduction

In general, installing a reactive power source can reduce the peak power losses as well as reducing the overall energy losses. This is important, especially for an industrial power system because it reduces the peak demand in the entire system. Reducing the peak demand will help reduce the size and number of electric power plants and power transmission lines

[89,90]. The benefit of the peak power loss reduction would be calculated by using equation 7.2.

$$\Phi_P = k_P \times \Delta P_{Loss} \tag{7.2}$$

where

$\Delta P_{Loss} = P_{Loss}^{Old} - P_{Loss}^{New}$ in kW

$P_{Loss}^{Old}, P_{Loss}^{New}$ are the peak power losses before and after installation

k_P is the marginal per-unit cost of the peak power loss (£/kWh)

Φ_P is the benefit of the peak power-loss reduction (£)

However, because AC railway electrification equipment is designed with a large peak power margin in order to be able to operate in emergency feeding conditions, the advantage of this effect is small in AC railways and will be ignored.

7.2.3 Installation cost

The installation cost of the reactive power source can be divided into two parts: i) investment or capital cost, and ii) maintenance cost. For simplification, the capital cost can be considered as shown in equation 7.3.

$$CC = \Theta + k_Q \times Q_C \tag{7.3}$$

where

Θ is the construction cost and property cost (£)

Q_C is the size of the reactive power sources to be installed (kVar)

k_Q is the per-unit cost of the reactive power source (£/kVar)

CC is the capital cost (£)

7.2.4 Maintenance cost

In addition, the maintenance cost can be described as follows [91].

$$MC = \sum_{y=1}^{LIFE} M \times \left(\frac{100 + \lambda}{100} \right)^{y-1} \tag{7.4}$$

where

λ is the annual percentage increment of the maintenance cost (%)

M is the annual maintenance cost (£)

LIFE is the lifetime of the reactive power source to be installed (Years)

MC is the total maintenance cost (£)

7.2.5 Cost function

The cost function, Ψ, for the provision of reactive power compensation is summarised the following formula.

$$\Psi = CC + MC - \Phi_E - \Phi_P \tag{7.5}$$

In fact, this function is intrinsically non-linear and discrete. It is mainly used to find the optimal size and optimal location of the reactive power source for installation [92,93]. The exact power rating of the reactive power source is dependent on what is available from manufacturers and is usually quantised, for example ABB provides 13 different kVar ratings for medium-voltage capacitors as 155, 220, 270, 310, 360, 410, 540, 595, 660, 725, 800, 900, and 1000 kVar [94]. Another key factor is location. It must usually be at any of the feeder substations and, naturally, this variable is also discrete. However, in AC electric railways where the installation is limited to being at the substation, the MPTSC or the ITSC (if any) [7–9], this means that, in any one power feeding section, there are only three possible locations (only two if ITSC does not exist). Thus, the problem is reduced to that of optimal sizing only.

7.3 Cost Estimation for Increasing MVA Capacity of PWM Locomotives

As analysed in Chapter 5, the reactive power limits depend not only on the locomotive properties but also the operating conditions. For example, consider the single-train simulation

as given in Fig. 6.8., section 6.2.6. Assume that the locomotive used has the following properties (as in Figs 5.4 and 5.5).

Traction transformer:	2 identical secondary windings with turn ratio 1:11, 1:16 and 1:33
Input inductance:	L_S = 0.981 mH
DC link voltage:	V_{DC} = 1800, 2500 and 3000 V
Modulation index:	m ∈ [0.45,0.95]
Maximum power:	2×2.5 MW

To increase the reactive power limit, the effect of the traction transformer's turn ratio and the DC link voltage is investigated. The first test is performed by simulating the train movement with the fixed turn ratio at 1:16 while varying the DC link voltage (1800, 2500 and 3000 V). The simulation results are shown in Fig. 7.1.

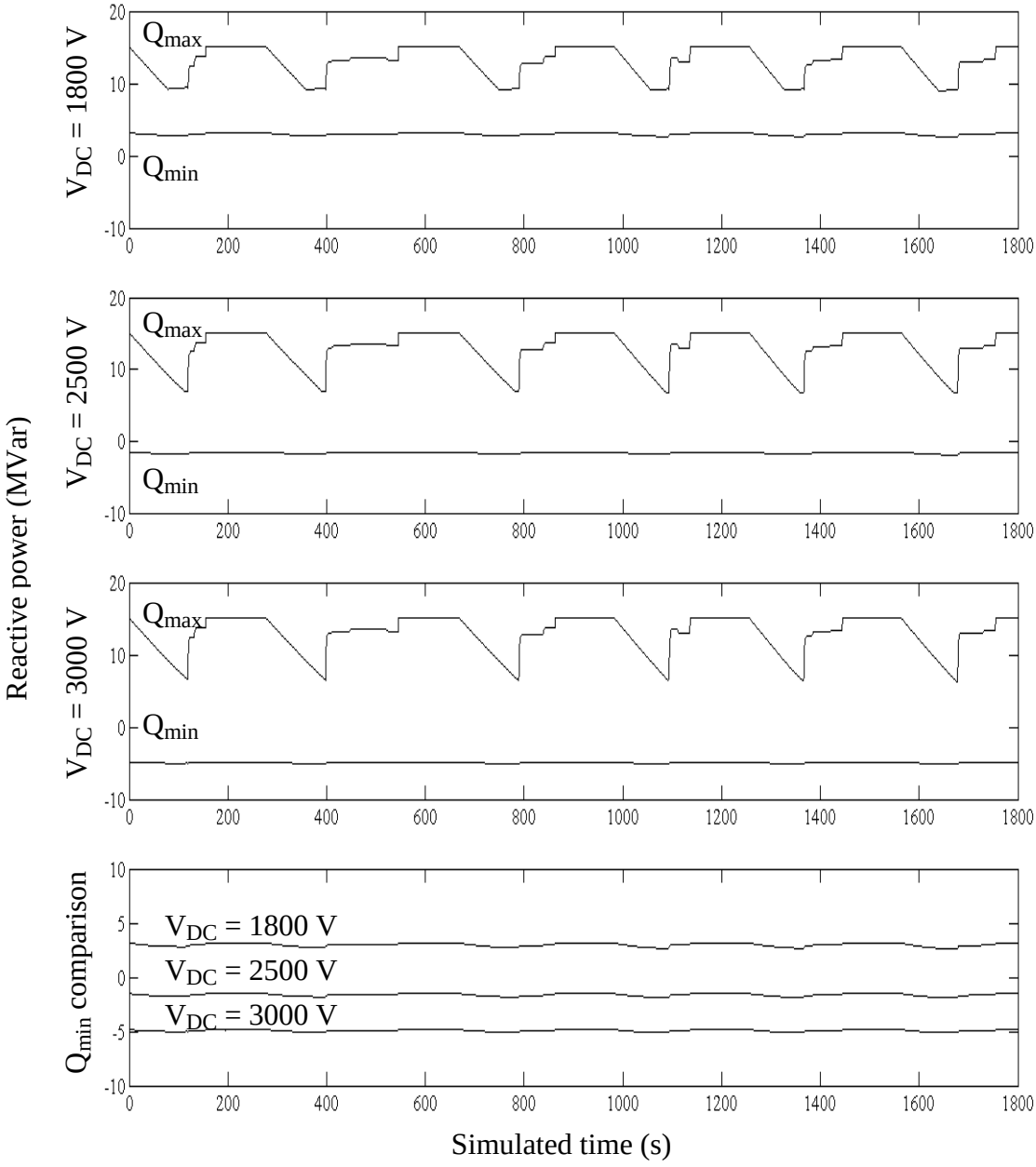


Fig. 7.1: Effect of the DC link voltage on the increase in the reactive power limit

The second test is to vary the turns ratio (1:11, 1:16 and 1:33) with a constant DC link voltage of 2500 V as shown in Fig. 7.2.

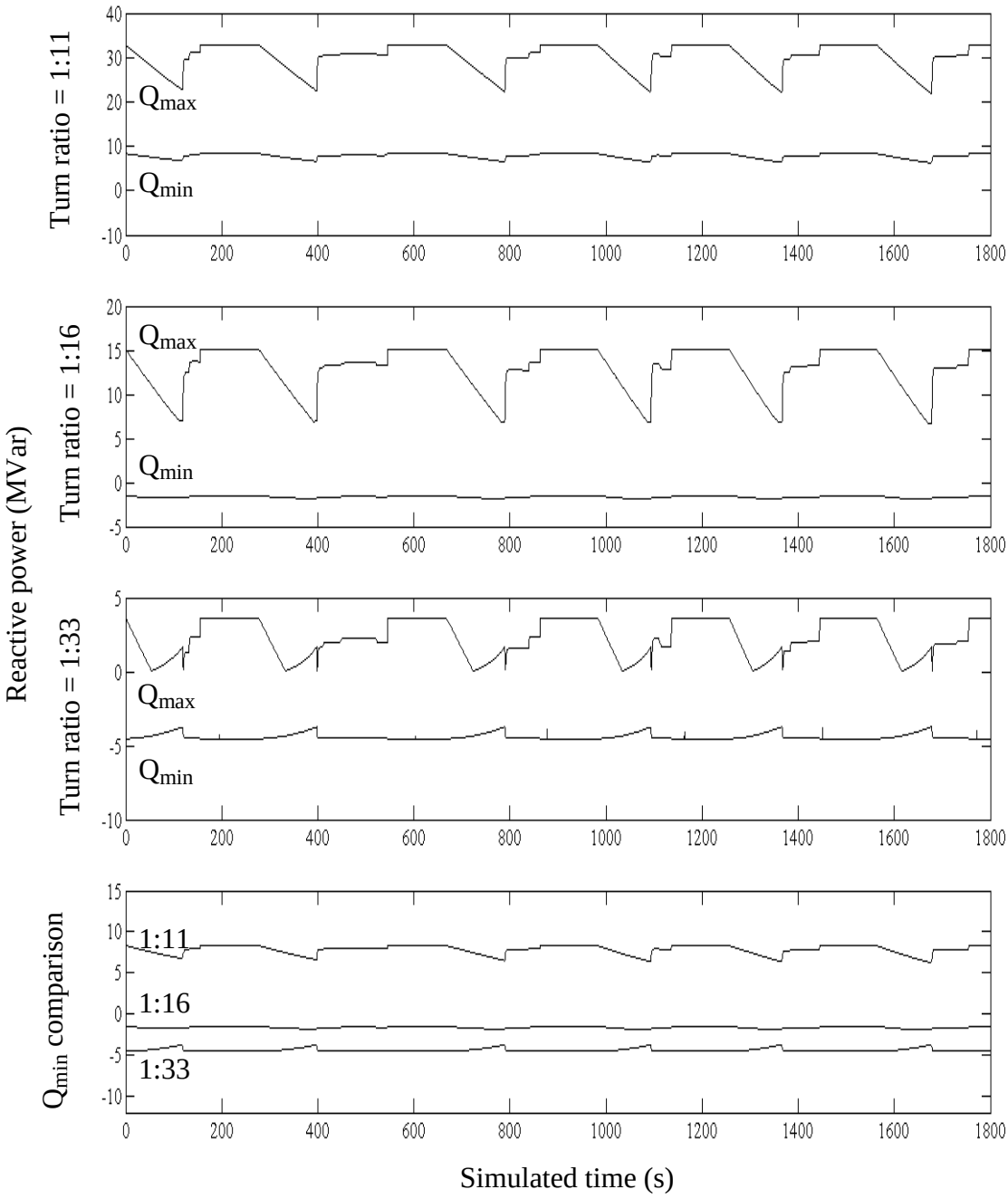


Fig. 7.2: Effect of the traction transformer's turn ratio on the increase in the reactive power limit

These tests are based on the single-train movement simulation only to avoid the interaction from other factors. In this circumstance, the pantograph voltage is only slightly dropped from the nominal voltage, about 24 – 25 kV during the services. It can be assumed constant and does not give a significant effect on the reactive power limit. Thus, only the turns ratio greatly

influences the secondary voltage $\hat{V}_S$, see Fig. 5.5 in Chapter 5, and thus causes the change in the reactive power limit as shown in equations 5.6 and 5.7. The DC link voltage also influences the reactive power limit as shown in Fig. 7.1. The higher the reactive power limit, the higher the DC link voltage. This directly impacts on the individual on-board traction controller and the DC link capacitor.

This indicates that special and careful design of the PWM locomotive controller may be necessary if the area control system is to be implemented. The change in the reactive power limit requires a revised design for the traction transformer, DC link capacitor, traction controller, etc. This also leads to an increase of the locomotive maximum current, therefore re-cabling with higher current capacity conductors may be required. Inevitably, the specially designed PWM locomotive will be more expensive than a conventional one.

For simplification, the cost related to increasing the PWM locomotive rating in this thesis is estimated by using an incremental cost concept. This estimation is based on the fact that the older phase-angle controlled locomotives are replaced in phases by PWM locomotives; eventually in the future a full replacement will be done. However, if the specially designed PWM locomotives are used instead of conventional ones, the area control system could be implemented. This would give a great return in the energy consumption and the reactive power compensation. The incremental cost defined here is the cost difference between the special design and the typical design for a PWM locomotive. Although a controller in the central control centre is required, there is no additional hardware cost because there must already an existing control centre for automatic train control and/or signalling, and this is likely to include two-way communication with the trains.

Similar to the conventional reactive power sources, the incremental cost for increasing the power rating of the traction equipment is given as follows.

$$\Gamma = n \times k_S \times \Delta S \tag{7.6}$$

where

k_S is the incremental per-unit cost of power (£/MVA, £/MVar)

ΔS is the increased power capacity (MVA, MVar)

n is a total number of the specially designed PWM locomotives

Γ is the incremental cost (£)

Although any power equipment is normally priced by its MVA rating, for convenience, the incremental cost given here will be based on the MVar rating to simplify the problem. As with the alternative reactive power compensator, there is the remission of the electricity charge due to the power loss reduction as described in equation 7.1. Thus, the total cost function Ψ for this case can be obtained by the following equation.

$$\Psi = \Gamma - \Phi_E = n \times k_S \times \Delta S - k_E \times \Delta E_{Loss} \tag{7.7}$$

7.4 Design of the AC Railway Reactive Power Compensation

In this section, cost estimation and design of the reactive power compensation for an AC railway test system is demonstrated. Consider a double-track AC railway test system, which is 25-km long. Full information for this test case can be found in Appendix E.7. The speed trajectory and power curves of train number 1 on the up-track are shown in Fig. 7.3. Fig. 7.4 presents distance vs time curves and MVA drawn from the substation.

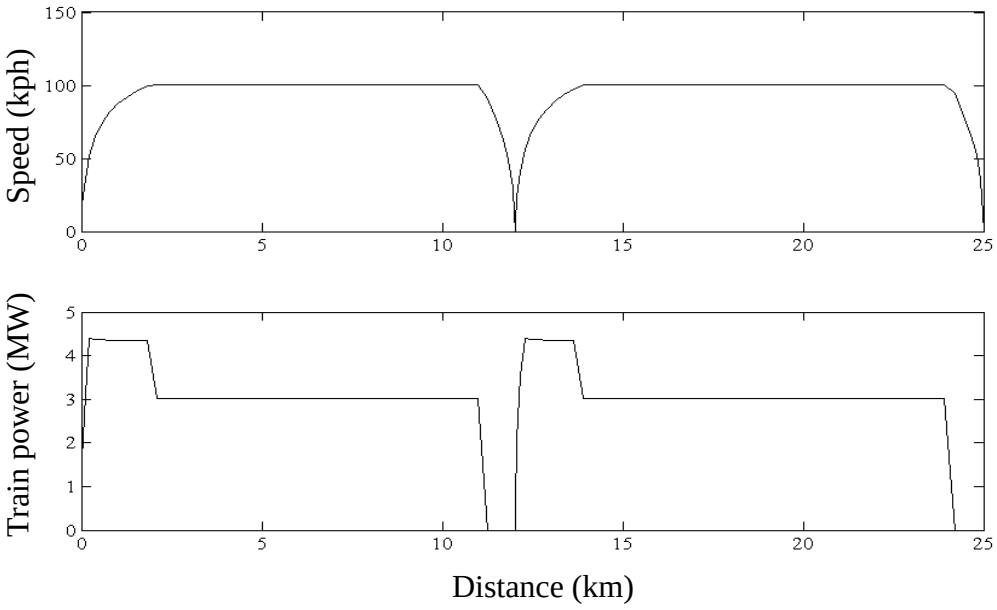


Fig. 7.3: Speed trajectory and power of train 1 on the up-track

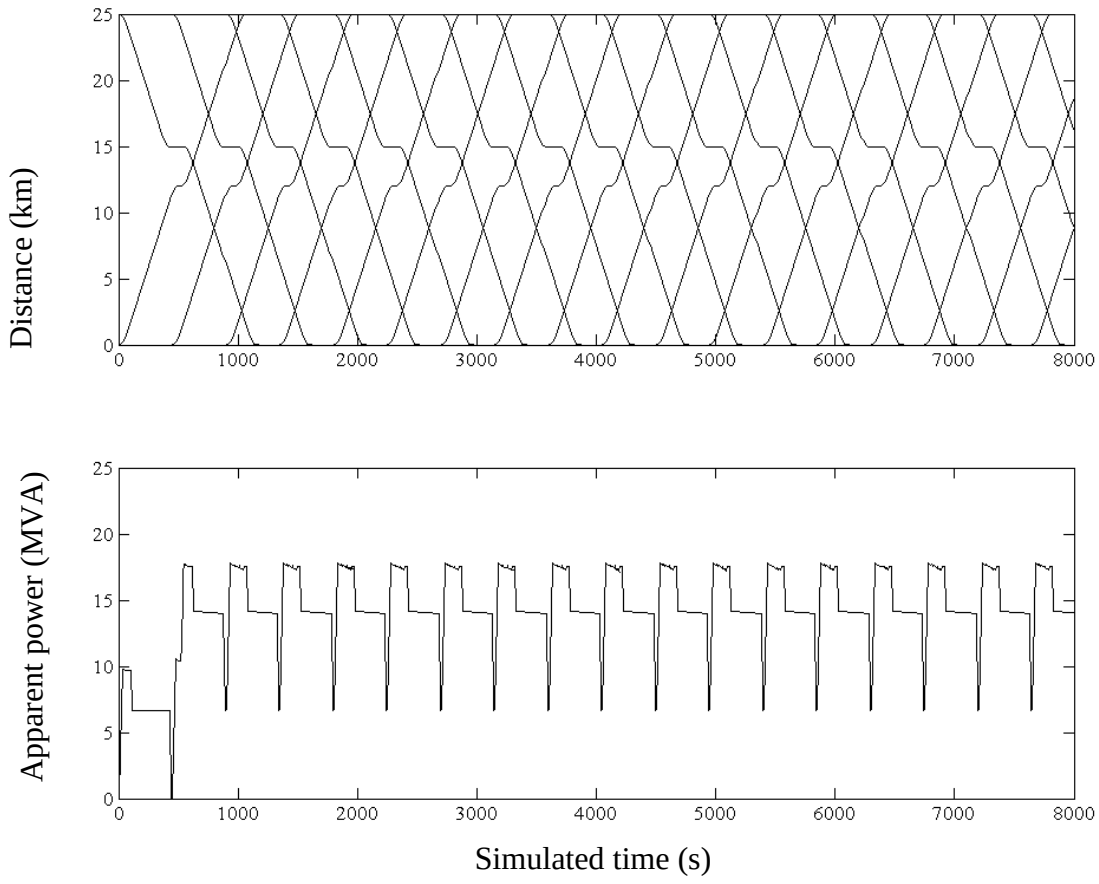


Fig. 7.4: Distance-time curves and MVA drawn from the substation

7.4.1 SVC installation

To design the reactive power compensation by installing the SVC as previously described is to find the optimal size and location to install the compensator. In the AC railway test system, the substation and the MPTSC are the only two locations available to place such equipment. Sixteen SVC ratings are considered (0, -1, -2, …, -15 MVar), where 0 MVar is the condition that no SVC is installed.

The objective function for this design problem consists of:

- Capital cost CC, $CC = \Theta + \sum_{i=SS,\ MPTSC} k_Q \times Q_C$, where $\Theta = 0$.

  * Neglecting the construction and property cost

- Neglecting the maintenance cost, M = 0
- Neglecting the benefit from the peak power loss reduction
- Considering the benefit from the remission of the electricity charge, Φ_E

The cost function can be expressed as in equation 7.8

$$\Psi = \sum_{i=SS,\ MPTSC} k_{Q,i} \times Q_{C,i} - k_E \times \Delta E_{Loss} \tag{7.8}$$

Thus, the optimal design is to minimise the cost function as the problem stated below.

$$\underset{Q_{C,i} \in \Lambda}{\text{Minimise}} \quad \Psi \tag{7.9}$$

where $\Lambda \in \{0, -1, -2, \ldots, -15\}$, the set of SVC ratings.

This problem is concerned with two main search processes as follows.

i) Find appropriate location and optimal size for the SVC installation to minimise the capital cost (CC)

ii) From i), perform optimal control for the designed SVC to maximise overall energy loss reduction (ΔE_{Loss})

The first scheme is clearly a discrete optimisation problem. There are two control variables: i) Location and ii) SVC Size, they can be combined into a single variable as given in equation 7.10.

$$\mathbf{w} = \begin{bmatrix} Q_{C,SS} & Q_{C,MPTSC} \end{bmatrix}^T \tag{7.10}$$

where

$Q_{C,SS}$ is the SVC size installed at the substation, $Q_{C,SS} = 0$ for no installation

$Q_{C,MPTSC}$ is the SVC size installed at the substation, $Q_{C,MPTSC} = 0$ for no installation

This optimisation can be performed by the combinatorial or discrete optimisation [95]. With a given discrete search space Λ, there are 16^2 or 256 possible solutions. In this part, Genetic Algorithms or GA [96-98] has been employed to generate the control vector **w**. The GA can be summarised as follows.

- **Genetic algorithms**

Genetic algorithms are one of the well-known intelligent search mechanisms based on the Darwinian principle of natural selection [96,99,100]. It consists of a population of bit strings representing the control variables and three genetic operations, 1) Selection or Reproduction, 2) Crossover and 3) Mutation. The algorithm starts with a random creation of an initial population. During each iterative step, which is referred to as a generation, the strings within the current population are evaluated for their fitness values via the objective function. A new population, called an offspring, is then created using these three genetic operators. This stochastic process is intended to generate a new and better population from the old population. Assuming the algorithm converges, a set of solutions with better fitness is obtained. Thus, the optimal solution is found. It can be briefly described step-by-step as:

Given that a population set has N members and each member consists of M variables. Each individual member of the population is called a chromosome.

1) *Initialisation*: Set k = 0. Generate an initial population, $\mathbf{p}_{k,i} = [\mathbf{w}_1\ \mathbf{w}_2\ \ldots\ \mathbf{w}_M]^T$ where i = 1, 2, …, N by using a random process and then evaluate their corresponding fitness function.
2) *Evolution*: Apply the genetic operators to create an offspring population as the sequence below:
   - Selection: Form a set $\mathbf{m}_k$ (Mating pool) with the same number of members as $\mathbf{p}_k$. Each member of $\mathbf{m}_k$ is created from $\mathbf{p}_k$ by using a random procedure, e.g. the roulette-wheel or tournament schemes [96], with the assumption that each chromosome has a different chance (probability) to survive. The higher the fitness value, the higher the chance or probability.
   - Crossover: This operation is applied to a sub-set $\mathbf{m}_k$ by taking a pair of chromosomes called the parents. The parents will yields a pair of offspring chromosomes. This operation involves exchanging sub-strings of the parent chromosomes. It is performed by choosing a random position in the string and then swapping either the left or right sub-string of this position (one-point crossover) with its chromosome mate.
   - Mutation: For a chromosome to be mutated, the values of a few positions in the string are randomly modified. To prevent complete loss of the genetic information carried through the selection and crossover processes, mutation (if used at all) is limited to typically 2.5% of the population [96].
3) *Fitness test*: Evaluate the fitness value for the generated offspring population.
4) *Convergence check*: Check for violation of all termination criteria. If not satisfied, repeat the evolution process.

The second scheme is in fact a sub-problem in the first scheme. During the fitness test, each individual chromosome will be evaluated for its fitness value one-by-one. The considered

chromosome will be decoded into its real value, the real SVC rating such as -5 MVar rather than the string chromosome 1010, see Table 7.1. With the SVC rating generated by the GA, the real power loss optimisation will be performed subject to the SVC installed at the substation or MPTSC, if any. The steepest descent algorithm is the main optimisation for this sub-problem as described in Chapter 5.

This design procedure can be summarised in the flow diagram shown in Fig. 7.5.

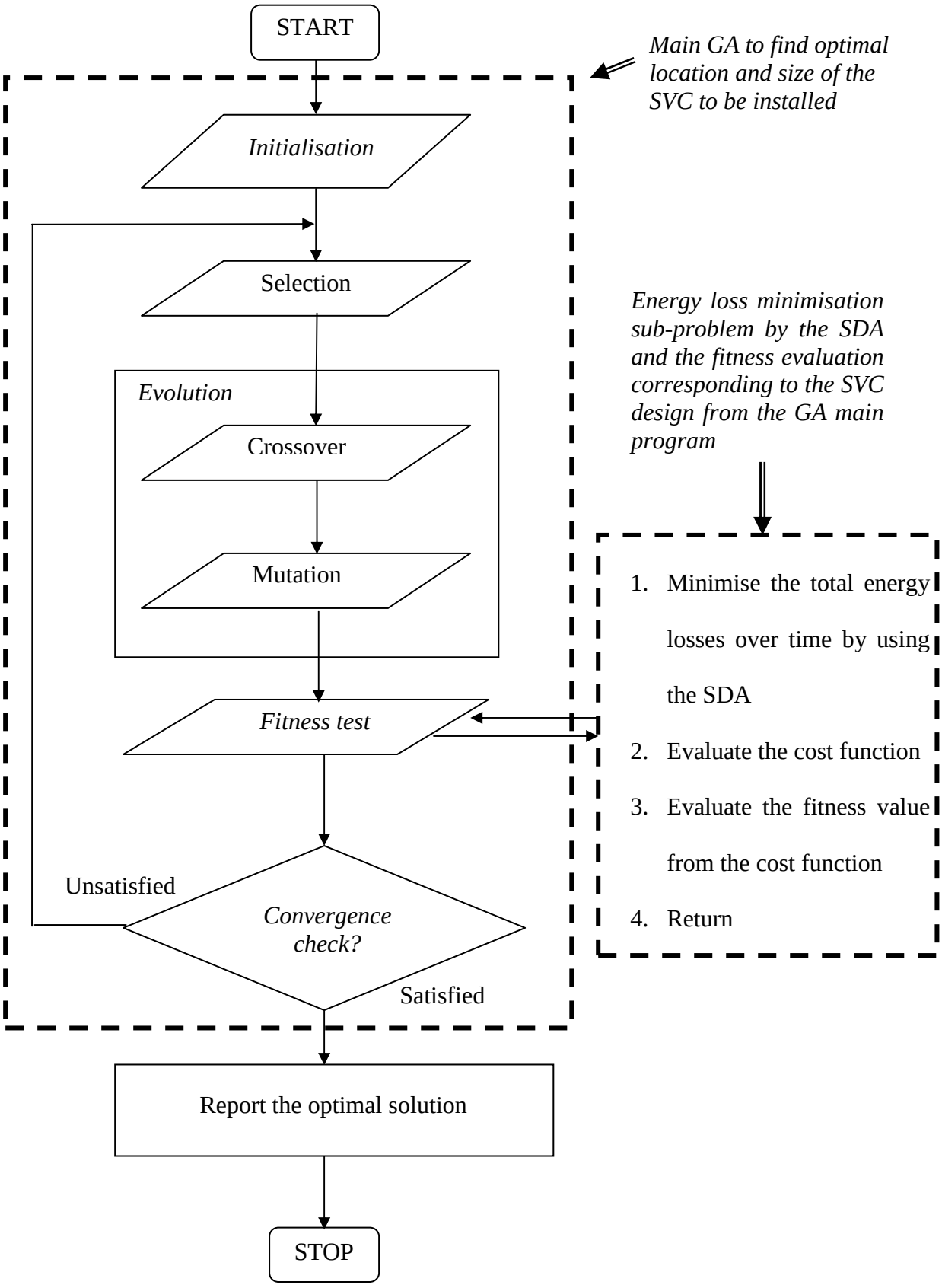


Fig. 7.5: Flow diagram for the SVC design problem

In this particular problem, the two variables are both assigned by sixteen different SVC ratings (Λ). Each variable can be represented by only a set of 4-bit strings. The binary codes for this problem are shown in Table 7.1.

Table 7.1: Binary codes for the SVC design problem

SVC rating (MVar)	Binary code
-15	0000
-14	0001
-13	0010
-12	0011
-11	0100
-10	0101
-9	0110
-8	0111
-7	1000
-6	1001
-5	1010
-4	1011
-3	1100
-2	1101
-1	1110
0	1111

The following are parameters used for this design

- k_C = 108,000 £/MVar [60]
- k_E = 0.04 £/kWh (Medium-sized industrial user) [101]
- Parameter setting for the GA method

  Population size = 5

  8 bits per one chromosome, e.g. (-2,-6) = 11011001

  Probability for crossover = 75%

  Probability for mutation = 2.5%

  Maximum generation = 30
- Fitness function = $-\Psi$
- Maximum error for the SDA = 0.0001 for all

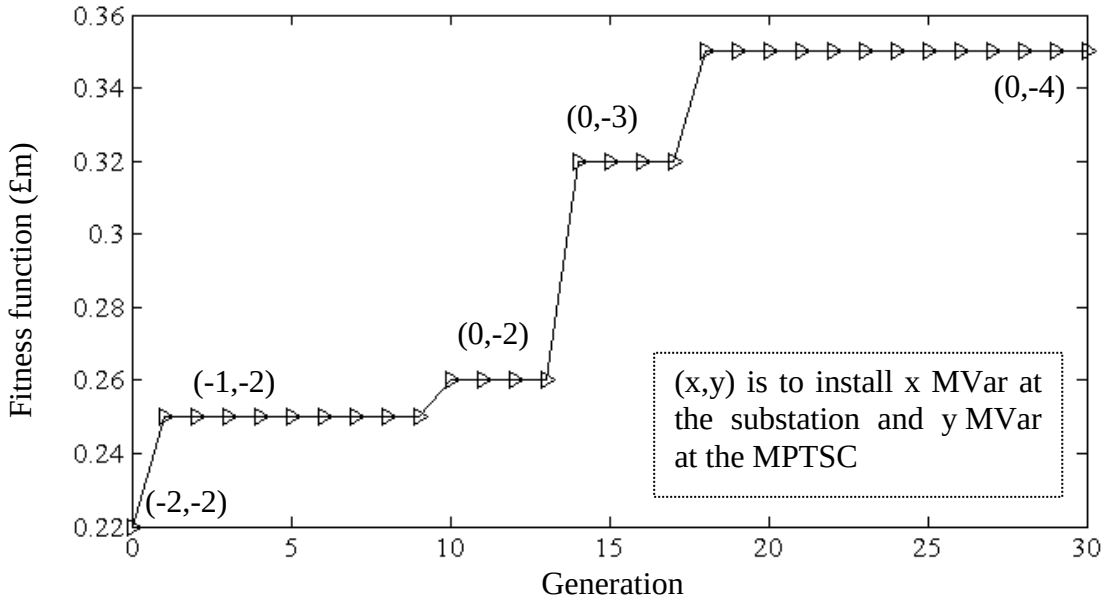


Fig. 7.6: Best fitness value improvement during the optimisation process for 20-year planning

Table 7.2: Long-term planning for SVC installation in the AC railway power feeding system

Case	SVC rating (MVar)		Planning period (Year)	Total benefit or fitness value (£)
	SS	MPTSC		
1	0	0	5	0
2	0	-1	10	22,926.95
3	0	-3	15	161,818.46
4	0	-4	20	347,717.93
5	0	-5	30	772,803.02
6	0	-6	40	1,232,049.54
7	-2	-6	50	1,728,139.04
8	-3	-6	80	3,314,420.87

7.4.2 Special PWM locomotive design

Extension of the reactive power limit for the PWM locomotive is directly affected the cost of the PWM power converter module. Although some other components might be involved such as a traction transformer or re-cabling, they are neglected in this estimation. Only the cost of the PWM power converter is taken into account. In 1997, the cost of a three-phase PWM power converter implemented by using IGBTs was 50 US$ per kVA [102] or 18 £/kVA for a single-phase PWM power converter system, where US$1.0 = £0.6 and four switches are used in

the single-phase converter instead of six switches. To design such a locomotive requires the following assumptions.

- Half of the locomotives will be replaced by modern PWM locomotives
- Each PWM locomotive is designed to supply real power up to 7 MW maximum
- The PWM locomotive is designed to operate at unity power-factor operation only
- The increase of the MVar rating does not affect the MW rating
- Total cost is estimated for the power converter system only
- Twenty older locomotives will be replaced, n = 20
- All PWM locomotives are identical
- k_S = 20 £/kVar or 20,000 £/MVar

This problem is not as complicated as that of the SVC design. There is only one control variable, which is the lower bound of the reactive power limit and this applies to all PWM locomotives. The planning period is given as those of the SVC case, 5 years, 10 years, 15 years, 20 years, 30 years, 40 years, 50 year and 80 years. The design process starts by varying the reactive power limit from -7 MVar to -1 MVar with 1 MVar per step. At each step, the SDA is employed to minimise the total energy losses whilst accounting for the reactive power limit. The minimum energy loss will be used to evaluate the cost function or the total benefit as described in equation 7.7. The results are shown in Table 7.3.

Table 7.3: Long-term planning for the use of PWM locomotives as power compensators

Rating (MVar)	Total benefit (£), –Ψ							
	5 year	10 year	15 year	20 year	30 year	40 year	50 year	80 year
-1	**256,908**	528,030	799,152	1,070,274	1,612,518	2,154,762	2,697,006	4,323,738
-2	241,234	**538,490**	**835,746**	**1,133,002**	**1,727,514**	**2,322,026**	**2,916,538**	4,700,074
-3	179,722	482,599	785,476	1,088,353	1,694,107	2,299,861	2,905,615	**4,722,877**
-4	90,973	394,397	697,822	1,001,246	1,608,095	2,214,944	2,821,793	4,642,340
-5	-17,022	286,420	589,863	893,306	1,500,191	2,107,077	2,713,962	4,534,619
-6	-140,466	162,977	466,419	769,862	1,376,748	1,983,633	2,590,519	4,411,175
-7	-276,456	26,987	330,429	633,872	1,240,758	1,847,643	2,454,529	4,275,185

* Bold number is the optimal value

Table 7.4 summarises the cost estimation among the SVC installation, the use of the PWM locomotives for the reactive power compensation and the use of 100% UPF-operated PWM locomotives. For the UPF-operated case, no capital cost is applied due to the assumption that the older locomotives were designed to operate within a certain period of time and need to be replaced.

Table 7.4: Total benefit comparisons

Planning period (Year)	SVC	Use of the PWM locomotives	
		UPF operation	Optimal area control
5	£0	£217,796	£256,908
10	£22,927	£435,591	£538,490
15	£161,818	£653,387	£835,746
20	£347,718	£871,182	£1,133,002
30	£772,803	£1,306,773	£1,727,514
40	£1,232,050	£1,742,364	£2,322,026
50	£1,728,139	£2,177,955	£2,916,538
80	£3,314,421	£3,484,728	£4,722,877

This estimation illustrates have to design reactive power compensation in AC railways from the perspective of the total benefit, $-\Psi$. It is clear that, from the given test case scenario, the SVCs reduce the total power losses adequately and it is undoubtedly the most straightforward choice for the upgrade of an AC railway electrification system. With the introduction of PWM locomotives and the phased replacement of older phase-angle control types, there is an opportunity to achieve a greater total benefit as shown in Table 7.4, especially where the area control system is applied.

7.4.3 Consideration of discounting cash flow

Cost analysis is concerned with the comparison of the alternatives to choose the best way of making an investment. It is important to include account of the different economic values of investments made at different times during the analysis period. When money is invested, compound interest is paid on the capital sum. The interest rate comprises inflation, risk and real costs of postponing consumption. Thus, money used to invest in this situation

could be invest elsewhere and earn a dividend. To consider this effect, all future costs and benefits are discounted to convert them to the net present values (NPV) of costs as shown in equation 7.11. NPV is a measure of the economic worth of an investment [103]. A positive NPV indicates that the investment is justified economically at a given discount rate.

$$NPV = \sum_{i=1}^{n-1} \frac{b_i - c_i}{\left(1 + {}^{r}/_{100}\right)^i} \tag{7.11}$$

Where

n is the analysis period in years

b_i is the sum of all benefits in year i

c_i is the sum of all costs in year i

r is the discount rate in percentage

Modifying the cost function as described previously, the design of the reactive power compensation in AC railways with the account of discounting cash flow can be summarised in Tables 7. 5 and 7.6.

Table 7.5: NPV with 5% discount rate

Planning period (year)	NPV (£)		
	SVC	UPF	OPF
5	0	198,013	0
10	0	353,162	108,053
15	50,408	474,725	277,369
20	123,595	569,973	410,033
30	234,849	703,076	595,423
40	314,255	784,790	709,236
50	350,761	834,955	779,107
80	409,278	896,266	864,503

0 indicates that the NPV is negative and it is not considered for investment

Table 7.6: NPV with 10% discount rate

Planning period (year)	NPV (£)		
	SVC	UPF	OPF
5	0	181,632	0
10	3,095	294,411	40,912
15	24,305	364,438	138,448
20	45,920	407,919	199,009
30	66,993	451,681	259,963
40	78,398	468,554	283,463
50	82,795	475,058	292,523
80	85,396	478,906	297,882

0 indicates that the NPV is negative and it is not considered for investment

These calculations include the effect of discounting cash flow to convert all costs and benefits of each method to present values for comparison. As a result, the use of PWM locomotives to perform unity power factor operation is the best way for investment in this account.

7.5 Summary

This chapter demonstrates the cost estimation and design of reactive power compensation in AC railway electrification systems. Relevant costs for conventional Var design are reviewed. In comparison, cost estimation for increasing MVA capacity of PWM locomotives is studied. Also, simulation tests of the effect of some key parameters, traction transformer's turn ratio and DC link voltage, on locomotive's reactive power limits are included. The design procedure and the comparison among the results obtained from each method are proposed and discussed. With the assumed cost estimation parameters, the area control system is shown to give more benefit over either the classical SVC method or the use of unity power factor controllers on-board new PWM locomotives. However, when the discount cash flow is considered, the unity-power-factor control is more appropriate.

This page is intentionally left blank

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

8.1 Summary of the PhD Thesis

This thesis can be classified into four main parts: i) AC railway power flow calculation part 1 – Theory and application (Chapter 3), ii) AC railway power flow calculation part 2 – Exploitation of sparsity (Chapter 4), iii) Optimal area control system (Chapters 5–6) and iv) Cost estimation and design (Chapter 7).

In the first part, the power flow calculation for an AC railway power feeding system has been investigated. The well-known Newton-Raphson power flow method is first described and applied to the power feeding configuration, either single-phase or bi-phase system. Following this, a new power flow method named the "Sequential Linear Power Flow Method" is introduced and developed. The convergence and memory requirements for each method are carefully analysed and compared. The last section provides comparative tests using two standard IEEE test systems (with slight modification) and three AC railway test systems, one being a single-phase system and the other two are bi-phase systems.

The second part investigates enhancing the computing performance of the power flow solution in order to reduce the overall execution time. Methods of solving a set of linear equations have been carefully studied. The sparsity resulting from a band diagonal matrix formulation is exploited. A heuristic bus-ordering technique to create a sparse matrix in a band diagonal form is defined specially for AC railway power feeding systems. The effectiveness of the banded bus-ordering technique is evaluated for five different power-feeding topographies. Furthermore, discrete time simulations for the two-hour operation of the AC railway test system executed by the three different computers are compared.

The area control concept is introduced and illustrated in Chapters 5–6. Firstly, the practical use of the SVC the AC railway application is reviewed. Secondly, the real and reactive powers flowing through a PWM locomotive traction controller are analysed and discussed to establish the response time of the reactive power control. Thirdly, the optimal AC railway power flow problem is studied. The centrally controlled area controller is introduced and compared to the conventional SVC method. Fourthly, the centralised area control method is fully developed and tested.

