國 立 交 通 大 學

交通運輸研究所

博士論文

No. 068

鐵路列車連鎖延滯之模擬模式構建與應用

Development & Application of A Simulation Model on Railway Knock-on Delay

研 究 生：劉昭榮

指導教授：黃承傳 博士

中 華 民 國 一〇〇年七月

鐵路列車連鎖延滯之模擬模式構建與應用

Development & Application of A Simulation Model on Railway Knock-on Delay

研 究 生：劉昭榮　　Student：Jau-Rong Liu

指導教授：黃承傳博士　　Advisor：Dr. Cherng-Chwan Hwang

國 立 交 通 大 學

交通運輸研究所

博 士 論 文

A Dissertation

Submitted to Institute of Traffic and Transportation
College of Management
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy
in
Management

July 2011
Hsinchu, Taiwan, Republic of China

中 華 民 國 一 ◯ ◯ 年 七 月

鐵路列車連鎖延滯之模擬模式構建與應用

研究生：劉昭榮　　指導教授：黃承傳博士

國立交通大學交通運輸研究所

摘 要

鐵路系統容量不足往往是導致列車延滯之主因，尤其在初始延滯(first delay)發生後，後續所引發的連鎖延滯(knock-on delay)擴散效應對列車正常營運之影響甚大，因此欲確保系統之可靠度及服務品質，快速有效地降低連鎖延滯一直是重要課題。為有效釐清造成連鎖延滯之複雜因素及其交互作用，本研究依據臺鐵系統之運轉特性構建一模擬模式用以推估連鎖延滯，並以臺鐵西部幹線北部路段為案例，分析連鎖延滯之各種相關問題。

本研究經蒐集臺鐵七堵—樹林路段之營運資料驗證模式的合理性後，續針對不同初始延滯程度衍生之連鎖延滯擴散，及不同運轉調度策略(timetable recovery strategy)對連鎖延滯降低進行系統分析，初步發現連鎖延滯有往下游路段擴散，並呈非線性遞增之趨勢，且各種運轉調度策略對連鎖延滯均有明顯降低效果等現象。故本研究進而分別針對列車密度、初始延滯及運轉調度策略對連鎖延滯之影響及其整體交互作用影響進行分析，並以迴歸分析方法構建列車連鎖延滯與該三項因子間之指數關係函數，以作為估算連鎖延滯的簡便工具。另為利了解初始延滯發生區位、持續時間及運轉調度策略對連鎖延滯之影響，本研究亦進行其關聯分析。由研究結果顯示，若初始延滯發生之區位愈靠近上游路段，則連鎖延滯亦將愈大，而運轉調度策略對於降低連鎖延滯之效果也愈佳。

此外，鑑於影響連鎖延滯之部分關鍵因素具有隨機特性，故為確實反映臺鐵列車實際運轉因初始延滯發生後所產生之連鎖延滯特性，及深入探討產生連鎖延滯之班表穩定度問題，本研究更進一步分析站間運轉時間及停站時間之隨機特性，除彙整七堵—新竹路段對號列車及通勤電聯車之站間運轉時間資料，另依尖、離峰時段分開彙整停站時間之隨機資料，並將其納入模擬模式推估連鎖延滯。研究結果顯示，無論就所有車站或二端末車站(七堵及新竹站)而言，其尖峰或離峰時段之連鎖延滯模擬結果皆會接近一定值。本研究所提出之分析架構及內容，除可有效釐清連鎖延滯關鍵影響因素及程度，所構建之模擬模式更可作為相關改善策略之分析工具。研究成果可作為後續營運單位排班規劃、系統可靠度分析及服務品質改善之參考。

關鍵字：鐵路系統、連鎖延滯、初始延滯、運轉調度策略

Development & Application of A Simulation Model on Railway Knock-on Delay

Student：Jau-Rong Liu Advisor：Dr. Cherng-Chwan Hwang

Institute of Transportation Engineering and Management

National Chiao Tung University

Abstract

Train delays of a railway system are affected by many factors and one of the most important factors is insufficient line capacity. Once a first delay occurs, the delay propagation (i.e., knock-on delay) always interrupts train operation. Thus, how to promptly reduce the knock-on delays becomes an important issue for providing reliable timetable and high quality of service. In order to clarify the impacts of these complicated factors and their interactions on knock-on delay, this research develops a comprehensive simulation model to estimate knock-on delays, and a rail section from northern area of Taiwan railway system is selected for case study.

This research first collects real data from the section of Cidu to Shulin of Taiwan railway system to verify and validate the model. The impacts on knock-on delays of different first delays and timetable recovery strategies are then evaluated. The results show that knock-on delay propagates toward downstream sections when a first delay occurs, and the knock-on delay increases nonlinearly toward downstream sections. In addition, timetable recovery strategies are demonstrated to have significant impacts on the reduction of knock-on delays. Based on the simulation results of the case study, regression analyses are employed to calibrate the relationships between knock-on delays and three key factors, which are train density, first delay and timetable recovery strategy. The regression models indicate that the relationship between knock-on delay and these three key factors conforms to an exponential function. This

research also explores the effects on knock-on delays of different first delay locations and recovery strategies. The main findings are as follows: (1) the closer that the first delay occurs at upstream section, the greater the knock-on delays at all stations and two end stations are; (2) the effects of timetable recovery strategies are better in recovering to scheduled timetable when the first delay occurs at upstream section.

To clarify the stochastic nature of the key factors affecting knock-on delays and to evaluate timetable stability, this research also collects the running time and dwell time data of Cidu-Hsinchu section for express and commuter trains respectively for further analysis. The result shows that the knock-on delays at all stations and two end stations during peak and off-peak hours converge to constant values respectively. In summary, this research proposes a framework and a simulation model which can be applied to analyze the impacts on knock-on delay by all kinds of changes in infrastructures, operational situations and controlling strategies. It is expected that the results can be beneficial to timetable scheduling, system reliability analysis and service quality improvement.

Keywords: *Railway system, Knock-on delay, First delay, Timetable recovery strategy*

誌 謝

多年的辛苦終於到了收穫的時刻，回首當年於工作多年後毅然決定重拾書本報考考博士班，如今能順利完成論文取得學位，內心的悸動與感激難以言諭，因為如果沒有許多人的指導協助及一份自我的堅持傻勁，資質平庸的我是無法完成這「不可能的任務」。

要感謝的人實在很多，首先感謝的是指導教授黃承傳恩師的細心指導，經由恩師的專業學術薰陶與啟發，使我的治學態度更為嚴謹、思考問題更為週延，這些收穫必能使我往後之學術研究及待人處事，受益良多。交研所博士班修業期間，感謝能受教於黃台生老師、馮正民老師、汪進財老師、許鉅秉老師、邱裕鈞老師、陳穆臻老師之專業教誨指導，奠定我論文研究之根基；而論文口試暨審查期間，感謝開南大學陳武正教授、南台科技大學李治綱教授、台灣大學賴勇成教授、本校黃台生教授、邱裕鈞教授等各位委員，能撥冗審查並提供卓見與殷切指正，使本論文更臻充實完備，在此由衷表達謝忱敬意。

感謝所辦各位職員的熱心服務及行政協助，使我們求學階段無後顧之憂。另外，交研所各位學長姐、同學及學弟妹們，對於您們的砌磋指導與協助，因無法盡書在此一併感謝。另也要感謝交通部運研所的各位長官與同仁，尤其運計組歷任的國顯與振維組長，及組內幫我分擔業務的各位學長姐與同仁，沒有您們的鼓勵與支持，我將缺乏堅持到最後的動力。此外，要特別感謝中興工程顧問社土木水利中心研究團隊多年來的軌道專業協助，尤其鍾志成博士學長與張恩輔研究員對於論文模式及各項內容之耐心指教與軌道研究經驗傳授，使本論文之立論基礎更為紮實。

最後要感謝我摯愛的家人父母、岳父母、內人及兒女，尤其是內人若慧、小女昱忻及小犬昱甫，由於你們的體諒與關懷，讓我能更專心於研究，願以此論文致上我最深的謝意，並與您們共享這得來不易的小小成就。

劉昭榮 謹誌

2011 年 7 月於交大

目　錄

圖 目 錄

表 目 錄

符號說明

參數符號

$T_{S,X}$	先行列車離站與續行列車進站之號誌安全時距
$T_{S,M}$	先行列車進站與續行列車離站之號誌安全時距
A_t	延滯實際值
F_t	延滯推估值
y	連鎖延滯(秒)
x_1	列車密度(列/小時)
x_2	初始延滯(秒)
x_3	站間趕點虛擬變數(0 或 1)
x_4	站內趕點虛擬變數(0 或 1)
β_n	待校估之參數，$n \in \{0,1,2,3,4\}$


E S A
1896

第一章 緒論

1.1 研究背景與動機

軌道運輸系統的可靠度攸關運輸服務的品質，而其中列車服務的可靠度更是受到社會的期望；旅客對列車準點的重視程度，不亞於對搭乘舒適度、服務頻率或旅行時間等因素的要求，尤其在通勤旅客的眼裡，可靠度更是最被重視的因素。由過去許多關於可靠度及列車延滯之文獻顯示，列車延滯為衡量服務可靠度的重要指標之一，亦是影響可靠度之重要因素，故探討臺鐵系統營運服務品質之相關議題，需有效掌握「延滯」課題。Rietveld et al. (2001)曾提出下列幾種指標以作為評估可靠度的準則，包括：(1)準點率；(2)列車提早離站的機率；(3)實際到達時間與表定到達時間的差異，也就是延滯時間；(4)當一列車延滯時，其續行列車的平均延滯時間；(5)當一列車延滯超過某一時間，對續行列車所造成的平均延滯時間；(6)到達時間的標準差；(7)總旅客延滯小時/旅次數；(8)準點旅次數/旅次數；(9)延滯的列車數等，由上述評估指標亦可顯示「延滯」對軌道系統營運是否正常及服務品質之良窳扮演關鍵重要角色。

列車延滯依其類型可分為「列車排班的交會待避延滯（waiting time due to scheduled meeting and overtaking）」以及「列車實際運行的延滯（delay in current operation）」兩類(交通部運輸研究所，民 94)。前者係列車排班作業過程中，列車因排除衝突所需額外增加的停等時間，亦稱為排班延滯 (scheduled delay) ，通常是用來分析時刻表容量；而後者則是列車實際運行過程中所發生的延滯，亦稱為非排班延滯(unscheduled delay)，一般列車服務可靠度所討論的延滯通常是指後者。若細究延滯發生的原因，則列車實際運行延滯又可分為初始延滯（first delay/primary delay/exogenous delay）和連鎖延滯（secondary delay/ knock-on delay）兩類(Olsson et al., 2004)。初始延滯是由於外在因素直接影響列車運行導致的延滯，而連鎖延滯則是由於初始延滯的發生，造成列車間相互影響所導致的延滯，例如列車因旅客量過多導致停站時間過長，或列車車輛故障及事故所導致之列車停等，皆為初始延滯；而因初始延滯導致其續行列車必須機外停車，便是連鎖延滯。一般有關初始延滯問題，有部分文獻係以安全績效觀點(李治綱等人，民 98)進行分析，但有關連鎖延滯問題，因其可能係容量不足、初始延滯及各項軟硬體設施條件之交互作用等複雜關係所導致之結果，且其影響系統服務品質之程度及營運績效狀況通常最為直接也最嚴重，故若欲探究軌道系統之服務品質改善課題，則應特別針對連鎖延滯之影響因素及其影響程度進行深入分析。

由於列車延滯(尤其連鎖延滯)往往是容量不足所導致，而依據交通部運輸研究所「運輸系統容量分析暨應用研究─軌道系統(1/4)」(民 96)之研究結果顯示，影響軌道容量的因素可歸納成路線條件、交通條件及控制條件三大類如表 1.1 所示。不同的列車運轉條件與列車運轉性能交互影響下，會導致不同的路線容量及運轉時隔，若因路線容量不足而產生路段瓶頸，將會產生延滯甚至發生營運單位最關切之連鎖延滯擴散效應(delay propagation)，尤其在多車種、複線運轉及多類型月臺型式之傳統臺鐵系統，其交互作用之影響更為錯綜複雜。實務上由於營運單位無法有效精準掌握影響連鎖延滯之關鍵因素，皆於列車排班時以運轉寬裕時間(operational buffer time)或於發生營運特殊突發事件時，以趕點時間(recovery time)來降低延滯之衝擊，故欲釐清連鎖延滯之影響因素及其影響程度，除外生之初始延滯外，若能有效釐清前述影響容量之路線、交通及控制條件之交互作用影響，將更能掌握連鎖延滯之影響程度，及有效掌控其對運轉時隔之衝擊，也能確實估算可利用之路線容量，進而提昇列車營運之可靠度及服務品質。

鑑於臺鐵系統運轉特性甚為錯綜複雜且獨特，而其列車連鎖延滯之各項影響條件交互作用之問題特性，亦較適合以模擬方法進行分析，故雖目前已有許多市售商用模擬軟體，但皆無法滿足前述臺鐵系統之連鎖延滯問題需求，故自行構建連鎖延滯之模擬推估模式即成為本研究之最主要研究動機。

1.2 研究範圍與目的

有鑑於連鎖延滯擴散效應對於臺鐵系統正常營運之影響至為關鍵，且連鎖延滯牽涉之影響因素又相當錯綜複雜，而過去之相關研究主要著重在以平均延滯角度分析延滯影響因素，或就特定路段(如車站)小範圍作各項延滯影響程度之解析式分析，尚無就整體軌道系統影響連鎖延滯之各項條件因素進行綜合性分析，即使探討連鎖延滯之研究亦僅就單一車站範圍構建解析模式進行影響因素之交互作用分析，實難一窺連鎖延滯議題之全貌並完全反映實際列車運作及延滯之交互影響現象。故本研究除嘗試有效釐清影響連鎖延滯之關鍵因子、其交互作用及各種情境條件所產生連鎖延滯之影響程度，更希望透過本研究建構連鎖延滯之分析推估模式，以作為後續理論研究與實務應用之參考。

考量模擬模式相較於解析模式及最佳化模式，具有可詳細分析號誌閉塞系統、列車性能及路線條件等複雜因素交互作用之功能，分析結果有較高之精確性，故本研究首先自行開發出一套可分析推估連鎖延滯之模擬模式(運行條件及延滯分析關係如圖 1-1 所示)。本模式除可分析處理影響連鎖延滯各要素條件交互作用關

係之外，並選定臺鐵系統相關路段為案例，進行連鎖延滯之各項關聯分析。

表 1.1 影響路線供給容量的因素

分類	影響因素
路線條件	• 站間軌道數目與運轉方式 • 站內軌道及月臺佈置方式 • 站間距離 • 路線幾何條件 • 銜接點與折返點的配置 • 基地位置及配線 • 路線供電穩定度 • 路權型態
交通條件	• 列車性能 – 列車牽引性能 – 煞車性能 – 最大速度 – 阻力係數 • 列車密度 • 列車的方向分佈與交通組成 • 停站時間與停站型態 • 可用列車數 • 車廂設計 – 車輛的尺寸 – 樓地板面積及高度 – 座站位的比例與安排方式 – 車門大小及配置
控制條件	• 列車運轉調度策略方式 • 閉塞制度的種類 • 辦理閉塞的方式 • 閉塞號誌的配置方式 • 閉塞區間長度 • 路口號誌設計（B 型或 C 型路權）

資料來源：交通部運輸研究所(民 96)。

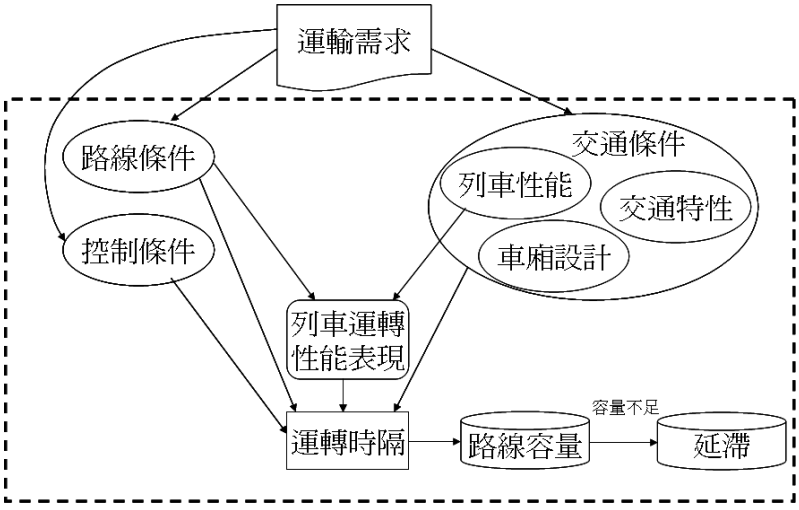


圖 1-1 列車運行條件交互作用及延滯分析關係

有鑑於前述路線、交通及控制條件交互關係甚為複雜，且由於本研究係定位在既定路線條件之營運班表下探討連鎖延滯之影響因素分析，故綜合而言本研究之主要目的有四，茲臚列如下：

一、 發展可綜合分析連鎖延滯各要素條件交互作用相關問題之模擬模式。

二、 應用所建構之模擬模式，進一步探討各種關鍵影響因素對連鎖延滯之影響程度，及與其他路線、交通及控制等條件之交互作用關係，以作為後續營運單位應用之參考。另為建立連鎖延滯與各關鍵影響因素之因果函數關係，故以所構建模擬模式產生之各種延滯情境模擬資料進行迴歸分析，以便利後續直接推估連鎖延滯。

三、 為掌握不同初始延滯之發生區位、持續時間及運轉調度策略(timetable recovery strategies)對連鎖延滯之影響程度，進行三者之關聯分析探討。

四、 為能更真實反映實務狀況，進一步將影響列車運行關鍵參數之隨機特性納入模式考量，並分析各種情境下之連鎖延滯。

1.3 研究內容與方法

如前述相關文獻提及軌道系統列車運轉之延滯概分為正常營運之排班延滯(scheduled delay)及特殊事件之非排班延滯(unscheduled delay)二類，本研究係針對第二類型非排班延滯之連鎖延滯(knock-on delay)進行分析；由於傳統區域鐵路(如臺鐵系統)營運過程中列車彼此間之互動關係非常複雜，要以解析式方法具體分析釐清列車因外生延滯後連鎖延滯擴散效應之各種問題甚為困難，因此本研究選擇採用適合分析列車複雜交互作用之模擬方法，來評估延滯與列車間及與硬體設施之關係。

本研究利用自行建構之模擬模式，主要考量列車流量、車種組成、站間運轉時間以及號誌時距等因素，分析釐清不同列車密度對連鎖延滯之影響程度，及其與路線、交通及控制等條件之交互作用關係；另本研究亦於案例分析部分依各種情境進行初始延滯持續時間、區位及運轉調度策略對連鎖延滯之關聯分析，並依據本模擬模式所產生的大量資料進行迴歸分析，建立適合臺鐵系統連鎖延滯推估之迴歸模式，最後並將影響列車營運關鍵之站間運轉時間(running time)及停站時間(dwell time)的隨機特性充分納入模擬模式進行分析。綜上，本研究之各項研究內容與方法茲臚列如下：

一、文獻回顧與評析

經由國內外相關文獻之回顧，檢視彙整過去關於延滯議題之相關研究，主要針對延滯與班表可靠度、列車異質性相關問題及容量與可靠度的關係，及其對連鎖延滯擴散之影響進行歸納與分析，以作為本研究後續各項分析及模式構建之依據與參考。

二、連鎖延滯之模擬分析

為利釐清各種關鍵影響因素對連鎖延滯之關係及影響程度，及其和路線、交通及控制等條件之複雜交互作用關係，本研究依據列車運行及延滯之各種特性，運用物件類別(Class)設計之方法，構建連鎖延滯模擬模式進行各項問題之系統分析，並以實際臺鐵系統營運資料進行模式驗證，以利本模擬模式作為後續各項連鎖延滯相關問題分析之工具。

三、連鎖延滯影響因素之關聯分析

為有效釐清造成連鎖延滯之複雜因素及其交互作用，本研究依據臺鐵系統之運作特性所開發之模擬模式作為分析工具，以臺鐵之七堵—樹林路段為案例分析關鍵因素對於連鎖延滯之影響。本研究除研析列車密度(train density)、初始延滯及運轉調度策略(timetable recovery strategy)對連鎖延滯之個別影響及其整體交互作用之影響外，更以迴歸分析方法構建列車連鎖延滯與該三項因子間的指數關係函數，以作為估算連鎖延滯的簡便工具。相關研究成果更可作為排班規劃、系統可靠度分析以及服務品質改善之參考。

四、初始延滯區位及運轉調度策略對連鎖延滯之關聯分析

為利瞭解外生初始延滯發生之區位、持續時間及運轉調度策略對連鎖延滯之影響，本研究同樣以臺鐵之七堵—樹林路段為案例進行分析，除可瞭解各項因子之影響程度外，更能掌握運轉調度策略對連鎖延滯恢復之能力，俾利營運單位參考因應。

五、考慮關鍵影響因素隨機特性之連鎖延滯模擬分析

為更確實反映實際臺鐵列車之運轉及產生之連鎖延滯隨機特性，本研究蒐集臺鐵系統實際營運資料，分別彙整對號列車及通勤電聯車於尖、離峰之站間運轉時間(running time)及停站時間(dwell time)分佈資料，以便將其隨機特性充分納入模擬模式進行連鎖延滯分析。

六、結論與建議

綜合前述各項議題分析結果，歸納出有關連鎖延滯之相關問題結論，並針對該些問題之後續解決方式及應用提出具體建議，供後續研究及營運單位參考。

1.4 研究流程與架構

根據上述研究目的及研究內容，本研究之流程及架構如圖 1-2 所示，茲摘述說明如下。首先鑑於鐵路列車運轉延滯之相關問題類型眾多且複雜，故本研究首應界定臺鐵系統列車運行連鎖延滯之研究標的，並藉由所蒐集分析之文獻中彙整相關研究成果及分析方法，釐清目前相關研究尚未解決之課題及未能深入研析之內容，俾利本研究目的、範圍及研究內容之界定。

在研究動機及研究標的確立後，將透過實務臺鐵系統之營運資料，建構連鎖延滯推估之模擬模式。由於連鎖延滯相關問題特性之不同，故本研究建構之模擬模式主要分為確定性(deterministic)及隨機性(stochastic)二階段。其中有關列車連鎖延滯與其影響因素(包括外生初始延滯發生地點、持續時間及不同運轉調度策略等)之關聯分析部分，因較屬因果關係分析，故適合採用前者確定性模擬模式進行評估。至於站間運轉時間及停站時間二關鍵因子對連鎖延滯之影響分析部分，因涉及班表穩定度問題且為利反映實務狀況，故較適合採用後者隨機性模擬模式進行評估，相關分析流程茲分述如下：

一、確定性連鎖延滯推估之模擬模式建構分析

本研究所建構之連鎖延滯模擬模式經驗證可行後，將先作為分析列車密度、初始延滯及運轉調度策略等關鍵因素對連鎖延滯之個別及整體交互作用影響分析之工具，並於掌握各關鍵影響因素對連鎖延滯之影響程度後，再進一步深入分析外生初始延滯發生地點、持續時間及縮短站間運轉時間與停站時間等不同運轉調度策略對連鎖延滯之影響程度，以提供營運單位營運調度之因應參考。

二、隨機性連鎖延滯推估之模擬模式建構分析

為確實反映部分參數資料於實際營運之隨機特性，本研究另針對列車延滯影響最鉅之站間運轉時間及停站時間進行隨機特性分析，再將該些隨機資料納入模擬模式推估連鎖延滯。由於臺鐵系統之列車種類甚多特性各異，故本研究將其概分為對號列車及通勤電聯車(EMU)二類，蒐集分析路段之延滯資料，並進行二類列車之各別站間運轉時間及停站時間的隨機分佈分析。其中由於各站之停站時間資料分佈變異頗大，故本研究除先將資料蒐集時段內過年及地震等異常易造成列車

延滯之事件日樣本去除外，為利分析特性一致更進一步將該資料針對尖、離峰進行分類，再進行後續之連鎖延滯推估。

經由連鎖延滯之模擬模式建構，本研究藉由所蒐集之案例資料，除驗證模式之適用性，另針對連鎖延滯關鍵影響變數之關聯性及相關重要課題進行深入分析，並探討重要參數之隨機特性對於連鎖延滯推估之影響。最後綜整前述各項研究內容及成果，提出鐵路列車運行連鎖延滯相關議題之結論與建議。綜合前述研究流程，本論文的章節架構如下：

第一章為緒論。首先說明本研究的研究背景與動機，並界定列車連鎖延滯問題，進而依序說明研究範圍與目的、研究內容與方法、研究流程與架構。

第二章為文獻回顧與評析。本章將就本研究與鐵路列車連鎖延滯相關之國內外重要文獻，加以回顧與綜整分析。內容主要包括鐵路列車運轉延滯問題、班表可靠度相關問題與延滯之關係、列車異質性相關問題與延滯之關係、容量與可靠度的關係、研究方法論應用等面向，廣泛蒐集與鐵路列車運轉及延滯之相關文獻進行彙整分析。

第三章為臺鐵列車連鎖延滯模擬模式之構建及分析。本章將考量鐵路列車運行之路線、交通及控制等條件之複雜交互作用關係及延滯之各種特性，蒐集臺鐵系統各項軟、硬體及模式所需之各項參數資料，構建適合臺鐵連鎖延滯模擬分析之模式，並進行初步之連鎖延滯模擬分析，以作為後續各項連鎖延滯相關問題分析之工具。

第四章為連鎖延滯影響因素之關聯分析。本章將以所建立之連鎖延滯模擬模式為分析工具，進一步選定列車密度、初始延滯及運轉調度策略等關鍵影響因素，以臺鐵之七堵—樹林路段為案例蒐集相關臺鐵資料，分析各因素對連鎖延滯之個別及整體交互作用之影響，以初步掌握各因素之影響程度，並嘗試建立其函數關係，俾利作為推估連鎖延滯的簡便工具。

第五章為初始延滯發生區位及運轉調度策略對連鎖延滯之關聯分析。本章繼續以先前建立之連鎖延滯模擬模式及七堵—樹林路段案例資料為基礎，進一步分析營運單位可能關切之外生初始延滯發生區位、持續時間及運轉調度策略對連鎖延滯之影響程度，以便掌握運轉調度策略對連鎖延滯恢復之能力及營運參考因應。

第六章為關鍵影響因素隨機特性之模擬分析。為更確實反映實際臺鐵列車之運轉及產生之連鎖延滯特性，充分考量站間運轉時間及停站時間之隨機特性，本

章除詳實彙整對號列車及通勤電聯車之站間運轉時間，更依尖、離峰時段分開彙整停站時間之隨機資料，並將其納入模擬模式進行連鎖延滯分析。

第七章為結論與建議，綜合前述各章之分析結果，歸納出有關連鎖延滯之相關問題結論，並提出後續可深入研究及參考施行之建議事項。

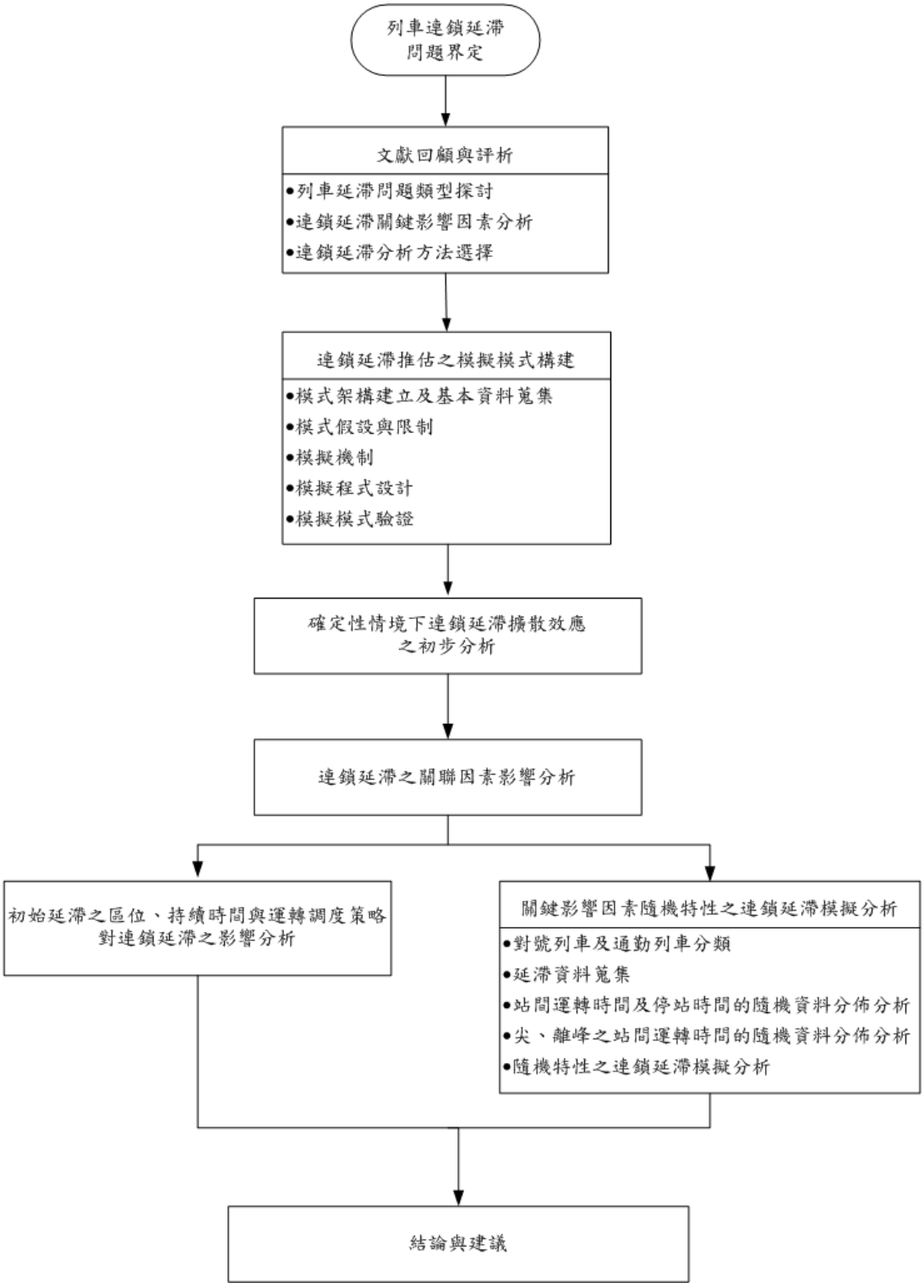


圖 1-2 研究流程圖

第二章 文獻回顧與評析

依照交通部運輸研究所之「軌道容量研究－臺鐵系統容量模式之建構分析（一）」(民 94)研究報告，列車延滯依其類型可分為「列車排班的交會待避延滯」及「列車實際運行的延滯」兩類，而 Olsson et al. (2004)亦將延滯發生的原因，歸納為初始延滯和連鎖延滯兩類。故探討軌道列車運行之延滯問題，基本上可將軌道運輸系統的運作本質視為一個等候系統，其抵達率為單位時間之「列車流量」，亦即列車密度(列車數/單位時間)，而軌道設施的服務率則為其「軌道容量」，當列車流量趨近於軌道容量時，延滯亦會趨近於無窮大。因此列車流量與平均延滯時間關係的曲線，於接近軌道容量時會有大幅陡升的現象，而延滯愈大、可靠度亦愈低，故延滯與可靠度二者存在抵換(trade-off)的關係。

為探究臺鐵系統延滯(尤其連鎖延滯)之各項複雜問題，並尋求連鎖延滯之可行改善策略，需先有效釐清延滯之類型及其影響因素。因此本研究首先就與列車延滯關係密切之班表可靠度、列車運行條件、路線容量等要項之相關文獻進行回顧，接續再就列車延滯及服務可靠度之分析方法，與運轉調度策略對列車延滯改善分析之相關文獻進行探討，以完整掌握連鎖延滯分析之理論架構及分析方法之邏輯，以作為本研究後續各項研析內容之參考基礎。茲將上述各類相關文獻之回顧結果彙整分述如下：

2.1 軌道列車延滯發生原因之分析

有關軌道列車延滯發生原因之分析主要包括延滯程度與延滯原因兩階段，前者所關注的議題在於延滯值的大小，同時也包含諸如平均值、變異數等敘述性統計值；後者則近一步分析延滯的原因，用以作為延滯改善策略研擬之參考(交通部運輸研究所，民 100)。延滯原因的分析，大多以軌道營運單位或監理單位所訂定之延滯原因類別為基礎，惟因延滯之原因錯綜複雜致常無法有效釐清，故為力求週延起見仍應將延滯原因歸納成初始延滯與連鎖延滯再進行分析。交通部運輸研究所(民 100)曾嘗試將臺鐵系統所記錄的延滯原因依據前述方式予以分類，再歸納彙整如表 2.1 以作為延滯分析之基礎，惟不可避免的是部分延滯原因可能同時歸屬於初始延滯與連鎖延滯，例如旅客過多導致上下車時間過長屬初始延滯，但亦可能因先前的初始延滯導致車站累積過多旅客需要上下車，使得連鎖延滯更加擴散而屬於連鎖延滯。

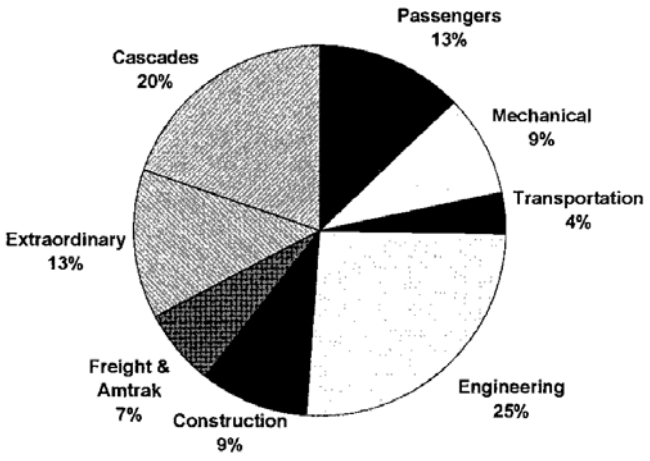
表 2.1 臺鐵初始延滯與連鎖延滯之原因分類表

分類	延滯原因
初始延滯	天候環境、行慢、慢行、車輛故障、ATP、號誌、事故、行包
連鎖延滯	交會、待避、路塞
兩者皆可能	旅客、調車、編組

資料來源：交通部運輸研究所(民 100)

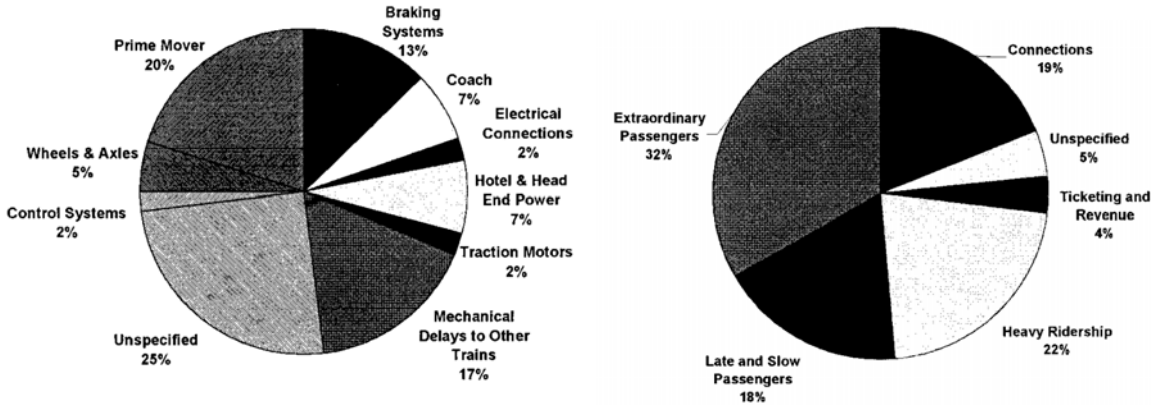
Nelson and O'Neil (2000)亦曾針對美國通勤鐵路的服務可靠度、延滯程度及原因進行探討分析，並將該鐵路延滯之原因分為工程、機械、運輸、施工、旅客、特殊狀況、連鎖延滯及貨運列車等 8 大類，該研究蒐集 40 多萬筆車次資訊進行統計，並以各延滯原因的持續分鐘數比例繪製其分佈圓餅圖如圖 2-1 (交通部運輸研究所，民 100)。另其亦就部分延滯項目之原因進行子項目分析，以機械類與旅客相關的延滯原因為例則可細分如圖 2-2。

Jong et al. (2010)則以臺灣高鐵系統為例，進一步套用風險矩陣的概念進行延滯分析，依不同延滯原因分別計算各種延滯發生之頻率與嚴重度。該研究並以國內「鐵路行車規則」第一百二十二條律定的 17 項事故（如表 2.2），配合臺灣高鐵自營運起 39 個月的延滯資料進行分析，得到列車延滯的風險矩陣如圖 2-3。整體而言，由該圖可發現位於此矩陣越右上方即代表其延滯原因之發生頻率與嚴重度均高，需要儘快改善(如列車遲延、號誌機故障)；位於越左下方的延滯原因則較不嚴重急迫(如列車或車輛衝撞、電車線故障)，而由圖 2-3 計算亦可得到如圖 2-4 的列車延滯風險排序圖。



資料來源：交通部運輸研究所(民 100)

圖 2-1 美國通勤鐵路各種延滯原因之統計圖



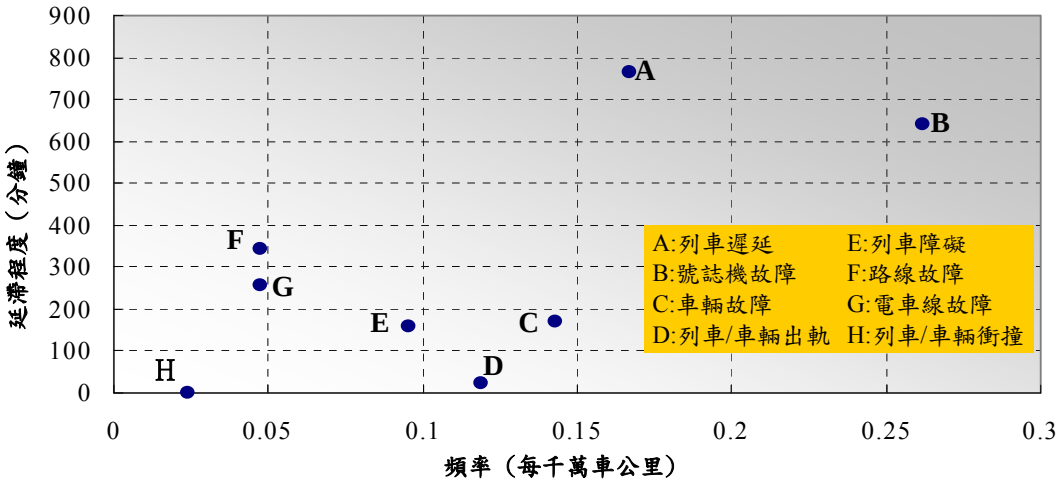
資料來源：交通部運輸研究所(民 100)

圖 2-2 美國通勤鐵路機械及旅客延滯的細部原因統計圖

表 2.2 鐵路行車規則第一百二十二條律定之行車事故項目

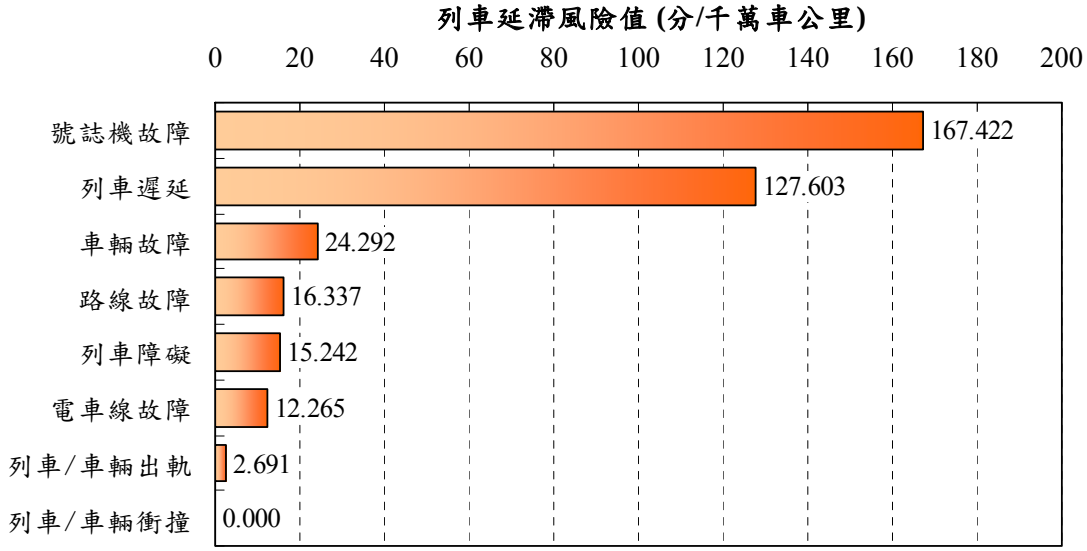
編號	行車事故項目名稱	編號	行車事故項目名稱
1	列車或車輛衝撞	10	車輛故障
2	列車或車輛傾覆	11	路線故障
3	列車或車輛火災	12	電車線故障
4	列車或車輛出軌	13	號誌機故障
5	列車分離	14	列車障礙
6	列車進入錯線	15	號誌機外停車
7	車輛溜逸	16	列車遲延
8	止衝擋衝撞	17	人員死傷
9	閉塞錯誤	—	—

資料來源：Jong et al. (2010)



資料來源：Jong et al. (2010)

圖 2-3 臺灣高鐵延滯風險矩陣圖



資料來源：Jong et al. (2010)

圖 2-4 臺灣高鐵延滯風險排序圖

2.2 班表可靠度與延滯關係及分析方法

由於軌道列車運行延滯與可靠度之間存在抵換關係，故在理論研究與實務應用上，常藉由班表可靠度之探討來衡量延滯之衝擊影響，而準點率則常用來評估列車服務可靠度之良窳，但因影響延滯與可靠度之因素錯綜複雜，故準點率並非唯一指標，適用評估指標之選定常需視分析問題特性及研究方法類型而定。

Middelkoop and Bouwman (2002)曾應用 SIMONE 模擬軟體來比較兩個小型路

網及時刻表之穩定度(stability)，其中對可靠度的評估是採用延滯的列車數、恢復時間，以及初始與最終延滯之比率等三個項目來作為評比的標準，並進行荷蘭 Weesp 車站轉乘等待時間的案例分析，發現增加 6 列貨物列車排入班表後，會造成延滯的程度增加 10%，顯示列車流量的提高會導致可靠度下降。惟其主要內容係強調將 SIMONE 商業模擬軟體，應用於荷蘭鐵路實際案例以進行時刻表穩定度之實驗測試，並無探討如何處理類似臺鐵多車種、複線運轉及多類型月臺型式之鐵路系統，及其路線、交通及控制條件交互作用產生複雜延滯擴散影響等問題。而 Briggs and Beck (2007)則是發展更複雜的指標：先統計在某一車站的列車實際運行資料，找出一個最能夠代表其延滯時間和發生率關係的 q-exponential 函數（q-exponential functions），再透過該函數中的各項係數來評估該車站在可靠度上的表現。此方法的優點為可分別評估各車站的可靠度，其結果更能反映旅客的感受，但缺點為其過程相當繁雜。Higgins et al. (1995)則考量了每個軌道路段、列車和班表，分別針對車站（含端點站）、軌道相關以及列車相依等三種型態的延滯建構解析模式，可用於改善班表的可靠度，之後又分別以貨運和客運為對象，以解析方法預估其可能延滯時間(Higgins et al., 1998)。

Carey (1999)為評估複線列車運行之可靠度(reliability)及準點率(punctuality)，以解析式方法構建連鎖延滯之機率密度函數計算公式，其係為給定班表之列車班距及外生初始延滯機率密度函數之函數關係，並透過該函數計算出連鎖延滯之擴散延滯量；另因為外生初始延滯之機率密度函數不易獲得，其亦應用一些啟發式求解方法進行問題之求解。而 Carey and Carville (2000)則以模擬模式分析單線區間之交會待避或多月臺車站特定區位之延滯擾動暨班表可靠度。

有關班表的可靠度分析，除了解析方法亦有若干研究採用模擬分析。Barter (1998)即以 RailPlan 模擬模式之基本概念，分析討論容量與列車服務之間的關係以及可靠度規劃（planning for reliability）問題。另在 EuROPE-TRIP 計畫(2000)中則是利用 VISION 模式，探討時刻表每多排入不同列車數對整體路線延滯之影響情形，發現排入的列車數與整體的延滯時間兩者之間存在著抵換（trade-off）關係。另由於列車在軌道上運行的原理與等候理論（queuing theory）相當類似，因此以該理論為基礎，可對列車延滯行為與現象進行探討，Hansen (2000)就利用了等候理論搭配 Max-Plus 代數法（max-plus algebra），來推估車站容量與列車服務的可靠度。

至於有關列車延滯及列車服務可靠度的研究方法，大致可分為解析方法

（Analytical Methods）、微觀模擬模式（Micro-Simulation Models）和統計分析（Statistical Analysis）三類(Briggs and Beck, 2007)，以下分別說明各研究方法之主要相關文獻。

一、解析方法

解析方法通常需要數學式之推理運算，其優點在於不若模擬模式需要大量的參數輸入，故無需精確班表作為輸入資訊，所需之運算時間也相對較少。而等候理論（Queuing Theory）和 Max-Plus 代數法（Max-Plus Algebra）等理論經常被用來分析列車可靠度，如前述 Hansen (2000)即結合兩種理論來推估列車服務的可靠度。

在 Ferreira and Higgins(1996)的研究中，其所發展的解析模式係考量單線區間每一列車延遲的風險值（Risk of Delay, RD），進而討論不同程度延遲對減少延遲風險所需時間的影響，並發現當列車班次較擁擠時，降低延遲風險所需的時間較多，可靠度也較低。而 Goverde(2005)亦是以 Max-Plus 代數法為基礎發展模式，當軌道系統中單一列車發生延滯時，透過模式可分析系統恢復常態運轉所需的時間，以此來判定班表受到延滯影響的敏感度和穩定度，作為設計可靠班表的輔助工具。但 Goverde 發展之模式主要是基於規格化時刻表（或稱週期時刻表、定型化時刻表），邱戊吉(民 99)則針對這項限制予以放寬，並以臺鐵基隆－新竹進行案例分析，在時空圖上針對每一個車次到開標記其允許延滯的多寡，允許延滯的值越低代表該處越不可靠或越不穩定。

由於在規劃階段通常無法清楚掌握軌道系統的許多細節，因此比較適合使用解析方法作為輔助決策的分析工具。解析模式通常允許某些簡化問題的假設，以減少數學的複雜度，但這也導致計算出的結果精確性與可靠度較差(交通部運輸研究所，民 100)。

二、微觀模擬模式

微觀模擬可詳細分析不同列車與其他列車以及設施互動之複雜過程，分析結果有較高的精確性，但需仰賴嚴謹的模式構建技術與電腦的運算。早年電腦性能較差，求解時間較長，僅能處理簡單或小規模的問題，如 Chen and Harker(1990)僅對單線區間，列車於路線上的交會待避行為建構模擬模式來分析列車延滯。周學怡(民 86)亦是以模擬模式分析單線區間的可靠度議題，該模式考量站間運轉時間、停站時間均不確定的情況下評估時刻表的可靠度。

近年來開始有較為複雜的模擬模式，例如 Carey and Carville(2000)針對多月臺車站，考量列車性能與車站月臺軌道配置建構模擬模式，基於在一可行班表下，外加延滯或擾動進行模擬以測試班表的可靠度。我國交通部運輸研究所(民 98)亦曾針對單一區段，考量列車流量、車種組成情況、站間運轉時間以及號誌時距等多項因素，建構列車服務可靠度模擬模式，探討列車流量與列車平均延滯時間的關係，並透過案例分析提出運轉寬裕係數建議值，以及了解站間運轉時間寬裕對可靠度的影響。

