การทำนายการใช้กำลังไฟฟ้าของประเทศไทยโดยการใช้โครงข่ายประสาทเทียม

LOAD FORECASTING OF THAILAND BY MEANS OF ARTIFICIAL NEURAL NETWORKS

โดย

นายกิตติโชติ อุดมประเสริฐกุล

นายภุชงค์ อุปัญญ์

นายมโนทอง วงศ์ลอคำ

นายอาคม เลียมไชสง

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2541

ปีการศึกษา 2541

การทำนายการใช้กำลังไฟฟ้าของประเทศไทยโดยการใช้โครงข่ายประสาทเทียม

LOAD FORECASTING OF THAILAND BY MEANS OF ARTIFICIAL NEURAL NETWORKS

โดย

นายกิตติโชติ อุดมประเสริฐกุล

นายภุชงค์ อุปัญญ์

นายมโนทอง วงศ์ลอคำ

นายอาคม เลียมไธสง

อาจารย์ที่ปรึกษา

รศ. ศิริวัฒน์ โพธิเวชกุล

ปริญญานิพนธ์ปีการศึกษา 2541

ภาควิชาวิศวกรรมไฟฟ้า

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การทำนายการใช้กำลังไฟฟ้าของประเทศไทยโดยการใช้โครงข่ายประสาทเทียม

ผู้จัดทำ

1. นายกิตติโชติ อุดมประเสริฐกุล
2. นายภุชงค์ อุปัญญ์
3. นายมโนทอง วงศ์ลอคำ
4. นายอาคม เลียมไธสง

อาจารย์ที่ปรึกษา

(รศ. ศิริวัฒน์ โพธิเวชกุล)

การทำนายการใช้กำลังไฟฟ้าของประเทศไทยโดยการใช้โครงข่ายประสาท

กิตติโชติ อุดมประเสริฐกุล
ภูชงค์ อุปัญญ์
มโนทอง วงศ์ลอคำ
อาคม เลียมไธสง
รศ. ศิริวัฒน์ โพธิเวชกุล อาจารย์ที่ปรึกษา
ปีการศึกษา 2541

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้ เสนอถึงวิธีการประยุกต์ใช้โครงข่ายประสาทเทียมในการทำนายความต้องการกำลังไฟฟ้าของประเทศไทยตั้งแต่วันจันทร์ถึงวันศุกร์ภายในระยะเวลา 3 ชั่วโมงล่วงหน้า การทำนายความต้องการกำลังไฟฟ้าช่วงระยะเวลาสั้นกำลังได้รับความสนใจจากผู้ผลิตกระแสไฟฟ้า เนื่องจากจะช่วยให้การผลิตกำลังไฟฟ้าได้ใกล้เคียงกับความต้องการที่แท้จริงนำมาซึ่งการประหยัดค่าใช้จ่ายในการผลิตกระแสไฟฟ้า โครงงานนี้ได้ใช้ค่าปัจจัยทางสภาวะอากาศเป็นตัวแปรอินพุทในการทำนายโดยไม่พิจารณาถึงสภาวะทางเศรษฐกิจและพบว่าผลจากการทำนายมีค่าความผิดพลาดเฉลี่ยต่ำกว่า 3 % ทั้งนี้เพื่อความสะดวกในการรับส่งข้อมูลอินพุทกับระบบฐานข้อมูลออราเคิลและการใช้งานจึงได้เชื่อมโยงโปรแกรมทั้งหมดเข้าสู่ระบบเครือข่ายอินเตอร์เน็ต

Electric Demand Forecasting of Thailand by Means of Neural Networks

Kittichote Udomprasertkul

Puchong Upun

Manothong Wonglorkham

Arkom Liamthaisong

Assoc. Prof. Siriwat Potivejkul Advisor

1998

Abstract

This thesis presents the approach to apply the artificial neural networks to forecast the electric demand of Thailand from Monday to Friday within 3 hours ahead. Short term load-forecasting is more interesting from many utilities. Because it can help utilities to generate electric power close to real demand and bring to saving operation costs. By neglecting the economic factors, this project utilize weather conditions as input parameters and the results have average error less than 3 percent. Oracle-database using Internet Networks linked all programs is the convenient ways for users to input data from everywhere.

สารบัญ

สารบัญรูป

สารบัญตาราง

บทที่ 1
บทนำ

เนื่องจากความเจริญก้าวหน้าของเทคโนโลยีทางด้านปัญญาประดิษฐ์ (Artificial Intelligence) [2] สาขาต่างๆในปัจจุบัน อาทิเช่น ระบบฟัซซี่ ลอจิค (Fuzzy Logic System), เจเนติค อัลกอริทึ่ม (Ganetic Algorithm)และโครงข่ายประสาทเทียม (Artificial Neural Networks) เป็นต้น ได้ก่อให้เกิดระบบซึ่งมีขีดความสามารถในการตัดสินปัญหาได้เองในขอบเขตระดับหนึ่ง กอปรกับการพัฒนาทางด้านคอมพิวเตอร์อย่างต่อเนื่องทำให้คอมพิวเตอร์ในปัจจุบันมีประสิทธิภาพสูง จึงได้เกิดแนวความคิดในการนำเทคโนโลยีดังกล่าวเข้ามาประยุกต์ใช้ในการแก้ปัญหาหรือการวิเคราะห์ในระบบไฟฟ้ากำลัง

ในปัจจุบัน จากการประสบความสำเร็จในการลอกเลียนแบบระบบการทำงานของเซลล์ประสาทของสิ่งมีชีวิตได้ก่อให้เกิดระบบโครงข่ายประสาทเทียมขึ้นซึ่งมีขีดความสามารถในการตัดสินใจคล้ายคลึงกับการตัดสินใจของมนุษย์เป็นอย่างมาก และจากประสิทธิภาพการทำงานของเครื่องคอมพิวเตอร์ที่สูงในปัจจุบัน จึงได้เกิดการประยุกต์นำเอาเทคโนโลยีโครงข่ายประสาทเทียมและคอมพิวเตอร์เข้ามาประยุกต์ใช้งานในระบบไฟฟ้ากำลัง [3] อาทิเช่น การทำนายโหลด (Load Forecasting), การวิเคราะห์เสถียรภาพทรานเซียนของระบบทั้งในทางสแตติคส์ (Statics) และไดนามิคส์ (Dynamics), การประมาลสัญญาณบอกเหตุ, การวิเคราะห์ตำแหน่งความผิดปกติหรือฟอลท์ที่เกิดขึ้นในระบบไฟฟ้ากำลัง [6], การวิเคราะห์การวางแผนในระบบไฟฟ้ากำลัง เป็นต้น โดยในปริญญานิพนธ์ฉบับนี้จะกล่าวถึง การนำโครงข่ายประสาทเทียมมาประยุกต์ใช้ในการทำนายโหลดหรือการทำนายการใช้กำลังไฟฟ้า โดยอาศัยข้อมูลทางสภาวะอากาศเป็นหลักในการทำนาย โดยในชั้นนี้จะไม่ได้ทำการวิเคราะห์และทดลองถึงผลของการเปลี่ยนแปลงของโหลดตามสภาวะทางเศรษฐกิจ

ในระบบไฟฟ้ากำลังนั้น โรงจักรไฟฟ้าจะทำหน้าที่ในการผลิตกระแสไฟฟ้าและทำการส่งจ่ายกำลังไฟฟ้าไปยังโหลดโดยผ่านทางระบบสายส่ง (Transmission Line) ซึ่งในการผลิตกระแสไฟฟ้าโรงจักรไฟฟ้าแต่ละโรงจักรจำเป็นที่จะต้องทำการผลิตกำลังไฟฟ้าให้เพียงพอต่อความต้องการของโหลด ทั้งนี้ก็เนื่องมาจากหากโรงจักรไฟฟ้าไม่สามารถผลิตกำลังไฟฟ้าได้เพียงพอต่อความต้องการของโหลดแล้วย่อมทำให้เกิดผลเสียหาย อาทิเช่น เจนเนอร์เรเตอร์ (Genertor) เกิดความเสียหาย, ทำให้อุปกรณ์เครื่องใช้ได้รับความเสียหาย, เกิดความเสียหายทางเศรษฐกิจและอุตสาหกรรม เป็นต้น ในทางกลับกัน หากโรงจักรไฟฟ้าทำการผลิตกระแสไฟฟ้ามากเกินความต้องการของโหลดแล้ว ย่อมก่อให้เกิดปัญหาในด้านการใช้

ทรัพยากรธรรมชาติอย่างไม่มีประสิทธิภาพอันจะก่อให้เกิดปัญหาการขาดแคลนทรัพยากรธรรมชาติตามมาในภายหลัง อีกทั้งยังก่อเกิดต้นทุนการผลิตที่สูงขึ้นอันเนื่องมาจากการผลิตกระแสไฟฟ้าเกินความต้องการ เป็นต้น

ในอดีตนั้นการทำนายโหลดมีด้วยกันหลายวิธี ได้แก่ การทำนายโหลดโดยอาศัยการสำรวจความต้องการกำลังไฟฟ้าในแต่ละพื้นที่, การทำนายโหลดโดยอาศัยข้อมูลทางสถิติ, การทำนายโหลดโดยอาศัยสมการทางคณิตศาสตร์ และวิธีสุดท้ายคือ การทำนายโหลดโดยอาศัยสมการทางคณิตศาสตร์ร่วมกับการพิจารณาถึงสภาวะทางเศรษฐกิจ ซึ่งจะได้กล่าวถึงในบทต่อไป