The final part is related to the design and cost estimation of the reactive power compensation. There are three main sections. The first concerns the external reactive power source installation and its cost estimation and energy-saving benefit. The second concerns estimating the cost of modifying the PWM locomotives to be able to support higher reactive-power capability. The last section presents a design example of an AC railway test system and comparison between the results obtained from the SVC installation and from the PWM locomotive schemes.

8.2 Conclusions

8.2.1 AC railway power flow calculation part 1 – Theory and application

The Newton-Raphson power flow solution method for both single-phase and bi-phase AC railway power feeding systems has been developed and used to represent classical methods for AC railway power flow calculation. The Newton-Raphson power flow method has quadratic convergence property and the test cases only required a few iterations to solve; 3–5 iterations were generally sufficient. A new method, the Sequential Linear Power Flow method has been successfully developed and its effectiveness has been compared to the Newton-Raphson method. Although it has only linear convergence, its execution time is about two times faster than that of the Newton-Raphson method. Also, the memory requirement is considerably

reduced and the computer programming is very much simpler. As a result, it is more efficient and faster to use the Sequential Linear Power Flow method as the principal power flow algorithm in AC railway power system analysis.

8.2.2 AC railway power flow calculation part 2 – Exploitation of sparsity

The AC railway power flow calculation can be further improved by careful enhancement of the linear system solver. Band matrix computation is one of the sparsity techniques available. This technique is very powerful due to the relevant algorithms being simple and to the compact data storage. It is not only advantageous in large-scale systems, but is also worth employing in small-scale systems. A heuristic banded bus-ordering technique has thus been developed and tested for the performance of the banded matrix version. It has been shown to many power feed configurations, this method efficiently creates a band diagonal matrix. Simulation results show that the banded bus-ordering technique together with the band matrix computation can reduce the overall execution time by about 50%. Furthermore, the use of recursive start conditions can also reduce execution time by 50%. The combination of the banded bus-ordering technique, band matrix computation and recursive start can reduce the execution time by as much as 75%.

8.2.3 Optimal area control system

The use of PWM locomotives as the mobile reactive power compensators has been studied theoretically. The limitation and response time of the reactive power flow have been analysed. The DC link voltage, the pantograph voltage, the modulation index and the phase angle of the PWM power converter, and the real power, etc, significantly influence the reactive power capability of the PWM locomotive. This indicates that the development of the PWM locomotive for reactive power compensation requires special and careful design for the locomotive to be able to support significantly more reactive power. To take full advantage of

the reactive power capacity, a centralised area control system has been proposed in order to minimise the total power losses. Both static test cases, where all trains are fixed in position, and dynamic simulations show that the area control system gives better performance compensating the excessive reactive power for an entire power feeding section, than does an SVC or even than operating the PWM locomotives at unity power factor. Even when only half of the locomotives are replaced by PWM ones, the power losses are reduced, and the system voltage profile and the power transfer limit are significantly improved.

8.2.4 Cost estimation and design

System designs for the reactive power compensation by the installation of an external reactive power source and by utilising the area control system has been undertaken. The key factors that influence the total cost have been examined. Life cycle costs for the reactive power compensation with SVC installation and with the optimal area control were found and compared. As a result, some reasonable costing assumptions, the proposed scheme is shown to give more benefit than the conventional SVC compensation methods.

8.3 Suggestions for Future Work

8.3.1 Railway power flow calculations

a) A long-distance railway system usually contains several power feeding sections. Under normal operation, each power feeding section is isolated and operated independently from the other supply sections. To simulate all separate power systems at the same time in a single computer is inefficient and consumes substantial memory spaces. This would causes very slow computation, even a very high-performance computer was employed. Parallel computing using a cluster of moderate computers is one possible solution to this problem.

b) In a densely populated, heavily industrialised or busily commercial area, there are several types of train services, such as long distance or suburban AC railways, DC mass transit systems, etc. It would be worth developing a power flow calculation that can efficiently handle both AC and DC railway systems and several classes of trains.

c) The introduction of PWM controllers on the trains, either simply as traction controllers or to provide reactive power compensation as well, can create problem at frequencies other than the fundamental. These controllers behave as harmonic frequency current generators and can cause substantial harmonic currents to be injected into the system at the load point. Hence, the harmonics can be distributed to other points in the entire power feeding system and cause substantial problems to other consumers. Thus, a harmonic power flow analysis is needed to characterise this effect. The lump parameter modelling used in this thesis is not suitable for analysing harmonic flows, and needs at least the addition of capacitance. Models based on a multi-conductor approach are usually used for this type of analysis.

d) The power feeding systems used in this thesis is assumed to operate in normal feeding conditions. However, railway schemes are designed to operate in various emergency conditions, e.g. short-circuits, power equipment outage, etc. The extension of the power flow analysis for such emergency conditions would help the system operator to prepare, prevent and restore the system.

8.3.2 Optimal area control system

a) In this thesis, normal operation with uniform train services is assumed. In a practical system, many trains suffer delay and the train traffic regulation is rarely straightforward. The extension to include unpredictable events like train delay would give a realistic operation closer to the real world. Some random or stochastic processes could be employed to simulate an

unexpected train delay caused by many possible circumstances, e.g. traffic congestion, short-term faults, etc.

b) The optimal area control system requires a two-way communication system on the railway. Investigation is needed as to whether the existing communication and train control system is able to support the controllers or whether a new wireless control system specially for this purpose is required.

c) The up-rated reactive power controllers introduces their own problems and a detailed study of harmonic effects caused by the reactive power control scheme is required possibly using more refined analysis techniques as stated above. It may prove necessary to design harmonic filters to eliminate or reduce the consequences of the harmonic power flow. Besides the harmonics, electromagnetic interference is another significant phenomenon that might be worsened when the area control system is applied.

d) When particular power supply failures occur, first or second stage emergency feeding configurations are used. The operation of the SVC or the area control system must be investigated to verify that under such emergency feeding conditions, the reactive power control can still function satisfactorily.

8.3.3 Cost estimation and design

There are some several factors that could be included in further research:

a) Include the property and maintenance costs

b) Determine the effect of the energy-saving principle, e.g. coast control

c) Include the ITSC for the external reactive power source installation

d) Consider the cost of on-board control circuits and a corresponding wireless control system that might be installed or extended

e) Design by using information from the forecasting of train services

This page is intentionally left blank

REFERENCES

1. L. WOLFGANG, *Railroad history*, San Diego Railroad Museum, March 2002.
2. W.S. CHAN, "Whole system simulator for AC traction", PhD Thesis, University of Birmingham, UK, July 1988.
3. Z. SHAO, "Auto-transformer power supply system for electric railways", PhD Thesis, University of Birmingham, UK, November 1988.
4. F.F. NOUVION, "Railway electrification technology – Technical paper", Indian Railways International Seminar and Exhibition on Railway Electrification, 1985.
5. R.J. HILL, "Electric railway traction – Part 2 traction drives with three-phase induction motors", Power Engineering Journal, pp. 143-152, June 1994.
6. R.J. HILL, "Electric railway traction – Part 1 electric traction and DC traction motor drives", Power Engineering Journal, pp. 47-56, February 1994.
7. R.J. HILL, "Electric railway traction – Part 3 traction power supplies", Power Engineering Journal, pp. 275-286, December 1994.
8. R.W. STURLAND, "Traction power supplies", GEC ALSTHOM Transmission & Distribution Projects Ltd.
9. R.W. WHITE, "AC supply systems and protection", Fourth Vocation School on Electric Traction Systems, IEE Power Division, April 1997.
10. H. BOZKAYA, "A comparative assessment of 50 kV auto-transformer and 25 kV booster transformer railway electrification systems", MPhil Thesis, University of Birmingham, UK, October 1987.
11. C. COURTOIS, "Why the 2x25 kV alternative?", Half-day Colloquium on 50 kV Auto-transformer Traction Supply Systems – The French Experience, IEE Power Division, pp. 1/1-1/4, November 1993.
12. O.L. ELGERD, *Electric Energy System Theory*, McGraw-Hill, London, 1983.
13. W. STEPHENSON, *Elements of power system analysis*, McGraw-Hill, London, 1982.
14. L. ELDEN: *Numerical Analysis: An Introduction*, Academic Press, Cambridge, 1990.
15. T. LIDEN, "The new train traffic simulation program developed for Banverket and its design", Int. Conf. on Computers in Railways III, COMPRAIL'92, August 1992, Washington DC, USA, pp. 533–544.
16. A. NYMAN, "TTS/SIMON Power Log – a simulation tool for evaluating electrical train power supply systems", Int. Conf. on Computers in Railways VI, COMPRAIL'98, September 1998, Lisbon, Portugal, pp. 427–436.
17. R.A. UHER, "Rail traction energy management model", Int. Conf. on Computers in Railways I, COMPRAIL'87, July 1987, Frankfurt, Germany, pp. 39–60.
18. B. MELLITT, C.J. GOODMAN, and R.I.M. ARTHURTON, "Simulator for studying operational and power supply conditions in rapid transit railways", Proc. IEE (125), 1978, pp. 298–303.
19. B. MELLITT, C.J. GOODMAN, and R.I.M. ARTHURTON, "Simulation studies of energy saving with chopper control on the Jubilee line", Proc. IEE (125), 1978, pp. 304–310.
20. M. McGUIRE, and D. LINDER, "Train simulation on British rail", Int. Conf. on Computers in Railways IV, COMPRAIL'94, September 1994, Madrid, Spain, pp. 437–444.
21. L. CUNHA, N. HENRIQUES, J. ESTEVES, and J. SANTANA, "Computer aided design tool for the study of electric traction fixed installations and train operation", Int. Conf. on Computers in Railways VI, COMPRAIL'98, September 1998, Lisbon, Portugal, pp. 479–487.
22. G. VAN-ALPHEN, V.W.M. VAN-RENS, and H.W.M. SMULDERS, "Analysis of AC traction power supplies using the SIMSPOG simulation tool", Int. Conf. on Computers in Railways VI, COMPRAIL'98, September 1998, Lisbon, Portugal, pp. 555–565.

23. J.P. AGRAWAL, *Power electronic systems – Theory and design*, Prentice Hall, New Jersy, USA, 2001.
24. W. SHEPHRED, L.N. HULLEY, and D.T.W. LIANG, *Power electronic and motor control*, 2nd Edition, Cambridge University Press, Cambridge, UK, 2000.
25. E ACHA, V.G. AGELIDIS, O. ANAYA-LARA, and T.J.E. MILLER, *Power electronic control in electrical systems*, Newnes Press, Oxford, UK, 2002.
26. Y.H. SONG, and A.T. JOHNS, *Flexible AC transmission systems (FACTS)*, The Institution of Electrical Engineers, London, UK, 1999.
27. L. HU, and R.E. MORRISON, "The stability and power flow limits of an electrified railway system with voltage controlled SVCs", 5th International Conference on Power Electronics and Variable-speed Drives, pp 562-567, London, UK, October 1994.
28. D. ZHANG, L. TONG, Z. YIN, and Z. WANG, "An analytical mathematical model for describing the dynamic behavior of the thyristor controlled series compensator", 1998 International Conference on Power System Technology, Vol. 1, pp. 420-424, Beijing, China, August 1998.
29. M.R. IRAVANI, P.L. DANDENO, K.H. NGUYEN, D. ZHU, and D. MARATUKULAM, "Applications of static phase shifters in power systems", IEEE Trans. on Power Delivery, Vol. 9, pp. 1600-1608, July 1994.
30. P. GARCIA-GONZALEZ, and A. GARCIA-CERRADA, "Control system for a PWM-based STATCOM", IEEE Trans. on Power Delivery, Vol. 15, No. 4, pp. 1252-1257, October 2000.
31. C. ZHAN, V.K. RAMACHADARAMURTHY, A. ARULAMPALAM, C. FITZER, S. KROMLODIS, M. BAMES, and N. JENKINS, "Dynamic voltage restorer based on voltage-space-vector PWM control", IEEE Trans. on Industry Applications, Vol. 37, pp. 1855-1863, November – December 2001.
32. W.D. YANG, Z. XU, and Z.X. HAN, "Co-ordinated hierarchical control strategy for multi-infeed HVDC systems", IEE Proceedings - Generation, Transmission and Distribution, Vol. 149, pp. 242-248, March 2002.
33. G. ASPLUND, "Application of HVDC Light to power system enhancement", Power Engineering Society Winter Meeting, Vol. 4, pp. 2498-2503, Singapore, January 2000.
34. M. TOUFAN, and U.D. ANNAKKAGE, "Simulation of the unified power flow controller performance using PSCAD/EMTPDC", Electric Power System Research, Vol. 46, pp. 67-75, 1998.
35. K.R. PADIYAR, and A.M. KULKARNI, "Control design of unified power flow controller", IEEE Trans. on Power Delivery, Vol. 13, No. 4, pp. 1348-1354, October 1998.
36. T. T. CHAN, "AI applications and solution techniques for AC railway system control and simulation", MPhil Thesis, Hong Kong Polytechnic, Hong Kong, March 1990.
37. B. YANG, "Study of single-phase pulse converters for feeding induction machine drives in railway traction application", PhD Thesis, University of Birmingham, UK, February 1992.
38. V. GALDI, C. J. GOODMAN, L. IPPOLITO and A. PICCOLO, "AC locomotive: Traction equipment modelling", 3rd International Scientific Conference in Drives and Supply Systems for Modern Electric Traction, pp. 9-14, Warsaw, Poland, September 1997.
39. P. HIS and S. CHEN, "Electric load estimation techniques for high-speed railway (HSR) traction power system", IEEE trans. on Vehicular Technology, Vol. 50, No. 5, pp. 1260-1266, September 2001.
40. R. TAKAGI, S. MIYAZAKI, N. UEDA, M. ASHIYA, and S. SONE, "Traction power control for optimization of AC railway power feeding systems", Int. conf. on Development in Mass Transit Systems, London, UK, April 1998.

41. Y. CAI, M.R. IRVING, and S.H. CASE, “Compound matrix partitioning and modification for the solution of branched autotransformer traction feeds”, IEE Proc.-Electr. Power Appl. (143), pp. 251–157, 1996.
42. G.W. STAGG, and A.H. EL-ABAID, *Computer methods in power system analysis*, McGraw-Hill, 1968.
43. J. YU, “Computer analysis of touch voltages and stray currents for DC railways”, PhD Thesis, University of Birmingham, UK, 1992.
44. D. YANG, *C++ and objected-oriented numeric computing for scientists and engineers*, Springer, 2001.
45. L. AMMERAAL, *C++ for programmers*, Johns Wiley & Sons, 2000.
46. MATHWORKS Inc., *Using MATLAB version 6*, User’s manual CD-ROM, 2001.
47. R.D. ZIMERMAN, and D. GAN, “MATPOWER’s User Manual: A MATLAB Power System Simulation Package”, School of Electrical Engineering, Cornell University, New York, USA, 1997.
48. W.H. PRESS, S.A. TEUKOLSKY, W.T. VETTERLING, and B.P. FLANNERY, *Numerical recipes in C*, Cambridge University Press, 1999.
49. L.V. FAUSETT, *Applied numerical analysis using Matlab*, Prentice-Hall, 1999.
50. L.K. SUI, “An object-oriented railway system and power network simulator”, PhD Thesis, University of Birmingham, UK, April 1995.
51. A. GREENBAUM, *Iterative methods for solving linear systems*, SIAM, 1997.
52. G. ENGELN-MUELLGES, F. UHLIG, and M. SCHON, *Numerical algorithms with C*, Springer, 1996.
53. M. HESTENES, and E. STIEFEL, “Methods of conjugate gradients for solving linear systems”, J. Res. Nat. Bur. Stand., 49, pp. 409-436, 1952.
54. A. GEORGE, and J.W. LIU, *Computer solution of large sparse linear positive definite systems*, Prentice-Hall, 1981.
55. C.J. GOODMAN, and L.K. SIU, “DC railway power network solutions by diakoptics”, ASME/IEEE Joint Railroad Conf., March 1994, Chicago, USA, pp. 103–110.
56. L.K. SIU, and C.J. GOODMAN, “An object-oriented power network simulator for multi-train simulations”, Int. Conf. on Computers in Railways, COMPRAIL’94, September 1994, Madrid, Spain, pp. 483–490.
57. C.J. GOODMAN, L.K. SIU, and T.K. HO, “A review of simulation models for railway systems”, IEE Int. Conf. on Developments in Mass Transit Systems, April 1998, London, UK, pp. 80–85.
58. ABB Power Transmission Company, *Multi-purpose SVC for Queensland 275/132 kV Network*, Pamphlet A02-0116E.
59. ABB Power Transmission Company, *Multiple SVC installations for traction load balancing in Central Queensland*, Pamphlet A02-0134E.
60. CANMET Energy Technology Centre, *Adaptive Var Compensator*, Natural Resources Canada.
61. J.M.W. WHITING, and H.J. THOMPSON, “Power converters”, Seventh Residential Course on Electric Traction Systems, IEE Railway Professional Network, October 2002.
62. C.J. GOODMAN, *Lecture notes on PWM power converter*, Power & Control Research Group, School of Engineering, University of Birmingham.
63. A.J. DURIE, “Fixed link DC voltage converter with GTO devices for single phase traction drives”, MPhil Thesis, University of Birmingham, UK, May 1987.
64. A. BELLINI, S. BIFARETTI, and S. COSTANTINI, “High power factor converters for single-phase AC traction drives”, Int. Conf. on Computers in Railways VIII, COMPRAIL’02, June 2002, Lemnos, Greece, pp. 559–568.

65. J.J. MCKEOWN, D. MEEGAN, D. SPREVAK, *An introduction to unconstrained optimisation*, Adam Hilger, Bristol, UK, 1990.
66. E.K.P. CHONG, S.H. ZAK, *An introduction to optimization*, Wiley-interscience, New York, USA, 2001.
67. J. NOCEDAL, AND S.J. WRIGHT, *Numerical optimization*, Springer, London. UK, 1999.
68. C.J. GOODMAN, "Train performance and simulation", Fourth Vocation School on Electric Traction Systems, IEE Power Division, April 1997.
69. S. SUJITJORN, B. MELLITT, and N.B. RAMBUKWELLA, "Energy minimisation in transit systems using a train based expert system for dynamic coast control", Int. Conf. on Computers in Railways I, COMPRAIL'87, July 1987, Frankfurt, Germany, pp. 303–316.
70. C.J. GOODMAN, and Z. ZHANG, "BR crossrail tunnel energy consumption study", Interim Report to British Rail, Traction Research Group, School of Electronic & Electrical Engineering, University of Birmingham, UK, 1992.
71. P.O. BARNWELL, "BR networker 1 optimisation study", Interim Report on Simulation Results, Traction Research Group, School of Electronic & Electrical Engineering, University of Birmingham, UK, 1987.
72. B. MELLITT, "System in common use", Fourth Vocation School on Electric Traction Systems, IEE Power Division, April 1997.
73. M.A. SHARDLOW, "DC motor control", Seventh Residential Course on Electric Traction Systems, IEE Railway Professional Network, October 2002.
74. A. PANKHURST, and H.J. THOMPSON, "Control of induction motors", Seventh Residential Course on Electric Traction Systems, IEE Railway Professional Network, October 2002.
75. R.J. HILL, "DC and AC traction motors", Seventh Residential Course on Electric Traction Systems, IEE Railway Professional Network, October 2002.
76. N. WEBSTER, "Electric multiple units", British Railways Pocket Book No. 4, Platform 5 Publishing Ltd., 2001.
77. M.C. DUFFY, *Electric railways 1880–1990*, Institute of Electrical Engineers, 2003.
78. A.J. GRIFFIN, "Methods of improving the voltage regulation on 25 kV electrified railways", Int. Conf. on Main Line Electrification, September 1989, York, UK, pp. 252–259.
79. T. KNESCHKE, and W. NAQWI, "Upgrade of SETA's regional rail power system", 1997 IEEE/ASM Joint Railroad Conference, March 1997, Boston, USA, pp. 35 – 47.
80. J.P. GUILLOUX, "High speed signalling in France", Int. Conf. on Main Line Electrification, September 1989, York, UK, pp. 397–402.
81. C.S. CHEN, H.J. CHUANG, and L.J. FAN, "Design of main transformer capacity for electrified railway power systems", Int. Conf. on Power System Technology, PowCon 2000, December 2000, Perth, Australia, pp. 311–316.
82. R.E. MORRISON, K. WARBURTON, D.J. YOUNG, and D. HACKWELL, "The use of static shunt compensation to upgrade existing electrified railways", Int. Conf. on Main Line Electrification, September 1989, York, UK, pp. 232–235.
83. B. CHEN, and J.M. KOKERNAK, "Thyristor controlled two-stage magnetic-valve reactor for Dynamic Var-compensation in electric railway power supply systems", Fifteen Annual IEEE Applied Electronics Conference and Exposition, APEC2000, February 2000, New Orleans, USA, pp. 1066–1072.
84. L. HU, and R.E. MORRISON, "The stability and power flow limits of an electrified railway system with voltage controlled SVCs", Fifth Int. Conf. on Power Electronics and Variable-speed Drives, October 1994, London, UK, pp. 562–567.

85. E. REZANIA, and S.M. SHAHIDEHPOUR, " Real power loss minimisation using interior point method", Electrical Power & Energy System, Elsevier Science, Vol. 23, pp. 45–56, 2001.
86. N.I. DEEB, and S.M. SHAHIDEHPOUR, "Decomposition approach for minimising real power losses in power systems", IEE Proceedings - Generation, Transmission and Distribution, Vol. 138, pp. 27–38, January 1991.
87. J. ZHU, and J.A. MOMOH, "Optimal Var pricing and Var placement using analytic hierarchical process", Electric Power Systems Research, Elsevier Science, Vol. 48, pp. 11–17, 1998.
88. M.E. BARAN, and F.F. WU, "Optimal capacitor placement on radial distribution systems", IEEE Trans. on Power Delivery, Vol. 4, pp. 725–734, 1989.
89. J.C. CARLISLE, and A.A. EL-KEIB, "A graph search algorithm for optimal placement of fixed and switched capacitors on radial distribution systems", IEEE Trans. on Power Delivery, Vol. 15, pp. 423–428, 2000.
90. A. KALYUZHNY, G. LEVITIN, D. ELMAKIS, and H. BEN-HAIM, "System approach to shunt capacitor allocation in radial distribution systems", Electric Power Systems Research", Elsevier Science, Vol. 56, pp. 51–60, 2000.
91. T.S. CHUNG, and G. SHAOYUN, "A recursive LP-based approach for optimal capacitor allocation with cost-benefit consideration", Electric Power Systems Research", Elsevier Science, Vol. 39, pp. 129–136, 1997.
92. S.K. GOSWAMI, T. GHOSE, and S.K. BASU, "An approximate method for capacitor placement in distribution system using heuristics and greedy search technique", Electric Power Systems Research", Elsevier Science, Vol. 51, pp. 143–151, 1997.
93. E.B. MAKRAM, D. LUKE, and W.P. ADAMS, "A new technique for optimal size and location of capacitor banks in the presence of harmonics and distortion", Elsevier Science, Vol. 34, pp. 149–156, 1997.
94. ABB MEDIUM VOLTAGE PRODUCT, *HiQ Capacitor Unit*, Power Capacitor Technology, 2003. (http://www.abb.com)
95. C.H. PAPADIMITRIOU, and K. STEIGLITZ, "Combinatorial optimization algorithms and complexity", Dover Publication Inc., New York, 1998.
96. D.E. GOLDBERG, and D. EDWARD, "Genetic algorithms in search, optimization and machine learning", Addison-Wesley, 1989.
97. K.Y. LEE, X. BAI, and Y. PARK, "Optimization method for reactive power planning by using a modified simple genetic algorithms", IEEE Trans. on Power Systems, Vol. 10, No. 4, pp. 1843–1850, 1995.
98. C. SU, G. RII, and C. TSAI, "Optimal capacitor allocation using fuzzy reasoning and genetic algorithms for distribution systems", Mathematical and computer modelling, Elsevier Science, Vol. 33, pp. 745–757, 2001.
99. D.B. FOGEL, *Evolutionary computation toward a new philosophy of machine intelligence*, IEEE Press, New York, 1995.
100. L. DAVID, *Genetic algorithms and simulated annealing*, Pitman, London, 1987.
101. Electricity Association Services Ltd., (http://www.electricity.org.uk)
102. S. WALL, and D. McSHANE, "A strategy for low-cost utility connection of battery energy storage systems", Journal of Power Sources, Elsevier Science, Vol. 67, pp. 193–200, 1997.
103. R. Robinson, U. Danielson, and M. Snaith, Road maintenance management: Concepts and systems, Macmillan, London, 1998.
104. V.N. VEDAMURTHY, and N.C.S.N. LYENGAR, *Numerical methods*, Vikas Publishing PVT Ltd, New Delhi, 2000.

APPENDIX A

IMPEDANCE AND ADMITTANCE MATRICES FOR A MULTI-CONDUCTOR SYSTEM

A.1 Multi-conductor approach for AC railways [2,3,10]

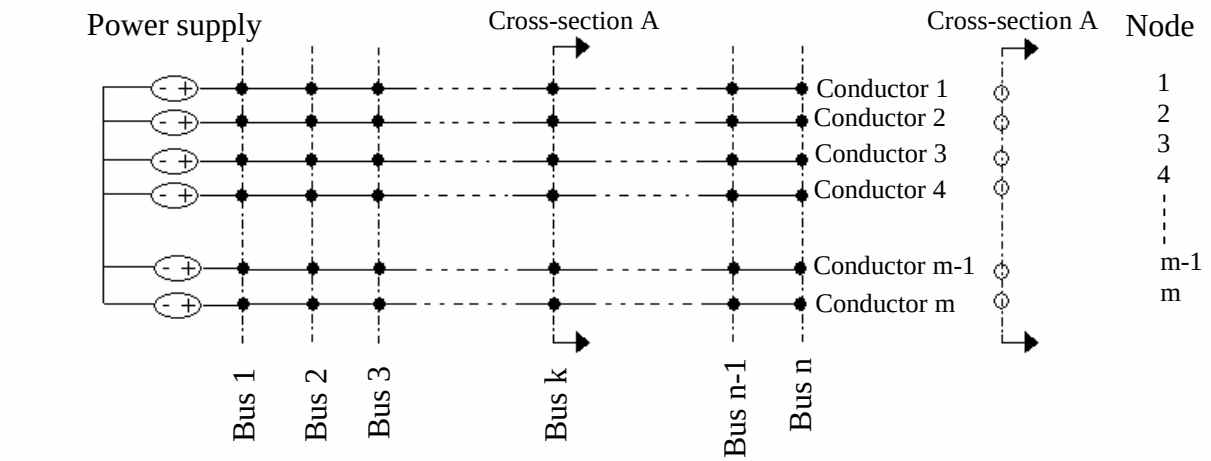


Fig. A.1: Multi-conductor system

One of the simplest and the most powerful methods for the power system analysis is based on a lumped network model. This method breaks the whole power system into small pieces of network components in the frequency domain. The general concept of the multi-conductor approach will be described following the main assumptions below.

1. The supply source is treated as an ideal sinusoidal voltage source in series with an impedance
2. Along the power feeding length, the so-called “Bus” is defined as a particular position and consists of many conductor nodes, where all conductors are perpendicularly penetrated by a cross-section plane as shown in Fig. A.1
3. The overhead power feeding system is treated as lumped parameters in every section and all conductors are considered as parallel laid wires along the feeding length

4. The feeding length is relatively short when compared with the electric wavelength and thus the propagation effects along the conductors are neglected
5. Summation of all conductor currents including the earth-leakage current is equal to zero
6. There is no mutual coupling between conductors of different feeding sections
7. Distributed shunt capacitance between conductors are all neglected
8. For AC railway power systems, rail-to-earth ballast resistance is assumed to be a lumped parameter concentrated at all the buses

Based on the assumptions made above, the system can be transformed into a linear multi-conductor model for each particular frequency. The linear equation that describes the system can be simply expressed as

$$[\mathbf{Y}][\mathbf{V}] = [\mathbf{J}] \tag{A1}$$

A group of m conductors results in the self and mutual impedance at each feeding section. In the autotransformer AC railway power feeding system described in this thesis, there are four main conductors to be considered.

1. *Catenary* (C): The overhead wire consists of a contact wire and one or two messenger wires. They share the current and are always at the same potential. To reduce the system complexity, their impedances are combined all together to form a single conductor that is the catenary used in this thesis.

2. *Rails* (R): The two rails in the same track are also treated as a single cylindrical conductor. They are connected to autotransformers at the centre tap.

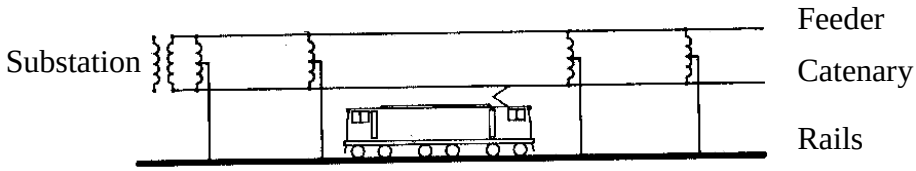
3. *Feeder* (F): Besides the catenary, another outer tap of the autotransformer is connected to the feeder wire.

4. *Earth* (E): Although there is a wire called “Earth wire” that is used for safety reasons and normally carries no current, it is ignored for the calculation of the matrix impedance in

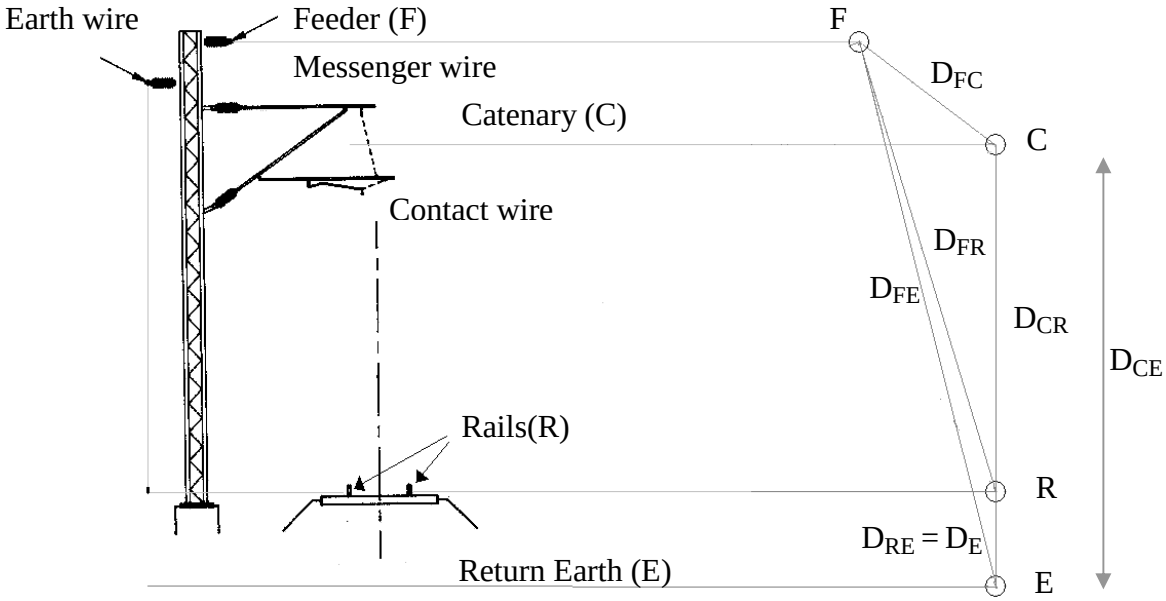
normal operation. It is assumed that the earth-leakage current from the rails flows underneath the track with a depth of D_E through a virtual conductor called the return earth, where $D_E = 655\sqrt{\frac{\rho_E}{f}}$ m, f is the operating frequency (Hz), ρ_E is the earth resistivity (Ω·m) and $\rho_E = 547f^{-0.434}$. This leads the presence of the return earth as the reference conductor.

The geometrical calculation of the impedance matrix for AC railway power feeding system can be illustrated in the next sub-section.

A.2 Geometrical calculation of the impedance for an AT railway power feeding system



a) Autotransformer feeding system



b) Overhead configuration

Fig. A.2: Geometry of the overhead AT railway power feeding system

Although there are four conductors, the return earth can be eventually eliminated due to the assumption that the return earth is set as the reference potential (0 V). The impedance matrix to represent a feeding section connected between two buses, i and j, can be expressed as in equation A.2.

$$\mathbf{Z}_{ij} = \begin{bmatrix} z_{CC} & z_{CR} & z_{CF} \\ z_{RC} & z_{RR} & z_{RF} \\ z_{FC} & z_{FR} & z_{FF} \end{bmatrix} \quad \Omega/m \quad \text{(A2)}$$

The self and mutual impedance is calculated by the following equations

$$z_{ii} = \frac{\rho_i}{A_i} + j2\pi f\mu_0\left(\frac{\mu_i}{8\pi} + \frac{1}{2\pi} ln \frac{D_{iE}}{r_i}\right) \quad \Omega/m \quad \text{(A3)}$$

$$z_{RR} = \frac{\rho_R}{A_R} + j2\pi f\mu_0\left(\frac{\mu_R}{8\pi} + \frac{1}{2\pi} ln \frac{D_{RE}}{r_R} + \frac{1}{2\pi} ln \frac{D_{RE}}{D_{RR}}\right) \quad \Omega/m \quad \text{(A4)}$$

$$z_{ij} = j\mu_0 f\, ln \frac{D_{iE}}{D_{ij}} \quad \Omega/m \quad \text{(A5)}$$

where

$i,j \in \{C,R,F\}$

r_i is the radius of conductor i (m), GMR is used for a group of conductor, e.g. rails.

A_i is the cross-sectional area of conductor i (m^2)

D_{iE} is the distance between conductor i and the return earth (m)

D_{ij} is the distance between conductor i and j (m)

D_{RR} is the distance between the two rails called the rail gauge (m)

ρ_i is the resistivity of the conductor i ($\Omega \cdot m$)

μ_i is the permeability of the conductor i (H/m)

$\mu_0 = 4\pi \times 10^{-7}$ H/m

Although equation A3 is adequately used to calculate the self-impedance of the rails, it is more accurate to include the effect of the rail gauge for this calculation as shown in equation A4.

A.3 Autotransformer model

z_1, z_2 are AT's primary and secondary leakage impedances

z_m is AT's magnetising impedance

I_m is AT's magnetising current

I_1, I_2 are AT's primary and secondary current

E_1, E_2 are electromotive forces of AT's primary and secondary windings

N_1, N_2 are turns of the two windings

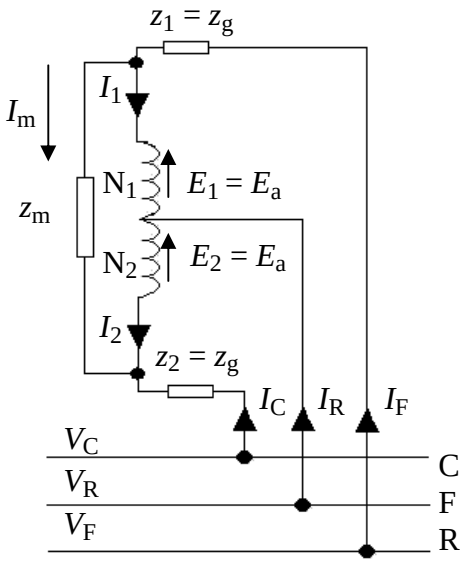


Fig. A.3: Equivalent circuit of an autotransformer

Assume that $N_1 = N_2$ and $z_1 = z_2 = z_g$, therefore

$$E_1 = E_2 = E_a = \frac{1}{2} z_m I_m \tag{A6}$$

$$I_m = \frac{1}{2}(I_F - I_C) \tag{A7}$$

$$I_C + I_R + I_F = 0 \tag{A8}$$

$$V_R - V_C = E_a - z_g I_C \tag{A9}$$

$$V_F - V_R = E_a + z_g I_F \tag{A10}$$

From equations A7 and A8,

$$I_C = -I_m - \frac{1}{2} I_R \tag{A11}$$

$$I_F = I_m - \frac{1}{2} I_R \tag{A12}$$

From equations A9 and A10,

$$V_F - V_C = 2E_a + z_g(I_F - I_C) = z_m I_m + 2z_g(I_F - I_C) = I_m(z_m + 2z_g) \tag{A13}$$

$$V_C + V_F - 2V_R = z_g(I_F + I_C) = -\frac{1}{2} z_g I_R \tag{A14}$$

Substitute equation A11 into A13, equation A12 into A14,

$$I_C = \left(\frac{1}{2z_g} - \frac{1}{z_m + 2z_g}\right) V_C - \frac{1}{z_g} V_R + \left(\frac{1}{2z_g} + \frac{1}{z_m + 2z_g}\right) V_F \tag{A15}$$

$$I_R = -\frac{1}{z_g} V_C + \frac{2}{z_g} V_R - \frac{1}{z_g} V_F \tag{A16}$$

$$I_F = \left(\frac{1}{2z_g} - \frac{1}{z_m + 2z_g}\right) V_C - \frac{1}{z_g} V_R + \left(\frac{1}{2z_g} + \frac{1}{z_m + 2z_g}\right) V_F \tag{A17}$$

Thus,

$$\mathbf{Y}_{AT} = \begin{bmatrix} \frac{1}{2z_g} - \frac{1}{z_m + 2z_g} & -\frac{1}{z_g} & \frac{1}{2z_g} + \frac{1}{z_m + 2z_g} \\ -\frac{1}{z_g} & \frac{2}{z_g} & -\frac{1}{z_g} \\ \frac{1}{2z_g} - \frac{1}{z_m + 2z_g} & -\frac{1}{z_g} & \frac{1}{2z_g} + \frac{1}{z_m + 2z_g} \end{bmatrix} \tag{A18}$$

A.4 Power substation model

z_0 = Short circuit impedance of the high voltage grid

V_0 = Nominal voltage

z_1, z_2 = Impedance of the primary and secondary windings

z_e = Impedance connected between the centre tap of the 2nd winding and the rails

a = Transformer's turn ratio

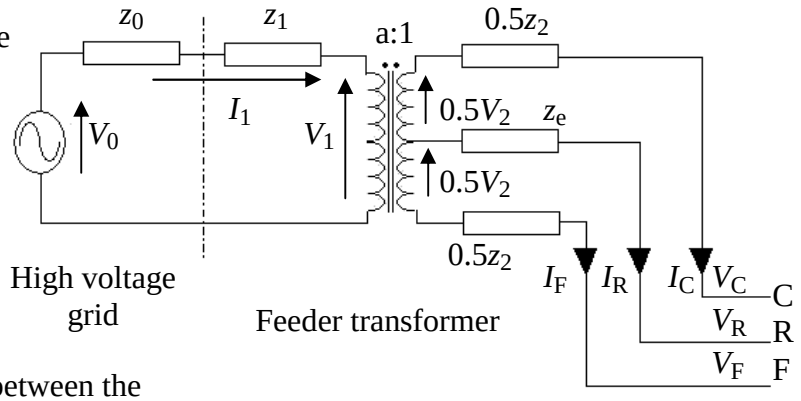


Fig. A.4: Equivalent circuit of a power substation

I_1 is the primary current of the transformer caused by two secondary current I_C and I_F as follows.

$$I_1 = \frac{1}{2a}\left(-I_C + I_F\right) \tag{A19}$$

Consider the primary-side circuit, from equation A19, therefore,

$$V_1 = V_0 - \frac{1}{2a}\left(z_0 + z_1\right)\left(-I_C + I_F\right) \tag{A20}$$

and the secondary-side C-R circuit,

$$V_C - V_R + \frac{1}{2}V_2 + \frac{1}{2}z_2\, I_C + z_e\left(I_C + I_F\right) = 0 \tag{A21}$$

Substitute $V_2 = 0.5V_1$ where V_1 is as in equation A20, hence,

$$\left(z_1' + 2\,z_2 + 4\,z_e\right)I_C + \left(4\,z_e - z_1'\right)I_F = -4V_C + 4V_R - 2V_0' \tag{A22}$$

Similarly, the secondary-side F-R circuit can be expressed as

$$\left(4\,z_e - z_1'\right)I_C + \left(z_1' + 2\,z_2 + 4\,z_e\right)I_F = 4V_R - 4V_F + 2V_0' \tag{A23}$$

where

$z_1' = z_0 + z_1$.