目前已有許多模擬模式已發展成商業軟體，且實務上也有不少應用，如前述荷蘭的 Railned 與 Incontrol Enterprise Dynamics 公司合作開發之 SIMONE，係專門針對荷蘭鐵路系統而設計，具備有直接存取荷蘭鐵路資料庫之介面，可直接擷取資料庫中路網、軌道、設施、流量需求與時刻表等資訊，評估時刻表可靠度以及分析延滯之產生與影響，並量化不同軌道設施配置之延滯。Middelkoop and Bouwman(2000、2002)曾使用該軟體探討荷蘭 Weesp 車站轉乘等待時間，以及比較兩個小型路網及時刻表之穩定度。

在國內 Hwang and Liu (2010)、黃範哲(民 90)以及謝興盛(民 92)，分別針對臺鐵、高鐵以及捷運建構模擬模式，藉由設定不同的初始延滯，比較其對列車服務可靠度的影響。模擬模式通常係依研究個案之特性而開發，因此與個別系統的依存度較高，且另一方面它需要更多軌道系統的細節資料，但應用於作業層次的問題通常有較好的效果。

三、統計分析

推論統計分析是根據實際營運資料，利用統計方法校估延滯的機率分佈或各種延滯影響因素與延滯值的迴歸式。周學怡(民 86)的研究雖為模擬模式，但模式中有關站間運轉時間的機率分配仍是以統計方法校估，該研究考慮臺鐵列車的車種與停靠型態差異很大，故並非直接以列車實際到開時間進行校估參數，而是以表定時間與實際時間之差異進行分析。經該研究的測試，同一車種的運轉時間差距與站間距離呈正相關。黃承傳與劉昭榮(民 100)為有效釐清造成連鎖延滯之複雜因素及其交互作用，及研析列車密度、初始延滯及運轉調度策略對連鎖延滯之個別影響及其整體交互作用之影響，亦以迴歸分析方法構建列車連鎖延滯與該 3 項因子之間的指數關係函數以作為估算連鎖延滯的簡便工具。

過去交通部運輸研究所(民 98)亦曾以臺鐵列車實際到開時間資料，透過迴歸分析方法建立列車服務可靠度迴歸模式，但列車在實際營運時含有許多隨機特

性，影響列車可靠度的因素眾多，且由於樣本資料的品質難以掌握，如資料正確性欠佳、數量不多、資訊太少無法針對延滯原因進行樣本篩選等諸多原因，以致於迴歸分析結果的正確程度不如預期。

統計方法隨著資料量愈多，可使模式的可信度愈高，但當未來系統改變，統計模式的解釋能力則會大幅下降。Murray and Grubesic (2007)即提出結合模擬模式和統計分析兩方法的可能性，模擬模式能產生與真實世界相近之結果，透過模擬模式所產生的大量資料可以作為建立統計分析模式的基礎。

2.3 列車運行條件與延滯關係

由於列車之運行狀況與軌道系統中之路線、交通及控制條件密切關聯，而延滯亦是該三大條件交互作用下之產物，故實應就各大條件中較關鍵之延滯影響因素深入探析，俾利掌握其因果關係。茲就關鍵之列車異質性、月臺指派及運轉整理(rescheduling)等方面之文獻彙整分析如下：

一、列車異質性與延滯問題探討

Vromans et al. (2006)分析列車服務頻率與列車平均延滯之關係，並釐清列車服務頻率與延滯及列車異質性間之關係，更以最短班距倒數和(the sum of shortest headway reciprocals, SSHR)及抵達班距倒數和(the sum of arrival headway reciprocals, SAHR)等二項指標來評估列車之異質性，證實提升列車營運可靠度最佳方法就是降低因列車異質性所導致之連鎖延滯效應。

Huisman and Boucherie (2001)藉由所蒐集之延滯資料構建一隨機模式來描述排班及非排班列車之移動情形，並藉由求解軌道系統之線性不等式得到每列車之行駛時間(running time)分佈，研究亦發現影響行駛時間之關鍵因素包括列車數、列車異質性、初始延滯、列車行駛順序及運轉寬裕時間等。

二、列車運行之車站及月臺指派問題

Chakroborty and Vikram (2008)構建一線性混合整數規劃模式，求解一多月臺繁忙車站之列車到達型態最佳化問題，該研究假設列車實際到站時間會在原訂班表時間之前 1 小時左右得知，模式主要處理列車運行不如預期排班導致月臺及股道壅塞之列車運行延滯，考慮要素包括：(1)列車運行延滯；(2)月臺與列車適合度配置(以印度車站為例)；(3)列車到站之最後停站月臺重新指派。

Carey and Crawford (2007)針對一個具大量不同列車速度組合、多月臺車站及

複雜路線之鐵路路網，研析實務上考量列車運行延滯、避免列車運行衝突之最佳化排班問題。為解決初始班表之列車運行衝突問題，該研究發展一啟發式求解方法，並透過模擬方法獲得無運行衝突之最佳班表，提供探討實際可運行班表、營運策略、車站配置及隨機延滯等問題之參考。

三、運轉整理與連鎖延滯擴散問題

運轉整理(rescheduling)之主要目的係在列車實際運轉過程中確認及解決列車衝突，並透過運轉整理使列車營運儘速回復至最初班表，Hansen 與 Pachl 於 Railway Timetable & Traffic (2008)一書中即提到運轉整理之功能包括：使列車運轉重回無列車衝突之班表、提供列車重新運轉之資訊、偵測列車衝突、自動解決列車衝突、產生無衝突重新指派之新班表及提供列車運轉交通控制所需之資料等。另該書亦指出解決列車衝突重新排班之方法包括：使用替代路線、延長停站時間、延長站間運轉時間、重新安排不停靠車站、運轉需要之增停車站及取消部分列車等，而此亦即是本研究運轉調度策略(timetable recovery strategy)所探討之範疇。另 Chang and Chung (2005)曾建構一列車運轉模式，分析列車管制彈性及列車運轉整理問題對列車班表制訂之影響，並以基因演算法(genetic algorithm)進行較有效率之求解。Corman et al. (2010)亦曾以禁忌搜尋演算法(tabu search algorithm)，應用於荷蘭鐵路路網列車衝突偵測及延滯排除之運轉整理案例求解，並與既有之即時交通管理系統 ROMA (railway traffic optimization by means of alternative graphs)比較，其結果不僅得到運轉整理之最佳解，且求解效率更佳。

四、運轉寬裕時間、股道佔用與連鎖延滯擴散問題

Nie and Hansen (2005)以系統分析方法分析列車運轉與鐵路網路中車站股道佔用之關聯問題，其係以所蒐集之路網車站、班表及列車到開偵測資料，與推估之閉塞區間時間(blocking time)、運轉寬裕時間(buffer time)及股道佔用(track occupancies)等各項資料進行比較及統計分析，並以荷蘭鐵路車站進行實證研究，結果顯示當車站平面交叉路段及進出站之股道被佔用超過 80%時，其列車運行速度及容量將顯著下降，月臺股道亦因列車運行交叉行為而使其旅客上下車時間延長。故若能有效蒐集列車行駛時間(running time)及營運延滯資料，進而精確推估閉塞區間時間及運轉寬裕時間，將可有效提升大型轉運車站之表訂列車到、開時間之可靠度，而路線股道交叉點及橫渡線之運轉寬裕時間，也正可反映實際列車運轉速度及閉塞區間時間之機率分配關係。

Yuan and Hansen (2007)有鑑於鐵路路網容量之有效使用與列車運行可靠度及

準點率之提升間具有 trade-off 關係，為有效解決車站月臺之股道路線運行衝突及轉乘旅客上下車所導致之延滯問題，遂建構一車站連鎖延滯之隨機解析模式。該研究考慮複雜的號誌系統、列車運行優先順序保持及股道佔用時間擾動等問題。由於該模式業已應用荷蘭鐵路實際資料加以驗證，故後續亦被應用於處理車站容量利用最佳化問題，另藉由該模式亦可分析通過平面交叉股道路線且具最大運行連鎖延滯擴散之最大列車班次數。該研究亦發現隨著表訂運轉寬裕時間(scheduled buffer time)的減少或列車密度的增加，其通過列車之平均列車運行連鎖延滯擴散將呈指數型增加，如圖 2-5 所示。

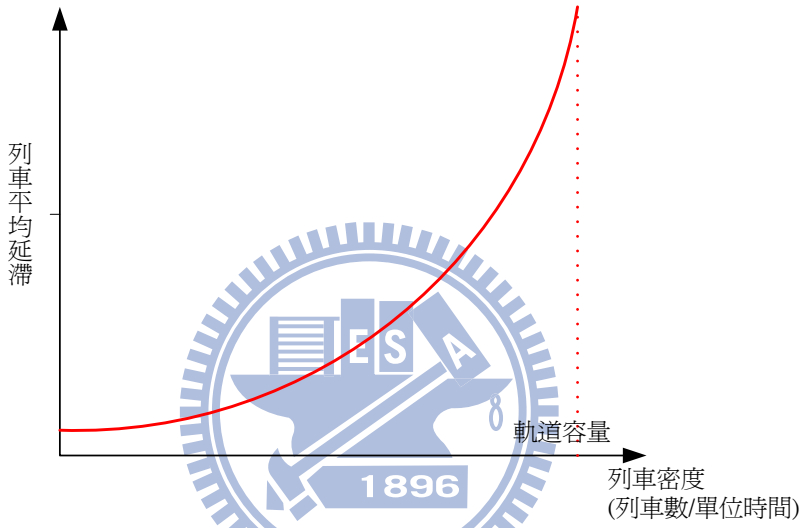


圖 2-5 列車延滯與列車密度關係示意圖

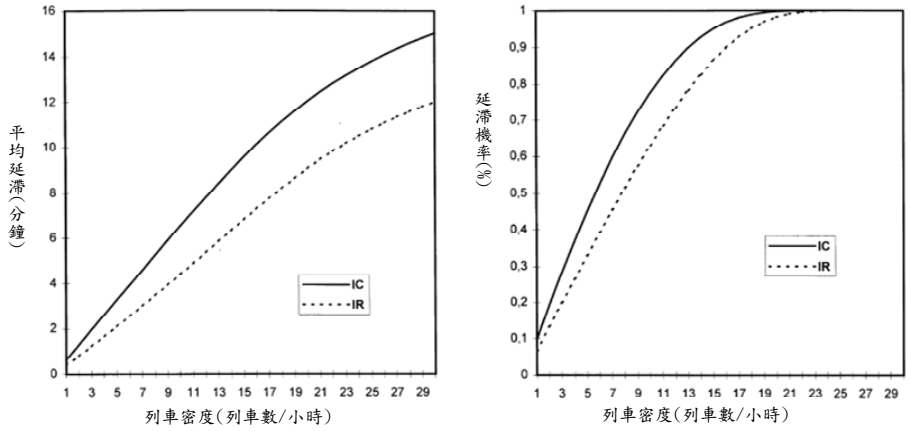
2.4 容量、可靠度與延滯之交互關係

容量、可靠度與延滯之交互關係嚴重影響列車是否正常運行，在 Olsson et al. (2004)對挪威鐵路系統所作的一系列研究中，曾透過統計方法以相關係數（pearson correlation）來確認影響準點率(可靠度)的因素，其中包含：旅客數、容量利用率、班次取消率、臨時速限、工程施工、平日/週末、到離站準點情形及運轉調度方式等。該研究結果的容量利用率和準點率的相關性雖然沒有預期中高，但就等候理論的邏輯以及過去許多相關研究結果，在在說明其關聯性：列車流量愈高、路線容量利用率愈高，則可靠度會愈低。若某一班列車發生延滯，其續行列車有可能受到影響而跟著延滯，這是所謂的延滯的連鎖效應（knock-on effects）。當列車流量愈高時，表示列車之間的距離愈近，因此延滯的連鎖效應影響愈大，使得整體的可靠度降低。

Huisman and Boucherie (2001)針對複線區段且有不同車種組成的情境，發展了

計算運轉時間的統計模式，此模式可應用於分析每小時列車數對平均延滯和延滯發生機率的影響，經以荷蘭阿默斯福特（Amersfoort）到茲沃勒（Zwolle）路段為例，其研究結果如圖 2-6。該圖顯示當每小時列車數增加時，除了平均延滯時間會跟著增加外，延滯發生的機率也會同樣提升。

Mattsson (2004)指出高列車流量代表著高路線容量利用率，而在實際營運上，容量利用率愈低表示有愈多未使用的容量可作為運轉調度的緩衝，以減少列車延滯造成的衝擊，當有列車發生延滯時，便能迅速恢復回常態運轉，因此可靠度較高。該文中也提到，Wahlborg 利用了 RailSys 模擬模式，研究瑞典韋斯特羅斯（Västerås）到斯德哥爾摩（Stockholm）路段，在不同的交通量下，從延滯發生到恢復成常態運轉所需的時間，結果如表 2.3。由該表可看出當有延滯發生時，容量利用率愈高則受影響的列車數愈多，且恢復運轉所需的時間也愈長，意味著系統的整體可靠度愈低。而賴勇成等人(民 99)則利用路線容量估算的概念計算班表回復時間，其係定義行車系統若發生事故而造成路線中斷時，該中斷時間內無法發車或不能通過之列車數，在事故修復後利用剩餘容量（時刻表容量與已使用容量之差值）消化延滯之列車使回復正常營運班表，其所需的消化時間則稱為班表恢復時間 T^R，如圖 2-7 所示。



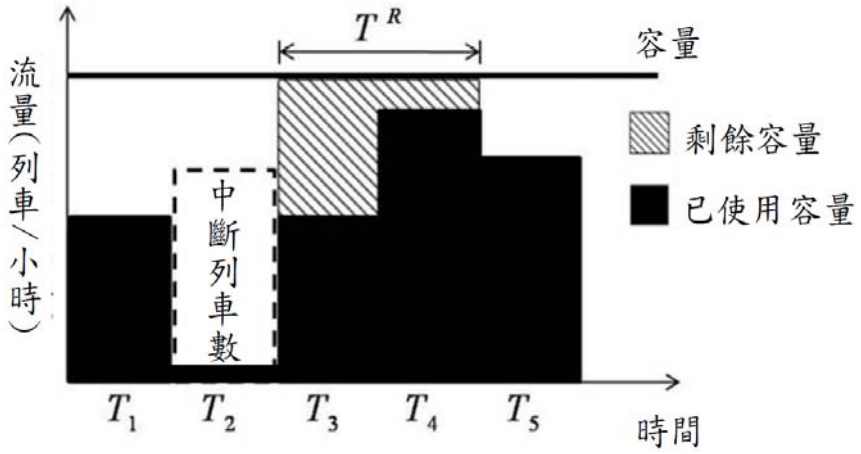
註：IC 表示城際鐵路(Intercity)；IR 表示區域鐵路(Interregional)。
資料來源：Huisman and Boucherie (2001)

圖 2-6 每小時列車數對平均延滯和延滯發生機率的影響

表 2.3 容量利用率和恢復運轉所需時間之關係

列車數	31	35	37
容量利用率（%）	62	72	76
當發生 15 分鐘延滯時，受到影響的列車數	4.8	7.0	8.0
當發生 15 分鐘延滯時，恢復運轉所需的時間（min）	41.4	47.9	51.3

資料來源：Mattsson (2004)



資料來源：賴勇成等人(民 99)

圖 2-7 班表回復時間示意圖

2.5 運轉調度策略對列車延滯改善相關分析

鐵路系統常因設備故障或意外事故等外生事件，導致列車產生初始延滯甚至連鎖延滯之擴散現象，為了維持運轉之可靠度及班表之穩定度，營運調度人員於延滯發生時，常依實務經驗採取各種運轉調度策略來降低延滯對正常營運之衝擊，而合適之運轉調度策略亦常是能否有效降低延滯(尤其連鎖延滯擴散)，而使班表維持正常穩定運轉之關鍵，故其對行車調度極為重要。

楊立安(民 96)為分析如何有效減少臺鐵系統之列車誤點時間或延滯擴散，藉由行車調度之專家訪談與文獻回顧得知調度因素與調度方案，並綜整國內、外有關運轉整理（Train Rescheduling）相關文獻(Cheng, Y., 1996、Cheng, Y., 1998、Missikoff, M., 1998、張恩輔，民 91、李治綱等人，民 91)之研究成果，以及運轉整理調度因素與調度方案相關研究(陳英相與張仁城，民 86、Fay, A., 2001、Rebreyend, P., 2005），利用模糊多準則決策得知調度因素以「列車等級」與「目前列車延滯的時間」為主，調度方案以「變更運轉順序」為主，「特開列車」為最後考量的調度方案；利用派翠網路（Petri Nets）具備平行、分散、及不確定性處理的能力，以 IF-THEN 的形式規則建構列車調度流程與知識規則，建構出列車運轉整理調度的模式與七種不同事故的類別，最後藉由矩陣方程式（Matrix Equation）與可達樹（Reachability Tree）驗證列車運轉整理（Train Rescheduling）調度模式的可行性。

陳朝輝(民 97)為分析各種運轉調度策略對捷運系統之績效，綜整分析國內外有關鐵路運轉整理、系統模擬及物件導向模式技術等相關研究文獻(黃哲旭，民 85、Parkinson, T., 1996、Brannlund et al., 1998、 O’ Dell & Wilson, 1999、Chang et

al., 2000、簡聰裕，民 89、Zhu and Schnieder, 2000、Won et al., 2001、Chen et al., 2003)，應用物件導向模式技術模擬分析臺北捷運系統之運轉整理績效。其係以統一塑模語言(UML)作為物件分析(OOA)與設計(OOD)工具，來描述系統內車輛、時刻表、車站、旅客等元素間的相互關係，並以物件程式語言(OOP)C++撰寫模式。該研究所探討之運轉整理策略包括：月臺趕點、站間趕點、抓前車、抓後車、刪除列車、與雙人乘務等策略。系統績效指標包括旅客觀點之平均旅行時間，以及行車調度人員觀點之列車平均延滯時間，並於開發的模式中加入最佳化模組，可以對模式關鍵影響因子進行最佳化搜尋。另針對臺北捷運淡水-新店線之單一來回運行路線進行模擬分析，以直接搜尋法之 Hooke-Jeeves 演算法，搜尋延滯事件發生後執行間距控制之最佳抓車策略，並測試上午尖峰於不同時間、地點、發生不同長度的初始延滯事件，評估實施各種運轉整理策略的運行績效。

其個案研究之模擬實驗結果發現：(1)延滯事件發生後，使用運轉整理策略確實可以改善系統績效，使用單一策略以站間趕點效果較佳。(2)初始延滯時間 5 分鐘，單純實施月臺趕點與站間趕點策略，即可追回旅客因延滯所增加的旅行時間。(3)延滯事件發生後，抓前車搭配趕點策略較抓後車效果為佳。(4)當初始延滯事件發生於行車方向後端車站，且延滯時間長度達 10 分鐘以上，以實施雙人乘務策略為最佳方案。(5)使用刪除列車策略，列車總延滯時間雖可縮減，但旅客平均總旅行時間則大幅增加。具體研究結論為使用混合策略較單一策略效果為佳，從旅客績效指標觀點，抓前車是有效的運轉整理策略，由營運者觀點，則刪除列車是有效策略。

2.6 綜合評析

根據本章所蒐集整理之列車延滯發生原因分析、班表可靠度、列車運行條件、路線容量、列車延滯及服務可靠度分析方法與運轉調度策略對列車延滯改善分析等議題之相關文獻可知，鐵路列車延滯相關問題之分析十分複雜，除需界定所分析之延滯類型及其對正常營運之關鍵影響(如本研究之連鎖延滯)之外，亦需掌握影響延滯之路線、交通及控制等三大條件關聯之可靠度、路線容量及運轉調度策略等要項之關係及影響程度。茲將回顧之重要發現彙整如下：

一、鐵路列車運轉延滯之複雜性

鐵路列車運轉之延滯，除不可控制之外生初始延滯外，其衍生之連鎖延滯擴散更是營運者不易有效控制之問題，且因延滯所涉及之影響因素包括路線、交通及控制等諸多條件，且各項因素彼此間之交互作用相當複雜，不易掌握分析，故

相關文獻之分析做法皆將研究範圍限縮至較小路段甚至車站區位。欲就整個路段進行連鎖延滯分析確有一定之難度，尤其對於具有多車種、多列車運轉特性之臺鐵系統難度更高。黃承傳、劉昭榮(民 100) 以臺鐵之七堵─樹林路段為案例構建模擬模式分析關鍵因素對於連鎖延滯之影響，除研析列車密度(train density)、初始延滯及運轉調度策略(timetable recovery strategy)對連鎖延滯之個別及其整體交互作用之影響外，更以迴歸分析方法構建列車連鎖延滯與該三項因子之間的指數關係函數以作為估算連鎖延滯的簡便工具，即希望針對鐵路列車運轉延滯複雜問題的研究能夠有所突破。

二、營運及延滯資料蒐集處理之困難度

臺鐵之營運資料本身即摻雜延滯(包括初始延滯及連鎖延滯)成分在內，故欲有效釐清所蒐集之運轉資料以作為模擬模式建構及後續案例分析，難度甚高，再加上許多關鍵參數(例如本研究之站間運轉時間及停站時間)運轉資料具有隨機特性，故有關所蒐集資料之前處理工作(如扣除有重大事件之營運資料)相當重要。由於相關資料內部之交互作用及複雜性甚高，故所蒐集資料之關鍵參數及變數判斷，及挑選之前處理工作將影響後續模式構建及連鎖延滯推估分析結果之成效。

三、延滯分析方法之限制

如本章文獻彙整所示，列車延滯及列車服務可靠度的研究方法主要包括解析方法、微觀模擬模式和統計分析三大類，惟各類方法論皆有其適用之問題特性及優缺點，但由於鐵路列車運轉延滯具有高度之複雜性，營運及延滯資料蒐集處理亦具有一定之困難度，故過去相關文獻研究若採解析方法和統計分析皆需將研究之問題範圍適度限縮始能處理，相較之下對於鐵路列車延滯複雜交互作用問題之處理，仍以微觀模擬模式有較好之成效，但有關其大量參數資料輸入處理、各項複雜運轉機制規則之設定等問題，皆需設法予以克服。

第三章 臺鐵列車連鎖延滯模擬模式之構建及分析

前已提及軌道系統列車運轉之延滯概分為正常營運之排班延滯(scheduled delay)及特殊事件之非排班延滯(unscheduled delay)二類，本研究係針對第二類型非排班延滯之連鎖延滯(knock-on delay)進行分析；由於傳統區域鐵路(如臺鐵系統)營運過程中列車彼此間之互動關係非常複雜，要以解析式方法具體分析釐清列車受外生延滯後之連鎖延滯擴散效應甚為困難，因此本研究選擇採用適合分析列車複雜交互作用之模擬方法，來分析評估連鎖延滯與列車間及與硬體設施之關係。

本章將考量鐵路列車運行之路線、交通及控制等條件之複雜交互作用關係及延滯之各種特性，蒐集臺鐵系統各項軟、硬體基礎資料及模式所需之各項參數資料，構建適合臺鐵連鎖延滯模擬分析之模擬模式，以作為後續各項連鎖延滯相關問題分析之基礎工具，並先進行初步之連鎖延滯模擬分析。

軌道系統之延滯主要係為路線容量不足所導致，而路線容量又受路線條件、交通條件及控制條件所影響(如圖 1-1 所示)，故延滯亦將受上述三大類條件之影響。而因列車運行延滯在軌道系統營運階段係扮演列車班距(train headway)及列車流量(traffic flow)之關鍵決定要素，故本研究所建構之模擬模式主要功能在於連鎖延滯之推估。其係先輸入龐雜之路線、交通及控制條件等基本條件資料，並考量各種實際運轉條件之假設限制，及列車衝突解決、列車延滯排除與運轉調度策略之模擬機制設定，再以物件類別設計之模擬模式流程處理所有條件之交互作用，最終推估出連鎖延滯結果，其估算結果可作為後續營運可靠度分析(train reliability)及排班調度之參考。

3.1 模式架構

本模式之架構如圖 3-1 所示，模式系統架構主要由基本輸入資料、模式假設與限制、模擬機制、模擬程式設計及模擬模式驗證等部分所組成，其中主要輸入資料包括基本條件資料、計畫班表、初始延滯，輸出資料則為模擬之實際班表，並據以計算連鎖延滯。各部分內容茲分述如下：

一、基本條件資料：包括軌道容量相關的路線條件、交通條件與控制條件如表 3.1。

二、計畫班表：包括各車次於每個車站的預定到/開時間、各車次之起迄點、列車順序及停站型態，故本模式可處理多種列車、不同列車等級、不同站間運轉時間、不同起迄站、不同停靠車站、不同停靠時間之情境。

三、初始延滯：本研究旨在處理連鎖延滯之影響，故模擬模式必須設定一項初始延滯的相關輸入資訊，初始延滯可設定在軌道路線之任一位置，延滯參數之設定如表 3.2 所示。

四、模擬之實際班表：模擬後之班表為模式之最後產出，其呈現列車受延滯影響後之實際到開時間，而模擬班表與計畫班表中各列車到開時間之差距即為連鎖延滯。

五、延滯分析：本研究之連鎖延滯分析，即計算案例路段中各車站之通過列車模擬離站時間與計畫班表之時間差。

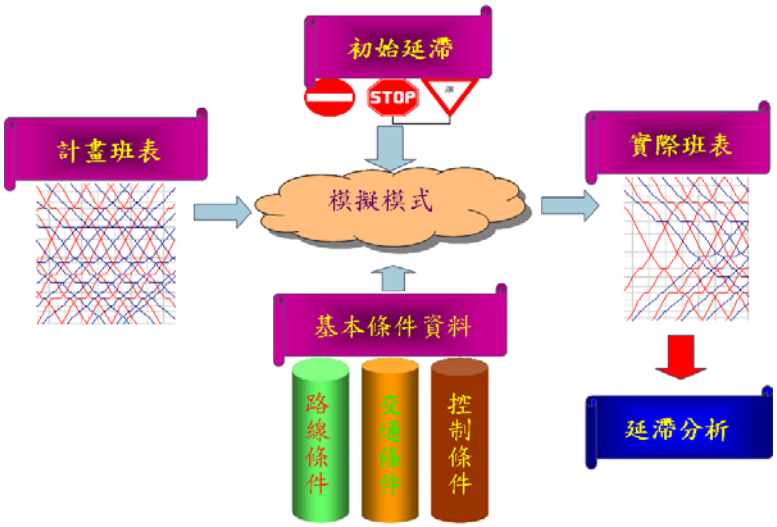


圖 3-1 模式架構圖

表 3.1 基本輸入資料表

條件 / 項目	路線條件	交通條件	控制條件
輸入參數內容	-路線 -車站列表 -車站軌道佈設	-計畫班表各項列車資訊 -最短停站時分 -最短站間運轉時間	-同向時隔 -反向時隔 -站間容量 -趕點能力

表 3.2 初始延滯之設定參數表

延滯參數	說明
延滯地點	若為站間軌道：需指定方向 若為站內軌道：需進一步指定股道
起始時間	該軌道資源開始無法被使用
解除時間	該軌道資源可再被使用之時間
延滯量	將「解除時間」減去「起始時間」即為延滯量(或連鎖延滯)

3.2 模式假設與限制

本模式之假設與操作限制如下。

一、車站間均有兩股道同時以複線方式運轉。

二、臺鐵系統列車於北上、南下之站間股道係獨立運轉，但當產生延滯時，於站內之共用股道則可依營運需要彈性調度不同方向之列車順序，故雙向之列車運轉將會產生交互作用。

三、臺鐵區域鐵路系統之車站配置型態大致可歸納為 5 種型式，如圖 3-2~圖 3-6。

四、由於臺鐵系統號誌設置之限制，各站間依不同條件皆有允許駐留列車數(亦即閉塞區間)之上限限制(為利模式操作本研究假設為 2 列車)。

五、運轉調度之站間趕點時間以站間運轉時間固定的縮減比率設定之（本研究假設為 0.9），至於站內趕點即依據各站實際允許之最小停站時間設定之。

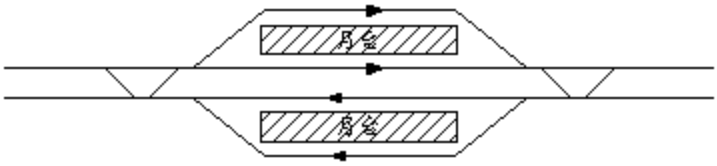


圖 3-2 型 I 車站軌道佈設

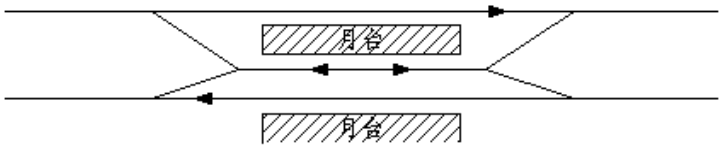


圖 3-3 型 II 車站軌道佈設

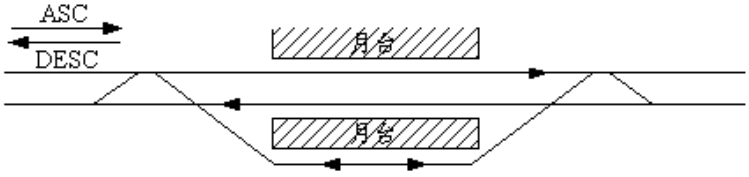


圖 3-4 型 III_R 車站軌道佈設

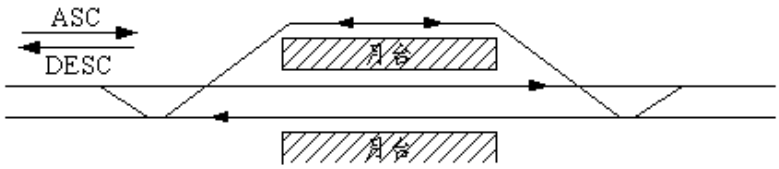


圖 3-5 型 III_L 車站軌道佈設

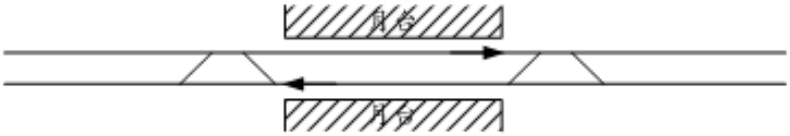


圖 3-6 型 VI 車站軌道佈設

3.3 模擬機制

為準確推估連鎖延滯，本模擬模式必須考慮所有列車運行之影響因素，相關因素之處理方式彙整如下：

一、列車衝突解決：站間軌道之列車運行規則為先進先出，且不允許站間追越；另站間軌道有容量限制，不允許過多的列車滯留同一站間；站內每一股道只允許一列車佔用，而先行列車離開車站後，續行列車進入該股道必須符合離—到時隔限制，如圖 3-7 所示；二連續列車抵達同一車站之不同股道，必須符合到—到時隔限制，如圖 3-8 所示；至於二連續列車由同一車站之不同股道離開，則必須符合離—離時隔限制，如圖 3-9 所示；另假如二列車由不同方向通過平面交叉點，則必須符合平面交叉時隔限制，如圖 3-10 所示。

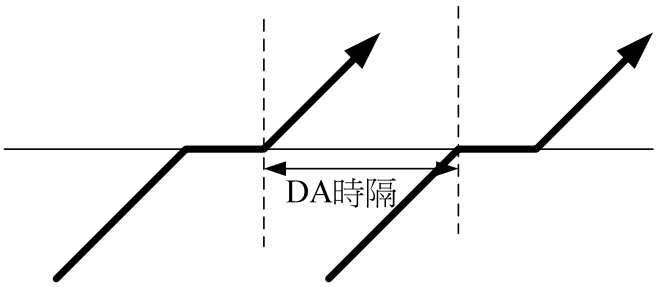
二、列車延滯排除：當模式執行時，若續行列車違反前述之運行規則而無法前進時，即產生延滯及可能之連鎖延滯現象；當模式偵測到此現象時，均統一將該事件的執行時間（系統時間）延後 1 秒執行，此方法可以使列車的運行無限的往後遞延直到前方的事故（延滯）原因排除為止。

三、運轉調度策略：鐵路營運單位在列車延滯時，通常會啟動運轉調整機制，而本模式所考慮的「運轉調度策略」允許使用單一或多種策略同時搭配使用，包括：

(一)站內趕點：縮短停站時間，但是必須遵守停站時間最短限制，而且需依公開時刻表不可提早開車。

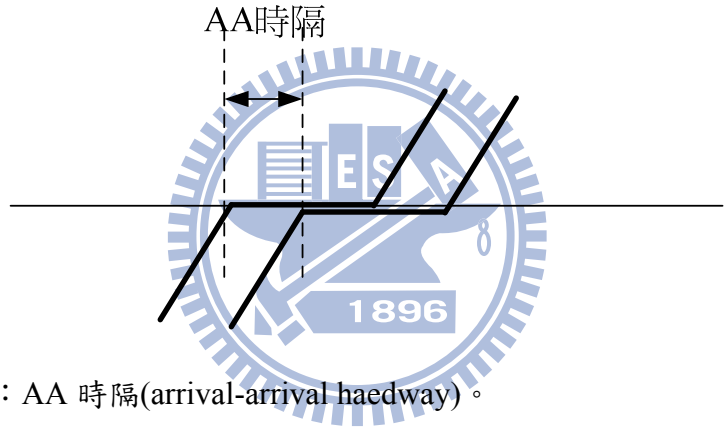
(二)站間趕點：縮短站間運轉時間，但仍需遵守趕點限制。

(三)同時採取站內趕點與站間趕點。



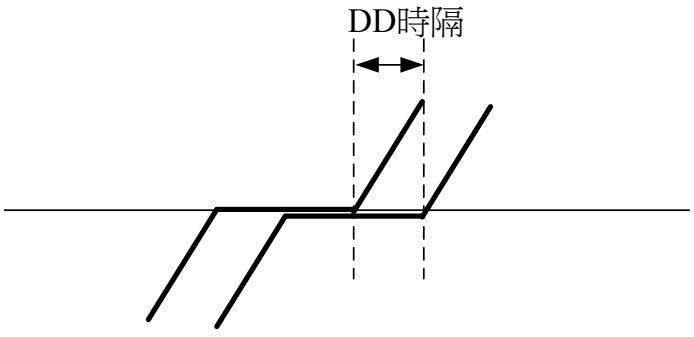
註：DA 時隔(departure-arrival headway)。

圖 3-7 同向列車離—到時隔示意圖



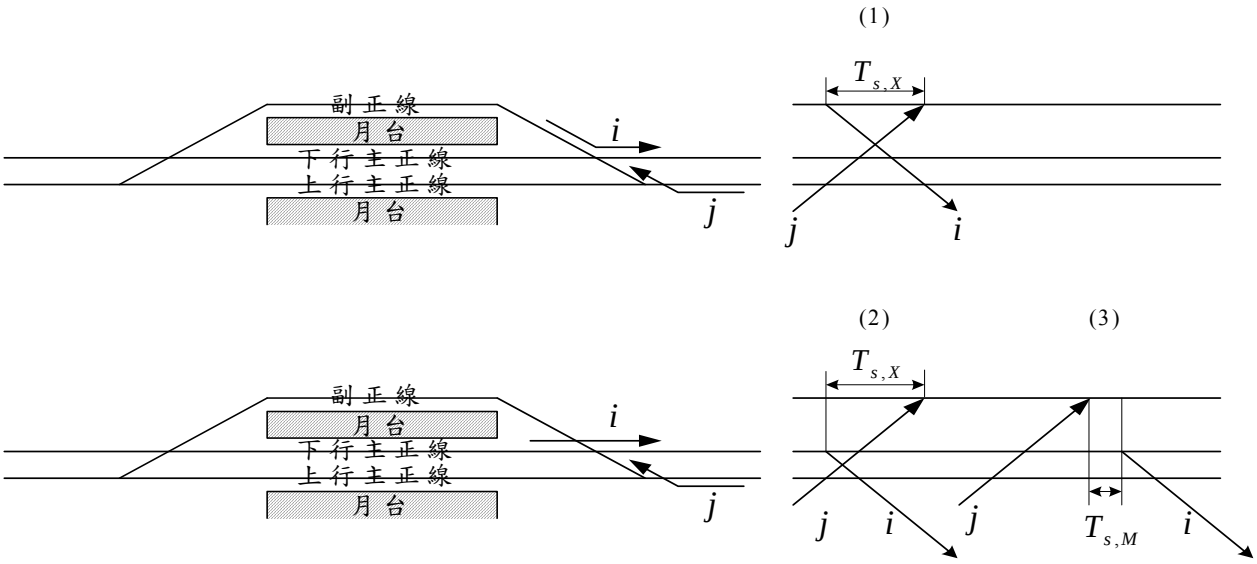
註：AA 時隔(arrival-arrival haedway)。

圖 3-8 同向列車到—到時隔示意圖



註：DD 時隔(departure-departure headway)。

圖 3-9 同向列車離—離時隔示意圖



註：$T_{S,X}$ 為先行列車離站與續行列車進站之號誌安全時距；

$T_{S,M}$ 為先行列車進站與續行列車離站之號誌安全時距。

資料來源：交通部運輸研究所(民 97)

圖 3-10 型 III 月臺之平面交叉時隔示意圖

3.4 模擬程式設計

本研究模擬程式之設計與開發，旨在設計一套資料結構來描述整個軌道行車系統之各種狀態，包含列車、車站、軌道等，同時針對列車運轉邏輯設計對應的狀態改變，透過不斷更新電腦記憶體中的變數來達到系統模擬之目的，最後將本研究關注的變數（即列車延滯）匯出，以利後續的績效評析。以下將分別就物件類別、行車系統模擬流程、初始延滯等三項設計進行深入說明。

一、物件類別(Class)設計

類別設計為物件導向程式設計中最重要之基礎(周斯畏，民 91)，一般而言包括：(1)邊界類別(boundary class)、(2)實體類別(entity class)及(3)控制類別(control classes)三大類，其中第一類是用來處理使用者介面，第二類則是用來處理問題領域的核心資料，最後一類則是用來處理前兩類資料的協調與互動；由於(1)與(3)類與核心模式較無關聯，故以下僅就與模式相關之實體類別部分進行說明。本研究共設計了 9 個類別來描述整個軌道系統，包括 RailSystem、Train、DwellPlan、Station、StationI、StationII、StationIII_R、StationIII_L、Tracks 等(如圖 3-11)。其中 RailSystem 是最上層的類別，除一些 Global 的變數之外，所有其他的類別物件

均包含於其中；而 Station 類別則是其他四個 Station 開頭類別的父類別，父類別用來描述最基本的車站（即圖 3-6 的捷運化車站），另外四個子類別則分別描述其他四型車站如圖 3-2~圖 3-5。Tracks 類別是被 Station 類別所包含，而 Train 與 DwellPlan 則是用來描述「計畫班表」與「實際班表」。另其中 Train 與 Station 都以集合物件的形式和 RailSystem 以合成（aggregation/composition）關係存在(周斯畏，民 91)，其他諸如 Train 與 DwellPlan、Station 與 Tracks 均為合成關係。至於 StationI、StationII、StationIII_R、StationIII_L 等四個類別則繼承（inheritance）自 Station 類別，用以實作各種不同型式車站的行車限制。

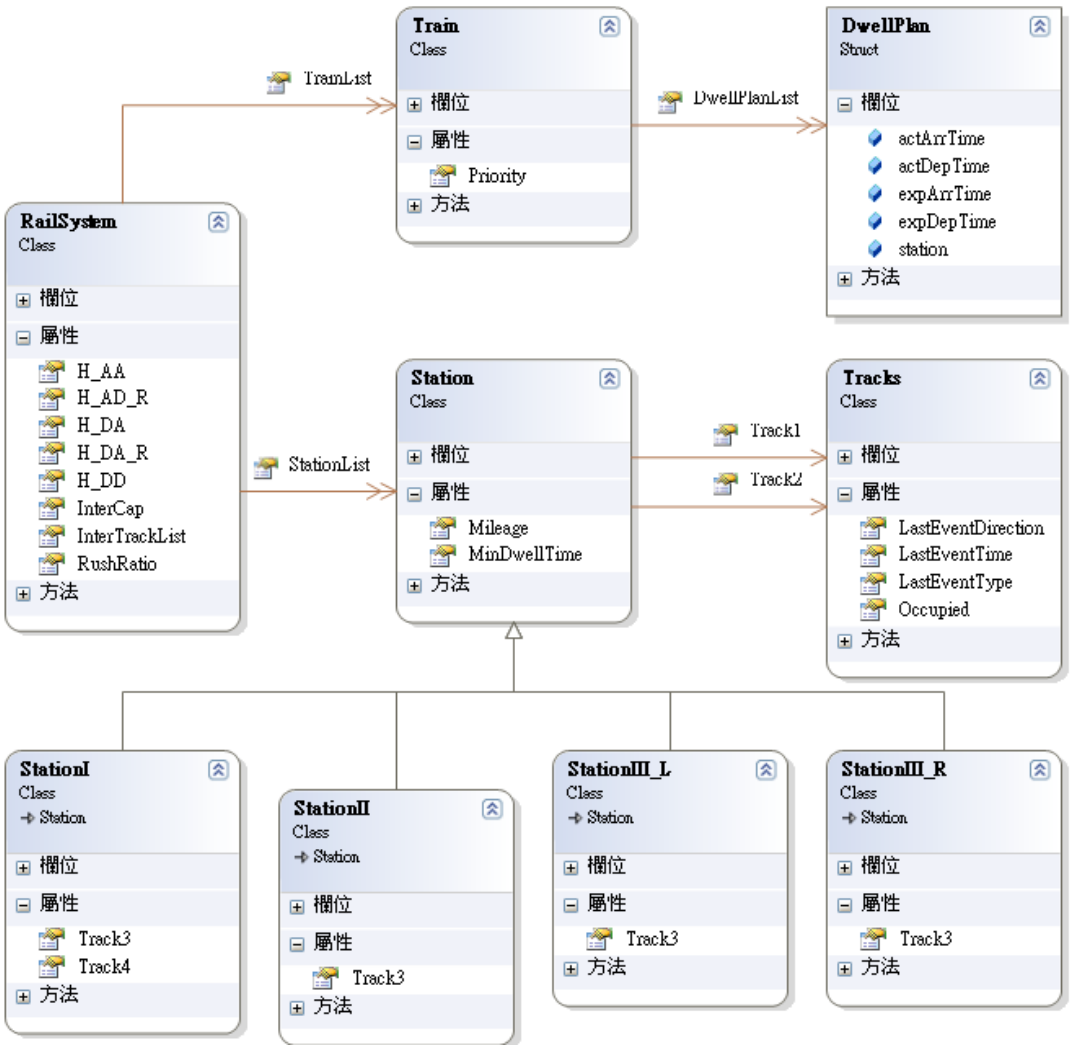


圖 3-11 類別設計架構圖

另本研究依圖 3-11 執行進一步的虛擬碼撰寫時發現，StationII、StationIII_R、StationIII_L 三個類別有許多屬性與行為是類似，包括站內有三股軌道、第三股軌道共用、共用軌道的使用牽涉反向時隔（平面交叉）等屬性。故為使程式容易維

護，本研究進一步將平面交叉的行為獨立處理，建立 CrossOverInfo 類別處理各種平面交叉，如此將可以簡化圖 3-11 的類別圖，亦即只要在車站物件建構（construction）時加入指定的平面交叉路徑設定即可。經整理後得到 StationII、StationIII_R、StationIII_L 三種車站的平面交叉路徑設定分別如圖 3-12、圖 3-13、圖 3-14 所示，在程式撰寫時即可根據前述三種路徑設定將平面交叉類型寫入程式，以處理避免列車衝突且需足夠間距(headway)之問題，如此即能達到程式碼共用與容易維護的目的。

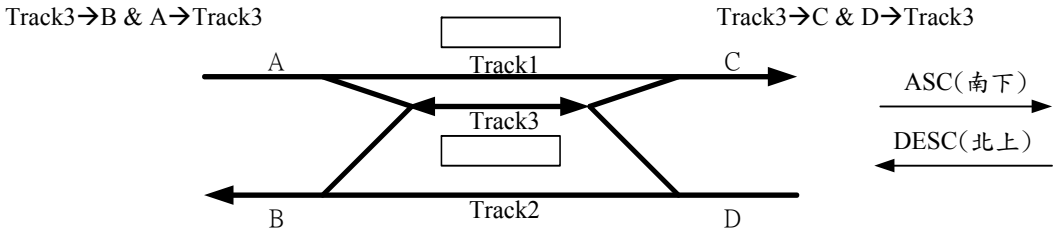


圖 3-12 StationII 型車站股道之平面交叉路徑設定

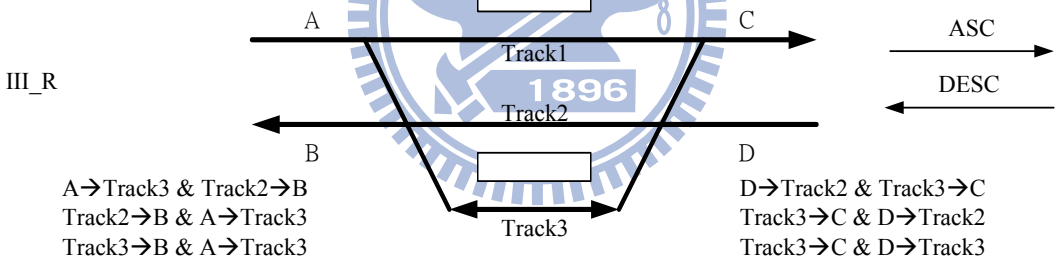


圖 3-13 StationIII_R 型車站股道之平面交叉路徑設定

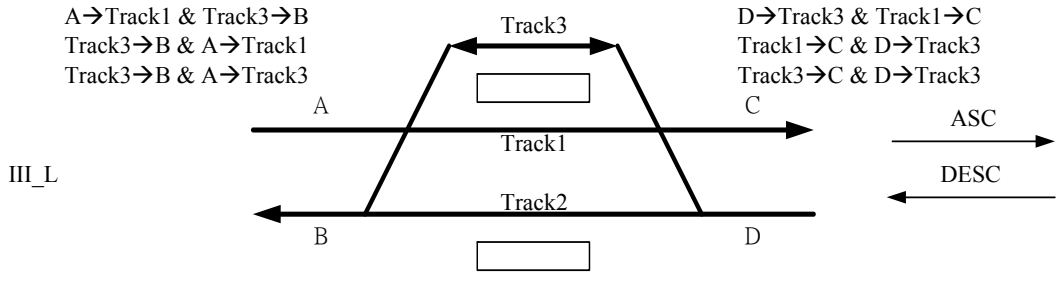


圖 3-14 StationIII_L 型車站股道之平面交叉路徑設定

二、行車系統模擬流程

本模擬模式估算連鎖延滯之整體流程如圖 3-15 所示，主要是以事件導向的方式進行，一開始將所有列車的第一個事件加入事件列表，經時間排序後每次依序執行第一個事件，共計有進站、通過與離站三種事件；若有列車衝突則都在延滯處理時把時間加計一秒後重新加回事件列表，當所有的事件均處理完成後，事件列表沒有任何事件，即完成模擬。至於衝突項目的檢查方式則視事件種類有所差異如表 3.3 所示。