ในการทำนายโหลดด้วยวิธีการดังที่ได้กล่าวมาแล้วในข้างต้น โดยส่วนใหญ่แล้วจะให้ผลการทำนายที่ให้ค่าความผิดพลาดค่อนข้างสูง ซึ่งค่าความผิดพลาดที่เกิดขึ้นดังกล่าวนั้นหากว่าเกิดขึ้นกับระบบไฟฟ้าที่มีขนาดใหญ่แล้วก็จะยิ่งทวีความผิดพลาดมากยิ่งขึ้นไปตามลำดับ จึงไม่เหมาะสมที่จะนำมาใช้ในระบบส่งจ่ายกำลังไฟฟ้าที่มีขนาดใหญ่ เช่น ในระบบไฟฟ้าของประเทศไทยนั้นเป็นระบบที่มีขนาดใหญ่เพียงระบบเดียว และมีความต้องการการใช้กำลังไฟฟ้ามาก ดังเช่น กำลังไฟฟ้าสูงสุดที่ทำการส่งจ่าย ณ วันที่ 22 มิถุนายน 2541 เวลา 14.00 น. นั้นมีความต้องการสูงถึง 13593.2 เมกะวัตต์ (จากรายงานของการไฟฟ้าฝ่ายผลิตประจำเดือนมิถุนายน ปี 2540) ซึ่งหากว่าค่าความผิดพลาดที่เกิดจากการทำนายมีเพียง 3 เปอร์เซ็นต์ก็จะทำให้ระบบมีการทำนายที่มีค่าความผิดพลาดได้ถึง 407.796 เมกะวัตต์ ซึ่งก่อให้เกิดความเสี่ยงในการผลิตกำลังไฟฟ้าเพื่อส่งจ่ายโหลดมากขึ้น แต่ในปัจจุบันเมื่อได้มีการทดลองการทำนายโหลดโดยอาศัยการประยุกต์ใช้โครงข่ายประสาทเทียมแล้ว ปรากฏว่าค่าความผิดพลาดที่เกิดขึ้นจากการทำนายโหลดนั้นมีค่าน้อยมาก ดังที่จะได้กล่าวต่อไป

1.1 วัตถุประสงค์ของโครงงานปริญญานิพนธ์

1.1.1 เพื่อทำการศึกษาถึงคุณลักษณะของโครงข่ายประสาทเทียม

1) ลักษณะการทำงานของโครงข่ายประสาทเทียม

2) ลักษณะการเรียนรู้ของโครงข่ายประสาทเทียม

1.1.2 เพื่อศึกษาความสัมพันธ์ของค่าพารามิเตอร์ที่มีผลต่อการเรียนรู้ของโครงข่ายประสาท ได้แก่

1) ค่าอัตราการเรียนรู้

2) ค่าสัมประสิทธิ์โมเมนตัม

1.1.3 เพื่อศึกษาถึงปัจจัยต่างๆทางสภาวะอากาศที่มีอิทธิพลต่อการเปลี่ยนแปลงของการใช้กำลังไฟฟ้าของประเทศไทย

1.1.4 เพื่อฝึกการทำงานร่วมกันเป็นกลุ่ม

1.2 ประโยชน์ที่ได้รับจากโครงงานปริญญานิพนธ์

1.2.1 ได้เรียนรู้ถึงหลักการการทำงานของโครงข่ายประสาทเทียมและสามารถเขียนโปรแกรมโครงข่ายประสาทเทียมได้

1.2.2 ได้เรียนรู้ถึงวิธีการต่างๆที่ใช้ในการทำนายการใช้กำลังไฟฟ้า

1.2.3 สามารถนำงานปริญญานิพนธ์ไปประยุกต์ใช้งานได้

1.2.4 ได้เรียนรู้เรื่องระบบฐานข้อมูล

1.3 ขอบเขตของโครงงานปริญญานิพนธ์

1.3.1 เขียนโปรแกรมโครงข่ายประสาทเทียมแบบฟีดฟอร์เวิร์ด (Feed Forward Neural Network) โดยใช้การเรียนรู้แบบแพร่ย้อนกลับ (Back-Propagation) ด้วยภาษาจาวา (Java Language) เพื่อใช้ในการทำนายการใช้กำลังไฟฟ้าของประเทศไทย

1.3.2 เขียนเวบ-เพจ (Web Page) เพื่อใช้ในการรับข้อมูลผ่านทางระบบอินเทอร์เน็ต (Internet) เข้ามาเก็บในระบบฐานข้อมูล (Database System) ด้วยภาษา เอชทีเอ็มแอล (HTML Language) และภาษา เอสคิวแอล (SQL-Language)

1.3.3 ทำการวิเคราะห์ผลของการทำนายโหลดโดยอาศัยการเขียนโปรแกรมแม็ทแล็บ(MATLAB)

บทที่ 2
โครงข่ายประสาท

คุณสมบัติที่สำคัญประการหนึ่งของสิ่งมีชีวิต ได้แก่ ความสามารถในการตอบสนองต่อสิ่งเร้าได้ทั้งในสิ่งมีชีวิตชั้นต่ำจนถึงสิ่งมีชีวิตชั้นสูง ซึ่งกระบวนการการตอบต่อสิ่งเร้าในสิ่งมีชีวิตชั้นสูงจะมีความซับซ้อนมากกว่าในสิ่งมีชีวิตชั้นต่ำ คือ จะใช้การประสานงานร่วมกันของเซลล์ประสาท, อวัยวะรับสัมผัสและการทำงานของอวัยวะต่างๆของร่างกายที่สอดคล้องกัน

การที่ระบบอวัยวะต่างๆของร่างกายสามารถทำงานได้สอดคล้องกันเพื่อให้ร่างกายอยู่ได้อย่างปกติและสามารถปรับตัวให้เข้ากับสภาวะแวดล้อมทั้งภายในและภายนอกได้ ก็เนื่องมาจากการควบคุมของระบบประสาทผ่านเซลล์ประสาทรับความรู้สึก (Sensory Neuron) เข้าสู่ศูนย์กลางของระบบประสาทในสมองและไขสันหลัง แล้วส่งสัญญาณที่เหมาะสมผ่านเซลล์ประสาทส่งความรู้สึก (Motor Neuron) ไปสู่เนื้อเยื่อที่ทำหน้าที่ตอบสนอง (Effector) ดังนั้นการทำงานของระบบประสาท ก็คือ การควบคุมระบบการทำงานของอวัยวะส่วนต่างๆให้สามารถทำงานได้อย่างสอดคล้องกันนั่นเอง

2.1 รูปร่างของเซลล์ประสาท

เซลล์ประสาทโดยทั่วไปจะประกอบด้วยส่วนที่สำคัญ 2 ส่วน คือ

2.1.1 ตัวเซลล์ (Cell Body หรือ Soma หรือ Perikaryon)

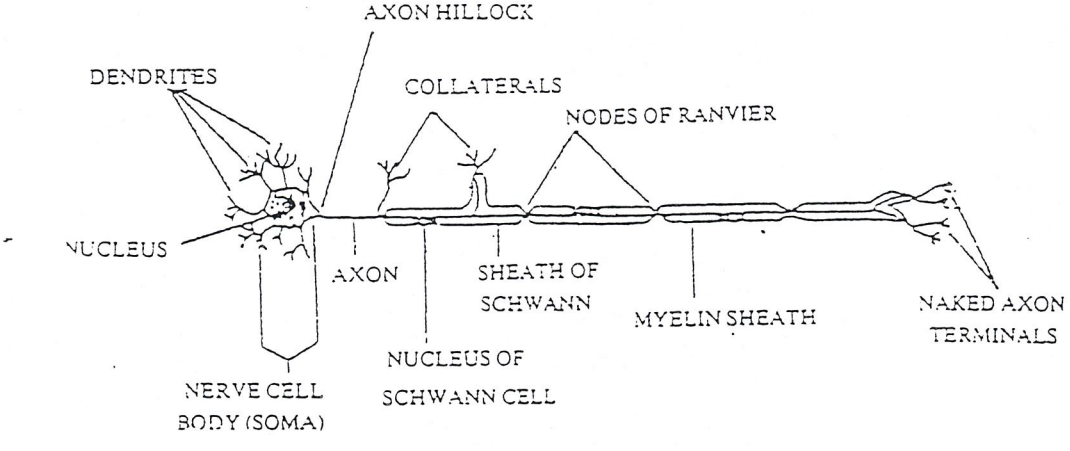
มีลักษณะค่อนข้างกลม ภายในมีนิวเคลียสลักษณะกลมขนาดใหญ่ โดยทั่วไปแล้วนิวเคลียสมักจะอยู่บริเวณกลางเซลล์ โดยสามารถมองเห็นคลีโอลัสชัดเจน

2.1.2 แขนง

สามารถจำแนกออกตามลักษณะของแขนงประสาทได้ 2 ชนิด คือ

(1) เดนไดรท์ (Dendrite) เป็นแขนงประสาทที่มีขนาดค่อนข้างสั้นที่ยื่นออกโดยรอบของตัวเซลล์ และเมื่อเดนไดรท์ยื่นออกจากตัวเซลล์ประสาทแล้วก็จะมีการแตกออกเป็นแขนงเล็กๆอีกทีหนึ่ง แขนงประสาทประเภทนี้เป็นส่วนที่ใช้ในการรับกระแสประสาทจากภายนอกเข้าสู่ตัวเซลล์ประสาท