Based on the assumption that all currents are supplied by the substation and eventually returned to the substation, that is

$$I_C + I_R + I_F = 0 \tag{A24}$$

By defining $z_A = z_1' + z_2$ and $z_B = z_2 + 4z_e$, thus,

$$(z_A + z_B)I_C + (-z_A + z_B)I_F = -4V_C + 4V_R - 2V_0' \tag{A25}$$

$$(-z_A + z_B)I_C + (z_A + z_B)I_F = 4V_R - 4V_F + 2V_0' \tag{A26}$$

With equations A24-A26, the equivalent voltage source in series with the impedance or the so-called Thevenin equivalent circuit in the multi-conductor model is formed as shown in equation A27.

$$\begin{bmatrix} z_A + z_B & 0 & -z_A + z_B \\ 1 & 1 & 1 \\ -z_A + z_B & 0 & z_A + z_B \end{bmatrix} \begin{bmatrix} I_C \\ I_R \\ I_F \end{bmatrix} = 2V_0' \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} + 4 \begin{bmatrix} -1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} V_C \\ V_R \\ V_F \end{bmatrix} \tag{A27}$$

Since

$$\begin{bmatrix} z_A + z_B & 0 & -z_A + z_B \\ 1 & 1 & 1 \\ -z_A + z_B & 0 & z_A + z_B \end{bmatrix}^{-1} = \begin{bmatrix} \frac{z_A + z_B}{4 z_A z_B} & 0 & \frac{z_A - z_B}{4 z_A z_B} \\ -\frac{1}{2 z_B} & 1 & -\frac{1}{2 z_B} \\ \frac{z_A - z_B}{4 z_A z_B} & 0 & \frac{z_A + z_B}{4 z_A z_B} \end{bmatrix} \tag{A28}$$

equation A27 can be written in the other form as the Norton equivalent circuit as shown in equation A29.

$$\begin{bmatrix} I_C \\ I_R \\ I_F \end{bmatrix} = -\frac{V_0'}{z_A} \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} + \begin{bmatrix} \frac{1}{z_A} + \frac{1}{z_B} & \frac{2}{z_B} & \frac{1}{z_A} - \frac{1}{z_B} \\ \frac{2}{z_B} & -\frac{4}{z_B} & \frac{2}{z_B} \\ \frac{1}{z_A} - \frac{1}{z_B} & \frac{2}{z_B} & \frac{1}{z_A} + \frac{1}{z_B} \end{bmatrix} \begin{bmatrix} V_C \\ V_R \\ V_F \end{bmatrix} \tag{A29}$$

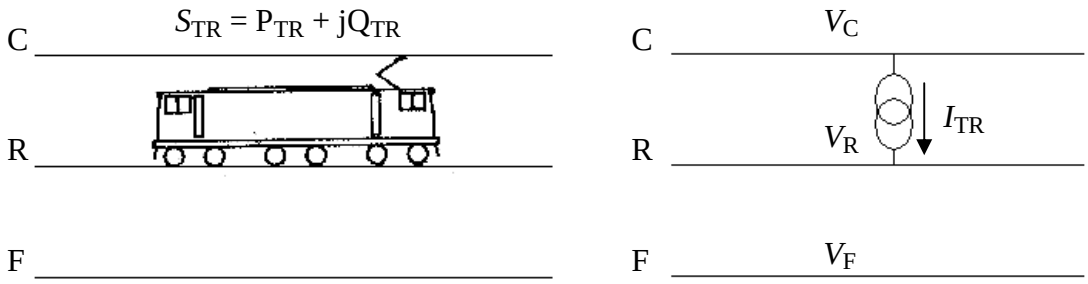
or $\mathbf{I}_{SS} = -\mathbf{J}_{SS} + \mathbf{Y}_{SS}\mathbf{V}_{SS}$

where

$\mathbf{J}_{SS}$ is the substation current-injected model

$\mathbf{V}_{SS}$ and $\mathbf{Y}_{SS}$ are the substation voltage and the substation admittance, respectively

A.5 Train or locomotive model



a) Train drawing power from the catenary b) Current model

Fig. A.5: Train and its equivalent circuit

The train current can be computed by using the train power and the pantograph voltage as follows.

$$I_{TR} = \left(\frac{S_{TR}}{V_C - V_R} \right)^* \tag{A30}$$

Therefore,

$$I_C = I_{TR}$$

$$I_R = -I_{TR}$$

$$I_F = 0$$

Thus, the current load model representing the train in the multi-conductor system is simply expressed as equation A31.

$$\begin{bmatrix} I_C \\ I_R \\ I_F \end{bmatrix} = I_{TR} \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix} \tag{A31}$$

APPENDIX B

NEWTON-RAPHSON METHOD FOR SOLVING NON-LINEAR EQUATIONS

B.1 Newton-Raphson Method [48,48,103]

Perhaps, the most powerful root-finding methods for a set of non-linear equations is Newton's method or the so-called Newton-Raphson method. It requires the evaluation of both the function f(**x**) and the derivative ∇f(**x**) at an arbitrary point **x**. At a current point $\mathbf{x}_k$, it extends the tangent line until the line crosses zero. Then, set the next guess $\mathbf{x}_{k+1}$ to the abscissa of the zero crossing as shown in Fig. B.1.

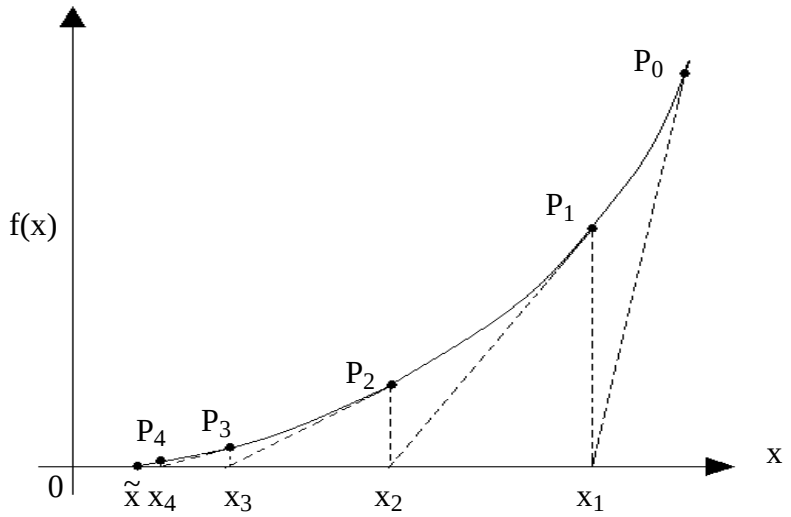


Fig. B.1: Geometrical interpretation for the Newton-Raphson method in a single-variable problem, where $\tilde{x}$ denotes a local minimum

Let $f_i(\mathbf{x})$, where i = 1, 2, …, n, be a set of functions to be zeroed. Starting with an initial point $\mathbf{x}_k$, where k = 0, a better approximation of the root is at $\mathbf{x}_k + \Delta\mathbf{x}_k$, which expectedly leads $f_i(\mathbf{x}_k + \Delta\mathbf{x}_k) = 0$. In the neighbourhood of **x**, each of the functions f_i can be expanded in Taylor series as

$$f_i(\mathbf{x}_k + \Delta\mathbf{x}_k) = f_i(\mathbf{x}_k) + \nabla f_i(\mathbf{x}_k)\Delta\mathbf{x}_k + O(\Delta\mathbf{x}) \tag{B1}$$

The matrix of partial derivatives appearing in the equation above is the Jacobian matrix **J**. By neglecting terms of order $\Delta\mathbf{x}^2$ and higher together with setting $f_i(\mathbf{x}_k + \Delta\mathbf{x}_k) = 0$, the approximation, $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}_k$, that moves each function towards zero is expressed as follows.

$$\Delta\mathbf{x}_k = -\mathbf{J}^{-1}\mathbf{F}(\mathbf{x}_k) \tag{B2}$$

where

$$\mathbf{F}(\mathbf{x}_k) = [f_1(\mathbf{x}_k) \quad f_2(\mathbf{x}_k) \quad \cdots \quad f_n(\mathbf{x}_k)]^T$$

This process will be iterated until it converges. In general, it is a good idea to check the change in both functions and variables.

APPENDIX C

AC/DC PWM POWER CONVERTER

Fig. C.1 illustrates a single-phase AC/DC PWM power converter. Each switch of the converter consists of a high-speed, high-power and self-commutated device, such as GTO, IGBT, etc, and an anti-parallel diode, so the current can flow in either direction.

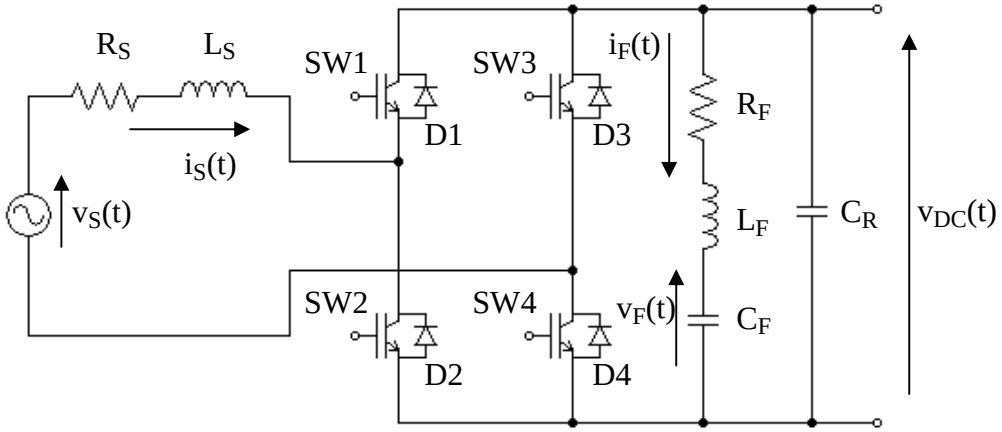
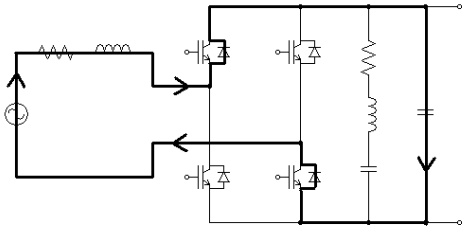


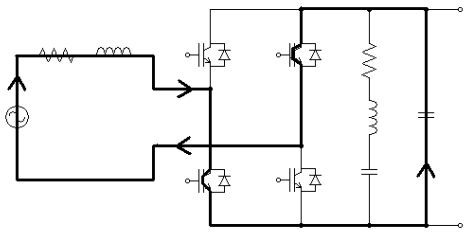
Fig. C.1: AC/DC PWM power converter

By turning the switches on and off with a well designed modulation strategies, the DC link voltage can be regulated while a near sinusoidal input current is drawn from or fed back to the supply with a range of adjustable power factor. In a single-phase converter, a DC link resonant filter is required in addition to the DC link capacitor, C_R, to eliminate the second order harmonic component caused by the DC link voltage ripple.

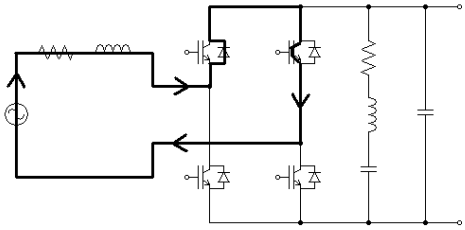
There are eight modes of operation for the single-phase PWM power converter depending on power supply and switching condition as shown in Figs C.2a–C.2h.



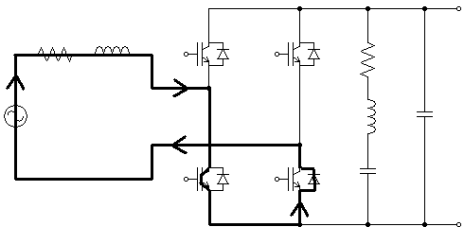
a) Mode 1, rectifying



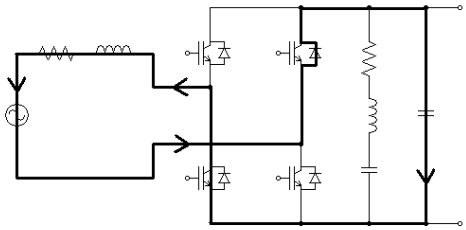
b) Mode 2, inverting



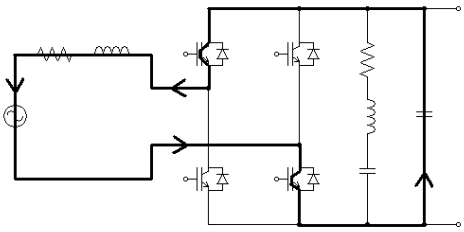
c) Mode 3, zeroing



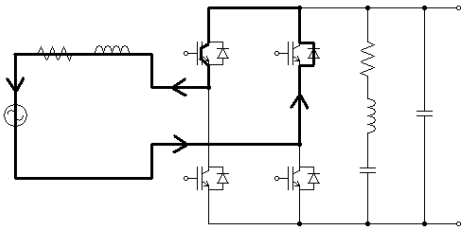
d) Mode 4, zeroing



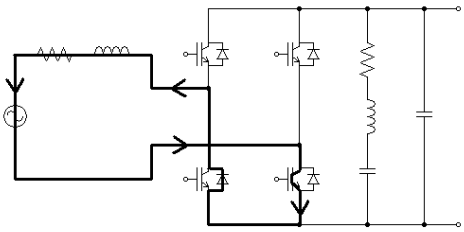
e) Mode 5, inverting



f) Mode 6, rectifying



g) Mode 7, zeroing



h) Mode 8, zeroing

Fig. C.2: Eight operating modes of the single-phase PWM power converter

The equivalent circuit for each mode is shown in Fig. C.2. Performances of the PWM power converter depends on the modulation techniques, which are to control on/off switching period in order to optimise some particular objective, such as power factor, harmonics, etc. Although there are eight modes as shown in the figure, they are divided into three modes of operation: i) rectifying mode, ii) inverting mode and iii) zero mode. To determine which mode will be used during the process is relied on the design of the modulation control strategies, such as natural sampling or regular sampling, unipolar or bipolar, etc. The derivation of the converter behaviour for each mode is mathematically summarised as follows.

- **Rectifying mode**

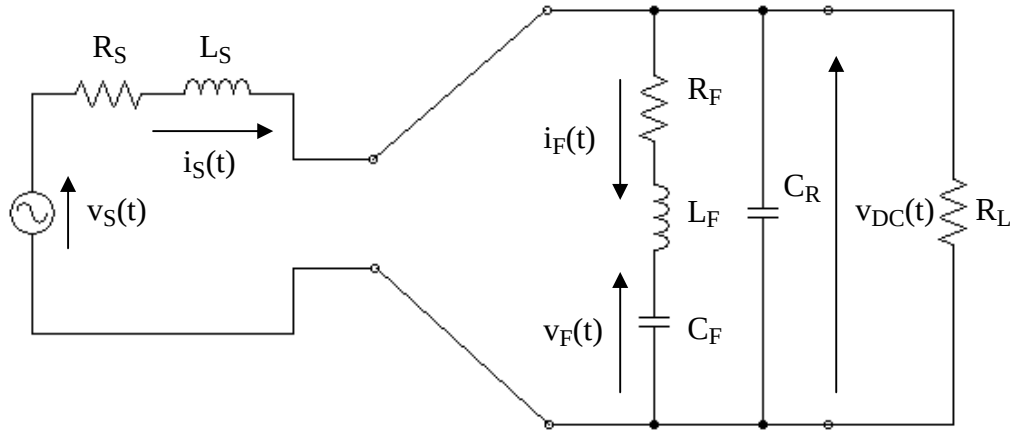


Fig. C.3: Rectifying mode of operation

Briefly, the set of equations, which characterises the rectifying mode, is shown in equation C.1.

$$\begin{bmatrix} \frac{d}{dt} i_S \\ \frac{d}{dt} v_{DC} \\ \frac{d}{dt} i_F \\ \frac{d}{dt} v_F \end{bmatrix} = \begin{bmatrix} -R_S/L_S & -1/L_S & 0 & 0 \\ 1/C_R & -1/R_L C_R & -1/C_R & 0 \\ 0 & 1/L_F & -R_F/L_F & -1/L_F \\ 0 & 0 & 1/C_F & 0 \end{bmatrix} \begin{bmatrix} i_S \\ v_{DC} \\ i_F \\ v_F \end{bmatrix} + \begin{bmatrix} 1/L_S \\ 0 \\ 0 \\ 0 \end{bmatrix} [v_S] \qquad \text{(C.1)}$$

- **Inverting mode**

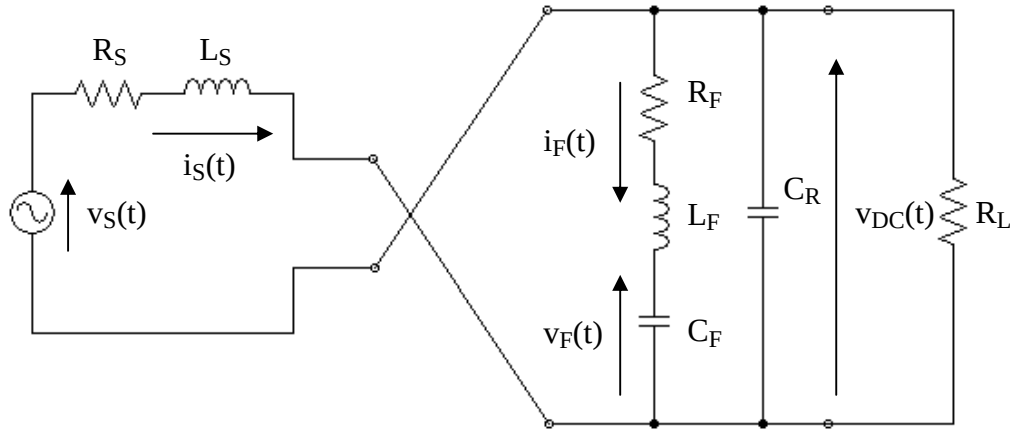


Fig. C.4: Inverting mode of operation

The set of equations, which characterises the inverting mode, is shown in equation C.2.

$$\begin{bmatrix} \frac{d}{dt} i_S \\ \frac{d}{dt} v_{DC} \\ \frac{d}{dt} i_F \\ \frac{d}{dt} v_F \end{bmatrix} = \begin{bmatrix} -R_S/L_S & 1/L_S & 0 & 0 \\ -1/C_R & -1/R_L C_R & -1/C_R & 0 \\ 0 & 1/L_F & -R_F/L_F & -1/L_F \\ 0 & 0 & 1/C_F & 0 \end{bmatrix} \begin{bmatrix} i_S \\ v_{DC} \\ i_F \\ v_F \end{bmatrix} + \begin{bmatrix} 1/L_S \\ 0 \\ 0 \\ 0 \end{bmatrix} [v_S] \qquad \text{(C.2)}$$

- **Zero mode**

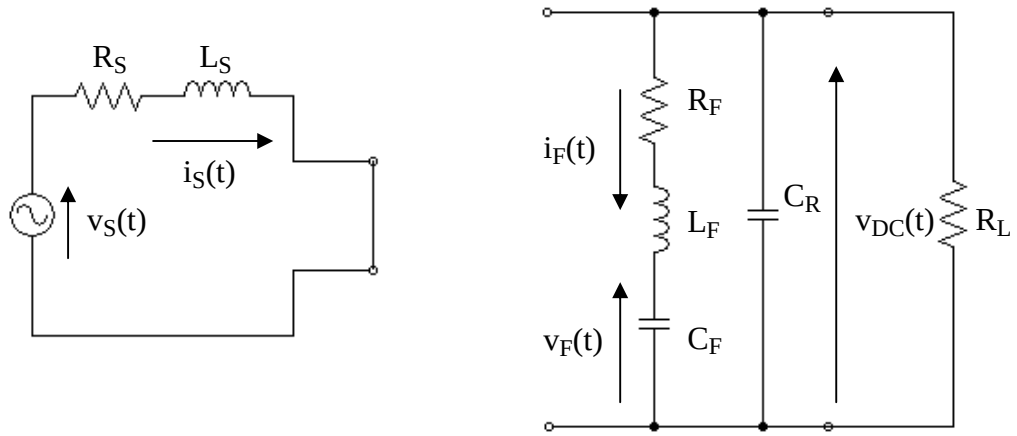


Fig. C.5: Zero mode of operation

The set of equations, which characterises the zero mode, is shown in equation C.3.

$$\begin{bmatrix} \frac{d}{dt} i_S \\ \frac{d}{dt} v_{DC} \\ \frac{d}{dt} i_F \\ \frac{d}{dt} v_F \end{bmatrix} = \begin{bmatrix} -R_S/L_S & 0 & 0 & 0 \\ 0 & -1/R_L C_R & -1/C_R & 0 \\ 0 & 1/L_F & -R_F/L_F & -1/L_F \\ 0 & 0 & 1/C_F & 0 \end{bmatrix} \begin{bmatrix} i_S \\ v_{DC} \\ i_F \\ v_F \end{bmatrix} + \begin{bmatrix} 1/L_S \\ 0 \\ 0 \\ 0 \end{bmatrix} [v_S] \qquad \text{(C.3)}$$

This is only a brief description of the single-phase PWM power converter. More detail can be found in References 37,62 and 63.

APPENDIX D

OPTIMISATION TECHNIQUES

D.1 Overview

Consider an optimisation problem described below.

Minimise $f(\mathbf{x})$

Subject to $\mathbf{x} \in \Omega$

The function $f(\mathbf{x})$ that is to be minimised is a real-valued function and called the objective function or a cost function. The vector $\mathbf{x}$ is an n dimensional vector, that is, $x \in \Re^n$, where $\Re$ is a set of real numbers. The set Ω is a subset of $\Re^n$, which is called the constraint set or feasible set.

The optimisation problem above is the decision problem that involves finding the best of vector $\mathbf{x}$ that gives the smallest value of the objective. This vector is called the minimiser of $f(\mathbf{x})$ over Ω. In addition, there is another form of problem that requires maximisation of the objective. However, this problem can be transformed into the above mentioned minimisation problem by minimising $-f(\mathbf{x})$, $\frac{1}{f(\mathbf{x})}$ where $f(\mathbf{x}) \neq 0$, $\log f(\mathbf{x})$, etc.

The problem above is a constrained optimisation problem. If $\Omega = \Re^n$, then this problem is as an unconstrained optimisation problem. In considering the general optimisation problem, the following definitions are specified.

- *Definition 1*: (Local minimiser). Suppose that $f(\mathbf{x}): \Re^n \to \Re$ is a real-valued function, which is defined on some set $\Omega \subset \Re^n$. A point $\tilde{\mathbf{x}} \in \Omega$ is a local minimiser of $f(\mathbf{x})$ over Ω if there exists $\varepsilon > 0$ such that $f(\mathbf{x}) \geq f(\tilde{\mathbf{x}})$ for all $\mathbf{x} \in \Omega$ and $\|\mathbf{x} - \tilde{\mathbf{x}}\| < \varepsilon$.

- *Definition 2*: (Global minimiser). A point $\tilde{\mathbf{x}} \in \Omega$ is a global minimiser of f(**x**) over Ω if $f(\mathbf{x}) \geq f(\tilde{\mathbf{x}})$ for $\forall \mathbf{x} \in \Omega$.

Assume that f(**x**) is differentiable at any point $\mathbf{x}_k$. Then by Taylor series expansion,

$$f(\mathbf{x}_{k+1}) = f(\mathbf{x}_k + \Delta\mathbf{x}_k) = f(\mathbf{x}_k) + \Delta\mathbf{x}_k^T \nabla f(\mathbf{x}_k) + O(\Delta\mathbf{x}_k) \quad \text{(D.1)}$$

Since $O(\Delta\mathbf{x}_k)$ is relatively small and can be neglected, to improve the current solution $\mathbf{x}_k$ at every update requires the condition $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$ to ensure that the objective function is reduced sufficiently at iteration k+1. If $\mathbf{x}_{k+1} = \tilde{\mathbf{x}}$, this leads the first-order necessary condition as follows.

$$\Delta\mathbf{x}_k^T \nabla f(\mathbf{x}_k) = f(\tilde{\mathbf{x}}) - f(\mathbf{x}_k) \leq 0 \quad \text{(D.2)}$$

And such a vector $\Delta\mathbf{x}_k$ has the descent property.

- *Definition 3*: (Search direction). A vector $\mathbf{p} \in \Re^n$ and $\mathbf{p} \neq 0$, is a search direction at $\mathbf{x} \in \Omega$ if there exists $\lambda_0 > 0$ such that $\mathbf{x} + \lambda\mathbf{p} \in \Omega$ for $\forall\lambda \in [0,\lambda_0]$, N.B. λ is so-called step length.

Let $\Delta\mathbf{x} = \lambda\mathbf{p}$. At each update, there are two key procedures. Firstly, it is to find the search direction **p** that has the descent property. Secondly, from a given search direction, the step length must to be minimised to ensure that the reduction of the objective function is sufficient.

D.2 Method of Steepest Descent

Steepest descent method is one of Gradient search methods. From the first-order necessary condition, the search direction must have the descent property. Let $\Delta\mathbf{x}_k = \lambda\mathbf{p}_k$. Thus,

$$\lambda \mathbf{p}_k^T \nabla f(\mathbf{x}_k) < 0 \tag{D.3}$$

It is the fact that the gradient information at the point $\mathbf{x}_k$, $\mathbf{x}_k \neq \tilde{\mathbf{x}}$, is a direction that gives the maximum rate of increase of the objective function. In the similar manner, the direction $-\nabla f(\mathbf{x}_k)$ gives the maximum rate of decrease of the objective function. Hence, in this method, the direction of negative gradient is applied as the search direction. This leads the following expression.

$$-\lambda \nabla f(\mathbf{x}_k)^T \nabla f(\mathbf{x}_k) = -\lambda \|\nabla f(\mathbf{x}_k)\|^2 < 0 \tag{D.4}$$

It is clear that the given search direction satisfies the descent property. It is necessary to find an appropriate λ that yields the minimum value of the objective function along the search direction. This sub-problem is called the line search sub-problem and briefly summarised as follows.

$$\underset{\lambda>0}{\text{Minimise}} \qquad F(\lambda) \equiv f(\mathbf{x}_k - \lambda \nabla f(\mathbf{x}_k)) \tag{D.5}$$

The iterative process will be performed until one of stopping criteria is met. In general, there are three stopping criteria that are widely used,

1. Change in the objective function: $|f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k)| < \delta_1$
2. Maximum change in variables: $\|\mathbf{x}_{k+1} - \mathbf{x}_k\| < \delta_2$

where

δ_1 and δ_2 are relatively small real numbers

APPENDIX E

SYSTEM DATA FOR SIMULATION

E.1 Test Systems for the Single-Phase AC Railway Power Flow Calculation

- **Standard IEEE 24-bus test system**

Table E.1: Bus information

Bus	Voltage		Generation		Demand		Bus type
	Magnitude (p.u.)	Phase (Degree)	P_{Gen} (MW)	Q_{Gen} (MVar)	P_{Dem} (MW)	$Q_{Dem\ (MVar)}$	
1	1.00	0	0	0	0	0	Load
2	1.00	0	0	0	0	0	Load
3	1.00	0	0	0	0	0	Load
4	1.00	0	0	0	0	0	Load
5	1.00	0	0	0	0	0	Load
6	1.00	0	0	0	0	0	Load
7	1.00	0	0	0	10	20	Load
8	1.00	0	0	0	0	0	Load
9	1.00	0	0	0	30	10	Load
10	1.00	0	0	0	0	0	Load
11	1.00	0	0	0	0	0	Load
12	1.00	0	0	0	0	0	Load
13	1.00	0	0	0	40	20	Load
14	1.00	0	0	0	0	0	Load
15	1.00	0	0	0	30	20	Load
16	1.00	0	0	0	0	0	Load
17	1.00	0	0	0	0	0	Load
18	1.00	0	0	0	40	10	Load
19	1.00	0	0	0	20	10	Load
20	1.00	0	0	0	30	10	Load
21	1.00	0	0	0	0	0	Load
22	1.00	0	0	0	0	0	Load
23	1.00	0	0	0	0	0	Load
24	1.00	0	0	0	20	10	Slack

Table E.2: Transmission line information

No.	From bus	To bus	r (p.u.)	x (p.u.)	y/2 (p.u.)	Tap*
1	2	24	0	0.0139	0	1
2	3	24	0	0.2112	0	1
3	5	24	0	0.0845	0	1
4	2	4	0	0.1264	0	1
5	2	6	0	0.192	0	1
6	3	20	0	0.119	0	1

7	1	3	0	0.0839	0	1
8	4	20	0	0.1037	0	1
9	5	10	0	0.0883	0	1
10	6	10	0	0.0605	0	1
11	7	22	0	0.0614	0	1
12	20	22	0	0.1651	0	1
13	10	22	0	0.1651	0	1
14	11	20	0	0.0839	0	1
15	11	20	0	0.0839	0	1
16	10	11	0	0.0839	0	1
17	10	11	0	0.0839	0	1
18	11	13	0	0.0476	0	1
19	11	14	0	0.0418	0	1
20	12	13	0	0.0476	0	1
21	12	23	0	0.0966	0	1
22	13	23	0	0.0865	0	1
23	14	16	0	0.0389	0	1
24	15	16	0	0.0073	0	1
25	15	21	0	0.049	0	1
26	15	21	0	0.049	0	1
27	1	15	0	0.0519	0	1
28	16	17	0	0.0259	0	1
29	16	19	0	0.0231	0	1
30	17	18	0	0.0144	0	1
31	8	17	0	0.1053	0	1
32	18	21	0	0.0259	0	1
33	18	21	0	0.0259	0	1
34	9	19	0	0.0396	0	1
35	9	19	0	0.0396	0	1
36	9	23	0	0.0216	0	1
37	9	23	0	0.0216	0	1
38	8	21	0	0.0678	0	1

* Tap is used for a transformer section and tap = 1 for a transmission line

- **Standard IEEE 57-bus test system**

Table E.3: Bus information

Bus	Voltage		Generation		Demand		Bus type
	Magnitude (p.u.)	Phase (Degree)	P_{Gen} (MW)	Q_{Gen} (MVar)	P_{Dem} (MW)	Q_{Dem} (MVar)	
1	1.00	0	0	0	10	5	Load
2	1.00	0	0	0	0	0	Load
3	1.00	0	0	0	0	0	Load
4	1.00	0	0	0	0	0	Load
5	1.00	0	0	0	16	4	Load
6	1.00	0	0	0	0	0	Load
7	1.00	0	0	0	0	0	Load
8	1.00	0	0	0	0	0	Load
9	1.00	0	0	0	8	3	Load
10	1.00	0	0	0	0	0	Load

11	1.00	0	0	0	0	0	Load
12	1.00	0	0	0	0	0	Load
13	1.00	0	0	0	0	0	Load
14	1.00	0	0	0	0	0	Load
15	1.00	0	0	0	0	0	Load
16	1.00	0	0	0	16	6	Load
17	1.00	0	0	0	0	0	Load
18	1.00	0	0	0	0	0	Load
19	1.00	0	0	0	0	0	Load
20	1.00	0	0	0	0	0	Load
21	1.00	0	0	0	11	4	Load
22	1.00	0	0	0	15	4	Load
23	1.00	0	0	0	0	0	Load
24	1.00	0	0	0	0	0	Load
25	1.00	0	0	0	0	0	Load
26	1.00	0	0	0	0	0	Load
27	1.00	0	0	0	0	0	Load
28	1.00	0	0	0	10	3	Load
29	1.00	0	0	0	0	0	Load
30	1.00	0	0	0	0	0	Load
31	1.00	0	0	0	6	3	Load
32	1.00	0	0	0	0	0	Load
33	1.00	0	0	0	0	0	Load
34	1.00	0	0	0	0	0	Load
35	1.00	0	0	0	0	0	Load
36	1.00	0	0	0	15	4	Load
37	1.00	0	0	0	0	0	Load
38	1.00	0	0	0	0	0	Load
39	1.00	0	0	0	0	0	Load
40	1.00	0	0	0	0	0	Load
41	1.00	0	0	0	12	5	Load
42	1.00	0	0	0	0	0	Load
43	1.00	0	0	0	0	0	Load
44	1.00	0	0	0	9	2	Load
45	1.00	0	0	0	10	2	Load
46	1.00	0	0	0	0	0	Load
47	1.00	0	0	0	0	0	Load
48	1.00	0	0	0	0	0	Load
49	1.00	0	0	0	11	2	Load
50	1.00	0	0	0	0	0	Load
51	1.00	0	0	0	0	0	Load
52	1.00	0	0	0	14	4	Load
53	1.00	0	0	0	0	0	Load
54	1.00	0	0	0	0	0	Load
55	1.00	0	0	0	0	0	Load
56	1.00	0	0	0	0	0	Load
57	1.00	0	0	0	0	0	Slack

Table E.4: Transmission line information

No.	From bus	To bus	r (p.u.)	x (p.u.)	y/2 (p.u.)	Tap*
1	56	2	0.01	0.03	0	1
2	2	3	0.01	0.03	0	1
3	3	4	0.02	0.06	0	1
4	4	5	0.01	0.03	0	1
5	4	6	0.03	0.09	0	1
6	6	7	0.02	0.06	0	1
7	6	8	0.015	0.045	0	1
8	8	9	0.025	0.075	0	1
9	9	10	0.025	0.075	0	1
10	9	55	0.02	0.06	0	1
11	9	12	0.01	0.03	0	1
12	9	13	0.01	0.03	0	1
13	13	14	0.02	0.06	0	1
14	13	15	0.01	0.03	0	1
15	56	15	0.03	0.09	0	1
16	56	16	0.02	0.06	0	1
17	56	17	0.015	0.045	0	1
18	3	15	0.025	0.075	0	1
19	4	18	0.025	0.075	0	1
20	4	18	0.02	0.06	0	1
21	5	6	0.01	0.03	0	1
22	7	8	0.01	0.03	0	1
23	10	12	0.02	0.06	0	1
24	38	48	0.025	0.075	0	1
25	12	13	0.03	0.09	0	1
26	12	16	0.02	0.06	0	1
27	12	17	0.015	0.045	0	1
28	14	15	0.025	0.075	0	1
29	18	19	0.025	0.075	0	1
30	19	20	0.02	0.06	0	1
31	20	21	0.01	0.03	0	1
32	21	22	0.01	0.03	0	1
33	22	23	0.02	0.06	0	1
34	23	24	0.01	0.03	0	1
35	24	25	0.03	0.09	0	1
36	24	25	0.02	0.06	0	1
37	24	26	0.015	0.045	0	1
38	26	27	0.025	0.075	0	1
39	27	28	0.025	0.075	0	1
40	28	29	0.02	0.06	0	1
41	7	29	0.01	0.03	0	1
42	25	30	0.01	0.03	0	1
43	30	31	0.02	0.06	0	1
44	31	32	0.01	0.03	0	1
45	32	33	0.03	0.09	0	1
46	32	34	0.02	0.06	0	1
47	34	35	0.015	0.045	0	1
48	35	36	0.025	0.075	0	1
49	36	37	0.025	0.075	0	1

50	37	38	0.02	0.06	0	1
51	37	39	0.01	0.03	0	1
52	36	40	0.01	0.03	0	1
53	22	38	0.02	0.06	0	1
54	11	41	0.01	0.03	0	1
55	41	42	0.03	0.09	0	1
56	41	43	0.02	0.06	0	1
57	38	44	0.015	0.045	0	1
58	15	45	0.025	0.075	0	1
59	14	46	0.025	0.075	0	1
60	46	47	0.02	0.06	0	1
61	47	48	0.01	0.03	0	1
62	48	49	0.01	0.03	0	1
63	49	50	0.02	0.06	0	1
64	50	51	0.01	0.03	0	1
65	10	51	0.03	0.09	0	1
66	13	49	0.02	0.06	0	1
67	29	52	0.015	0.045	0	1
68	52	53	0.025	0.075	0	1
69	53	54	0.025	0.075	0	1
70	54	55	0.02	0.06	0	1
71	11	43	0.01	0.03	0	1
72	44	45	0.01	0.03	0	1
73	40	57	0.02	0.06	0	1
74	57	41	0.01	0.03	0	1
75	57	42	0.03	0.09	0	1
76	39	1	0.02	0.06	0	1
77	57	1	0.015	0.045	0	1

* Tap is used for a transformer section and tap = 1 for a transmission line

- **12-train single-phase AC railway test system**

Table E.5: Bus information

Bus	Voltage		Generation		Demand		Bus type
	Magnitude (p.u.)	Phase (Degree)	P_{Gen} (MW)	Q_{Gen} (MVar)	P_{Dem} (MW)	Q_{Dem} (MVar)	
1	1	0	0	0	1.6	1.2	Train
2	1	0	0	0	1.6	1.2	Train
3	1	0	0	0	1.6	1.2	Train
4	1	0	0	0	1.6	1.2	Train
5	1	0	0	0	1.6	1.2	Train
6	1	0	0	0	1.6	1.2	Train
7	1	0	0	0	1.6	1.2	Train
8	1	0	0	0	1.6	1.2	Train
9	1	0	0	0	3.2	2.4	MPTSC
10	1	0	0	0	3.2	2.4	SS
11	1	0	0	0	0	0	Slack

Table E.6: Transmission line

No.	From bus	To bus	r (p.u.)	x (p.u.)	y/2 (p.u.)	Tap*
1	1	2	1.60E-03	4.80E-03	0	1
2	2	3	1.60E-03	4.80E-03	0	1
3	3	4	1.60E-03	4.80E-03	0	1
4	4	9	1.60E-03	4.80E-03	0	1
5	1	10	1.60E-03	4.80E-03	0	1
6	5	6	1.60E-03	4.80E-03	0	1
7	6	7	1.60E-03	4.80E-03	0	1
8	7	8	1.60E-03	4.80E-03	0	1
9	8	9	1.60E-03	4.80E-03	0	1
10	5	10	1.60E-03	4.80E-03	0	1
11	10	11	8.00E-04	4.80E-03	0	1

E.2 Test Systems for the Bi-Phase AC Railway Power Flow Calculation

Both double- and quad-track AC railway test system has the following properties.