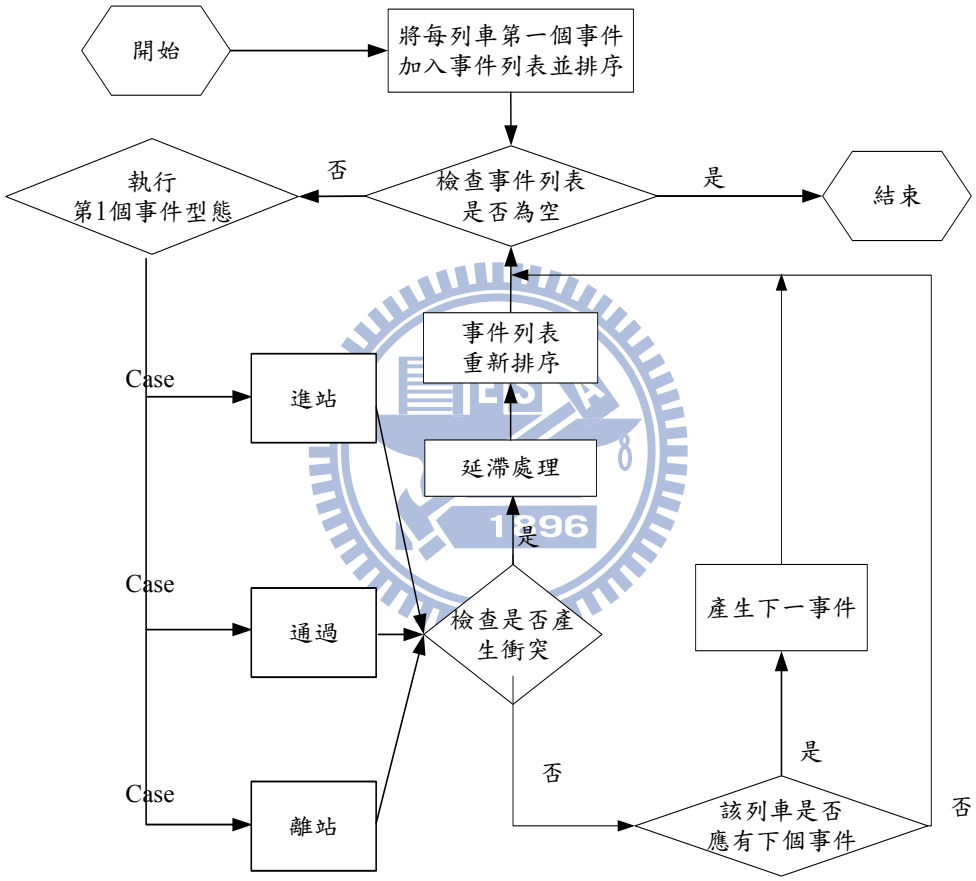


圖 3-15 連鎖延滯之模擬流程圖

表 3.3 各種不同事件的衝突檢查項目

事件種類	進站	通過	離站
檢查項目清單	是否存在站間追越 站內股道數是否足夠 進站時隔是否充足 反向時隔是否充足	是否存在站間追越 站內股道數是否足夠 進站時隔是否充足 反向時隔是否充足 離站時隔是否充足 股道容量是否充足	離站時隔是否充足 反向時隔是否充足 股道容量是否充足

至於本模式之程式語言選擇，因臺鐵之連鎖延滯推估問題錯綜複雜，經評估仍宜採模擬模式進行，但目前並無合適的商用軟體可供使用，故自行撰寫程式。考量類別庫之完整性及未來的擴充支援，本研究採用微軟.NET Framework 類別庫；原則上只要符合 CLI （Common Language Infrastructure）標準程式語言均可呼叫.NET Framework 類別庫，且目前支援最完整也是主流的.NET 語言即是 C#，故本研究以 C#撰寫之，同時 IDE（Integrated Development Environment）整合開發平臺則選用 Visual Studio 2005 Professional。

三、初始延滯運作

由於初始延滯可能於不同時空發生且程度不一，而本模擬模式係著重處理列車的基本運轉以及連鎖延滯擴散效應，程式未具有模擬「初始延滯」之功能，故於推估連鎖延滯時須從事件驅動引擎中新增兩種事件，包括列車運轉之障礙開始與障礙結束（如圖 3-15 模擬架構）。第一種事件安排於初始延滯發生時執行，觸動時將會鎖定使用者所設定之軌道，使其無法被列車佔用；第二種事件則在第一種事件發生後的若干時間（使用者指定之延滯程度）執行，觸動時將會解除鎖定，透過上述方式將可在任意時空產生不同程度的初始延滯。

3.5 模擬模式驗證

為測試模擬模式之適用性，本模式選定臺鐵系統交通量最繁忙之七堵—樹林路段為案例，擷取 2009 年 7 月 29 日(週三)一般日之中央列車控制 CTC (centralized train control) 資料庫，選定上午 10:10 通過萬華—臺北路段之 42 車次，其 18.2 分鐘之初始延滯所產生實際列車到—開之延滯資料進行驗證分析，結果顯示計有臺北、松山、南港、汐止及七堵等 5 個站共 40 部列車受該初始延滯影響，相關驗證分析討論茲述如下：

一、模式驗證指標選定

檢視上述 40 筆列車樣本，其實際延滯值不僅頗分散(平均值為 282.6 秒、標準差為 428.8 秒)且延滯值跳動、無如預期向下游擴散之現象，經研判該結果可能係實務上調度員遇事故時，已依經驗判斷採行不同列車調度策略所致，且調度策略因時因地因車皆不同；另該延滯結果亦可能是整體軌道系統、前後路段及各項複雜外生事件因素交互作用之結果，故所蒐集之實際延滯資料屬性不易釐清歸納其趨勢傾向，亦不適於進行各項統計檢定分析。另由於其中 13 筆樣本之實際延滯值為 0，若以式(1)一般慣用之平均絕對百分比誤差(mean absolute percentage error,

MAPE)或式(2)之均方根誤差(root mean square error, RMSE)作為推估模式精確度之驗證指標，將不適用且會產生極大偏誤之現象，故在臺鐵實際延滯資料具高度複雜性不易釐清之前提下，鑑於臺鐵實務上若列車延滯不超過 5 分鐘則不視為誤點之認定標準，本模式之驗證指標將以受影響列車之模式推估延滯與該列車實際延滯之差異絕對值是否超過 5 分鐘，作為模擬模式之精確度評估指標。

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right| \tag{1}$$

$$RMSE = \sqrt{\frac{\sum_{t=1}^{n}(A_t - F_t)^2}{n}} \tag{2}$$

A_t：延滯實際值； F_t：延滯推估值

二、模式驗證結果分析

應用上述模擬模式之精確度評估指標，經彙整本案例 40 筆受影響列車樣本之模式推估延滯與實際延滯資料，其差異分佈及其差異絕對值之統計如圖 3-16 及表 3.4 所示，結果顯示有 30%之模擬延滯時間與實際延滯時間幾近相同，有 90%之列車樣本模擬延滯時間與實際資料差距在 5 分鐘之內，故整體而言顯示本模擬模式估算結果之準確度可被接受。

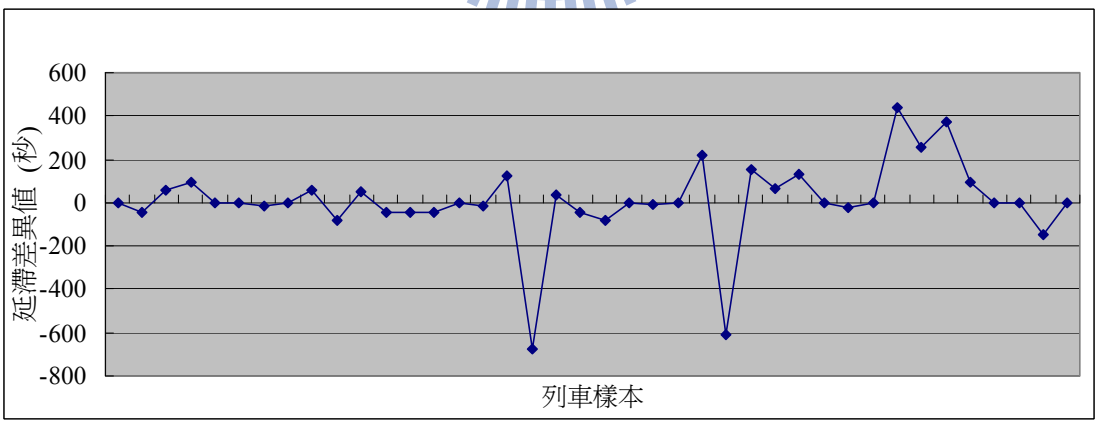


圖 3-16 受影響列車樣本之模擬延滯與實際延滯差異分佈圖

表 3.4 受影響列車樣本之模擬延滯與實際延滯差異統計表

延滯差異之絕對值(秒)	列車樣本數	樣本百分比(%)	累積百分比(%)
diff.=0	12	30.0%	30.0%
0<diff.<=60	13	32.5%	62.5%
60<diff.<=120	6	15.0%	77.5%
120<diff.<=180	3	7.5%	85.0%
180<diff.<=240	1	2.5%	87.5%
240<diff.<=300	1	2.5%	90.0%
diff.>300	4	10.0%	100.0%
	合計 =40	100.0%	

3.6 案例分析

由於重建相同軌道營運條件難度甚高，現場調查(field test)之成本亦不低，故在現實世界中若欲以現場實驗調查方式，分析比較在相同軌道營運狀況下不同排班調度策略對連鎖延滯之恢復效果差異，幾乎是不可能的。但相反地，若以模擬模式模擬推估連鎖延滯，則相對簡單可能。故基於本研究所構建模擬模式之處理能量限制，本研究選定臺鐵系統另一路段進行連鎖延滯擴散之初步模式案例分析。考量基隆—新竹路段係為臺鐵系統東、西部客運幹線主要的重疊路段，且其亦為北部都會區通勤服務之主要路段，無論在班次頻率、停站型態及列車種類之複雜程度，皆是臺鐵系統服務最忙碌之路段，且該路段所呈現之各因素特性及交互作用亦最錯綜複雜，故本研究於模式初步驗證具適用性後，再選用該路段進行模式案例分析測試。

一、基本資料輸入

如前所述及國內外相關文獻所示，模式需先輸入案例臺鐵系統基隆—新竹路段各項參數資料，應用上述之模擬模式針對雙向營運之推拉式自強號(PP)及通勤電聯車(EMU)兩種類型列車共計 184 列，分析初始延滯與連鎖延滯之關係，並加入不同調度策略分析其對連鎖延滯降低改善之程度。

二、模擬結果分析

為利本模擬模式之測試分析，本研究設計不同情境進行測試分析，其係設定初始延滯事件發生在上午 8 點整南下方向之松山—臺北站間，並設定 0 秒至 3,600 秒之不同延滯持續時間情境，以評估不同初始延滯情境下之連鎖延滯影響程度，各項分析結果摘述如后。

三、分析結果

(一)初始延滯持續時間之影響

圖 3-17 及圖 3-18 分別呈現原訂班表及初始延滯持續 3,600 秒之模擬班表時空圖，比較二圖可發現，當有 3,600 秒之初始延滯發生於南下方向之松山—臺北站間時，將會有 12 部列車受到連鎖延滯之影響而產生誤點。

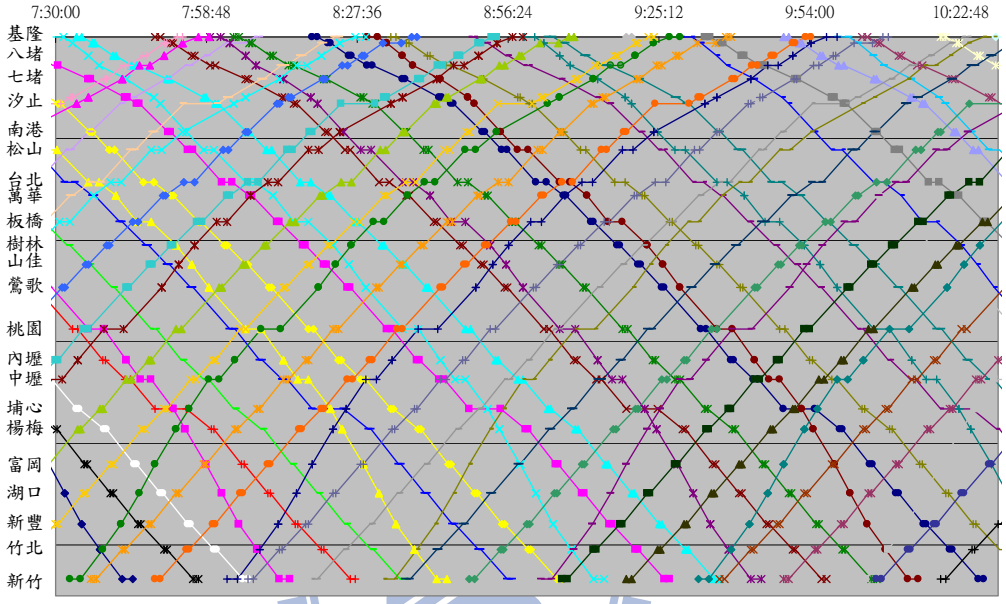


圖 3-17 原訂班表時空圖

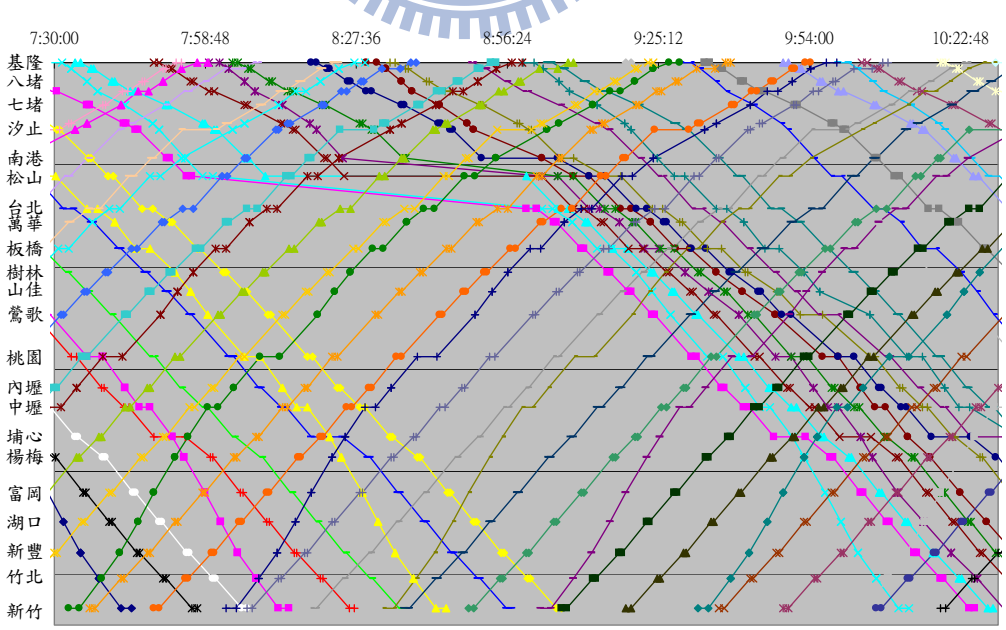


圖 3-18 模擬班表時空圖(初始延滯持續 3,600 秒)

(二)總連鎖延滯之比較分析

為利分析案例路段所有車站與二端末站(基隆及新竹站)之總連鎖延滯差異，本研究分別針對前述案例情境之模擬結果，分別計算前述二項所有列車之總連鎖延滯並繪製如圖 3-19；由圖中可發現，所有車站之總連鎖延滯明顯遠高於二端末站之總連鎖延滯。

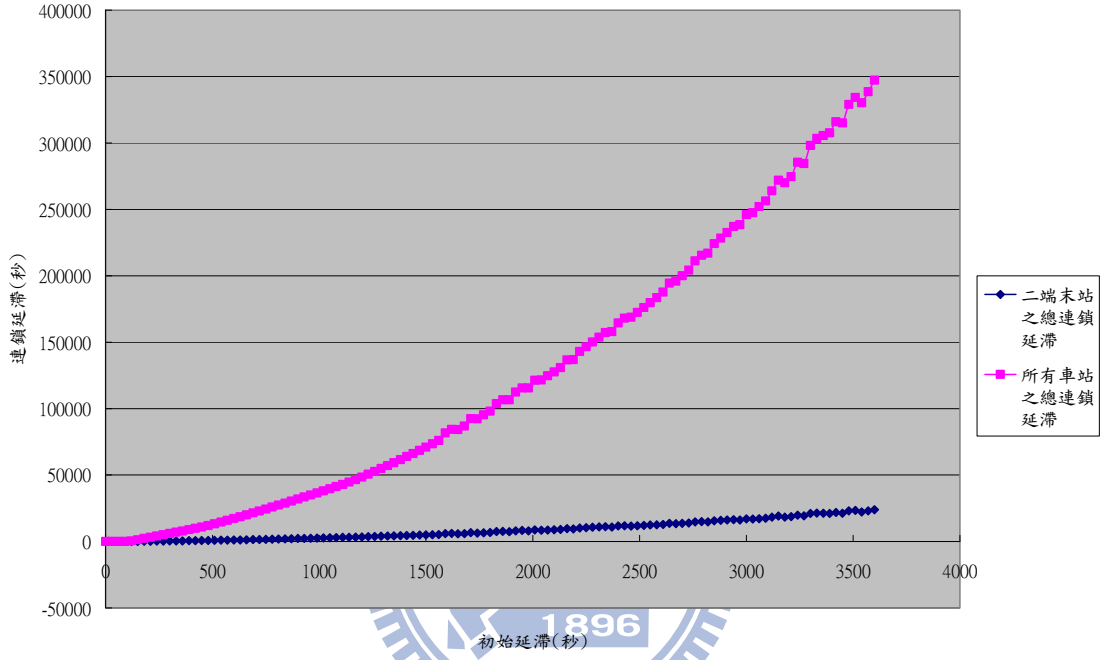


圖 3-19 所有車站與二端末站之總連鎖延滯差異

(三)受影響列車之總連鎖延滯比較分析

為了解當有初始延滯發生時不同順序列車受連鎖延滯之影響程度，本研究另分析受影響列車於所有車站之總連鎖延滯差異，圖 3-20 即呈現當有 3,600 秒之初始延滯發生於南下方向之松山—臺北站間時，受影響之 12 部列車於所有車站之總連鎖延滯情形；結果顯示通勤電聯車 EMU_0006 因最接近初始延滯發生之時間及地點，故其總連鎖延滯影響亦是 12 部列車中最高。

(四)各車站之總連鎖延滯比較分析

圖 3-21 呈現受連鎖延滯影響之列車在不同初始延滯持續時間之總連鎖延滯比較，由該圖可發現，若初始延滯發生於南下方向之松山—臺北站間，則整體而言事件南端松山—新竹路段之各站總連鎖延滯將明顯高於北端基隆—松山路段之各站總連鎖延滯。究其原因，可能係因連鎖延由初始延滯發生之松山—臺北站間往

下游路段之車站擴散所致。

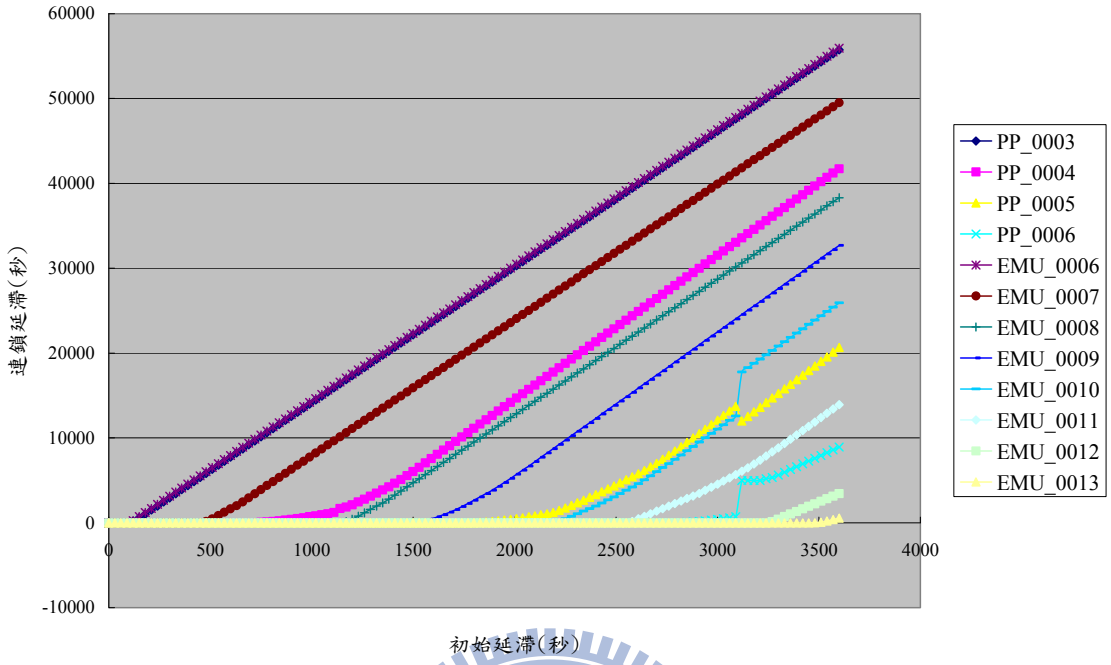


圖 3-20 受影響列車之總連鎖延滯比較

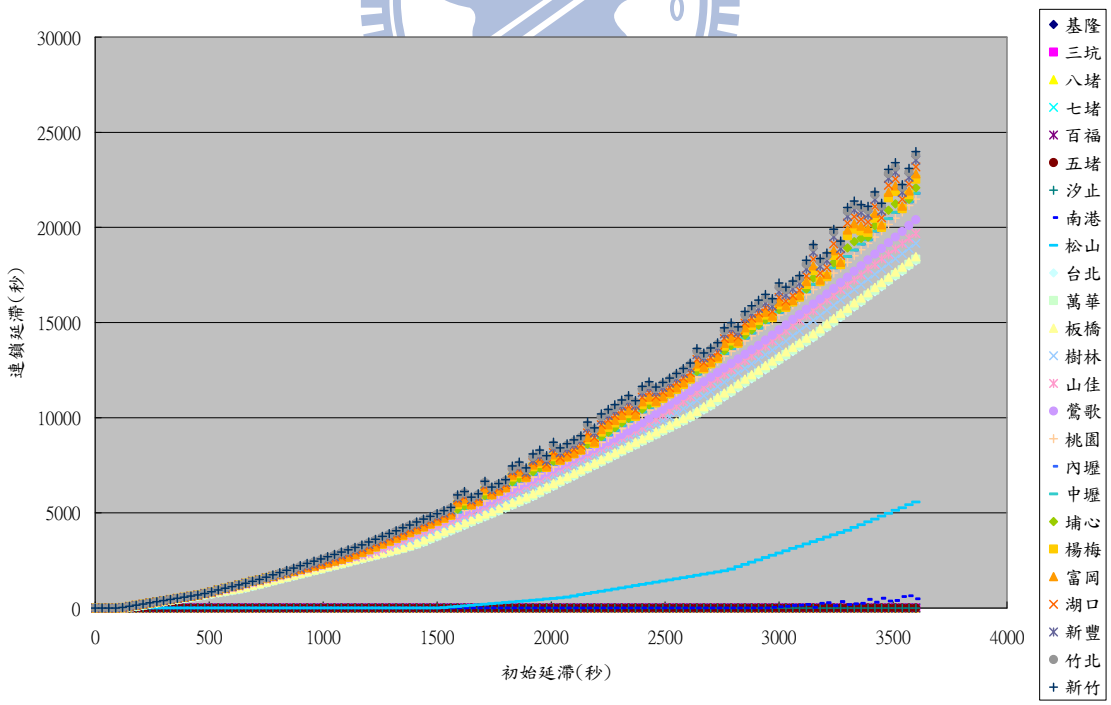


圖 3-21 各車站之總連鎖延滯比較

(五)運轉調度策略對連鎖延滯之相關改善分析

為了降低連鎖延滯對正常班表之衝擊，當列車營運產生較大之延滯時，營運調度人員通常會採取減少停站時間或站間運轉時間等調度策略進行趕點，使列車營運不致偏離既定班表過多。有關運轉調度策略對連鎖延滯之相關改善分析，茲分述如下：

1.不同運轉調度策略對連鎖延滯之改善分析

圖 3-22 及圖 3-23 呈現不同運轉調度策略對連鎖延滯之影響程度，若以本案例為例，將不同運轉調度策略對連鎖延滯之降低程度進行排序，圖 3-22 顯示就所有車站之連鎖延滯降低而言，顯然最佳策略係同時採取減少停站時間及站間運轉時間之運轉調度策略。但若就分析路段二端末車站之連鎖延滯降低而言，圖 3-23 則呈現最佳策略係為採取減少停站時間之運轉調度策略，其原因經推測可能係因列車於站內股道之選擇及列車進出站優先順序已抵銷部分連鎖延滯所致。

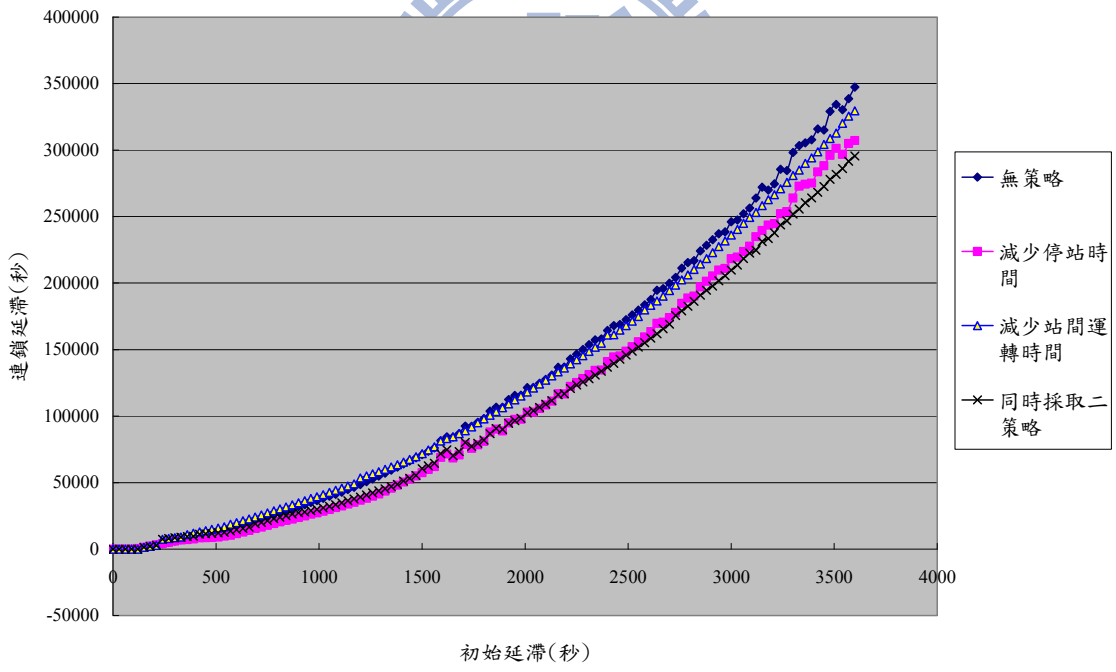


圖 3-22 不同運轉調度策略對所有站之連鎖延滯改善比較

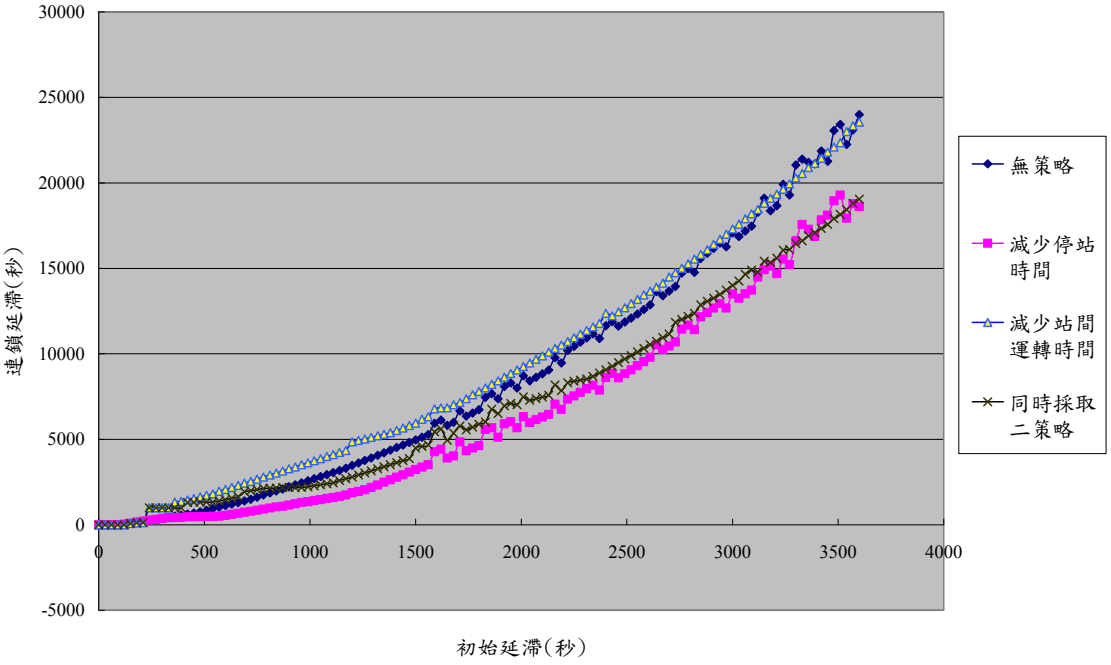


圖 3-23 不同運轉調度策略對二端末車站之連鎖延滯改善比較

2.各站同時採取減少停站時間及站間運轉時間調度策略對連鎖延滯之改善分析

圖 3-24 呈現各站同時採取減少停站時間及站間運轉時間調度策略對連鎖延滯之改善程度，相較於無任何運轉調度策略情境，若同時採取前述二項調度策略，很明顯地其對連鎖延滯之改善效果相當顯著。以本案例之新竹站為例，若同時採取前述二項調度策略，則其總連鎖延滯將從 23,989 秒降低至 19,050 秒。

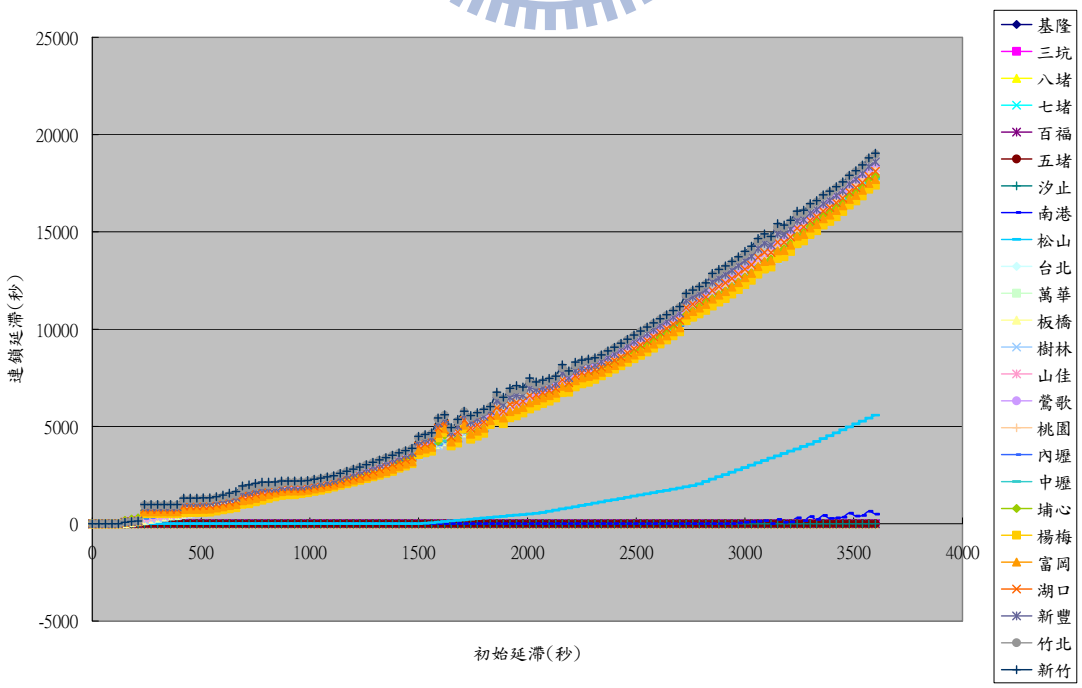


圖 3-24 各站同時採取減少停站時間及站間運轉時間調度策略對連鎖延滯之改善

3.7 小結

經由前述連鎖延滯模擬模式之構建及相關各面向之推估分析，本章可初步獲致下列之結論：

(一)連鎖延滯擴散：若設定初始延滯事件發生在南下方向之松山—臺北站間，則總連鎖延滯模擬結果顯示，在事件南端松山—新竹路段中任一車站之總連鎖延滯比北端基隆—松山路段中任一車站之總連鎖延滯都高，顯示連鎖延滯有往下游路段擴散之現象，且連鎖延滯有隨初始延滯增加而呈非線性遞增之趨勢。

(二)調度策略降低連鎖延滯之比較：整體而言，隨著初始延滯增加趕點策略對連鎖延滯降低之程度就愈明顯；若依所有車站之連鎖延滯減少程度排序，最佳策略為同時採取減少停站時間及減少站間運轉時間二種趕點策略，但若僅考慮路段二端點站之連鎖延滯，則以減少站間運轉時間為最佳策略，因臺鐵之站距較長，站間運行時間亦較長，故採用站間趕點之效果亦可能較好，而此結果亦符合先驗知識。

(三)趕點策略對連鎖延滯降低之程度：以分析路段端點新竹站為例，相較於未採取任何趕點策略情境，當有 3,600 秒之初始延滯且同時採取上述二種趕點策略時，其連鎖延滯可由 23,989 秒降低為 19,050 秒，延滯時間可降低 4,939 秒(相當約 82 分鐘)，改善成效相當顯著。

第四章 連鎖延滯影響因素之關聯分析

本章旨在接續前述連鎖延滯之影響初步分析成果，主要第一個目的即欲進一步探討不同列車密度(train density，列車數/小時)對連鎖延滯之影響關係程度，及其與路線、交通及控制等條件之交互作用關係，並建構連鎖延滯推估之模擬模式以作為後續實務應用之參考。至於第二個目的則在建立連鎖延滯與各關鍵影響因素之因果關係函數，其方法係以所建構之模擬模式先產生各種情境之延滯模擬資料，據以進行迴歸分析，初步將其函數關係設定如式(3)。

連鎖延滯 ＝ f(路線容量，列車密度，初始延滯，運轉調度策略)　　(3)

有鑑於前述路線、交通及控制條件交互關係甚為複雜，且由於本研究係定位在既定路線條件之營運班表下探討連鎖延滯之影響因素關係，故除選定相關文獻所指出直接影響連鎖延滯之初始延滯外，另就交通及控制條件中分別選出較具代表性之列車密度及運轉調度策略進行分析。各節研究內容如下：

4.1 路段選擇及輸入資料

因七堵—樹林為臺鐵系統北部區域主要通勤旅次服務路段，更是臺鐵東部幹線及西部幹線始發行駛重疊路段，無論列車班次密度、停站型態、行駛列車種類等特性皆最為複雜繁忙，該路段最能代表臺鐵複雜之系統特性及各項軟硬體設施之交互作用，故本章選擇該路段作為關聯影響因素之案例分析路段。

考量前章研究係選用七堵—樹林路段之資料進行模式驗證，基於一致性及為避免假日加班車對正常班表擾動之影響，故本章引用該路段 2009 年 10 月 20 日一般日(週二)之 CTC 資料進行列車密度、初始延滯與連鎖延滯關聯分析。有關本模擬模式所需之輸入資料，包括車站里程、列車特性、班表趕點規則、營運參數、停站類型、股道配置及全域參數(如班距、路段容量及站間趕點率)等，有關車站里程、站內股道配置及班表逐一之設定，詳如圖 4-1 之模擬程式輸入檔及附錄二說明。

4.2 不同列車密度對連鎖延滯之影響分析

由前章研究結果顯示，當系統發生初始延滯而導致軌道系統之路線容量不足時，列車之運行將受路線條件、交通條件及控制條件之交互作用影響而產生連鎖延滯之現象。雖然路線條件(例如站間軌道數目與運轉方式、站內軌道及月臺配置方式等)，交通條件(例如列車交通組成、停站時間與停站型態等)，以及控制條件(例如辦理閉塞方式、閉塞號誌配置方式、閉塞區間長度等)之改善，皆可能對連鎖延

滯產生一定程度之降低效用，但考量實務上既有硬體設施之改善比較困難，故本章研究係聚焦於探討營運列車密度對班表可靠度及連鎖延滯程度之影響。

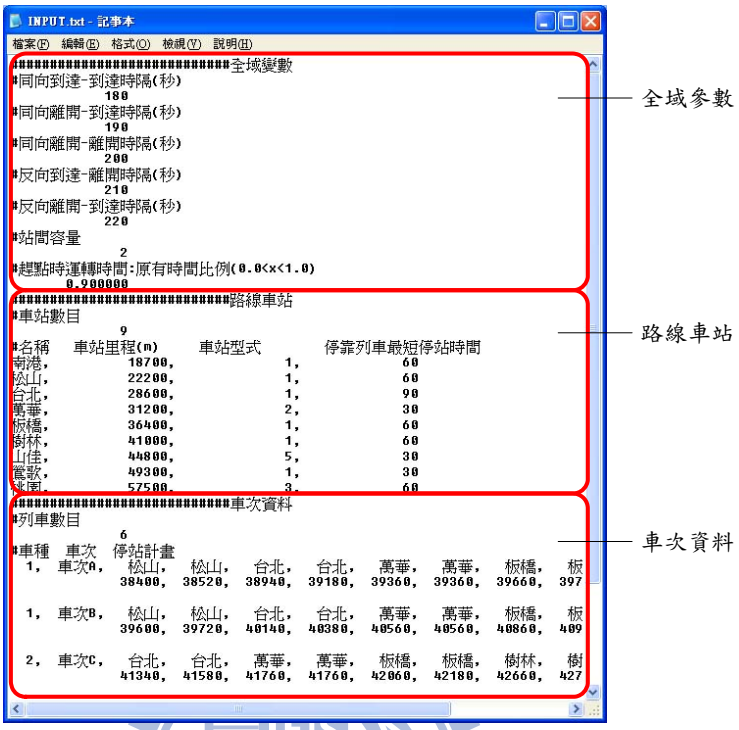


圖 4-1 模擬程式輸入檔內容

國外許多相關研究(Huisman and Boucherie, 2001、Olsson et al., 2004、交通部運輸研究所,民 94、Hwang and Liu, 2010)均已指出連鎖延滯隨列車流量密度增加而呈非線性指數函數遞增之趨勢，本節假設在硬體路線條件及系統設施不變前提下，運用本模擬模式逐一測試分析初始延滯與列車流量密度(屬交通條件)對連鎖延滯之影響情形，相關分析步驟如下：

一、 班表製作:本研究依據七堵—樹林路段 2009 年 10 月 20 日(週二)之 CTC 資料，採用該日該路段雙向 256 列車次作為分析基礎，另為分析發生初始延滯時每小時不同列車密度對連鎖延滯之影響有何不同，經以列車離站時間搜尋本路段各站間時空條件，因南下松山→臺北路段於上午 07:15~08:15 有最高尖峰小時流量密度 11 列次(相關車次清單如表 4.1 所示)，故本研究選定作為後續不同列車密度模擬分析之基礎。

二、 實驗設計：為控制各種條件一致下進行列車密度對連鎖延滯之影響分析，本研究於找出尖峰小時最高列車密度之路段，以其作為設定初始延滯之時間與空間基礎後，依前述之時空條件得到其在無初始延滯情況下之可行班表時空圖如圖 4-2 所示，若設定松山→臺北路段於上午 07:15 有 60 分鐘之初始延滯，

則其模擬班表時空圖如圖 4-3 所示，該圖顯示下游路段及後續列車之連鎖延滯擴散現象相當明顯。至於不同列車密度之實驗設計，本研究則以該日上午 07:15~08:15 通過松山→臺北路段最高之 11 列次逐步刪減 1 列次作為不同之分析情境，以 11 列次刪減任一列次為每小時 10 列車為例，即共有 11 種情境組合，依此類推；故窮舉各種列車密度所刪減列車之可能情境，經加總所有列車密度之列車組合情境共有 2,048 組，如表 4.2 所示。

表 4.1 松山→臺北路段尖峰小時 11 列次清單

車次	離站時間	車次	離站時間
13	07:15:30	1005	07:20:00
2708	07:27:00	3005	07:33:00
2710	07:38:00	3043	07:44:00
1007	07:49:00	2165	08:00:00
2712	08:05:00	3003	08:09:30
1064	08:14:30	--	--

表 4.2 尖峰小時通過列車數之各種組合情境

列車數／每小時	情境組合數	列車數／每小時	情境組合數	列車數／每小時	情境組合數
11	1	7	330	3	165
10	11	6	462	2	55
9	55	5	462	1	11
8	165	4	330	0	1

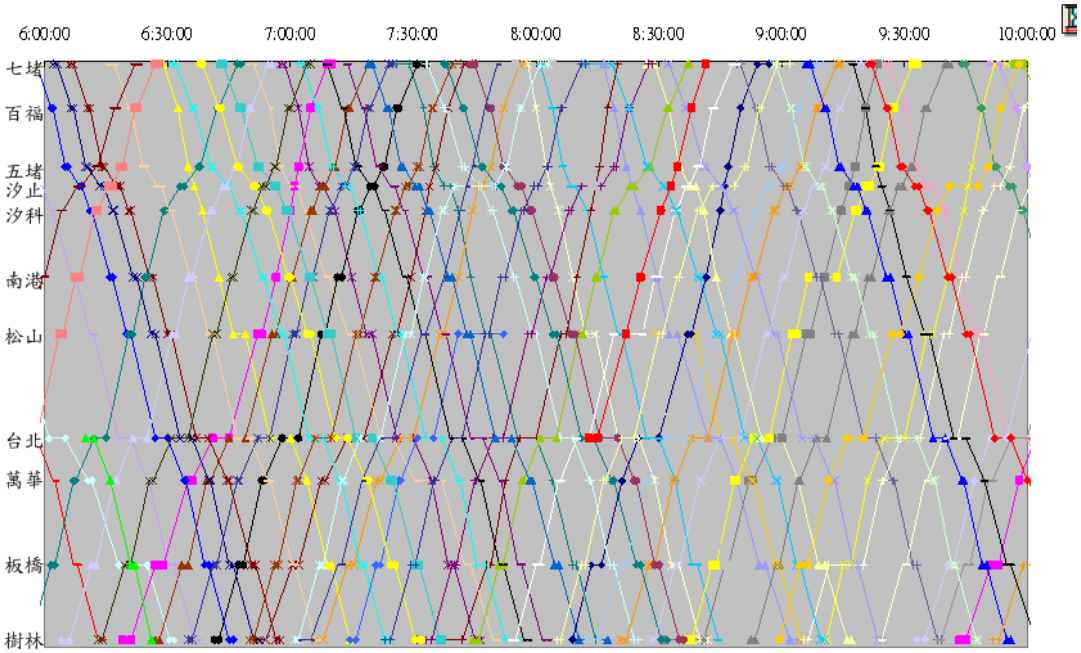


圖 4-2 七堵—樹林路段無初始延滯之時空圖

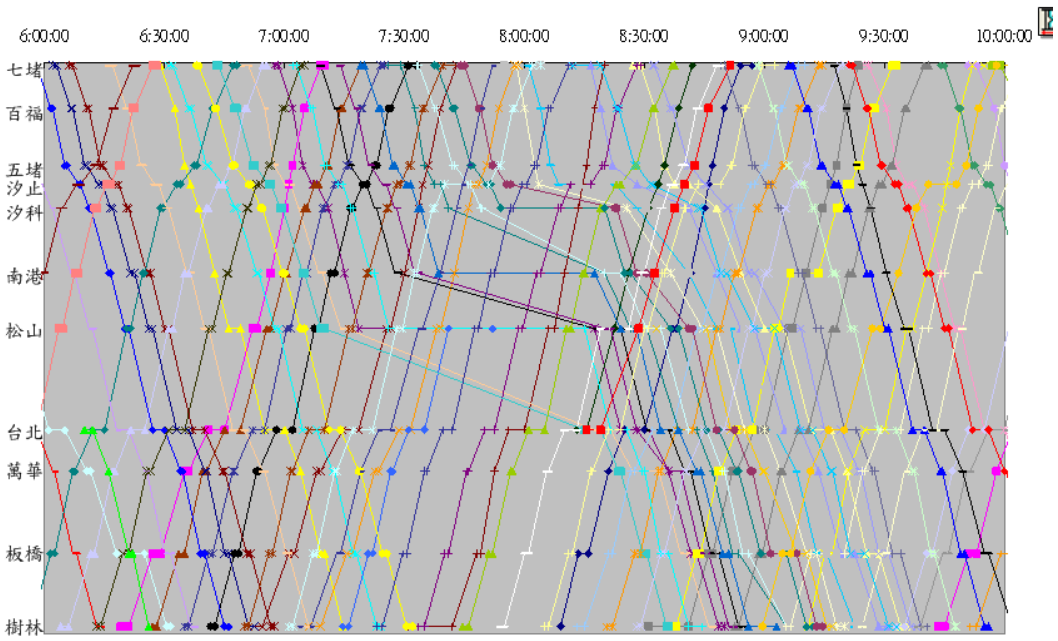


圖 4-3 七堵—樹林路段於松山→臺北路段有 60 分鐘初始延滯之模擬時空圖

三、 連鎖延滯資料：有關列車延滯之相關分析，多以分析路段之最末站(本案例即為七堵站及樹林站)作為量測基準來計算列車的平均延滯，故本研究之連鎖延滯分析亦採計最末站之延滯；為利逐步釐清列車密度與初始延滯對連鎖延滯之影響，本研究針對列車密度之 2,048 種情境及初始延滯之 61 種情境(0~60 分鐘每次增加 1 分鐘為一種情境)，以模擬模式產生 124,928 筆不同情境之連鎖延滯資料，作為後續分析之基礎。

四、 無調度策略下列車密度、初始延滯與連鎖延滯之關聯分析

為利了解列車密度與初始延滯對連鎖延滯之關聯關係暨其影響程度，以及列車密度或初始延滯單獨對連鎖延滯之影響關係，本節除針對特定初始延滯情境下列車密度與連鎖延滯，及特定列車密度情境下初始延滯與連鎖延滯等二個面向進行其關聯分析外，亦以 3D 圖同時呈現各情境下三者之關係。

(一)列車密度與連鎖延滯之落點分析：選定初始延滯為 60 分鐘下，將 X 軸設定為列車密度(0~11 列車)，Y 軸設定為每種列車密度之最末站連鎖延滯，則 2,048 個樣本點分佈如圖 4-4 所示，由分佈情形亦可看出最末站之連鎖延滯隨列車密度增加而遞增之趨勢。

(二)列車密度與連鎖延滯之關聯分析：為利比較不同初始延滯下之連鎖延滯是否有顯著差異，本研究以圖 4-4 之特定初始延滯為基礎，擴充設定 11 種初始延滯(0~60 分鐘每次增加 6 分鐘)情境，以 X 軸設定為列車密度(0~11 列車)，Y