(2) แอกซอน (Axon) เป็นแขนงประสาทที่มีขนาดค่อนข้างยาว โดยจะยื่นออกจากตัวเซลล์ประสาทเพียงแขนงเดียว โดยบริเวณที่แอกซอนยื่นออกจากตัวเซลล์จะมีลักษณะเป็นรูปปิรามิด เรียกว่า แอกซอน ฮิลลอค (Axon Hillock) และตลอดแนวความยาวของแอกซอนอาจมีแขนงย่อยแตกออกไปได้อีก เรียกว่า คอลลาเทอรัลแบรนช์ (Collateral Branch) ซึ่งส่วนปลายของแอกซอนจะไปสัมผัสกับเซลล์ประสาทอื่นๆต่อไป แอกซอนของเซลล์ประสาทส่วนใหญ่จะมีปลอกไมอิลิน (Myelin Sheath) หุ้ม ซึ่งเป็นสารประกอบเชิงซ้อนของโปรตีนและไขมันม้วนพันกันอยู่หลายชั้น โดยที่ปลอกไมอิลินจะหุ้มแอกซอนเป็นช่วงๆไป โดยเว้นบริเวณไม่หุ้มเป็นระยะๆประมาณ 1 มม. เรียกว่า โนดออฟรองเวียร์ (Node of Ranvier) และส่วนปลายของแอกซอนจะไม่มีปลอกไมอิลินหุ้ม ในระบบประสาทอัตโนมัติ ที่บริเวณปลายสุดของแอกซอนจะมีลักษณะโป่งออกเป็นกระเปาะ เรียกว่า เอนบัลล์ (End Bull) หรือเทอร์มินอล- บัททีลส์ (Terminal Buttons) แขนงประสาทประเภทนี้เป็นส่วนที่ใช้ในการส่งกระแสประสาทออกจากตัวเซลล์



รูปที่ 2-1 แสดงส่วนประกอบของเซลล์ประสาท

2.2 คุณสมบัติพื้นฐานของเซลล์ประสาท

คุณสมบัติพื้นฐานของเซลล์ประสาทในทางชีววิทยามีอยู่ 2 หลักการ คือ

2.2.1 การตอบสนองต่อสิ่งแวดล้อมทั้งภายนอกและภายในที่กระตุ้นต่อเซลล์ประสาท

ในทางชีววิทยานั้น เซลล์ประสาทของสิ่งมีชีวิตจะทำงานโดยอาศัยการตอบสนองต่อสิ่งเร้าทั้งภายในและภายนอกที่มากระตุ้นต่อตัวเซลล์ประสาท เมื่อเซลล์ประสาทถูกกระตุ้นด้วยสิ่งเร้า อาทิเช่น อุณหภูมิ, การสัมผัส ฯลฯ แล้วเซลล์ประสาทจะทำการตอบสนองต่อสิ่งเร้านั้นโดยอาศัยการเปรียบเทียบความแรงของสัญญาณกระตุ้นกับค่าเทรชโฮลด์ (Threshole) ของตัวเซลล์ ซึ่งหากสัญญาณของกระแสกระตุ้นนั้นมีค่ามากกว่าค่าเทรชโฮลด์แล้ว เซลล์ประสาทก็จะเกิดกระบวนการแอคชั่นโพเทนเชี่ยนขึ้นเพื่อตอบสนองต่อสิ่งเร้านั้นๆ

โดยคุณสมบัติพื้นฐานทางกายภาพของเซลล์ประสาท เมื่อเซลล์ประสาทถูกกระตุ้นโดยสิ่งเร้า จะทำให้ศักดาไฟฟ้าบริเวณเยื่อหุ้มเซลล์เกิดการเปลี่ยนแปลงโดยอาศัยขบวนการโซเดียมโพแทสเซียมปั๊ม (Na^{+}–K^{+} Pump) ทำให้เกิดการเปลี่ยนแปลงความเข้มข้นของโซเดียมและโพแทสเซียมภายในเซลล์ก่อให้เกิดกระบวนการแอคชั่นโพเทนเชี่ยล (Action Potcntial) ขึ้นเพื่อทำการตอบสนองต่อสิ่งเร้า โดยค่าต่ำสุดที่สามารถทำให้เกิดแอคชั่นโพเทนเชี่ยลนั้นเรียกว่า ค่าเทรชโฮลด์

2.2.2 ความสามารถในการนำกระแสประสาทจากที่หนึ่งไปยังอีกที่หนึ่ง

เมื่อเกิดกระบวนการแอคชั่นโพเทนเชี่ยลขึ้นที่ตำแหน่งใดๆของเส้นประสาทแล้ว จะทำให้สภาพทางไฟฟ้าของเยื่อหุ้มเซลล์ตำแหน่งนั้นเกิดการเปลี่ยนแปลงไปในขณะที่บริเวณรอบข้างยังคงรักษาสภาพไฟฟ้าปกติไว้ จึงทำให้เกิดการเคลื่อนที่ของกระแสไฟฟ้าขึ้น และการไหลของกระแสไฟฟ้านี้เราเรียกว่ากระแสประสาท (Nerve Impulse)

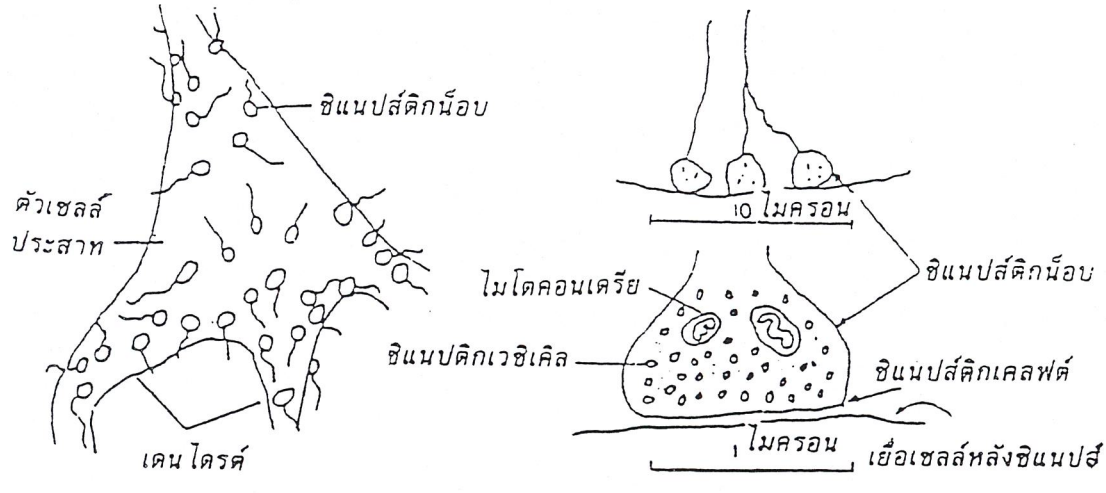
2.3 การส่งกระแสประสาทระหว่างเซลล์ประสาท

เมื่อเกิดการนำกระแสประสาทขึ้นในแอกซอนของเซลล์ประสาทหนึ่งๆแล้ว ก็จะเกิดการส่งกระแสประสาทต่อไปยังเดนไดรท์ของเซลล์ประสาทอีกเซลล์หนึ่ง โดยที่บริเวณส่วนปลายของแอกซอนและเดนไดรท์ดังกล่าวจะอยู่ใกล้ชิดกันมาก เราเรียกการส่งกระแสประสาทลักษณะนี้ว่า การไซแนปส์ (Synapse)

การไซแนปส์ หมายถึง บริเวณที่จะแสดงให้เห็นถึงหน้าที่ที่เกิดขึ้น (ไม่ใช่ลักษณะทางกายวิภาค) ระหว่างปลายสุดของเอกซอนของเซลล์ประสาทตัวหนึ่งซึ่งจะเรียกว่า เซลล์ก่อนไซแนปส์ (Presynaptic Neuron) กับเดนไดรท์ของเซลล์ประสาทอีกตัวหนึ่งซึ่งจะเรียกว่า เซลล์หลังไซแนปส์ (Postsynaptic Neuron) ที่นี่จะเป็นบริเวณที่ควบคุมให้มีการส่งกระแสประสาทต่อไปหรือยับยั้งการส่งกระแสประสาท หรือเปลี่ยนแปลงลักษณะของกระแสประสาท

2.3.1 ลักษณะโครงสร้างของบริเวณที่มีการไซแนปส์

ที่บริเวณส่วนปลายสุดของแอกซอนของเซลล์ประสาทก่อนไซแนปส์จะประกอบไปด้วย ไซแนปส์ติกน็อบ (Synaptic Knob) จำนวนมาก ซึ่งภายในไซแนปส์ติกน็อบก็ประกอบไปด้วยไมโตคอนเดรีย และถุงเล็กๆที่ห่อหุ้มด้วยเยื่อหุ้มเซลล์ประสาทอีกจำนวนหนึ่ง เรียกว่า ไซแนปส์ติกเวซิเคิลส์ (Synaptic Vesicles) ซึ่งบรรจุสารเคมีอยู่ภายใน ที่เรียกว่า สารนิวโรทรานส์มิตเตอร์ (Neurotransmitter Substance) สารนี้เมื่อหลั่งออกมาจะทำให้คุณสมบัติของเยื่อหุ้มเซลล์ของเซลล์หลังการไซแนปส์เกิดการเปลี่ยนแปลง ไซแนปส์ติกน็อบจะอยู่ห่างจากเยื่อหุ้มของเซลล์ประสาทหลังไซแนปส์ เรียกระยะห่างนี้ว่า ไซแนปส์ติกเคลฟท์ (Synaptic Cleft)