Substation:

25 MVA, 50 kV (AT feeding system)

$Z_A = 3 + j27.22\ \Omega$ $Z_B = 1 + j13.11\ \Omega$

30-km feeding length

Overhead catenary feeding system:

$$\mathbf{Z}_{OH} = \begin{bmatrix} 0.1192 + j0.7522 & 0.0574 + j0.3877 & 0.0568 + j0.3953 \\ 0.0574 + j0.3877 & 0.1648 + j0.6709 & 0.0571 + j0.3410 \\ 0.0568 + j0.3953 & 0.0571 + j0.3410 & 0.2036 + j0.8847 \end{bmatrix} \quad \Omega/km$$

Rail-to-earth ballast resistance = 10 Ω·km

Autotransformer:

Leakage impedance = 0.1564 + j0.0997 Ω

Magnetising impedance = 101.4 + j279.1 kΩ

Turn ratio = 1:1

E.3 Double-track AT railway power feeding system (Fig. 4.17)

Substation:

25 MVA, 50 kV (AT feeding system)

$Z_A = 3 + j27.22\ \Omega$ $Z_B = 1 + j13.11\ \Omega$

30-km feeding length

Overhead catenary feeding system:

$$\mathbf{Z}_{OH} = \begin{bmatrix} 0.1192 + j0.7522 & 0.0574 + j0.3877 & 0.0568 + j0.3953 \\ 0.0574 + j0.3877 & 0.1648 + j0.6709 & 0.0571 + j0.3410 \\ 0.0568 + j0.3953 & 0.0571 + j0.3410 & 0.2036 + j0.8847 \end{bmatrix} \quad \Omega/\text{km}$$

Rail-to-earth ballast resistance = 10 Ω·km

Autotransformer:

Leakage impedance = 0.1564 + j0.0997 Ω

Magnetising impedance = 101.4 + j279.1 kΩ

Turn ratio = 1:1

Position: 10 km, 20 km, 30 km (Up-track and Down-track)

Train model (BR class 319):

Effective mass	355 tonnes
Maximum tractive effort	98 kN
Maximum speed	100 kph
Drag force coefficients	a = 3402, b = 238.1, c = 8.9
First-corner speed	59.94 kph
Speed ratio	1.2
Speed control strategy	Proportional speed control
Line-side to wheel efficiency	80%

Train schedule:

10 minutes headway

Passenger station:

0 km, 4 km, 11 km, 16 km, 20 km, 25 km, 30 km (7 passenger stations)

Track gradient profile:

Fig. E.1a

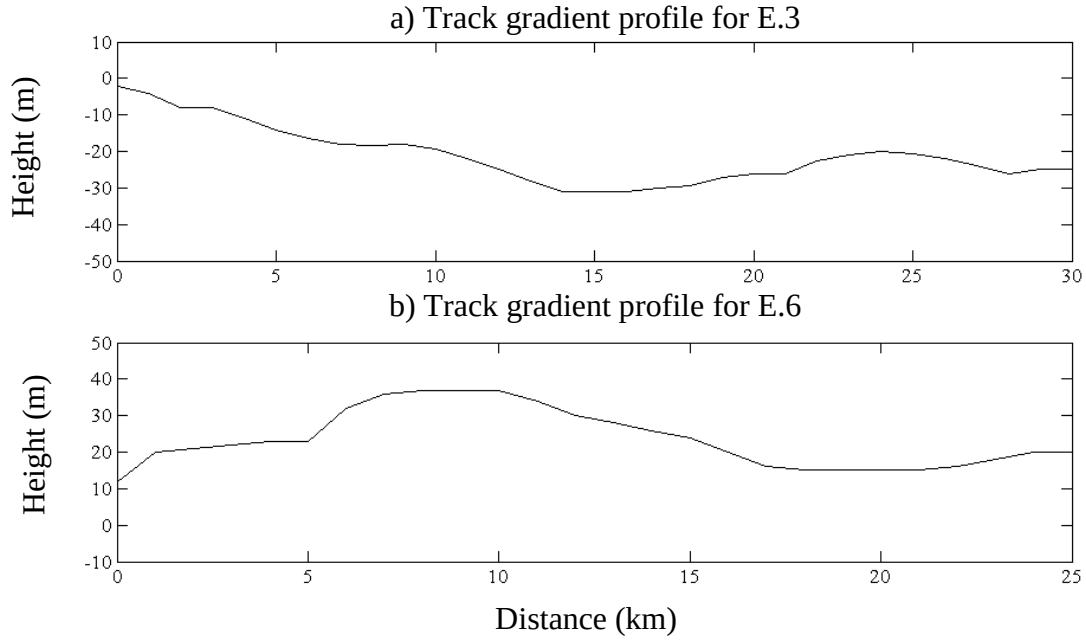


Fig. E.1: Track gradient profile for E.3 and E.6

E.4 Double-track AC Railway Power Feeding System (Fig. 5.21)

Substation:

25 kV voltage source in series with 0.5 + j3.0 Ω

25-km feeding length

Overhead catenary feeding system:

Equivalent feeding impedance = 0.1 + j0.3 Ω/km

Train: (Base case)

	TR1	TR2	TR3	TR4
Position	5 km, Up-track	17 km, Up-track	8 km, Down-track	20 km, Down-track
Real power (MW)	0.8	1.0	0.8	0.8
Power factor	0.8 lagging	0.8 lagging	0.8 lagging	0.8 lagging

E.5 Double-track AC Railway Power Feeding System (Fig. 5.21, Second test)

Power system information is as described in Section E.4 but there are 8 trains.

Train: (Base case)

	TR1	TR2	TR3	TR4
Position	5 km, Up-track	10 km, Up-track	15 km, Up-track	20 km, Up-track
Real power (MW)	1.0	1.0	1.0	1.0
Power factor	0.8 lagging	0.8 lagging	0.8 lagging	0.8 lagging
	TR5	TR6	TR7	TR8
Position	5 km, Down-track	10 km, Down-track	15 km, Down-track	20 km, Down-track
Real power (MW)	1.0	1.0	1.0	1.0
Power factor	0.8 lagging	0.8 lagging	0.8 lagging	0.8 lagging

E.6 Double-track AC Railway Power Feeding System (Fig. 6.12)

Substation:

25 kV voltage source in series with 0.5 + j3.0 Ω

25-km feeding length

Overhead catenary feeding system:

Equivalent feeding impedance = 0.1 + j0.3 Ω/km

Train model:

Effective mass	200 tonnes
Maximum tractive effort	150 kN
Maximum speed	50 kph
Drag force coefficients	a = 18500, b = 138.9, c = 57.0
First-corner speed	25 kph
Speed control strategy	Proportional speed control

Line-side to wheel efficiency	80%

Train schedule:

5 minutes headway

Passenger station:

0 km, 5 km, 11 km, 16 km, 21 km, 25 km (6 passenger stations)

Track gradient profile:

Fig. E.1b

E.7 Double-track AC Railway Power Feeding System (Section 7.4)

Substation:

25 kV voltage source in series with 0.5 + j3.0 Ω

25-km feeding length

Overhead catenary feeding system:

Equivalent feeding impedance = 0.1 + j0.3 Ω/km

Train model:

Effective mass	200 tonnes
Maximum tractive effort	150 kN
Maximum speed	100 kph
Drag force coefficients	a = 18500, b = 138.9, c = 57.0
First-corner speed	50 kph
Speed control strategy	Proportional speed control
Line-side to wheel efficiency	80%

Train schedule:

7.5 minutes headway

Passenger station:

0 km, 12 km, 25 km (3 passenger stations)

Track gradient profile:

Fig. E.1b

APPENDIX F

NUMERICAL RESULTS

F.1 IEEE 24-bus test system

Table F.1: Power flow solutions obtained by using SLPFM, NRPFM and FDCPFM

Bus	SLPFM		NRPFM		FDCPFM	
	Magnitude (p.u.)	Phase (Degree)	Magnitude (p.u.)	Phase (Degree)	Magnitude (p.u.)	Phase (Degree)
1	0.8981	-9.92	0.8981	-9.92	0.8981	-9.92
2	0.9909	-0.69	0.9909	-0.69	0.9909	-0.69
3	0.9204	-7.23	0.9204	-7.23	0.9204	-7.23
4	0.9463	-4.52	0.9463	-4.52	0.9463	-4.52
5	0.9619	-3.00	0.9619	-3.00	0.9619	-3.00
6	0.9401	-4.95	0.9401	-4.95	0.9401	-4.95
7	0.8864	-8.16	0.8864	-8.16	0.8864	-8.16
8	0.8841	-12.01	0.8840	-12.01	0.8840	-12.01
9	0.8825	-12.14	0.8825	-12.14	0.8825	-12.14
10	0.9252	-6.39	0.9252	-6.39	0.9252	-6.39
11	0.9057	-8.79	0.9057	-8.79	0.9057	-8.79
12	0.8874	-11.30	0.8873	-11.30	0.8873	-11.30
13	0.8892	-10.98	0.8892	-10.98	0.8892	-10.98
14	0.8951	-10.25	0.8950	-10.25	0.8950	-10.25
15	0.8854	-11.65	0.8854	-11.65	0.8854	-11.65
16	0.8857	-11.63	0.8857	-11.63	0.8857	-11.63
17	0.8841	-12.00	0.8841	-12.00	0.8841	-12.00
18	0.8833	-12.21	0.8833	-12.21	0.8833	-12.21
19	0.8828	-12.06	0.8828	-12.06	0.8828	-12.06
20	0.9131	-7.93	0.9131	-7.93	0.9131	-7.93
21	0.8840	-12.02	0.8840	-12.02	0.8840	-12.02
22	0.9003	-7.72	0.9003	-7.72	0.9003	-7.72
23	0.8836	-11.95	0.8836	-11.95	0.8836	-11.95
24	1.0000	0.00	1.0000	0.00	1.0000	0.00

F.2 IEEE 57-bus test system

Table F.2: Power flow solutions obtained by using SLPFM, NRPFM and FDCPFM

Bus	Magnitude (p.u.)			Angle (degree)		
	NRPFM	FDCPFM	SLPFM	NRPFM	FDCPFM	SLPFM
1	0.9700	0.9697	0.9697	-1.7820	-1.7822	-1.7822
2	0.8410	0.8406	0.8406	-11.8160	-11.8157	-11.8163
3	0.8410	0.8410	0.8411	-11.7810	-11.7811	-11.7817
4	0.8390	0.8390	0.8390	-11.9930	-11.9933	-11.9939
5	0.8360	0.8364	0.8364	-12.2640	-12.2637	-12.2644
6	0.8370	0.8371	0.8371	-12.1750	-12.1747	-12.1753
7	0.8370	0.8368	0.8368	-12.1900	-12.1905	-12.1911
8	0.8380	0.8376	0.8376	-12.1200	-12.1204	-12.1210
9	0.8400	0.8401	0.8401	-11.8560	-11.8555	-11.8562
10	0.8410	0.8413	0.8414	-11.7420	-11.7420	-11.7426
11	0.9980	0.9977	0.9977	-0.1520	-0.1526	-0.1526
12	0.8400	0.8400	0.8400	-11.8580	-11.8577	-11.8583
13	0.8440	0.8436	0.8436	-11.5410	-11.5409	-11.5415
14	0.8460	0.8458	0.8458	-11.3370	-11.3375	-11.3380
15	0.8450	0.8449	0.8449	-11.4320	-11.4319	-11.4324
16	0.8360	0.8359	0.8359	-12.1970	-12.1966	-12.1973
17	0.8400	0.8400	0.8400	-11.8540	-11.8540	-11.8546
18	0.8410	0.8415	0.8415	-11.7460	-11.7457	-11.7463
19	0.8470	0.8471	0.8471	-11.1940	-11.1939	-11.1944
20	0.8520	0.8517	0.8517	-10.7580	-10.7577	-10.7582
21	0.8540	0.8540	0.8540	-10.5410	-10.5414	-10.5418
22	0.8590	0.8590	0.8591	-10.1010	-10.1005	-10.1009
23	0.8620	0.8620	0.8620	-9.7970	-9.7972	-9.7976
24	0.8630	0.8635	0.8635	-9.6460	-9.6464	-9.6467
25	0.8700	0.8700	0.8700	-9.0530	-9.0533	-9.0536
26	0.8580	0.8577	0.8577	-10.1710	-10.1709	-10.1713
27	0.8480	0.8481	0.8481	-11.0610	-11.0610	-11.0615
28	0.8390	0.8388	0.8388	-11.9710	-11.9712	-11.9718
29	0.8360	0.8359	0.8359	-12.2690	-12.2686	-12.2693
30	0.8750	0.8755	0.8755	-8.5660	-8.5659	-8.5661
31	0.8870	0.8867	0.8867	-7.6090	-7.6094	-7.6095
32	0.8940	0.8941	0.8941	-7.0330	-7.0326	-7.0327
33	0.8940	0.8941	0.8941	-7.0330	-7.0326	-7.0327
34	0.9090	0.9091	0.9091	-5.9080	-5.9076	-5.9076
35	0.9210	0.9206	0.9206	-5.0880	-5.0881	-5.0882
36	0.9400	0.9402	0.9402	-3.7670	-3.7675	-3.7675
37	0.9200	0.9195	0.9195	-5.1780	-5.1781	-5.1781
38	0.8730	0.8727	0.8727	-8.9100	-8.9103	-8.9106
39	0.9360	0.9359	0.9359	-4.0060	-4.0058	-4.0058
40	0.9600	0.9597	0.9597	-2.4600	-2.4597	-2.4597
41	0.9980	0.9977	0.9977	-0.1520	-0.1526	-0.1526
42	0.9990	0.9988	0.9988	-0.0750	-0.0762	-0.0762
43	0.9980	0.9977	0.9977	-0.1520	-0.1526	-0.1526
44	0.8610	0.8609	0.8609	-10.0140	-10.0138	-10.0141
45	0.8550	0.8549	0.8549	-10.5700	-10.5700	-10.5704
46	0.8500	0.8496	0.8496	-10.9910	-10.9911	-10.9916

47	0.8530	0.8527	0.8527	-10.7160	-10.7162	-10.7167
48	0.8540	0.8542	0.8542	-10.5800	-10.5795	-10.5799
49	0.8480	0.8485	0.8485	-11.1290	-11.1288	-11.1293
50	0.8460	0.8461	0.8461	-11.3320	-11.3320	-11.3326
51	0.8450	0.8449	0.8449	-11.4340	-11.4341	-11.4346
52	0.8320	0.8325	0.8325	-12.6110	-12.6113	-12.6120
53	0.8350	0.8346	0.8346	-12.4000	-12.4000	-12.4007
54	0.8370	0.8367	0.8367	-12.1900	-12.1897	-12.1904
55	0.8380	0.8384	0.8384	-12.0220	-12.0223	-12.0229
56	0.8400	0.8401	0.8401	-11.8500	-11.8502	-11.8508
57	1.0000	1.0000	1.0000	0.0000	0.0000	0.0000

APPENDIX G

SIMULATION PROGRAMS

➢ Program Structure and Description

This appendix contains six simulation programs as follows.

G.1 MATLAB codes for AC railway power flow calculation (Chapter 3)

i) SLPFM for single-phase AC railway power flow analysis

File list

1. *ndpf.m* *(Main)*
2. *mk_ybus.m*
3. *delrow.m*
4. *delcol.m*
5. *lineflow.m*

ii) SLPFM for bi-phase AC railway power flow analysis

File list

1. *atpflow.m* *(Main)*
2. *ybus_form.m*
3. *jbus_form.m*

iii) NRPFM for bi-phase AC railway power flow analysis

File list

1. *natpflow.m* *(Main)*
2. *ybus_form.m*
3. *calc_jbus.m*
4. *jcalc.m*

File description:

1. *ndpf.m*: This file is the main operation to simulate single-phase AC railway power flow solution by using the SLPFM as newly developed in this thesis.

2. *mk_ybus.m*: This file is the sub-routine to form the system bus admittance matrix or $\mathbf{Y}_{bus}$ for a single-phase AC railway power feeding system.

3. *lineflow.m*: This file is the sub-routine to calculate powers or currents flowing through every feed sub-section, and the total power losses.

4. *delcol.m*: This file is the mathematical sub-routine to eliminate any column of a given matrix.

5. *delrow.m*: This file is the mathematical sub-routine to eliminate any row of a given matrix.

6. *atpflow.m*: This is the function to perform the bi-phase AC railway power flow calculation with the SLPFM.

7. *natpflow.m*: This is the function to perform the bi-phase AC railway power flow calculation with the NRPFM.

8. *ybus_form.m*: This file is the sub-routine to form the system bus admittance matrix for a bi-phase AC railway power feeding system. It is commonly used by "atpflow.m" and "natpflow.m".

9. *jbus_form.m*: This file is the sub-routine to calculate the current matrix for the SLPFM in the bi-phase AC railway power flow calculation.

10. *calc_jbus.m*: This file is the sub-routine to calculate the current mismatch for the NRPFM in the bi-phase AC railway power flow calculation.

11. *jcalc.m*: This file is the sub-routine to calculate the Jacobian matrix.

G.2 C++ codes for AC railway system simulation (Chapter 4)

i) Full matrix computation

File list

1. *mts_main.cpp* *(Main)*
2. *ATpowernet.h*
3. *jcomplex.h*
4. *net_capture.h*
5. *systeminfo.h*
6. *tmm.h*

ii) Band matrix computation

File list

1. *mtsband_main.cpp* *(Main)*
2. *ATpowernet.h*
3. *jcomplex.h*
4. *net_capture.h*
5. *systeminfo.h*
6. *tmm.h*

File description:

1. *mts_main.cpp*: This file is the main function of the AC railway system simulator, which uses the full matrix computation for the power flow solver.

2. *mtsband_main.cpp*: This file is the main function of the AC railway system simulator, which uses the band matrix computation for the power flow solver.

3. *ATpflow.h*: This file contains a bi-phase AC railway power network class and its member functions to perform the power flow calculation.

4. *jcomplex.h*: This file contains a complex number class and its member functions.

5. *net_capture.h*: This file contains network, bus data, AT information, substation performance classes and their member functions.

6. *tmm.h*: This file contains train movement, train performance classes and their member functions

7. *systeminfo.h*: This file provides program initialisation, global variable declaration, input data, counters, some commonly used variables by many classes, etc.

G.3 MATLAB codes for a single-phase AC/DC PWM power converter (Chapter 5)

File list

1. *pwmsim.m* *(Main)*
2. *swfn1.m*
3. *swfn.m*
4. *fangle.m*
5. *fdistr.m*
6. *pcalc1.m*
7. *dsim.m*
8. *fpid*
9. *fzpwm.m*
10. *frpwm.m*
11. *fipwm.m*

File description:

1. *pwmsim*.m: This file is the main function of the single-phase AC/DC PWM power converter simulation.

2. *swfn.m, swfn.m*: These files are the sub-routine to generate PWM signals to drive the power switches.

3. *fangle.m*: This file is the sub-routine to classify modes of operation at a particular interval.

4. *fdistri.m*: Due to the use of the ode45.m provided by MATLAB for solving a set of differential equations, it is necessary to adjust its unstructured output data and then rewrite it into a new structured data file ready for used.

5. *pcalc1.m*: Thisfile is the sub-routine to calculate real power, reactive power, average DC link voltage, etc.

6. *dsim.m*: This file transforms relevant continuous time-series variables into discrete-time variables.

7. *fpid.m*: This file performs the PID controller.

8. *fzpwm.m, frpwm.m, fipwm.m*: These three files describe the mathematical model for zero, rectifying and inverting modes of operation, respectively.

G.4 C++ Codes for optimal AC railway power flow calculation (Chapter 5)

File List
1. orpf.cpp (Main)

File description:
1. *orpf.cpp*: This file is the main function of the optimal AC railway power flow calculation. It contains ybus formulation, network capture, bus information, power flow solver (SLPFM) and optimisation (steepest descent method) sub-routines.

G.5 C++ Codes for Area Control Simulation (Chapter 6)

File List
1. rsimopf.cpp (Optimal area control)
2. rsimupf.cpp (UPF control)
3. rsimsvc.cpp (SVC control)

File description:
1. *rsimopf.cpp, rsimupf.cpp, rsimsvc.cpp*: These three files are the main function to perform the area control simulation, the unity-power-factor PWM locomotive control and the SVC control, respectively.

N.B. rsimupf.cpp is also used for simulating the base case by setting the power factor of all locomotives as 0.8 lagging.

G.6 C++ Codes for Cost Estimation and Design (Chapter 7)

File list
1. svcdes.cpp (Main)

File description:
1. *svcdes.cpp*: This file is the main function for SVC installation design. It contains similar main functions as those of "rsimsvc.cpp" and also a GA sub-routine for optimisation.

➢ Programming Codes

Source codes for the above mentioned programs are shown as follows.

G.1 MATLAB codes for power flow calculation

i) SLPFM for single-phase AC railway power flow analysis

File list

1. *ndpf.m (Main function)*
2. *mk_ybus.m*
3. *delrow.m*
4. *delcol.m*
5. *lineflow.m*

```
function [V0,sflow,pref,losses,time,k]=ndpf(V0,tolerance,option)
% Sequential Linear Power Flow Method
% V: Bus voltages including the slack bus
% sflow: Line flow
% pref: Generation of the reference bus
% losses: System losses
% time: Execution time (sec.)
% z: No. of Iteration
% V0: Assumed initial bus voltages
% tolerance: maximum voltage error allowance
% option ( 1: Display the results, else: display nothing)

global busdata ldata MVAbase Qlimit n_PV Vsch ybus
sysdata;                    % Input data file
mk_ybus;
N=length(V0);
V0(N)=Vsch(N);
magV=abs(V0);
magY=abs(ybus);
angV=angle(V0);
angY=angle(ybus);
Psch=(busdata(:,4)-busdata(:,6))/MVAbase;
Qsch=(busdata(:,5)-busdata(:,7))/MVAbase;
Psch=delrow(Psch,size(busdata,1));
Qsch=delrow(Qsch,size(busdata,1));
Qlimit=Qlimit/MVAbase;
N=length(ybus);
ybus1=delrow(ybus,N);
ybus1=delcol(ybus1,N);
zbus=inv(ybus1);
tic;
for k=1:1000
  for t=1:N-1
    if busdata(t,8)==1
      I(t,1)=conj((Psch(t)+Qsch(t)*i)/V0(t))-ybus(t,N)*V0(N);
    elseif busdata(t,8)==2
      q=0;
      for u=1:length(ybus)
        q=q-magV(t)*magV(u)*magY(t,u)*sin(angY(t,u)+angV(u)-
          angV(t));
      end
      if q>Qlimit(t,2)
        q=Qlimit(t,2);
      elseif q(t,1)<Qlimit(t,1)
        q=Qlimit(t,1);
      end
      I(t,1)=conj((Psch(t)+q*i)/V0(t))-ybus(t,N)*V0(N);
    elseif busdata(t,8)==3
      I(t,1)=(busdata(t,4)-busdata(t,6))+(busdata(t,5)-busdata(t,7))*i-
        ybus(t,N)*V0(N);
    elseif busdata(t,8)==4
      I(t,1)=V0(t)/(busdata(t,6)+busdata(t,7)*i)-ybus(t,N)*V0(N);
    elseif busdata(t,8)==5
      I(t,1)=busdata(t,4)+busdata(t,5)*i-V0(t)/(busdata(t,6)+busdata(t,7)*i)-
        ybus(t,N)*V0(N);
    end
  end
  V1=zbus*I;
  V1=[V1;Vsch(N)];
  err=max(abs(V1-V0)./abs(V0));
  V0=V1;
  if err<tolerance
    break;
  end
end
time=toc;
[sflow,losses]=lineflow(V0);
demand=sum(busdata(:,6));
Gen=sum(busdata(:,4));
pref=demand+losses-Gen;

if (option==1)|(option==2)
  disp(' ');
  disp('***          Power flow solution          ***');
  disp(' ');
  disp('Bus voltage')
  disp([V0 abs(V0) angle(V0)*180/pi]);
  disp(' ');
  disp('Line flow');
  disp(sflow);
  disp(' ');
  disp(['Transmission losses  ' num2str(losses) '  MW']);
  disp(' ');
  disp(['This programme converged in ' num2str(time) ' seconds, or in '
```

```
            num2str(k) ' iterations']);
  disp(' ');
end

return
```

```
function ybus=mk_ybus()
% Ybus formualtion program by inspection method

global ybus ldata
ybus=[];                        % Initial Ybus
u=max(ldata(:,2));
v=max(ldata(:,3));
n_bus=max(u,v);
n_line=size(ldata,1);           % Total number of lines
ybus(n_bus,n_bus)=0;
for k=1:n_line          % Loop for adding a line portion, line-by-line
  p=ldata(k,2);                 % From bus
  q=ldata(k,3);                 % To bus
  Ypq=1/(ldata(k,4)+ldata(k,5)*i);          % Line admittance
  line_charge=ldata(k,6)*i;                 % Half-line charging
  t=ldata(k,7);                 % Tap setting (1 for a transmission line)
  ybus(p,q)=ybus(p,q)-t*Ypq;    % Mutual admittance updated 1
  ybus(q,p)=ybus(p,q);              % Mutual admittance updated 2
% Admittance added to a self-admittance, Ybus(p,p)
  ybus(p,p)=ybus(p,p)+t*Ypq+t*(t-1)*Ypq+line_charge;
  ybus(q,q)=ybus(q,q)+t*Ypq+(1-t)*Ypq+line_charge;
end

return
```

```
function M=delrow(A,r)
s_A=size(A,1);
if (r<=0)|(r>s_A)
  error('You cannot delete that row!');
end
M=[A(1:r-1,:);A(r+1:s_A,:)];
return
```

```
function M=delrow(A,r)
s_A=size(A,1);
if (r<=0)|(r>s_A)
  error('You cannot delete that row!');
end
M=[A(1:r-1,:);A(r+1:s_A,:)];
```



```
return
```

```
function [S,loss]=lineflow(V0)
global ldata MVAbase busdata
for t=1:size(ldata,1)
  p=ldata(t,2);
  q=ldata(t,3);
  MPT=ldata(t,8)*MVAbase;
  S1(t,1)=p;
  S1(t,2)=q;
  y=1/(ldata(t,4)+ldata(t,5)*i);
  S1(t,3)=MVAbase*conj(conj(V0(p))*(V0(p)-
          V0(q))*y+conj(V0(p))*V0(p)*ldata(t,6)*i);
  end

for t=1:size(ldata,1)
  p=ldata(t,3);
  q=ldata(t,2);
  MPT=ldata(t,8)*MVAbase;
  S2(t,1)=p;
  S2(t,2)=q;
  y=1/(ldata(t,4)+ldata(t,5)*i);
  S2(t,3)=MVAbase*conj(conj(V0(p))*(V0(p)-
          V0(q))*y+conj(V0(p))*V0(p)*ldata(t,6)*i);
end
loss=abs(real(sum(S1(:,3)+S2(:,3))));
S=[S1;S2];
return
```

ii) SLPFM for bi-phase AC railway power flow analysis

File list

1. *atpflow.m (Main function)*
2. *ybus_form.m*
3. *jbus_form.m*

```
function [V,err,k]=atpflow(file_name,max_err_tol)
% Sequential Linear Power Flow Solution for AT-fed AC Railway Power
% Systems

global Ybus TV0 Jbus NB Jbus1 Vbus1
ybus_form(file_name);
Zbus=inv(Ybus);
initial_TV0;
```

```
max_err=10000;
k=0;
while (max_err>max_err_tol)
   jbus_form;
   TV=Zbus*Jbus;
   max_err=max(abs(TV-TV0));
   k=k+1;
   err(k,1)=max_err;
   TV0=TV;
   if k==1
      Jbus1=Jbus;
      Vbus1=TV0;
   end
   if k>1000
      disp('Sequential linear power flow program does not converge within 1000 iterations');
      break;
   end
end
for u=1:NB
   V(u,1)=TV(3*u-2,1);
   V(u,2)=TV(3*u-1,1);
   V(u,3)=TV(3*u,1);
end
return
```

```
function ybus_form(file_name)

global Bdata Ldata Zoh Yat Jss Yss YRE_ballast TckAtInfo Ybus NB

run(file_name);
NB=size(Bdata,1);
Ybus=zeros(3*NB,3*NB);
nline=size(Ldata,1);

for u=1:nline
   m=Ldata(u,1);
   n=Ldata(u,2);
   Ytemp=1/Ldata(u,3)*inv(Zoh);
   YRE_ballast_half=YRE_ballast*Ldata(u,3)/2;
   Ybus(3*m-2:3*m,3*n-2:3*n)=Ybus(3*m-2:3*m,3*n-2:3*n)-Ytemp;
   Ybus(3*n-2:3*n,3*m-2:3*m)=Ybus(3*m-2:3*m,3*n-2:3*n);
   Ybus(3*m-2:3*m,3*m-2:3*m)=Ybus(3*m-2:3*m,3*m-2:3*m)+
            Ytemp+YRE_ballast_half;
   Ybus(3*n-2:3*n,3*n-2:3*n)=Ybus(3*n-2:3*n,3*n-2:3*n)+
            Ytemp+YRE_ballast_half;
end
Ntrack=size(TckAtInfo,1);
for u=1:Ntrack
   for v=1:size(TckAtInfo,2)
      m=TckAtInfo(u,v);
      Ybus(3*m-2:3*m,3*m-2:3*m)=Ybus(3*m-2:3*m,3*m-2:3*m)+Yat;
   end
end
Ybus(1:3,1:3)=Ybus(1:3,1:3)+Yss;
return
```

```
function jbus_form()
global NB Bdata TV0 Jss Jbus
Jbus=zeros(3*NB,1);
for u=1:NB
   It=-(Bdata(u,2)-Bdata(u,3)*i)/conj(TV0(3*u-2,1)-TV0(3*u-1,1));
   Jt=It*[1;-1;0];
   Jbus(3*u-2:3*u,1)=Jbus(3*u-2:3*u,1)+Jt;
end
Jbus(1:3,1)=Jbus(1:3,1)+Jss;
return
```

iii) NRPFM for bi-phase AC railway power flow analysis

File list

1. *natpflow.m (Main function)*
2. *ybus_form.m (As shown in Section ii)*
3. *calc_jbus.m*
4. *jcalc.m*

```
function [V,max_error,k]=natpflow(file_name,max_err_allow)
% Newton-Raphson Based Power Flow Solution for AT-fed AC Railway
% Power Systems
global TV0 NB Ybus VBase
ybus_form(file_name);
k=0;
initial_TV0;
[G,H]=calc_jbus;
F=[G;H];
max_err=max(abs(F));
while (max_err>=max_err_allow)
   [A,B,C,D]=jcalc;
   J=[A B;C D];
   dV=inv(J)*F;
   dx=dV(1:3*NB,1);
```

```
    dy=dV(3*NB+1:6*NB);
    dTV=dx+i*dy;
    TV=TV0-dTV;
    TV0=TV;
    k=k+1;
    [G,H]=Calc_Jbus;
    F=[G;H];
    max_err=max(abs(dTV));
    max_error(k,1)=max_err;
    if k>=20
       disp('Newton-Raphson power flow program does not converge within
             100 iterations');
       break;
    end
end
for u=1:NB
    V(u,1)=TV0(3*u-2,1);
    V(u,2)=TV0(3*u-1,1);
    V(u,3)=TV0(3*u,1);
end
return
```

```
function [G,H]=calc_jbus()
global Ybus Jss TV0 NB Bdata
x=real(TV0);
y=imag(TV0);
g=real(Ybus);
b=-imag(Ybus);
s=real(Jss);
t=imag(Jss);
for k=1:NB
    wk=Bdata(k,2);
    zk=Bdata(k,3);
    dx=x(3*k-2,1)-x(3*k-1,1);
    dy=y(3*k-2,1)-y(3*k-1,1);
    DEM=dx^2+dy^2;
    for h=1:3
       ih=3*k-3+h;
       G(ih,1)=0;
       H(ih,1)=0;
       for u=1:NB
          for m=1:3
             im=3*u-3+m;
             G(ih,1)=G(ih,1)+g(ih,im)*x(im,1)+b(ih,im)*y(im,1);
             H(ih,1)=H(ih,1)+g(ih,im)*y(im,1)-b(ih,im)*x(im,1);
          end
       end
       if k==1
          G(ih,1)=G(ih,1)-s(ih,1);
          H(ih,1)=H(ih,1)-t(ih,1);
       end
       if h==1
          G(ih,1)=G(ih,1)+(wk*dx+zk*dy)/DEM;
          H(ih,1)=H(ih,1)+(wk*dy-zk*dx)/DEM;
       elseif h==2
          G(ih,1)=G(ih,1)-(wk*dx+zk*dy)/DEM;
          H(ih,1)=H(ih,1)-(wk*dy-zk*dx)/DEM;
       end
    end
end
return
```

```
function [J1,J2,J3,J4]=jcalc()
global Ybus TV0 NB Bdata
x=real(TV0);
y=imag(TV0);
g=real(Ybus);
b=-imag(Ybus);
for k=1:NB
    wk=Bdata(k,2);
    zk=Bdata(k,3);
    dx=x(3*k-2,1)-x(3*k-1,1);
    dy=y(3*k-2,1)-y(3*k-1,1);
    FG=wk*dx+zk*dy;
    FH=wk*dy-zk*dx;
    DEM=dx^2+dy^2;
    for u=1:NB
       for h=1:3
          ih=3*k-3+h;
          for m=1:3
             im=3*u-3+m;
             J1(ih,im)=g(ih,im);
             J2(ih,im)=b(ih,im);
             J3(ih,im)=-b(ih,im);
             J4(ih,im)=g(ih,im);
             if (u==k)&(h==1)&(m==1)
                J1(ih,im)=J1(ih,im)+(wk*DEM-2*dx*FG)/(DEM^2);
                J2(ih,im)=J2(ih,im)+(zk*DEM-2*dy*FG)/(DEM^2);
                J3(ih,im)=J3(ih,im)+(-zk*DEM-2*dx*FH)/(DEM^2);
                J4(ih,im)=J4(ih,im)+(wk*DEM-2*dy*FH)/(DEM^2);
             elseif (u==k)&(h==1)&(m==2)
                J1(ih,im)=J1(ih,im)+(-wk*DEM+2*dx*FG)/(DEM^2);
                J2(ih,im)=J2(ih,im)+(-zk*DEM+2*dy*FG)/(DEM^2);
                J3(ih,im)=J3(ih,im)+(zk*DEM+2*dx*FH)/(DEM^2);
                J4(ih,im)=J4(ih,im)+(-wk*DEM+2*dy*FH)/(DEM^2);
             elseif (u==k)&(h==2)&(m==1)
```

```
            J1(ih,im)=J1(ih,im)-(wk*DEM-2*dx*FG)/(DEM^2);
            J2(ih,im)=J2(ih,im)-(zk*DEM-2*dy*FG)/(DEM^2);
            J3(ih,im)=J3(ih,im)-(-zk*DEM-2*dx*FH)/(DEM^2);
            J4(ih,im)=J4(ih,im)-(wk*DEM-2*dy*FH)/(DEM^2);
         elseif (u==k)&(h==2)&(m==2)
            J1(ih,im)=J1(ih,im)-(-wk*DEM+2*dx*FG)/(DEM^2);
            J2(ih,im)=J2(ih,im)-(-zk*DEM+2*dy*FG)/(DEM^2);
            J3(ih,im)=J3(ih,im)-(zk*DEM+2*dx*FH)/(DEM^2);
            J4(ih,im)=J4(ih,im)-(-wk*DEM+2*dy*FH)/(DEM^2);
         end
      end
   end
end
return
```

G.2 C++ codes for AC Railway System Simulation

i) Full matrix computation

File list

1. *mts_main.cpp*
2. *ATpowernet.h*
3. *jcomplex.h*
4. *net_capture.h*
5. *systeminfo.h*
6. *tmm.h*

" mts_main.cpp"

```
#include "systeminfo.h"
#include "tmm.h"
#include "net_capture.h"
#include "ATpowernet.h"
void printresult(train trains[2][Nmax],VATS Vats)
{
        int i,j;
        // Track 1
        for (i=0;i<=Nmax-1;i++)
        {
                fprintf(fpos1,"%.2f ",trains[0][i].position);
                fprintf(fspd1,"%.2f ",trains[0][i].speed);
                fprintf(ftef1,"%.2f ",trains[0][i].TE);
                fprintf(fpow1,"%.2f ",trains[0][i].Pt);
                fprintf(fvar1,"%.2f ",trains[0][i].Qt);
                fprintf(ftvp_c1,"%.2f ",trains[0][i].volt[0].mag());
                fprintf(ftvp_r1,"%.2f ",trains[0][i].volt[1].mag());
                fprintf(ftvp_f1,"%.2f ",trains[0][i].volt[2].mag());
        }
        // Track 2
        for (i=0;i<=Nmax-1;i++)
        {
                if (trains[1][i].track_no!=0)
                        fprintf(fpos2,"%.2f ",trains[1][i].position);
                else
                        fprintf(fpos2,"%.2f ",_Fdist*1000);
                fprintf(fspd2,"%.2f ",trains[1][i].speed);
                fprintf(ftef2,"%.2f ",trains[1][i].TE);
                fprintf(fpow2,"%.2f ",trains[1][i].Pt);
                fprintf(fvar2,"%.2f ",trains[1][i].Qt);
                fprintf(ftvp_c2,"%.2f ",trains[1][i].volt[0].mag());
                fprintf(ftvp_r2,"%.2f ",trains[1][i].volt[1].mag());
                fprintf(ftvp_f2,"%.2f ",trains[1][i].volt[2].mag());
        }
        fprintf(fats,"%.2f %.2f ",Vats.Vss[0].mag(),Vats.Vss[1].mag());
        fprintf(fats,"%.2f %.2f ",Vats.Vmp[0].mag(),Vats.Vmp[1].mag());
        fprintf(fsub,"%.2f %.2f ",_Gen.re,_Gen.im);
        fprintf(fsub,"%.2f %.2f ",_Load.re,_Load.im);
        fprintf(fsub,"%.4f %.4f ",_Loss.re,_Loss.im);
        for (i=0;i<=_Ntrack-1;i++)
        {
                for (j=0;j<=_Nauto-2;j++)
                fprintf(fats,"%.2f %.2f %.2f ",Vats.Vat[i][j][0].mag(),
                        Vats.Vat[i][j][1].mag(),Vats.Vat[i][j][2].mag());
        }
}
void main(void)
{
        int k,i,j,tid=0,Nstop,Tsim,IDST=0;
        double chktime=0,tsim;
        clock_t Tstart,Tstop;
        printf("\nEnter the Simulated Time Period (Hour) : ");
        scanf("%d",&Nstop);
        loadsyst();
        loadmodel();
        gradient_profile();
        initial();
        train trains[2][Nmax];
        ATinfo ATdata;
        TRinfo TRdata;
        ATdata.ATbus_order();
        busdata bus;
        netdata net;
        ATpflow powsys;
        ATsyst();
```

```
        VATS Vats;
        while (IDST==0)
        {
                printf("\nEnter the Starting Condition (Select 1 or 2) ");
                printf("\n   1. Nominal Start: ");
                printf("\n   2. Recursive Start: ");
                printf("\nThe choice is : ");
                scanf("%d",&STC);
                if ((STC==1)||(STC==2))
                        IDST=1;
        }
        Tstart=clock();
        for (k=0;k<=Nstop*120*60-1;k++)
        {
                simtime=simtime+dtime;
                printf("\n%.1f ",simtime);
                writesimtime();
        if ((chktime<_headway)&&(tid>0))
                chktime=chktime+dtime;
        else
        {
                chktime=0;
                for (i=0;i<=_Ntrack-1;i++)
                        trains[i][tid].addtrain(i+1,1);
                        tid++;
        }
        for (i=0;i<=_Ntrack-1;i++)
        {
                for (j=0;j<=tid-1;j++)
                {
                        trains[i][j].trainmovement();
                        trains[i][j].stationstop();
                }
        }
        bus.insertAT(Vats);
        TRdata.get_train(trains,tid,bus,ATdata);
        bus.insertTR(TRdata,trains);
        net.netbuild(TRdata,ATdata,bus);
        powsys.runpflow(bus,net,ATdata,25);
        powsys.upd_volt(TRdata,trains);
        Vats.upd_volt(ATdata);
        printresult(trains,Vats);
}
Tstop=clock();
Tsim=(Tstop-Tstart)/CLK_TCK;
tsim=Tsim;
final();
cout<<endl<<"Execution time = "<<(tsim/60)<<" Minutes";
}
```