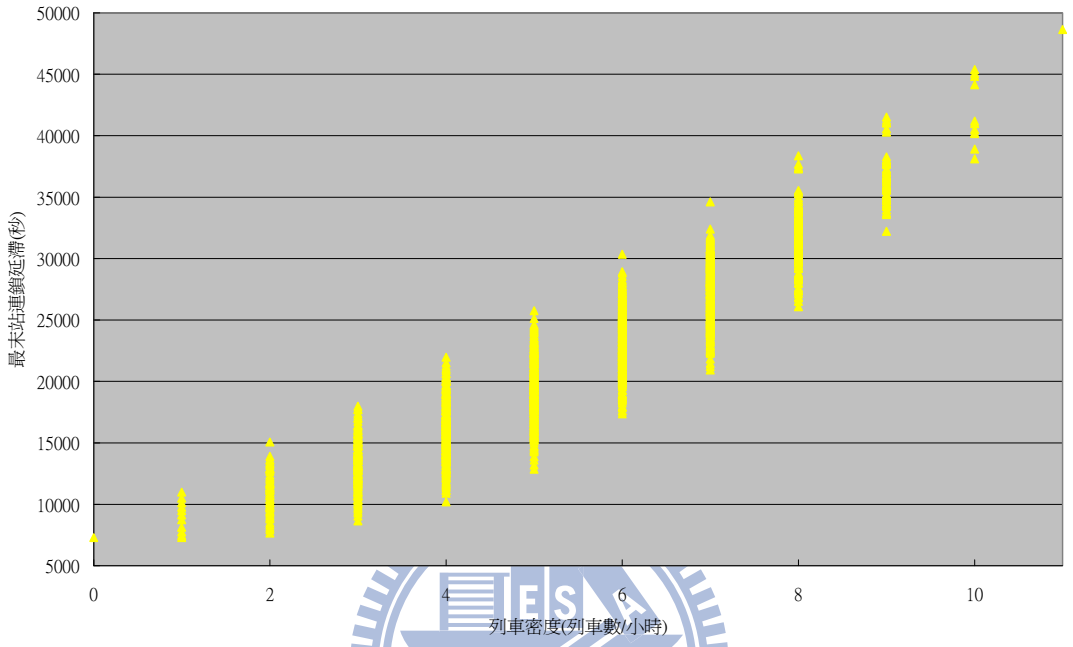


圖 4-4 初始延滯為 60 分鐘下列車密度與連鎖延滯落點圖(無調度策略)

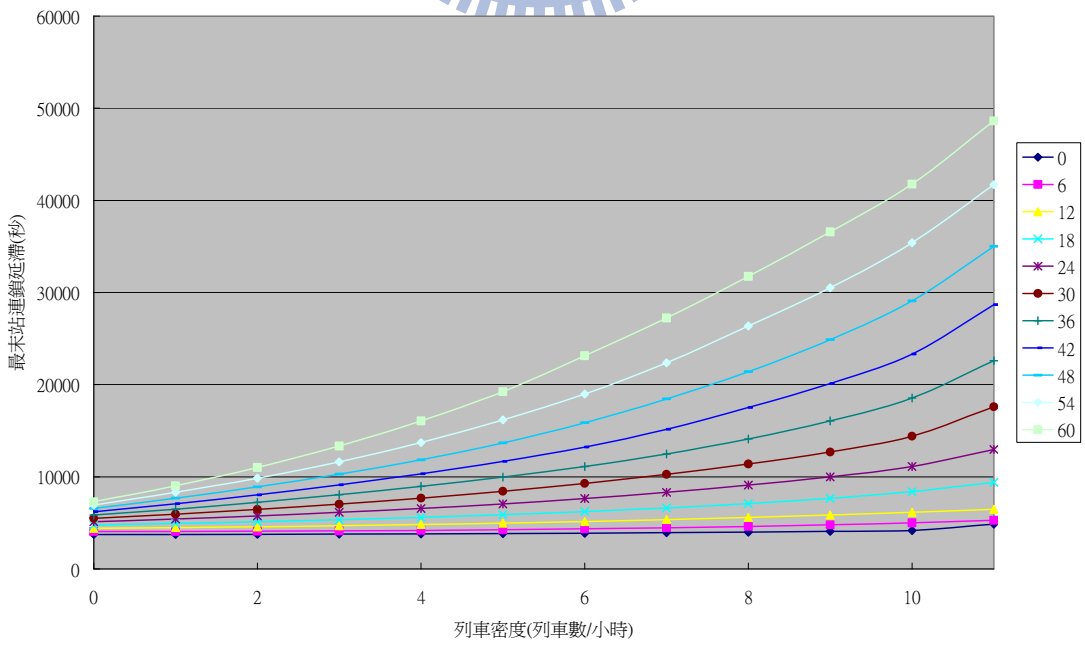


圖 4-5 各種初始延滯情境下列車密度與連鎖延滯關係圖

(三)初始延滯與連鎖延滯之落點分析：因列車密度為 6 列車之情境組合樣本數最多(462 個)，故選定列車密度為 6 列車下，將 X 軸設定為初始延滯(0~60 分鐘每次增加 3 分鐘)，Y 軸設定為最末站之連鎖延滯，則每種初始延滯之 462 個情境組合下共計 9,702 個樣本點之落點分佈如圖 4-6 所示，由分佈情形亦可看出最末站之連鎖延滯隨初始延滯增加而遞增之趨勢。

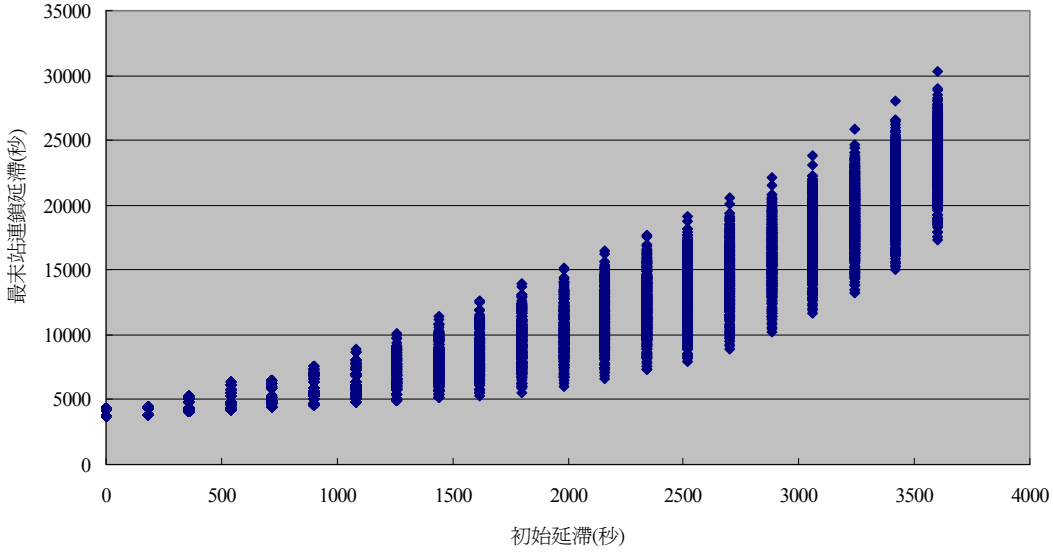


圖 4-6 列車密度為 6 列車下之初始延滯與連鎖延滯落點圖(無調度策略)

(四)初始延滯與連鎖延滯之關聯分析：為利比較不同列車密度下之連鎖延滯是否有顯著差異，本研究以圖 4-6 之特定列車密度分析為基礎，擴充設定 12 種列車密度(0~11 列車) 情境，以 X 軸設定為初始延滯(0~60 分鐘每次增加 6 分鐘)，Y 軸設定為每種列車密度之最末站連鎖延滯，將每種列車密度情境組合於各種初始延滯下之連鎖延滯取其平均值作為樣本點，畫出各種列車密度下初始延滯與連鎖延滯關係曲線如圖 4-7 所示，而由圖中曲線趨勢亦顯示，因列車密度愈大其連鎖延滯擴散效應愈明顯，故其連鎖延滯隨初始延滯增加而呈非線性陡升之趨勢亦會愈明顯。

(五)列車密度/初始延滯/連鎖延滯之 3D 圖分析：為利呈現無任何趕點調度策略下列車密度/初始延滯/連鎖延滯之交互關係分佈情形，以每一種列車密度所對應之連鎖延滯取平均為樣本點，X 軸設定為列車密度(0~11 列車)，Y 軸設定為初始延滯(0~60 分鐘每隔 1 分鐘取一樣本點)，Z 軸設定為最末站之連鎖延滯，則其 3D 關係如圖 4-8 所示，由曲面走向可發現，確存在列車密度/初始延滯愈大則連鎖延滯愈大之現象。

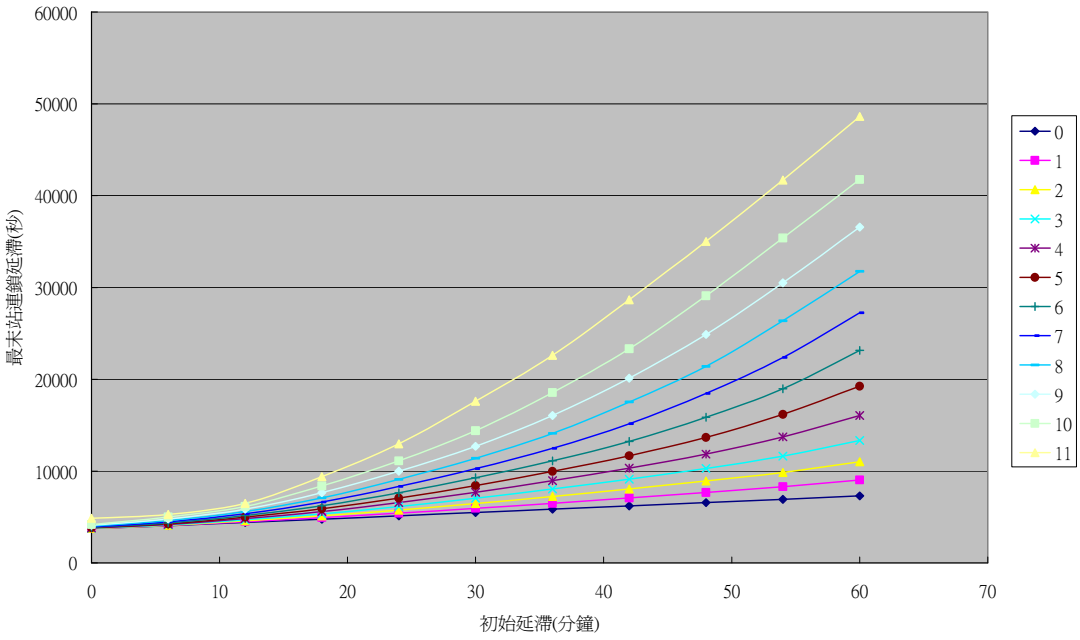


圖 4-7 各種列車密度情境下初始延滯與連鎖延滯關係圖

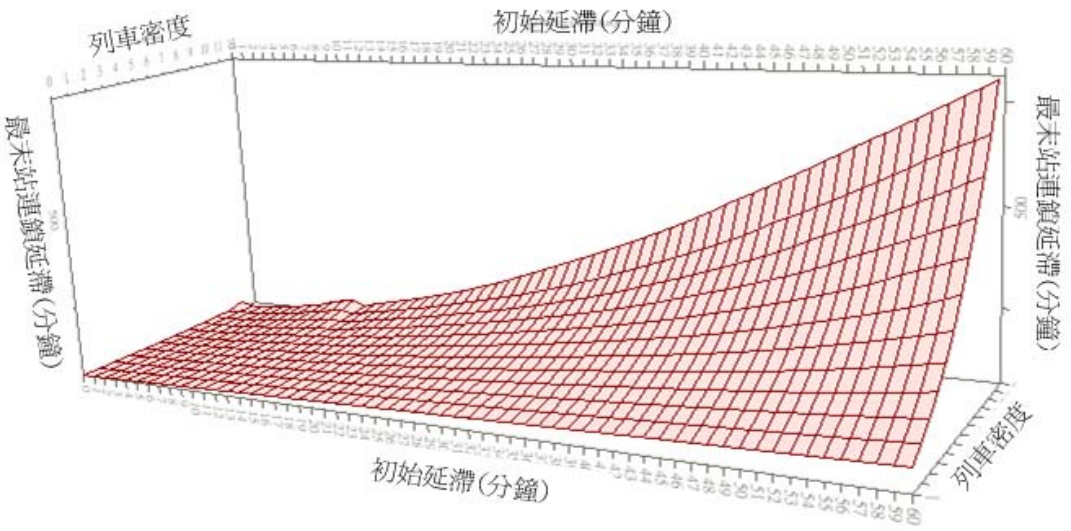


圖 4-8 列車密度、初始延滯與連鎖延滯之 3D 關係圖

4.3 採取調度策略下列車密度、初始延滯與連鎖延滯之關聯分析

為利比較不同調度策略對連鎖延滯之影響情形，本研究選定初始延滯較大之 60 分鐘情境，比較四種不同調度策略下之結果如圖 4-9 所示。圖 4-9 中之策略 A 為站間趕點，策略 B 則為站內趕點，每一種策略組合情境為一條曲線。由圖中曲線可得知，各種策略情境皆呈列車密度愈高則其連鎖延滯陡升之趨勢，且站內趕點(策略 B)之改善效果明顯比站間趕點(策略 A)好，而同時採取該二種趕點策略對

連鎖延滯之改善效果更佳；故後續有關採取調度策略下連鎖延滯之關聯分析即以同時採取該二種趕點策略之情境進行分析。

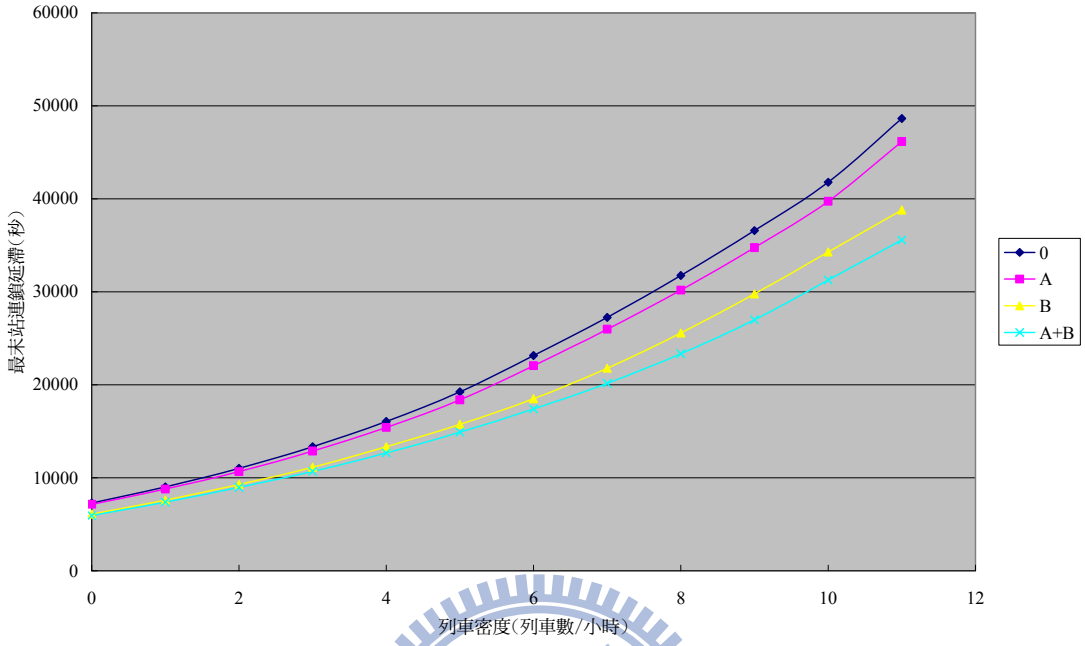


圖 4-9 不同調度策略組合情境下列車密度與連鎖延滯關係圖

至於加入趕點調度策略情境之列車密度/初始延滯/連鎖延滯分析部分，係以列車密度之 2,048 種情境、初始延滯之 21 種情境(0~60 分鐘每隔 3 分鐘為一種情境)及有無站內趕點或站間趕點策略之 4 種情境，應用模擬模式產生共計 172,032 筆連鎖延滯之資料，作為後續分析之基礎。

一、列車密度與連鎖延滯之落點分析：選定初始延滯為 60 分鐘下，將 X 軸設定為列車密度(0~11 列車)，Y 軸設定為每種列車密度之最末站連鎖延滯，則 2,048 個樣本點分佈如圖 4-10 所示，由分佈情形可看出最末站之連鎖延滯隨列車密度增加而遞增之趨勢。而與圖 4-4 無調度策略下之連鎖延滯相較，除連鎖延滯明顯降低外，其各列車密度樣本點也明顯較為集中。

二、列車密度與連鎖延滯之關聯分析：為利比較不同初始延滯下之連鎖延滯是否有顯著差異，本研究以圖 4-10 之特定初始延滯為基礎，擴充設定 11 種初始延滯(0~60 分鐘每隔 6 分鐘)情境，以 X 軸設定為列車密度(0~11 列車)，Y 軸設定為每種列車密度之最末站連鎖延滯，將各種初始延滯下之 2,048 樣本點依各列車密度之連鎖延滯取平均後各畫出列車密度與連鎖延滯關係曲線如圖 4-11 所示，而由圖中曲線趨勢顯示，因初始延滯愈大其連鎖延滯擴散效應愈明顯，

故其連鎖延滯隨列車密度增加而呈非線性陡升之趨勢亦愈明顯。與無調度策略之圖 4-5 相較，其最明顯差異處在於，除連鎖延滯明顯降低外，曲線陡升現象亦趨於平緩，顯示同時採取二種調度策略可有效降低連鎖延滯，並增加班表之可靠度。

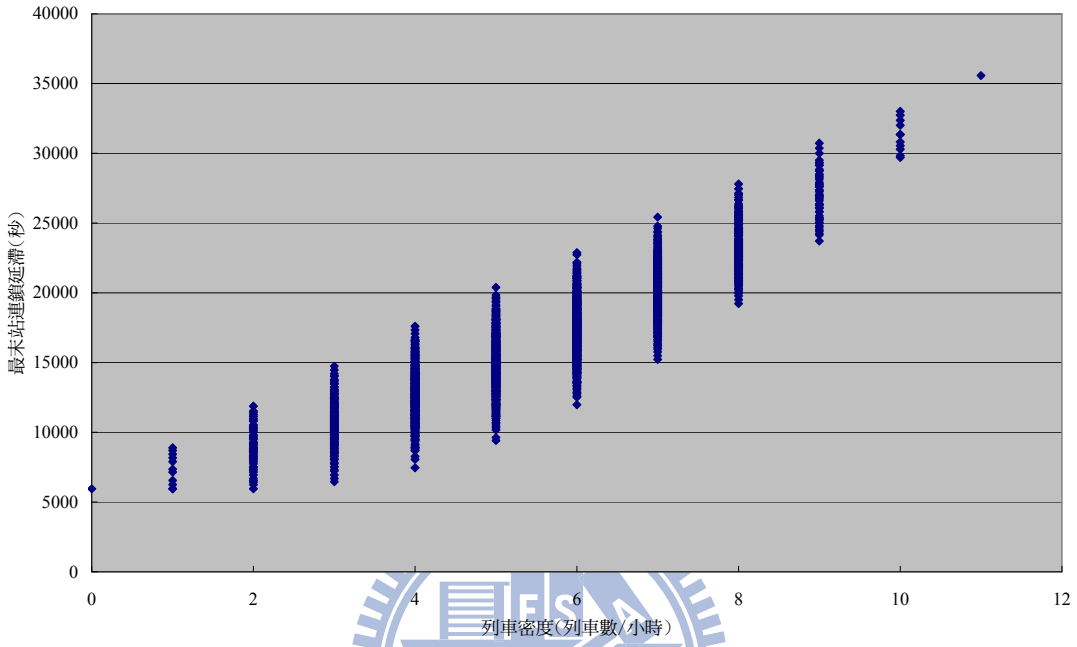


圖 4-10 初始延滯為 60 分鐘下列車密度與連鎖延滯分佈圖(同時採取二種調度策略)

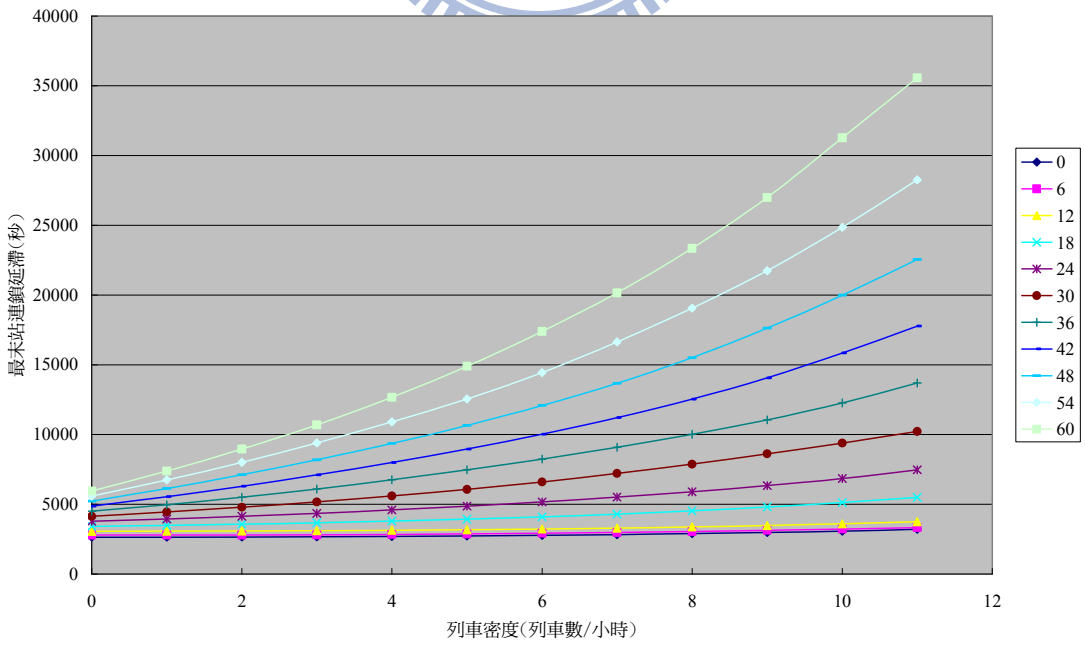


圖 4-11 各種初始延滯情境下列車密度與連鎖延滯關係圖

三、初始延滯與連鎖延滯落點分析：因列車密度為 6 列車之列車情境組合樣本數最多(462 個)，故選定列車密度為 6 列車下，將 X 軸設定為初始延滯(0~60 分鐘每隔 3 分鐘)，Y 軸設定為最末站之連鎖延滯，則每種初始延滯有 462 個情境共計 9,702 個樣本點落點分佈如圖 4-12 所示，由分佈情形亦可看出最末站之連鎖延滯隨初始延滯增加而遞增之趨勢。而與圖 4-6 無調度策略下之連鎖延滯相較，除連鎖延滯明顯降低外，其分佈之型態大致相同。

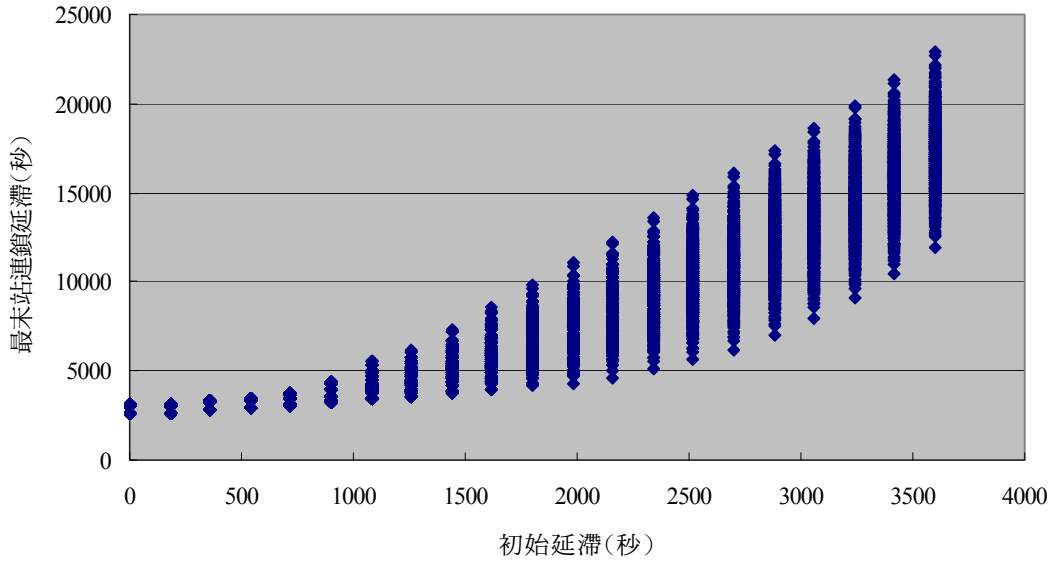


圖 4-12 列車密度 6 列車下之初始延滯與連鎖延滯落點圖(同時採取二種調度策略)

四、初始延滯與連鎖延滯之關聯分析：為利比較不同列車密度下之連鎖延滯是否有顯著差異，本研究以圖 4-12 之特定列車密度分析為基礎，擴充設定 12 種列車密度(0~11 列車)情境，以 X 軸設定為初始延滯(0~60 分鐘每隔 6 分鐘)，Y 軸設定為每種列車密度之最末站連鎖延滯，將每種列車密度情境組合於各種初始延滯下之連鎖延滯取平均作為樣本點，畫出各種列車密度下初始延滯與連鎖延滯關係曲線如圖 4-13 所示，而由圖中曲線趨勢顯示，亦因列車密度愈大其連鎖延滯擴散效應愈明顯，故其連鎖延滯隨初始延滯增加而呈非線性陡升之趨勢亦會愈明顯。而與不考慮調度策略之圖 4-7 相較，其最明顯差異處在於，除連鎖延滯明顯降低外，曲線陡升之現象趨於平緩，顯示同時採取二種調度策略可有效降低連鎖延滯，並可增加班表之可靠度。

五、同時採取二種調度策略下列車密度/初始延滯/連鎖延滯之 3D 圖分析：為利呈現同時採用站內趕點及站間趕點二種調度策略下列車密度/初始延滯/連鎖延滯之交互關係分佈情形，以每一種列車密度所對應之連鎖延滯取平均為樣本

點，X 軸設定為列車密度(0~11 列車)，Y 軸設定為初始延滯(0~60 分鐘每隔 1 分鐘取一樣本點)，Z 軸設定為最末站之連鎖延滯，則其 3D 關係如圖 4-14 所示，由曲面走向可發現，確存在列車密度/初始延滯愈高則連鎖延滯愈高之現象。另若欲瞭解各情境下連鎖延滯之分佈範圍，則依連鎖延滯最大值(Max)、最小值(Min)及平均值(Avg)等範圍值分別繪製，其 3D 關係如圖 4-15 所示，而由圖中顯示，隨著列車密度/初始延滯愈增加，連鎖延滯之分佈範圍亦愈大。

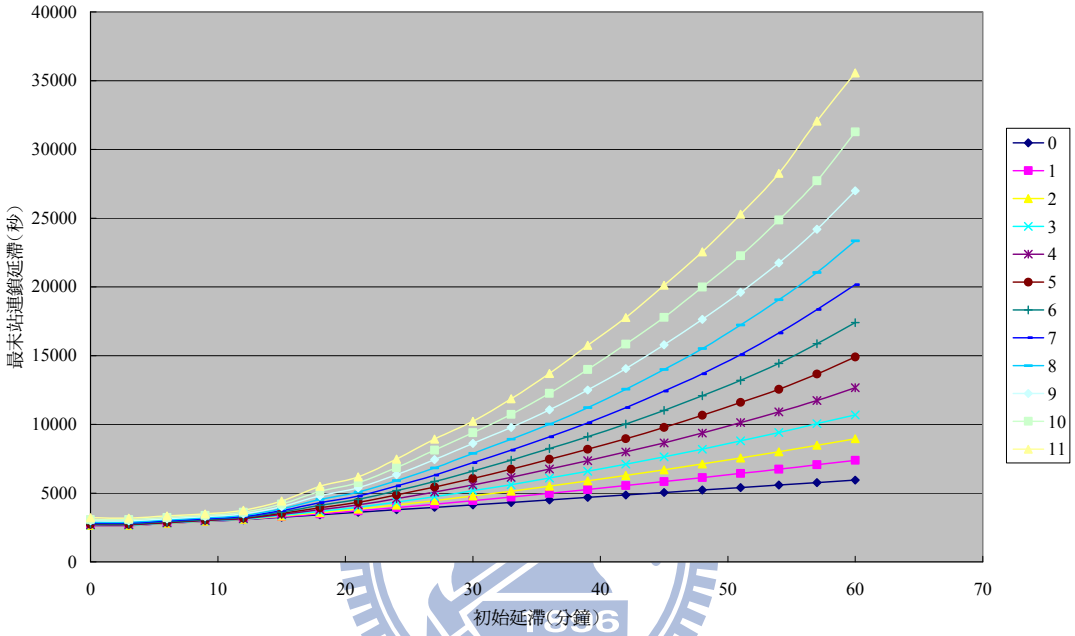


圖 4-13 各種列車密度情境下初始延滯與連鎖延滯關係圖

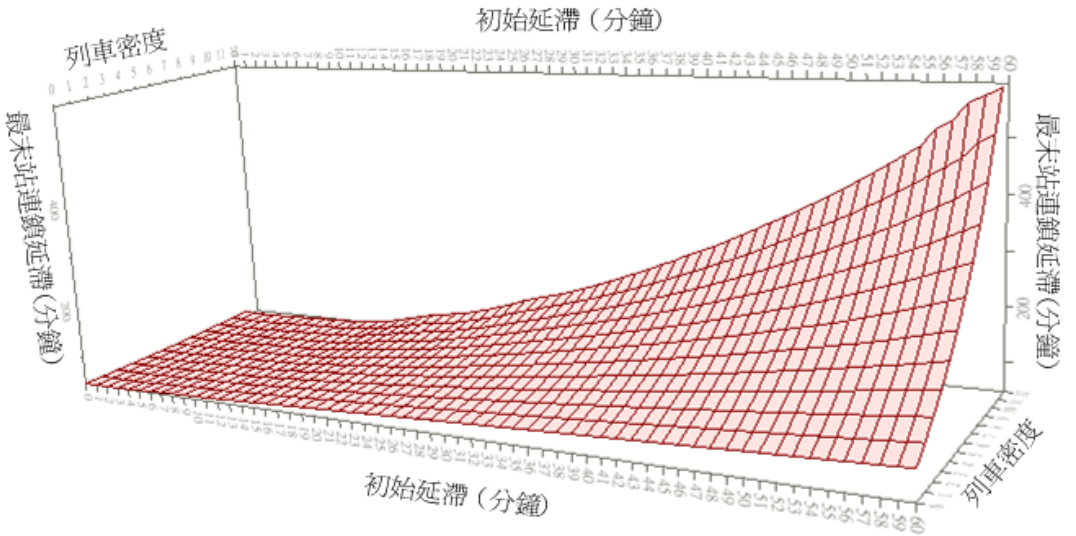


圖 4-14 同時採取二種調度策略下列車密度/初始延滯/連鎖延滯(平均值)之關係圖

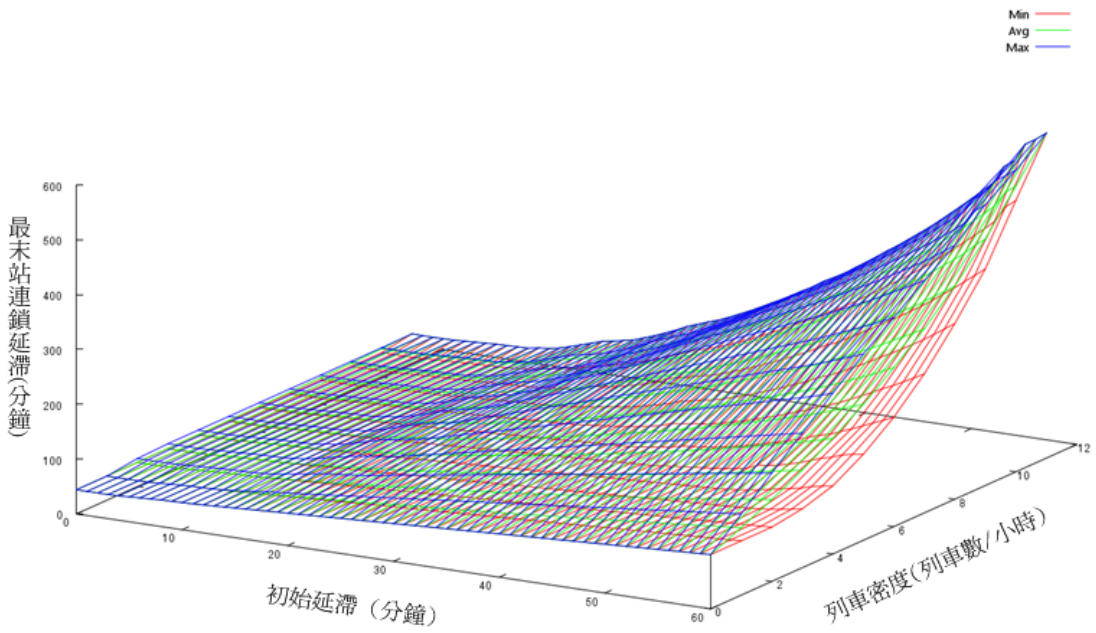


圖 4-15 同時採取二種調度策略下列車密度/初始延滯/連鎖延滯(範圍值)之關係

六、不同調度策略組合情境下列車密度/初始延滯/連鎖延滯之 3D 圖分析：為利瞭解不同調度策略組合下列車密度/初始延滯/連鎖延滯之交互關係分佈，每一種列車密度所對應之連鎖延滯取平均為樣本點，將 X 軸設定為列車密度(0~11 列車)，Y 軸設定為初始延滯(0~60 分鐘每隔 1 分鐘取一樣本點)，Z 軸設定為最末站之連鎖延滯，依無任何調度策略、站間趕點(策略 A)、站內趕點(策略 B)及採用二種調度策略(策略 A+B)等四種組合情境分別繪製，則其 3D 關係如圖 4-16 所示。由曲面走向亦可發現在三度空間之連鎖延滯隨列車密度/初始延滯增加而增高之現象，及各策略組合之差異狀況。

4.4 迴歸分析

雖然以往已有相關文獻指出(連鎖)延滯與關鍵影響變數間存在非線性之類似指數函數關係，經由本研究上述各變數之關聯分析亦已發現類似之趨勢，惟鑑於過去並未有相關研究針對臺灣鐵路系統構建其估算模式，本研究乃進一步依據上述之模擬資料應用統計迴歸分析方法嘗試加以構建，以作為直接推估連鎖延滯之用。茲以列車密度(2,048 種情境)、初始延滯(21 種情境)及調度策略(4 種情境)之情境組合所模擬產生之 172,032 組資料，嘗試以不同函數式進行迴歸分析，相關分析過程與結果分述如下：

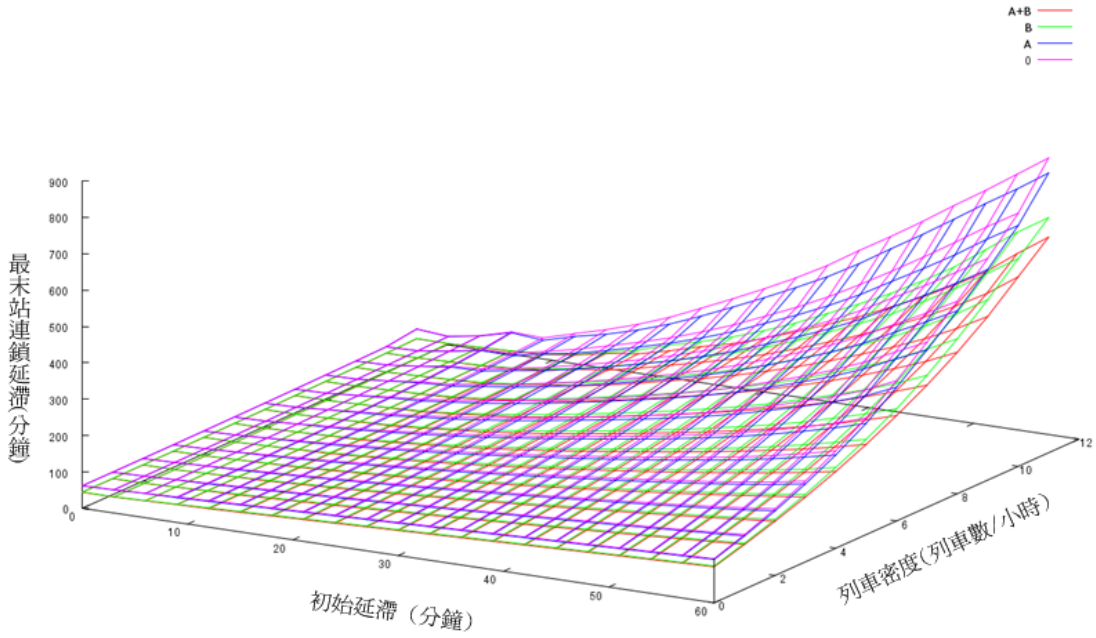


圖 4-16 不同調度策略組合情境下列車密度/初始延滯/連鎖延滯之關係圖

一、迴歸函數設定：為利構建適合臺鐵系統之連鎖延滯估算模式，本研究以前述三項關鍵變數為基礎，考量相關文獻已指出(連鎖)延滯與關鍵影響變數間之非線性函數關係結論，及前述本研究彙整分析發現之個別變數對連鎖延滯之非線性函數影響關係，經嘗試以非線性之指數函數、乘冪函數(個別變數及雙重變數乘冪)及用以對照比較之線性函數等不同模式，如式(4)～式(8)等五種函數式進行迴歸分析，其中因站間趕點及站內趕點等二種調度策略係以虛擬變數之方式設定，故其皆以線性型態於模式中呈現(指數函數於取 ln 後為線性)。

$$y = \beta_0 e^{\beta_1 x_1} e^{\beta_2 x_2} e^{\beta_3 x_3} e^{\beta_4 x_4} \tag{4}$$

$$y = \beta_0 + \beta_1 x_1^{\,2} + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 \tag{5}$$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2^{\,2} + \beta_3 x_3 + \beta_4 x_4 \tag{6}$$

$$y = \beta_0 + \beta_1 x_1^{\,2} + \beta_2 x_2^{\,2} + \beta_3 x_3 + \beta_4 x_4 \tag{7}$$

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 \tag{8}$$

其中：y=連鎖延滯(秒)；

x_1=列車密度(列/小時)；

x_2=初始延滯(秒)；

x_3=站間趕點虛擬變數(0 或 1)；

x_4=站內趕點虛擬變數(0 或 1)；

β_n=待校估之參數，$n \in \{0,1,2,3,4\}$。

二、迴歸模式分析

(一)統計量檢定：上述各函數式之迴歸分析結果歸納如表 4.3 所示，其結果顯示各函數式之參數推估值皆相當顯著，顯示列車密度/初始延滯/調度策略三項自變數皆為因變數連鎖延滯之關鍵影響變數。另由各函數迴歸分析之判定係數 R^2（coefficient of determination）及 F 檢定值可得知，各函數式之自變數對因變數連鎖延滯皆具解釋能力，但其中非線性函數之判定係數皆高於線性函數，並以式(4)指數函數之判定係數 0.9265 明顯高於其他函數型式，顯示以指數函數最適合描述該三項自變數與因變數之因果關係，此亦符合相關文獻之結論及本研究模擬結果所呈現之型式。

表 4.3 五種迴歸模式之校估結果

參數 / 迴歸模式	β_0 (t 值)	β_1 (t 值)	β_2 (t 值)	β_3 (t 值)	β_4 (t 值)	R^2 (F 值)
$y=\beta_0 e^{\beta_1 x_1} e^{\beta_2 x_2} e^{\beta_3 x_3} e^{\beta_4 x_4}$	**2165.659** (4695.43)	**0.08685** (360.10)	**0.00051** (1379.29)	**-0.047** (-58.53)	**-0.292** (-365.31)	**0.9265** (542248)
$y=\beta_0+\beta_1 x_1^2+\beta_2 x_2+\beta_3 x_3+\beta_4 x_4$	-527.014 (-29.73)	91.406 (285.91)	4.329 (792.77)	-453.212 (-38.08)	-2321.202 (-195.02)	**0.8134** (187426)
$y=\beta_0+\beta_1 x_1+\beta_2 x_2^2+\beta_3 x_3+\beta_4 x_4$	-589.956 (-28.50)	1014.979 (316.89)	0.001 (913.84)	-453.212 (-42.66)	-2321.202 (-218.51)	**0.8513** (246271)
$y=\beta_0+\beta_1 x_1^2+\beta_2 x_2^2+\beta_3 x_3+\beta_4 x_4$	1976.018 (138.30)	91.406 (322.41)	0.001 (919.74)	-453.212 (-42.94)	-2321.202 (-219.92)	**0.8532** (250018)
$y=\beta_0+\beta_1 x_1+\beta_2 x_2+\beta_3 x_3+\beta_4 x_4$	-3092.987 (-126.43)	1014.979 (281.40)	4.329 (788.76)	-453.212 (-37.88)	-2321.201 (-194.03)	**0.8115** (185103)

註：所有係數值皆為顯著。

(二)模式應用分析

1.依據前述之模式檢定結果，式(4)之指數函數可直接應用估算臺鐵系統路段端末站之連鎖延滯，而無需以模擬模式估算。整體而言，由該指數函數可發現列車密度與初始延滯等二項變數對於連鎖延滯有正向增加之影響，且以列車密度之影響較大；而站間趕點及站內趕點等二種調度策略則對連鎖延滯有負向降低之影響，此傾向結果甚符先驗知識。另該指數函數亦可透過虛擬變數(0 或 1)之設定，進行站間趕點及站內趕點等二種調度策略對連鎖延滯之影響分析，當該虛擬變數設定為 1 則代表採取該變數之調度策略，若二虛擬變數均設定為 1 則代表同時採取二種調度策略。

2.由該函數式之係數值顯示，平均而言若採取站間趕點調度策略則對連鎖延滯之折減比例約為 95.4%，若採取站內趕點調度策略則對連鎖延滯之折減比例

可達約 74.7%，若同時採取二項調度策略則對連鎖延滯之折減比例可達約71.2%，顯見本臺鐵案例路段之資料其站內趕點調度策略對連鎖延滯降低之功效較為明顯。至於迴歸模式(5)(6)(8)係本研究用以比較分析之乘冪函數及線性函數，雖其 R^2 值尚高且各變數係數皆顯著，但其截距項為負並不合理，故該三項迴歸模式不適合應用於連鎖延滯之分析。

3.另有關前述建議採用之指數函數可能存在無法適用自變數為極端值之問題(如列車密度等自變數皆為 0 但因變數連鎖延滯為 2165.659)，經檢視因本研究之實驗設計為控制各種條件一致下進行列車密度對連鎖延滯之影響分析，故本研究係以松山→臺北路段之尖峰小時最高列車密度作為設定列車密度及初始延滯之時空基礎，以進行七堵—樹林路段之連鎖延滯分析，故因自變數與因變數之時空範圍不同，以列車運轉特性雖控制松山—臺北變因條件，但其他路段的延滯變因或班表擾動因無法完全控制，故亦可能導致七堵—樹林路段端末站產生連鎖延滯，此亦即為該指數函數之常數項不為 0 之主要原因。

(三)3D 圖分析：為便於將所推導之指數函數迴歸式估計結果與模擬資料相互比較，本研究以所構建指數函數 $y = 2165.659 \bullet e^{0.08685x_1} \bullet e^{0.00051x_2} \bullet e^{-0.047x_3} \bullet e^{-0.292x_4}$ ，依無任何調度策略、站間趕點、站內趕點及採用二種調度策略等四種運轉調度策略組合，分別繪製其三維之指數迴歸式如圖 4-17 所示，其分佈趨勢與圖 4-16 極為相近。

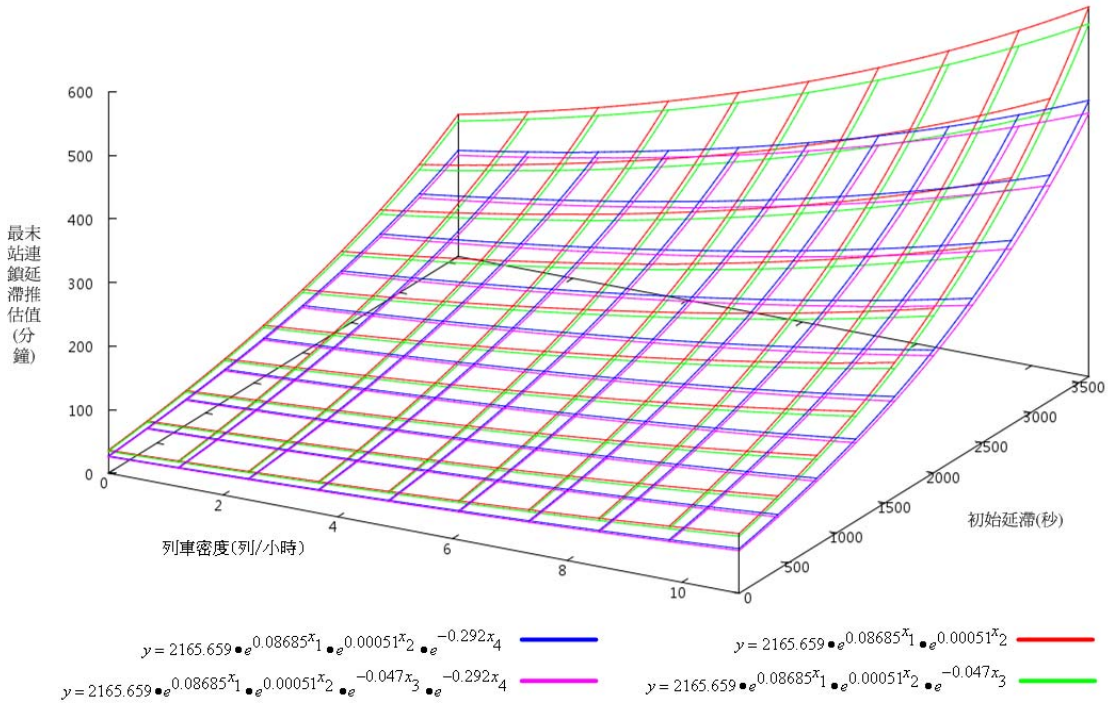


圖 4-17 指數迴歸模式所求解之列車密度/初始延滯/連鎖延滯關係圖

4.5 小結

軌道系統容量不足往往是導致列車延滯之主因，尤其在初始延滯(first delay)發生後，後續所引發的連鎖延滯(knock-on delay)擴散效應對列車正常營運之影響更是鉅大，因此欲確保系統之可靠度及服務品質，如何快速有效地降低連鎖延滯是一項重要的課題。而由於軌道系統之路線條件、交通條件及控制條件與其交互作用是影響路線容量之要素，任一環節出問題即會造成容量不足而導致延滯。

本章以多車種、複線運轉及多類型月臺型式之傳統臺鐵系統為例，針對系統各條件中影響列車連鎖延滯最關鍵之列車密度/初始延滯/調度策略，研析其對連鎖延滯之影響，期能有助於降低連鎖延滯，提昇列車營運之可靠度及服務品質。就實務調度觀點而言，本研究除可具體量化關鍵影響變數對於連鎖延滯之影響程度及其對於降低延滯之改善效果外，所採用之站內趕點及站間趕點二項調度策略分析結果，亦可提供營運者作為選擇實務調度策略之參考。

另由表 4.1 以南下松山→臺北路段於上午 07:15~08:15 之最高尖峰小時流量密度 11 列次逐步刪減車次，模擬分析不同列車密度之連鎖延滯，雖模擬結果會隨輸入班表不同而異，但因其輸入班表需為可營運班表，故所得之連鎖延滯推估結果仍具有該類型班表之一般化連鎖延滯特性及其趨勢意義。

第五章 初始延滯區位及運轉調度策略對連鎖延滯之關聯分析

為利瞭解外生初始延滯發生之區位、持續時間及運轉調度策略對連鎖延滯之影響，本章繼續以先前建立之連鎖延滯模擬模式及七堵—樹林路段案例資料為基礎，更進一步分析營運單位關切之外生初始延滯發生區位、持續時間及運轉調度策略對連鎖延滯之影響程度，以便更能掌握運轉調度策略對連鎖延滯降低及營運班表恢復之能力，俾利營運單位參考因應。