รูปที่ 2-2 แสดงลักษณะโครงสร้างของการไซแนปส์

2.4 โครงข่ายประสาทเทียม (Artificial Neural Networks)

2.4.1 แนวความคิดเริ่มต้นของโครงข่ายประสาทเทียม

ในการลอกเลียนแบบลักษณะการทำงานของเซลล์ประสาทภายในสมองมนุษย์ เราจะอธิบายด้วยหลักการพื้นฐานและพฤติกรรมของเซลล์ประสาทในทางคณิตศาสตร์ ซึ่งโดยรวมแล้วมีหลักการมาจากโครงข่ายของเซลล์ประสาทในทางชีววิทยาอย่างใกล้เคียงที่สุด โดยคุณลักษณะพื้นฐานของโครงข่ายประสาทอาจจะถูกแบ่งออกได้ตามสถาปัตยกรรมของโครงข่ายและคุณสมบัติในหน้าที่การทำงานหรือที่เรียกกันว่า นิวโรไดนามิค (Neurodynamics) [6]

สถาปัตยกรรมของโครงข่ายประสาทเทียม เราสามารถนิยามได้ถึงลักษณะทางโครงสร้างของโครงข่ายประสาทเทียม ซึ่งหมายถึง จำนวนของเซลล์ประสาทภายในโครงข่ายและลักษณะของการเชื่อมต่อกันระหว่างเซลล์ประสาท โดยปกติภายในโครงข่ายจะประกอบด้วยการเชื่อมต่อเข้าด้วยกันของเซลล์ประสาทจำนวนมาก ทั้งนี้ในแต่ละเซลล์จะเรียกว่า หน่วยประมวลผลย่อย (Processing Element) อาทิเช่น อินพุท, ความเข้มในการเชื่อมต่อผ่านไซแนปส์ (หรือที่เรียกกันโดยทั่วไปว่า ค่าน้ำหนักไซแนปส์), การกระตุ้นที่เกิดขึ้นภายในเซลล์ประสาท, ผลตอบสนอง (เอาท์พุท) และค่าไบอัส (Bias)

นิวโรไดนามิกส์หรือคุณสมบัติหน้าที่การทำงานของโครงข่ายประสาท นิยามได้ถึง คุณลักษณะเฉพาะของโครงข่ายประสาท อาทิเช่น รูปแบบการเรียนรู้ของโครงข่ายประสาท, การตอบสนองทางเอาท์พุท, การเชื่อมต่อ, การเปรียบเทียบข้อมูลใหม่กับข้อมูลเก่าที่มีอยู่และการแบ่งแยกกลุ่มของข้อมูลใหม่ เป็นต้น

โครงข่ายประสาทจะมีรูปแบบการประมวลผลข้อมูลอยู่บนพื้นฐานของการแยกส่วนแบบขนาน (Parallel Decomposition) แทนที่การประมวลผลแบบอัลกอริทึ่ม (Algorithm) ตามลำดับขั้น โดยโครงข่ายประสาทจะประมวลผลโดยอาศัยการแยกส่วนข้อมูลที่มีความซับซ้อนออก และส่งไปยังหน่วยพื้นฐานหรือหน่วยประมวลย่อย ตัวอย่างเช่น สีที่ถูกผสมกันแล้วจะไม่สามารถที่จะจำแนกออกเป็นสีพื้นฐานได้ แต่ในทางทฤษฎีของโครงข่ายประสาทเทียมแล้ว สีอาจจะถูกจำลองขึ้นมาใหม่โดยยังคงไว้ซึ่งลักษณะของสีเดิม และที่หน่วยประมวลผลย่อยก็จะทำการจดจำความสัมพันธ์ของรูปแบบของข้อมูลที่ป้อนเข้าสู่โครงข่ายต่อไป เช่น เมื่อเรามองวัตถุ สมองจะไม่เก็บข้อมูลของวัตถุที่เรามองเห็นในรูปของหน่วยความจำแบบเมตริกซ์ของแต่ละพิกเซลล์ (Pixel) แต่สมองจะทำการเก็บบันทึกคุณลักษณะพื้นฐานของวัตถุที่มองเห็น เช่น ขนาด, รูปทรง, สี เป็นต้น

2.4.2 โครงข่ายประสาทเทียม

การจำลองแบบของโครงข่ายประสาทจะเรียกว่า พาราดิกึ่มส์ (Paradigms) ซึ่งหมายถึง ตัวอย่างต้นแบบหรือสิ่งที่สร้างได้เหมือนกับสิ่งหนึ่ง ซึ่งในปัจจุบันได้เกิดการพัฒนาแบบจำลองโครงข่ายประสาทขึ้นมามากมาย โดยอาศัยความรู้ความเข้าใจถึงลักษณะการทำงานและลักษณะของเซลล์ในทางชีววิทยา โดยแต่ละโครงข่ายที่ถูกพัฒนาขึ้นต่างก็มีคุณสมบัติเฉพาะของโครงข่ายที่แตกต่างกัน ทั้งนี้ก็เนื่องมาจากการพัฒนาขึ้นเพื่อการประยุกต์ใช้งานที่แตกต่างกัน ซึ่งนั่นก็หมายความว่า โครงข่ายประสาทเทียมรูปแบบหนึ่งๆ อาจมีความเหมาะสมต่อการนำไปใช้ในการแก้ปัญหาแบบหนึ่งแต่อาจจะไม่เหมาะสมต่อการนำไปใช้งานในรูปแบบอื่นๆ

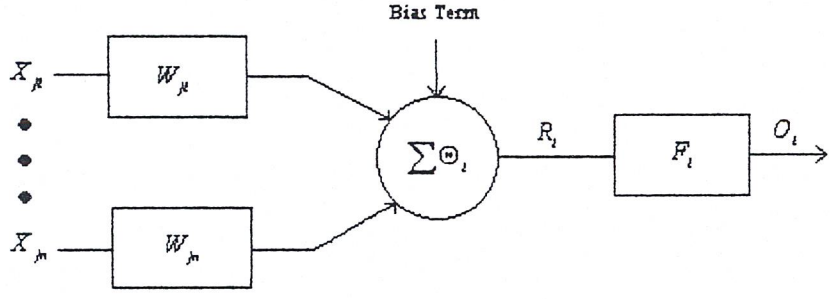
กระบวนการทางคณิตศาสตร์จะมีความเหมาะสมมากสำหรับเครื่องจักรที่มีลักษณะการทำงานแบบลำดับขั้น (Sequential Computation) อาทิเช่น เครื่องคอมพิวเตอร์ ที่มีหลักการทำงานอยู่บนพื้นฐานของคณิตศาสตร์ทางบูลีน (Boolean) โดยจะมีการทำงานในลักษณะการวนอยู่ในรอบการทำงานในการดึงคำสั่งจากหน่วยความจำ (Fetch) และปฏิบัติตามคำสั่ง (Execute) แต่การทำงานของโครงข่ายประสาทไม่ได้ทำงานในลักษณะนี้ โครงข่ายประสาทไม่รู้จักความเท่ากัน, อนุกรม, อินทิเกรตและอัลกอริทึมที่มนุษย์สร้างขึ้น

แต่อย่างไรก็ตามเครื่องคอมพิวเตอร์ คือ เครื่องมือที่ดีที่สุดที่สามารถหาได้ในปัจจุบันและยังมีประสิทธิภาพในด้านความเร็วในการประมวลผลข้อมูลที่สูงและแม่นยำ ดังนั้นเราจึงได้นำคุณสมบัติเด่นดังกล่าวของเครื่องคอมพิวเตอร์ร่วมกับข้อดีด้านความสามารถในการประมวลผลข้อมูลที่มีความซับซ้อนของโครงข่ายประสาทเทียม เป็นแนวคิดของการแก้ปัญหาในกรณีที่การแก้ปัญหาด้วยรูปแบบของอัลกอริทึมแบบลำดับขั้นไม่ได้ผลหรือมีประสิทธิภาพต่ำ ขบวนการทางซอฟแวร์จะใช้ในการจำลองคุณสมบัติของโครงข่ายประสาทเทียมและแบบปัญหาที่ต้องการ ดังนั้นโดยแท้จริงแล้ว การทำงานของระบบโดยรวมก็ยังคงเป็นการทำงานแบบลำดับขั้นทางฮาร์ดแวร์ (Hardware) ที่แสดงถึงการทำงานของโครงข่ายประสาทเทียมนั่นเอง