"ATpflow.h"

```
class ATpflow
{
   public:
        friend class train;
        int NB;
        double max_err;
        rect_cpx Ybus[Bmax][Bmax];
        rect_cpx L[Bmax][Bmax];
        rect_cpx U[Bmax][Bmax];
        rect_cpx Vbus[Bmax];
        rect_cpx Vbus0[Bmax];
        rect_cpx Jbus[Bmax];
        rect_cpx Sload;
        rect_cpx Sloss;
        rect_cpx Sgen;
        void initial_ybus(void);
        void initial_jbus(void);
        void Form_Ybus(busdata,netdata,ATinfo);
        void Form_Jbus(busdata);
        void LUfactor(void);
        void LUsolve(void);
        void initial(busdata);
        void check_convergence(void);
        void powersolve(busdata);
        void runpflow(busdata,netdata,ATinfo,double);
        void upd_volt(TRinfo,train (*)[Nmax]);
        ATpflow(void);
        ~ATpflow(void);
};
void ATpflow::initial_ybus(void)
{
        int i,j;
        for (i=0;i<=3*NB-1;i++)
        {
                for (j=0;j<=3*NB-1;j++)
                        Ybus[i][j]=rect_cpx(0,0);
        }
}
void ATpflow::Form_Ybus(busdata bus,netdata net,ATinfo atr)
{
        int i,j,m,n,p,q;
        rect_cpx Ytemp,Yre_ball_half;
        initial_ybus();
        for (i=0;i<=net.Nnet-1;i++)
        {
                m=net.frombus[i];
                n=net.tobus[i];
```

```
                for (p=0;p<=2;p++)
                {
                        for (q=0;q<=2;q++) {
                                Ytemp=Yoh[p][q]/rect_cpx(0.001*net.dist[i],0);
                        Yre_ball_half=Yre_ballast[p][q]*rect_cpx(0.001*net.dist[i]/2,0);
                          Ybus[3*m-3+p][3*n-3+q]=Ybus[3*m-3+p][3*n-3+q]-Ytemp;
                                Ybus[3*n-3+p][3*m-3+q]=Ybus[3*m-3+p][3*n-3+q];
                                Ybus[3*m-3+p][3*m-3+q]=Ybus[3*m-3+p][3*m-3+q]
                                        +Ytemp+Yre_ball_half;
Ybus[3*n-3+p][3*n-3+q]=Ybus[3*n-3+p][3*n-3+q]+Ytemp+Yre_ball_half;
                                }
                }
        }
        for (i=0;i<=_Ntrack-1;i++)
        {
                for (j=0;j<=_Nauto-1;j++)
                {
                        m=atr.ATbus[i][j];
                        for (p=0;p<=2;p++)
                        {
                                for (q=0;q<=2;q++)
                                {
                                        if ((i==1)&&(j==1))
                        Ybus[3*m-3+p][3*m-3+q]=Ybus[3*m-3+p][3*m-3+q]+Yat[p][q];
                                        else
                        Ybus[3*m-3+p][3*m-3+q]=Ybus[3*m-3+p][3*m-3+q]+Yat[p][q];
                                        }
                        }
                }
        }
        for (i=0;i<=2;i++)
        {
                for (j=0;j<=2;j++)
                        Ybus[i][j]=Ybus[i][j]+Yss[i][j];
        }
}
void ATpflow::LUfactor(void)
{
        int i,j,k,dim;
        dim=3*NB;
        for (i=0;i<=dim-1;i++)
        {
                for (j=0;j<=dim-1;j++)
                {
                        U[i][j]=Ybus[i][j];
                        if (i==j)
                                L[i][j]=rect_cpx(1,0);
                        else
                                L[i][j]=rect_cpx(0,0);
                }
        }
        for (j=0;j<=dim-2;j++)
        {
                for (i=j+1;i<=dim-1;i++)
                {
                        L[i][j]=U[i][j]/U[j][j];
                        for (k=0;k<=dim-1;k++)
                                U[i][k]=U[i][k]-L[i][j]*U[j][k];
                }
        }
}
void ATpflow::LUsolve(void)
{
        int i,j,dim;
        dim=3*NB;
        rect_cpx Y[Bmax],XYsum;
        Y[0]=Jbus[0];
        for (i=1;i<=dim-1;i++)
        {
                XYsum=rect_cpx(0,0);
                for (j=0;j<=i-1;j++)
                XYsum=XYsum+L[i][j]*Y[j];
                Y[i]=Jbus[i]-XYsum;
        }
        Vbus[dim-1]=Y[dim-1]/U[dim-1][dim-1];
        for (i=3*NB-2;i>=0;i--)
        {
                XYsum=rect_cpx(0,0);
                for (j=i+1;j<=dim-1;j++)
                XYsum=XYsum+U[i][j]*Vbus[j];
                Vbus[i]=(Y[i]-XYsum)/U[i][i];
        }
}
void ATpflow::initial_jbus(void)
{
        int i;
        for (i=0;i<=NB-1;i++)
        {
                Jbus[3*i]=rect_cpx(0,0);
                Jbus[3*i+1]=rect_cpx(0,0);
                Jbus[3*i+2]=rect_cpx(0,0);
        }
}

void ATpflow::Form_Jbus(busdata bus)
{
        int i,j;
        rect_cpx It,Jt[3];
        double P,Q;
        initial_jbus();
```

```
	for (i=0;i<=NB-1;i++)
	{
		P=-bus.Pdem[i];
		Q=bus.Qdem[i];
		It=rect_cpx(P,Q)/rect_conj(Vbus[3*i]-Vbus[3*i+1]);
		Jt[0]=It;
		Jt[1]=rect_cpx(0,0)-It;
		Jt[2]=rect_cpx(0,0);
		for (j=0;j<=2;j++)
		Jbus[3*i+j]=Jbus[3*i+j]+Jt[j];
	}
	for (i=0;i<=2;i++)
		Jbus[i]=Jbus[i]+Jss[i];
}
void ATpflow::check_convergence(void)
{
	int i;
	double err;
	max_err=0;
	for (i=0;i<=3*NB-1;i++)
	{
		err=fabs(rect_mag(Vbus[i]-Vbus0[i]));
		if (err>max_err)
		max_err=err;
	}
}
void ATpflow::powersolve(busdata bus)
{
	int i;
	Form_Jbus(bus);
	LUfactor();
	LUsolve();
	check_convergence();
	for (i=0;i<=3*NB-1;i++)
	{
		Vbus0[i]=Vbus[i];
		_Vbus[i]=Vbus[i];
	}
}
void ATpflow::initial(busdata bus)
{
	int i;
	rect_cpx Vp(0.5*V0,0),Vn(100,0);

	for (i=0;i<=NB-1;i++)
	{
		switch (STC)
		{
			case 1:
				Vbus0[3*i]=Vp;
				Vbus0[3*i+1]=Vn;
				Vbus0[3*i+2]=Vp;
				Vbus[3*i]=Vp;
				Vbus[3*i+1]=Vn;
				Vbus[3*i+2]=Vp;
				break;
			case 2:
				if ((_NT-_NT0!=0)||(simtime==0.5))
				{
					Vbus0[3*i]=Vp;
					Vbus0[3*i+1]=Vn;
					Vbus0[3*i+2]=Vp;
					Vbus[3*i]=Vp;
					Vbus[3*i+1]=Vn;
					Vbus[3*i+2]=Vp;
				}
				else
				{
					Vbus0[3*i]=bus.Volt[i][0];
					Vbus0[3*i+1]=bus.Volt[i][1];
					Vbus0[3*i+2]=bus.Volt[i][2];
					Vbus[3*i]=bus.Volt[i][0];
					Vbus[3*i+1]=bus.Volt[i][1];
					Vbus[3*i+2]=bus.Volt[i][2];
				}
				break;
		}

	}
	_NT0=_NT;
}

void ATpflow::lineflow(busdata bus,netdata net)
{
	int h,i,j,k;
	double D;
	rect_cpx Yc,Yr,Yf,Sc,Sr,Sf,Yre_ball_half;
	Sload=rect_cpx(0,0);
	Sloss=rect_cpx(0,0);
	for (h=0;h<=net.Nnet-1;h++)
	{
		for (k=1;k<=2;k++)
		{
			D=net.dist[h];
			if (k==1)
			{
				i=net.frombus[h]-1;
				j=net.tobus[h]-1;
			}
			else
```

```
                                {
                                        j=net.frombus[h]-1;
                                        i=net.tobus[h]-1;
                                }
                                Yc=rect_cpx(1,0)/(rect_cpx(D,0)*Zc);
                                Yr=rect_cpx(1,0)/(rect_cpx(D,0)*Zr);
                                Yf=rect_cpx(1,0)/(rect_cpx(D,0)*Zf);
                                Yre_ball_half=Yre_ballast[1][1]/rect_cpx(2*D,0);
                                Sc=Vbus[3*i].conj()*Yc*(Vbus[3*i]-Vbus[3*j]);
                                        Sr=Vbus[3*i+1].conj()*Yr*(Vbus[3*i+1]-Vbus[3*j+1])+
                                                Vbus[3*i+1].conj()*Yre_ball_half*Vbus[3*i+1];
                                         Sf=Vbus[3*i+2].conj()*Yf*(Vbus[3*i+2]-Vbus[3*j+2]);
                                Sloss=Sloss+Sc+Sr+Sf;
                        }
                }
                for (h=0;h<=NB-1;h++)
                        Sload=Sload+rect_cpx(bus.Pdem[h],bus.Qdem[h]);
                Sgen=Sload+Sloss;
                _Gen=Sgen;
                _Load=Sload;
                _Loss=Sloss;
}
void ATpflow::runpflow(busdata bus,netdata net,ATinfo atr,double MAX_ERR)
{
                int Iter_no=0,Iter_max=1000;
                double err=1000;
                NB=bus.Nbus;
                initial(bus);
                Form_Ybus(bus,net,atr);
                while (err>MAX_ERR)
                {
                        powersolve(bus);
                        err=max_err;
                        Iter_no++;
                        if (Iter_no>Iter_max)
                        {
                        printf("\nThe system does not converge!!!");
                                break;
                        }
                }
}
void ATpflow::upd_volt(TRinfo tro,train tm[2][Nmax])
{
                int i,j,m,n;
                for (i=0;i<=_Ntrack-1;i++)
                {
                        for (j=0;j<=tro.Tnum[i]-1;j++)
                        {
                                m=tro.TRbus[i][j]-1;
                                n=tro.TRid[i][j];
                                trn[i][n].volt[0]=Vbus[3*m];
                                trn[i][n].volt[1]=Vbus[3*m+1];
                                trn[i][n].volt[2]=Vbus[3*m+2];
                        }
                }
}

ATpflow::ATpflow(void)
{
                // class constructor
}

ATpflow::~ATpflow(void)
{
                // class constructor
}
```

"jcomplex.h"

```
#if !defined(__MATH_H)
#include <math.h>
#endif

#if !defined(__IOSTREAM_H)
#include <iostream.h>
#endif
const double pi=3.14159265358979;
class rect_cpx
{
        public:
                double re;
                double im;
                double mag(void);
                double arg(void);
                rect_cpx conj(void);
                rect_cpx operator + (const rect_cpx& p);
                rect_cpx operator - (const rect_cpx& p);
                rect_cpx operator * (const rect_cpx& p);
                rect_cpx operator / (const rect_cpx& p);
                rect_cpx(void): re (0.0),im (0.0) {};
                rect_cpx(double,double);
                ~rect_cpx(void);
};
inline ostream& operator << (ostream& s,const rect_cpx& p)
{
        if (p.im>=0)
                s<<p.re<<" + j"<<p.im;
        else
                s<<p.re<<" - j"<<(-p.im);
```

```
	return s;
}
double rect_cpx::mag(void)
{
	return sqrt(re*re+im*im);
}
double rect_cpx::arg(void)
{
	double zetha;
	if (im==0)
	{
		if (re>0)
			zetha=pi/2;
		else
			zetha=-pi/2;
	}
	else
		zetha=atan(re/im);
	if ((zetha>0)&&(re<0))
		zetha=zetha+pi;
	else if ((zetha<0)&&(re<0))
		zetha=zetha+pi;
	return zetha;
}
rect_cpx rect_cpx::conj(void)
{
	return rect_cpx(re,-im);
}
rect_cpx rect_conj(rect_cpx p)
{
	return rect_cpx(p.re,-p.im);
}
double rect_mag(rect_cpx p)
{
	return sqrt(p.re*p.re+p.im*p.im);
}

inline rect_cpx rect_cpx::operator + (const rect_cpx& p)
{
	return rect_cpx(re+p.re,im+p.im);
}
inline rect_cpx rect_cpx::operator - (const rect_cpx& p)
{
	return rect_cpx(re-p.re,im-p.im);
}
inline rect_cpx rect_cpx::operator * (const rect_cpx& p)
{
	return rect_cpx(re*p.re-im*p.im,im*p.re+re*p.im);
}
inline rect_cpx rect_cpx::operator / (const rect_cpx& p)
{
	double dem,x,y;
	dem=p.re*p.re+p.im*p.im;
	x=(re*p.re+im*p.im)/dem;
	y=(im*p.re-re*p.im)/dem;
	return rect_cpx(x,y);
}
rect_cpx::rect_cpx(double x,double y)
{
	re=x;
	im=y;
}
rect_cpx::~rect_cpx(void)
{
	// Class destructor
}
```

"net_capture.h"

```
class VATS	// Voltage class at the substation, AT points and the MPTSC
{
	public:
		friend class ATpflow;
		friend class busdata;
		friend class ATinfo;
		rect_cpx Vss[3];
		rect_cpx Vat[2][ATmax][3];
		rect_cpx Vmp[3];
		void display(void);
		void upd_volt(ATinfo);
		VATS(void);
		~VATS(void);
};
class ATinfo
{
	public:
		friend class netdata;
		friend class ATpflow;
		int ATbus[2][ATmax];
		double ATpos[2][ATmax];
		void ATbus_order(void);
		int check_id(int,int);
		ATinfo(void);
		~ATinfo(void);
};
class TRinfo
{
	public:
		friend class busdata;
```

```
		friend class netdata;
		friend class train;
		int Tnum[2];
		int TRbus[2][Nmax];
		int TRid[2][Nmax];
		double TRpos[2][Nmax];
		double TRpow[2][Nmax];
		double TRvar[2][Nmax];
		void get_train(train (*)[Nmax],int,busdata,ATinfo);
		TRinfo(void);
		~TRinfo(void);
};
class busdata
{
	public:
		friend class netdata;
		friend class ATpflow;
		int Nbus;
		int bus_no[Bmax];
		double Pdem[Bmax];
		double Qdem[Bmax];
		int btype[Bmax];
		double buspos[Bmax];
		rect_cpx Volt[Bmax][3];
		void insertAT(VATS);
		void insertTR(TRinfo,train (*)[Nmax]);
		busdata(void);
		~busdata(void);
};
class netdata
{
	public:
		friend class ATpflow;
		int Nnet;
		int net_no[Lmax];
		int frombus[Lmax];
		int tobus[Lmax];
		double dist[Lmax];
		void netbuild(TRinfo,ATinfo,busdata);
		netdata(void);
		~netdata(void);
};
void busdata::insertAT(VATS Vats)
{
	// Bus number 1 is reserved for the substation
	// btype : 0 => Substation, 1 => AT, 2 => MPTSC, 3 => Train
	int i,j;
	Nbus=0;
	bus_no[Nbus]=Nbus+1;
	Pdem[Nbus]=0.0;
	Qdem[Nbus]=0.0;
	btype[Nbus]=0;
	buspos[Nbus]=0.0;
	Volt[0][0]=Vats.Vss[0];
	Volt[0][1]=Vats.Vss[1];
	Volt[0][2]=Vats.Vss[2];
	Nbus++;
	for (i=0;i<=_Ntrack-1;i++)
	{
		for (j=0;j<=_Nauto-2;j++)
		{
			bus_no[Nbus]=Nbus+1;
			Pdem[Nbus]=0.0;
			Qdem[Nbus]=0.0;
			btype[Nbus]=1;
			buspos[Nbus]=(j+1)*_ATdist*1000;
			Volt[Nbus][0]=Vats.Vat[i][j][0];
			Volt[Nbus][1]=Vats.Vat[i][j][1];
			Volt[Nbus][2]=Vats.Vat[i][j][2];
			Nbus++;
		}
	}
	bus_no[Nbus]=_MPTSC;
	Pdem[Nbus]=0.0;
	Qdem[Nbus]=0.0;
	btype[Nbus]=2;
	buspos[Nbus]=_Fdist*1000;
	Volt[_MPTSC-1][0]=Vats.Vmp[0];
	Volt[_MPTSC-1][1]=Vats.Vmp[1];
	Volt[_MPTSC-1][2]=Vats.Vmp[2];
	Nbus++;
}
void busdata::insertTR(TRinfo tr,train trn[2][Nmax])
{
	int i,j,m;
	for (i=0;i<=_Ntrack-1;i++)
	{
		for (j=0;j<=tr.Tnum[i]-1;j++)
		{
			bus_no[Nbus]=tr.TRbus[i][j];
			Pdem[Nbus]=tr.TRpow[i][j];
			Qdem[Nbus]=tr.TRvar[i][j];
			btype[Nbus]=3;
			buspos[Nbus]=tr.TRpos[i][j];
			m=tr.TRid[i][j];
			Volt[Nbus][0]=trn[i][m].volt[0];
			Volt[Nbus][1]=trn[i][m].volt[1];
			Volt[Nbus][2]=trn[i][m].volt[2];
			Nbus++;
		}
```

```
	}
	_NB=Nbus;
}
busdata::busdata(void)
{
	// Class constructor
}
busdata::~busdata(void)
{
	// Class destructor
}
void netdata::netbuild(TRinfo trn,ATinfo atr,busdata bus)
{
	int busorder[2][Bmax],ind,t,a,b,i,m,Tnum;
	busorder[0][0]=1;
	busorder[1][0]=1;
	for (i=0;i<=_Ntrack-1;i++)
	{
		Tnum=trn.Tnum[i];
		ind=0;
		t=0;a=0,b=1;
		if (i==0)
			m=1;
		else
			m=-1;
		while (ind!=1)
		{
		  if(trn.TRpos[i][(1-i)*(Tnum-1)-t*m]<atr.ATpos[i][a])
		  {
			busorder[i][b]=trn.TRbus[i][(1-i)*(Tnum-1)-t*m];
			if (t+1==Tnum)
				ind=2;
			else
				t++;
		  }
		  else
		  {
			busorder[i][b]=atr.ATbus[i][a];
			if (atr.ATbus[i][a]==_MPTSC)
				ind=3;
			else
				a++;
		  }
		  b++;
		  if (ind==2)
		  {
			while (atr.ATbus[i][a]!=_MPTSC)
			{
				busorder[i][b]=atr.ATbus[i][a];
				a++;
				b++;
			}
			busorder[i][b]=atr.ATbus[i][a];
			busorder[i][b+1]=0;
			ind=1;
		  }
		  else if (ind==3)
		  {
			while (t+1!=Tnum)
			{
			busorder[i][b]=trn.TRbus[i][(1-i)*(Tnum-1)-t*m];
				t++;
				b++;
			}
			busorder[i][b]=trn.TRbus[i][(1-i)*(Tnum-1)-t*m];
			busorder[i][b+1]=0;
			ind=1;
		  }
		}
	}
	ind=0;
	for (i=0;i<=_Ntrack-1;i++)
	{
		m=0;
		while (busorder[i][m+1]!=0)
		{
			net_no[ind]=ind+1;
			a=busorder[i][m];
			b=busorder[i][m+1];
			frombus[ind]=a;
			tobus[ind]=b;
			dist[ind]=fabs(bus.buspos[b-1]-bus.buspos[a-1]);
			if (fabs(dist[ind])<5.0)
				dist[ind]=5.0;
			m++;
			ind++;
		}
	}
	Nnet=ind;
}
netdata::netdata(void)
{
	// Class constructor
}
netdata::~netdata(void)
{
	// Class destructor
}
void ATinfo::ATbus_order(void)
{
```

```
	// Bus 1 is the substation
	int i,j,count=2;
	double temp,Nauto;
	_MPTSC=_Ntrack*(_Nauto-1)+2;
	Nauto=_Nauto;
	_ATdist=_Fdist/Nauto;
	for (i=0;i<=_Ntrack-1;i++)
	{
		for (j=0;j<=_Nauto-1;j++)
		{
			if (j==_Nauto-1)
			{
				ATbus[i][j]=_MPTSC;
				ATpos[i][j]=_Fdist*1000;
			}
			else
			{
				temp=j;
				ATbus[i][j]=count;
				ATpos[i][j]=(temp+1)*_ATdist*1000;
				count++;
			}
		}
	}
}
ATinfo::ATinfo(void)
{
	// Class constructor
}
ATinfo::~ATinfo(void)
{
	// Class destructor
}
void TRinfo::get_train(train tr[2][Nmax],int NT,busdata bus,ATinfo atr)
{
	int i,j,count,k;
	k=_MPTSC+1;
	for (i=0;i<=_Ntrack-1;i++)
	{
		count=0;
		for (j=0;j<=NT-1;j++)
		{
			if (tr[i][j].track_no>0)
			{
				TRbus[i][count]=k;
				TRpos[i][count]=tr[i][j].position;
				TRid[i][count]=j;
				TRpow[i][count]=tr[i][j].Pt;
				TRvar[i][count]=tr[i][j].Qt;
				count++;
				k++;
			}
		}
		Tnum[i]=count;
	}
	_NT=Tnum[0]+Tnum[1];
}
TRinfo::TRinfo(void)
{
	// Class constructor
}

TRinfo::~TRinfo(void)
{
	// Class destructor
}

void VATS::upd_volt(ATinfo atr)
{
	int i,j,m;
	for (i=0;i<=_Ntrack-1;i++)
	{
		for (j=0;j<=_Nauto-2;j++)
		{
			m=atr.ATbus[i][j]-1;
			Vat[i][j][0]=_Vbus[3*m];
			Vat[i][j][1]=_Vbus[3*m+1];
			Vat[i][j][2]=_Vbus[3*m+2];
		}
	}
	Vss[0]=_Vbus[0];
	Vss[1]=_Vbus[1];
	Vss[2]=_Vbus[2];
	Vmp[0]=_Vbus[3*(_MPTSC-1)];
	Vmp[1]=_Vbus[3*(_MPTSC-1)+1];
	Vmp[2]=_Vbus[3*(_MPTSC-1)+2];
}
VATS::VATS(void)
{
	int i,j;
	Vss[0]=rect_cpx(0.5*V0,0);
	Vss[1]=rect_cpx(100,0);
	Vss[2]=rect_cpx(0.5*V0,0);
	Vmp[0]=rect_cpx(0.5*V0,0);
	Vmp[1]=rect_cpx(100,0);
	Vmp[2]=rect_cpx(0.5*V0,0);
	for (i=0;i<=_Ntrack-1;i++)
	{
		for (j=0;j<=_Nauto-2;j++)
		{
```

```
                        Vat[i][j][0]=rect_cpx(0.5*V0,0);
                        Vat[i][j][1]=rect_cpx(100,0);
                        Vat[i][j][2]=rect_cpx(0.5*V0,0);
                }
        }
}
VATS::~VATS(void)
{
        // Class destructor
}
```

"systeminfo.h"

```
#include <iostream.h>
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>
#include <time.h>
#include "jcomplex.h"
const int Nmax=15;
const int Mmax=20;
const int STmax=20;
const int Bmax=100,Lmax=200,ATmax=5;
double simtime=0,dtime=0.5;
double Ulim,Ucoast,Dref[50],Href[50];
int _Nmodel;
double _TEFmax[Mmax];           // Maximum tractive effort (N)
double _BEF[Mmax];               Braking effort (N)
double _Mass[Nmax];             // Train effective mass (kg)
double _V0nom[Mmax];            // Nominal pantograph voltage (V)
double _U0max[Mmax];            // Maximum speed at the nominal voltage (m/s)
double _U0x[Mmax];              // First-corner speed at the nominal voltage (m/s)
double _Zxy0[Mmax];             // Speed ratio
double _Cdrag[Mmax][3];         // Drag force coefficients
int _Nst;                       // Total number of passenger stations
int _Ntrack;                    // Total number of tracks
int _Nauto;                     // Total number of ATs per track
int Ntck;                       // Number of a current track
int _train_m[4][Nmax];          // Train models
double _stoptime;               // Station stopping time (s)
double _headway;                // Headway (s)
double _stdist[STmax];          // Station distance (km)
double _Ulimit;                 // Maximum running speed (m/s)
double _Ucoasting;              // Coasting speed (m/s)
double _Fdist;                  // Feeding distance (km)
double _ATdist;                 // AT distance (km)
char _model_accurate;           // Traction drive model
int _MPTSC,_NB;
int _NT=0,_NT0=-1;              // Total number of Trains
int STC=1;      // Starting Condition:  l=> Nominal Start, 2=> Recursive Start
int sp_cont;
FILE *fpos1,*fspd1,*ftef1,*fpow1,*fvar1,*fats,*fcvg;
FILE *fpos2,*fspd2,*ftef2,*fpow2,*fvar2,*fsub;
FILE *ftvp_c1,*ftvp_c2,*ftvp_r1,*ftvp_r2,*ftvp_f1,*ftvp_f2;
double V0;
rect_cpx Yss[3][3],Yat[3][3],Jss[3],Yoh[3][3],Yre_ballast[3][3],Vbus0[Mmax];
rect_cpx _Vbus[Bmax],Zc,Zr,Zf;
rect_cpx _Gen,_Load,_Loss;
void loadsyst(void)
{
        int i,j,a,b;
        double temp;
        char name01[30];
        FILE *f1;
        printf("\nLoad System Specification : ");
        scanf("%s",name01);
        printf("\nDefine Traction Drive Model Used (1 or 2) : ");
        scanf("%d",&b);
        if (b==2)
                _model_accurate='B';
        else
                _model_accurate='A';

        f1=fopen(name01, "rt");
        fscanf(f1,"%d %d %lf %lf %lf %lf %lf %d %d\n",&_Nst,
                                &_Ntrack,&temp,&_headway,&_Ulimit,&_Ucoasting,
                                                &_Fdist,&_Nauto,&sp_cont);
        for (i=0;i<=_Nst-1;i++)
                fscanf(f1,"%lf",&_stdist[i]);
        fscanf(f1,"\n");
        for (i=0;i<=_Ntrack-1;i++)
        {
                a=10;
                j=0;
                while (a>0)
                {
                fscanf(f1,"%d ",&a);
                        _train_m[i][j]=a;
                        j++;
                }
                fscanf(f1,"\n");
        }
        fclose(f1);
        _stoptime=temp;
}
void loadmodel(void)
{
        int i;
```

```
	char name01[30];
	FILE *f1;
	printf("\nLoad Train Models : ");
	scanf("%s",name01);
	f1=fopen(name01, "rt");
	fscanf(f1,"%d\n",&_Nmodel);
	for (i=0;i<=_Nmodel-1;i++)
	{
		fscanf(f1,"%lf %lf %lf %lf %lf %lf %lf %lf %lf
			%lf\n",&_TEFmax[i],&_BEF[i],&_Mass[i],
			&_V0nom[i],&_U0max[i],&_U0x[i],&_Zxy0[i],
			&_Cdrag[i][0],&_Cdrag[i][1],&_Cdrag[i][2]);
	}
	fclose(f1);
}
void gradient_profile(void)
{
	int i=0,FCHECK=0;
	char name01[30];
	FILE *fgrads;
	printf("\nLoad Gradient Track Profiles : ");
	scanf("%s",name01);
	fgrads=fopen(name01,"rt");
	while (FCHECK!=EOF)
	{
		FCHECK=fscanf(fgrads,"%lf %lf\n",&Dref[i],&Href[i]);
		i++;
	}
	fclose(fgrads);
}
void initial(void)
{
	fats=fopen("fats.dat","wt");
	fsub=fopen("subst.dat","wt");
	fcvg=fopen("fcvg.dat","wt");
	// Track 1
	fpos1=fopen("dist1.dat","wt");
	fspd1=fopen("speed1.dat","wt");
	ftef1=fopen("teffort1.dat","wt");
	fpow1=fopen("power1.dat","wt");
	fvar1=fopen("react1.dat","wt");
	ftvp_c1=fopen("volt_c1.dat","wt");
	ftvp_r1=fopen("volt_r1.dat","wt");
	ftvp_f1=fopen("volt_f1.dat","wt");
	// Track 2
	fpos2=fopen("dist2.dat","wt");
	fspd2=fopen("speed2.dat","wt");
	ftef2=fopen("teffort2.dat","wt");
	fpow2=fopen("power2.dat","wt");
	fvar2=fopen("react2.dat","wt");
	ftvp_c2=fopen("volt_c2.dat","wt");
	ftvp_r2=fopen("volt_r2.dat","wt");
	ftvp_f2=fopen("volt_f2.dat","wt");
	Ulim=_Ulimit/3.6;
	Ucoast=_Ucoasting/3.6;
}

void final(void)
{
	fclose(fats);
	fclose(fsub);
	fclose(fcvg);
	fclose(fpos1);
	fclose(fspd1);
	fclose(ftef1);
	fclose(fpow1);
	fclose(fvar1);
	fclose(ftvp_c1);
	fclose(ftvp_r1);
	fclose(ftvp_f1);
	fclose(fpos2);
	fclose(fspd2);
	fclose(ftef2);
	fclose(fpow2);
	fclose(fvar2);
	fclose(ftvp_c2);
	fclose(ftvp_r2);
	fclose(ftvp_f2);
}
void writesimtime(void)
{
	fprintf(fats,"\n%.1f ",simtime);
	fprintf(fsub,"\n%.1f ",simtime);
	fprintf(fcvg,"\n%.1f ",simtime);
	// Track 1
	fprintf(fpos1,"\n%.1f ",simtime);
	fprintf(fspd1,"\n%.1f ",simtime);
	fprintf(ftef1,"\n%.1f ",simtime);
	fprintf(fpow1,"\n%.1f ",simtime);
	fprintf(fvar1,"\n%.1f ",simtime);
	fprintf(ftvp_c1,"\n%.1f ",simtime);
	fprintf(ftvp_r1,"\n%.1f ",simtime);
	fprintf(ftvp_f1,"\n%.1f ",simtime);
	// Track 2
	fprintf(fpos2,"\n%.1f ",simtime);
	fprintf(fspd2,"\n%.1f ",simtime);
	fprintf(ftef2,"\n%.1f ",simtime);
	fprintf(fpow2,"\n%.1f ",simtime);
	fprintf(fvar2,"\n%.1f ",simtime);
	fprintf(ftvp_c2,"\n%.1f ",simtime);
```

```
        fprintf(ftvp_r2,"\n%.1f ",simtime);
        fprintf(ftvp_f2,"\n%.1f ",simtime);
}
void ATsyst(void)
{
        FILE *f01;
        int i,FCHECK=0;
        char name1[30];
        rect_cpx Za,Zb;
        double Rre,Rb,Za_re,Za_im,Zb_re,Zb_im,G1,G2,G3,B1,B2,B3;
        double Rc,Xc,Rr,Xr,Rf,Xf;
        printf("\nLoad Power Supply System : ");
        scanf("%s",name1);
        f01=fopen(name1,"rt");
        fscanf(f01,"%lf %lf %lf %lf %lf %lf",&V0,&Za_re,&Za_im,
                &Zb_re,&Zb_im,&Rre,&Rb);
        for (i=0;i<=2;i++)
        {
                fscanf(f01,"%lf %lf %lf %lf %lf %lf",&G1,&B1,&G2,
                        &B2,&G3,&B3);
                Yoh[i][0]=rect_cpx(G1,B1);
                Yoh[i][1]=rect_cpx(G2,B2);
                Yoh[i][2]=rect_cpx(G3,B3);
        }
        for (i=0;i<=2;i++)
        {
                fscanf(f01,"%lf %lf %lf %lf %lf %lf",&G1,&B1,&G2,
                        &B2,&G3,&B3);
                Yat[i][0]=rect_cpx(G1,B1);
                Yat[i][1]=rect_cpx(G2,B2);
                Yat[i][2]=rect_cpx(G3,B3);
        }
        fscanf(f01,"%lf %lf %lf %lf %lf %lf",&Rc,&Xc,&Rr,
                &Xr,&Rf,&Xf);
        Zc=rect_cpx(Rc,Xc);
        Zr=rect_cpx(Rr,Xr);
        Zf=rect_cpx(Rf,Xf);
        fclose(f01);
        Za=rect_cpx(Za_re,Za_im);
        Zb=rect_cpx(Zb_re,Zb_im);
        Jss[0]=rect_cpx(V0,0)/Za;
        Jss[1]=rect_cpx(0,0);
        Jss[2]=rect_cpx(-V0,0)/Za;
        Yss[0][0]=rect_cpx(1,0)/Za+rect_cpx(1,0)/Zb;
        Yss[0][1]=rect_cpx(2,0)/Zb;
        Yss[0][2]=rect_cpx(1,0)/Za-rect_cpx(1,0)/Zb;
        Yss[1][0]=rect_cpx(2,Zb);
        Yss[1][1]=rect_cpx(-4,0)/Zb;
        Yss[1][2]=rect_cpx(2,0)/Zb;
        Yss[2][0]=rect_cpx(1,0)/Za-rect_cpx(1,0)/Zb;
        Yss[2][1]=rect_cpx(2,0)/Zb;
        Yss[2][2]=rect_cpx(1,0)/Za+rect_cpx(1,0)/Zb;
        Yre_ballast[0][0]=rect_cpx(0,0);
        Yre_ballast[0][1]=rect_cpx(0,0);
        Yre_ballast[0][2]=rect_cpx(0,0);
        Yre_ballast[1][0]=rect_cpx(0,0);
        Yre_ballast[1][1]=rect_cpx(1/Rb,0);
        Yre_ballast[1][2]=rect_cpx(0,0);
        Yre_ballast[2][0]=rect_cpx(0,0);
        Yre_ballast[2][1]=rect_cpx(0,0);
        Yre_ballast[2][2]=rect_cpx(0,0);
}
```

"tmm.h"

```
class train
{
  public:
    friend class ATpflow;
    friend class busdata;
    int st_id;                          // Station id
    int track_no;                       // Number of track
    int region;                         // Operating region
    char mode;                          // Mode of operation
    char model;                         // Traction drive model
    double TE;                          // Tractive effort (N)
    double speed;                       // Train speed (m/s): kph/3.6
    double position;    // Train position (km)
    rect_cpx volt[3];   // Pantograph voltage (V)
    rect_cpx Vprev[3];          // Previous time-step pantograph voltage (V)
    double Pt;                  // Train real power (W)
    double Qt;                  // Train reactive power (Var)
    double d2stop;      // Distance to the next passenger station (km)
    double stoptime;    // Stoptime (s)
    double BE;                  // Braking effort (N)
    double TEmax;       // Maximum tractive effort (N)
    double Meff;                // Effective mass (kg)
    double Umax;        // Maximum train speed (m/s): kph/3.6
    double _V0;                 // Nominal pantograph voltage (V)
    double U0_x;                // First corner speed (m/s): kph/3.6
    double Zxy;                 // Ux ~ Uy relation factor
    double acc;                 // Train acceleration rate (m/s^2)
    double C_drag[3]; // Drag force coefficients: a + bU + cU^2 (N)
    void initial(void);  // Program initialisation
    void addtrain(int,int);     // Add a train into a track
    void removetrain(void);     // Remove a train out of the track
    void getvolt(void);         // Pantograph voltage calculation (V)
    void TEcalc(void);          // Tractive effort calculation (N)
    void getacc(void);          // Acceleration rate (m/s^2)
```

```
    void getmode(void);             // Obtaining mode of operation
    void update_speed(void);        // Updating train speed
    void update_position(void);     // Updating train position
    void powercalc(void);                   // Power calculation
    double fgrad_calc(void);        // Gradient force calculation
    void trainmovement(void);       // Train movement main procedure
    void stationstop(void);                 // Check stopping status at a station
    train(void);                            // Create a train constructor
    ~train(void);                           // Create a train destructor
};
void train::getvolt(void)
{
        Vprev[0]=volt[0];
        Vprev[1]=volt[1];
        Vprev[2]=volt[2];
        if ((track_no>_Ntrack)||(track_no<0))
        {
                volt[0]=rect_cpx(0,0);
                volt[1]=rect_cpx(0,0);
                volt[2]=rect_cpx(0,0);
        }
}
void train::TEcalc(void)
{
        double Ux,Uy,voltage;
        rect_cpx V;
        V=volt[0]-volt[1];
        voltage=V.mag();
        switch (model)
        {
        case 'A':
        case 'a':
        Ux=U0_x*voltage/_V0;
        if (speed<=Ux)
                TE=TEmax;
                        else
                                TE=Ux*TEmax/speed;
                                break;
                case 'B':
        case 'b':
        Ux=U0_x*voltage/_V0;
                        Uy=Zxy*Ux;
        if (speed<=Ux)
                                TE=TEmax;
                        else if (speed<=Uy)
                                TE=Ux*TEmax/speed;
                        else
                                TE=Ux*Uy*TEmax/(speed*speed);
                        break;
    default:
        cout<<endl<<"Traction drive model does not exist";
        }
}
void train::getmode(void)
{
        double a,dU;
        a=BE/Meff;
        if (d2stop>0.5*speed*speed/a)
        {
                switch(sp_cont)
                {
                        case 1:     // Hyteresis speed control
                                if ((region==1)&&(speed<Ulim))
                                {
                                        mode='a';
                                        region=1;
                                }
                                else if ((region==1)&&(speed>=Ulim))
                                {
                                        mode='c';
                                        region=3;
                                }
                                else if ((region==3)&&(speed>Ucoast))
                                {
                                        mode='c';
                                        region=3;
                                }
                                else if ((region==3)&&(speed<Ucoast))
                                {
                                        mode='a';
                                        region=1;
                                }
                                else if ((region==3)&&(speed>Ulim))
                                {
                                        mode='c';
                                        region=3;
                                }
                                else
                                {
                                        mode='c';
                                        region=3;
                                }
                                break;
                        case 2:     // Proportional speed control
                                dU=Ulim-Ucoast;
                                if (speed<Ucoast)
                                        mode='a';
                                else if (speed<Ucoast+0.4*dU)
                                        mode='d';
                                else if (speed<Ucoast+0.6*dU)
```

```
                                        mode='c';
                                else if (speed>Ulim)
                                        mode='b';
                                else
                                        mode='e';
                                break;
                        }
                }
                else
                mode='b';
}
double train::fgrad_calc(void)
{
                int k;
                double ANG_Lp,gravity=9.81;
                k=position/1000;
                switch(track_no)
                {
                        case 1:
                        ANG_Lp=atan((Href[k+1]-Href[k])/1000);
                                break;
                        case 2:
                        ANG_Lp=atan((Href[k]-Href[k+1])/1000);
                                break;
                }
                return Meff*gravity*fabs(sin(ANG_Lp));
}
void train::getacc(void)
{
                double T,Fdrag,Fgrad,dU;
                dU=Ulim-Ucoast;
                Fdrag=C_drag[0]+C_drag[1]*speed+C_drag[2]*speed*speed;
                Fgrad=fgrad_calc();
                switch (mode)
                {
                        case 'a':       // Acceleration mode with full tractive effort
                                        TE=TEmax;
                                        T=TE-Fgrad-Fdrag;
                                        acc=T/Meff;
                                        break;
                        case 'b':       // Braking mode mode with full braking effort
                                        TE=0;
                                        T=-BE;
                                        acc=T/Meff;
                                        break;
                        case 'c':       // Coasting mode
                                        TE=0;
                                        T=-Fgrad-Fdrag;
                                        acc=T/Meff;
                                        break;
                        case 'd':       // Acceleration mode with proportional gain
                                        TE=-TEmax/0.4*speed+TEmax/0.4*(Ucoast+0.4*dU);
                                        T=TE-Fgrad-Fdrag;
                                        acc=T/Meff;
                                        break;
                        case 'e':       // Braking mode with proportional gain
                                        TE=-BE/0.4*speed+BE/0.4*(Ucoast+0.6*dU);
                                        T=TE-Fgrad-Fdrag;
                                        acc=T/Meff;
                                        break;
                }
}
void train::powercalc(void)
{
                double eff=0.8,tanPF=1.25;
                Pt=TE*speed/eff;
                Qt=Pt/tanPF;
}
void train::update_position(void)
{
                double dS;
                dS=speed*dtime+0.5*acc*dtime*dtime;
                switch (track_no)
                {
                        case 1:
                                        if (d2stop>0)
                                        {
                        position=position+dS;
                        d2stop=d2stop-dS;
                }
                if (position>_Fdist*1000)
                {
                        position=_Fdist*1000;
                        track_no=-1;
                }
                break;
        case 2:
                if (d2stop>0)
                {
                        position=position-dS;
                        d2stop=d2stop-dS;
                }
                if (position<0)
                {
                        position=0;
                        track_no=-2;
                }
                break;
        case -1:
                 position=_Fdist*1000;
```

```
                TE=0;
                break;
        case -2:
        case 0:
                 position=0;
                 TE=0;
                break;
        }
}
void train::update_speed(void)
{
                if ((track_no<=_Ntrack)&&(track_no>0))
                                speed=speed+acc*dtime;
                else
                                speed=0;
}
void train::trainmovement(void)
{
                getmode();
                TEcalc();
                getacc();
                update_speed();
                update_position();
                getvolt();
                powercalc();
}
void train::addtrain(int no_track,int train_model)
{
                int TM;
                model=_model_accurate;
                TM=train_model;
                track_no=no_track;
                TE=0;
                speed=0;
                Pt=0;
                Qt=0;
                region=1;
                _V0=_V0nom[TM-1];
                BE=_BEF[TM-1];
                TEmax=_TEFmax[TM-1];
                Meff=_Mass[TM-1];
                U0_x=_U0x[TM-1]/3.6;
                Zxy=_Zxy0[TM-1];
                volt[0]=rect_cpx(0.5*V0,0);
                volt[1]=rect_cpx(100,0);
                volt[2]=rect_cpx(0.5*V0,0);
                C_drag[0]=_Cdrag[TM-1][0];
                C_drag[1]=_Cdrag[TM-1][1];
                C_drag[2]=_Cdrag[TM-1][2];
                switch (track_no)
                {
                                case 1:
                                position=0;
                                                st_id=0;
                                                d2stop=(_stdist[st_id+1]-_stdist[st_id])*1000;
                                break;
                                case 2:
                                position=_Fdist*1000;
                                                st_id=_Nst-1;
                                                d2stop=(_stdist[st_id]-_stdist[st_id-1])*1000;
                                break;
                }
}
void train::stationstop(void)
{
  if (track_no!=0)
  {
                                if (d2stop<=0)
                                {
                                                stoptime=stoptime+dtime;
                                                TE=0;
                                speed=0;
                                }
                                else
                                                stoptime=0;
                                if (stoptime>=_stoptime)
                                {
                                   if (st_id<=_Nst)
                                   {
                                                switch (track_no)
                                                {
                                                                case 1:
                                                                                st_id++;
                                                d2stop=(_stdist[st_id+1]-_stdist[st_id])*1000;
                                                                                break;
                                                                case 2:
                                                                                st_id--;
                                                d2stop=(_stdist[st_id]-_stdist[st_id-1])*1000;
                                                                                break;
                                                }
                                                stoptime=0;
                                                region=1;
                                }
                        }
                }
}

train::train(void)
{
                track_no=0;
```

```
	region=0;
	speed=0;
	position=0;
	Pt=0;
	Qt=0;
	TE=0;
	volt[0]=rect_cpx(0,0);
	volt[1]=rect_cpx(0,0);
	volt[2]=rect_cpx(0,0);
}

train::~train(void)
{
	//		cout<<endl<<"The train was removed";
}
```

ii) Band matrix computation

File list

1. *mtsband_main.cpp* *(Main function)*
2. *ATpowernet.h*
3. *jcomplex.h* *(As described in Section i)*
4. *net_capture.h*
5. *systeminfo.h* *(As described in Section i)*
6. *tmm.h* *(As described in Section i)*