為了確實比較不同外生初始延滯發生區位對連鎖延滯之影響差別，本研究透過前述自行構建之模擬模式，設計不同之初始延滯發生區位情境進行分析比較。若從初始延滯發生於站間區位之觀點，本研究係假設於分析路段七堵—樹林之 10 個站間區位皆有可能發生初始延滯，而初始延滯之持續時間則假設由 0 秒至 1 小時，至於分析之結果則以每小時之平均連鎖延滯呈現。此外，除了探討不同初始延滯發生區位、持續時間對連鎖延滯之影響程度外，本研究亦分析同時採取站間趕點及站內趕點等二種運轉調度策略對連鎖延滯之影響，並針對初始延滯發生於分析路段之上、下游站間區位對於連鎖延滯之影響差異進行分析，再以該路段所有車站及二端末站之觀點，分析比較其連鎖延滯之差異程度，最後以 3D 圖呈現各情境下之初始延滯發生區位、持續時間及有無運轉調度策略之連鎖延滯關係。相關分析茲述如下：

5.1 基本資料蒐集

有關本研究之模擬模式需要輸入許多資料，包括車站里程、列車運轉特性、班表回復規則、營運參數、列車停站型態、車站股道配置、班距、路段容量、最小站間運轉時間與表訂站間運轉時間比值、及表訂班表等。經彙整本研究臺鐵系統案例路段之實際各項交通及控制條件資料如表 5.1 所示，有關列車站間運轉時間之最大趕點比例係依據實務臺鐵系統規定設定為 0.9，至於列車型式係考量推拉式自強號(PP)及通勤電聯車(EMU)兩種類型列車，計雙向共 256 列次。

表 5.1 七堵—樹林路段之交通及控制條件資料

車站	站間距離(公里)	推拉式自強號(PP)		通勤電聯車(EMU)		最小停站時間(秒)	車站股道配置
		停站時間(秒)	站間運轉時間(秒)	停站時間(秒)	站間運轉時間(秒)		
七堵		120		60		60	型 I
	2.7		210		210		
百福		--		30		30	型 IV
	3.6		150		180		
五堵		--		30		30	型 IV
	1.2		90		180		
汐止		--		60		60	型 I
	1.5		120		120		
汐科		--		30		30	型 IV
	4.1		240		270		
南港		--		60		60	型 I
	3.5		210		210		
松山		120		60		60	型 I
	6.4		360		360		
臺北		240		120		90	型 I
	2.6		240		240		
萬華		--		30		30	型 II
	5.2		300		270		
板橋		90		60		60	型 I
	4.6		270		300		
樹林		--		30		30	型 I

5.2 所有車站之連鎖延滯模擬分析

本研究之模擬模式將依據不同輸入資料產生不同之連鎖延滯模擬結果，首先為利了解初始延滯持續時間所產生連鎖延滯對正常營運列車之干擾影響，本章先假設有一初始延滯發生於上午 7:00 南下之松山—臺北站段、持續時間為 1 小時之案例進行模擬，並呈現比較所有車站之連鎖延滯模擬結果，圖 5-1 及圖 5-2 即分別為本案例之原表訂班表時空圖，及上述 1 小時初始延滯發生後之模擬時空圖，比較該二張時空圖可發現，圖 5-2 呈現之模擬結果共有 13 列車受連鎖延滯影響。為利進一步分析不同初始延滯發生區位對連鎖延滯之影響，本研究以南下各站間發生初始延滯之情境進行連鎖延滯分析，相關模擬結果分述如下：

一、無運轉調度策略下不同初始延滯發生區位、持續時間情境下所有車站之連鎖延滯分析

圖 5-3 顯示無運轉調度策略下不同初始延滯發生之區位、持續時間情境下所有車站之連鎖延滯模擬分析結果，很明顯地，結果顯示當初始延滯發生在愈上游路段、初始延滯之持續時間愈長，其連鎖延滯愈大。

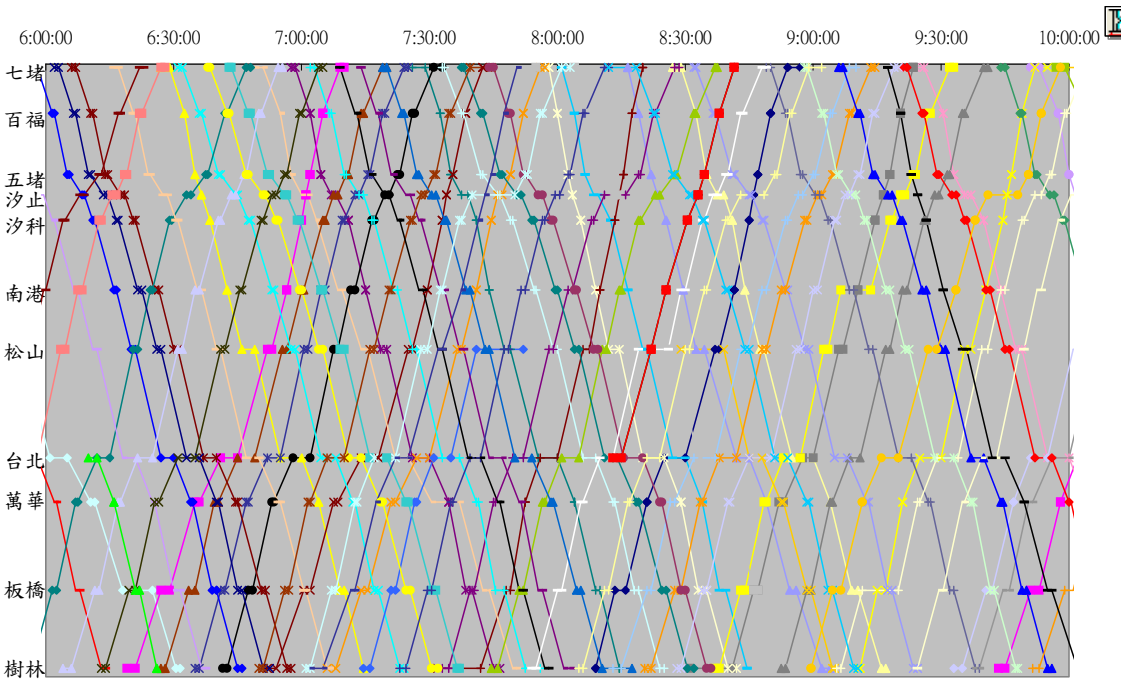


圖 5-1 原表訂班表時空圖

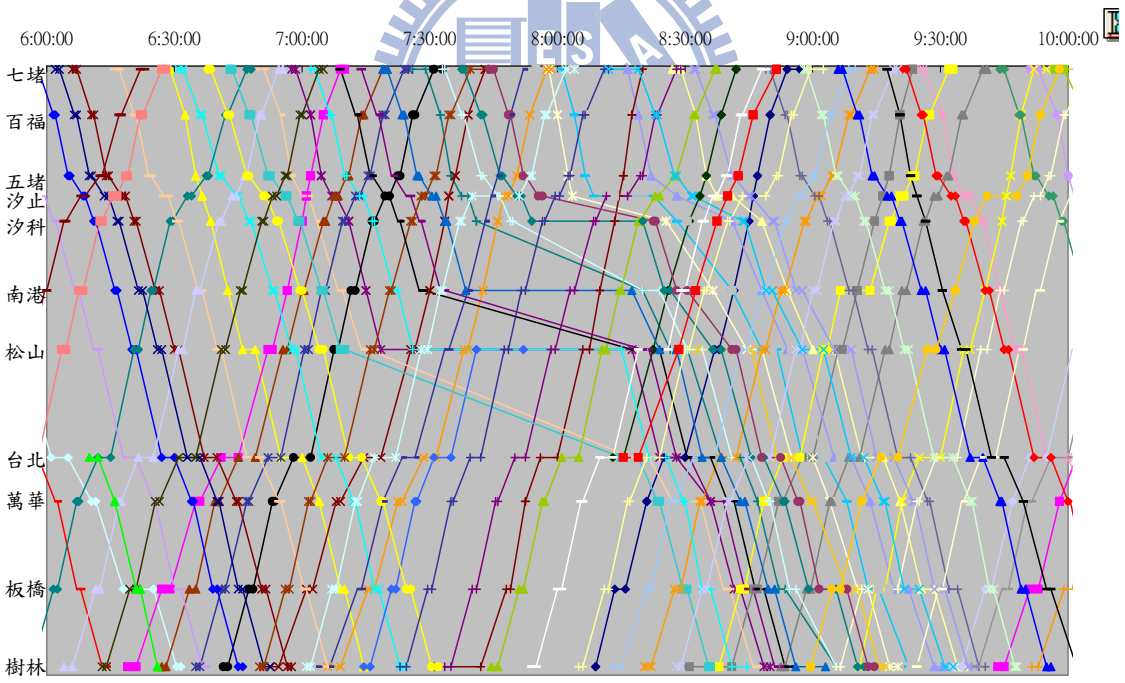


圖 5-2 初始延滯持續 1 小時之班表模擬時空圖

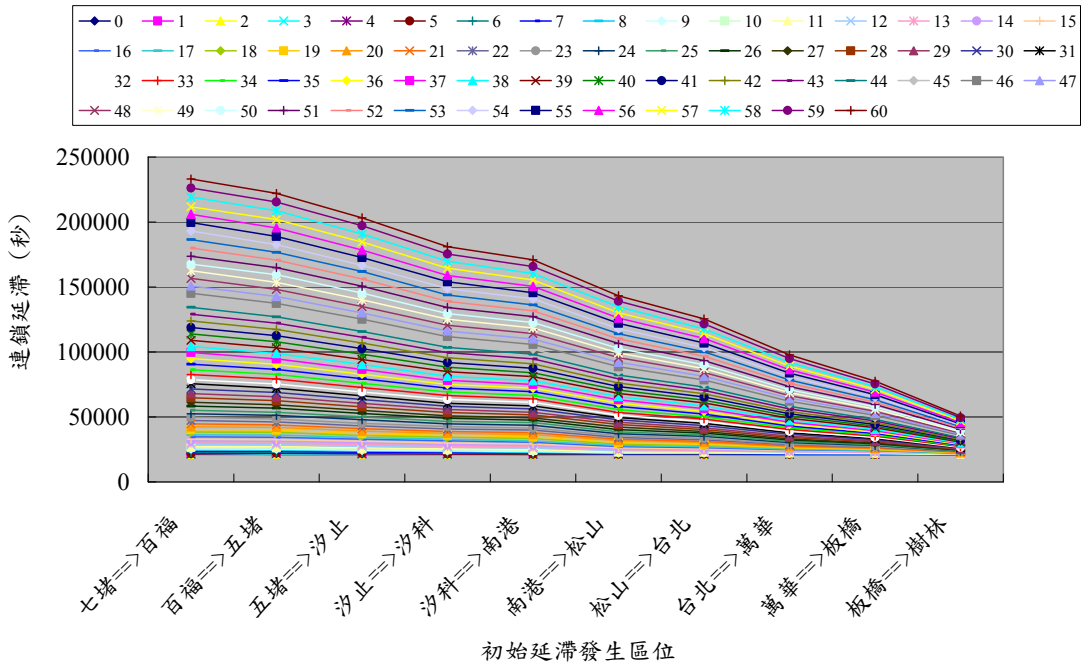


圖 5-3 無運轉調度策略下所有車站之連鎖延滯

二、同時採取二運轉調度策略不同初始延滯發生區位、持續時間情境下所有車站之連鎖延滯分析

圖 5-4 顯示同時採取站間趕點及站內趕點二運轉調度策略下，不同初始延滯發生區位、持續時間情境下所有車站之連鎖延滯模擬分析結果，與圖 5-3 之結果比較，同時採取上述二運轉調度策略時，其連鎖延滯已明顯降低許多。

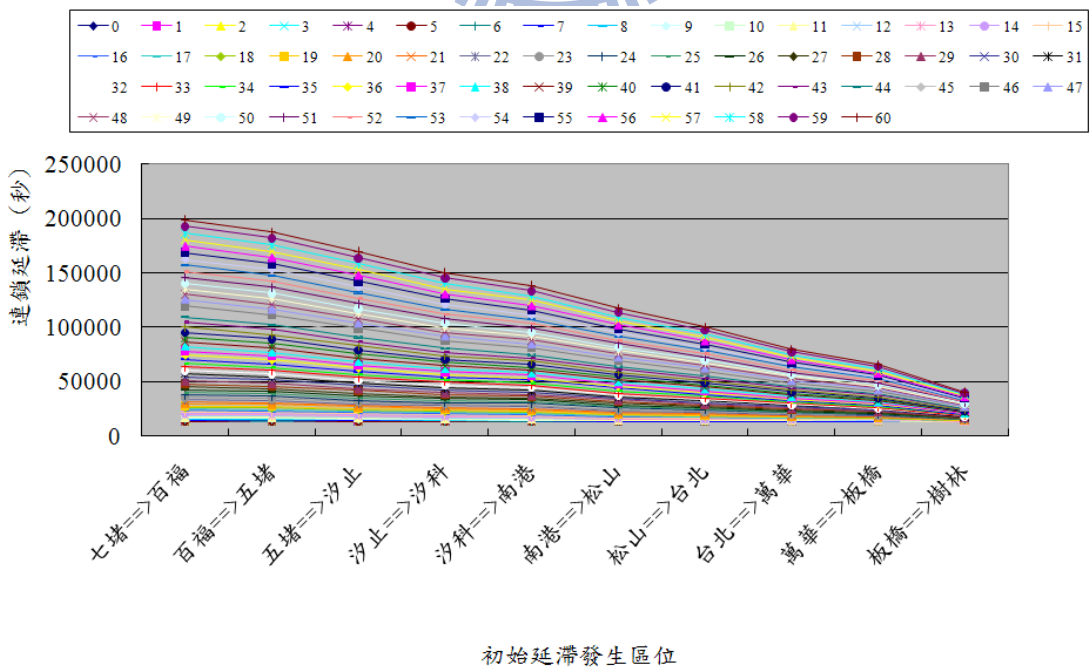


圖 5-4 同時採取二運轉調度策略下所有車站之連鎖延滯

三、同時採取二運轉調度策略下所有車站之連鎖延滯降低分析

圖 5-5 顯示同時採取站間趕點及站內趕點二運轉調度策略下，不同初始延滯發生區位、持續時間情境下所有車站連鎖延滯降低之模擬分析結果，由圖中可發現，若初始延滯發生於上游路段，則同時採取上述二運轉調度策略對連鎖延滯降低之效果將比初始延滯發生於下游路段之情境為佳。另圖中趨勢亦顯示若初始延滯發生區位距離分析路段之二處端末站愈遠，則二運轉調度策略對連鎖延滯降低之效果亦愈佳。

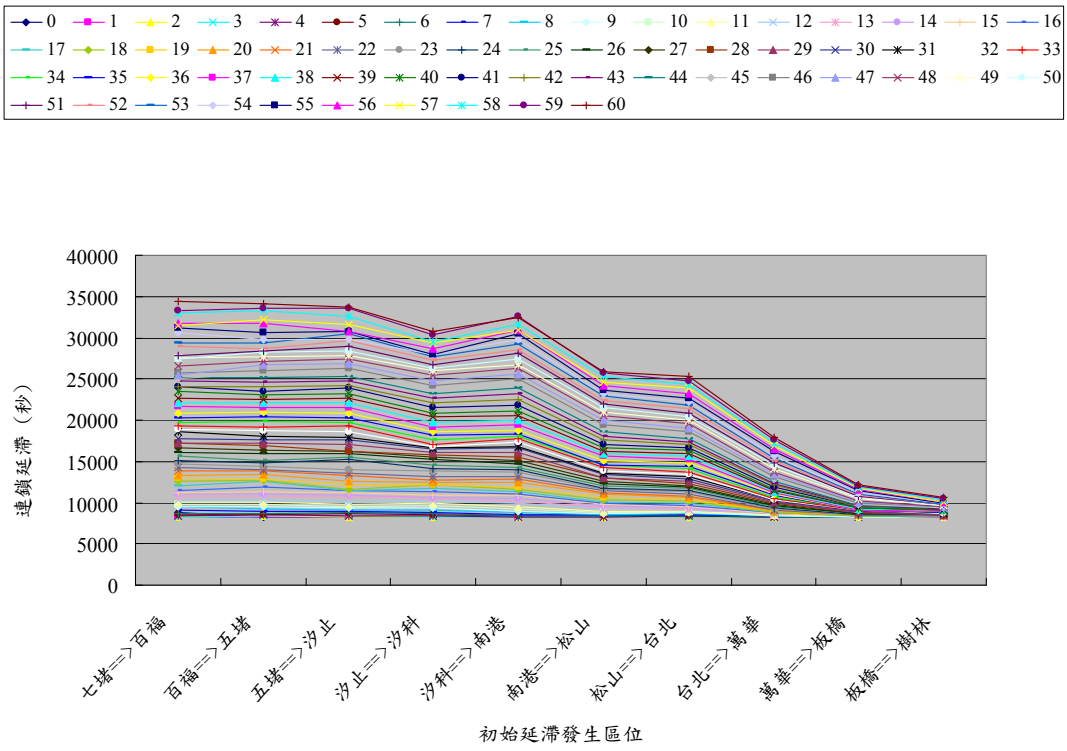


圖 5-5 同時採取二運轉調度策略下所有車站之連鎖延滯降低

四、初始延滯發生區位、持續時間與所有車站連鎖延滯降低之關聯分析

圖 5-6 顯示初始延滯發生區位、持續時間，與同時採取站間趕點及站內趕點二運轉調度策略下所有車站連鎖延滯降低之 3D 關係，圖中顯示若初始延滯之持續時間較長且初始延滯發生區位靠近上游路段(例如七堵—南港段)，則同時採取二運轉調度策略後對連鎖延滯降低之程度就愈大，其亦顯示當初始延滯發生愈靠近上游路段，則運轉調度策略對營運班表恢復至原訂班表之效果也愈好。

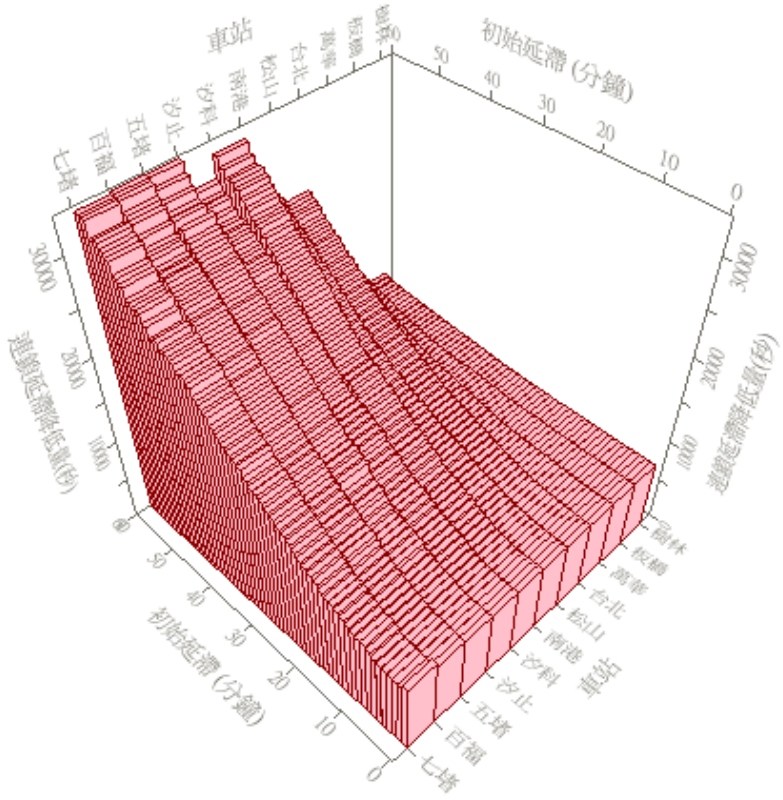


圖 5-6 初始延滯發生區位、持續時間與所有車站連鎖延滯降低之 3D 關係

5.3 端末車站之連鎖延滯模擬分析

為了與研究路段所有車站之連鎖延滯模擬結果相比較，本節另就研究路段二端末車站之連鎖延滯進行模擬分析，同樣以南下方向之連鎖延滯分析為例，相關模擬結果分述如下：

一、無運轉調度策略下不同初始延滯發生區位、持續時間情境下二端末車站之連鎖延滯分析

圖 5-7 顯示無運轉調度策略下，所有列車在不同初始延滯發生區位、持續時間情境下二端末車站之連鎖延滯模擬分析結果，整體而言除了板橋—樹林路段外，初始延滯發生在各站間路段之連鎖延滯模擬結果並無明顯之差別。而依據分析結果顯示，當初始延滯發生在板橋—樹林路段、持續時間為 1 小時，其受連鎖延滯影響之 13 部列車其二端末車站之連鎖延滯較其他站間路段情境約低了 72 分鐘，其主要原因係當初始延滯發生區位太靠近端末車站時，則其連鎖延滯之擴散影響將不明顯所致。

二、同時採取二運轉調度策略不同初始延滯發生區位、持續時間情境下二端末車站之連鎖延滯分析

圖 5-8 顯示同時採取二運轉調度策略下，所有列車在不同初始延滯發生區位、持續時間情境下二端末車站之連鎖延滯模擬分析結果，與圖 5-7 之結果比較，顯然地同時採取上述二運轉調度策略時，松山—樹林間之四個站間路段其連鎖延滯相較於其他站間路段為高，其原因則係靠近端末車站運轉調度策略較未及發揮功效。

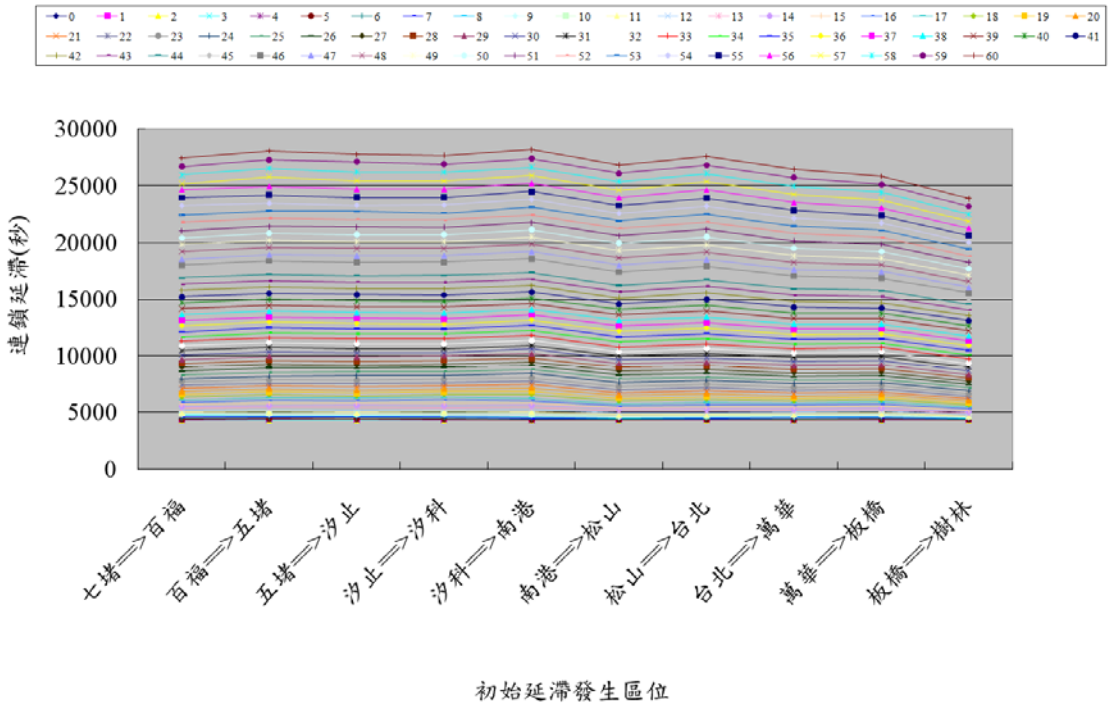


圖 5-7 無運轉調度策略下二端末車站之連鎖延滯

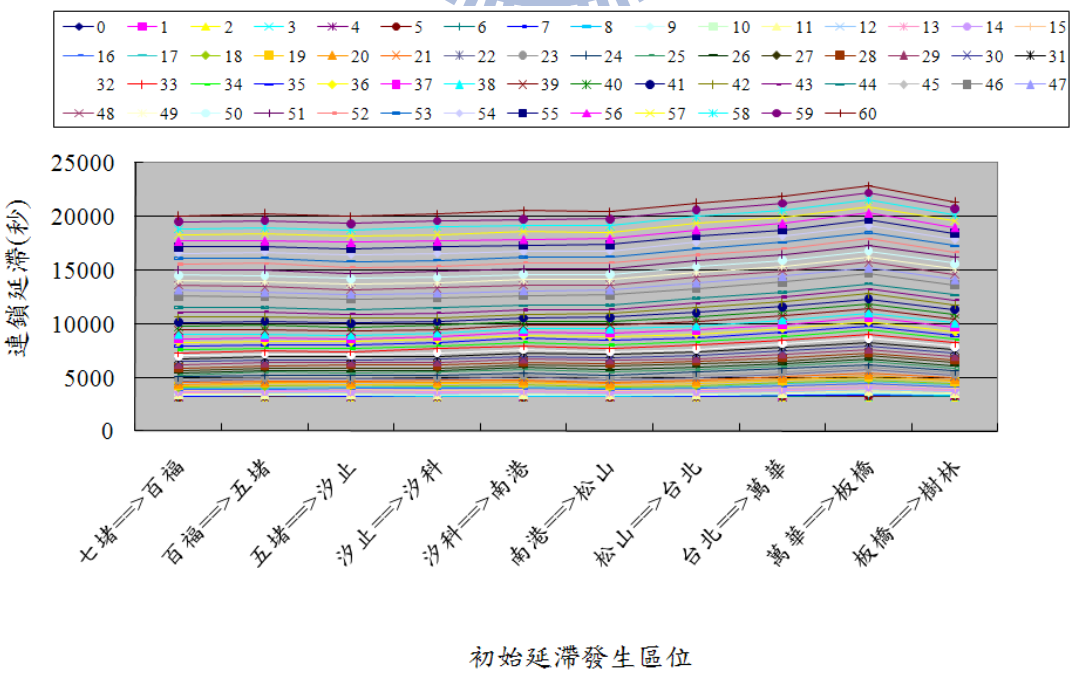


圖 5-8 同時採取二運轉調度策略下二端末車站之連鎖延滯

三、同時採取二運轉調度策略下二端末車站之連鎖延滯降低分析

圖 5-9 顯示同時採取站間趕點及站內趕點二運轉調度策略下，不同初始延滯發生區位、持續時間情境下二端末車站連鎖延滯降低之模擬分析結果，由圖中可發現，若初始延滯發生於上游路段(七堵—南港區段)，當初始延滯持續時間達 1 小時，則上游路段之連鎖延滯降低量將大於下游路段約 122 分鐘，此結果亦顯示就二端末車站而言，若初始延滯發生區位距離分析路段端末站愈遠，則二運轉調度策略對連鎖延滯降低之效果亦愈佳。

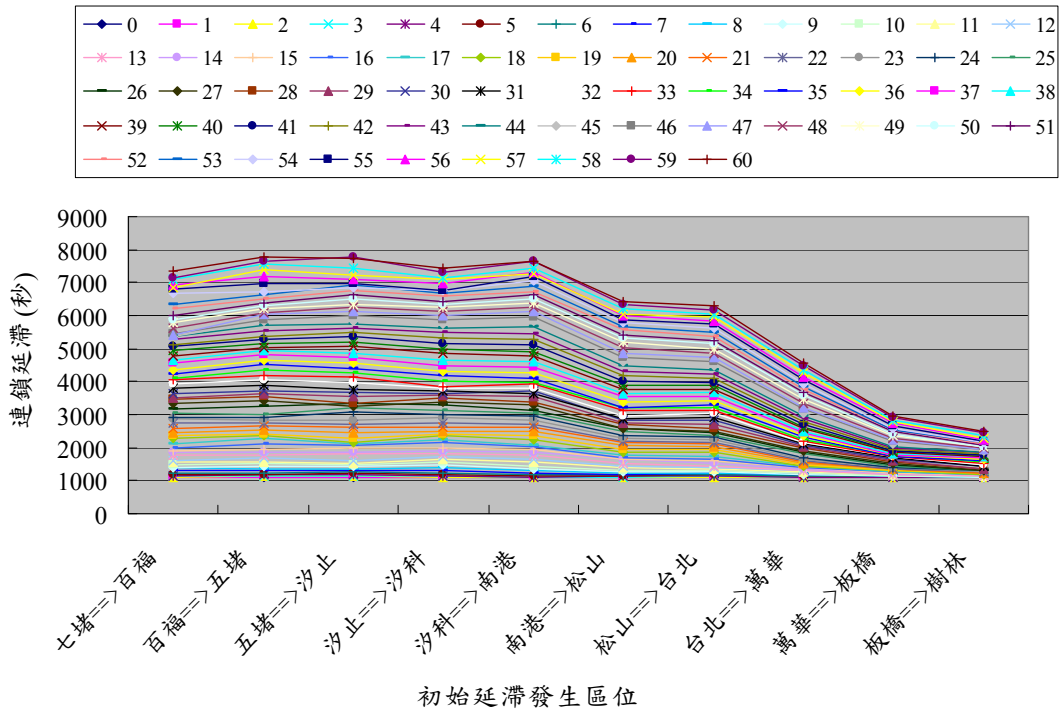


圖 5-9 同時採取二運轉調度策略下二端末車站之連鎖延滯降低

四、初始延滯發生區位、持續時間與二端末車站之連鎖延滯降低之關聯分析

圖 5-10 顯示初始延滯發生區位、持續時間，與同時採取站間趕點及站內趕點二運轉調度策略下二端末車站連鎖延滯降低之 3D 關係，其關係型態與圖 5-6 所有車站無運轉調度策略之情境結果類似，亦即顯示若初始延滯之持續時間較長且初始延滯發生區位靠近上游路段(例如七堵—南港段)，則同時採取二運轉調度策略後對連鎖延滯降低之程度就愈大，其亦顯示當初始延滯發生區位愈靠近上游路段，則其恢復至原訂班表之效果也愈好。

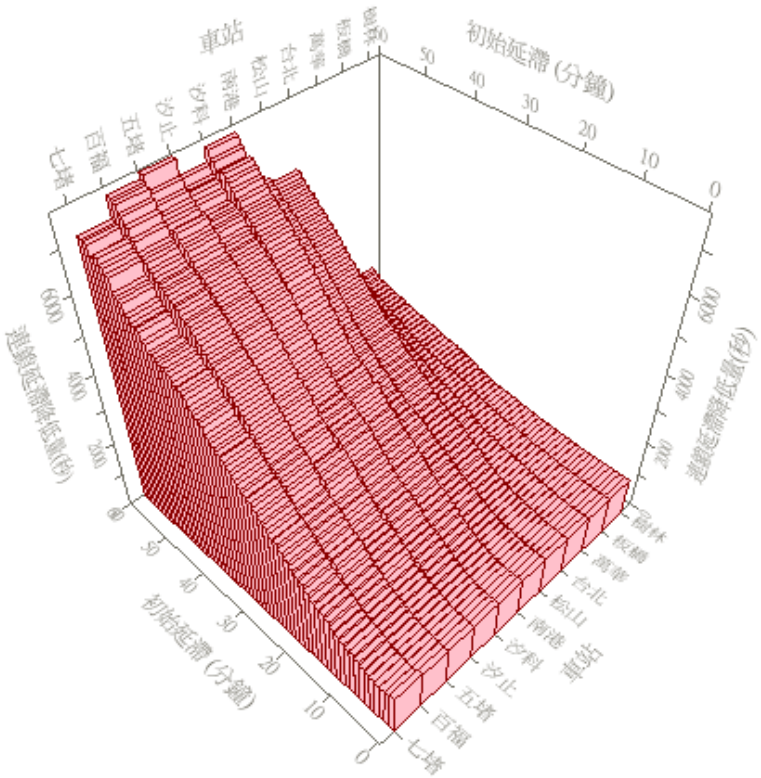


圖 5-10 初始延滯發生區位、持續時間與二端末車站連鎖延滯降低之 3D 關係

5.4 小結

本章主要目的即欲探索在臺鐵系統交通量最繁忙之北部七堵—樹林路段，若有初始延滯發生，其發生區位、持續時間及運轉調度策略對連鎖延滯降低之關聯及影響程度。運用本研究所構建之模擬模式工具，本章在各種不同軌道配置、交通及班表恢復之運轉調度控制條件組合情境下，分別分析及比較所有車站及二端末車站之連鎖延滯相關模擬分析結果，所分析之七堵—樹林路段共包括 11 個車站、256 車次、推拉式自強號(PP)及通勤電聯車(EMU)兩種類型列車，及 5 種車站股道配置型態。

由分析結果顯示，本研究所構建之模擬模式可合理模擬臺鐵系統複雜的連鎖延滯相關問題，及探索各種初始延滯情境對連鎖延滯之影響，本章案例分析亦得到許多豐碩結果，例如初始延滯發生之區位愈靠近上游路段，則所有車站及二端末車站之連鎖延滯亦將愈大；另當初始延滯發生區位愈靠近上游路段，則運轉調度策略對於連鎖延滯降低之程度就愈大，亦即其恢復至原訂班表之效果也愈好。上述分析結果，除可對於初始延滯與連鎖延滯之複雜交互作用關係有更深入之了解，亦可作為營運單位後續列車營運及運轉調度之參考。


E S A
1896

第六章 關鍵影響因素隨機特性之模擬分析

如前述研究流程所述，本研究所建構之模擬模式主要分為確定性(deterministic)及隨機性(stochastic)二階段，其中有關列車連鎖延滯與其影響因素(包括外生初始延滯地點、持續時間及不同運轉調度策略等)之關聯分析部分，因涉及運轉調度策略及各項軟、硬體條件交互作用複雜關係，為釐清其彼此間之因果關係，故較適合採用確定性模擬模式評估分析，前述各章已就其各項關鍵參數以給定班表及各項軟、硬體設施資料輸入進行模擬分析。惟由確定性模擬分析過程可發現，影響連鎖延滯結果之部分關鍵參數確具有隨機特性，故若欲深入探討有初始延滯發生後產生連鎖延滯之班表穩定度問題，則應較適合採用隨機性模擬模式進行評估分析，以真實反映列車之實際交互作用狀況。

為確實反映部分參數資料於實際營運之隨機特性，本章進一步針對列車延滯影響最鉅之站間運轉時間及停站時間等二項參數進行隨機特性分析，並將其隨機資料納入模擬模式推估連鎖延滯。由於臺鐵系統之列車種類甚多且特性各異，為利分析本章將其概分為對號列車及通勤電聯車(EMU)二類進行研究路段延滯資料之蒐集分析，再進行二類列車之各別站間運轉時間及停站時間的隨機資料分佈分析；而為使隨機參數之分佈資料具有一致性及代表性，資料處理過程中除先將特性異常樣本(outliers)先予去除外，亦針對分佈變異較大之停站時間資料再依尖、離峰進行分群後分別分析。

有關本章之隨機特性分析，茲分別就資料蒐集分析、延滯特性參數校估、延滯特性參數分析、去除異常事件日之延滯特性參數分佈分析及隨機特性連鎖延滯模擬結果等部分說明如下。

6.1 資料蒐集分析

為利進行隨機特性連鎖延滯模擬分析，首先必須蒐集臺鐵系統之列車延滯資料；惟依據交通部運輸研究所(民 100)之研究顯示，過去有關列車延滯相關分析的研究，大多是以終點站之列車平均延滯作為量測基準，少部份才會針對服務範圍內之車站逐站統計列車延滯。惟不同的時間範圍內，由於列車的密度、車種可能會不一樣，因此列車的延滯也會隨時間而異，但同時分析列車延滯的時空變化情形者並不多。而本章所用進行臺鐵列車延滯時空分佈分析的資料，是根據臺鐵中央行車控制系統（Centralized Traffic Control, CTC）所記錄的列車實際到離站資訊來計算，依 CTC 記錄的列車實際到離站情況，列車延滯量即為公告時刻表的表訂

到/離站時間與實際到/離站時間之差值。

有關本章所需之臺鐵系統列車延滯資料，資料來源為 2009 年 12 月 1 日到 2010 年 3 月 25 日期間臺鐵 CTC 所記錄的列車到離站延滯時間，至於空間範圍為臺鐵西部幹線北部區域之七堵站至新竹站，共計蒐集 18 個車站、17 個站區間資料(部分車站及站間資料缺漏)；而為利延滯分析，本章針對站間運轉時間及停站時間二項參數進行分析，共計蒐集 421,869 筆站間運轉時間及 372,327 筆停站時間資料，並將其與表訂班表時間比較即可得到列車延滯資料。

另為利區隔不同車種之運轉特性，本章亦概分對號列車及通勤電聯車(EMU)二類車種搭配站間運轉時間及停站時間，進行資料蒐集整理及後續之校估分析。

6.2 延滯特性參數校估分析

為利延滯特性參數之校估，應先了解延滯特性參數之觀測方式，茲參考交通部運輸研究所(民 100)之研究建議，有關其觀測方式在表定時刻表與實際到開資料均蒐集齊全的前提下，可能的觀測方式至少包括以下四種，茲說明其優缺點如下。

一、觀測「到站延滯」與「離站延滯」之分佈

此為最直接的觀測方式，因臺鐵系統既有公告時刻表之列車可能提早進站，故「到站延滯」的分佈可能存在負值；但就「離站延滯」的分佈而言，因列車不允許提早發車，故不存在延滯值小於零的狀況。惟此觀測方式最大的缺點，就是延滯值可能會累積先前上游車站的延滯，造成重複計算的問題，故此觀測方式不適用於本連鎖延滯推估模式。

二、觀測「實際站間運轉時間絕對值」與「實際停站時間絕對值」之分佈

此觀測方式雖可改善前一種觀測方式之上游車站延滯累積問題，亦即觀察實際列車的站間運轉時間變異及停站時間變異，蒐集系統的變異特性參數可供延滯分析模式使用，但分析過程需將樣本依列車種別、時段、區間進行適當的分類。特別是不同區間由於路線長度不同，路線較長的區間可能運轉時間變異較大，在沒有標準化的情況下，可能每個區間均需分析，另停站時間亦因有車站規模大小不一而需要標準化的問題存在，故此觀測方式對於本研究可能過於複雜不適用。

三、觀測「站間運轉時間實際值與表定值差值」與「停站時間實際值與表定值差值」之分佈

為了克服前一種觀測方式存在有不同區間長度不一的狀況，本觀測方式直接計算站間運轉時間的實際值與表定值之差，由於觀測值均減掉一個標準值（即表定站間運轉時間或表定停站時間），因此統計出來的參數就可以套用到各種不同長度的區間或套用在各種不同規模的車站。

四、觀測「站間運轉時間實際值與表定值比值」與「停站時間實際值與表定值比值」之分佈

此種觀測方式類似前一種，亦是要解決第二種觀測方式沒有依區間長度或車站規模來標準化的問題，只是本觀測方式採用的標準化方法並非用實際值減去表定值來處理，而是利用實際值除以表定值來呈現，對於本研究而言，此種觀測方式最為簡便，故本章後續之特性參數校估即運用此方法。

6.3 延滯特性參數分佈分析

依據上述參數校估方式，本章後續之延滯模擬分析，將分為對號列車及通勤電聯車(EMU)二車種，分別先彙整處理「站間運轉時間實際值與表定值比值」與「停站時間實際值與表定值比值」之累積分佈資料，以作為後續連鎖延滯模擬分析之參數資料輸入基礎。另本研究隨機特性分析之目的係為擷取隨機資料，而不在於探討校估各項隨機資料之分佈函數型態，故站間運轉時間及停站時間之隨機分佈資料主要係呈現實際值與表訂值比值累積機率佔大宗之 0%~250%資料(雖仍有少部分比值大於 250%)。相關分析茲分述如下。

一、對號列車之站間運轉時間分佈

此項資料為實際通過 17 個站間之對號列車共計 176,248 筆站間運轉時間資料，分別計算其與該站間運轉時間表定值之比值後並進行排序加總，再將各站間之「站間運轉時間實際值與表定值比值」作為 x 軸數值、其累計機率作為 y 軸數值，繪製其分佈情形如圖 6-1。任一隨機站間運轉時間值只要透過亂數種子產生之亂數，對應該站間運轉時間分佈曲線，即可透過該站間運轉時間表訂值還原得其隨機值。另由該圖各曲線分佈可知，由於各站間之運轉時間分佈尚稱一致且未有特別異常之趨勢，故為利後續可直接運用泛用站間運轉時間而無需由原站間曲線擷取，故本研究即以所有站間運轉時間樣本繪得之「總計」曲線，直接進行「站間運轉時間實際值與表定值比值」之分佈分析，其隨機資料之獲取亦如上述方式。

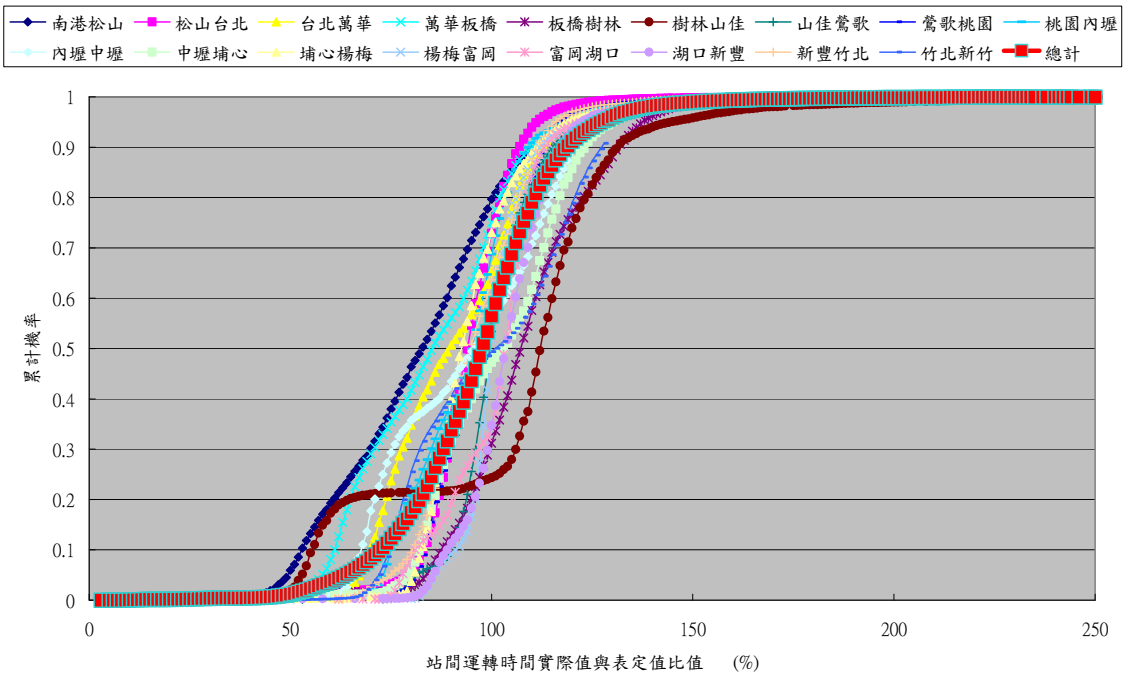


圖 6-1 對號列車之站間運轉時間分佈圖

二、通勤電聯車之站間運轉時間分佈

此項資料為實際通過 17 個站間之通勤電聯車共計 245,620 筆站間運轉時間資料，同樣分別計算其與該站間運轉時間表定值之比值後並進行排序加總，將各站間之「站間運轉時間實際值與表定值比值」作為 x 軸數值、其累計機率作為 y 軸數值，繪製其分佈如圖 6-2，再以亂數種子產生之亂數，即可透過該站間運轉時間之表訂值還原得其隨機值。同樣地，由於各站間之運轉時間分佈尚稱一致且未有特別異常之趨勢，故亦可以所有站間運轉時間樣本繪得之「總計」曲線，直接進行「站間運轉時間實際值與表定值比值」之分佈分析，以便泛用隨機站間運轉時間之擷取。

三、對號列車之停站時間分佈

此項資料為實際所有對號列車於 18 個車站共計 92,903 筆停站時間資料，分別計算各站所有列車每筆實際停站時間與表定值之比值後，再將各站之比值先排序後累加，並以「停站時間實際值與表定值比值」作為 x 軸數值、其累計機率作為 y 軸數值，即可分別繪製各站之停站時間分佈如圖 6-3。有關其各站停站時間之特性茲整理如下：

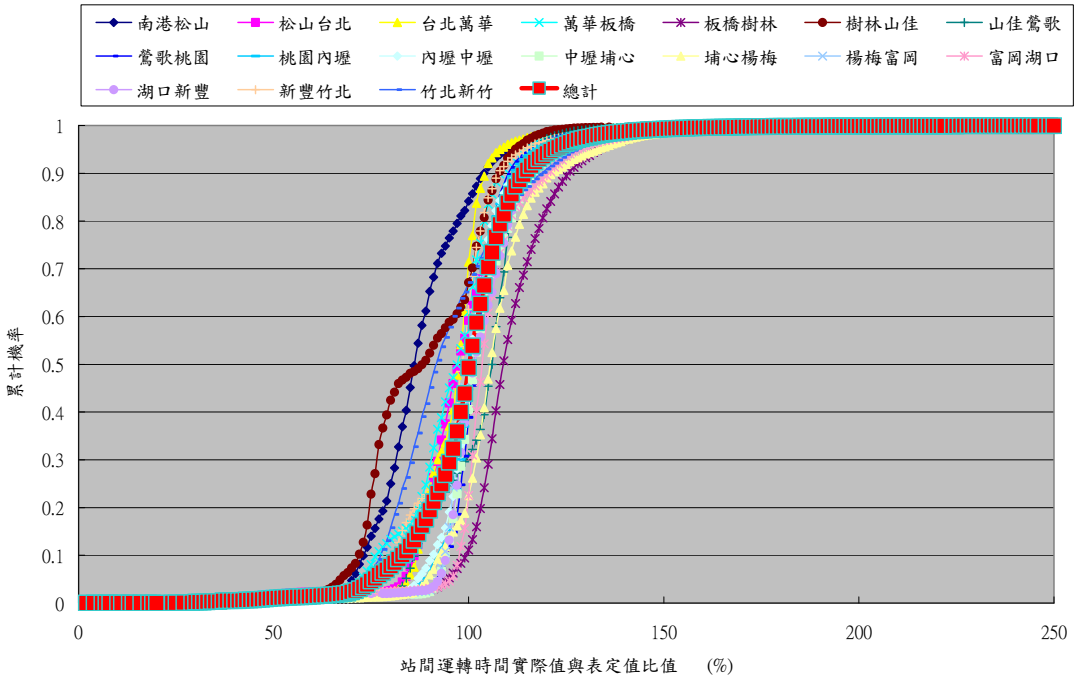


圖 6-2 通勤電聯車之站間運轉時間分佈圖

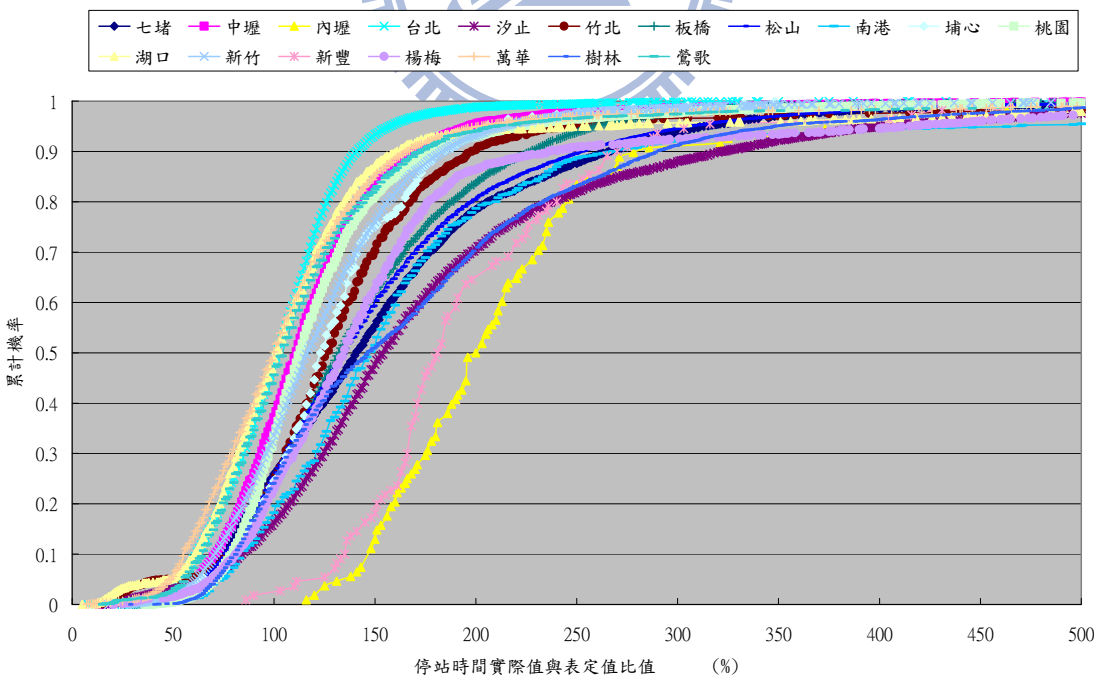


圖 6-3 對號列車之停站時間分佈圖

(一)比值分佈：有關各站停站時間之特性茲彙整如表 6.1，由所蒐集之資料顯示，各站除極少數約 0.045%樣本資料之比值超過 1000%特別異常外，大部分資料之比值約介於 200%~1000%之間，惟幾乎所有車站之停站時間樣本超過表定時間者皆大於 50%，且內壢、新豐、汐止…等共計 10 站之停站時間樣本超過表定時間者比例更超過 70%，而甚至內壢及新豐等二站所有的停站時間樣本皆超過表定時間，顯見所蒐集之臺鐵系統列車運轉資料必定存在許多干擾因素，以致停站時間資料出現許多異常結果，顯示資料仍需再分群過濾始能應用分析。