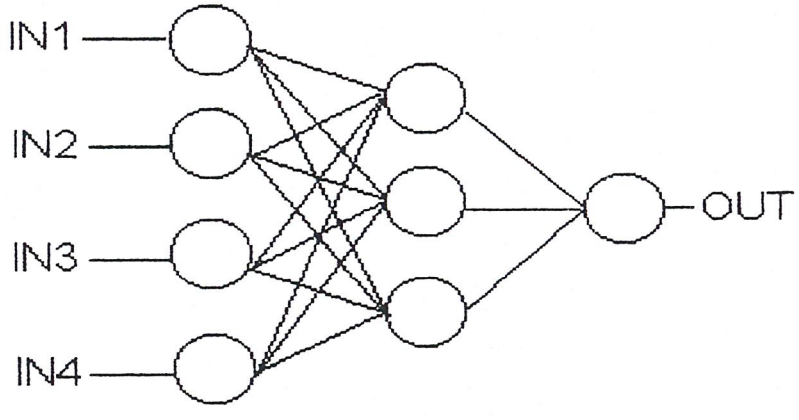
ในการสร้างแบบจำลองพื้นฐานของเซลล์ประสาท เราจะต้องพิจารณาถึงหน่วยพื้นฐานของโครงข่ายประสาทเทียมที่ถูกสร้างขึ้น โดยที่เซลล์ประสาทแต่ละตัวของโครงข่ายประสาทเทียมจะเรียกว่าเซลล์ประสาทเทียม หรือนิวรอล (Neural) หรือโหนด (Node) [1]

นิวรอลพื้นฐานจะมีกลุ่มของอินพุท n จำนวน โดยจะแทนด้วยสัญลักษณ์ X_i เพื่อแสดงถึง นิวรอลตัวที่ i โดยแต่ละอินพุทจะถูกถ่วงด้วยค่าน้ำหนัก เช่น ค่าความเข้มของการเชื่อมโยง (Connection Strength) หรือแฟคเตอร์ค่าน้ำหนัก (W_i) รวมทั้งเทอมการไบแอส (W_0) และค่าเทรชโฮลด์ ซึ่งหากผลรวมของค่าสัญญาณที่ได้จากหน่วยประมวลผลย่อยมีค่ามากกว่าหรือเท่ากับค่าเทรชโฮลด์แล้ว เซลล์

ประสาทก็จะทำการสร้างผลตอบสนองเอาท์พุทออกมา และหากผลตอบสนองที่ได้นี้ผ่านฟังก์ชั่นกระตุ้นแล้วเซลล์ก็จะตอบสนองสัญญาณเอาท์พุทจริงออกมา โดยค่าสัญญาณตอบสนองเอาท์พุทของโหนดในชั้นแรกก็จะกลายเป็นค่าอินพุทของโหนดในชั้นถัดไป ดังนั้นโครงข่ายประสาทเทียมขนาดใหญ่เราสามารถจำลองได้ด้วยการเชื่อมต่อของโหนดหลายๆ โหนด



รูปที่ 2-3 แสดงแบบจำลองพื้นฐานของเซลล์ประสาท



รูปที่ 2-4 แสดงแบบจำลองโครงสร้างของโครงข่ายประสาทเทียม

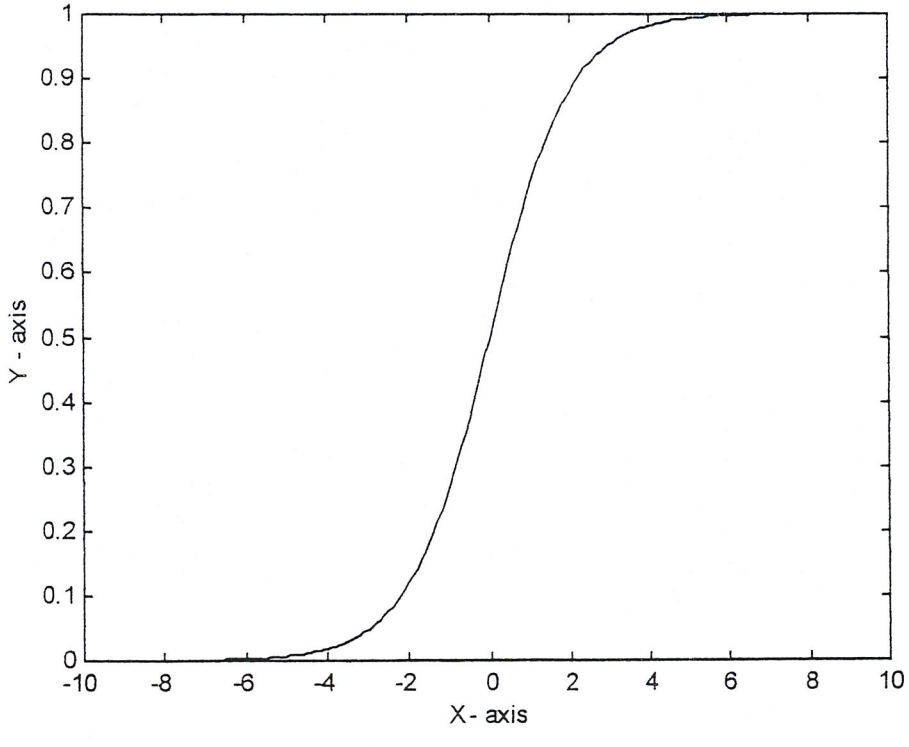
$$O_i = F_i\left(\sum_{i=1}^{N} W_{ij} X_{ij}\right) \qquad (2.1)$$

โดยที่ f_i คือ ฟังก์ชันกระตุ้นแบบไม่เชิงเส้น

w_{ij} คือ ค่าน้ำหนักไซแนปส์

O_i คือ ค่าเอาท์พุท (Output) ที่ออกจากโหนด (Node)ในแต่ละชั้นของโครงข่าย

x_i คือ ค่าอินพุท (Input) ที่เข้ามาในโหนด (Node)ของชั้นอินพุทของโครงข่าย



รูปที่ 2-5 แสดงลักษณะของฟังก์ชั่นกระตุ้นแบบซิกมอยด์

จากสมการที่ 2.1 ฟังก์ชันกระตุ้นที่ใช้จะนิยมใช้ฟังก์ชั่นกระตุ้นแบบไม่เป็นเชิงเส้น ทั้งนี้เพื่อให้โครงข่ายสามารถจำกัดขอบเขตผลตอบสนองของเซลล์ประสาทได้ นั่นคือ ผลตอบสนองจริงของเซลล์ประสาทจะมีเงื่อนไขหรือจะถูกลดทอนเท่ากับผลลัพธ์ของขนาดการกระตุ้นที่มากหรือน้อย และสามารถควบคุมขอบเขตของฟังก์กระตุ้นได้

โดยปกติในทางปฏิบัติ ฟังก์ชั่นกระตุ้นที่นิยมใช้คือ ฟังก์ชั่นซิกมอยด์ (Sigmoid) เนื่องจากเป็นฟังก์ชั่นที่มีคุณลักษณะโมโนโตมิค (Monotomic) หมายถึง เป็นฟังก์ชั่นที่ให้ค่าที่มีค่าเดียว ในทุกค่าของ X อีกทั้งยังคงเป็นฟังก์ชั่นต่อเนื่อง และสามารถทำการดิฟเฟอร์เรนเชียลได้ง่าย เพราะค่าดิฟเฟอร์เรนเชียลของฟังก์ชั่นซิกมอยด์จะอยู่ในรูปของสมการของฟังก์ชั่นปกติ คือ $f'(x) = f(x)[1 - f(x)]$ ซึ่งจะมีความสะดวกต่อการคำนวณในอัลกอริที่มการเรียนรู้แบบแพร่ย้อนกลับ (Back-Propagation Learning Algorithm) เป็นอย่างมาก แต่ในฟังก์ชั่นแบบฮาร์ดลิมิตเตอร์ จะไม่เป็นโมโนโตมิค เพราะเป็นฟังก์ชั่นที่ไม่มีความต่อเนื่องที่จุดกำเนิด (Origin) ดังนั้นจึงทำให้การหาค่าดิฟเฟอร์เรนเชียลของฟังก์ชั่นทำได้ยากขึ้น แต่อย่างไรก็ตามเซลล์ประสาทยังสามารถให้ผลตอบสนองที่อยู่ในขอบเขตบนและขอบเขตล่างเท่านั้น และค่าดิฟเฟอร์เรนเชียลของฟังก์ชั่นในช่วงขอบเขตล่างจะคงที่

2.5 การเรียนรู้ของโครงข่ายประสาทเทียม

การเรียนรู้ของโครงข่ายประสาทเทียม เป็นเรื่องที่สำคัญอย่างมากในงานวิจัยทั้งทางด้านโครงข่ายทางชีววิทยาและโครงข่ายประสาทเทียม การเรียนรู้ของโครงข่ายประมาทจะไม่มีข้อสรุปการประมวลผลที่ตายตัว นั่นคือ การประมวลผลที่แตกต่างกันจะมีความเหมาะสมในประเภทของแบบจำลองโครงข่ายที่แตกต่างกัน ดังนั้นการประมวลผลการเรียนรู้จึงมีประสิทธิภาพที่แตกต่างกัน แนวคิดของการเรียนรู้ของโครงข่ายมักจะมีพื้นฐานมาจากสังเกตจากพฤติกรรมในห้องทดลอง และนำไปสู่การใช้งานจริงโดยการเลือกชนิดของการเรียนรู้และโครงข่ายที่มีประสิทธิภาพและมีความเหมาะสมมากที่สุด