"mtsband_main.h"

```
#include "systeminfo.h"
#include "tmm.h"
#include "net_capture.h"
#include "ATpowernet.h"
void printresult(train trains[2][Nmax],VATS Vats)
{
	int i,j;
	// Track 1
	for (i=0;i<=Nmax-1;i++)
	{
		fprintf(fpos1,"%.2f ",trains[0][i].position);
		fprintf(fspd1,"%.2f ",trains[0][i].speed);
		fprintf(ftef1,"%.2f ",trains[0][i].TE);
		fprintf(fpow1,"%.2f ",trains[0][i].Pt);
		fprintf(fvar1,"%.2f ",trains[0][i].Qt);
		fprintf(ftvp_c1,"%.2f ",trains[0][i].volt[0].mag());
		fprintf(ftvp_r1,"%.2f ",trains[0][i].volt[1].mag());
		fprintf(ftvp_f1,"%.2f ",trains[0][i].volt[2].mag());
	}
	// Track 2
	for (i=0;i<=Nmax-1;i++)
	{
		if (trains[1][i].track_no!=0)
			fprintf(fpos2,"%.2f ",trains[1][i].position);
		else
			fprintf(fpos2,"%.2f ",_Fdist*1000);
		fprintf(fspd2,"%.2f ",trains[1][i].speed);
		fprintf(ftef2,"%.2f ",trains[1][i].TE);
		fprintf(fpow2,"%.2f ",trains[1][i].Pt);
		fprintf(fvar2,"%.2f ",trains[1][i].Qt);
		fprintf(ftvp_c2,"%.2f ",trains[1][i].volt[0].mag());
		fprintf(ftvp_r2,"%.2f ",trains[1][i].volt[1].mag());
		fprintf(ftvp_f2,"%.2f ",trains[1][i].volt[2].mag());
	}
	fprintf(fats,"%.2f %.2f ",Vats.Vss[0].mag(),Vats.Vss[1].mag());
	fprintf(fats,"%.2f %.2f ",Vats.Vmp[0].mag(),Vats.Vmp[1].mag());
	fprintf(fsub,"%.2f %.2f ",_Gen.re,_Gen.im);
	fprintf(fsub,"%.2f %.2f ",_Load.re,_Load.im);
	fprintf(fsub,"%.4f %.4f ",_Loss.re,_Loss.im);

	for (i=0;i<=_Ntrack-1;i++)
	{
		for (j=0;j<=_Nauto-2;j++)
			fprintf(fats,"%.2f %.2f %.2f ",Vats.Vat[i][j][0].mag(),
				Vats.Vat[i][j][1].mag(),Vats.Vat[i][j][2].mag());
	}
}
void main(void)
{
	int k,i,j,tid=0,Nstop,Tsim,IDST=0;
	double chktime=0,tsim;
	clock_t Tstart,Tstop;
	printf("\nEnter the Simulated Time Period (Hour) : ");
	scanf("%d",&Nstop);
	loadsyst();
	loadmodel();
	gradient_profile();
	initial();
	train trains[2][Nmax];
	ATinfo ATdata;
	TRinfo TRdata;
	ATdata.ATbus_order();
	busdata bus;
	netdata net;
	ATpflow powsys;
	ATsyst();
	VATS Vats;
	while (IDST==0)
```

"ATpflow.h"

```
class ATpflow
{
        public:
                friend class train;
                int NB;
                double max_err;
                rect_cpx Ybus[Bmax][Bmax];
                rect_cpx L[Bmax][Bmax];
                rect_cpx U[Bmax][Bmax];
                rect_cpx Vbus[Bmax];
                rect_cpx Vbus0[Bmax];
                rect_cpx Jbus[Bmax];
                rect_cpx Sload;
                rect_cpx Sloss;
                rect_cpx Sgen;

                void initial_ybus(void);
                void initial_jbus(void);
                void Form_Ybus(busdata,netdata,ATinfo);
                void Form_Jbus(busdata);
                void LUfactor(int);
                void LUsolve(int);
                void initial(busdata);
                void check_convergence(void);
                void powersolve(busdata);
                void runpflow(busdata,netdata,ATinfo,double);
                void upd_volt(TRinfo,train (*)[Nmax]);
                void lineflow(busdata,netdata);
                ATpflow(void);
                ~ATpflow(void);
};
void ATpflow::initial_ybus(void)
{
        int i,j;
        for (i=0;i<=3*NB-1;i++)
        {
                for (j=0;j<=3*NB-1;j++)
                        Ybus[i][j]=rect_cpx(0,0);
  }
}
void ATpflow::Form_Ybus(busdata bus,netdata net,ATinfo atr)
{
        int i,j,m,n,p,q;
        rect_cpx Ytemp,Yre_ball_half;
        initial_ybus();
        for (i=0;i<=net.Nnet-1;i++)
        {
                m=net.frombus[i];
```

```
{
        printf("\nEnter the Starting Condition (Select 1 or 2) ");
        printf("\n   1. Flat Start: ");
        printf("\n   2. Recursive Start: ");
        printf("\nThe choice is : ");
        scanf("%d",&STC);
        if ((STC==1)||(STC==2))
                IDST=1;
}
Tstart=clock();
for (k=0;k<=Nstop*120*60-1;k++)
{
        simtime=simtime+dtime;
        printf("\n%.1f ",simtime);
        writesimtime();
        if ((chktime<_headway)&&(tid>0))
                chktime=chktime+dtime;
        else
        {
                chktime=0;
                for (i=0;i<=_Ntrack-1;i++)
                        trains[i][tid].addtrain(i+1,1);
                tid++;
        }
        for (i=0;i<=_Ntrack-1;i++)
        {
                for (j=0;j<=tid-1;j++)
                {
                        trains[i][j].trainmovement();
                        trains[i][j].stationstop();
                }
        }
        TRdata.get_train(trains,tid);
        net.netbuild(TRdata,ATdata,bus);
        bus.insertAT(Vats);
        bus.insertTR(TRdata,trains);
        powsys.runpflow(bus,net,ATdata,25);
        powsys.upd_volt(TRdata,trains);
        Vats.upd_volt(ATdata);
        printresult(trains,Vats);
}
Tstop=clock();
Tsim=(Tstop-Tstart)/CLK_TCK;
tsim=Tsim;
final();
cout<<endl<<"Execution time = "<<(tsim/60)<<" Minutes";
}
```

```
                n=net.tobus[i];
                for (p=0;p<=2;p++)
                {
                        for (q=0;q<=2;q++)
                        {
                                Ytemp=Yoh[p][q]/rect_cpx(0.001*net.dist[i],0);
                        Yre_ball_half=Yre_ballast[p][q]*rect_cpx(0.001*net.dist[i]/2,0);
                          Ybus[3*m-3+p][3*n-3+q]=Ybus[3*m-3+p][3*n-3+q]-Ytemp;
                                Ybus[3*n-3+p][3*m-3+q]=Ybus[3*m-3+p][3*n-3+q];
                         Ybus[3*m-3+p][3*m-3+q]=Ybus[3*m-3+p][3*m-3+q]+Ytemp
                                                                +Yre_ball_half;
                Ybus[3*n-3+p][3*n-3+q]=Ybus[3*n-3+p][3*n-3+q]+Ytemp+Yre_ball_half;
                        }
                }
        }
        for (i=0;i<=_Ntrack-1;i++)
        {
                for (j=0;j<=_Nauto-1;j++)
                {
                        m=atbus[i][j];
                        for (p=0;p<=2;p++)
                        {
                                for (q=0;q<=2;q++)
                                {
                                        if ((i==1)&&(j==1))
                        Ybus[3*m-3+p][3*m-3+q]=Ybus[3*m-3+p][3*m-3+q]+Yat[p][q];
                                        else
                        Ybus[3*m-3+p][3*m-3+q]=Ybus[3*m-3+p][3*m-3+q]+Yat[p][q];
                                }
                        }
                }
        }
        for (i=0;i<=2;i++)
        {
                for (j=0;j<=2;j++)
                        Ybus[i][j]=Ybus[i][j]+Yss[i][j];
        }
}
void ATpflow::LUfactor(int bw)
{
        int i,j,k,dim,tmp1,tmp2,tmp3;
        dim=3*NB;
        for (i=0;i<=dim-1;i++)
        {
                for (j=0;j<=dim-1;j++)
                {
                        U[i][j]=Ybus[i][j];
                        if (i==j)
                                L[i][j]=rect_cpx(1,0);
                        else
                                L[i][j]=rect_cpx(0,0);
                }
        }
        for (j=0;j<=dim-2;j++)
        {
                tmp1=intmin(dim-1,j+bw);
                for (i=j+1;i<=tmp1;i++)
                {
                        L[i][j]=U[i][j]/U[j][j];
                        tmp2=intmax(0,i-bw);;
                        tmp3=intmin(dim-1,i+bw);
                        for (k=tmp2;k<=tmp3;k++)
                                U[i][k]=U[i][k]-L[i][j]*U[j][k];
                }
        }
}
void ATpflow::LUsolve(int bw)
{
        int i,j,dim,tmp1;
        dim=3*NB;
        rect_cpx Y[Bmax],XYsum;
        Y[0]=Jbus[0];
        for (i=1;i<=dim-1;i++)
        {
                XYsum=rect_cpx(0,0);
                tmp1=intmax(0,i-bw);;
                for (j=tmp1;j<=i-1;j++)
                XYsum=XYsum+L[i][j]*Y[j];
                Y[i]=Jbus[i]-XYsum;
        }
        Vbus[dim-1]=Y[dim-1]/U[dim-1][dim-1];
        for (i=3*NB-2;i>=0;i--)
        {
                XYsum=rect_cpx(0,0);
                tmp1=intmin(dim-1,i+bw);
                for (j=i+1;j<=tmp1;j++)
                XYsum=XYsum+U[i][j]*Vbus[j];
                Vbus[i]=(Y[i]-XYsum)/U[i][i];
        }
}
void ATpflow::initial_jbus(void)
{
        int i;
        for (i=0;i<=NB-1;i++)
        {
                Jbus[3*i]=rect_cpx(0,0);
                Jbus[3*i+1]=rect_cpx(0,0);
                Jbus[3*i+2]=rect_cpx(0,0);
        }
}
```

```
void ATpflow::Form_Jbus(busdata bus)
{
	int i,j;
	rect_cpx It,Jt[3];
	double P,Q;
	initial_jbus();
	for (i=0;i<=NB-1;i++)
	{
		P=-bus.Pdem[i];
		Q=bus.Qdem[i];
		It=rect_cpx(P,Q)/rect_conj(Vbus[3*i]-Vbus[3*i+1]);
		Jt[0]=It;
		Jt[1]=rect_cpx(0,0)-It;
		Jt[2]=rect_cpx(0,0);
		for (j=0;j<=2;j++)
		Jbus[3*i+j]=Jbus[3*i+j]+Jt[j];
	}
	for (i=0;i<=2;i++)
		Jbus[i]=Jbus[i]+Jss[i];
}
void ATpflow::check_convergence(void)
{
	int i;
	double err;
	max_err=0;
	for (i=0;i<=3*NB-1;i++)
	{
		err=fabs(rect_mag(Vbus[i]-Vbus0[i]));
		if (err>max_err)
		max_err=err;
	}
}
void ATpflow::powersolve(busdata bus)
{
	int i,bw;
	bw=3*(_Ntrack+1)-1;
	Form_Jbus(bus);
	LUfactor(bw);
	LUsolve(bw);
	check_convergence();
	for (i=0;i<=3*NB-1;i++)
	{
		Vbus0[i]=Vbus[i];
		_Vbus[i]=Vbus[i];
	}
}
void ATpflow::initial(busdata bus)
{
	int i;
	rect_cpx Vp(0.5*V0,0),Vn(100,0);
	for (i=0;i<=NB-1;i++)
	{
		switch (STC)
		{
			case 1:
				Vbus0[3*i]=Vp;
				Vbus0[3*i+1]=Vn;
				Vbus0[3*i+2]=Vp;
				Vbus[3*i]=Vp;
				Vbus[3*i+1]=Vn;
				Vbus[3*i+2]=Vp;
				break;
			case 2:
				if ((_NT-_NT0!=0)||(simtime==0.5))
				{
					Vbus0[3*i]=Vp;
					Vbus0[3*i+1]=Vn;
					Vbus0[3*i+2]=Vp;
					Vbus[3*i]=Vp;
					Vbus[3*i+1]=Vn;
					Vbus[3*i+2]=Vp;
				}
				else
				{
					Vbus0[3*i]=bus.Volt[i][0];
					Vbus0[3*i+1]=bus.Volt[i][1];
					Vbus0[3*i+2]=bus.Volt[i][2];
					Vbus[3*i]=bus.Volt[i][0];
					Vbus[3*i+1]=bus.Volt[i][1];
					Vbus[3*i+2]=bus.Volt[i][2];
				}
				break;
		}

	}
	_NT0=_NT;
}
void ATpflow::runpflow(busdata bus,netdata net,ATinfo atr,double
MAX_ERR)
{
	int Iter_no=0,Iter_max=1000,bw;
	double err=1000;
	bw=3*(_Ntrack+1)-1;
	NB=_NB;
	initial(bus);
	Form_Ybus(bus,net,atr);
	while (err>MAX_ERR)
	{
		powersolve(bus);
```

```
                    err=max_err;
                    Iter_no++;
                    if (Iter_no>Iter_max)
                    {
                    printf("\nThe system does not converge!!!");
                            break;
                    }
            }
}
void ATpflow::upd_volt(TRinfo tro,train tm[2][Nmax])
{
            int i,j,m,n;
            for (i=0;i<=_Ntrack-1;i++)
            {
                    for (j=0;j<=tro.Tnum[i]-1;j++)
                    {
                            m=trbus[i][j]-1;
                            n=tro.TRid[i][j];
                            trn[i][n].volt[0]=Vbus[3*m];
                            trn[i][n].volt[1]=Vbus[3*m+1];
                            trn[i][n].volt[2]=Vbus[3*m+2];
                    }
            }
}
ATpflow::ATpflow(void)
{
            // class constructor
}
ATpflow::~ATpflow(void)
{
            // class constructor
}
```

"net_capture.h"

```
int intmax(int a,int b)
{
            if (a>b)
                    return a;
            else
                    return b;
}

int intmin(int a,int b)
{
            if (a>b)
                    return b;
            else
                    return a;
}
class VATS        // Voltage class at the substation, AT points and the MPTSC
{
            public:
                    friend class ATpflow;
                    friend class busdata;
                    friend class ATinfo;
                    rect_cpx Vss[3];
                    rect_cpx Vat[2][ATmax][3];
                    rect_cpx Vmp[3];
                    void upd_volt(ATinfo);
                    VATS(void);
                    ~VATS(void);
};
class ATinfo
{
            public:
                    friend class netdata;
                    friend class ATpflow;
                    double ATpos[2][ATmax];
                    void ATbus_order(void);
                    int check_id(int,int);
                    ATinfo(void);
                    ~ATinfo(void);
};
class TRinfo
{
            public:
                    friend class busdata;
                    friend class netdata;
                    friend class train;
                    int Tnum[2];
                    int TRid[2][Nmax];
                    double TRpos[2][Nmax];
                    double TRpow[2][Nmax];
                    double TRvar[2][Nmax];
                    void get_train(train (*)[Nmax],int);
                    TRinfo(void);
                    ~TRinfo(void);
};
class busdata
{
            public:
                    friend class netdata;
                    friend class ATpflow;
                    int bus_no[Bmax];
                    double Pdem[Bmax];
                    double Qdem[Bmax];
                    int btype[Bmax];
                    double buspos[Bmax];
```

```
                rect_cpx Volt[Bmax][3];
                void insertAT(VATS);
                void insertTR(TRinfo,train (*)[Nmax]);
                busdata(void);
                ~busdata(void);
};
class netdata
{
        public:
                friend class ATpflow;
                int Nnet;
                int net_no[Lmax];
                int frombus[Lmax];
                int tobus[Lmax];
                double dist[Lmax];
                void netbuild(TRinfo,ATinfo,busdata);
                netdata(void);
                ~netdata(void);
};
void busdata::insertAT(VATS Vats)
{
        // Bus number 1 is reserved as the substation bus
        // btype : 0 => Substation, 1 => AT, 2 => MPTSC, 3 => Train
        int i,j,nbus=0;
        bus_no[nbus]=1;
        Pdem[nbus]=0.0;
        Qdem[nbus]=0.0;
        btype[nbus]=0;
        buspos[nbus]=bpos[nbus];
        Volt[0][0]=Vats.Vss[0];
        Volt[0][1]=Vats.Vss[1];
        Volt[0][2]=Vats.Vss[2];
        for (i=0;i<=_Ntrack-1;i++)
        {
                for (j=0;j<=_Nauto-2;j++)
                {
                        nbus=atbus[i][j]-1;
                        bus_no[nbus]=atbus[i][j];
                        Pdem[nbus]=0.0;
                        Qdem[nbus]=0.0;
                        btype[nbus]=1;
                        buspos[nbus]=bpos[nbus];
                        Volt[nbus][0]=Vats.Vat[i][j][0];
                        Volt[nbus][1]=Vats.Vat[i][j][1];
                        Volt[nbus][2]=Vats.Vat[i][j][2];
                }
        }
        nbus=_MPTSC-1;
        bus_no[nbus]=_MPTSC;
        Pdem[nbus]=0.0;
        Qdem[nbus]=0.0;
        btype[nbus]=2;
        buspos[nbus]=bpos[nbus];
        Volt[nbus][0]=Vats.Vmp[0];
        Volt[nbus][1]=Vats.Vmp[1];
        Volt[nbus][2]=Vats.Vmp[2];
}
void busdata::insertTR(TRinfo tr,train trn[2][Nmax])
{
        int i,j,m,nbus;

        for (i=0;i<=_Ntrack-1;i++)
        {
                for (j=0;j<=tr.Tnum[i]-1;j++)
                {
                        nbus=trbus[i][j]-1;
                        bus_no[nbus]=trbus[i][j];
                        Pdem[nbus]=tr.TRpow[i][j];
                        Qdem[nbus]=tr.TRvar[i][j];
                        btype[nbus]=3;
                        buspos[nbus]=bpos[nbus];
                        m=tr.TRid[i][j];
                        Volt[nbus][0]=trn[i][m].volt[0];
                        Volt[nbus][1]=trn[i][m].volt[1];
                        Volt[nbus][2]=trn[i][m].volt[2];
                }
        }
}
busdata::busdata(void)
{
        // Class constructor
}
busdata::~busdata(void)
{
        // Class destructor
}
void netdata::netbuild(TRinfo trn,ATinfo atr,busdata bus)
{
        int busorder[2][Bmax],nb[2],ind,t,a,b,i,m,Tnum,nbus,nt;
        busorder[0][0]=1;
        busorder[1][0]=1;
        bpos[0]=0.0;
        nb[0]=2;
        nb[1]=3;
        _MPTSC=trn.Tnum[0]+trn.Tnum[1]+_Ntrack*(_Nauto-1)+2;
        for (i=0;i<=_Ntrack-1;i++)
        {
                Tnum=trn.Tnum[i];
                ind=0;
                t=0;a=0,b=1;
```

```
                nt=(1-i)*(Tnum-1)-t*m;
                trbus[i][nt]=nb[i];
                busorder[i][b]=nb[i];
                bpos[nb[i]-1]=trn.TRpos[i][nt];
                nb[i]=nb[i]+2;
                t++;
                b++;
            }
            atbus[i][_Nauto-1]=_MPTSC;
            busorder[i][b]=_MPTSC;
            busorder[i][b+1]=0;
            ind=1;
        }
    }
    nbus=_MPTSC-1;
    _NB=_MPTSC;
    bpos[nbus]=_Fdist*1000;
    ind=0;
    for (i=0;i<=_Ntrack-1;i++)
    {
        m=0;
        while (busorder[i][m+1]!=0)
        {
            net_no[ind]=ind+1;
            a=busorder[i][m];
            b=busorder[i][m+1];
            frombus[ind]=a;
            tobus[ind]=b;
            dist[ind]=fabs(bpos[b-1]-bpos[a-1]);
            if (fabs(dist[ind])<5.0)
                dist[ind]=5.0;
            m++;
            ind++;
        }
    }
    Nnet=ind;
}
netdata::netdata(void)
{
    // Class constructor
}
netdata::~netdata(void)
{
    // Class destructor
}
void ATinfo::ATbus_order(void)
{
    // Bus 1 is the substation
    int i,j,count=2;
if (i==0)
    m=1;
else
    m=-1;
while (ind!=1)
{
    if (trn.TRpos[i][(1-i)*(Tnum-1)-t*m]<atr.ATpos[i][a])
    {
        nt=(1-i)*(Tnum-1)-t*m;
        trbus[i][nt]=nb[i];
        busorder[i][b]=nb[i];
        bpos[nb[i]-1]=trn.TRpos[i][nt];
        nb[i]=nb[i]+2;
        if (t+1==Tnum)
            ind=2;
        else
            t++;
    }
    else
    {
        atbus[i][a]=nb[i];
        busorder[i][b]=nb[i];
        bpos[nb[i]-1]=atr.ATpos[i][a];
        nb[i]=nb[i]+2;
        if (a==_Nauto-1)
            ind=3;
        else
            a++;
    }
    b++;
    if (ind==2)
    {
        while (a!=_Nauto-1)
        {
            atbus[i][a]=nb[i];
            busorder[i][b]=nb[i];
            bpos[nb[i]-1]=atr.ATpos[i][a];
            nb[i]=nb[i]+2;
            a++;
            b++;
        }
        atbus[i][a]=_MPTSC;
        busorder[i][b]=_MPTSC;
        busorder[i][b+1]=0;
        ind=1;
    }
    else if (ind==3)
    {
        while (t+1<=Tnum)
        {
```

```
		double temp,Nauto;
		Nauto=_Nauto;
		_ATdist=_Fdist/Nauto;
		for (i=0;i<=_Ntrack-1;i++)
		{
			for (j=0;j<=_Nauto-1;j++)
			{
				if (j==_Nauto-1)
				{
					ATpos[i][j]=_Fdist*1000;
				}
				else
				{
					temp=j;
					ATpos[i][j]=(temp+1)*_ATdist*1000;
					count++;
				}
			}
		}
}
void ATinfo::display(void)
{
		int i,j;
		printf("\nAT bus information\n\n");
		for (i=0;i<=_Ntrack-1;i++)
		{
			for (j=0;j<=_Nauto-1;j++)
				printf("%d ",atbus[i][j]);
			printf("\n");
		}
		printf("\nAT bus position\n\n");
		for (i=0;i<=_Ntrack-1;i++)
		{
			for (j=0;j<=_Nauto-1;j++)
				printf("%12.2f ",ATpos[i][j]);
			printf("\n");
		}
}
ATinfo::ATinfo(void)
{
		// Class constructor
}
ATinfo::~ATinfo(void)
{
		// Class destructor
}
void TRinfo::get_train(train tr[2][Nmax],int NT)
{
		int i,j,count;
		for (i=0;i<=_Ntrack-1;i++)
		{
			count=0;
			for (j=0;j<=NT-1;j++)
			{
				if (tr[i][j].track_no>0)
				{
					TRpos[i][count]=tr[i][j].position;
					TRid[i][count]=j;
					TRpow[i][count]=tr[i][j].Pt;
					TRvar[i][count]=tr[i][j].Qt;
					count++;
				}
			}
			Tnum[i]=count;
		}
		_NT=Tnum[0]+Tnum[1];
}
TRinfo::TRinfo(void)
{
		// Class constructor
}

TRinfo::~TRinfo(void)
{
		// Class destructor
}
void VATS::upd_volt(ATinfo atr)
{
		int i,j,m;
		for (i=0;i<=_Ntrack-1;i++)
		{
			for (j=0;j<=_Nauto-2;j++)
			{
				m=atbus[i][j]-1;
				Vat[i][j][0]=_Vbus[3*m];
				Vat[i][j][1]=_Vbus[3*m+1];
				Vat[i][j][2]=_Vbus[3*m+2];
			}
		}
		Vss[0]=_Vbus[0];
		Vss[1]=_Vbus[1];
		Vss[2]=_Vbus[2];
		Vmp[0]=_Vbus[3*(_MPTSC-1)];
		Vmp[1]=_Vbus[3*(_MPTSC-1)+1];
		Vmp[2]=_Vbus[3*(_MPTSC-1)+2];
}
VATS::VATS(void)
{
		int i,j;
		Vss[0]=rect_cpx(0.5*V0,0);
```

```
	Vss[1]=rect_cpx(100,0);
	Vss[2]=rect_cpx(0.5*V0,0);
	Vmp[0]=rect_cpx(0.5*V0,0);
	Vmp[1]=rect_cpx(100,0);
	Vmp[2]=rect_cpx(0.5*V0,0);
	for (i=0;i<=_Ntrack-1;i++)
	{
		for (j=0;j<=_Nauto-2;j++)
		{
			Vat[i][j][0]=rect_cpx(0.5*V0,0);
			Vat[i][j][1]=rect_cpx(100,0);
			Vat[i][j][2]=rect_cpx(0.5*V0,0);
		}
	}
}
VATS::~VATS(void)
{
	// Class destructor
}
```

G.3 MATLAB Codes for Single-phase AC/DC PWM Power Converter

File list

1. *pwmsim.m*
2. *swfn1.m*
3. *swfn.m*
4. *fangle.m*
5. *fdistr.m*
6. *pcalc1.m*
7. *dsim.m*
8. *fpid*
9. *fzpwm.m*
10. *frpwm.m*
11. *fipwm.m*

```
function [is,vdc,If,vf,idc,t]=pwmsim(N,x0,Lmodel)
% Single-phase PWM power converter simulation programme
% With PID controllers
global Rs Ls Rf Lf Cf Cr ws m delta Vsp Ps Qs Is Vdc phi Vs T Im Rm Pm L_model MI DELTA
L_model=Lmodel;
fs=50;
ws=2*pi*fs;
Ts=1/fs;
Rs=0.05;
Ls=0.981e-3;
Rf=0.01265;
Lf=0.253e-3;
Cf=10e-3;
Cr=10e-3;
Im=555.5556;
Rm=3.24;
Pm=1e6;
Vsp=1500;
Xs=ws*Ls;
m_max=0.95;m_min=0.45;
delta_max=30;delta_min=10;
kpq=0.2;kiq=0.05;kdq=0.01;      % PID controller for Q
kpv=0.8;kiv=0.01;kdv=0.02;      % PID controller for Vdc
Vdc0=1800;                      % Initial Vdc
Qs0=0;                          % Initial Q
m0=0.8240;                      % Initial modulation index
delta0=16.5879;                 % Initial phase angle
m=m0;
delta=delta0;
MR=12;fmod=50;     % Modulation ratio and modulation frequency
Vdc_ref=1800;Qs_ref=0;          % reference for Vdc and Q
t=[];is=[];vdc=[];If=[];vf=[];idc=[];          % Initial state variables
for z=1:N
	tx=0:Ts/1000:Ts;
	[us,Vmod,Vcar]=swfn1(tx,fmod,MR);
	[Tstrt,Tstp,opmode]=fangle(tx,us);
	is1=[];vdc1=[];If1=[];vf1=[];idc1=[];t1=[];
	for k=1:length(opmode)
		tspan1=[Tstrt(k) Tstp(k)];
		switch opmode(k)
		case 0
		[tz,x]=ode45('fzpwm',tspan1,x0);   % MATLAB function
		is1=[is1;x(:,1)];vdc1=[vdc1;x(:,2)];
		If1=[If1;x(:,3)];vf1=[vf1;x(:,4)];
		idc1=[idc1;tz-tz];
		t1=[t1;tz];M=length(t1);
		x0=[is1(M) vdc1(M) If1(M) vf1(M)];
		case 1
		[tz,x]=ode45('frpwm',tspan1,x0); % MATLAB function
		is1=[is1;x(:,1)];vdc1=[vdc1;x(:,2)];
		If1=[If1;x(:,3)];vf1=[vf1;x(:,4)];
		idc1=[idc1;x(:,1)];
		t1=[t1;tz];M=length(t1);
		x0=[is1(M) vdc1(M) If1(M) vf1(M)];
```

```
                case -1
                [tz,x]=ode45('fipwm',tspan1,x0); % MATLAB function
                is1=[is1;x(:,1)];vdc1=[vdc1;x(:,2)];
                If1=[If1;x(:,3)];vf1=[vf1;x(:,4)];
                idc1=[idc1;-x(:,1)];
                t1=[t1;tz];M=length(t1);
                x0=[is1(M) vdc1(M) If1(M) vf1(M)];
                otherwise
                        error('Programme terminated');
                return
        end
end
t=[t;t1+(z-1)*Ts];is=[is;is1];vdc=[vdc;vdc1];
If=[If;If1];vf=[vf;vf1];idc=[idc;idc1];
%vs1=Vsp*sin(ws*t1);
[iss,ts]=fdistr(is1,t1,500);
vs1=Vsp*sin(ws*ts);
vdcc=fdistr(vdc1,t1,500);
[Ps(z),Qs(z),Vdc(z),Is(z),Vs(z),phi(z)]=pcalc1(ts,iss,vs1,vdcc);
dVdc(z)=Vdc_ref-Vdc(z);
dQs(z)=Qs_ref*1e6-Qs(z);
switch z
case 1
  dVdc10=[0 0];
  dVdc0=[0 0];
  dQs10=[0 0];
  dQs0=[0 0];
case 2
  dVdc10=[dVdc1(z-1) 0];
  dVdc0=[dVdc(z-1) 0];
  dQs10=[dQs1(z-1) 0];
  dQs0=[dQs(z-1) 0];
otherwise
  dVdc10=[dVdc1(z-1) dVdc1(z-2)];
  dVdc0=[dVdc(z-1) dVdc(z-2)];
  dQs10=[dQs1(z-1) dQs1(z-2)];
  dQs0=[dQs(z-1) dQs(z-2)];
end
dVdc1(z)=fpid([kpv kiv kdv],dVdc(z),dVdc10,dVdc0);
dQs1(z)=fpid([kpq kiq kdq],dQs(z),dQs10,dQs0);
dVdc0=dVdc(z);
dVdc10=dVdc1(z);
dQs0=dQs(z);
dQs10=dQs1(z);
if L_model=='I'
  dDelta=2*Im*Xs*dVdc1(z)/m/Vsp/Vdc0/cos(delta0*pi/180);
  dm=-2*Xs*dQs1(z)/Vsp/Vdc0/cos(delta0*pi/180);
  MI(z)=m+dm;DELTA(z)=delta+dDelta*180/pi;
elseif L_model=='R'
  dDelta=2*Xs*dVdc1(z)/m0/Vsp/Rm/cos(delta0*pi/180);
  dm=-2*Xs*dQs1(z)/Vsp/Vdc0/cos(delta0*pi/180);
  if z==-2
        if dm>=0.02
                dm=0.02;
        elseif dm<=-0.02
                dm=-0.02;
        end
        if dDelta>=1*pi/180
                dDelta=1*pi/180;
        elseif dDelta<=-1*pi/180
                dDelta=-1*pi/180;
        end
  end
  MI(z)=m+dm;DELTA(z)=delta+dDelta*180/pi;
else
   error(' ')
   error('ERROR for using this PROGRAM')
   error(' ')
   return
end

if MI(z)>m_max
   MI(z)=m_max;
elseif MI(z)<m_min
   MI(z)=m_min;
end
if DELTA(z)>delta_max
   DELTA(z)=delta_max;
elseif DELTA(z)<delta_min
   DELTA(z)=delta_min;
 end
 m=MI(z);delta=DELTA(z);
 if z>80
    Qs_ref=0.2;
 elseif z>40
    Qs_ref=-0.5;
 end
 disp(0.02*z)
end
Ps=Ps*1e-6;Qs=Qs*1e-6;
T=0.02:0.02:0.02*N;
return
```

```
function [us,Vmod,Vcar]=swfn1(t,fmod,MR)
% PWM Switching Pattern Generation Program
for u=1:length(t)
  [us(u),Vmod(u),Vcar(u)]=swfn(t(u),fmod,MR);
end
```

```
return
```

```
function [us,Vmod,Vcar]=swfn(t,fmod,MR)
% Switching Pattern Generation Program
global m delta
Vmod=m*sin(2*pi*fmod*t-delta*pi/180);
Tcar=1/(MR*fmod);
ts=2*t/Tcar;
T=0.5*Tcar*(ts-floor(ts));
if Vmod>=0
        if T<=0.25*Tcar
                Vcar=1-4*T/Tcar;
        else
                Vcar=4*T/Tcar-1;
        end
        if Vmod>=Vcar
                us=1;
        else
                us=0;
        end
else
        if T<=0.25*Tcar
                Vcar=4*T/Tcar-1;
        else
                Vcar=1-4*T/Tcar;
        end
        if Vmod<=Vcar
                us=-1;
        else
                us=0;
        end
end
return
```

```
function [strt,stp,opmode]=fangle(t,us)
% Switching-period calculation program
M=length(t);
strt=[t(1)];
stp=[];
if us(1)==0
  opmode=0;
elseif us(1)==1
  opmode=1;
elseif us(1)==-1
  opmode=-1;
end
for k=1:M-1
  if us(k+1)==1
    if us(k+1)-us(k)==1
      stp=[stp;0.5*t(k)+0.5*t(k+1)];
      strt=[strt;0.5*t(k)+0.5*t(k+1)];
      opmode=[opmode;1];
    end
  elseif us(k+1)==-1
    if us(k+1)-us(k)==-1
      stp=[stp;0.5*t(k)+0.5*t(k+1)];
      strt=[strt;0.5*t(k)+0.5*t(k+1)];
      opmode=[opmode;-1];
    end
  elseif us(k+1)==0
    if us(k+1)-us(k)==-1
      stp=[stp;0.5*t(k)+0.5*t(k+1)];
      strt=[strt;0.5*t(k)+0.5*t(k+1)];
      opmode=[opmode;0];
    elseif us(k+1)-us(k)==1
      stp=[stp;0.5*t(k)+0.5*t(k+1)];
      strt=[strt;0.5*t(k)+0.5*t(k+1)];
      opmode=[opmode;0];
    end
  end
end
stp=[stp;t(length(t))];
return
```

```
function [y,t]=fdistr(x,t,N)
M=length(t);
t1=t(1):(t(M)-t(1))/(N-1):t(M);
y(1)=x(1);
y(N)=x(M);
for k=2:N-1
  id=find(t-t1(k)>=0);
  y(k)=0.5*(x(id(1))+x(id(1)+1));
end
t=t1';
y=y';
return
```

```
function [Ps,Qs,Vdc,Is,Vs,phi]=pcalc1(t,is,vs,vdc)
global m delta Vsp Ls ws Rm Im L_model vdc0 vdcc0 is0 iss0 ISS VDCC
N=length(t);
Xs=ws*Ls;
t=t-t(1);
numis=[62.8 0];denis=[1 62.8 98696];
```

```
numvdc=[0 63165.5];denvdc=[1 414.7 63165.5];
iss=dsim(numis,denis,is,t,iss0,is0);
iss0=[iss(N) iss(N-1)];is0=is(N);
vdcc=dsim(numvdc,denvdc,vdc,t,vdcc0,vdc0);
vdcc0=[vdcc(N) vdcc(N-1)];vdc0=vdc(N);
ISS=[ISS iss];
VDCC=[VDCC vdcc];
Is=rms(is,t);
Vs=Vsp/sqrt(2);
Vdc=mean(vdcc);
for k=1:N-1
  if (iss(k+1)<0)&(iss(k)>0)
    tch=0.5*(t(k)+t(k+1));
  end
end
phi=(tch-0.01)*360/0.02;
Ps=Vs*Is*cos(phi*pi/180);
Qs=Vs*Is*sin(phi*pi/180);
return
```

```
function y=dsim(num,den,r,t,y0,r0)
a=num(1);b=num(2);
c=den(1);d=den(2);f=den(3);
N=length(t);
y01=y0(1);y02=y0(2);
dt=t(2)-t(1);
for k=1:N
  switch k
  case 1
    y01=y0(1);
    y02=y0(2);
    r01=r0;
  case 2
    y01=y(1);
    y02=y0(1);
    r01=r(1);
  otherwise
    y01=y(k-1);
    y02=y(k-2);
    r01=r(k-1);
  end
  A=(c/dt^2)+d/dt+f;
  B=-d/dt-c*2/(dt^2);
  D=c/dt^2;
  E=a/dt+b;
  F=-a/dt;
  y(k)=E/A*r(k)+F/A*r01-B/A*y01-D/A*y02;
end
return
```

```
function y=fpid(Kpid,u,y0,u0)
dT=0.02;
kp=Kpid(1);ki=Kpid(2);kd=Kpid(3);
y=y0(1)+(kp+dT*ki+kd/dT)*u-(kp+2*kd/dT)*u0(1)+kd/dT*u0(2);
return
```

```
function xdot=fzpwm(t,x)
global Rs Ls Rf Lf Cf Cr ws Vsp L_model Im Rm Pm
if L_model=='I'
  Iload=Im;
elseif L_model=='R'
  Iload=x(2)/Rm;
elseif L_model=='P'
   if x(2)==0
     Iload=0;
   else
     Iload=Pm/x(2);
   end
else
  error('ERROR for using this programme');
  return
end
A=[-Rs/Ls 0 0 0;
  0 0 -1/Cr 0;
  0 1/Lf -Rf/Lf -1/Lf;
  0 0 1/Cf 0];
B=[1/Ls 0;
  0 -1/Cr;
  0 0;
  0 0];
U=[Vsp*sin(ws*t);Iload+t-t];
xdot=A*x+B*U;
return
```

```
function xdot=frpwm(t,x)
global Rs Ls Rf Lf Cf Cr ws Vsp L_model Im Rm Pm
if L_model=='I'
  Iload=Im;
elseif L_model=='R'
  Iload=x(2)/Rm;
elseif L_model=='P'
   if x(2)==0
     Iload=0;
   else
```

```
        Iload=Pm/x(2);
    end
else
  error('ERROR for using this programme');
  return
end
A=[-Rs/Ls -1/Ls 0 0;
  1/Cr 0 -1/Cr 0;
  0 1/Lf -Rf/Lf -1/Lf;
  0 0 1/Cf 0];
B=[1/Ls 0;
  0 -1/Cr;
  0 0;
  0 0];
U=[Vsp*sin(ws*t);Iload+t-t];
xdot=A*x+B*U;
return
```

```
function xdot=fipwm(t,x)
global Rs Ls Rf Lf Cf Cr ws Vsp Im Rm Pm L_model
if L_model=='I'
  Iload=Im;
elseif L_model=='R'
  Iload=x(2)/Rm;
elseif L_model=='P'
   if x(2)==0
      Iload=0;
   else
      Iload=Pm/x(2);
   end
else
  error('ERROR for using this programme');
  return
end
A=[-Rs/Ls 1/Ls 0 0;
  -1/Cr 0 -1/Cr 0;
  0 1/Lf -Rf/Lf -1/Lf;
  0 0 1/Cf 0];
B=[1/Ls 0;
  0 -1/Cr;
  0 0;
  0 0];
U=[Vsp*sin(ws*t);Iload+t-t];
xdot=A*x+B*U;
return
```