(二)可能干擾因素：如前述資料顯示，停站時間最為異常之內壢及新豐二站，其異常原因可能為前車延滯，導致後車必須於此二站停車交會待避；至於其他大部分之車站停站時間有相當比例是原表定停站時間之 2~5 倍，亦顯示其可能係因延滯(含外生之初始延滯及衍生之連鎖延滯)所導致之停站時間增加。另由表 6.1 亦發現有部分低於表定停站時間之趕點樣本，顯示實際列車營運為了降低列車延滯之影響，部分列車必須透過減少停站時間進行趕點。

表 6.1 對號列車之停站時間分佈特性彙整

車站	比值範圍(%)	資料筆數	準點樣本數	超過表定值之樣本數	趕點樣本數(未達表定值)	待趕點樣本數比例
七堵	34-2148	4103	1	3070	1032	74.82%
汐止	15-1631	5791	38	4845	908	83.66%
南港	45-1211	750	8	613	129	81.73%
松山	15-2050	13543	136	9994	3413	73.79%
臺北	8-1043	13584	194	7083	6307	52.14%
萬華	7-2693	3512	58	1749	1705	49.80%
板橋	24-2730	13799	130	10408	3261	75.43%
樹林	28-2554	4343	43	3254	1046	74.93%
鶯歌	15-4750	2007	35	1091	881	54.36%
桃園	36-4673	8600	161	5737	2702	66.71%
內壢	116-573	108	0	108	0	100.00%
中壢	13-1891	7997	105	5027	2865	62.86%
埔心	50-348	224	3	169	52	75.45%
楊梅	20-4676	2359	24	1817	518	77.02%
湖口	5-1115	2533	53	1292	1188	51.01%
新豐	86-738	110	0	108	2	98.18%
竹北	15-975	1375	14	1035	326	75.27%
新竹	18-3316	8165	100	5425	2640	66.44%

四、通勤電聯車之停站時間分佈

此項資料為實際所有通勤電聯車於 20 個車站共計 279,424 筆停站時間資料，分別計算各站所有列車每筆實際停站時間與表定值之比值後，再分別將各站之比值先排序後累加，並以「停站時間實際值與表定值比值」作為 x 軸數值、其累計機率作為 y 軸數值，即可分別繪製各站之停站時間分佈如圖 6-4。有關其各站停站時間之特性茲整理如下：

(一)比值分佈：有關各站之停站時間特性茲彙整如表 6.2，由所蒐集之資料顯示，大部分資料之比值約介於 100%~700%之間，但和對號列車停站時間資料比較，各站比值超過 1000%特別異常之樣本比例大幅提升至約 0.42%，顯示停站時間特別異常之情況更普遍。至於停站時間超過表定時間之樣本部分，和對號列車停站時間資料比較其比例降低許多，尤其像松山、臺北、萬華、中壢等通勤旅次量較大之車站，其停站時間超過表定時間比例皆小於 50%；但整體而言，所蒐集之通勤電聯車停站時間資料仍存在許多干擾因素，以致停站時間資料出現許多異常結果，資料仍需再分群過濾始能應用分析。

(二)可能干擾因素：如前述資料顯示，停站時間異常之原因可能係因通勤電聯車之行車優先順序低於對號列車，故除前車延滯所導致之後車停車交會待避外，其最大之干擾因素可能仍是因初始延滯所衍生之連鎖延滯及延滯擴散所導致之停站時間增加，尤其更可合理推測比值超過 1000%特別異常之停站時間是連鎖延滯擴散所造成，應更深入分析。至於表 6.2 中所顯示超過表定停站時間之待趕點樣本數比例部分，其停站時間待趕點比例和對號列車資料相較提高許多，顯示實際列車營運為了降低列車延滯之影響，通勤電聯車可能更需透過減少停站時間進行趕點。

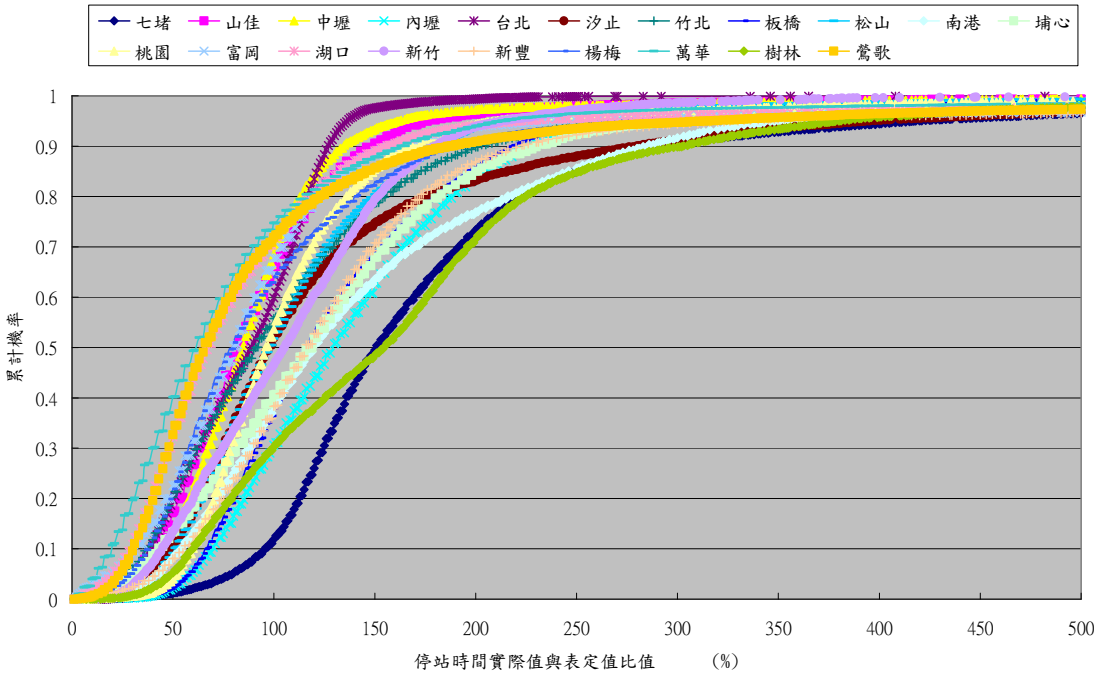


圖 6-4 通勤電聯車之停站時間分佈圖

表 6.2 通勤電聯車之停站時間分佈特性彙整

車站	比值範圍 (%)	資料筆數	準點樣本數	超過表定值之樣本數	趕點樣本數 (未達表定值)	待趕點樣本數比例
七堵	10-3486	15656	128	13794	1734	88.11%
汐止	1-2730	16702	223	8045	8434	48.17%
南港	2-5410	16537	208	10222	6107	61.81%
松山	4-6193	17028	224	8208	8596	48.20%
臺北	11-12170	17351	185	6961	10205	40.12%
萬華	1-7770	16770	237	4239	12294	25.28%
板橋	3-3793	17427	251	10738	6438	61.62%
樹林	7-28436	14078	102	9816	4160	69.73%
山佳	1-3080	14028	392	4760	8876	33.93%
鶯歌	1-6913	14249	139	4007	10103	28.12%
桃園	8-5016	14299	223	6622	7454	46.31%
內壢	3-4003	14186	288	9783	4115	68.96%
中壢	1-2505	12836	237	4057	8542	31.61%
埔心	1-6476	12866	221	7645	5000	59.42%
楊梅	1-4096	12486	131	4498	7857	36.02%
富岡	1-2223	12394	296	3426	8672	27.64%
湖口	1-6976	12066	161	3414	8491	28.29%
新豐	3-3113	11994	292	7399	4303	61.69%
竹北	2-2863	12000	192	5312	6496	44.27%
新竹	11-870	4471	31	2389	2051	53.43%

6.4 去除異常事件日之延滯特性參數分析

根據前述有關站間運轉時間與停站時間之延滯特性參數分佈結果，無論是對號列車或通勤電聯車之分佈資料，均顯示有相當比例異常樣本，尤其特別異常樣本資料亦不少。鑑於列車營運所產生之延滯皆可能擴散導致大量班次之站間運轉時間與停站時間異常，故為利探討列車在一般營運情況下的延滯狀況，本節將檢視 2009 年 12 月 1 日到 2010 年 3 月 25 日近四個月期間是否有特殊異常事件日。

經檢視案例資料期間，發現新年連續假期（12 月 31 日到 1 月 3 日）、春節連續假期（2 月 12 到 21 日），以及當時於 3 月 4 日和 8 日兩天國內發生規模較大之地震，故共計 16 天的異常營運資料將去除不列入分析。另由於各站之停站時間分佈型態較為異常且各站之差鉅頗大，為利後續連鎖延滯分析，本節再將停站時間資料區分為尖、離峰進行分析，其中尖峰時段係定義為上午 6~9 點及下午 4~7 點，其餘則為離峰時段。有關去除異常事件日後之各分群資料茲分析如下：

一、對號列車之站間運轉時間分佈

此項資料為實際通過 17 個站間之對號列車共計 149,231 筆站間運轉時間資料，分別計算其與該站間運轉時間表定值之比值後並進行排序加總，再分別將各站間之「站間運轉時間實際值與表定值比值」作為 x 軸數值、其累計機率作為 y 軸數值，繪製其分佈如圖 6-5，另如同前節作法，由於各站間之運轉時間分佈尚稱一致且未有特別異常之趨勢，故本研究亦以所有站間運轉時間樣本繪得之「總計」曲線，直接進行「站間運轉時間實際值與表定值比值」之分佈分析，其隨機資料之擷取亦如前述方式。

二、通勤電聯車之站間運轉時間分佈

此項資料為實際通過 17 個站間之通勤電聯車共計 211,778 筆站間運轉時間資料，同樣分別計算其與該站間運轉時間表定值之比值後並進行排序加總，再分別將各站間之「站間運轉時間實際值與表定值比值」作為 x 軸數值、其累計機率作為 y 軸數值，繪製其分佈如圖 6-6，另如同前節作法，由於各站間之運轉時間分佈尚稱一致且未有特別異常之趨勢，故本研究亦以所有站間運轉時間樣本繪得之「總計」曲線，直接進行「站間運轉時間實際值與表定值比值」之分佈分析，其隨機資料之擷取亦如前述方式。

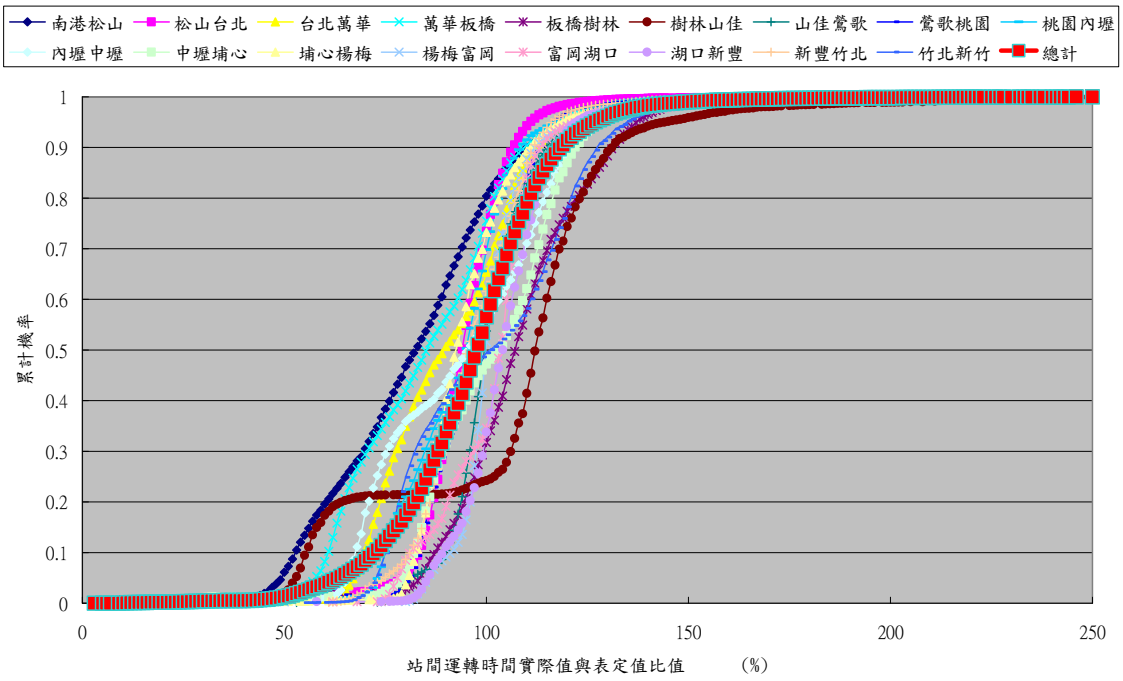


圖 6-5 對號列車之站間運轉時間分佈圖(去除異常日樣本)

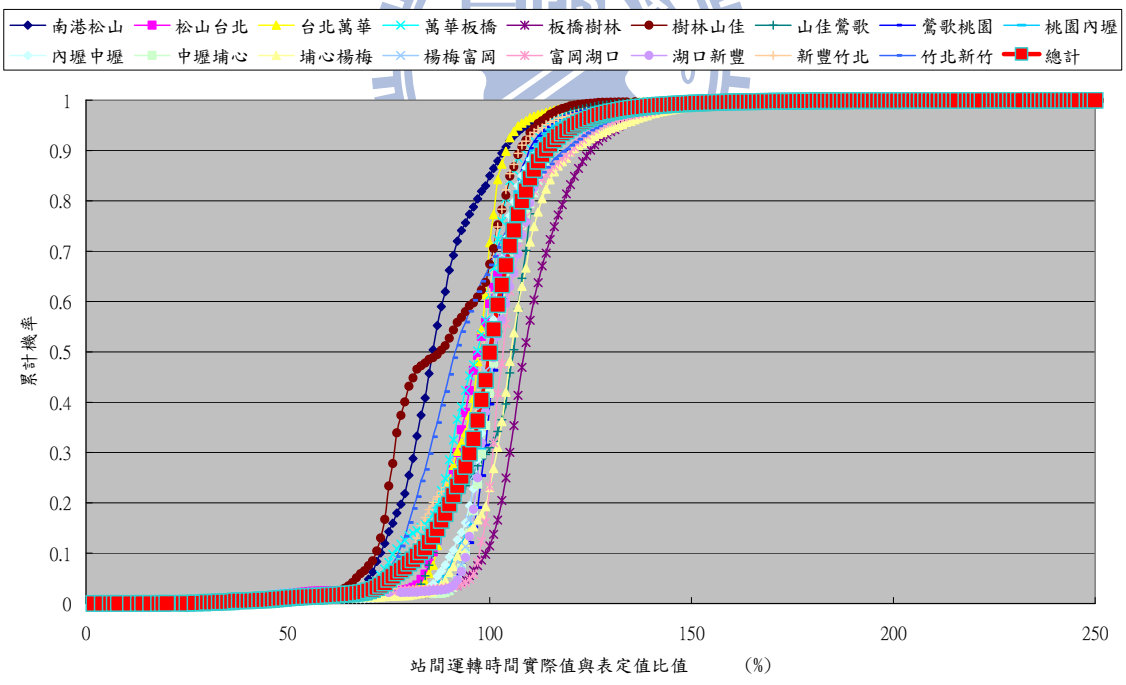


圖 6-6 通勤電聯車之站間運轉時間分佈圖(去除異常日樣本)

三、對號列車於尖峰時段之停站時間分佈

此項資料為實際所有對號列車於18個車站之尖峰時段共計26,983筆停站時間資料，分別計算各站所有列車每筆實際停站時間與表定值之比值後，再分別將各站之比值先排序後累加，並以「停站時間實際值與表定值比值」作為 x 軸數值、其累計機率作為 y 軸數值，分別繪製各站之停站時間分佈如圖 6-7。

有關各站於尖峰時段停站時間之特性顯示於各站之比值分佈情形，其各站停站時間之特性茲彙整如表 6.3，因內壢及埔心站於尖峰時段無對號列車停靠，故僅剩 16 個車站資料。由所蒐集之資料顯示，經去除異常事件日資料後，各站樣本資料之比值超過 1000%特別異常之比例更降低至約 0.037%，大部分資料之比值更集中約介於 50%~200%之間，若以比值介於 80%~120%資料為例，其比例即達約 39%，顯見分群後之對號列車於尖峰時段之停站時間分佈已更趨一致，應更能作為進一步分析連鎖延滯之基礎。

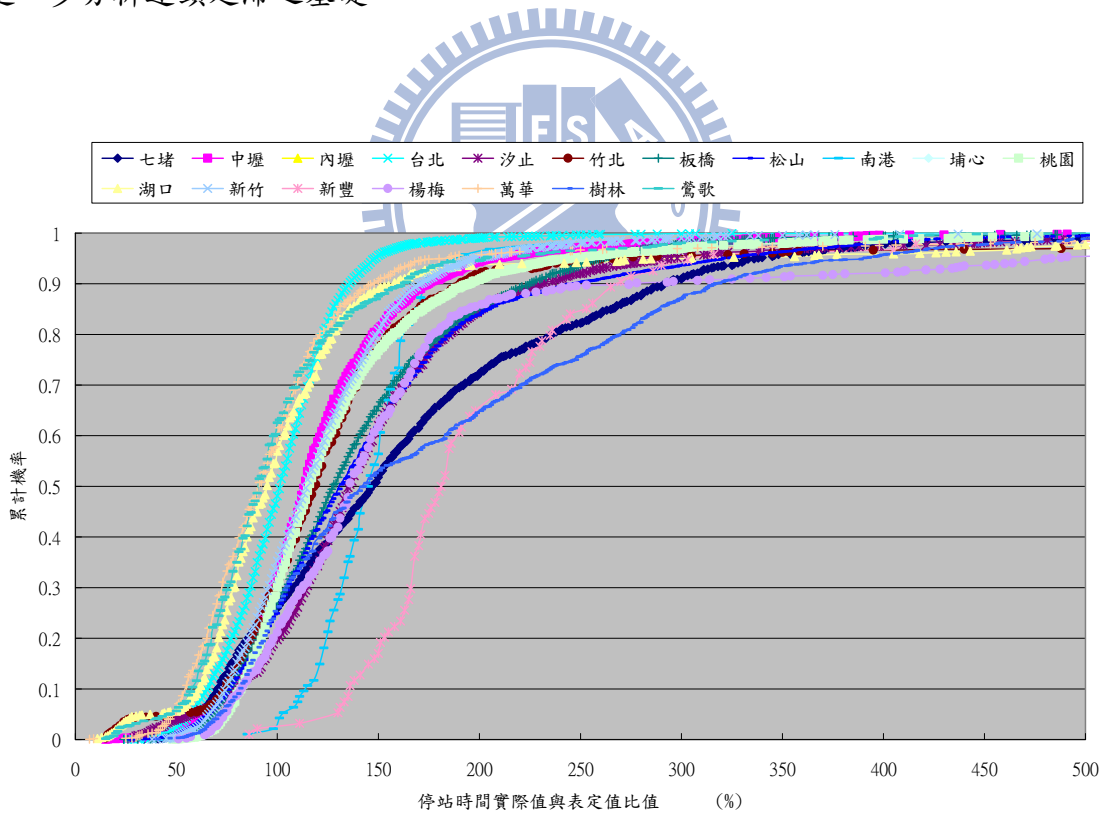


圖 6-7 對號列車於尖峰時段之停站時間分佈圖(去除異常日樣本)

表 6.3 對號列車於尖峰時段之停站時間分佈特性彙整

車站	比值範圍(%)	資料筆數	準點樣本數	超過表定值之樣本數	趕點樣本數(未達表定值)	待趕點樣本數比例
七堵	34-2148	1392	8	1031	353	74.07%
汐止	29-1091	1415	12	1136	267	80.28%
南港	85-576	94	0	92	2	97.87%
松山	26-651	3619	44	2673	902	73.86%
臺北	26-1043	3929	61	2039	1829	51.90%
萬華	7-713	1145	23	460	662	40.17%
板橋	24-2730	3937	36	2878	1023	73.10%
樹林	40-2554	1225	12	903	310	73.71%
鶯歌	15-683	569	13	213	343	37.43%
桃園	48-4673	2564	46	1793	725	69.93%
內壢	--	--	--	--	--	--
中壢	15-723	2260	30	1538	692	68.05%
埔心	--	--	--	--	--	--
楊梅	51-990	661	8	521	132	78.82%
湖口	11-1050	927	22	399	506	43.04%
新豐	86-738	94	0	92	2	97.87%
竹北	15-975	657	7	456	194	69.41%
新竹	38-784	2495	37	1616	842	64.77%

四、對號列車於離峰時段之停站時間分佈

此項資料為實際所有對號列車於 18 個車站之離峰時段共計 51,680 筆停站時間資料，分別計算各站所有列車每筆實際停站時間與表定值之比值後，再分別將各站之比值先排序後累加，並以「停站時間實際值與表定值比值」作為 x 軸數值、其累計機率作為 y 軸數值，分別繪製各站之停站時間分佈如圖 6-8。

有關其各站於離峰時段停站時間之特性顯示於各站之比值分佈情形，其各站停站時間之特性彙整如表 6.4，因新豐站於離峰時段無對號列車停靠，故僅剩 17 個車站資料。由所蒐集之資料顯示，經去除異常事件日資料後，各站樣本資料之比值超過 1000%特別異常之比例更降低至約 0.033%，大部分資料之比值更集中約介於 50%~200%之間，若以比值介於 80%~120%資料為例，其比例即達約 36%，顯見分群後之對號列車於離峰時段之停站時間分佈資料已更趨一致，亦更能作為進一步分析連鎖延滯之基礎。

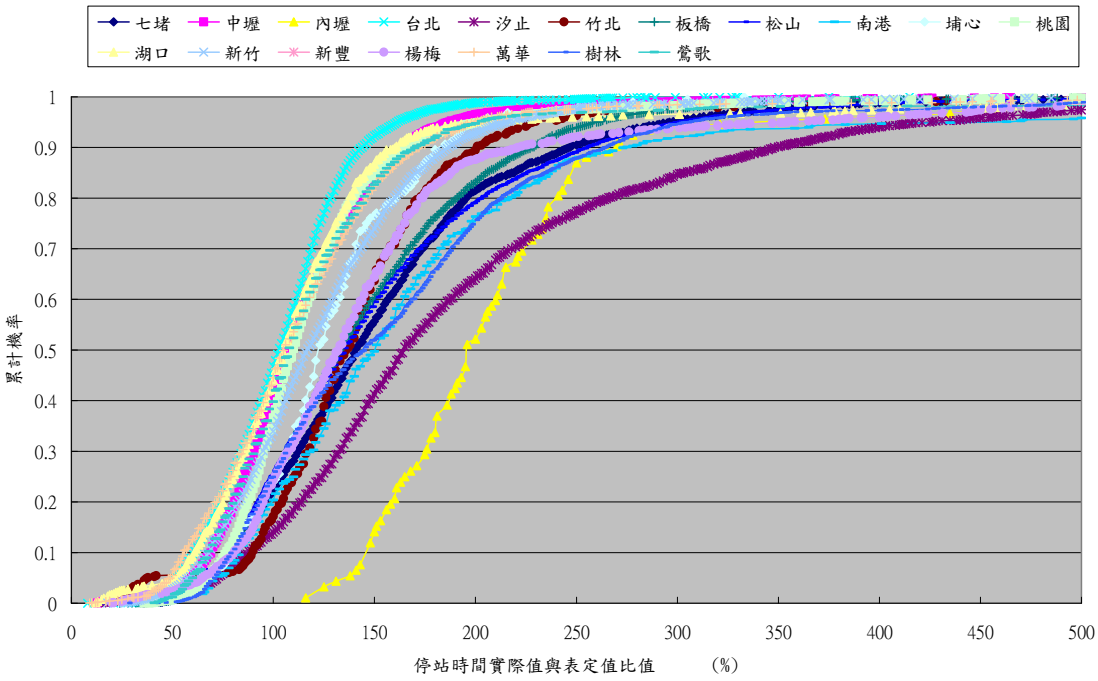


圖 6-8 對號列車於離峰時段之停站時間分佈圖(去除異常日樣本)

表 6.4 對號列車於離峰時段之停站時間分佈特性彙整

車站	比值範圍(%)	資料筆數	準點樣本數	超過表定值之樣本數	趕點樣本數(未達表定值)	待趕點樣本數比例
七堵	36-990	2129	18	1648	463	77.41%
汐止	15-1083	3529	14	3035	480	86.00%
南港	45-1211	545	5	434	106	79.63%
松山	15-2050	7784	71	5743	1970	73.78%
臺北	8-415	7508	101	3995	3412	53.21%
萬華	9-2693	1853	29	1027	797	55.42%
板橋	24-1368	7695	74	5827	1794	75.72%
樹林	28-831	2480	26	1836	618	74.03%
鶯歌	41-4750	1140	17	686	437	60.18%
桃園	36-1981	4736	101	3002	1633	63.39%
內壢	116-573	92	0	92	0	100.00%
中壢	13-1434	4512	54	2670	1788	59.18%
埔心	50-333	192	1	146	45	76.04%
楊梅	20-4676	1339	14	1013	312	75.65%
湖口	12-778	1217	22	681	514	55.96%
新豐	--	--	--	--	--	--
竹北	23-943	512	4	424	84	82.81%
新竹	18-3316	4417	45	2918	1454	66.06%

五、通勤電聯車於尖峰時段之停站時間分佈

此項資料為實際所有通勤電聯車於20個車站之尖峰時段共計91,764筆停站時間資料，分別計算各站所有列車每筆實際停站時間與表定值之比值後，再分別將各站之比值先排序後累加，並以「停站時間實際值與表定值比值」作為x軸數值、其累計機率作為y軸數值，分別繪製各站之停站時間分佈如圖6-9。

有關其各站於尖峰時段停站時間之特性顯示於各站之比值分佈情形，其各站停站時間之各項特性彙整如表6.5。由所蒐集之資料顯示，經去除異常事件日資料後，各站樣本資料之比值超過 1000%特別異常之比例已由原先之 0.42%降低至約0.37%，大部分資料之比值更集中約介於20%~250%之間，若以比值介於80%~120%資料為例，其比例即達約 30%，顯見分群後之通勤電聯車於尖峰時段之停站時間分佈資料已更趨一致，應更能作為進一步分析連鎖延滯之基礎。

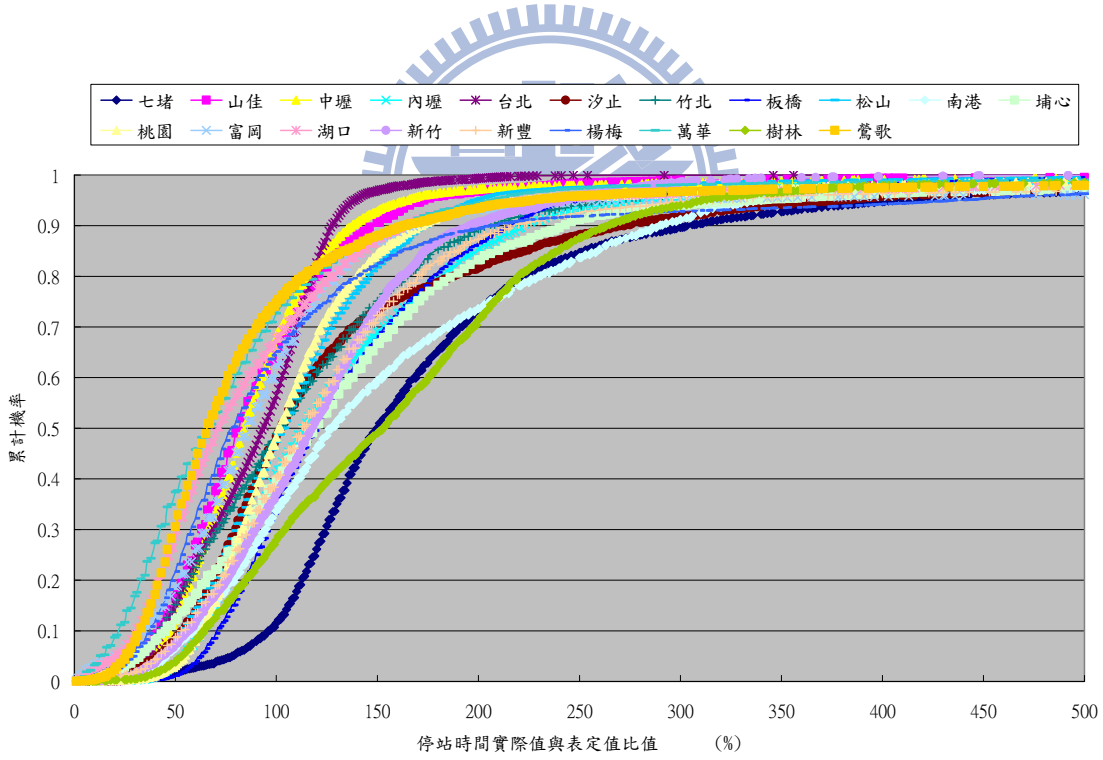


圖 6-9 通勤電聯車於尖峰時段之停站時間分佈圖(去除異常日樣本)

表 6.5 通勤電聯車於尖峰時段之停站時間分佈特性彙整

車站	比值範圍(%)	資料筆數	準點樣本數	超過表定值之樣本數	趕點樣本數(未達表定值)	待趕點樣本數比例
七堵	10-3486	5215	43	4613	559	88.46%
汐止	5-2730	5299	74	2734	2491	51.59%
南港	4-4023	5360	69	3559	1732	66.40%
松山	15-3836	5433	84	2860	2489	52.64%
臺北	15-938	5456	61	2376	3019	43.55%
萬華	1-6410	5596	96	1529	3971	27.32%
板橋	23-3793	5868	99	3824	1945	65.17%
樹林	7-2266	4652	39	3354	1259	72.10%
山佳	3-3080	4664	110	1586	2968	34.01%
鶯歌	1-4416	4828	33	1220	3575	25.27%
桃園	10-5016	4842	75	2506	2261	51.76%
內壢	9-4003	4612	75	2724	1813	59.06%
中壢	5-2141	4311	66	1387	2858	32.17%
埔心	1-2663	4241	65	2559	1617	60.34%
楊梅	1-1843	4188	31	1460	2697	34.86%
富岡	1-1736	4106	115	1536	2455	37.41%
湖口	1-1490	3962	55	1278	2629	32.26%
新豐	6-2466	3757	77	2239	1441	59.60%
竹北	3-1503	3664	62	1870	1732	51.04%
新竹	11-511	1710	18	1076	616	62.92%

六、通勤電聯車於離峰時段之停站時間分佈

此項資料為實際所有通勤電聯車於 20 個車站之離峰時段共計 148,892 筆停站時間資料，分別計算各站所有列車每筆實際停站時間與表定值之比值後，再分別將各站之比值先排序後累加，並以「停站時間實際值與表定值比值」作為 x 軸數值、其累計機率作為 y 軸數值，分別繪製各站之停站時間分佈如圖 6-10。

有關其各站於離峰時段停站時間之特性顯示於各站之比值分佈情形，其各站停站時間之各項特性彙整如表 6.6。由所蒐集之資料顯示，經去除異常事件日資料後，各站樣本資料之比值超過 1000%特別異常之比例已由原先之 0.42%降低至約 0.38%，大部分資料之比值更集中約介於 20%~280%之間，若以比值介於 80%~120%資料為例，其比例即達約 28%，顯見分群後之通勤電聯車於離峰時段之停站時間分佈資料已更趨一致，應更能作為進一步分析連鎖延滯之基礎。

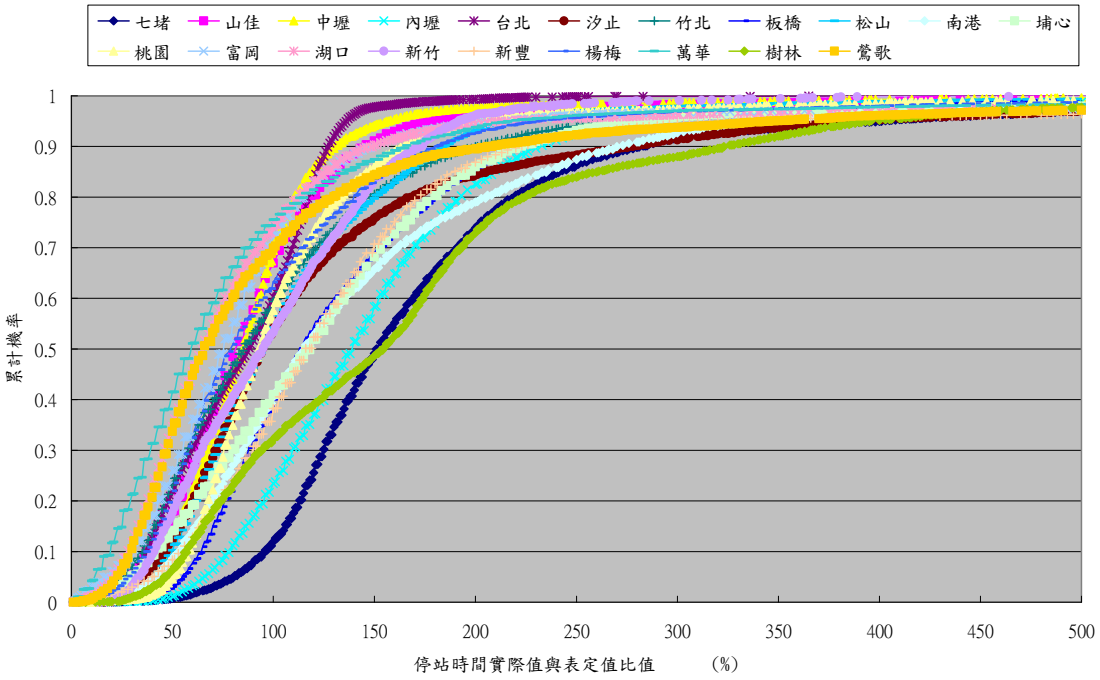


圖 6-10 通勤電聯車於離峰時段之停站時間分佈圖(去除異常日樣本)

表 6.6 通勤電聯車於離峰時段之停站時間分佈特性彙整

車站	比值範圍(%)	資料筆數	準點樣本數	超過表定值之樣本數	趕點樣本數(未達表定值)	待趕點樣本數比例
七堵	16-3343	8295	69	7303	923	88.04%
汐止	1-2603	9085	116	4173	4796	45.93%
南港	2-2750	8855	116	5265	3474	59.46%
松山	8-2646	9191	101	4209	4881	45.79%
臺北	14-12170	9469	105	3760	5604	39.71%
萬華	1-7700	8849	105	2165	6579	24.47%
板橋	3-2905	9127	113	5469	3545	59.92%
樹林	7-28436	7484	50	5077	2357	67.84%
山佳	1-2903	7425	226	2453	4746	33.04%
鶯歌	1-6913	7463	83	2241	5139	30.03%
桃園	9-3275	7490	121	3174	4195	42.38%
內壢	3-2433	7608	132	5824	1652	76.55%
中壢	1-2505	6757	142	2063	4552	30.53%
埔心	1-1733	6846	122	4040	2684	59.01%
楊梅	2-4096	6559	81	2416	4062	36.83%
富岡	1-2050	6576	143	1811	4622	27.54%
湖口	1-2053	6442	85	1661	4696	25.78%
新豐	3-3113	6579	164	4065	2350	61.79%
竹北	2-1813	6665	112	2708	3845	40.63%
新竹	13-691	2127	10	993	1124	46.69%

6.5 隨機特性連鎖延滯模擬結果

經過前述延滯特性參數分佈分析結果得知，本研究將以 6.4 節去除 16 天異常事件日之延滯特性分佈資料進行後續連鎖延滯模擬分析，並依據站間運轉時間與停站時間之關鍵延滯參數分佈特性，依對號列車及通勤電聯車等二種車種、及尖、離峰等二種時段進行區分，分別針對研究案例臺鐵西部幹線七堵站至新竹站區間「所有車站」及「二端末車站」之連鎖延滯進行分析。

有關前述連鎖延滯分析之情境假設包括：尖峰時段定義為每日上午 6~9 點及下午 4~7 點，其餘則為離峰時段；另尖峰情境之連鎖延滯分析係假設於上午 7:15 於南下松山—臺北站間有 1 小時之初始延滯，而離峰情境之連鎖延滯分析則假設於上午 10:15 於南下松山—臺北站間有 1 小時之初始延滯。

至於隨機站間運轉時間之擷取係依據前述 6.4 節不同車種之站間運轉時間分佈「總計」曲線，依產生之亂數對應而得；而隨機停站時間之擷取作法類似，但需依不同車種、尖離峰時段及不同車站依產生之亂數對應而得。有關各情境之模擬分析結果茲分述如下：

一、所有車站之連鎖延滯模擬結果

本案例分析是依據 6.4 節之各項隨機資料，及班表列車順序與前述各項情境假設，分別依據上午 7:15(尖峰)及 10:15(離峰)於南下松山—臺北站間有 1 小時之初始延滯假設，模擬得到產生連鎖延滯之模擬班表，最後再與表定班表各列車時間比較，即可得到所有車站之連鎖延滯模擬結果。另由於隨機模擬結果具有擾動震盪之特性，故本研究由系統亂數種子所產生的 1,000 組亂數進行連鎖延滯模擬推估，並透過連鎖延滯推估量逐次累計再取單次模擬平均值之作法，嘗試尋找連鎖延滯之模擬穩定值，經彙整七堵—新竹區間於尖、離峰時段有 1 小時初始延滯所有 20 個車站之連鎖延滯模擬結果如表 6.7 所示。

由彙整結果可發現，當南下松山—臺北站間於尖峰時段有 1 小時之初始延滯發生時，其所有 20 個車站之連鎖延滯值約經過 100 次模擬後，平均值即會穩定接近至約 173 小時。另當同樣南下松山—臺北站間於離峰時段有 1 小時之初始延滯發生時，其所有 20 個車站之連鎖延滯值約經過 123 次模擬後，平均值即會穩定接近至約 153 小時，其平均連鎖延滯模擬值與模擬次數關係如圖 6-11 所示；而由圖中模擬值亦可發現，尖峰之連鎖延滯模擬值始終大於離峰之連鎖延滯模擬值，可見模擬結果亦符合先驗知識。

表 6.7 所有車站之連鎖延滯模擬結果彙整表

亂數序號	尖峰有 1 小時之初始延滯發生			離峰有 1 小時之初始延滯發生		
	單次連鎖延滯模擬值(秒)	累計連鎖延滯模擬值(秒)	平均連鎖延滯模擬值(小時)	單次連鎖延滯模擬值(秒)	累計連鎖延滯模擬值(秒)	平均連鎖延滯模擬值(小時)
1	604,531	604,531	167.93	555,356	555,356	154.27
2	572,088	1,176,619	163.42	536,438	1,091,794	151.64
3	647,629	1,824,248	168.91	524,907	1,616,701	149.69
4	791,307	2,615,555	181.64	511,670	2,128,371	147.80
5	691,081	3,306,636	183.70	542,019	2,670,390	148.36
6	753,869	4,060,505	187.99	533,196	3,203,586	148.31
7	639,110	4,699,615	186.49	557,899	3,761,485	149.27
8	692,964	5,392,579	187.24	483,401	4,244,886	147.39
9	644,109	6,036,688	186.32	553,135	4,798,021	148.09
10	625,210	6,661,898	185.05	609,879	5,407,900	150.22
11	627,382	7,289,280	184.07	578,774	5,986,674	151.18
12	636,482	7,925,762	183.47	538,099	6,524,773	151.04
13	566,826	8,492,588	181.47	503,787	7,028,560	150.18
14	716,185	9,208,773	182.71	516,348	7,544,908	149.70
15	591,704	9,800,477	181.49	465,384	8,010,292	148.34
16	581,235	10,381,712	180.24	499,451	8,509,743	147.74
17	625,623	11,007,335	179.86	570,904	9,080,647	148.38
18	645,830	11,653,165	179.83	564,326	9,644,973	148.84
19	714,484	12,367,649	180.81	593,286	10,238,259	149.68
20	605,169	12,972,818	180.18	526,728	10,764,987	149.51
21	606,974	13,579,792	179.63	535,481	11,300,468	149.48
22	546,440	14,126,232	178.36	549,220	11,849,688	149.62
23	556,002	14,682,234	177.32	650,680	12,500,368	150.97
24	628,573	15,310,807	177.21	618,887	13,119,255	151.84
25	665,531	15,976,338	177.51	545,183	13,664,438	151.83
26	769,760	16,746,098	178.91	611,757	14,276,195	152.52
27	563,842	17,309,940	178.09	655,443	14,931,638	153.62
28	616,882	17,926,822	177.85	478,531	15,410,169	152.88
29	836,087	18,762,909	179.72	556,469	15,966,638	152.94
…	…	…	…	…	…	…
998	587,418	626,681,340	174.43	503,660	555,116,281	154.51
999	549,482	627,230,822	174.41	455,225	555,571,506	154.48
1000	676,983	627,907,805	174.42	494,713	556,066,219	154.46

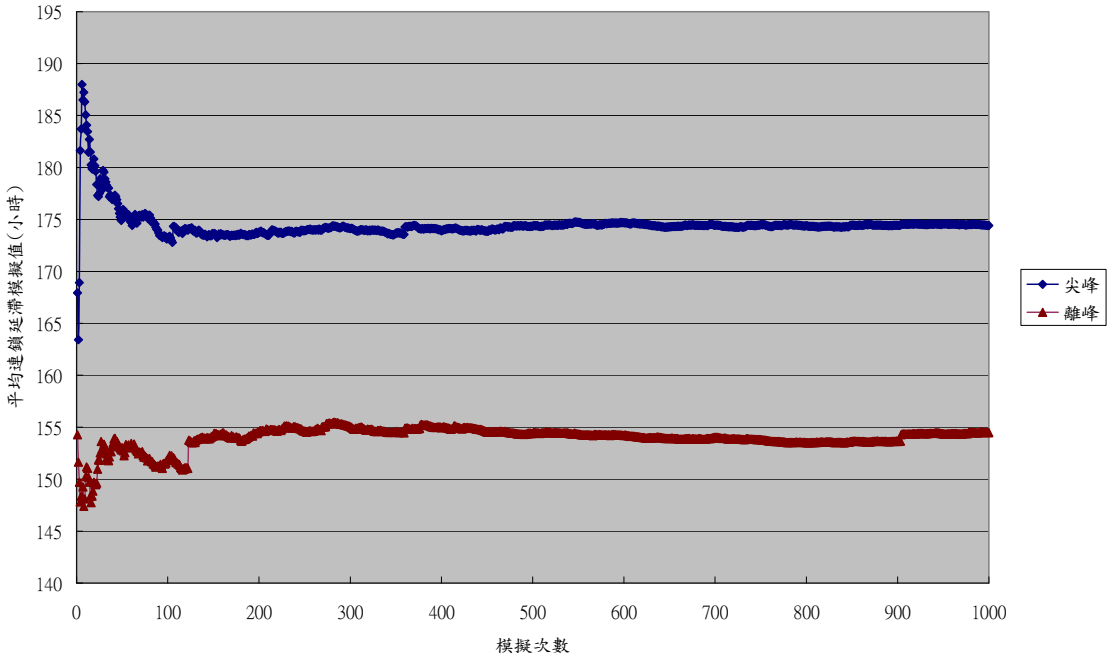


圖 6-11 所有車站之平均連鎖延滯模擬值與模擬次數關係

二、分析路段二端末車站之連鎖延滯模擬結果

有關二端末車站之連鎖延滯模擬作法，如同前述所有車站之作法及各項情境假設，分別依據上午 7:15(尖峰)及 10:15(離峰)於南下松山—臺北站間有 1 小時之初始延滯假設，模擬得到產生連鎖延滯之模擬班表，最後針對分析路段七堵—新竹路段二端末車站之各列車時間與表定班表各列車時間比較，即可得到二端末車站之連鎖延滯模擬結果。另本分析亦由系統亂數種子所產生的 1,000 組亂數進行連鎖延滯模擬推估，並透過連鎖延滯推估量逐次累計再取單次模擬平均值之作法，可得到連鎖延滯之模擬穩定值，經彙整七堵—新竹區間於尖、離峰時段有 1 小時初始延滯於七堵及新竹二端末車站之合計連鎖延滯模擬結果如表 6.8 所示。

由彙整結果可發現，當南下松山—臺北站間於尖峰有 1 小時之初始延滯發生時，其七堵及新竹二端末車站之連鎖延滯值約經過 108 次模擬後，平均值即會穩定接近至約 32.7 小時。另當同樣南下松山—臺北站間於離峰有 1 小時之初始延滯發生時，其七堵及新竹二端末車站之連鎖延滯值約經過 175 次模擬後，平均值即會穩定接近至約 28.5 小時，其平均連鎖延滯模擬值與模擬次數關係如圖 6-12 所示；而由圖中模擬值亦可發現，尖峰之連鎖延滯模擬值始終大於離峰之連鎖延滯模擬值，其模擬結果係符合先驗知識。