ขบวนการเรียนรู้ของโครงข่าย คือ ช่วงที่โครงข่ายจะปรับตัวเอง (การปรับค่าน้ำหนักไซแนปส์ ; w_{ij}) จากการกระตุ้นของอินพุทเพื่อสร้างผลตอบสนองของโครงข่ายให้ได้ตามที่ต้องการและท้ายที่สุดโครงข่ายประสาทก็จะสามารถสร้างผลตอบสนองที่ต้องการได้ในทุกๆ รูปแบบของอินพุทนั้นๆ โครงข่ายก็จะสิ้นสุดการเรียนรู้อย่างสมบูรณ์ โดยความรู้ของโครงข่ายต่อรูปแบบอินพุททั้งหมดจะหมายถึง ค่าน้ำหนักของการไซแนปส์ทุกๆค่าในโครงข่ายนั้นเอง แบบจำลองโครงข่ายประสาทเทียมที่พัฒนาแล้วจะประกอบด้วยเซลล์ประสาทที่เชื่อมต่อถึงกันในรูปแบบต่างๆ ซึ่งจะก่อให้เกิดขบวนการเรียนรู้ที่แตกต่างกันดังที่ได้กล่าวมาแล้วในข้างต้น

2.5.1 การเรียนรู้โดยมีผู้สอน (Supervised Learning)

ในระหว่างช่วงของการฝึกหัดโครงข่าย กลุ่มของรูปแบบอินพุทที่ใช้ในการฝึกหัดโครงข่ายจะถูกป้อนเข้าเป็นสัญญาณกระตุ้นอินพุทและโครงข่ายจะสร้างผลตอบสนองขึ้นที่เอาท์พุท ผลตอบสนองนี้จะถูกเปรียบเทียบกับค่าเอาท์พุทที่ต้องการในแต่ละรูปแบบสัญญาณอินพุทใช้กระตุ้น หรือที่เรียกว่าผลตอบสนองเป้าหมาย (Target Response) โดยผลตอบสนองของโครงข่ายจะมีความแตกต่างจากผลตอบสนองเป้าหมายที่ต้องการ สัญญาณที่โครงข่ายสร้างขึ้นจะมีความผิดพลาดเกิดขึ้น โดยค่าความผิดพลาดที่เกิดขึ้นจะถูกนำมาใช้ในการคำนวณปรับปรุงค่าน้ำหนักไซแนปส์ทุกๆค่าของโครงข่ายในแต่ละรอบการเรียนรู้จนกว่าโครงข่ายจะสามารถสร้างผลตอบสนองได้เหมือนกับเอาท์พุทเป้าหมาย นั่นก็คือ กระบวนการเรียนรู้จะสิ้นสุดได้ก็ต่อเมื่อค่าความผิดพลาดที่เกิดขึ้นมีเท่ากับศูนย์หรือมีค่าน้อยที่สุดเท่าที่จะเป็นไปได้

ในกระบวนการปรับค่าความผิดพาดให้น้อยที่สุดนั้น เริ่มแรกต้องมีวงจรในการสร้างค่าความผิดพลาดขึ้น เรียกว่า ผู้สอน (Teacher) หรือผู้ควบคุม (Supervised) ซึ่งก็คือ ค่าผลลัพธ์เป้าหมายที่ต้องการในแต่ละรูปแบบของอินพุท ดังนั้นเราจึงเรียกการเรียนรู้ของโครงข่ายในลักษณะนี้ว่า การเรียนรู้ลักษณะมีผู้สอน (Supervised Learning)

เมื่อพิจารณาถึงโครงข่ายประสาทเทียม การคำนวณในแต่ละรอบจะทำให้ค่าความผิดพลาดที่เกิดขึ้นในแต่ละรอบมีค่าลดลงทั้งนี้ก็ขึ้นอยู่กับรูปแบบของอัลกอริทึมที่ใช้ และพารามิเตอร์ของโครงข่าย โดยที่ค่าพารามิเตอร์ของโครงข่ายที่น่าสนใจ คือ เวลาที่ใช้ในการปรับค่าน้ำหนัก, จำนวนรอบที่โครงข่ายที่ใช้ในการเรียนรู้, รูปแบบอินพุทและค่าความผิดพาดของโครงข่ายที่ยอมรับได้ต่ำสุด

2.5.2 การเรียนรู้ในลักษณะไม่มีผู้สอน (Unsupervised Learning)

การเรียนรู้ลักษณะนี้จะแตกตางกับการเรียนรู้แบบมีผู้สอนตรงที่ โครงข่ายจะไม่ต้องการผู้สอนในขณะการฝึกหัด นั่นคือ จะไม่มีค่าเอาท์พุทเป้าหมายในช่วงการฝึกหัดหรือการเรียนรู้ของโครงข่าย เมื่อโครงข่ายได้รับการกระตุ้นจากรูปแบบของอินพุทที่แตกต่างกันออกไป โครงข่ายจะพยายามจัดรูปแบบของอินพุทไปเป็นหมวดหมู่อย่างไม่มีเหตุผล หรือกล่าวอีกนัยหนึ่งก็คือ โครงข่ายจะทำการจัดหาผลตอบสนองทางเอาท์พุทที่แสดงกลุ่มสัญญาณเอาท์พุทที่เข้ากัน

ในลักษณะการเรียนรู้ชนิดนี้ โครงข่ายจะไม่ต้องการผู้สอนแต่ต้องการเพียงสิ่งชี้แนะ (Guide line) ในการพิจารณาเริ่มแรกว่า โครงข่ายจะสร้างกลุ่มข้อมูลได้อย่างไรตามคุณลักษณะต่างๆของอินพุท ดังนั้นถ้าไม่มีตัวชี้แนะโครงข่ายจะไม่ทราบว่าจะใช้คุณลักษณะชนิดใดในการจัดกลุ่มข้อมูล

2.5.3 การเรียนรู้ในลักษณะถูกกระตุ้น (Reinforced Learning)

การเรียนรู้ในลักษณะนี้ โครงข่ายจะต้องมีจำนวนนิวรอนหนึ่งตัวหรือมากกว่าที่ชั้นเอาท์พุท และลักษณะของผู้สอนจะไม่เหมือนกับการเรียนรู้ในลักษณะที่มีผู้สอน แต่จะมีการตรวจสอบค่าความผิดพาดที่เกิดขึ้นโดยผู้สอน คือ จะทำการตรวจสอบว่าเอาท์พุทจริงเหมือนกับเอาท์พุทเป้าหมายหรือไม่ ในช่วงของการเรียนรู้ สัญญาณอินพุทจะถูกส่งเข้าไปยังโครงข่ายและโครงข่ายจะสร้างผลตอบสนองที่เอาท์พุท โดยที่ผู้สอนจะไม่แสดงเอาท์พุทเป้าหมายไปยังโครงข่ายแต่จะแสดงว่าผ่านหรือผิดพาดเท่านั้น

หากผู้สอนแสดงค่า "ผิดพลาด" โครงข่ายก็จะทำการปรับปรุงค่าพารามิเตอร์ใหม่ และจะพยายามต่อไปจนกระทั่งได้ผลตอบสนองเอาท์พุทที่ถูกต้อง ในช่วงระหว่างการฝึกหัดของโครงข่ายจะไม่มีสัญญาณตอบสนองต่อการปรับตัวของโครงข่ายว่ามีถูกทิศทางหรือไม่

2.5.4 การเรียนรู้ในลักษณะที่มีการแข่งขัน (Competitive learning)

ในการเรียนรู้ในลักษณะนี้จะไม่เหมือนการเรียนรู้ในลักษณะอื่นๆที่ได้กล่าวมาในข้างต้น เพราะการเรียนรู้ลักษณะนี้โครงข่ายจะต้องมีเซลล์ประสาทหลายตัวที่ชั้นเอาท์พุท ทำให้คุณลักษณะการทำงานและสถาปัตยกรรมของโครงข่ายแตกต่างกัน เมื่อสัญญาณกระตุ้นที่อินพุทถูกป้อนเข้าสู่โครงข่าย เอาท์พุทแต่ละตัวของเซลล์ประสาทจะแข่งขันกับเอาท์พุทตัวอื่นๆในการสร้างสัญญาณเอาท์พุทที่ให้ค่าใกล้เคียงกับค่าเป้าหมายมากที่สุด และเอาท์พุทตัวนี้ก็จะกลายเป็นตัวที่มีอิทธิพลเหนือเอาท์พุทตัวอื่น ทำให้โครงข่ายหยุดการสร้างสัญญาณเอาท์พุทสำหรับการกระตุ้นนั้น และสำหรับในการกระตุ้นในรูปแบบต่อไปเอาท์พุทของนิวรอลตัวอื่นจะกลายเป็นเอาท์พุทที่มีอิทธิพลเหนือกว่า โดยหลักการเช่นนี้จะทำให้เอาท์พุทของเซลล์ประสาทแต่ละตัวถูกฝึกหัดให้ตอบสนองสำหรับการกระตุ้นที่แตกต่างกัน

2.5.5 กฎของเดลต้า (Delta Rule)

กฎของเดลต้าจะตั้งอยู่บนพื้นฐานของแนวคิดในการปรับค่าน้ำหนักของการไซแนปส์อย่างต่อเนื่อง เช่น ความแตกต่างของของค่าความผิดพลาด (Delta) ระหว่างค่าเอาท์พุทที่ต้องการกับค่าเอาท์พุทจริงที่ได้จากผลตอบสนองจากหน่วยประมวลผลย่อยของโครงข่าย โดยที่ค่าความผิดพลาดเหล่านี้จะถูกทำให้ลดลงจากกฎการเรียนรู้ของ Widrow –Hoff ในแบบตัวอย่างของ ADALINE โดยใช้หลักการทำให้ค่ากำลังสองเฉลี่ยน้อยที่สุด