G.4 C++ Codes for Optimal AC Railway Power Flow Calculation

File List

1. orpf.cpp

```
#include <stdio.h>
#include <iostream.h>
#include <conio.h>
#include <complex.h>
#include <math.h>
#include <time.h>
#define pi 3.14159265358979
#define lmax 500
#define busmax 100
#define Nmax 100
#define FALSE 0
#define TRUE 1
#define Qlower -5.00
#define Qupper 5.00
complex ybus[busmax][busmax],x[Nmax],Sgen[busmax],Sdem[busmax],
      V0[busmax],Icalc[busmax],Vbus[busmax],YZt[busmax],
      IEt[busmax],aug[busmax][busmax];
double Vsch[busmax],aVsch[busmax],Pgen[busmax],Qgen[busmax],Pdem[busmax],
                  Qdem[busmax],max_err,SVP;
double rs[lmax],xs[lmax],half_yp[lmax],tap[lmax],Sbase,Vbase;
double grad[Nmax],xf[Nmax],x0[Nmax],opt_info[Nmax][Nmax];
int N,n_line,NB,max_iter,nbus[busmax],btype[busmax],no_iter=0;
int nline[lmax],fbus[lmax],tbus[lmax],Nx=2,cont_bus[20];
char name01[30],name02[30],name_out[30];
clock_t start,end;
FILE *f01;
double fsquare(double V)
{
          return V*V;
}
void mybus()
{
  int i,j,p,q,u,v;
  complex ypq,l_charge,t;
  FILE *f1;
  f1=fopen(name01, "rt");
  fscanf(f1,"%d %d\n",&NB,&n_line);
          N=NB-1;
  for (i=0;i<=n_line-1;i++)
    fscanf(f1,"%d %d %d %lf %lf %lf %lf\n",&nline[i],&fbus[i],&tbus[i],
                  &rs[i],&xs[i],&half_yp[i],&tap[i]);
```

```
  fclose(f1);
  for (u=1;u<=NB;u++)
  {
    for (v=1;v<=NB;v++)
          ybus[u-1][v-1]=complex(0,0);
  }
  for (j=1;j<=n_line;j++)
  {
    p=fbus[j-1];
    q=tbus[j-1];
    ypq=complex(1,0)/complex(rs[j-1],xs[j-1]);
    l_charge=complex(0,half_yp[j-1]);
    t=complex(tap[j-1],0);
    ybus[p-1][q-1]=ybus[p-1][q-1]-t*ypq;
    ybus[q-1][p-1]=ybus[p-1][q-1];
    ybus[p-1][p-1]=ybus[p-1][p-1]+t*ypq+t*(t-complex(1,0))*ypq+l_charge;
    ybus[q-1][q-1]=ybus[q-1][q-1]+t*ypq+(complex(1,0)-t)*ypq+l_charge;
  }
}
void pivot(int p,int *piv_foundp)
{
  double xmax;
  complex xtemp;
  int j,k,max_row;
  xmax=abs(aug[p][p]);
  max_row=p;
  for (j=1;j<N;j++)
  {
    if (abs(aug[j][p])>xmax)
    {
          xmax=abs(aug[j][p]);
      max_row=j;
    }
  }
  if (xmax==0)
          *piv_foundp=FALSE;
  else
  {
    *piv_foundp=TRUE;
    if (max_row!=p)
    {
      for (k=p;k<N+1;k++)
      {
          xtemp=aug[p][k];
          aug[p][k]=aug[max_row][k];
          aug[max_row][k]=xtemp;
      }
    }
  }
}
void gauss(int *sol_existsp)
{
  int j,k,p;
  complex piv_recip,xmult;
  for (p=0;*sol_existsp&&p<(N-1);p++)
  {
    pivot(p,sol_existsp);
    if (*sol_existsp)
    {
      piv_recip=complex(1,0)/aug[p][p];
      aug[p][p]=complex(1,0);
      for (k=p+1;k<N+1;k++)
          aug[p][k]*=piv_recip;
      for (j=p+1;j<N;j++)
      {
          xmult=-aug[j][p];
          aug[j][p]=complex(0,0);
          for (k=p+1;k<N+1;k++)
            aug[j][k]+=xmult*aug[p][k];
      }
    }
  }
  if (abs(aug[N-1][N-1])==0)
     *sol_existsp=FALSE;
  else if (*sol_existsp)
  {
    piv_recip=complex(1,0)/aug[N-1][N-1];
    aug[N-1][N-1]=complex(1,0);
    aug[N-1][N]*=piv_recip;
  }
}
void backsub()
{
  complex sum;
  int i,j;
  Vbus[N-1]=aug[N-1][N];
  for (i=N-2;i>=0;i--)
  {
    sum=complex(0,0);
    for (j=i+1;j<N;j++)
          sum+=aug[i][j]*Vbus[j];
    Vbus[i]=aug[i][N]-sum;
  }
}
void lin_solv()
{
  int sol_existsp;
  gauss(&sol_existsp);
  if (sol_existsp)
          backsub();
```

```
  else
  {
        printf("No unique solution\n");
        getch();
  }
}
void readbusdata()
{
  int i,h=0;
  FILE *f1;
  f1=fopen(name02, "rt");
  fscanf(f1,"%d %lf %lf %lf\n",&max_iter,&max_err,&Sbase,&Vbase);
  for (i=0;i<=NB-1;i++)
  {
    fscanf(f1,"%d %lf %lf %lf %lf %lf %lf %d\n",&nbus[i],
        &Vsch[i],&aVsch[i],&Pgen[i],&Qgen[i],&Pdem[i],
        &Qdem[i],&btype[i]);
V0[i]=complex(Vsch[i]*cos(pi/180*aVsch[i]),Vsch[i]*sin(pi/180*aVsch[i]));
    if (btype[i]==5)
    {
        cont_bus[h]=nbus[i];
        h++;
    }
    switch (btype[i])
    {
      case 1:
      case 2:
      case 5:
        Sgen[i]=complex(Pgen[i],Qgen[i]);
        Sdem[i]=complex(Pdem[i],Qdem[i]);
        break;
      case 3:
      case 4:
        YZt[i]=complex(Pgen[i],Qgen[i]);
        IEt[i]=complex(Pdem[i]*cos(pi/180*Qdem[i]),
              Qdem[i]*sin(pi/180*Qdem[i]));
        break;
    }
  }
  Nx=h;
  fclose(f1);
}
void icalc()
{
        int i;
        for (i=0;i<=NB-2;i++)
        {
              switch (btype[i])
              {
                case 1:
                case 5:
                  Icalc[i]=conj((Sgen[i]-Sdem[i])/V0[i])-ybus[i][NB-1]*V0[NB-1];
                  break;
                case 2:
                  Icalc[i]=Sgen[i]-Sdem[i]-ybus[i][NB-1]*V0[NB-1];
                  break;
                case 3:
                  Icalc[i]=(IEt[i]-V0[i])/YZt[i]-ybus[i][NB-1]*V0[NB-1];
                  break;
                case 4:
                  Icalc[i]=IEt[i]-YZt[i]*V0[i]-ybus[i][NB-1]*V0[NB-1];
                  break;
              }
        }
}
void aug_form()
{
        int i,j;
        for (i=0;i<=NB-2;i++)
        {
              for (j=0;j<=NB-2;j++)
                    aug[i][j]=ybus[i][j];
              aug[i][N]=Icalc[i];
        }
}
void voltcopy()
{
        int i;
        for (i=0;i<=NB-2;i++)
        V0[i]=Vbus[i];
}
double error_check()
{
        double voltage_error=0;
        int i;
        for (i=0;i<=NB-2;i++)
        {
              if (voltage_error<(abs(Vbus[i]/V0[i]-complex(1,0))))
                    voltage_error=abs(Vbus[i]/V0[i]-complex(1,0));
        }
  return voltage_error;
}
void Volt_print()
{
  int s;
  printf("\n<Bus voltage>\n");
printf("=============================================================\n");
  printf("  Bus   Voltage-magnitude(p.u.)   Voltage-angle(Degree)\n");
printf("=============================================================\n");
  for (s=0;s<=NB-1;s++)
```

```
                printf("%4d%19.6lf%25.6lf\n",s+1,abs(V0[s])*Vbase/1000,
                        180/pi*arg(V0[s]));
printf("============================================================\n");
}
double lineflow()
{
  int i;
  complex Sloss;
  Sloss=complex(0,0);
  for (i=0;i<=n_line-1;i++)
  {
        complex y,Sflow1,Sflow2;
        int p,q;
        p=fbus[i]-1;
        q=tbus[i]-1;
        y=complex(1,0)/complex(rs[i],xs[i]);
        Sflow1=conj(V0[p])*(V0[p]-V0[q])*y
                +conj(V0[p])*V0[p]*complex(0,half_yp[i]);
        p=tbus[i]-1;
        q=fbus[i]-1;
        Sflow2=conj(V0[p])*(V0[p]-V0[q])*y
                +conj(V0[p])*V0[p]*complex(0,half_yp[i]);
        Sloss=Sloss+Sflow1+Sflow2;
  }
  return real(Sloss);
}
double pflow()
{
        double voltage_error,loss;
        int k;
        for (k=1;k<=max_iter;k++)
        {
                icalc();
                aug_form();
                int r,s;
                lin_solv();
                voltage_error=error_check();
                if (voltage_error<max_err)
                        break;
                else
                        voltcopy();
        }
        loss=lineflow();
        return loss;
}
void print_bus()
{
        int i;
        clrscr();
        cout<<"Bus data"<<endl;
        cout<<"============================================================
======"<<endl;
        cout<<" No.  abs(V)  arg(V)   Pgen    Qgen    Pdem    Qdem
                Remark"<<endl;
cout<<"============================================================"<
<endl;
        for (i=0;i<=NB-1;i++)
        {
                printf("%2d%8.4f%8.4f%8.4f%8.4f%8.4f%8.4f%2d\n",
                        nbus[i],Vsch[i],aVsch[i],Pgen[i],Qgen[i],Pdem[i],Qdem[i],btyp
                        e[i]);
        }
}
double vprofile(complex Vf[busmax],int M)
{
        double V=0;
        int i;
        for (i=0;i<=M-2;i++)
        V=V+fsquare(1-abs(Vf[i]));
  return V;
}
double fobj(double x1[Nmax])
{
        double LOSS;
        int b,i;
        for (i=0;i<=Nx-1;i++)
        {
                b=cont_bus[i];
                Qdem[b-1]=x1[i];
                Sdem[b-1]=complex(Pdem[b-1],Qdem[b-1]);
        }
        LOSS=pflow();
        SVP=vprofile(V0,NB);
        return LOSS;
}
void fgrad(double x1[Nmax],double g[Nmax])
{
        double h,fp,fn;
        int i;
        for (i=0;i<=Nx-1;i++)
        {
                h=0.001;
                x1[i]=x1[i]+h;
                fp=fobj(x1);
                x1[i]=x1[i]-2*h;
                fn=fobj(x1);
                g[i]=(fp-fn)/2/h;
                x1[i]=x1[i]+h;
        }
}
```

```
double flsearch(double z)
{
        double xt[Nmax];
        int i;
        for (i=0;i<=Nx-1;i++)
                xt[i]=x0[i]-z*grad[i];
        return fobj(xt);
}
double quadratic(double xl,double xu,double epsilon)
{
  double xmid=(xl+xu)/2,fl,fu,fmid,xopt;
  double blm,bmu,bul,clm,cmu,cul;
  fl=flsearch(xl);
  fu=flsearch(xu);
  fmid=flsearch(xmid);
  blm=xl*xl-xmid*xmid;bmu=xmid*xmid-xu*xu;bul=xu*xu-xl*xl;
  clm=xl-xmid;cmu=xmid-xu;cul=xu-xl;
  if (clm*fu+cmu*fl+cul*fmid==0)
        xopt=xmid;
  else
        xopt=0.5*(blm*fu+bmu*fl+bul*fmid)/(clm*fu+cmu*fl+cul*fmid);
  if (xopt<xl)
        xopt=xl;
  else if (xopt>xu)
        xopt=xu;
  if (xu-xl<epsilon)
        xopt=xopt;
  else if ((xopt<xmid)&&(xopt>xl))
        xopt=quadratic(xl,xmid,epsilon);
  else if ((xopt<xu)&&(xopt>xmid))
                xopt=quadratic(xmid,xu,epsilon);
  return xopt;
}
double findmax(double x1[Nmax])
{
  double fmax=0;
  int i;
  for (i=0;i<=Nx-1;i++)
  {
     if (fmax<fabs(x1[i]))
        fmax=fabs(x1[i]);
  }
  return fmax;
}
double lsearch()
{
        double f0,f1,xv,xl=0,epsilon=0.0001,xu,fg,k;
        f0=flsearch(0);
        f1=flsearch(1);
        if (f0<=f1)
                xu=1;
        else
        {
                xv=2;
                k=1;
                while (f0>f1)
                {
                        xv=xv+k*(xv-xl);
                        f1=flsearch(xv);
                        k=2*k;
                }
                xu=xv;
        }
        fg=quadratic(xl,xu,epsilon);
        return fg;
}
double steepest(double xlimit[Nmax][2],double grad_err)
{
  double lambda,g0,g1,dg=1,f;
  int i;
  fgrad(x0,grad);
  g0=findmax(grad);
  while (dg>grad_err)
  {
    no_iter++;
    lambda=lsearch();
    for (i=0;i<=Nx-1;i++)
    {
        xf[i]=x0[i]-lambda*grad[i];
    }
    for (i=0;i<=Nx-1;i++)
    {
        if (xf[i]>xlimit[i][1])
        xf[i]=xlimit[i][1];
        else if (xf[i]<xlimit[i][0])
      xf[i]=xlimit[i][0];
    }
    fgrad(xf,grad);
    for (i=0;i<=Nx-1;i++)
        x0[i]=xf[i];
    f=flsearch(0);
    g1=findmax(grad);
    dg=fabs(g1-g0);
    g0=g1;
    getch();
  }
  return f;
}
void main()
{
```

```
  double xlimit[Nmax][2],grad_err=0.0001,floss;
  clrscr();
  cout<<"\n Please Enter the Name of Line Data File : ";
  scanf("%s",name01);
  cout<<"\n Please Enter the Name of Bus Data File : ";
  scanf("%s",name02);
  start=clock();
  mybus();
  readbusdata();
  int b,i;
  for (i=0;i<=Nx-1;i++)
  {
    b=cont_bus[i];
    x0[i]=Qdem[b-1];
  }
  for (i=0;i<=Nx-1;i++)
  {
    xlimit[i][0]=Qlower;
    xlimit[i][1]=Qupper;
  }
  floss=steepest(xlimit,grad_err);
  end=clock();
  clrscr();
  cout<<"The optimal solution, x* = ";
  for (i=0;i<=Nx-1;i++)
  {
            printf("%12.8f ",x0[i]);
  }
  cout<<endl<<endl<<"f(x*) = ";
  printf("%12.8f MW\n",floss);
  cout<<endl<<"The sum of squares of the voltage deviations = ";
  printf("%12.8f \n",SVP);
  cout<<"The total number of iteration used was "<<no_iter<<endl;
  cout<<"The calculation time was "<<(end-start)/CLK_TCK<<"
          second"<<endl;
  getch();
  cout<<endl;
  Volt_print();
}
```

G.5 C++ Codes for Area Control Simulation

File List

1. *rsimopf.cpp* *(Optimal area control)*
2. *rsimupf.cpp* *(UPF control)*
3. *rsimsvc.cpp* *(SVC control)*

"rsimopf.cpp"

```
#include <stdio.h>
#include <iostream.h>
#include <iostream.h>
#include <complex.h>
#include <stdio.h>
#include <conio.h>
#include <complex.h>
#include <stdlib.h>
#include <time.h>
#define pi 3.14159265358979
#define lmax 500
#define busmax 100
#define Nmax 100
#define FALSE 0
#define TRUE 1
#define Max_no_train 40
complex ybus[busmax][busmax],x[Nmax],Sgen[busmax],Sdem[busmax],
      V0[busmax],Icalc[busmax],Vbus[busmax],YZt[busmax],
      IEt[busmax],aug[busmax][busmax],Ssub;
double Vsch[busmax],aVsch[busmax],Pgen[busmax],Qgen[busmax],Pdem[busmax],
          Qdem[busmax],max_err=1e-8,FLOSS;
double rs[lmax],xs[lmax],half_yp[lmax],tap[lmax],Sbase=1e6,Vbase=25e3;
double grad[Nmax],xf[Nmax],x0[Nmax],opt_info[Nmax][Nmax];
int N,Nx,n_line,NB,max_iter=1000,nbus[busmax],btype[busmax],
          cont_bus[40];
int nline[lmax],fbus[lmax],tbus[lmax],NT_FW,NT_BW,no_iter,cont_ind[40];
double Rlpu=0.00016/2,Xlpu=0.00048/2,Rsub=0.0008,Xsub=0.0048,Ypq=0,
          TAP=1.0;
double Qupper=0,Qlower=-2,Qcont[50],SVP;
FILE *f01,*f02,*f03,*f04,*f05,*f06;
double T_POSITION_FW[40],T_POWER_FW[40],T_POSITION_BW[40],
          T_POWER_BW[40];
int NTset_FW[12],NTset_BW[12];
double TEmax=150000,Umax=80/3.6,eff_mass=200000,L_acc=3000,
          L_dec=3000,L_from_st[40],L_to_st[40],Tstop=3600,Tstart=0,
          Tstep=0.5,U_TE=50/3.6,Tstation=30;
double tk,U[40],L[40],TE[40],ST_POS[40],PF_FW[40],PF_BW[40];
double a1=18500,a2=138.9,a3=57.0;
int st_ind[40],de_st[40];
int st_indb[40],de_stb[40],tkcount;
double D0[40],Df[40],U0[40],L0[40];
double L_from_stb[40],L_to_stb[40],Ub[40],Lb[40],TEb[40],
          ST_POS_b[40];
double D0b[40],Dfb[40],U0b[40],L0b[40];
double Dref[50],Href[50],Fgrad;
double fsquare(double V)
{
          return V*V;
```

```
}
void f_fillarray(double a[40],double F)
{
        int i;
        for (i=0;i<=39;i++)
        a[i]=F;
}
void i_fillarray(int a[40],int I)
{
        int i;
        for (i=0;i<=39;i++)
        a[i]=I;
}
void mybus()
{
  int j,p,q,u,v;
  complex ypq,l_charge,t;
        N=NB-1;
  for (u=1;u<=NB;u++)
  {
    for (v=1;v<=NB;v++)
        ybus[u-1][v-1]=complex(0,0);
  }
  for (j=1;j<=n_line;j++)
  {
    p=fbus[j-1];
    q=tbus[j-1];
    ypq=complex(1,0)/complex(rs[j-1],xs[j-1]);
    l_charge=complex(0,half_yp[j-1]);
    t=complex(tap[j-1],0);
    ybus[p-1][q-1]=ybus[p-1][q-1]-t*ypq;
    ybus[q-1][p-1]=ybus[p-1][q-1];
    ybus[p-1][p-1]=ybus[p-1][p-1]+t*ypq+t*(t-complex(1,0))*ypq+l_charge;
    ybus[q-1][q-1]=ybus[q-1][q-1]+t*ypq+(complex(1,0)-t)*ypq+l_charge;
  }
}
void creat_busdata()
{
  double PF;
  int i,h=0;
  for (i=0;i<=NB-1;i++)
  {
    nbus[i]=i+1;Vsch[i]=1.0;aVsch[i]=0;
    if (i<=NT_FW-1)
    {
      PF=PF_FW[NTset_FW[i]-1];
      Pgen[i]=0;
      Qgen[i]=0;
      Pdem[i]=T_POWER_FW[i]/Sbase;
      if (PF==0)
      {
        Qdem[i]=0;
        btype[i]=5;
        cont_ind[h]=NTset_FW[i];
        h++;
      }
      else
      {
        Qdem[i]=Pdem[i]/PF*sqrt(1-PF*PF);
        btype[i]=1;
      }
    }
    else if (i<=NT_FW+NT_BW-1)
    {
      PF=PF_BW[NTset_BW[i-NT_FW]-1];
      Pgen[i]=0;
      Qgen[i]=0;
      Pdem[i]=T_POWER_BW[i-NT_FW]/Sbase;
      if (PF==0)
      {
        Qdem[i]=0;
        btype[i]=5;
        cont_ind[h]=NTset_BW[i-NT_FW]+20;
        h++;
      }
      else
      {
        Qdem[i]=Pdem[i]/PF*sqrt(1-PF*PF);
        btype[i]=1;
      }
    }
    else if (i==NB-2)
    {
      Pgen[i]=0;
      Qgen[i]=0;
      Pdem[i]=0;
      Qdem[i]=0;
      btype[i]=1;
    }
    else if (i==NB-1)
    {
      Pgen[i]=0;
      Qgen[i]=0;
      Pdem[i]=0;
      Qdem[i]=0;
      btype[i]=0;
    }
  }
}
void readbusdata()
```

```
{
  int i,h=0;
  creat_busdata();
  for (i=0;i<=NB-1;i++)
  {
    V0[i]=complex(Vsch[i]*cos(pi/180*aVsch[i]),
            Vsch[i]*sin(pi/180*aVsch[i]));
    if (btype[i]==5)
    {
      cont_bus[h]=nbus[i];
      h++;
    }
    switch (btype[i])
    {
      case 1:
      case 2:
      case 5:
            Sgen[i]=complex(Pgen[i],Qgen[i]);
            Sdem[i]=complex(Pdem[i],Qdem[i]);
          break;
      case 3:
      case 4:
            YZt[i]=complex(Pgen[i],Qgen[i]);
            IEt[i]=complex(Pdem[i]*cos(pi/180*Qdem[i]),
                    Qdem[i]*sin(pi/180*Qdem[i]));
            break;
    }
  }
  Nx=h;
}
void pivot(int p,int *piv_foundp)
{
  double xmax;
  complex xtemp;
  int j,k,max_row;
  xmax=abs(aug[p][p]);
  max_row=p;
  for (j=1;j<N;j++)
  {
    if (abs(aug[j][p])>xmax)
    {
      xmax=abs(aug[j][p]);
      max_row=j;
    }
  }
  if (xmax==0)
    *piv_foundp=FALSE;
  else
  {
    *piv_foundp=TRUE;
    if (max_row!=p)
    {
      for (k=p;k<N+1;k++)
      {
        xtemp=aug[p][k];
        aug[p][k]=aug[max_row][k];
        aug[max_row][k]=xtemp;
      }
    }
  }
}
void gauss(int *sol_existsp)
{
  int j,k,p;
  complex piv_recip,xmult;
  for (p=0;*sol_existsp&&p<(N-1);p++)
  {
    pivot(p,sol_existsp);
    if (*sol_existsp)
    {
      piv_recip=complex(1,0)/aug[p][p];
      aug[p][p]=complex(1,0);
      for (k=p+1;k<N+1;k++)
            aug[p][k]*=piv_recip;
      for (j=p+1;j<N;j++)
      {
        xmult=-aug[j][p];
        aug[j][p]=complex(0,0);
        for (k=p+1;k<N+1;k++)
            aug[j][k]+=xmult*aug[p][k];
      }
    }
  }
  if (abs(aug[N-1][N-1])==0)
    *sol_existsp=FALSE;
  else if (*sol_existsp)
  {
    piv_recip=complex(1,0)/aug[N-1][N-1];
    aug[N-1][N-1]=complex(1,0);
    aug[N-1][N]*=piv_recip;
  }
}
void backsub()
{
  complex sum;
  int i,j;
  Vbus[N-1]=aug[N-1][N];
  for (i=N-2;i>=0;i--)
  {
    sum=complex(0,0);
```

```
    for (j=i+1;j<N;j++)
            sum+=aug[i][j]*Vbus[j];
    Vbus[i]=aug[i][N]-sum;
  }
}
void lin_solv()
{
  int sol_existsp;
  gauss(&sol_existsp);
  if (sol_existsp)
            backsub();
  else
            printf("No unique solution\n");
}
void icalc()
{
  int i;
  for (i=0;i<=NB-2;i++)
  {
    switch (btype[i])
    {
      case 1:
      case 5:
            Icalc[i]=conj((Sgen[i]-Sdem[i])/V0[i])-ybus[i][NB-1]*V0[NB-1];
          break;
      case 2:
            Icalc[i]=Sgen[i]-Sdem[i]-ybus[i][NB-1]*V0[NB-1];
          break;
      case 3:
            Icalc[i]=(IEt[i]-V0[i])/YZt[i]-ybus[i][NB-1]*V0[NB-1];
          break;
      case 4:
            Icalc[i]=IEt[i]-YZt[i]*V0[i]-ybus[i][NB-1]*V0[NB-1];
          break;
    }
  }
}
void aug_form()
{
  int i,j;
  for (i=0;i<=NB-2;i++)
  {
    for (j=0;j<=NB-2;j++)
        aug[i][j]=ybus[i][j];
    aug[i][N]=Icalc[i];
  }
}
void voltcopy()
{
  int i;
  for (i=0;i<=NB-2;i++)
    V0[i]=Vbus[i];
}
double error_check()
{
  double voltage_error=0;
  int i;
  for (i=0;i<=NB-2;i++)
  {
    if (voltage_error<(abs(Vbus[i]/V0[i]-complex(1,0))))
      voltage_error=abs(Vbus[i]/V0[i]-complex(1,0));
  }
  return voltage_error;
}
void voltprint()
{
  int s,h=0;
  if (tkcount==0)
  {
    fprintf(f05,"%10.2lf",tk);
    for (s=0;s<=Max_no_train-1;s++)
    {
            if (s+1==NTset_FW[h])
            {
              fprintf(f05,"%15.8lf",abs(V0[s])*Vbase/1e3);
              h++;
            }
            else
              fprintf(f05,"%15.8lf",0);
    }
    h=0;
    for (s=0;s<=Max_no_train-1;s++)
    {
            if (s+1==NTset_BW[h])
            {
              fprintf(f05,"%15.8lf",abs(V0[s+NT_FW])*Vbase/1e3);
              h++;
            }
            else
              fprintf(f05,"%15.8lf",0);
            }
fprintf(f05,"%15.8lf%15.8lf\n",abs(V0[NT_FW+NT_BW])*Vbase/1e3,
            abs(V0[NT_FW+NT_BW+1])*Vbase/1e3);
  }
}
void qprint()
{
  int s;
  f_fillarray(Qcont,0);
  int h;
```

```
  for (h=0;h<=Nx-1;h++)
    Qcont[cont_ind[h]]=x0[h];
  if (tkcount==0)
  {
    fprintf(f06,"%10.2lf",tk);
    for (s=0;s<=49;s++)
      fprintf(f06,"%15.8lf",Qcont[s]);
    fprintf(f06,"\n");
  }
}
double lineflow()
{
  int i;
  complex Sloss;
  Sloss=complex(0,0);
  for (i=0;i<=n_line-1;i++)
  {
    complex y,Sflow1,Sflow2;
    int p,q;
    p=fbus[i]-1;
    q=tbus[i]-1;
    y=complex(1,0)/complex(rs[i],xs[i]);
    Sflow1=conj(V0[p])*(V0[p]-V0[q])*y
           +conj(V0[p])*V0[p]*complex(0,half_yp[i]);
    p=tbus[i]-1;
    q=fbus[i]-1;
    Sflow2=conj(V0[p])*(V0[p]-V0[q])*y
           +conj(V0[p])*V0[p]*complex(0,half_yp[i]);
    Sloss=Sloss+Sflow1+Sflow2;
  }
  return real(Sloss);
}
double pflow()
{
  double voltage_error,loss;
  int k;
  for (k=1;k<=max_iter;k++)
  {
    icalc();
    aug_form();
    int r,s;
    lin_solv();
    voltage_error=error_check();
    if (voltage_error<max_err)
      break;
    else
      voltcopy();
  }
  loss=lineflow();
  return loss;
}
double vprofile(complex Vf[busmax],int M)
{
        double V=0;
        int i;
        for (i=0;i<=M-2;i++)
  {
        V=V+fsquare(25-Vbase/1000*abs(Vf[i]));
  }
  return sqrt(V/(M-1));
}
double fobj(double x1[Nmax])
{
  double loss;
  int b,i;
  for (i=0;i<=Nx-1;i++)
  {
    b=cont_bus[i];
    Qdem[b-1]=x1[i];
      Sdem[b-1]=complex(Pdem[b-1],Qdem[b-1]);
  }
  loss=pflow();
  SVP=vprofile(V0,NB);
  return loss;
}
void fgrad(double x1[Nmax],double g[Nmax])
{
  double h,fp,fn;
  int i;
  for (i=0;i<=Nx-1;i++)
  {
    h=0.001;
    x1[i]=x1[i]+h;
    fp=fobj(x1);
    x1[i]=x1[i]-2*h;
    fn=fobj(x1);
    g[i]=(fp-fn)/2/h;
    x1[i]=x1[i]+h;
  }
}
double flsearch(double z)
{
  double xt[Nmax];
  int i;
  for (i=0;i<=Nx-1;i++)
    xt[i]=x0[i]-z*grad[i];
  return fobj(xt);
}

double quadratic(double xl,double xu,double epsilon)
```

```
{
  double xmid=(xl+xu)/2,fl,fu,fmid,xopt;
  double blm,bmu,bul,clm,cmu,cul;
  fl=flsearch(xl);
  fu=flsearch(xu);
  fmid=flsearch(xmid);
  blm=xl*xl-xmid*xmid;bmu=xmid*xmid-xu*xu;bul=xu*xu-xl*xl;
  clm=xl-xmid;cmu=xmid-xu;cul=xu-xl;
  if (clm*fu+cmu*fl+cul*fmid==0)
        xopt=xmid;
  else
        xopt=0.5*(blm*fu+bmu*fl+bul*fmid)/(clm*fu+cmu*fl+cul*fmid);
  if (xopt<xl)
        xopt=xl;
  else if (xopt>xu)
        xopt=xu;
  if (xu-xl<epsilon)
        xopt=xopt;
  else if ((xopt<xmid)&&(xopt>xl))
        xopt=quadratic(xl,xmid,epsilon);
  else if ((xopt<xu)&&(xopt>xmid))
        xopt=quadratic(xmid,xu,epsilon);
  return xopt;
}

double findmax(double x1[Nmax])
{
  double fmax=0;
  int i;
  for (i=0;i<=Nx-1;i++)
  {
    if (fmax<fabs(x1[i]))
        fmax=fabs(x1[i]);
  }
  return fmax;
}
double lsearch()
{
        double f0,f1,k,xv,xl=0,epsilon=0.001,xu,fg;
        f0=flsearch(0);
        f1=flsearch(1);
        if (f0<=f1)
                xu=1;
        else
        {
                xv=1;
                k=1;
                while (f0>f1)
                {
                        xv=xv+k*(xv-xl);
                        f1=flsearch(xv);
                        k=2*k;
                }
                xu=xv;
        }
        fg=quadratic(xl,xu,epsilon);
        return fg;
}
double steepest(double xlimit[Nmax][2],double grad_err)
{
  double lambda,g0,g1,dg=1,f;
  int i;
  no_iter=0;
  fgrad(x0,grad);
  g0=findmax(grad);
  while (dg>grad_err)
  {
    no_iter++;
    lambda=lsearch();
    for (i=0;i<=N-1;i++)
    {
      xf[i]=x0[i]-lambda*grad[i];
    }
    for (i=0;i<=Nx-1;i++)
    {
      if (xf[i]>xlimit[i][1])
        xf[i]=xlimit[i][1];
      else if (xf[i]<xlimit[i][0])
        xf[i]=xlimit[i][0];
    }
    fgrad(xf,grad);
    for (i=0;i<=Nx-1;i++)
    {
        x0[i]=xf[i];
    }
    f=flsearch(0);
    g1=findmax(grad);
    dg=fabs(g1-g0);
    g0=g1;
  }
  return f;
}
void opf()
{
  double xlimit[Nmax][2],grad_err=0.001,floss;
  mybus();
  readbusdata();
  int b,i;
  for (i=0;i<=Nx-1;i++)
  {
```

```
    b=cont_bus[i];
    x0[i]=Qdem[b-1];
  }
  for (i=0;i<=Nx-1;i++)
  {
    xlimit[i][0]=Qlower;
    xlimit[i][1]=Qupper;
  }
  floss=steepest(xlimit,grad_err);
  FLOSS=floss;
  SVP=vprofile(V0,NB);
  Ssub=conj(V0[NB-1])*(V0[NB-1]-V0[NB-2])/complex(Rsub,Xsub);
  if (tkcount==0)
  {
          fprintf(f04,"%8.2f%15.8f%15.8f%15.8f%15.8f%25.8f\n",tk,
                  abs(Ssub),floss,real(Ssub),imag(Ssub),SVP);
          voltprint();
        qprint();
  }
}
void init_pf()
{
  double PF=0;          // PF=0 is an controlled PWM locomotive
  // Power factor for the forward track set
PF_FW[0]=PF;PF_FW[2]=PF;PF_FW[4]=PF;PF_FW[6]=PF;
PF_FW[8]=PF;PF_FW[10]=PF;PF_FW[12]=PF;PF_FW[14]=PF;
PF_FW[16]=PF;PF_FW[18]=PF;PF_FW[1]=0.8;PF_FW[3]=0.8;
PF_FW[5]=0.8;PF_FW[7]=0.8;PF_FW[9]=0.8;PF_FW[11]=0.8;
PF_FW[13]=0.8;PF_FW[15]=0.8;PF_FW[17]=0.8;PF_FW[19]=0.8;
  // Power factor for the backward track set
PF_BW[0]=0.8;PF_BW[2]=0.8;PF_BW[4]=0.8;PF_BW[6]=0.8;
PF_BW[8]=0.8;PF_BW[10]=0.8;PF_BW[12]=0.8;PF_BW[14]=0.8;
PF_BW[16]=0.8;PF_BW[18]=0.8;PF_BW[1]=PF;PF_BW[3]=
PF;PF_BW[5]=PF;PF_BW[7]=PF;PF_BW[9]=PF;PF_BW[11]=PF;
PF_BW[13]=PF;PF_BW[15]=PF;PF_BW[17]=PF;PF_BW[19]=PF;
}
double tractive_effort(double V)
{
  if (V>U_TE)
    return TEmax*U_TE/V;
  else
    return TEmax;
}
void fgrad_calc(int ind, double Lref)
{
  int Lp,k;
  double ANG_Lp,gravity=9.81;
  char str[25],*endptr;
  itoa(Lref,str,10);
  Lp=strtol(str,&endptr,10);
  if (ind==-1)
    Lp=25000-Lp;
  k=Lp/1000;
  ANG_Lp=atan((Href[k+1]-Href[k])/1000);
  Fgrad=eff_mass*gravity*fabs(sin(ANG_Lp));
}
int mode_operation(double V,double TEF,double L2st,int z,int dest[40])
{
  double a;
  int mode_op;
  a=(-TEF-(a1+a2*V+a3*V*V)-Fgrad)/eff_mass;
  if (L2st<=-V*V/2/a)
  {
    dest[z]=1;
    mode_op=-1;
  }
  else if (V<Umax)
    mode_op=1;
  else
    mode_op=0;
  return mode_op;
}
double calc_acc(double V,double TEF,int opmode)
{
  double Fdrag,Fn;
  Fdrag=a1+a2*V+a3*V*V;
  switch (opmode)
  {
    case 1:
      Fn=TEF-Fdrag-Fgrad;
      break;
    case 0:
      Fn=-Fdrag-Fgrad;
      break;
    case -1:
      Fn=-TEF-Fdrag-Fgrad;
  }
  return Fn/eff_mass;
}
void gradient_profile()
{
  int i;
  FILE *fgrads;
  fgrads=fopen("gradprf.dat","rt");
  for (i=0;i<=29;i++)
  {
    fscanf(fgrads,"%lf %lf\n",&Dref[i],&Href[i]);
  }
  fclose(fgrads);
}
```

```
void main()
{
  double acc,tref[40],trefb[40],total_power;
  int v,opmode,Ntf=1,Ntb=1,NOTR;
  ST_POS[0]=0;ST_POS[1]=4;ST_POS[2]=9;ST_POS[3]=14;
  ST_POS[4]=20;ST_POS[5]=25;
  ST_POS_b[0]=0;ST_POS_b[1]=5;ST_POS_b[2]=11;ST_POS_b[3]=16;
  ST_POS_b[4]=21;ST_POS_b[5]=25;
  f01=fopen("pos01opf50.dat","wt");
  f02=fopen("speed01opf50.dat","wt");
  f03=fopen("power01opf50.dat","wt");
  f04=fopen("volt01opf50.dat","wt");
  f05=fopen("volt02opf50.dat","wt");
  f06=fopen("react01opf50.dat","wt");
  f_fillarray(tref,-1000);
  f_fillarray(U0,0);
  f_fillarray(L0,0);
  f_fillarray(D0,ST_POS[0]);
  f_fillarray(Df,ST_POS[1]*1000);
  i_fillarray(st_ind,0);
  i_fillarray(de_st,0);
  f_fillarray(trefb,-1000);
  f_fillarray(U0b,0);
  f_fillarray(L0b,0);
  f_fillarray(D0b,ST_POS_b[0]);
  f_fillarray(Dfb,ST_POS_b[1]*1000);
  i_fillarray(st_indb,0);
  i_fillarray(de_stb,0);
  init_pf();
  tkcount=0;
  tk=Tstart-Tstep;
  clrscr();
  gradient_profile();
  while (tk<=Tstop)
  {
    tk=tk+Tstep;
    NT_FW=0;
    NT_BW=0;
    total_power=0;
    if (tkcount==0)
    {
          fprintf(f01,"%lf ",tk);
          fprintf(f02,"%lf ",tk);
          fprintf(f03,"%lf ",tk);
    }
    if (tk-(Ntf-1)*300==0)
    Ntf++;
    for (v=0;v<=Max_no_train-1;v++)
    {
          if (tk-tref[v]<=Tstation)          // To check the stoping time at the station
          {
            U0[v]=U[v];
    L0[v]=L[v];
    TE[v]=0;
  }
  else if (v>Ntf-2)          // To check the train service status
  {
    TE[v]=0;
    L_from_st[v]=0;
    L_to_st[v]=0;
    L[v]=0;
    U[v]=0;
  }
  else          // Normal train service status
  {
        TE[v]=tractive_effort(U0[v]);
        L_from_st[v]=L0[v]-D0[v];
        L_to_st[v]=Df[v]-L0[v];
      fgrad_calc(1,L0[v]);
        opmode=mode_operation(U0[v],TE[v],L_to_st[v],v,de_st);
        acc=calc_acc(U0[v],TE[v],opmode);
        L[v]=L0[v]+U0[v]*Tstep+0.5*acc*Tstep*Tstep;
        U[v]=U0[v]+acc*Tstep;
        if (U[v]>Umax)
          U[v]=Umax;
       else if (U[v]<0)
          U[v]=0;
       if (L[v]>=25000)
       {
          L[v]=25000;
        U[v]=0;
        TE[v]=0;
       }
        U0[v]=U[v];
        L0[v]=L[v];
        if ((L_to_st[v]<=0)&&(U[v]<=0))
      {
          tref[v]=tk;
        st_ind[v]++;
        D0[v]=ST_POS[st_ind[v]]*1000;
        Df[v]=ST_POS[st_ind[v]+1]*1000;
        de_st[v]=0;
    }
  }
  if (tkcount==0)
        fprintf(f01,"%lf ",L0[v]);
        if (L0[v]>=25000)
        {
          if (tkcount==0)
        {
```

```
                fprintf(f02,"%lf ",0);
                fprintf(f03,"%lf ",0);
            }
           }
           else
           {
            total_power=total_power+TE[v]*U0[v]/0.8;
            if (tkcount==0)
            {
                        fprintf(f02,"%lf ",U0[v]);
                        fprintf(f03,"%lf ",TE[v]*U0[v]/0.8);
             }
           }
         if ((L0[v]<24999)&&(v<Ntf-1))
         {
              NT_FW++;
           NTset_FW[NT_FW-1]=v+1;
           T_POSITION_FW[NT_FW-1]=L0[v];
           T_POWER_FW[NT_FW-1]=TE[v]*U0[v]/0.8;
         }
}
if (tk-(Ntb-1)*300==0)
          Ntb++;
for (v=0;v<=Max_no_train-1;v++)
{
  if (tk-trefb[v]<=Tstation)    // To check the stoping time at the station
  {
    U0b[v]=Ub[v];
    L0b[v]=Lb[v];
    TEb[v]=0;
   }
   else if (v>Ntb-2)                          // To check the train service status
  {
    TEb[v]=0;
    L_from_stb[v]=0;
    L_to_stb[v]=0;
    Lb[v]=0;
    Ub[v]=0;
  }
  else                  // Normal train service status
  {
          TEb[v]=tractive_effort(U0b[v]);
          L_from_stb[v]=L0b[v]-D0b[v];
          L_to_stb[v]=Dfb[v]-L0b[v];
       fgrad_calc(-1,L0[v]);
          opmode=mode_operation(U0b[v],TEb[v],L_to_stb[v],v,de_stb);
          acc=calc_acc(U0b[v],TEb[v],opmode);
          Lb[v]=L0b[v]+U0b[v]*Tstep+0.5*acc*Tstep*Tstep;
          Ub[v]=U0b[v]+acc*Tstep;
          if (Ub[v]>Umax)
              Ub[v]=Umax;
        else if (Ub[v]<0)
              Ub[v]=0;
        if (Lb[v]>=25000)
        {
              Lb[v]=25000;
            Ub[v]=0;
            TEb[v]=0;
        }
            U0b[v]=Ub[v];
            L0b[v]=Lb[v];
            if ((L_to_stb[v]<=0)&&(Ub[v]<=0))
       {
              trefb[v]=tk;
            st_indb[v]++;
            D0b[v]=ST_POS_b[st_ind[v]]*1000;
            Dfb[v]=ST_POS_b[st_ind[v]+1]*1000;
            de_stb[v]=0;
        }
  }
  if (tkcount==0)
          fprintf(f01,"%lf ",L0b[v]);
   if (L0b[v]>=25000)
   {
          if (tkcount==0)
        {
                        fprintf(f02,"%lf ",0);
                        fprintf(f03,"%lf ",0);
        }
    }
    else
    {
         total_power=total_power+TEb[v]*U0b[v]/0.8;
          if (tkcount==0)
          {
                        fprintf(f02,"%lf ",U0b[v]);
                        fprintf(f03,"%lf ",TEb[v]*U0b[v]/0.8);
             }
    }
    if ((L0[v]<24999)&&(v<Ntb-1))
    {
          NT_BW++;
        NTset_BW[NT_BW-1]=v+1;
        T_POSITION_BW[NT_BW-1]=25000-L0[v];
        T_POWER_BW[NT_BW-1]=TE[v]*U0[v]/0.8;
     }
}
printf("%8.2lf\n",tk);
if (NT_FW>1)
{
```

```
    for (v=0;v<=NT_FW-2;v++)
    {
        nline[v]=v+1;fbus[v]=v+1;tbus[v]=v+2;
rs[v]=fabs(T_POSITION_FW[v]-T_POSITION_FW[v+1])*Rlpu/1000;
xs[v]=fabs(T_POSITION_FW[v]-T_POSITION_FW[v+1])*Xlpu/1000;
        half_yp[v]=Ypq;tap[v]=TAP;
    }
  }
  else
    v=0;
  NOTR=v;
  if (NT_BW>1)
  {
    for (v=0;v<=NT_BW-2;v++)
    {
        nline[NOTR+v]=NOTR+v+1;
        fbus[NOTR+v]=NOTR+v+1;tbus[NOTR+v]=NOTR+v+2;
        rs[NOTR+v]=fabs(T_POSITION_FW[v]-
                T_POSITION_FW[v+1])*Rlpu/1000;
        xs[NOTR+v]=fabs(T_POSITION_FW[v]-
                T_POSITION_FW[v+1])*Xlpu/1000;
        half_yp[NOTR+v]=Ypq;tap[NOTR+v]=TAP;
    }
  }
  else
        v=0;
  NOTR=NOTR+v;
  nline[NOTR]=NOTR+1;fbus[NOTR]=1;
  tbus[NOTR]=NT_FW+NT_BW+2;
  rs[NOTR]=fabs(T_POSITION_FW[0])*Rlpu/1000;
  xs[NOTR]=fabs(T_POSITION_FW[0])*Xlpu/1000;
  half_yp[NOTR]=Ypq;tap[NOTR]=TAP;
  nline[NOTR+1]=NOTR+2;fbus[NOTR+1]=NT_FW+1;
  tbus[NOTR+1]=NT_FW+NT_BW+2;
  rs[NOTR+1]=fabs(T_POSITION_BW[0])*Rlpu/1000;
  xs[NOTR+1]=fabs(T_POSITION_BW[0])*Xlpu/1000;
  half_yp[NOTR+1]=Ypq;tap[NOTR+1]=TAP;
  nline[NOTR+2]=NOTR+3;fbus[NOTR+2]=NT_FW;
  tbus[NOTR+2]=NT_FW+NT_BW+1;
  rs[NOTR+2]=fabs(T_POSITION_FW[NT_FW-1]-25000)*Rlpu/1000;
  xs[NOTR+2]=fabs(T_POSITION_FW[NT_FW-1]-25000)*Xlpu/1000;
  half_yp[NOTR+2]=Ypq;tap[NOTR+2]=TAP;
  nline[NOTR+3]=NOTR+4;fbus[NOTR+3]=NT_FW+NT_BW;
  tbus[NOTR+3]=NT_FW+NT_BW+1;
  rs[NOTR+3]=fabs(T_POSITION_BW[NT_BW-1]-25000)*Rlpu/1000;
  xs[NOTR+3]=fabs(T_POSITION_BW[NT_BW-1]-25000)*Xlpu/1000;
  half_yp[NOTR+3]=Ypq;tap[NOTR+3]=TAP;
  nline[NOTR+4]=NOTR+5;fbus[NOTR+4]=NT_FW+NT_BW+2;
  tbus[NOTR+4]=NT_FW+NT_BW+3;
  rs[NOTR+4]=Rsub;xs[NOTR+4]=Xsub;half_yp[NOTR+4]=Ypq;
  tap[NOTR+4]=TAP;
  NB=NT_FW+NT_BW+3;n_line=NB;
  opf();
  if (tkcount==0)
  {
        fprintf(f01,"\n");
        fprintf(f02,"\n");
        fprintf(f03,"%lf\n",total_power);
  }
 }
 fclose(f01);
 fclose(f02);
 fclose(f03);
 fclose(f04);
 fclose(f05);
 fclose(f06);
}
```