表 6.8 七堵及新竹二端末車站之連鎖延滯模擬結果彙整表

亂數序號	尖峰有 1 小時之初始延滯發生			離峰有 1 小時之初始延滯發生		
	單次連鎖延滯模擬值(秒)	累計連鎖延滯模擬值(秒)	平均連鎖延滯模擬值(小時)	單次連鎖延滯模擬值(秒)	累計連鎖延滯模擬值(秒)	平均連鎖延滯模擬值(小時)
1	111,927	111,927	31.09	99,290	99,290	27.58
2	114,586	226,513	31.46	93,656	192,946	26.80
3	119,953	346,466	32.08	93,322	286,268	26.51
4	154,364	500,830	34.78	92,778	379,046	26.32
5	129,098	629,928	35.00	104,759	483,805	26.88
6	144,384	774,312	35.85	106,140	589,945	27.31
7	128,550	902,862	35.83	105,962	695,907	27.62
8	137,709	1,040,571	36.13	93,652	789,559	27.42
9	124,763	1,165,334	35.97	98,842	888,401	27.42
10	118,927	1,284,261	35.67	103,944	992,345	27.57
11	119,984	1,404,245	35.46	100,985	1,093,330	27.61
12	122,947	1,527,192	35.35	97,627	1,190,957	27.57
13	106,419	1,633,611	34.91	97,402	1,288,359	27.53
14	128,894	1,762,505	34.97	98,380	1,386,739	27.51
15	127,367	1,889,872	35.00	84,376	1,471,115	27.24
16	113,495	2,003,367	34.78	91,909	1,563,024	27.14
17	115,698	2,119,065	34.63	103,477	1,666,501	27.23
18	124,789	2,243,854	34.63	106,539	1,773,040	27.36
19	132,275	2,376,129	34.74	104,187	1,877,227	27.44
20	111,684	2,487,813	34.55	102,208	1,979,435	27.49
21	108,659	2,596,472	34.34	105,058	2,084,493	27.57
22	108,841	2,705,313	34.16	99,821	2,184,314	27.58
23	109,533	2,814,846	34.00	124,048	2,308,362	27.88
24	117,598	2,932,444	33.94	126,106	2,434,468	28.18
25	117,121	3,049,565	33.88	95,010	2,529,478	28.11
26	140,027	3,189,592	34.08	110,370	2,639,848	28.20
27	101,651	3,291,243	33.86	109,956	2,749,804	28.29
28	115,283	3,406,526	33.79	87,642	2,837,446	28.15
29	169,693	3,576,219	34.25	103,085	2,940,531	28.17
…	…	…	…	…	…	…
998	111,411	117,812,417	32.79	98,470	102,973,089	28.66
999	103,947	117,916,364	32.79	85,890	103,058,979	28.66
1000	123,837	118,040,201	32.79	94,369	103,153,348	28.65

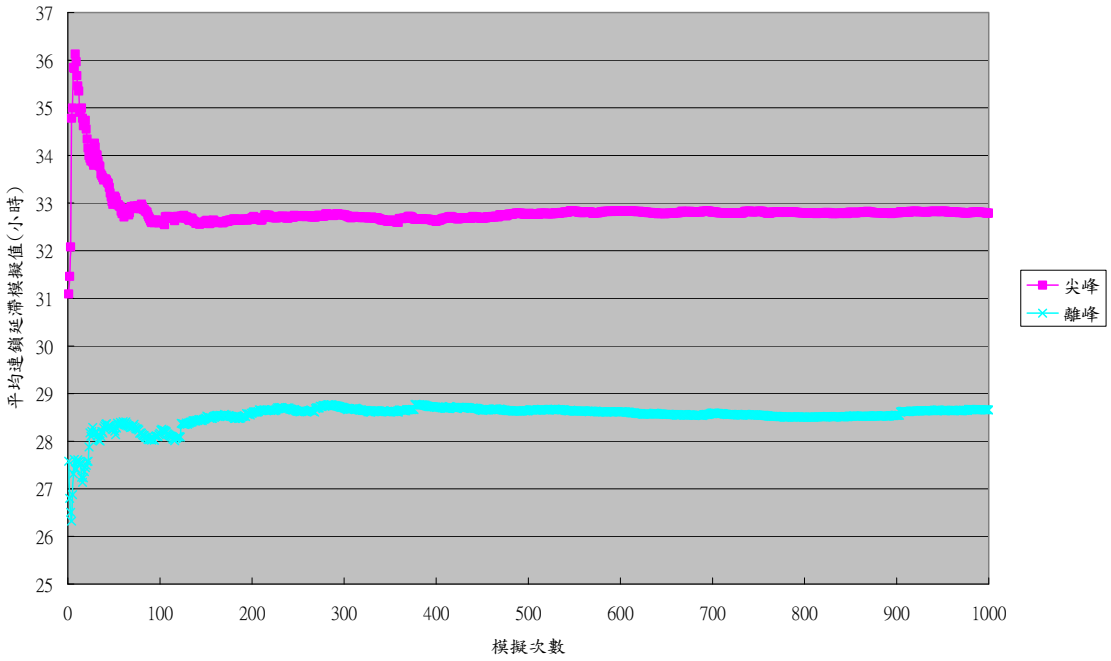


圖 6-12 七堵及新竹二端末車站之平均連鎖延滯模擬值與模擬次數關係

6.6 小結

本章為分析連鎖延滯之關鍵影響因素隨機特性，以更確實反映實際臺鐵列車之列車運轉交互作用及可能產生之連鎖延滯狀況，除蒐集近四個月之臺鐵實際營運資料以充分考量站間運轉時間及停站時間之隨機特性，並詳實彙整對號列車及通勤列車之站間運轉時間及尖、離峰停站時間之隨機資料，並將其納入模擬模式進行連鎖延滯分析。

另由於分析路段間各站之隨機停站時間資料分佈變異頗大，顯示臺鐵系統之營運資料內部已隱藏許多「初始延滯」及「連鎖延滯」，故本研究除需先將蒐集期間內過年及地震等異常事件日易造成列車延滯之樣本去除外，另再將該資料分類為尖、離峰處理其分佈資料，以分別進行後續之連鎖延滯模擬推估分析，其結果已獲得改善。

經假設上午尖峰時段內之 7:15 及離峰時段內之 10:15 於南下松山—臺北站間有 1 小時之初始延滯，經本模擬模式推估結果顯示，無論所有車站或二端末車站(七堵及新竹站)之連鎖延滯模擬其平均值皆會穩定接近於一定值，其中所有車站之連鎖延滯於尖、離峰有初始延滯情境下，其平均模擬值分別約接近於 173 小時及 153 小時；而二端末車站之連鎖延滯於尖、離峰有初始延滯情境下，其平均模擬值則分別約接近於 32.7 小時及 28.5 小時，相關分析結果皆符合先驗知識，應可作為營運單位實際運轉調度之參考。


E S A
1896

第七章 結論與建議

軌道系統容量不足往往是導致列車延滯之主因，尤其在初始延滯(first delay)發生後，後續所引發的連鎖延滯(knock-on delay)擴散效應對列車正常營運之影響甚大，因此欲確保系統之可靠度及服務品質，快速有效地降低連鎖延滯一直是重要課題。為有效釐清造成連鎖延滯之複雜因素及其交互作用，本研究依據臺鐵系統之運轉特性構建一模擬模式用以推估連鎖延滯，並以臺鐵西部幹線北部路段為案例，分析連鎖延滯之各種相關問題。依列車運行條件交互作用及延滯分析關係顯示(如圖 1-1)，軌道路線容量之決定係由複雜之路線、交通、控制等三大條件共同決定，而容量關鍵要素之運轉時隔又受列車間之運轉性能交互影響，尤其包含多車種、複線運轉及多類型月臺型式之傳統臺鐵系統更為錯綜複雜，而因容量不足產生路段瓶頸所導致之「延滯」結果及衍生之「連鎖延滯」擴散效應(delay propagation)影響，更是營運單位最急於克服解決之問題。

本章將綜合本論文前述各章內容，提出本研究之重要研究成果與貢獻，並彙整研提鐵路列車營運所面臨之連鎖延滯相關問題及因應建議，以期提供實務單位營運調度參考，並針對後續值得研究及待解決之課題，提出具體建議。

7.1 結論

一、構建可綜合分析臺鐵列車連鎖延滯之模擬模式

1.本研究所構建之模擬模式主要功能在於推估複雜之連鎖延滯，其係先透過路線、交通及控制條件之基本條件資料輸入，並考量各種實際運轉條件之假設限制，及列車衝突解決、列車延滯排除與運轉調度策略之模擬機制設定，再透過週延之物件類別導向之模擬模式流程處理所有條件交互作用，以確實推估最終之連鎖延滯。所構建之模擬模式經驗證結果顯示，有 30%之模擬延滯時間與實際延滯時間幾近相同，有 90%之列車樣本模擬延滯時間與實際資料差距在 5 分鐘之內，故整體而言顯示本模擬模式估算結果之準確度可被接受。

2.連鎖延滯擴散評估：應用本研究所構建之模擬模式，若設定初始延滯事件發生在交通量最繁忙之南下方向松山—臺北站間，則總連鎖延滯模擬結果在松山—新竹路段中任一車站之總連鎖延滯比基隆—松山路段中任一車站之總連鎖延滯都高，顯示連鎖延滯有往下游路段擴散之現象，且連鎖延滯有隨初始延滯增加而呈非線性遞增之趨勢。

3.調度策略降低連鎖延滯之比較評估：經由本模擬模式分析顯示，整體而言隨著初始延滯增加，趕點策略對連鎖延滯降低之程度就更明顯；若依所有車站之連鎖延滯減少程度排序，最佳策略為同時採取減少列車停站時間及減少站間運轉時間二種趕點策略，但若僅考慮路段端點站之連鎖延滯，則以減少站間運轉時間為最佳策略，其係因臺鐵之站距較長，站間運行時間亦較長，故採用站間趕點之效果亦可能較好，而此結果亦符合先驗知識。

4.趕點策略對連鎖延滯降低之程度：相較於未採取任何趕點策略情境，當同時採取上述二種趕點策略，其改善成效相當顯著。

二、連鎖延滯之關聯影響因素分析

1.本研究係以自行構建之模擬模式，作為釐清對連鎖延滯影響因素之核心分析工具；本模擬模式除已經過驗證其估算之準確度可被接受，並可有效分析不同路線、交通及控制條件之複雜交互作用，及其對連鎖延滯擴散效應與各種運轉調度策略成效之比較外，亦可應用於本研究所探討的列車密度/初始延滯/調度策略等關鍵因素對連鎖延滯之影響情形。

2.經由列車密度、初始延滯與連鎖延滯之關聯分析後，本研究釐清除了初始延滯之外，列車密度亦是影響連鎖延滯之關鍵因素；而由列車密度與連鎖延滯之關聯分析結果顯示，因初始延滯愈大其連鎖延滯擴散效應愈明顯，故其連鎖延滯隨列車密度增加而呈非線性陡升之趨勢亦愈明顯。而由初始延滯與連鎖延滯之關聯分析，亦可得到連鎖延滯會隨初始延滯增加而呈非線性陡升之趨勢。

3.為分析比較不同調度策略對連鎖延滯之改善程度，本研究以七堵—樹林路段之列車密度(2,048 種情境)、初始延滯(21 種情境)及設定有無站間趕點與站內趕點之四種組合情境，經以模擬模式產生共計 172,032 筆連鎖延滯之資料進行關聯分析，其結果顯示站內趕點之調度策略優於站間趕點，而同時採取該二種趕點策略對連鎖延滯之改善效果又優於採取任何單一調度策略，符合先驗的常識判斷。

4.在同時採用站內趕點與站間趕點二種策略之情境下，由列車密度與連鎖延滯之關聯分析結果顯示，外生初始延滯愈大其連鎖延滯擴散效應愈明顯，故其連鎖延滯隨列車密度增加而呈非線性陡升之趨勢亦愈明顯，另在列車密度為每小時 6 列車情境下，由初始延滯與連鎖延滯之關聯分析結果亦顯示有類似現象。而與無任何調度策略之連鎖延滯相較，其最明顯差異處除連鎖延滯顯著降低外，曲線陡升之現象亦趨於平緩，顯示同時採取二種調度策略可有效降低連鎖延滯，並可增加班

表之可靠度。

5.相關文獻曾指出連鎖延滯與關鍵影響變數間通常會呈現類似指數函數之關係，本研究由模擬模式產生之 172,032 組資料為樣本，分別以五種函數型式進行迴歸分析，結果顯示列車密度/初始延滯/調度策略等三項自變數皆為因變數連鎖延滯之顯著影響變數，且其中以指數函數 $y = 2165.659 \bullet e^{0.08685x_1} \bullet e^{0.00051x_2} \bullet e^{-0.047x_3} \bullet e^{-0.292x_4}$ 之判定係數 0.9265 為最高，顯見指數函數亦為最適合估算本研究案例連鎖延滯之迴歸模式。

6.由本研究構建之連鎖延滯迴歸函數式之係數值顯示，平均而言，若採取站間趕點調度策略則對連鎖延滯之折減比例約為 95.4%，若採取站內趕點調度策略則對連鎖延滯之折減比例可達約 74.7%，若同時採取二項調度策略則對連鎖延滯之折減比例可達約 71.2%，顯見本研究分析之臺鐵案例路段資料顯示站內趕點調度策略對連鎖延滯降低之功效較為明顯。

三、初始延滯區位及運轉調度策略對連鎖延滯之關聯分析

1.本研究另以臺鐵系統交通量最繁忙之北部七堵—樹林路段為例，針對有初始延滯發生，其發生區位、持續時間及運轉調度策略對連鎖延滯降低之關聯性及影響程度進行分析，結果顯示若初始延滯發生之區位愈靠近上游路段，則所有車站及分析路段二端末車站之連鎖延滯亦將愈大。

2.當初始延滯發生之區位愈靠近上游路段，則運轉調度策略對於連鎖延滯降低之程度就愈大，亦即其恢復至原訂班表之效果也愈好。相關分析結果，除可對於初始延滯與連鎖延滯之複雜交互作用關係有更深入之了解，亦可作為營運單位後續列車營運及運轉調度之參考。

四、關鍵影響因素隨機特性之連鎖延滯模擬分析

1.由於分析路段間各站之隨機停站時間資料分佈變異頗大，顯示臺鐵系統之營運資料內部已隱藏許多「初始延滯」及「連鎖延滯」，故本研究除需先將所蒐集時段內過年及地震等旅次特性異常易造成列車延滯之樣本去除外，另因發現其資料可能存在時段差異特性，故亦將該資料分類為尖、離峰處理其分佈資料，再分別進行後續之連鎖延滯模擬推估分析，其結果已獲得改善。

2.經假設上午尖峰時段內之 7:15 及離峰時段內之 10:15 於南下松山—臺北站間有 1 小時之初始延滯，經本模擬模式推估結果顯示，無論所有車站或二端末車站(七

堵及新竹站)之連鎖延滯模擬其平均值皆會穩定接近於一定值，其中所有車站之連鎖延滯於尖、離峰有初始延滯情境下，其平均模擬值分別約接近於 173 小時及 153 小時；而二端末車站之連鎖延滯於尖、離峰有初始延滯情境下，其平均模擬值則分別約接近於 32.7 小時及 28.5 小時，相關分析結果皆符合先驗知識，應可作為營運單位實際運轉調度之參考。

7.2 建議

1.由於臺鐵傳統區域鐵路系統之連鎖延滯所牽涉問題錯綜複雜，故進行相關研究時皆需依據詳盡正確之營運及延滯資料作為分析基礎。經由本研究發現，目前相關營運資料皆需由中央列車控制 CTC(centralized train control)資料庫擷取，但因 CTC 僅可偵測列車實際通過進站號誌機與出發號誌機之時間，而非列車實際停站與離站啟動之時間，故如欲取得真正的列車延滯資料仍需經過轉換處理，蒐集處理之難度甚高。故為利臺鐵營運單位之服務可靠度掌握提升及後續延滯相關研究之分析所需，亟需臺鐵系統完整營運資料庫之建立。

2.本研究進行連鎖延滯相關分析過程中發現，由於臺鐵系統之營運資料本身即摻雜延滯(包括初始延滯及連鎖延滯)成分在內，因常未能有效釐清所蒐集之運轉資料以作為模擬模式建構及後續案例分析，故造成模式驗證時無法適用各項統計檢定分析，更增加連鎖延滯後續分析之複雜度。另由於列車運轉之許多關鍵參數皆具有隨機特性(例如本研究已考慮之站間運轉時間及停站時間等二項參數)，但其亦面臨蒐集資料前之複雜處理工作(如扣除有重大事件之營運資料)，極為費時。故有關前述各類型延滯量及相關重要參數資料之蒐集處理，皆屬延滯相關研究所必須，建議營運單位若能建立完善之延滯資料庫，並提供初步之延滯時空及原因分析，將可直接作為後續延滯更深入分析之基礎。

3.由於路線、交通、控制等三大類條件之交互作用皆會影響整體列車營運之「連鎖延滯」，各項因素彼此間之交互作用相當複雜不易掌握分析，且由於模擬模式具依個案所需輸入資料不同且數量極大之特性，故除前述 CTC 營運資料庫之建立外，若欲進行軟、硬體之路線、交通、控制等各項條件改善對於連鎖延滯之全方位影響分析時，將無法因應。建議仍需儘速結合臺鐵系統之運務、工務、機務、電務等四大部門建立整合型臺鐵系統營運資料庫，以符所需。

4.本研究就錯綜複雜之關係中僅先探討釐清列車密度(交通條件)、調度策略(控制條件)及初始延滯(外生條件)對於連鎖延滯之影響，表 1.1 所列舉的可能影響條件因素中仍有許多項目值得繼續探討，建議後續研究可應用本模擬模式，篩選其他較

重要之關鍵因素進行分析，以逐步釐清影響連鎖延滯之關鍵因子。另臺鐵目前為因應高鐵系統加入城際運輸市場，正面臨營運轉型而推動捷運化計畫，如何從工程規劃角度分析站內軌道及月臺佈置方式(路線條件)對連鎖延滯之影響，以作為營運單位改善可靠度之參考，亦是值得探討的重要課題。

5.鑑於臺鐵系統各路段之路線容量不儘相同，若能以列車密度與路段容量的比例(即路線利用率，V/C)取代列車密度進行分析，似應較能反映路段之營運特性，惟路段容量之估算因係為另一複雜問題，故本研究並未將容量估計納入研究範圍，建議後續研究可考慮予以納入分析。

6.由本研究結果得知，當列車流量密度趨近於軌道容量時，連鎖延滯會隨初始延滯及列車流量密度增加而呈非線性指數函數遞增，故如何迅速有效降低連鎖延滯對列車正常營運之衝擊，一直是營運單位及相關研究所積極追求之目標。本研究雖已針對營運單位最常採用之站間趕點及站內趕點等運轉調度策略模擬分析其對連鎖延滯改善之程度，惟有關連鎖延滯降低之議題，仍有許多運轉整理策略(如減班、變換列車順序、改變列車交通組成等)、單線雙向運轉調度策略、初始延滯發生於站內股道之列車行駛股道選擇及運轉時隔等相關問題，值得後續研究進一步分析，以使連鎖延滯各面向議題之研究更為週延完整。


E S A
1896

參考文獻

中文部分

1. 李治綱、陳朝輝、簡聰裕(民 91)，「捷運鐵路列車延滯事件發生後行車調度策略之模擬分析」，**運輸計劃季刊**，第 31 卷，第 2 期，頁 299-322。
2. 李治綱、鍾志成、林杜寰、張仕龍、張恩輔、陳一昌、張開國、吳熙仁(民 98)，「公共運輸之安全績效：臺灣鐵路管理局之個案分析」，**運輸計劃季刊**，第 38 卷，第 4 期，頁 381~406。
3. 邱戊吉(民 99)，應用 Max-Plus 代數分析鐵路時刻表穩定性，國立交通大學交通運輸研究所碩士論文。
4. 交通部運輸研究所(民 94)，**軌道容量研究－臺鐵系統容量模式之建構分析（一）**。
5. 交通部運輸研究所(民 96)，**運輸系統容量分析暨應用研究—軌道系統(1/4)**。
6. 交通部運輸研究所(民 97)，**運輸系統容量分析暨應用研究—軌道系統(2/4)**。
7. 交通部運輸研究所(民 98)，**運輸系統容量分析暨應用研究—軌道系統(3/4)。**
8. 交通部運輸研究所(民 100)，**軌道系統容量與可靠度分析研究(1/3)**。
9. 周斯畏(民 91)，**物件導向系統分析與設計使用 UML 與 C++**，臺北市：全華科技圖書股份有限公司。
10. 周學怡(民 86)，列車運行計畫之可靠性分析，國立成功大學交通管理科學研究所碩士論文。
11. 黃哲旭(民 85)，捷運鐵路列車模擬模式之研究，國立成功大學交通管理科學研究所碩士論文。
12. 黃範哲(民 90)，高速鐵路系統運轉整理之研究，國立成功大學交通管理科學研究所碩士論文。
13. 黃承傳、劉昭榮(民 100)，「鐵路列車密度與初始延滯以及調度策略對連鎖延滯之影響分析」，**運輸計劃季刊**，第 40 卷，第 1 期，頁 63-98。
14. 謝興盛(民 92)，捷運列車延誤時班距調整模式之模擬分析-以臺北捷運中、高運量系統為例，國立成功大學交通管理科學研究所博士論文。
15. 簡聰裕(民 89)，捷運系統運轉整理之研究，國立成功大學交通管理科學研究所碩士論文。
16. 楊立安(民 96)，臺灣鐵路運轉整理之研究，國立高雄第一科技大學運籌管

理系碩士論文。
17. 陳朝輝(民 97)，應用物件導向模式技術於捷運系統運轉整理之模擬分析，國立成功大學交通管理科學研究所博士論文。
18. 陳英相與張仁城(民 86)，**鐵路運轉規章（含概要、大意）**，臺北，千華圖書出版事業有限公司。
19. 張恩輔(民 91)，捷運系統運轉整理之模擬分析，國立成功大學交通管理科學研究所碩士論文。
20. 賴勇成、李宗晏、劉牧阡、陳冠廷(民 99)，「臺鐵時刻表穩定度與效率評估」，第 25 屆中華民國運輸學會論文集。

英文部分

1. Barter, W. M. (1998), "Application of Computer Simulation to Rail Capacity Planning", *Computers in Railways VI*, pp. 199-211.
2. Braalund, U., Lindberg, P.O., Nou, A. and Nilsson, J.E. (1998), "Railway timetabling using Lagrangian Relaxation", *Transportation Science*, Vol. 32(4), pp.358-369.
3. Briggs, K. and Beck, C. (2007), "Modeling Train Delays with Q-exponential Functions", *Physica A: Statistical Mechanics and its Applications*, Vol. 378, Issue 2, pp. 498-504.
4. Carey, M. (1999), "Ex ante Heuristic Measures of Schedule Reliability", *Transportation Research Part B*, Vol. 33, pp. 473-494.
5. Carey, M. and Carville, S. (2000), "Testing Schedule Performance and Reliability for Train Stations", *Journal of the Operational Research Society*, Vol. 51, pp. 666-682.
6. Carey, M. and Crawford, I. (2007), "Scheduling Trains on a Network of Busy Complex Stations", *Transportation Research Part B*, Vol. 41, pp. 159-178.
7. Chakroborty, P. and Vikram, D. (2008), "Optimum Assignment of Trains to Platforms under Partial Schedule Compliance", *Transportation Research Part B*, Vol. 42, pp. 169-184.
8. Chang, S. C. and Chung, Y. C. (2005), "From Timetabling to Train Regulation – A New Train Operation Model", *Information and Software Technology*, Vol. 47, pp. 575-583.
9. Chang, Y. H., Yeh, C. H. and Shen, C. C. (2000), "A Multiobjective model for Passenger Train Services Planning: Applications to Taiwan's High Speed Rail

Line", *Transportation Research B*, Vol. 34, pp.91-106.

10. Chen, B. and Harker, P.T. (1990), "Two-moment estimation of the delay on single-track rail lines with scheduled traffic", *Transportation Science*, Vol. 24, No. 4, pp. 261-275.
11. Chen, C.H., Lee, C.K. and Chang, E.F. (2003), "Simulation Analysis on the Dispatching Operation of Rapid Rail Transit", *Journal of the Eastern Asia Society for Transportation Studies*, V.5, pp323-338.
12. Cheng, Y. (1996), "Optimal train traffic rescheduling simulation by a knowledge-based system combined with critical path method," *Simulation Practice and Theory*, Vol. 4, pp.399-413.
13. Cheng, Y. (1998), "Rule-based train traffic reactive simulation model," *Applied Artificial Intelligence*, Vol.12, pp.5-27.
14. Corman, F., D'Ariano, A., Pacciarelli, D. and Pranzo, M. (2010), "A Tabu Search Algorithm for Rerouting Trains during Rail Operations", *Transportation Research Part B*, Vol. 44, pp. 175-192.
15. EuROPE-TRIP (2000), Ferrovie dello Stato Spa - Divisione Infrastruttura, *European Railways Optimisation Planning Environment - Transportation Railways Integrated Planning*, Final Report, Roma Italy.
16. Fay, A. (2001), "A Fuzzy Petri Net approach to decision-making in case of railway track closures," ***IFSA World Congress and 20th NAFIPS International Conference, 2001. Joint 9th***, Vol.5, pp.2858-2863.
17. Ferreira, L. and Higgins, A. (1996), "Modelling Reliability of Train Arrival Times", *Journal of Transportation Engineering*, Vol. 122, pp. 414-420.
18. Goverde, R.M.P. (2005), "Railway Timetable Stability Analysis Using Max-Plus System Theory", *Transportation Research Part B*, Vol. 41, pp. 179–201.
19. Hansen, I. A. (2000), "Station Capacity and Stability of Train Operations", *Computers in Railways VII*, pp. 809-816.
20. Hansen, I. A. and Pachl, J. (2008), ***Railway Timetable & Traffic: Analysis – Modelling – Simulation***, Railway Gazette, Hamburg Germany.
21. Higgins, A., Kozan, E. and Ferreira L. (1995), "Modeling Delay Risks Associated with Train Schedules", *Transportation Planning and Technology*, Vol. 19, Issue 2, pp. 89-108.
22. Higgins, A. and Kozan, E. (1998), "Modeling Train Delay in Urban

Networks", *Transportation Science*, Vol. 32, No.4, pp.346-357.

23. Huisman, T. and Boucherie, R. J. (2001), "Running Times on Railway Sections with Heterogeneous Train Traffic", *Transportation Research Part B*, Vol. 35, pp. 271-292.
24. Hwang, C. C. and Liu, J. R. (2010), "A Simulation Model for Estimating Knock-on Delay of Taiwan Regional Railway", *Journal of the Eastern Asia Society for Transportation Studies*, Vol.8, pp. 1082-1097.
25. Jong, J.C., Lin, T. H., Lee, C. K. and Hu, H. L. (2010), "The Analysis on Train Reliability of Taiwan High Speed Rail", *Proceedings of 12th COMPRAIL Conference*, Wessex Institute of Technology, pp. 169-180.
26. Mattsson, L. G. (2004), "Train Service Reliability - A Survey of Methods for Deriving Relationships for Train Delays", Written at the Request of the Swedish Institute for Transport and Communications Analysis, Stockholm Sweden.
27. Middelkoop, D. and Bouwman, M. (2000), "Train Network Simulator for Support of Network Wide Planning of Infrastructure and Timetables", *Computers in Railways VII*, pp. 267-276.
28. Middelkoop, D. and Bouwman, M. (2002), "Testing the Stability of the Rail Network", *Computers in Railways VIII*, pp. 995-1002.
29. Missikoff, M. (1998), "An Object-oriented Approach to an Information and Decision Support System for Railway Traffic Control," *Engineering Applications of Artificial Intelligence*, Vol.11, pp.25-40.
30. Murray, A. T. and Grubesic, T. H. (2007), **Critical Infrastructure Reliability and Vulnerability**, Springer Berlin Heidelberg.
31. Nelson, D. and O'Neil, K., (2000) "Commuter Rail Service Reliability On-Time Performance and Causes for Delays", *Transportation Research Record*, Vol. 1704, pp. 42-50.
32. Nie, L. and Hansen, I. A. (2005), "System Analysis of Train Operations and Track Occupancy at Railway Stations", *European Journal of Transport and Infrastructure Research*, Vol. 5, No. 1, pp. 31-54.
33. O'Dell, S. and Wilson, N.H.M. (1999), "Optimal Real-time Control Strategies for Rail Transit Operations during Disruption", *In Computer Aided Transit Scheduling*, Wilson, N.H.M. (Ed.), pp.299-323, Springer-Verlag.
34. Olsson, Nils, O.E. and Haugland, H. (2004), "Influencing Factors on Train

Punctuality—Results from Some Norwegian Studies", *Transport Policy 11*, pp. 387-397.

35. Parkinson, T. (1996), "Rail Transit Capacity", Transportation Research Board, National Academy Press.
36. Rebreyend, P. (2005), "DisTrain: A simulation tool for train dispatching," *Proceedings of 8th International IEEE Conference on Intelligence Transportation Systems Vienna, Austria*, pp.801-806.
37. Rietveld, P., Bruinsma, F. R. and Van Vuuren, D. J. (2001), "Coping with Unreliability in Public Transport Chains: a case for the Netherlands", *Transportation Research Part A*, Vol. 35, 539-559.
38. Vromans, M. J., Dekker, R. and Kroon, L. G. (2006), "Reliability and Heterogeneity of Railway Services", *European Journal of Operational Research*, Vol. 172, pp. 647-665.
39. Won, H.S., Kyung, S. C. and Sung, M. R. (2001), "An Analysis of Railroad Skip Stop System without Siding Track", *9th World Conference on Transport Research*.
40. Yuan, J. and Hansen, I. A. (2007), "Optimizing Capacity Utilization of Stations by Estimating Knock-on Train Delays", *Transportation Research Part B*, Vol. 41, pp. 202-217.
41. Zhu, P. and Schnieder E. (2000), "Determining Traffic Delays through Simulation", *In Computer-Aided Scheduling of Public Transport*, Braunschweig, Germany, pp387-397.


E S A
1896

附錄一 原始程式碼

```
using System;
using System.Collections.Generic;
using System.Text;


namespace ClassLibrary1
{
    public class TrainEvent : IComparable<TrainEvent>
    {

        public static int Num = 0;
        public int CompareTo(TrainEvent otherEvent)
        {
            if (Time > otherEvent.Time)
            {
                return 1;
            }
            else if (Time < otherEvent.Time)
            {
                return -1;
            }
            else
            {
                return m_SerNum.CompareTo(otherEvent.m_SerNum);
            }

        }

        private Directions m_Dir;
        private int m_Time = -RailSystem.DayTime;
        private EventTypes m_Type;
        private string m_TrainID = null;
        private string m_Station = null;
        private int m_SerNum = 0;

        public TrainEvent(string TrainID, int Time, string Station, EventTypes Type, Directions
Dir)
        {
            Num++;
            m_TrainID = TrainID;
            m_Time = Time;
            m_Station = Station;
            m_Type = Type;
            m_Dir = Dir;
            m_SerNum = Num;
        }

        public string TrainID
        {
            get
            {
                return m_TrainID;
            }
            set
            {
                m_TrainID = value;
            }
        }

        public string Station
        {
            get
            {
                return m_Station;
            }
            set
            {
                m_Station = value;
```

```
            }
        }

        public int Time
        {
            get
            {
                return m_Time;
            }
            set
            {
                m_Time = value;
            }
        }

        public Directions Dir
        {
            get
            {
                return m_Dir;
            }
            set
            {
                m_Dir = value;

            }
        }

        public EventTypes Type
        {
            get
            {
                return m_Type;
            }
            set
            {
                m_Type = value;
            }
        }
    }
}

using System;
using System.Collections.Generic;
using System.Text;

namespace ClassLibrary1
{
    public class DwellPlan
    {
        public string station;
        public int expArrTime;
        public int expDepTime;
        public int actArrTime;


        public int actDepTime;

        public DwellPlan(string stationName, int arrivalTime, int departureTime)
        {
            station = stationName;
            expArrTime = arrivalTime;
            expDepTime = departureTime;
            actArrTime = int.MinValue;
            actDepTime = int.MinValue;
        }
    }

    public class Train
    {
        private int m_priority;
        public int Priority
        {
```

```
            get { return m_priority; }
            set { m_priority = value; }
        }

        private List<DwellPlan> m_dwellPlanList;
        public List<DwellPlan> DwellPlanList
        {
            get { return m_dwellPlanList; }
        }

        public Train()
        {
            m_dwellPlanList = new List<DwellPlan>();
        }

        private Directions m_Dir;
        public Directions Dir
        {
            get { return m_Dir; }
            set { m_Dir = value; }
        }

        private int m_Task=0;
        public int Task
        {
            get { return m_Task; }
        }
        public bool NextTask(){
            m_Task++;
            if (m_Task < DwellPlanList.Count)
                return true;
            else
                return false;
        }
        public void ResetTask()
        {
            m_Task = 0;
        }

        private string m_Location = null;
        public string Location
        {
            get { return m_Location; }
            set { m_Location = value; }
        }

        public int IsArrDelay(int Time)
        {
            return Time - DwellPlanList[Task].expArrTime;
        }

        public int IsDepDelay(int Time)
        {
            return Time - DwellPlanList[Task].expDepTime;
        }

        public int DefaultDwellTime()
        {
            return DwellPlanList[Task].expDepTime - DwellPlanList[Task].expArrTime;
        }
        public int DefaultRunningTime()
        {
            return DwellPlanList[Task].expArrTime - DwellPlanList[Task-1].expDepTime;
        }
    }
}
using System;
using System.Collections.Generic;
using System.Text;
using System.Diagnostics;

namespace ClassLibrary1
{
```

```
public enum EventTypes { ARRIVE, DEPART, PASS, FD_START, FD_STOP };
public enum Directions { ASC=0, DESC=1 };

public class Tracks
{
    public Tracks(string Name)
    {
        m_Name = Name;
        Reset();
    }
    private string m_Name;
    public string Name
    {
        get { return m_Name; }
    }

    private bool m_blocked;
    public bool Blocked
    {
        get { return m_blocked; }
        set { m_blocked = value; }
    }

    private string m_occupied;
    public string Occupied
    {
        get { return m_occupied; }
    }

    private EventTypes m_type;
    public EventTypes LastEventType
    {
        get { return m_type; }
    }

    private Directions m_direction;
    public Directions LastEventDirection
    {
        get { return m_direction; }
    }

    private int m_time = -RailSystem.DayTime;
    public int LastEventTime
    {
        get { return m_time; }
    }

    private int m_LastArrTime = -RailSystem.DayTime;
    public int LastArrtTime
    {
        get { return m_LastArrTime; }
    }


    public void Reset()
    {
        m_time = -RailSystem.DayTime;
        m_occupied = null;
        m_blocked = false;
        m_LastArrTime = -RailSystem.DayTime;
    }

    public void SetLastEvent(ref TrainEvent TE)
    {
        m_type = TE.Type;
        m_time = TE.Time;
        m_direction = TE.Dir;

        if (EventTypes.ARRIVE==m_type)
        {
            m_LastArrTime = TE.Time;
            m_occupied = TE.TrainID;
        }
```

```
            else if (EventTypes.DEPART == m_type)
            {
                Debug.Assert(m_occupied == TE.TrainID);
                m_occupied = null;
            }
            else if (EventTypes.PASS == m_type)
            {
                m_occupied = null;
            }
            else
            {
                Debug.Assert(false);
            }
        }
    }
}

using System;
using System.Collections.Generic;
using System.Text;
using System.Diagnostics;

namespace ClassLibrary1
{

    public class Station
    {
        public int totalDelay = 0;

        private int m_headwayDA;
        public int H_DA
        {
            get { return m_headwayDA; }
            set { m_headwayDA = value; }
        }

        private int m_headwayAA;
        public int H_AA
        {
            get { return m_headwayAA; }
            set { m_headwayAA = value; }
        }

        private int m_headwayDD;
        public int H_DD
        {
            get { return m_headwayDD; }
            set { m_headwayDD = value; }
        }

        private int m_headwayAD_R;
        public int H_AD_R
        {
            get { return m_headwayAD_R; }
            set { m_headwayAD_R = value; }
        }

        private int m_headwayDA_R;
        public int H_DA_R
        {
            get { return m_headwayDA_R; }
            set { m_headwayDA_R = value; }
        }

        protected Tracks m_track1;
        public Tracks Track1
        {
            get { return m_track1; }
        }

        protected Tracks m_track2;
        public Tracks Track2
        {
```

```
            get { return m_track2; }
        }

        private int m_mileage;
        /// <summary>
        /// </summary>
        public int Mileage
        {
            get { return m_mileage; }
            set { m_mileage = value; }
        }

        private int m_minDwellTime;
        public int MinDwellTime
        {
            get { return m_minDwellTime; }
            set { m_minDwellTime = value; }
        }

        public Station()
        {
            m_track1 = new Tracks("Track1");
            m_track2 = new Tracks("Track2");
        }

        public virtual string GetStationType()
        {
            return "Type IV";
        }

        protected bool CheckTrackNoneOccupy(ref Tracks track)
        {
            if (null == track.Occupied && !track.Blocked)
                return true;
            else
                return false;
        }

        protected bool CheckDA(ref Tracks track, int Time)
        {
            if ( this.H_DA <= (Time - track.LastEventTime) )
                return true;
            else
                return false;
        }

        protected bool CheckAA(ref TrainEvent TE, ref Tracks AnotherOne)
        {
            if (AnotherOne.LastEventTime < 0)
                return true;
            if (AnotherOne.LastEventDirection != TE.Dir)
                return true;

            if ((TE.Time - AnotherOne.LastArrtTime) >= H_AA)
            {
                return true;
            }
            else {
                return false;
            }


            if (EventTypes.ARRIVE == AnotherOne.LastEventType || EventTypes.PASS ==
AnotherOne.LastEventType)
            {
                if ((TE.Time - AnotherOne.LastEventTime) >= H_AA)
                    return true;
                else
                    return false;
            }
            else
            {
                return true;
```

```
            }
        }
        protected bool CheckDD(ref TrainEvent TE, ref Tracks AnotherOne)
        {
            if (AnotherOne.LastEventTime < 0)
                return true;
            if (AnotherOne.LastEventDirection != TE.Dir)
                return true;

            if (EventTypes.DEPART == AnotherOne.LastEventType)
            {
                if ((TE.Time - AnotherOne.LastEventTime) >= H_DD)
                    return true;
                else
                    return false;
            }
            else
            {
                return true;
            }
        }

        protected bool CheckAP_DP(ref TrainEvent TE, ref Tracks SecTrack)
        {
            if (SecTrack.LastEventTime < 0)
                return true;
            if (TE.Dir != SecTrack.LastEventDirection)
                return true;

            Debug.Assert(EventTypes.ARRIVE == SecTrack.LastEventType || EventTypes.DEPART ==
SecTrack.LastEventType);
            if (EventTypes.ARRIVE == SecTrack.LastEventType)
            {
                if ((TE.Time - SecTrack.LastEventTime) >= H_AA)
                    return true;
                else
                    return false;
            }
            else
            {
                if ((TE.Time - SecTrack.LastEventTime) >= H_DD)
                    return true;
                else
                    return false;
            }
        }


        public virtual bool IsAllTrackEmpty()
        {
            if (null == m_track1.Occupied && null == m_track2.Occupied)
                return true;
            else
                return false;
        }

        public virtual void Reset()
        {
            m_track1.Reset();
            m_track2.Reset();
        }

        public virtual bool Arrival(ref TrainEvent TE, bool IsFollowingTrainPassing)
        {
            if (Directions.ASC == TE.Dir)
            {
                if (CheckTrackNoneOccupy(ref m_track1) && CheckDA(ref m_track1, TE.Time))
                {
                    m_track1.SetLastEvent(ref TE);
                    return true;
                }
                else
                {
```

```
                return false;
            }
        }
        else
        {
            if (CheckTrackNoneOccupy(ref m_track2) && CheckDA(ref m_track2, TE.Time))
            {
                m_track2.SetLastEvent(ref TE);
                return true;
            }
            else
            {
                return false;
            }
        }
    }

    public virtual bool Depature(ref TrainEvent TE)
    {
        if (Directions.ASC == TE.Dir)
        {
            m_track1.SetLastEvent(ref TE);
        }
        else
        {
            m_track2.SetLastEvent(ref TE);
        }


        return true;
    }
    public virtual bool Passing(ref TrainEvent TE)
    {
        return Arrival(ref TE,false);
    }

    public virtual void SetTrackBlock(string track, bool block)
    {
        Debug.Assert(track != "Track3");
        Debug.Assert(track != "Track4");
        if (m_track1.Name == track)
        {
            m_track1.Blocked = block;
        }
        else if (m_track2.Name == track)
        {
            m_track2.Blocked = block;
        }
    }
}

public class StationI : Station
{
    private Tracks m_track3;
    public Tracks Track3
    {
        get { return m_track3; }
    }

    private Tracks m_track4;
    public Tracks Track4
    {
        get { return m_track4; }
    }

    public StationI()
    {
        m_track3 = new Tracks("Track3");
        m_track4 = new Tracks("Track4");
    }

    public override string GetStationType()
    {
```

```
            return "Type I";
        }

        public override bool IsAllTrackEmpty()
        {
            if (null == m_track1.Occupied && null == m_track2.Occupied && null == m_track3.Occupied
&& null == m_track4.Occupied)
                return true;
            else
                return false;
        }

        public override void Reset()
        {
            m_track1.Reset();
            m_track2.Reset();
            m_track3.Reset();
            m_track4.Reset();
        }

        public override bool Arrival(ref TrainEvent TE, bool IsFollowingTrainPassing)
        {
            if (Directions.ASC == TE.Dir)
                return Arrival(ref TE, IsFollowingTrainPassing, ref m_track1, ref m_track3);
            else
                return Arrival(ref TE, IsFollowingTrainPassing, ref m_track2, ref m_track4);
        }



        private bool Arrival(ref TrainEvent TE, bool IsFollowingTrainPassing, ref Tracks Pri, ref
Tracks Sec)
        {
            if (CheckTrackNoneOccupy(ref Sec) && CheckDA(ref Sec, TE.Time) && CheckAA(ref TE, ref
Pri))
            {
                Sec.SetLastEvent(ref TE);
                return true;
            }
            else if (CheckTrackNoneOccupy(ref Pri) && CheckDA(ref Pri, TE.Time) && CheckAA(ref TE,
ref Sec))
            {
                Pri.SetLastEvent(ref TE);
                return true;
            }
            else
            {
                return false;
            }
        }

        public override bool Depature(ref TrainEvent TE)
        {
            if (Directions.ASC == TE.Dir)
                return Depature(ref TE, ref m_track1, ref m_track3);
            else
                return Depature(ref TE, ref m_track2, ref m_track4);
        }

        private bool Depature(ref TrainEvent TE, ref Tracks Pri, ref Tracks Sec)
        {
            Debug.Assert((Pri.Occupied == TE.TrainID && Sec.Occupied != TE.TrainID) ||
(Pri.Occupied != TE.TrainID && Sec.Occupied == TE.TrainID));
            if (Pri.Occupied == TE.TrainID)
            {
                if (  CheckDD(ref TE, ref Sec) )
                {
                    Pri.SetLastEvent(ref TE);
                    return true;
                }
                else
                {
                    return false;
```

```
                }
            }
            else
            {
                if (CheckDD(ref TE, ref Pri))
                {
                    Sec.SetLastEvent(ref TE);
                    return true;
                }
                else
                {
                    return false;
                }
            }

        }

        public override bool Passing(ref TrainEvent TE)
        {
            if (Directions.ASC == TE.Dir)
                return Passing(ref TE, ref m_track1, ref m_track3);
            else
                return Passing(ref TE, ref m_track2, ref m_track4);
        }

        private bool Passing(ref TrainEvent TE, ref Tracks Pri, ref Tracks Sec)
        {
            if (CheckTrackNoneOccupy(ref Pri) && CheckDA(ref Pri, TE.Time) && CheckAP_DP(ref TE,
ref Sec) )
            {
                Pri.SetLastEvent(ref TE);
                return true;
            }
            else
            {
                return false;
            }
        }

        public override void SetTrackBlock(string track, bool block)
        {
            if (Track3.Name == track)
            {
                Track3.Blocked = block;
            }
            else if (Track4.Name == track)
            {
                Track4.Blocked = block;
            }
            else
            {
                base.SetTrackBlock(track, block);
            }
        }

    }

    public class StationII : Station
    {
        private Tracks m_track3;
        public Tracks Track3
        {
            get { return m_track3; }
        }

        public StationII()
        {
            m_track3 = new Tracks("Track3");
        }

        public override string GetStationType()
        {
            return "Type II";
```

```
        }

        public override bool IsAllTrackEmpty()
        {
            if (null == m_track1.Occupied && null == m_track2.Occupied && null ==
m_track3.Occupied)
                return true;
            else
                return false;
        }
        public override bool Arrival(ref TrainEvent TE, bool IsFollowingTrainPassing)
        {
            return true;
        }
        public override bool Depature(ref TrainEvent TE)
        {
            return true;
        }
        public override bool Passing(ref TrainEvent TE)
        {
            return true;
        }
    }

    public class StationIII_R : Station
    {
        private Tracks m_track3;
        public Tracks Track3
        {
            get { return m_track3; }
        }

        private string m_Type;
        public  string Type
        {
            get { return m_Type; }
        }


        public StationIII_R(string type)
        {
            m_track3 = new Tracks("Track3");
            m_Type = type;
            CrossOverInfo Rule;
            switch (type)
            {
                case "III_R":
                    Rule = new CrossOverInfo("0==>Track3", "Track2==>1", false);
                    CrossInfoList.Add(Rule);
                    Rule = new CrossOverInfo("Track2==>1", "0==>Track3", true);
                    CrossInfoList.Add(Rule);
                    Rule = new CrossOverInfo("Track3==>1", "0==>Track3", true);
                    CrossInfoList.Add(Rule);
                    Rule = new CrossOverInfo("1==>Track2", "Track3==>0", false);
                    CrossInfoList.Add(Rule);
                    Rule = new CrossOverInfo("Track3==>0", "1==>Track2", true);
                    CrossInfoList.Add(Rule);
                    Rule = new CrossOverInfo("Track3==>0", "1==>Track3", true);
                    CrossInfoList.Add(Rule);
                    break;
                case "III_L":
                    Rule = new CrossOverInfo("0==>Track1", "Track3==>1", false);
                    CrossInfoList.Add(Rule);
                    Rule = new CrossOverInfo("Track3==>1", "0==>Track1", true);
                    CrossInfoList.Add(Rule);
                    Rule = new CrossOverInfo("Track3==>1", "0==>Track3", true);
                    CrossInfoList.Add(Rule);
                    Rule = new CrossOverInfo("1==>Track3", "Track1==>0", false);
                    CrossInfoList.Add(Rule);
                    Rule = new CrossOverInfo("Track1==>0", "1==>Track3", true);
                    CrossInfoList.Add(Rule);
                    Rule = new CrossOverInfo("Track3==>0", "1==>Track3", true);
                    CrossInfoList.Add(Rule);
```

```
                break;
            case "II":
                Rule = new CrossOverInfo("Track3==>1", "0==>Track3", true);
                CrossInfoList.Add(Rule);
                Rule = new CrossOverInfo("Track3==>0", "1==>Track3", true);
                CrossInfoList.Add(Rule);
                break;
            default:
                Debug.Assert(false);
                break;
        }
    }

    public override string GetStationType()
    {
        return "Type III_R";
    }
    public override bool IsAllTrackEmpty()
    {
        if (null == m_track1.Occupied && null == m_track2.Occupied && null ==
m_track3.Occupied)
            return true;
        else
            return false;
    }

    public override void Reset()
    {
        m_track1.Reset();
        m_track2.Reset();
        m_track3.Reset();

        foreach (CrossOverInfo rule in CrossInfoList)
        {
            rule.Reset();
        }

    }

    private List<CrossOverInfo> CrossInfoList = new List<CrossOverInfo>();

    private void UpdateCrossoverInfo(ref TrainEvent TE, string TrackName)
    {
        string RouteName = null;

        switch (TE.Type)
        {
            case EventTypes.ARRIVE:
                RouteName = ((int)TE.Type).ToString() + RailSystem.AndSymbol + TrackName;
                UpdateCrossoverInfo(TE.Time, RouteName);
                break;
            case EventTypes.DEPART:
                RouteName = TrackName + RailSystem.AndSymbol + ((int)TE.Type).ToString();
                UpdateCrossoverInfo(TE.Time, RouteName);
                break;
            case EventTypes.PASS:
                RouteName = ((int)TE.Type).ToString() + RailSystem.AndSymbol + TrackName;
                UpdateCrossoverInfo(TE.Time, RouteName);
                RouteName = TrackName + RailSystem.AndSymbol + ((int)TE.Type).ToString();
                UpdateCrossoverInfo(TE.Time, RouteName);
                break;
            default:
                Debug.Assert(false);
                break;
        }
    }

    private void UpdateCrossoverInfo(int Time, string RouteName)
    {
        foreach( CrossOverInfo rule in CrossInfoList )
            rule.UpdateFirstRoute(Time, RouteName);
    }
```

```
        private bool CheckCrossoverHeadway(ref TrainEvent TE, string TrackName)
        {
            string RouteName=null;

            switch (TE.Type)
            {
                case EventTypes.ARRIVE:
                    RouteName= ((int)TE.Type).ToString()+RailSystem.AndSymbol+TrackName;
                    return CheckCrossoverHeadway(TE.Time, RouteName);
                    //break;
                case EventTypes.DEPART:
                    RouteName= TrackName+RailSystem.AndSymbol+((int)TE.Type).ToString();
                    return CheckCrossoverHeadway(TE.Time, RouteName);
                    //break;
                case EventTypes.PASS:
                    RouteName= ((int)TE.Type).ToString()+RailSystem.AndSymbol+TrackName;
                    bool b1 = CheckCrossoverHeadway(TE.Time, RouteName);
                    RouteName= TrackName+RailSystem.AndSymbol+((int)TE.Type).ToString();
                    bool b2 = CheckCrossoverHeadway(TE.Time, RouteName);
                    return ( b1 && b2);
                    //break;
                default:
                    Debug.Assert(false);
                    return true;
            }
        }
        private bool CheckCrossoverHeadway(int Time, string RouteName)
        {
           foreach (CrossOverInfo rule in CrossInfoList)
           {
               if (false == rule.CheckHeadway(Time, RouteName, H_DA_R, H_AD_R) )
                   return false;
           }
           return true;
        }


        public override bool Arrival(ref TrainEvent TE, bool IsFollowingTrainPassing)
        {
            if (Directions.ASC == TE.Dir)
                return Arrival(ref TE, IsFollowingTrainPassing, ref m_track1, ref m_track3);
            else
                return Arrival(ref TE, IsFollowingTrainPassing, ref m_track2, ref m_track3);
        }
        private bool Arrival(ref TrainEvent TE, bool IsFollowingTrainPassing, ref Tracks Pri, ref
Tracks Sec)
        {
            if (IsFollowingTrainPassing)
            {
                if (CheckTrackNoneOccupy(ref Sec) && CheckDA(ref Sec, TE.Time) && CheckAA(ref TE,
ref Pri) && CheckCrossoverHeadway(ref TE, Sec.Name))
                {
                    Sec.SetLastEvent(ref TE);
                    UpdateCrossoverInfo(ref TE, Sec.Name);
                    return true;
                }else if ( CheckTrackNoneOccupy(ref Pri) && CheckDA(ref Pri, TE.Time) &&
CheckAA(ref TE, ref Sec) && CheckCrossoverHeadway(ref TE, Pri.Name) )
                {
                    Pri.SetLastEvent(ref TE);
                    UpdateCrossoverInfo(ref TE, Pri.Name);
                    return true;
                }else
                {
                    return false;
                }
            }else
            {
                if (CheckTrackNoneOccupy(ref Pri) && CheckDA(ref Pri, TE.Time) && CheckAA(ref TE,
ref Sec) && CheckCrossoverHeadway(ref TE, Pri.Name))
                {
                    Pri.SetLastEvent(ref TE);
                    UpdateCrossoverInfo(ref TE, Pri.Name);
```

```
                return true;
            }
            else if (CheckTrackNoneOccupy(ref Sec) && CheckDA(ref Sec, TE.Time) && CheckAA(ref
TE, ref Pri) && CheckCrossoverHeadway(ref TE, Sec.Name))
            {
                Sec.SetLastEvent(ref TE);
                UpdateCrossoverInfo(ref TE, Sec.Name);
                return true;
            }
            else
            {
                return false;
            }
        }
    }

    public override bool Depature(ref TrainEvent TE)
    {
        if (Directions.ASC == TE.Dir)
            return Depature(ref TE, ref m_track1, ref m_track3);
        else
            return Depature(ref TE, ref m_track2, ref m_track3);
    }

    private bool Depature(ref TrainEvent TE, ref Tracks Pri, ref Tracks Sec)
    {
        Debug.Assert((Pri.Occupied == TE.TrainID && Sec.Occupied != TE.TrainID) ||
(Pri.Occupied != TE.TrainID && Sec.Occupied == TE.TrainID));
        if (Pri.Occupied == TE.TrainID)
        {
            if (CheckDD(ref TE, ref Sec) && CheckCrossoverHeadway(ref TE, Pri.Name) )
            {
                Pri.SetLastEvent(ref TE);
                UpdateCrossoverInfo(ref TE, Pri.Name);
                return true;
            }
            else
            {
                return false;
            }
        }
        else
        {
            if (CheckDD(ref TE, ref Pri) && CheckCrossoverHeadway(ref TE, Sec.Name))
            {
                Sec.SetLastEvent(ref TE);
                UpdateCrossoverInfo(ref TE, Sec.Name);
                return true;
            }
            else
            {
                return false;
            }
        }
    }

    public override bool Passing(ref TrainEvent TE)
    {
        if (Directions.ASC == TE.Dir)
            return Passing(ref TE, ref m_track1, ref m_track3);
        else
            return Passing(ref TE, ref m_track2, ref m_track3);
    }

    private bool Passing(ref TrainEvent TE, ref Tracks Pri, ref Tracks Sec)
    {
        if (CheckTrackNoneOccupy(ref Pri) && CheckDA(ref Pri, TE.Time) && CheckAP_DP(ref TE,
ref Sec) && CheckCrossoverHeadway(ref TE, Pri.Name))
        {
            Pri.SetLastEvent(ref TE);
            UpdateCrossoverInfo(ref TE, Pri.Name);
            return true;
        }
```

```
            else
            {
                return false;
            }
        }

        public override void SetTrackBlock(string track, bool block)
        {
            Debug.Assert(track != "Track4");
            if (Track3.Name == track)
            {
                Track3.Blocked = block;
            }
            else
            {
                base.SetTrackBlock(track, block);
            }
        }
    }

    public class StationIII_L : Station
    {
        private Tracks m_track3;
        public Tracks Track3
        {
            get { return m_track3; }
        }

        public StationIII_L()
        {
            m_track3 = new Tracks("Track3");
        }

        public override string GetStationType()
        {
            return "Type III_L";
        }

        public override bool IsAllTrackEmpty()
        {
            if (null == m_track1.Occupied && null == m_track2.Occupied && null ==
m_track3.Occupied)
                return true;
            else
                return false;
        }

        public override bool Arrival(ref TrainEvent TE, bool IsFollowingTrainPassing)
        {
            return true;
        }
        public override bool Depature(ref TrainEvent TE)
        {
            return true;
        }
        public override bool Passing(ref TrainEvent TE)
        {
            return true;
        }
    }


    public class CrossOverInfo
    {
        private int m_LastFirstRouteTime=-RailSystem.DayTime;
        private string m_FirstRoute;
        private string m_SecondRoute;
        bool m_Is_H_DA_R;
        public CrossOverInfo(string FirstRoute, string SecondRoute, bool Is_H_DA_R)
        {
            m_FirstRoute = FirstRoute;
            m_SecondRoute = SecondRoute;
            m_Is_H_DA_R = Is_H_DA_R;
```

```
        }
        public void UpdateFirstRoute(int NowTime, string RouteName)
        {
            if (m_FirstRoute == RouteName)
                m_LastFirstRouteTime = NowTime;
        }


        public void Reset()
        {
            m_LastFirstRouteTime = -RailSystem.DayTime;
        }
        public bool CheckHeadway(int NowTime, string RouteName, int H_DA_R, int H_AD_R)
        {
            if (m_SecondRoute == RouteName )
            {
                if (m_Is_H_DA_R)
                {
                    if ((NowTime - m_LastFirstRouteTime) >= H_DA_R)
                        return true;
                    else
                        return false;
                }
                else
                {
                    if ((NowTime - m_LastFirstRouteTime) >= H_AD_R)
                        return true;
                    else
                        return false;
                }
            }
            else
            {
                return true;
            }
        }
    }
}
```