2.6 คุณลักษณะของโครงข่ายประสาทเทียม

โครงข่ายประสาทเทียม คือ การเชื่อมต่อกันของหน่วยประมวลผลย่อย (Processing Element) ที่จำลองมาจากการทำงานของเซลล์ประสาทตัวหนึ่ง โครงข่ายที่ได้จะมีความสามารถในการปรับตัวเองในช่วงของการเรียนรู้ได้ ในทางคณิตศาสตร์ โครงข่ายประสาทสามารถจำลองได้ด้วยกลุ่มของสมการดิฟเฟอเรนเชียลของผลตอบสนองของหน่วยประมวลผลย่อยที่เชื่อมต่อเข้าด้วยกัน ซึ่งคล้ายกับทฤษฎีการควบคุมแบบป้อนกลับ โดยเสถียรภาพที่เกิดขึ้นนั้นจะขึ้นอยู่กับค่าพารามิเตอร์ของระบบ ซึ่งอาจจะเกิดการแกว่ง (Oscillation) หรือเหตุการณ์ที่ระบบขาดเสถียรภาพไปอย่างสิ้นเชิง ผลกระทบที่เกิดกับโครงข่ายในลักษณะนี้ คือ การปรับค่าความผิดพลาดกำลังสองเฉลี่ย (Sum Square Error) ของระบบ.

2.6.1 พารามิเตอร์ของโครงข่ายที่สำคัญ

ประสิทธิภาพของโครงข่ายจะแสดงถึงจำนวนรูปแบบเอาท์พุทที่โครงข่ายสามารถให้ผลตอบสนองที่ถูกต้อง เมื่อรูปแบบอินพุทถูกป้อนเข้าสู่โครงข่ายซึ่งจะมีความสมบูรณ์ หรือสมบูรณ์บางส่วน หรือมีสัญญาณรบกวน (Noisy) ในรูปแบบของอินพุท สิ่งที่ควรจะต้องพิจารณาในการออกแบบโครงข่ายประสาทเทียมมีดังนี้

1.) สถาปัตยกรรมของโครงข่าย
2.) จำนวนของชั้นในโครงข่าย
3.) จำนวนนิวรอลหรือจำนวนโหนดต่อชั้น
4.) อัลกอริทึมในการเรียนรู้ในการปรับตัวของโครงข่าย
5.) จำนวนรอบต่อรูปแบบของการฝึกหัด
6.) เวลาที่ใช้ในการคำนวณต่อรอบ
7.) ความเร็วในการตอบสนองของโครงข่าย
8.) ประสิทธิภาพของโครงข่าย
9.) ความจุของโครงข่าย (Network capacity) หรือ สัญญาณรูปแบบมากที่สุด ซึ่งโครงข่ายสามารถที่จะเรียนรู้ได้
10.) ระดับความสามารถในการปรับตัวเองของโครงข่าย
11.) เทอมของการไบอัส
12.) เทอมช่วงค่าเทรชโฮลด์
13.) ขอบเขตของค่าน้ำหนัก

14.) การเลือกฟังก์ชันกระตุ้น

15.) ระดับการทนทานต่อสิ่งรบกวนของโครงข่าย คือ ระดับการเสียหายของสัญญาณกระตุ้นอื่นพุทที่ทำให้โครงข่ายสร้างผลตอบสนองเอาท์พุทไม่เหมือนกับเอาท์พุทเป้าหมาย

16.) สภาวะอยู่ตัว (Steady-State) ของโครงข่ายหรือค่าน้ำหนักไซแนปส์สุดท้ายในช่วงการเรียนรู้

2.7 รูปแบบและลักษณะของโครงข่ายประสาทเทียม(Artificial Neural Network Topology)

โครงข่ายประสาทเทียมจะประกอบด้วยเซลล์ประสาทจำนวนมากที่เชื่อมโยงเข้าด้วยกันในเส้นทางที่แน่นอน โดยปกติการกระตุ้นอินพุทจะถูกป้อนเข้าสู่ชั้นอินพุทและผลตอบสนองแต่ละหน่วยประมวลผลย่อยจะถูกส่งไปเป็นอินพุทในชั้นถัดไปของโครงข่าย

ชั้นของโหนดที่อยู่ระหว่างอินพุทและเอาท์พุท เรียกว่า ชั้นซ่อน (Hidden Layer) ซึ่งอาจมีจำนวน 1 ชั้นหรือมากกว่าก็ได้ การมีจำนวนของชั้นซ่อนมากเท่าใดก็จะทำให้โครงข่ายประสาทเทียมมีขีดความสามารถในการแก้ปัญหาที่มีความซับซ้อนที่ดียิ่งขึ้น

ในส่วนของรูปแบบและลักษณะของโครงข่ายประสาทก็จะมีความแตกต่างกันออกไปตามลักษณะของการเชื่อมต่อของแต่ละโหนดในโครงข่ายประสาท อาทิเช่น

- Multilayer feedforward
- Multilayer cooperative/competitive
- Bilayer feedforward/backward
- Monolayer heterofeedback

2.7.1 แบบจำลองของ Mc Culloch-Pitts

เป็นแบบจำลองทางคณิตศาสตร์ของนิวรอนในทางชีววิทยาในอุดมคติเพียงตัวเดียวโดยไม่มีการเรียนรู้และการปรับค่าน้ำหนัก ทั้งนี้ก็เนื่องจากว่าแบบจำลองนี้ไม่มีลักษณะทางกลไกในการเปรียบเทียบผลตอบสนองเอาท์พุทที่ต้องการและไม่มีการป้อนกลับของสัญญาณ (Open-loop) โดยมีจุดประสงค์เดียวคือ จำลองเซลล์ประสาทเพียงเซลล์เดียว

ในแบบจำลองนี้ อินพุทแต่ละตัวจะได้รับการกระตุ้น X_i ซึ่งจะถูกถ่วงด้วยค่าน้ำหนักไซแนปส์ (W_{ij}) แสดงถึงความเข้มของการเชื่อมโยงในระหว่างการไซแนปส์, ค่าอินพุทที่ถูกถ่วงน้ำหนักแล้วจะถูกนำมา

รวมกัน หากมีค่าถึงระดับเทรสโฮลด์ สัญญาณเอาท์พุทก็จะถูกสร้างขึ้นและส่งผ่านทรานส์เฟอร์ฟังก์ชั่นที่ไม่เป็นเชิงเส้นต่อไป

$$O_i = f(\sum_{i=1}^{N}(x_{ij}w_{ij}) - \theta_i) \qquad (2.2)$$

โดยที่ x_{ij} คือ สัญญาณอินพุทหรือสัญญาณกระตุ้นที่อินพุท j บนนิวรอลตามลำดับ i

f คือ ฟังก์ชันกระตุ้นที่ไม่เป็นเชิงเส้น

O_i คือ ผลตอบสนองเอาท์พุทนิวรอลลำดับ i

N คือ จำนวนของสัญญาณกระตุ้นอินพุททั้งหมด

2.7.2 รูปแบบเปอร์เซปตรอน (Perceptron)

แบบจำลองเปอร์เซปตรอน คือ แบบจำลองโครงข่ายประสาทที่ต้องการการเรียนรู้แบบมีผู้สอน ทำให้โครงข่ายมีความสามารถในการจัดแบ่งกลุ่มรูปแบบของข้อมูลได้อย่างเหมาะสม ซึ่งกลไกในการเรียนรู้ของโครงข่ายแบบนี้จะขึ้นอยู่กับความแตกต่างของความผิดพลาดที่เกิดขึ้นระหว่างเป้าหมายและข้อมูลเอาท์พุทจริง

สำหรับทรานส์เฟอร์ฟังก์ชั่นของเปอร์เซปตรอน จะใช้ฟังก์ชันกระตุ้นที่มีลักษณะเป็น ฮาร์ด ลิมิตเตอร์ (Hard Limiter) คือ

$$f_{HL}(y) = 1 \quad \text{สำหรับ} \quad y > 0$$
$$f_{HL}(y) = 0 \quad \text{สำหรับ} \quad y \le 0$$

โดยส่วนมาก ค่าไบอัสจะเท่ากับ 1 และค่าไบอัสของน้ำหนักไซแนปส์ (w_0) จะรวมอยู่ในส่วนของเวกเตอร์อินพุท

$$X = [x_1 x_2 x_3 \ldots x_n 1]$$

ในระหว่างการฝึกหัดโครงข่าย เอาท์พุทที่ได้จากผลตอบสนองของโครงข่าย (O) จะถูกเปรียบเทียบกับค่าเอาท์พุทเป้าหมาย (T) ในแต่ละรูปแบบของอินพุท และสัญญาณความผิดพลาด (E) จะถูกนำมาใช้ในกระบวนการการปรับค่าน้ำหนักของไซแนปส์ต่อไป

$$E = T - O \tag{2.3}$$

O คือ เอาท์พุทตอบสนองของเปอร์เซปตรอน

T คือ เอาท์พุทเป้าหมายในแต่ละรูปแบบอินพุท

E คือ ค่าความผิดพลาดของเปอร์เซปตรอน

การปรับน้ำหนักไซแนปส์จากทฤษฎี เกรเดียน เดสเซนต์ (ภาคผนวก ข.)