"rsimupf.cpp"

It is similar to the codes of "rsimopf.cpp". The following are a few functions that are changed.

```
void init_pf()
{
  double PF=1.0;
   // Power factor for the forward track set
PF_FW[0]=PF;PF_FW[2]=PF;PF_FW[4]=PF;PF_FW[6]=PF;
PF_FW[8]=PF;PF_FW[10]=PF;PF_FW[12]=PF;PF_FW[14]=PF;
PF_FW[16]=PF;PF_FW[18]=PF;PF_FW[1]=0.8;PF_FW[3]=0.8;
PF_FW[5]=0.8;PF_FW[7]=0.8;PF_FW[9]=0.8;PF_FW[11]=0.8;
PF_FW[13]=0.8;PF_FW[15]=0.8;PF_FW[17]=0.8;PF_FW[19]=0.8;
   // Power factor for the backward track set
PF_BW[0]=0.8;PF_BW[2]=0.8;PF_BW[4]=0.8;PF_BW[6]=0.8;
PF_BW[8]=0.8;PF_BW[10]=0.8;PF_BW[12]=0.8;PF_BW[14]=0.8;
PF_BW[16]=0.8;PF_BW[18]=0.8;PF_BW[1]=PF;PF_BW[3]=PF;
PF_BW[5]=PF;PF_BW[7]=PF;PF_BW[9]=PF;PF_BW[11]=PF;
PF_BW[13]=PF;PF_BW[15]=PF;PF_BW[17]=PF;PF_BW[19]=PF;
}
void pflow()
{
  double voltage_error,floss,SVP;
  int k;
  mybus();
  readbusdata();
  for (k=1;k<=max_iter;k++)
  {
    icalc();
    aug_form();
    int r,s;
```

```
    lin_solv();
    voltage_error=error_check();
    if (voltage_error<max_err)
            break;
    else
            voltcopy();
  }
  floss=lineflow();
  SVP=vprofile(V0,NB);
  Ssub=conj(V0[NB-1])*(V0[NB-1]-V0[NB-2])/complex(Rsub,Xsub);
  if (tkcount==0)
  {
            fprintf(f04,"%8.2f%15.8f%15.8f%15.8f%15.8f%15.8f\n",
                    tk,abs(Ssub),floss,real(Ssub),imag(Ssub),SVP);
            voltprint();
  }
}
```

"rsimsvc.cpp"

It is similar to the codes of "rsimopf.cpp". The following are a few functions that are changed.

```
void opf()
{
  double xlimit[Nmax][2],grad_err=0.0001,floss;
  mybus();
  readbusdata();
  int b,i;
  for (i=0;i<=Nx-1;i++)
  {
    b=cont_bus[i];
    x0[i]=Qdem[b-1];
  }
 xlimit[0][0]=-3;
 xlimit[0][1]=0;
  floss=steepest(xlimit,grad_err);
  FLOSS=floss;
  SVP=vprofile(V0,NB);
  Ssub=conj(V0[NB-1])*(V0[NB-1]-V0[NB-2])/complex(Rsub,Xsub);
  if (tkcount==0)
  {
            fprintf(f04,"%8.2f%15.8f%15.8f%15.8f%15.8f%25.8f\n",tk,
                    abs(Ssub),floss,real(Ssub),imag(Ssub),SVP);
            voltprint();
            qprint();
  }
}
void init_pf()
{
  double PF=0.8;                    // PF=0.8 is the base case power factor
  // Power factor for the forward track set
PF_FW[0]=PF;PF_FW[2]=PF;PF_FW[4]=PF;PF_FW[6]=PF;
PF_FW[8]=PF;PF_FW[10]=PF;PF_FW[12]=PF;PF_FW[14]=PF;
PF_FW[16]=PF;PF_FW[18]=PF;PF_FW[1]=PF;PF_FW[3]=PF;
PF_FW[5]=PF;PF_FW[7]=PF;PF_FW[9]=PF;PF_FW[11]=PF;
PF_FW[13]=PF;PF_FW[15]=PF;PF_FW[17]=PF;PF_FW[19]=PF;
  // Power factor for the backward track set
PF_BW[0]=PF;PF_BW[2]=PF;PF_BW[4]=PF;PF_BW[6]=PF;
PF_BW[8]=PF;PF_BW[10]=PF;PF_BW[12]=PF;PF_BW[14]=PF;
PF_BW[16]=PF;PF_BW[18]=PF;PF_BW[1]=PF;PF_BW[3]=PF;
PF_BW[5]=PF;PF_BW[7]=PF;PF_BW[9]=PF;PF_BW[11]=PF;
PF_BW[13]=PF;PF_BW[15]=PF;PF_BW[17]=PF;PF_BW[19]=PF;
}
```

G.6 C++ Codes for Cost Estimation and Design

File list

1. svcdes.cpp

This program is the modified version of "rsimsvc.cpp". The following are a few changes and additional functions to perform GA in the source codes. The main function of "rsimsvc.cpp" is renamed to "sub_main".

```
void main()
{
  int z,count=0;
  double fopt,Fcol[1000];
  popres[0]=4;popres[1]=4;
  popmin[0]=-15;popmin[1]=-15;
  popmax[0]=0;popmax[1]=0;
  Nbit=popres[0]+popres[1];
  randomize();
  initpop(ps);
  decode();
  findfmin();
  fopt=Fmin;
  Fcol[0]=fopt;
  printpop(0);
  printbest(0);
  for (z=1;z<=IterMax;z++)
  {
    setcross(1.0);
    crossover();
    setmutat(0.1);
    mutation();
    decode();
```

```
    findfmin();
    if (Fmin<fopt)
    {
            fopt=Fmin;
            count=0;
    }
    else
    {
            count++;
    }
    Fcol[z]=fopt;
    cout<<"Iteration "<<z<<endl;
    cout<<bestpop[0]<<" "<<bestpop[1]<<endl;
    printf("Objtive function = %12.2f\n\n",fopt);
  }
  printbest(-2);
  cout<<"Number of Iteration used was "<<z<<endl<<endl;
}
double int2num(int number)
{
  double x;
  char string[25];
  itoa(number,string,10);
  x=atof(string);
  return x;
}
int num2int(double number)
{
  int x;
  char string[25];
  gcvt(number,5,string);
  x=atoi(string);
  return x;
}
double opf(int u)
{
  double xlimit[Nmax][2],grad_err=0.0001,floss;
  mybus();
  readbusdata();
  int b,i;
  for (i=0;i<=Nx-1;i++)
  {
    b=cont_bus[i];
    x0[i]=Qdem[b-1];
  }
  for (i=0;i<=Nx-1;i++)
  {
    xlimit[i][0]=popnum[u][i];
    xlimit[i][1]=0;
  }
  floss=steepest(xlimit,grad_err);
  for (i=0;i<=Nx-1;i++)
  {
    printf("%12.8f ",x0[i]);
  }
  cout<<endl<<endl<<"f(x*) = ";
  printf("%12.8f MW\n",floss);
  Eloss=Eloss+floss*0.5;            // Update Total energy losses
  return floss;
}
void int2bin(int num,int Nb,double ps[Nmax])
{
  int remder,k;
  k=Nb-1;
  while (num!=0)
  {
    remder=num%2;
    num=num/2;
    if (remder==0)
            ps[k]=0;
    else
            ps[k]=1;
    k=k-1;
  }
  while (k>0)
  {
            ps[k]=0;
            k=k-1;
  }
}
void initpop(double pop[][Nmax])
{
  int count,i,j,k,num;
  double pss[BITmax];
  Fmin=InitMin;
  for (i=0;i<=Np-1;i++)
  {
    count=0;
    for (j=0;j<=Ne-1;j++)
    {
      num=random(pow(2,int2num(popres[j])));
      int2bin(num,popres[j],pss);
      for (k=0;k<=popres[j]-1;k++)
      {
            pop[i][count+k]=pss[k];
      }
      count=count+popres[j];
    }
  }
}
```

```
double feval(int u)
{
  double fcalc;
  sub_main();
  esave=(Eloss-Eloss_base)*ke*365*yr;
  return –eave-(popnum[u][0]+popnum[u][1])*kq;
}
void decode()
{
  int i,j,k,count;
  for (i=0;i<=Np-1;i++)
  {
    count=0;
    for (j=0;j<=Ne-1;j++)
    {
      popnum[i][j]=0;
      for (k=0;k<=popres[j]-1;k++)
            popnum[i][j]=popnum[i][j]+pow(2,popres[j]-k-1)*ps[i][count+k];
      popnum[i][j]=popmin[j]+(popmax[j]-
            popmin[j])*popnum[i][j]/(pow(2,popres[j])-1);
      count=count+popres[j];
    }
    popnum[i][Ne]=feval(i);
  }
}
void findfmin()
{
  int i,j,idmin=-1;
  for (i=0;i<=Np-1;i++)
  {
    if (popnum[i][Ne]<Fmin)
    {
      idmin=i;
      Fmin=popnum[i][Ne];
    }
  }
  if (idmin!=-1)
  {
    for (j=0;j<=Ne-1;j++)
      bestpop[j]=popnum[idmin][j];
  }
}
void setcross(double pxcross)
{
  int i,j,xc,rflag;
  for(i=0;i<=num2int(int2num(Np)*pxcross)-1;i++)
  {
    do
    {
      xc=random(Np);
      rflag=0;
      for (j=0;j<=i-1;j++)
      {
            if (xc==popxc[j])
            {
              rflag=1;
            break;
          }
      }
    }
    while (rflag==1);
      popxc[i]=xc;
  }
  Nxc=num2int(int2num(Np)*pxcross*0.5);
}

void setmutat(double pmutat)
{
  int i,j,xm,rflag;
  for(i=0;i<=num2int(int2num(Np)*pmutat)-1;i++)
  {
    do
    {
      xm=random(Np);
      rflag=0;
      for (j=0;j<=i-1;j++)
      {
            if (xm==popmu[j])
            {
              rflag=1;
            break;
          }
      }
    }
    while (rflag==1);
      popmu[i]=xm;
  }
  Nmu=num2int(int2num(Np)*pmutat);
}
void crossover()
{
  int i,j,idxcross;
  double temp1,temp2;
  for (i=0;i<=Nxc-1;i++)
  {
    idxcross=random(Nbit);
    for (j=0;j<=idxcross;j++)
    {
      temp1=ps[popxc[i]][j];
      temp2=ps[popxc[Nxc*2-1-i]][j];
```

```
        ps[popxc[i]][j]=temp2;
        ps[popxc[Nxc*2-1-i]][j]=temp1;
      }
    }
}
void mutation()
{
   int i,idmutat;
   for (i=0;i<=Nmu-1;i++)
   {
     idmutat=random(Nbit);
     if (ps[popmu[i]][idmutat]==0)
       ps[popmu[i]][idmutat]=1;
     else
       ps[popmu[i]][idmutat]=0;
   }
}
void printpop(int n)
{
   int i,j,k;
   printf("\n================\n");
   printf(" Iteration %d\n",n);
   printf("================\n");
   printf("%3d  ",1);
   for (j=0;j<=Nbit-1;j++)
     printf("%1.0f",ps[0][j]);
   for (k=0;k<=Ne-1;k++)
               printf("%4.0f",popnum[0][k]);
   printf("%18.4f",popnum[0][Ne]);
   cout<<endl;
   for(i=1;i<=Np-1;i++)
   {
     printf("%3d  ",i+1);
     for (j=0;j<=Nbit-1;j++)
       printf("%1.0f",ps[i][j]);
     for (k=0;k<=Ne-1;k++)
       printf("%4.0f",popnum[i][k]);
     printf("%18.4f",popnum[i][Ne]);
     cout<<" "<<endl;
   }
   cout<<endl;
}
void printbest(int n)
{
   int i;
   if (n<0)
   {
     printf("\n\n==================================\n");
     printf("The best solution for this problem:\n");
     printf("==================================\n\n");
   }
   else
     printf("Iteration %d:\n\n",n);
   for (i=0;i<=Ne-1;i++)
     printf("%4.0f",bestpop[i]);
   printf("%16.4f",Fmin);
   cout<<endl<<endl;
}
```

APPENDIX H

PUBLICATION

1. "Sequential linear power flow solution for AC electric railway power supply systems", 8th international conference on Computers in Railways, COMPRAIL2002, June 2002, Lemnos, Greece, pp. 531 – 540.

Published by

WIT Press
Ashurst Lodge, Ashurst, Southampton, SO40 7AA, UK
Tel: 44 (0) 238 029 3223; Fax: 44 (0) 238 029 2853
E-Mail: witpress@witpress.com
http://www.witpress.com

For USA, Canada and Mexico

Computational Mechanics Inc
25 Bridge Street, Billerica, MA 01821, USA
Tel: 978 667 5841; Fax: 978 667 7582
E-Mail: info@compmech.com
US site: http://www.compmech.com

British Library Cataloguing-in-Publication Data

A Catalogue record for this book is available
from the British Library

ISBN: 1-85312-913-5
ISSN: 1462-608X

The texts of the papers in this volume were set individually by the authors or under their supervision. Only minor corrections to the text may have been carried out by the publisher.

Printed and bound in Great Britain by Bookcraft

COMPUTERS IN RAILWAYS VIII

EDITORS:

J. ALLAN
Brown & Root, UK
R.J. HILL
University of Bath, UK
C.A. BREBBIA
Wessex Institute of Technology, UK
G. SCIUTTO
University Degli Studi di Genova, Italy
S. SONE
University of Kogokuin, Japan

ASSOCIATE EDITOR:

J. SAKELLARIS
University of Thessaloniki, Greece

ERGA OSE S.A.

WITPRESS Southampton, Boston

Sequential linear power flow solution for AC electric railway power supply systems

C.J. Goodman & T. Kulworawanichpong
The University of Birmingham, UK

Abstract

Characterisation of AC electric railway power supply systems requires a power flow algorithm as a basic tool to determine bus voltages, line currents or power flows through feeder lines. The execution of the algorithm needs efficient search methods. Among these, the Gauss-Seidel and the Newton-Raphson methods have been successfully developed to obtain network solutions over several decades. Over the previous decade, some emerging power-electronic compensation technologies have affected practical power system behaviours, including AC railways. This advancement in technology has introduced new complications to the existing AC railway power systems. Thus, the optimisation techniques need refinement and more efficient algorithms need to be developed. This should enable more accurate optimum solution and require considerably less calculation time.

This paper describes a novel AC railway power flow method, which uses a current load model instead of a classical power load model. Therefore, nodal analysis together with a simple search algorithm is sufficient to solve such a problem. Also, this simplified algorithm considerably reduces the overall calculation time. This new method is named Sequential Linear Power Flow and is explained in depth in this paper. For benchmarking, the well-known Newton-Raphson power flow algorithm has been used to perform identical calculations. The test systems were the modified standard IEEE 24-bus, 57-bus and a ten-train AC railway system, respectively. The computing results indicate that the proposed method requires significantly less execution time than that required by the Newton-Raphson method while the same accuracy is attained.

1 Introduction

The main function of an AC railway power supply system is to deliver electric energy to the electric locomotives which are connected to the system adequately, sufficiently and economically. To analyse reliability, stability and security of power supplies needs basic tools such as power flow algorithms for characterising voltages, currents and power flows through the feeder lines. As long as AC railway power systems can be described by conventional power supply analysis [1], classical power flow methods are applicable to obtain their solutions. The Gauss-Seidel and the Newton-Raphson methods are two well-known iterative techniques used for numerical solutions of power flow problems. For several decades, the Newton-Raphson method has been successfully developed and broadly accepted as the most powerful algorithm. Before describing the newly developed method, both classical power flow techniques are briefly reviewed for comparative purposes.

Suppose there is an N-bus power system, one of which is treated as a reference bus, whose voltage magnitude and angle are both specified. The bus labelled N will be selected as the reference bus throughout this paper. According to the Gauss-Seidel algorithm [2], bus voltages are updated at each iteration, repeatedly. The following is the expression to update bus voltages for the Gauss-Seidel method.

$$V_i^{(k+1)} = \frac{1}{Y_{ii}}\left\{\frac{P_i - jQ_i}{V_i^{(k)*}} - \sum_{m=1}^{i-1} Y_{im} V_m^{(k+1)} - \sum_{n=i+1}^{N} Y_{in} V_n^{(k)}\right\} \quad (1)$$

where the superscript k denotes the k-th iteration

$\mathbf{V}_i$ denotes the voltage vector at the i-th bus

P_i and Q_i are real and reactive power consumption at the i-th bus, respectively

$\mathbf{Y}_{im}$ is the i-th row and m-th column element of the system bus admittance matrix

* denotes the complex conjugate

For the Newton-Raphson method [2], Taylor Series expansion is used to approximate the power flow equations and decompose them into real and reactive power flow equations separately but there is still interaction between them. In the decomposed equations, all variables are real. Equations 2 and 3 represent real and reactive power mismatch equations after decomposition, respectively, and equation 4 shows the compact matrix form of the well-known Newton-Raphson power flow.

$$\Delta P_i = P_{i,sch} - \sum_{\substack{k=1 \\ k\neq i}}^{N} |Y_{ki} V_k V_i| \cos(\theta_{ki} + \delta_k - \delta_i) = 0 \quad (2)$$

$$\Delta Q_i = Q_{i,sch} + \sum_{\substack{k=1 \\ k\neq i}}^{N} |Y_{ki} V_k V_i| \sin(\theta_{ki} + \delta_k - \delta_i) = 0 \quad (3)$$

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} = \begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix} \begin{bmatrix} \Delta\delta \\ \Delta|V| \end{bmatrix} \quad (4)$$

where $P_{i,sch}$ and $Q_{i,sch}$ denote scheduled real and reactive power at the i-th bus, respectively

ΔP_i and ΔQ_i denote real and reactive power mismatches at the i-th bus, respectively

θ_{ki} denotes the angle of the complex vector $\mathbf{Y}_{ki}$

δ_i denotes the angle of the complex vector $\mathbf{V}_i$

$\mathbf{J}_1$, $\mathbf{J}_2$, $\mathbf{J}_3$ and $\mathbf{J}_4$ denote Jacobian sub-matrices

The Gauss-Seidel method proceeds by updating each unknown bus voltage separately from the first to the last and thus converges linearly, which needs much more execution time than the Newton-Raphson method which converges quadratically [3]. Although the Newton-Raphson method converges rapidly, usually within 3-5 iterations, it is still very complicated, requires substantial memory to store its variables and substantial computation because of complexity of its formulae.

In practice, AC railway power supply systems are special and there are some key differences compared to industrial power systems [4]. Most obviously the feed arrangement is single-phase and only has one feed (reference bus) as shown in figure 1. The loads are electric locomotives, can move along the rail-tracks and locomotives can be added into the system or be removed from the system at any instance, therefore the total number of buses, N, varies with time. Furthermore, with traction controllers such as shown in figure 2, the modelling of the load is more variable and impedance or current load models are more appropriate to describe the locomotive characteristics than the power load model [5]. These factors lead to a need to re-analyse the power flow problem carefully for the possibility of developing a new power flow technique that is simpler, but still converges rapidly and exploits the special properties of the railway case.

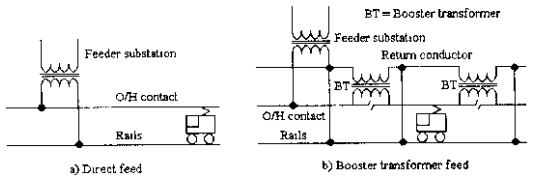


Figure 1: Single-phase AC railway feeding systems

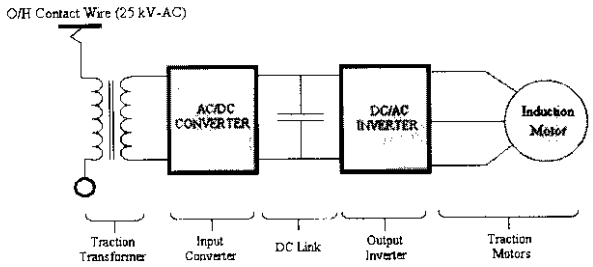


Figure 2: A simplified schematic diagram of an electric locomotive

2 Sequential linear power flow method

Nodal analysis is one of the well-known methods for solving the unknown voltage at each node in a linear circuit. No iteration is needed to obtain bus voltages when all loads are in the form of currents [6]. Since loads in utility power systems are normally modelled as power loads, nodal analysis has not been able to be used to solve such power flow problems. In contrast, in AC railways where current load models are often acceptable, it is possible to derive power flow equations by applying nodal analysis.

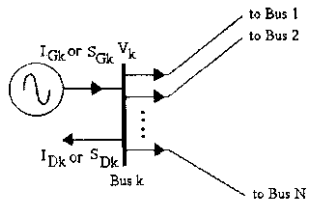


Figure 3: A general bus with generation, load and outgoing lines

As shown in figure 3, the net outgoing current flow from the k-th bus is obtained by applying KCL [6] as follows.

$$\sum_{\substack{i=1 \\ i\neq k}}^{N} y_{ki}(V_k - V_i) = I_{k,sch} \tag{5}$$

where y_{ki} denotes the admittance connected between the k-th bus and the i-th bus

$I_{k,sch}$ denotes the scheduled current at the k-th bus

Since the primitive admittance y_{ki} does not usually appear in any formulae used in network solvers and the N-th bus is the reference bus, equation 5 can be rewritten as equation 6 with the elements of the bus admittance matrix and the voltage at the N-th bus as given variables, as follows.

$$\sum_{i=1}^{N-1} Y_{ki} V_i = I_{k,sch} - Y_{kN} V_N \tag{6}$$

Similarly, with N-1 unknown bus voltages, the matrix form of the node equations for this system is generated as

$$\begin{bmatrix} Y_{11} & Y_{12} & \cdots & Y_{1,N-1} \\ Y_{21} & Y_{22} & \cdots & Y_{2,N-1} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{N-1,1} & Y_{N-1,2} & \cdots & Y_{N-1,N-1} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_{N-1} \end{bmatrix} = \begin{bmatrix} I_{1,sch} - Y_{1N} V_N \\ I_{2,sch} - Y_{2N} V_N \\ \vdots \\ I_{N-1,sch} - Y_{N-1,N} V_N \end{bmatrix} = \begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_{N-1} \end{bmatrix} \tag{7}$$

or in the short form of the matrix equation as

$$[\mathbf{Y}_{NODE}][\mathbf{V}]=[\mathbf{I}] \tag{8}$$

where $[\mathbf{Y}_{NODE}]$ is a matrix derived from $[\mathbf{Y}_{BUS}]$ by eliminating the N-th row and the N-th column.

Interestingly, equation 7 is linear when all loads are written as current models, not power models as industrial power systems. To extend the equation above to power load models, one possible method is to transform all the power loads into current loads. From the power-current relationship of a load, whose voltage is previously known somehow, the k-th element of the current matrix can simply be computed by equation 9.

$$I_k = I_{k,sch} - Y_{kN} V_N = \left[\frac{S_{k,sch}}{V_k}\right]^* - Y_{kN} V_N \tag{9}$$

where $S_{k,sch}$ denotes the scheduled complex power of the k-th bus.

Applying the Gauss iterative method, with given initial bus voltages, [V] is updated iteration-by-iteration, thus the bus voltage vector at the k+1 iteration can simply be obtained as equation 10.

$$[\mathbf{V}]^{(k+1)} = [\mathbf{Y}_{NODE}]^{-1}[\mathbf{I}]^{(k)} \tag{10}$$

As can be seen, unlike the Jacobian matrix, $[\mathbf{Y}_{NODE}]^{-1}$ is completely known at the beginning and can be used repeatedly without re-calculation throughout the process. In addition, such a matrix is an (N-1)-by-(N-1) matrix whereas the Jacobian matrix is a 2(N-1)-by-2(N-1) matrix, which considerably reduces the memory required by the power flow programme.

This method transforms the non-linear matrix equation into a linear matrix equation at each iteration, so that the developed method is named the Sequential Linear Power Flow Method (SLPFM). In summary, the SLPFM is described by the flow chart in figure 4.

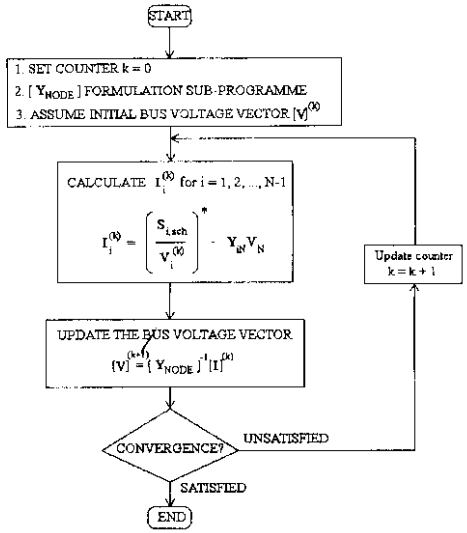
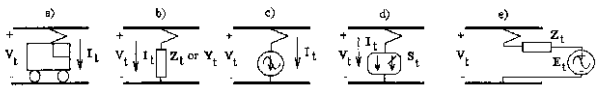


Figure 4: Flow chart for the SLPFM

3 Current load modelling of an electric locomotive

The power flow problem is generally concerned with obtaining system solutions, with the problem formulation being generated from all elements connected in the system. Therefore, each element has to be modelled as simply as possible to decrease problem complexity, but still adequately represent the original. Just as a load in an industrial power system is modelled by a power model, a locomotive in an AC railway can simply be represented in the same way. Nevertheless, because of certain differences, it is sometimes specified in other forms, such as current, impedance or some more complicated model as shown in figure 5, dependent on the type of electric locomotive [7].



a) The locomotive
b) The impedance or the admittance model
c) The current model
d) The power model
e) The linear model by Thevenin's equivalent circuit

Figure 5: Locomotive models for power flow analysis

The SLPFM employs the current model in figure 5-c to characterise the locomotives. Thus, the locomotive current model must be calculated from its dependence on the original load model. It can simply be obtained case-by-case as follows.

- The impedance or admittance model

$$I_t = \frac{V_t}{Z_t} \text{ or } Y_t V_t \tag{11}$$

- The power model

$$I_t = \left[\frac{S_t}{V_t}\right]^* \tag{12}$$

- The linear model by Thevenin's equivalent circuit

$$I_t = \frac{V_t - E_t}{Z_t} \tag{13}$$

Using the above equations, any locomotive model can easily be transformed into a current model and the transformation needs only simple numerical calculations. This leads to the SLPFM being capable of solving precisely complicated railway systems where there exist several locomotives modelled by various different models.

4 Simulation results

By using the latest version of MATPOWER [8] (a well-known Newton-Raphson power flow programme for the MATLAB environment developed by the School of Electrical Engineering, Cornell University, USA in December 1997) to solve the modified (to have only one reference bus) standard IEEE 24-bus, 57-bus and a ten-train railway test systems as shown in figures 6,7 and 8 respectively, the effectiveness of the developed method compared to the standard Newton-Raphson power flow has been tested.

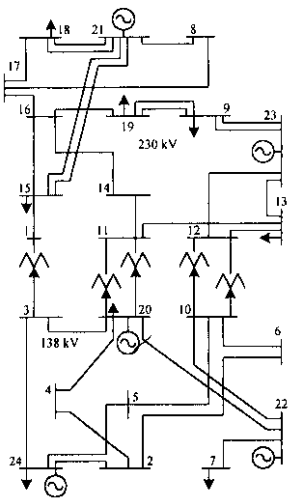


Figure 6: The modified standard IEEE 24-bus test system

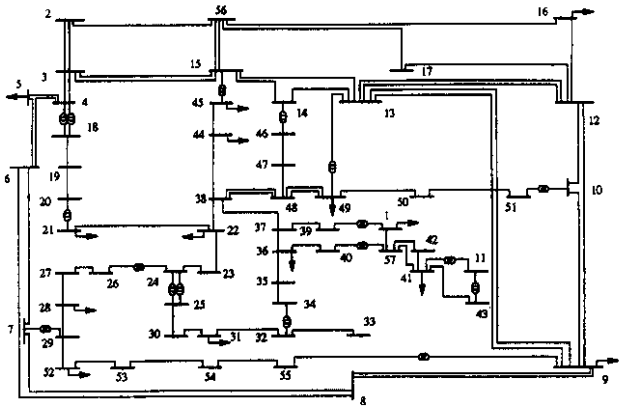


Figure 7: The modified standard IEEE 57-bus test system

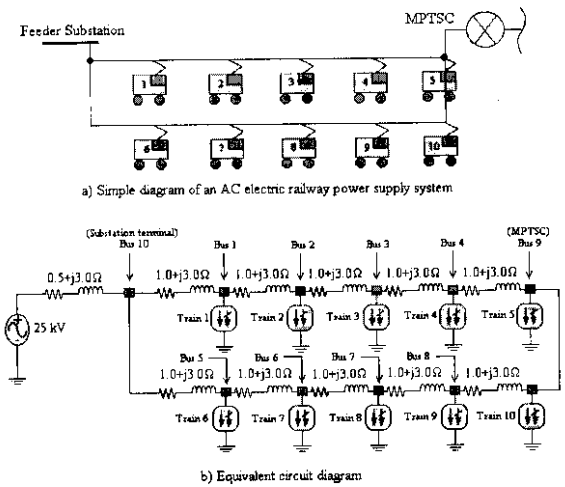


Figure 8: The ten-train AC railway test system with all power load models

These tests were performed on a Pentium 133 MHz, 48 MB RAM, with MATLAB 5.3 student version. Of the power flow solutions for three cases, only the simulated result from the ten-train railway test system has been selected to present as an example in this paper and are shown in table 1. Table 2 is a table summary that illustrates the effectiveness for each of the three cases by giving the required iterations and the calculation times. It should be noted that both power flow methods used $1x10^{-8}$ per-unit as the terminating condition; maximum power mismatch in per-unit for the Newton-Rapson method and maximum voltage error in per-unit for the SLPFM.

Table 1. Results from the ten-train railway test system

BUS	Voltage magnitude in per-unit		Voltage angle in degree	
	Newton-Raphson	SLPFM	Newton-Raphson	SLPFM
1	0.841	0.841	-8.9	-8.9
2	0.795	0.795	-11.6	-11.6
3	0.761	0.762	-13.9	-13.9
4	0.739	0.739	-15.5	-15.5
5	0.841	0.841	-8.9	-8.9
6	0.795	0.795	-11.6	-11.6
7	0.761	0.762	-13.9	-13.9

8	0.739	0.739	-15.5	-15.5
9	0.782	0.728	-16.4	-16.4
10	0.898	0.898	-5.9	-5.9
11	1.000	1.000	0.0	0.0

Table 2. Required iteration and calculation time comparisons

Cases	The total number of iteration required		Calculation time in second	
	Newton-Raphson	SLPFM	Newton-Raphson	SLPFM
24 buses	5	19	0.61	0.33
57 buses	3	7	0.44	0.24
10 trains	5	21	0.44	0.22
Average calculation time per iteration			0.12	0.02

5 Conclusions

The results confirm the effectiveness of the SLPFM to solve AC railway power flow problems with their special features. Even though the developed method requires more iterations to obtain the solution, because of spending less calculation time per iteration, the overall calculation time is only approximately 50% of that used by the standard Newton-Raphson method. Significantly, the developed method does not require the calculation of derivatives. Furthermore, the SLPFM uses current load models computed from various alternative load models, hence enabling the use of different types of load models characterising different types of locomotives in one problem.

6 References

[1] Elgerd, O.L., *Electric Energy System Theory*, McGraw-Hill, 1983.
[2] Stephenson, W., *Elements of Power System Analysis*, McGraw-Hill, 1982.
[3] Elden, L., *Numerical Analysis: An Introduction*, Academic Press Inc., pp. 71-97, 1990.
[4] Hill, R.J., *Electric Railway Traction Part 3 Traction Power Supplies*, Power Engineering Journal, December 1994.
[5] Hill, R.J., *Electric Railway Traction Part 2 Traction Drives with Three-Phase Induction Motors*, Power Engineering Journal, June 1994.
[6] Hayt, W.H. & Kimmerly, J.E., *Engineering Circuit Analysis*, McGraw-Hill, pp. 55-64, 1993.
[7] Tong, C.T., *A.I. Applications and Solution Techniques for AC Railway System Control and Simulation*, MPhil Thesis, Hong Kong Polytechnic, Hong Kong, March 1990.
[8] Zimmerman, R.D. & Gan, D., *MATPOWER User's Manual: A MATLAB Power System Simulation Package*, School of Electrical Engineering, Cornell University, USA, December 1997.