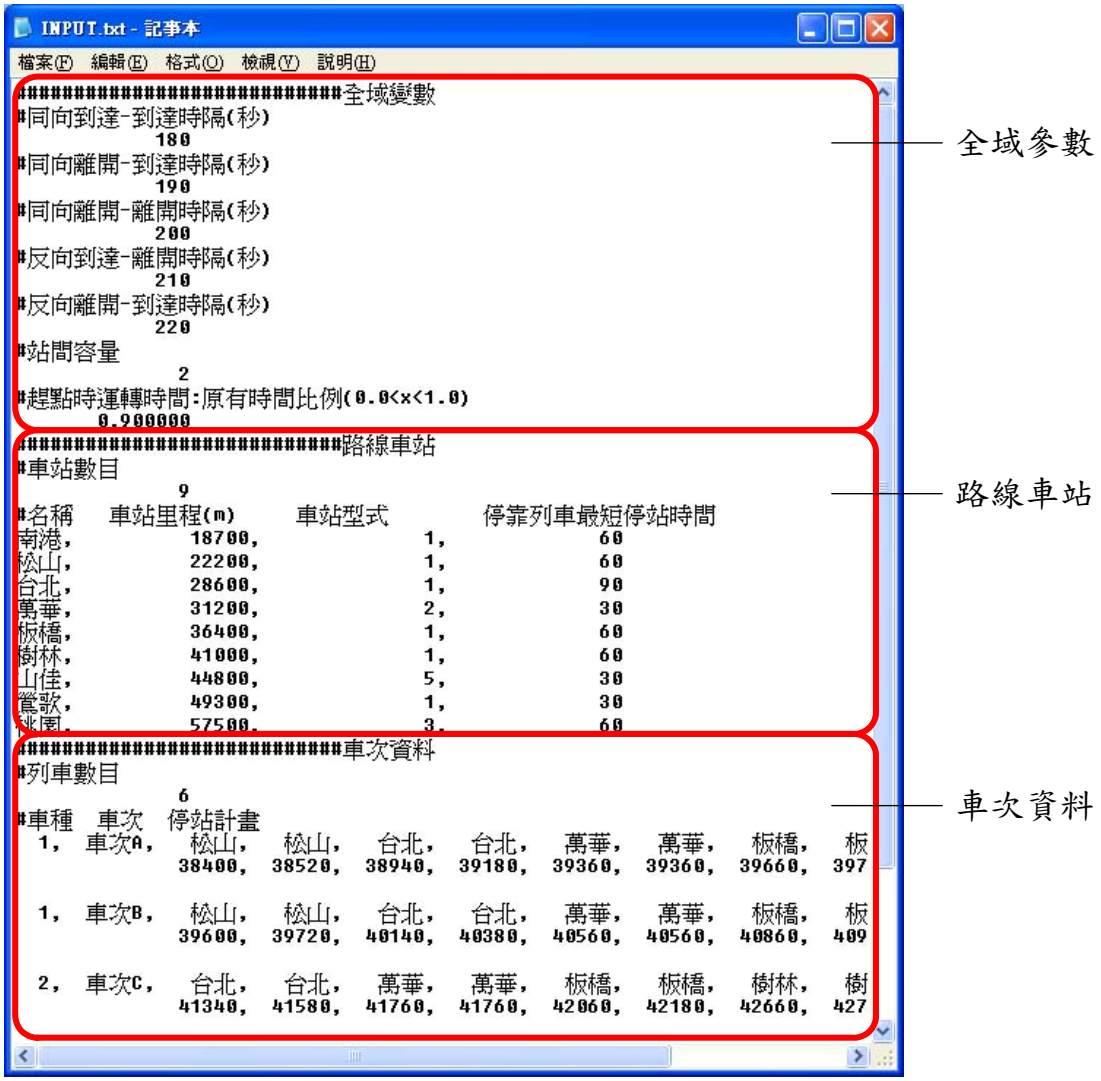
附錄二 原始資料輸入及模擬結果輸出說明

一、模式輸入

本程式須從輸入檔讀入資料後方能進行模擬，而輸入檔內容必須符合格式，才能使程式正確讀入進行分析，以獲得正確結果，本章將介紹輸入檔格式以及程式輸入介面的操作方式。

(一)輸入檔格式

程式的輸入檔內容如附圖 1 所示。



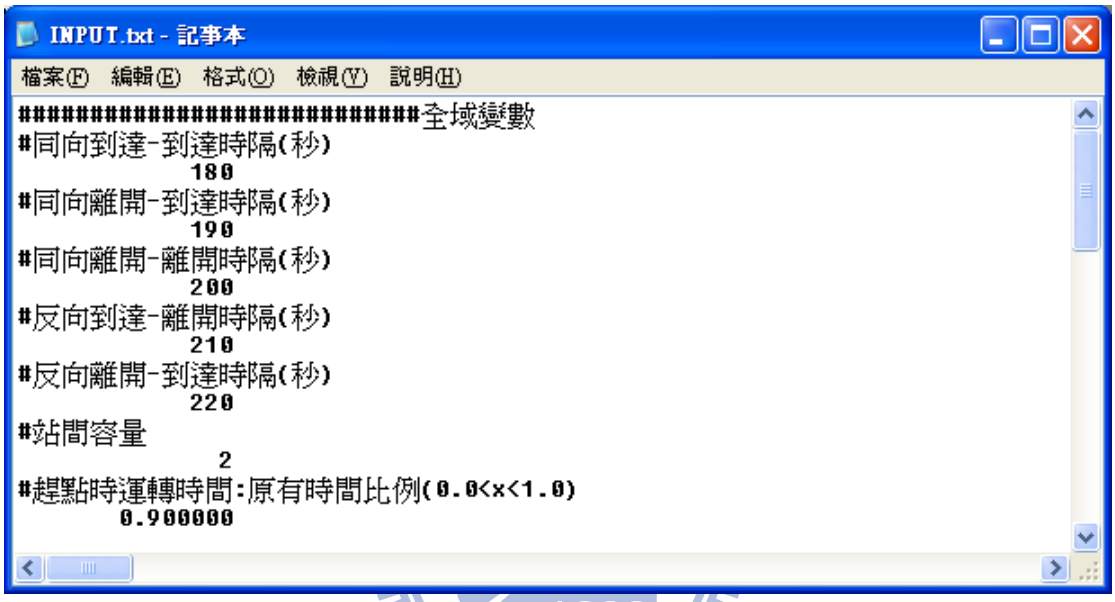
附圖 1 程式輸入檔

其中以「#」符號開始的為註解，程式在讀取輸入檔時，會自動略過該行之資料，此功能可供使用者加入額外資訊以供檢視。例如圖中第二行「#同向到達-到達

時隔（秒）」，便是用來說明第三行的「180」的意義。一份完整的輸入檔共可分為三個主要片段，分別為全域參數、路線車站和車次等資料，將依序介紹於後續章節。

(二)全域變數資料

全域變數的格式如附圖 2，所包含的各項參數與其相關說明限制彙整於附表 1，各項參數必須以空白、換行或「,」符號區隔，且須按照順序填寫。



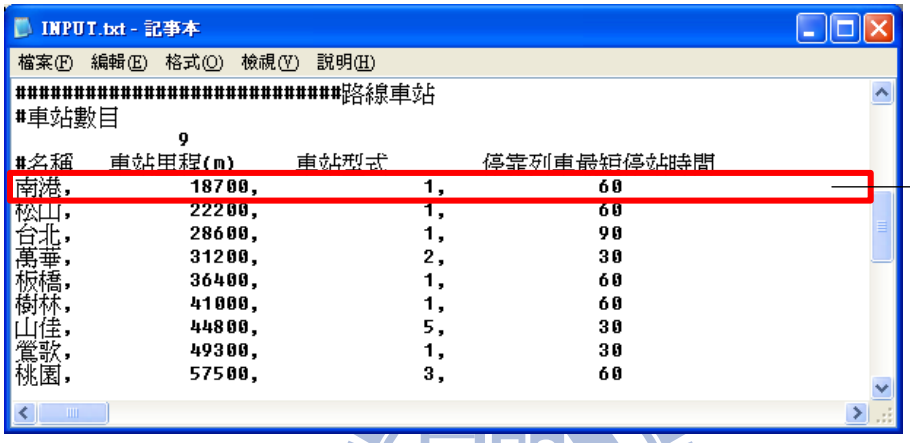
附圖 2 全域變數輸入格式

附表 1 全域變數資料參數型態、說明與限制彙整表

參數	型態	說明	限制
同向到達-到達時隔	整數	以秒為單位，連續兩列同向列車進站所需保持的時隔	大於或等於 0
同向離開-到達時隔	整數	以秒為單位，先行列車離站後，到續行列車進站之間所需保持的時隔	大於或等於 0
同向離開-離開時隔	整數	以秒為單位，連續兩列同向列車離站所需保持的時隔	大於或等於 0
反向到達-離開時隔	整數	以秒為單位，平面交叉時隔	大於或等於 0
反向離開-到達時隔	整數	以秒為單位，平面交叉時隔	大於或等於 0
站間容量	整數	站間軌道可容納的列車數	大於或等於 0
趕點比例	實數	列車於站間進行趕點時，其最小運轉時間與基準運轉時間的比率	介於 0～1 之間

(三)路線車站資料

路線車站的資料片段格式如附圖 3 所示，首先為欲分析路線的車站總數，接著為各車站的資料，每一筆車站資料包含車站名稱、里程、車站型式，和停靠列車最短停站時間等參數，其相關說明限制彙整於附表 2，各項參數必須以空白、換行或「,」符號區隔，且須按照順序填寫。此外需特別注意的是，各車站資料需以里程遞增的次序填寫。



附圖 3 路線車站輸入格式

附表 2 路線車站資料參數型態、說明與限制彙整表

參數	型態	說明	限制
車站名稱	文字	車站的名稱	不得含空白、「,」等特殊字元
車站里程	整數	以公尺為單位，車站的里程	大於 0
車站型式	列舉	車站內軌道佈設型式	註
停靠列車最短停站時間	整數	以秒為單位，若列車停靠該車站，最少的停站時間	大於或等於 0

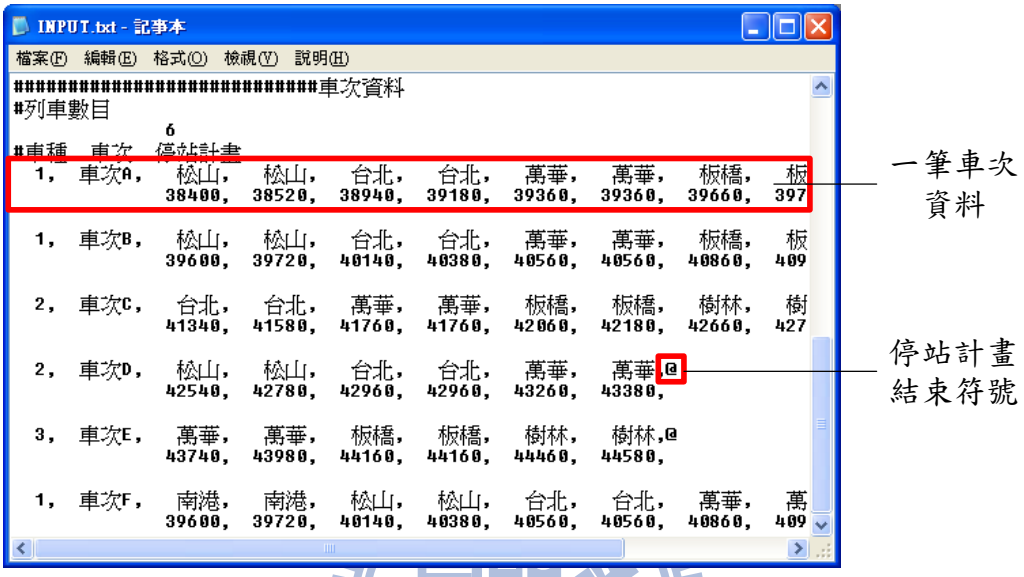
註：本程式考量 5 種佈設型式，以數字 1～5 表示。

(四)車次資料

在車次資料的片段中，資料格式如附圖 4，首先為列車數目，表示所有車次的總數，接著為各車次資料，每一筆車次資料佔兩行，每筆車次資料之間必須空一行，每一筆車次資料皆包含車種、車次名稱和停站計畫，其相關說明限制彙整於附表 3。

每筆車次資料的各項參數必須以空白、換行或「,」符號區隔，並且須按照順序填寫，首先為車種，其次為車次名稱，最後為停站計畫，其中停站計畫包含該

車所經過車站的進站時間與離站時間，如附圖 4 所示，於車次名稱之後開始填寫該車所經過的車站名稱，而第二行填寫於該車站的進站與離站時間。在此需特別注意，對於每一車站皆要有進站與離站時間，若該列車為通過該站，則於該站的進站與離站時間為相等，此外在最後一個離站的車站名稱之後需加上「@」符號，以表示結束。



附圖 4 車次資料輸入格式

附表 3 車次資料參數型態、說明與限制彙整表

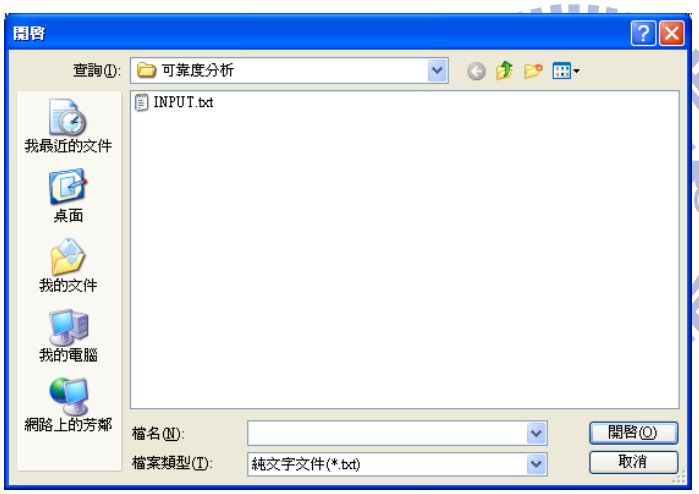
參數		型態	說明	限制
車種		整數	由使用者自行定義不同的數字代表不同的車種	大於 0
車次名稱		文字	車次的名稱	不得含空白、「,」等特殊字元
停站計畫	站	文字	列車進/離站所經過的車站名稱	名稱以及順序必須和「路線車站」所定義的資料一致
	時	整數	列車在各車站進/離站的時間，從 0 點開始計算，以秒為單位，例如早上 6 點為 21600 秒	介於 0～86399 之間

(五)程式輸入介面

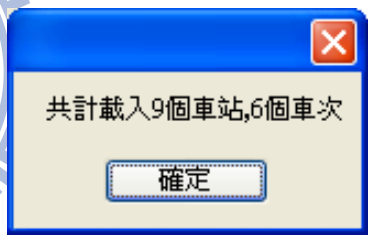
程式介面如附圖 5，選擇左上角【開啟輸入檔】按鈕，會出現附圖 6 之對話盒來協助使用者選擇欲讀取的輸入檔，選定輸入檔後按下【開啟】按鈕，程式便會

讀取輸入檔，當讀取成功後會出現附圖 7 之訊息，顯示該輸入檔所包含的車站數與車次數，若該數量與輸入檔的設定有所出入，請重新檢視輸入檔設定是否有誤。

附圖 5 程式介面



附圖 6 開啟輸入檔對話盒



附圖 7 輸入檔讀取成功之訊息

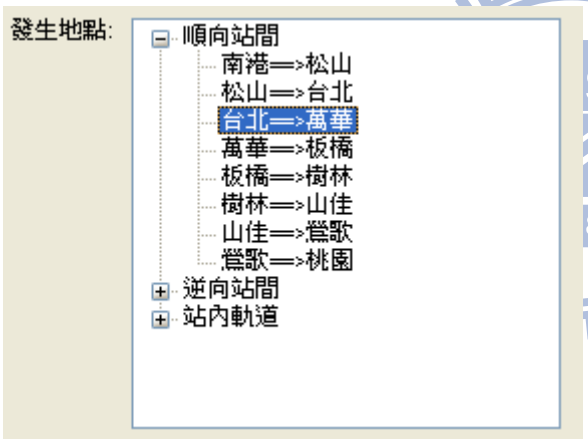
二、模擬執行

本程式係針對一可行班表，模擬初始延滯(亦稱主要延滯)發生對整體班表的影響，因此模擬的執行過程須先設定初始延滯的時間地點，此外還可選擇不同運轉整理策略，以比較不同策略對可靠度的改善效果。

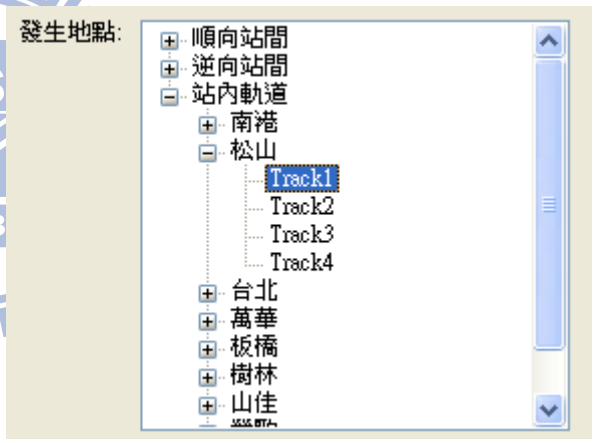
在程式的介面中，設定初始延滯的區域在介面的右方，如附圖 8，在該區中可設定初始延滯的開始時間、嚴重程度和發生地點。其中開始時間表示初始延滯發生的時間，嚴重程度則是表示初始延滯的持續時間，發生地點則可選擇初始延滯

是發生在站間軌道還是站內軌道，分別如附圖 9 和附圖 10。此外，本程式另外提供時間格式轉換小工具，其介面如附圖 11，協助使用者轉換不同的時間格式，免去換算的麻煩。

附圖 8 設定初始延滯



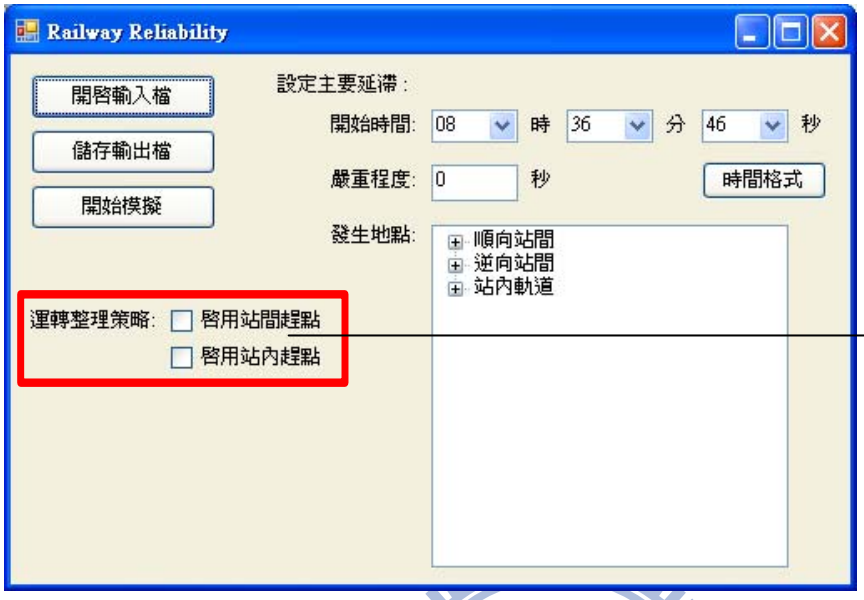
附圖 9 選擇初始延滯發生在站間軌道



附圖 10 選擇初始延滯發生在站內軌道

附圖 11 時間格式轉換小工具

在運轉整理策略方面，本程式提供兩種策略供使用者選用，分別是(1)站間趕點和(2)站內趕點，於介面上直接勾選便可啟用，如附圖 12，兩種策略互相獨立，可同時選用。



附圖 12 選擇運轉整理策略

在完成初始延滯的設定與運轉整理策略的選擇後，於介面上按下【開始模擬】按鈕，程式即開始進行模擬運算，當附圖 13 之訊息出現後，表示模擬運算完成。

附圖 13 完成模擬之訊息

三、模式輸出

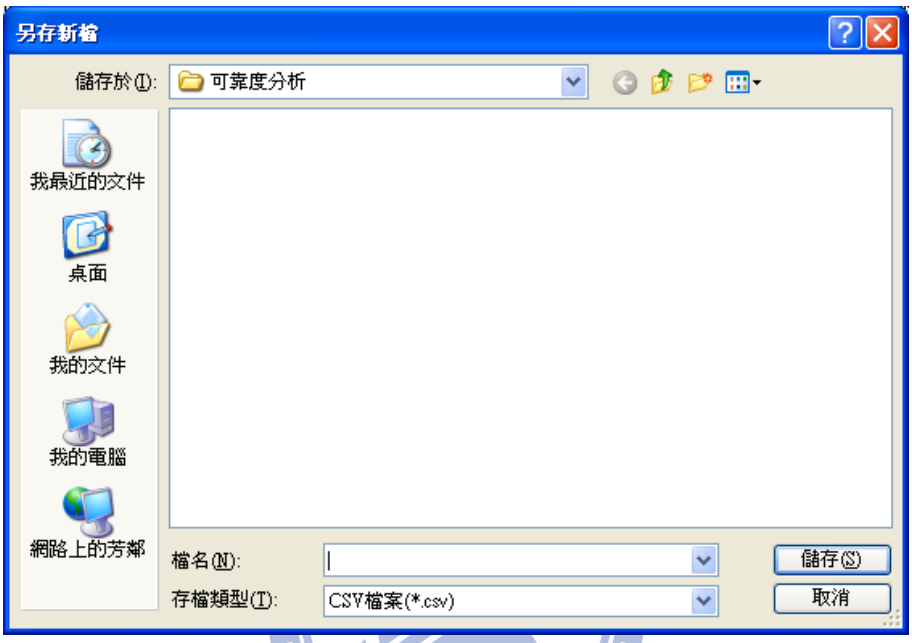
本程式可將模擬結果輸出成文字檔，但是僅從文字檔較難看出初始延滯對列車運行的影響，因此本研究另外利用 MS EXCEL 製作了 EXCEL 工具，供模擬結果後處理使用。

(一)程式輸出介面

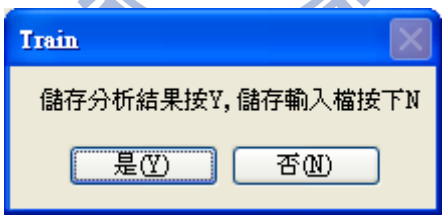
在程式介面（如附圖5）選擇【儲存輸出檔】按鈕，會出現附圖 14 之對話盒來協助使用者進行程式輸出工作，透過對話盒選定路徑與指定檔名後，按下【儲

存】按鈕後，會再出現附圖 15 之對話盒，若要儲存模擬分析的結果就按【是】；若要儲存原輸入檔的內容則按【否】，如此便能完成程式輸出工作。

程式的輸出檔類型為「CSV」檔案，以 Notepad 開啟的畫面如附圖 16，此外亦可由 MS EXCEL 直接開啟。其格式內容和輸入檔相同，唯獨在車次資料的部分是列出在完成模擬後，各列車於各車站的到離時間，而非原本輸入檔中的時間。



附圖 14 儲存輸出檔對話盒

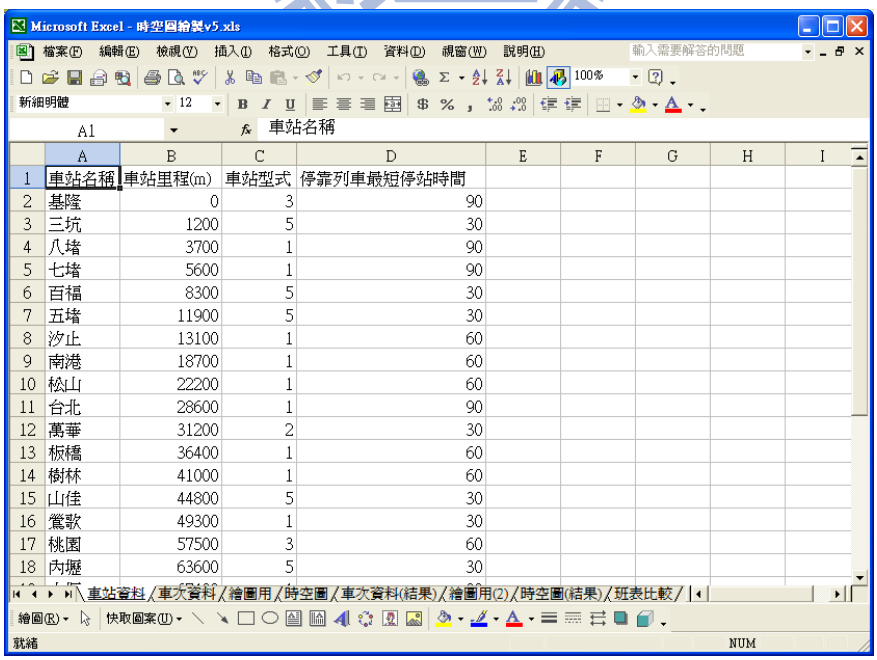


附圖 15 選擇輸出項目對話盒

附圖 16 輸出檔格式

(二)EXCEL工具

本研究所製作的 EXCEL 工具如附圖 17，一共包含 8 個工作表、2 個表單和 6 個巨集，各工作表的內容說明如附表 4，本工具提供時空圖繪製和班表比較兩項功能，其操作方式分別說明於下。



	A	B	C	D
1	車站名稱	車站里程(m)	車站型式	停靠列車最短停站時間
2	基隆	0	3	90
3	三坑	1200	5	30
4	八堵	3700	1	90
5	七堵	5600	1	90
6	百福	8300	5	30
7	五堵	11900	5	30
8	汐止	13100	1	60
9	南港	18700	1	60
10	松山	22200	1	60
11	台北	28600	1	90
12	萬華	31200	2	30
13	板橋	36400	1	60
14	樹林	41000	1	60
15	山佳	44800	5	30
16	鶯歌	49300	1	30
17	桃園	57500	3	60
18	內壢	63600	5	30

附圖 17 EXCEL 工具介面

附表 4　EXCEL 工具中各工作表之內容說明

工作表名稱	內容說明
車站資料	輸入檔中各車站的資料
車次資料	輸入檔中各車次的資料
繪圖用	供繪製時空圖內部使用，勿隨意更改
時空圖	輸入檔中各車次的運行時空圖
車次資料（結果）	輸出檔中各車次的資料
繪圖用(2)	供繪製時空圖內部使用，勿隨意更改
時空圖（結果）	輸出檔中各車次的運行時空圖
班表比較	根據所選車次比較發生延滯前後的時空圖

(三)輸入資料

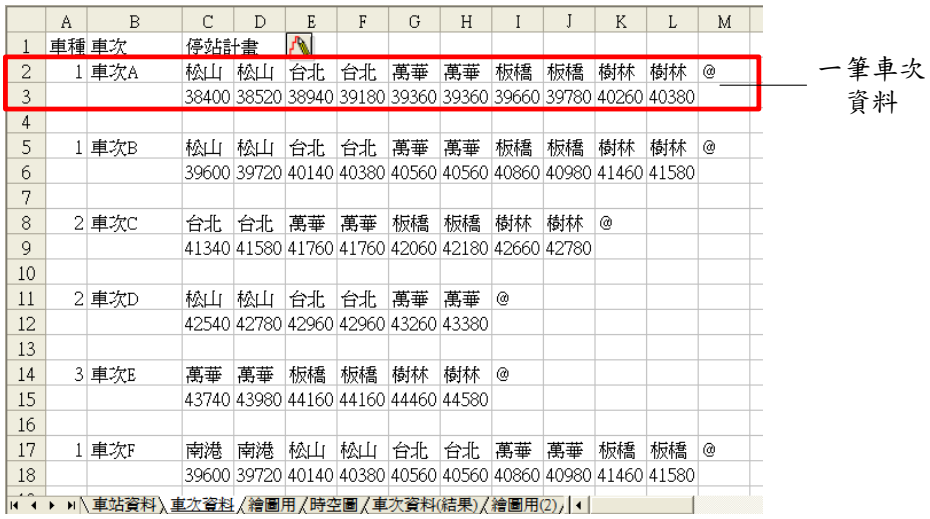
在進行時空圖繪製和班表比較之前，需先將程式的輸入與輸出檔的資料輸入到 EXCEL 工具中，首先在「車站資料」工作表中輸入各車站資料，如附圖 18，第一筆車站資料必須從儲存格 A2～D2 開始，依序填寫車站名稱、里程、軌道型式和最短停站時間，每筆車站資料佔用一行，且各車站必須按里程遞增次序填寫。

	A	B	C	D
1	車站名稱	車站里程(m)	車站型式	停靠列車最短停站時間
2	基隆	0	3	90
3	三坑	1200	5	30
4	八堵	3700	1	90
5	七堵	5600	1	90
6	百福	8300	5	30
7	五堵	11900	5	30
8	汐止	13100	1	60
9	南港	18700	1	60
10	松山	22200	1	60
11	台北	28600	1	90
12	萬華	31200	2	30
13	板橋	36400	1	60
14	樹林	41000	1	60
15	山佳	44800	5	30
16	鶯歌	49300	1	30
17	桃園	57500	3	60
18	內壢	63600	5	30

車站資料 / 車次資料 / 繪圖用 / 時空圖 / 車次資料(結果) / 繪圖用(2)

附圖 18 輸入車站資料

接著於「車次資料」工作表中填入輸入檔的車次資料，如附圖 19，第一筆車次資料必須從第 2 行開始，每筆車次資料佔用兩行，各車次資料間需空一行。每一車次資料於第一行 A 欄填寫車種，B 欄填寫車次名稱，C 欄以後填寫該車次各到離站的車站名稱，而第二行 C 欄以後填寫於各站到離站的時間。



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	車種	車次	停站計畫										
2	1	車次A	松山	松山	台北	台北	萬華	萬華	板橋	板橋	樹林	樹林	@
3			38400	38520	38940	39180	39360	39360	39660	39780	40260	40380	
4													
5	1	車次B	松山	松山	台北	台北	萬華	萬華	板橋	板橋	樹林	樹林	@
6			39600	39720	40140	40380	40560	40560	40860	40980	41460	41580	
7													
8	2	車次C	台北	台北	萬華	萬華	板橋	板橋	樹林	樹林	@		
9			41340	41580	41760	41760	42060	42180	42660	42780			
10													
11	2	車次D	松山	松山	台北	台北	萬華	萬華	@				
12			42540	42780	42960	42960	43260	43380					
13													
14	3	車次E	萬華	萬華	板橋	板橋	樹林	樹林	@				
15			43740	43980	44160	44160	44460	44580					
16													
17	1	車次F	南港	南港	松山	松山	台北	台北	萬華	萬華	板橋	板橋	@
18			39600	39720	40140	40380	40560	40560	40860	40980	41460	41580	

車站資料 / 車次資料 / 繪圖用 / 時空圖 / 車次資料(結果) / 繪圖用(2) /

一筆車次資料

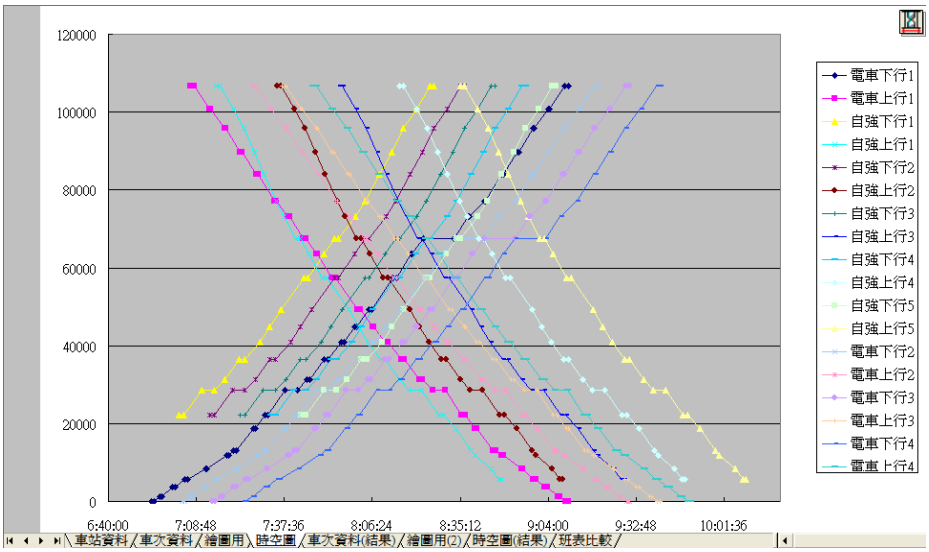
附圖 19 輸入車次資料

最後在「車次資料（結果）」工作表中填寫輸出檔的車次資料，該表的格式和「車次資料」工作表相同，填寫的原則亦相同，在此不再贅述。

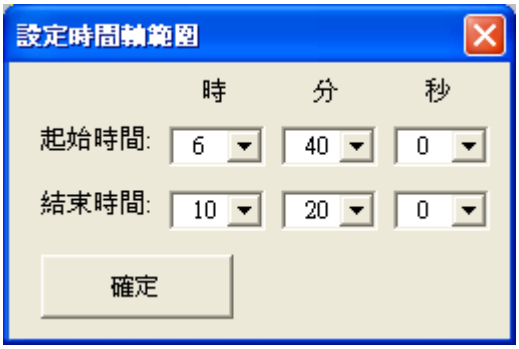
(四)時空圖繪製

當資料輸入完成後，於「車次資料」和「車次資料（結果）」工作表中，按下【  】圖示，此時便呼叫巨集繪製時空圖，其結果分別呈現於「時空圖」和「時空圖（結果）」工作表中，如附圖 20。

欲調整時空圖檢視的時間範圍，可在「時空圖」或「時空圖（結果）」工作表右上方，按下【  】圖示，會出現附圖 21 對話盒，設定起始時間和結束時間後按下【確定】按鈕，兩工作表中的時空圖時間範圍便會根據設定同時進行調整。



附圖 20 時空圖繪製結果

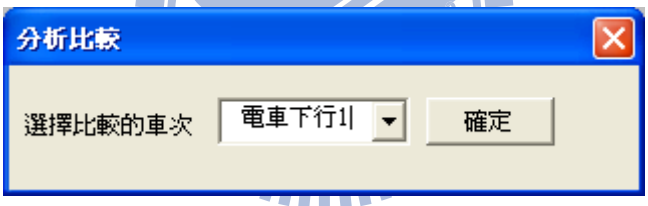


附圖 21 設定時間軸範圍

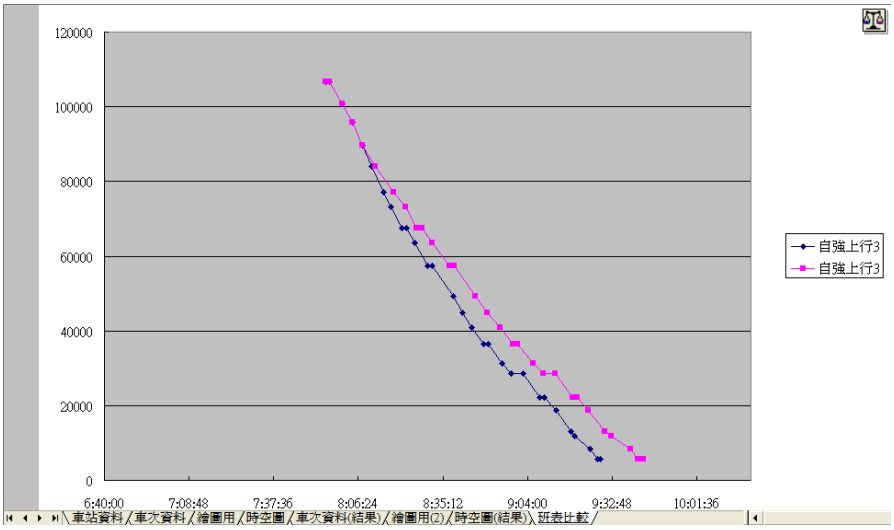
(五)班表比較

當時空圖繪製完成後，於 EXCEL 工具中切換「時空圖」或「時空圖（結果）」兩工作表，可達到班表比較的效果，但是當車次數量龐大時，使用者將難以判斷其差異，因此在本工具中另外提供針對某單一車次的班表比較功能。

按下「班表比較」工作表中右上方的【  】圖示，會出現附圖 22 之對話盒，從下拉式選單中選擇欲比較的車次後按下【確定】按鈕，畫面中將該車次原始與模擬後的班表同時繪在一起，如附圖 23，使用者便可一目了然該車次的延滯情形。



附圖 22 選擇比較的車次



附圖 23 班表比較結果

作 者 簡 歷

姓名：劉昭榮 (Liu, Jau-Rong)
地址：251 新北市淡水區自立路15巷9號14樓
電話：02-26241682
E-Mail：felix@iot.gov.tw

學歷

國立交通大學交通運輸研究所 博士
國立成功大學交通管理科學研究所 碩士
國立交通大學運輸工程與管理系 學士
高雄市立高雄中學 畢業

經歷

交通部運輸研究所研究員
交通部運輸研究所副研究員
交通部運輸研究所助理工程司

期刊論文

1. Liu, J. R. and Hwang, C. C. (2011), "The Influence of First-Delay Location and Timetable Recovery Strategies on Knock-on Delay of Taiwan Regional Railway", *Journal of the Eastern Asia Society for Transportation Studies*, Vol. 9.(已接受刊登)
2. 黃承傳、劉昭榮(民100)，「鐵路列車密度與初始延滯以及調度策略對連鎖延滯之影響分析」，**運輸計劃季刊**(TSSCI)，第40卷，第1期，頁63-98。
3. Hwang, C. C. and Liu, J. R. (2010), "A Simulation Model for Estimating Knock-on Delay of Taiwan Regional Railway", *Journal of the Eastern Asia Society for Transportation Studies,* Vol.8, pp. 1110-1125.
4. 鍾志成、李治綱、賴勇成、劉昭榮等人(民100)，「捷運系統路線容量分析－以臺北捷運高運量系統板南線為例」，**運輸學刊**（已接受刊登）。
5. 鍾志成、張恩輔、林國顯、劉昭榮(民100)，「捷運系統列車折返號誌時隔分析模式」，**運輸學刊**（已接受刊登）。

研討會論文

1. Hwang, C. C. and Liu, J. R. (2010), "A Simulation Model for Estimating Knock-on Delay of Taiwan Regional Railway", EASTS Conference 2009.
2. 鍾志成、李治綱、劉昭榮等人(民99)，「臺灣軌道容量分析發展與成果」，The 23rd ICTPA Annual Meeting。

3. 鍾志成、李治綱、賴勇成、劉昭榮等人(民99)，「捷運系統路線容量分析－以臺北捷運高運量系統板南線為例」，中華民國運輸學會25屆論文研討會論文集。
4. 鍾志成、張恩輔、林國顯、劉昭榮(民99)，「捷運系統列車折返號誌時隔分析模式」，中華民國運輸學會25屆論文研討會論文集。
5. 鍾志成、李治綱、劉昭榮等人(民98)，「臺鐵列車服務可靠度分析模式之研究」，中華民國運輸學會第24屆論文研討會論文集，頁1713-1732。
6. 鍾志成、李治綱、劉昭榮等人(民97)，「以定價敏感計量法探討區域鐵路系統之服務品質」，中華民國運輸學會23屆論文研討會論文集，頁1467-1484。

獲獎

1. 鍾志成、張恩輔、林國顯、劉昭榮合著之「捷運系統列車折返號誌時隔分析模式」論文，榮獲中華民國運輸學會99年度「運輸年會實務論文獎」。
2. 張有恆、劉昭榮合著之「航空公司飛航營運排班多目標規劃模式之研究」論文，榮獲中華民國運輸學會86年度「年度期刊論文獎」。