$$\Delta w = \eta[T - f(wx)]x \tag{2.4}$$

เมื่อมีการปรับปรุงค่าน้ำหนักไซแนปส์ใหม่อีกครั้ง มีสมการดังนี้

$$w_{(k+1)} = w_k + \eta[T - f(w_k x)]x \tag{2.5}$$

x คือ เวกเตอร์อินพุทที่รวมเทอมการไบอัส

w คือ ค่าเวคเตอร์ของน้ำหนักไซแนปส์ที่รวมเทอมค่า เทรสท์โฮลด์ของนิวรอล

η คือ ค่าอัตราการเรียนรู้ (Learning Rate)

2.7.3 รูปแบบเปอร์เซปตรอนที่ต่อหลายชั้น (Multilayer Proceptron)

โครงข่ายแบบนี้จะประกอบด้วยเปอร์เซปตรอนจำนวนมากในโครงสร้างที่เป็นลำดับชั้น โดยมีรูปแบบการคำนวณไปยังชั้นถัดไป (Feed-Foreward) เป็นกระบวนการที่สามารถใช้การเรียนรู้ที่แตกต่างกันได้ และโดยทั่วไปจะใช้ฟังก์ชั่นซิกมอยด์ (Sigmoid Function) เป็นฟังก์ชันกระตุ้น (Activate Function)

สมการของฟังก์ชันซิกมอยด์ คือ

$$f_s(R) = \frac{1}{(1+e^{-kR})} \tag{2.6}$$

k คือ ค่าเกนของฟังก์ชัน

อนุพันธ์ของฟังก์ชันซิกมอยด์ คือ

$$f'(x) = f(x)[1-f(x)] \tag{2.7}$$

2.7.4 กระบวนการเรียนรู้แบบเดลต้า

กระบวนการเรียนรู้แบบเดลต้า จะใช้พื้นฐานการหาค่าความผิดพลาดกำลังสองเฉลี่ยที่น้อยที่สุด (Least Square Error Minimization Method) โดยมีจุดประสงค์เพื่อแสดงถึงความแตกต่างระหว่างเอาท์-พุทจริงที่ได้จากการคำนวณและเอาท์พุทตอบสนองที่ต้องการในรูปของอินพุท และค่าน้ำหนักไซแนปส์ จากสมการดังต่อไปนี้

$$E = \frac{1}{2}(T_i - O_i)^2 \tag{2.8}$$

$$E = \frac{1}{2}(T_i - f(w_i x_i))^2 \tag{2.9}$$

2.8 โครงข่ายประสาทเทียมแบบฟีดฟอร์เวิร์ดและการเรียนรู้แบบแพร่ย้อนกลับ

2.8.1 โครงข่ายประสาทเทียมแบบฟีดฟอร์เวิร์ด (Feed-Foreward Neural Networks)

โครงข่ายประสาทเทียมแบบฟีดฟอร์เวิร์ดจะประกอบด้วยชั้นของโหนดหรือชั้นของเซลล์ประสาทที่มีการเชื่อมต่อของเซลล์ประสาทในระหว่างชั้นเซลล์ของโครงข่าย โดยในแต่ละโครงข่ายจะประกอบไปด้วยชั้นต่างๆ ดังต่อไปนี้ คือ [1]

- ชั้นอินพุท (Input Layer)
- ชั้นซ่อน (Hidden Layer)
- ชั้นเอาท์พุท (Output Layer)

ในการทำงานของโครงข่ายนั้น โครงข่ายในแต่ละชั้นจะทำการประมวลผลข้อมูลหรืออินพุทที่ได้รับและทำการสร้างผลตอบสนองหรือเอาท์พุทเพื่อส่งต่อไปเป็นข้อมูลอินพุทของเซลล์ประสาทในชั้นถัดไปโดยผ่านค่าน้ำหนักของการไซแนปส์ของการเชื่อมต่อระหว่างเซลล์นั้น (ในชั้นอินพุทจะไม่มีการคำนวณผลตอบสนองของนิวรอน) การส่งสัญญาณจะมีลักษณะเช่นนี้ไปเรื่อยๆจนกระทั่งถึงการคำนวณผลชั้นสุดท้ายหรือที่ชั้นเอาท์พุท ผลตอบสนองของเอาท์พุทของนิวรอลในชั้นนี้จะแสดงถึงค่าเอาท์พุทที่ได้ของโครงข่ายประสาทเทียม

2.8.2 การเรียนรู้แบบแพร่ย้อนกลับ (Back – Propagation Learning Algorithm)

การเรียนรู้แบบแพร่ย้อนกลับ คือ กระบวนการทางคณิตศาสตร์ที่ใช้ในการปรับค่าน้ำหนักของการไซแนปส์ระหว่างเซลล์ประสาทภายในโครงข่าย โดยใช้ค่าความผิดพลาดที่เกิดจากการตอบสนองของโครงข่ายต่อรูปแบบข้อมูลอินพุทและเอาท์พุทที่ต้องการมาเป็นค่าเริ่มต้นโดยมีเป้าหมาย คือ การทำให้ค่าความผิดพลาดที่เกิดขึ้นจากโครงข่ายนั้นมีค่าลดลง

ในกระบวนการเรียนรู้แบบแพร่ย้อนกลับนี้ เราสามารถทำการวิเคราะห์ในรูปแบบของกระบวนการทางคณิตศาสตร์ได้ดังนี้ คือ

สมการที่ใช้ในการปรับค่าน้ำหนัก คือ

$$w_{ij}^{l}(t+1) = w_{ij}^{l}(t) + \eta\delta_{i}^{l}o_{j}^{l-1} + \alpha\left(w_{ij}^{l}(t) - w_{ij}^{l}(t-1)\right) \qquad (2.10)$$

สำหรับค่าน้ำหนักที่ชั้นเอาท์พุท (ชั้น L)

$$\delta_i^L = (T_i - O_i^L)O_i^L(1-O_i^L) \qquad (2.11)$$

และสำหรับค่าน้ำหนักในชั้นซ่อน

$$\delta_i^l = (\sum_{r=1}^{N_l} \delta_r^{l+1} w_{ri}^{l+1})O_i^l(1-O_i^l) \qquad (2.12)$$

กำหนดให้โครงข่ายมีจำนวนชั้น L ชั้น และมี N_l นิวรอนในชั้น l

$W_{l,j,i}$ คือ ค่าน้ำหนักไซแนปส์ระหว่างนิวรอล i ในชั้น l-1 และนิวรอล j ในชั้น l

$O_{l,j}(X_p)$ คือ ค่าเอาท์พุทจริง (สำหรับรูปแบบอินพุท X_p ของนิวรอลลำดับ j ในชั้น l หลังจากที่ผ่านฟังก์ชันกระตุ้นแล้ว)

$T_{L,j}(X_p)$ คือ ค่าเอาท์พุทที่ต้องการ (เอาท์พุทตอบสนอง) สำหรับรูปแบบ X_p ของนิวรอลลำดับ j ในชั้น L

η คือ ค่าอัตราการเรียนรู้ของโครงข่ายโดยที่ $0 \leq \eta \leq 1$

α คือ ค่าสัมประสิทธิ์โมเมนตัมโดยที่ $0 \leq \alpha \leq 1$

2.9 การเตรียมข้อมูลเริ่มต้นก่อนการประยุกต์ใช้งาน

ก่อนที่จะเริ่มขบวนการเรียนรู้แบบแพร่ค่าย้อนกลับจะต้องพิจารณาถึงลักษณะดังต่อไปนี้

1) การตัดสินใจเลือกฟังก์ชันการใช้งานโครงข่าย เช่น การจดจำรูปแบบ , การทำนาย เป็นต้น
2) มีชุดรูปแบบของอินพุทและเอาท์พุทที่ใช้ในการฝึกหัดโครงข่ายอย่างสมบูรณ์
3) เลือกจำนวนของชั้นในโครงขายและจำนวนของโหนดที่ใช้ในแต่ละชั้น
4) เลือกฟังก์ชันกระตุ้นที่ไม่เป็นเชิงเส้น (โดยปกติ คือ ฟังก์ชันซิกมอยด์) และค่าเกนของฟังก์ชัน
5) เลือกอัตราการเรียนรู้และค่าสัมประสิทธิ์โมเมนตัมของโครงข่าย
6) เลือกหลักการหยุดอัลกอริทึมการเรียนรู้ เพราะเมื่อโครงข่ายเข้าสู่ขบวนการหยุดการเรียนรู้แล้วจะต้องคำนวณอยู่ในรอบการทำงานของอัลกอริทึมนั้น โดยเลือกจาก

6.1) ที่ค่าความผิดพลาดที่ยอมรับได้ที่โครงขายจะต้องสร้างผลตอบสนองที่มีคาวามแตกต่างจากเอาท์พุทที่ต้องการโดยรวมไม่เกินค่านี้

6.2) จำนวนรอบสูงสุดของการคำนวณ เนื่องจากในบางครั้งการฝึกหัดจะต้องใช้จำนวนรอบในการปรับค่าน้ำหนักไซแนปส์มากหรือค่าความผิดพลาดลดลงน้อยมากหรือเกิดการแกว่ง (Oscillate) เป็นต้น

6.3) ทำให้อัลกอริทึมสิ้นสุดจากโปรแกรมเอง โดยอาศัยการรับ " เมสเสจ (Messages) " จากวินโดวส์หลัก เช่น เมสเสจที่เกิดจากการกดปุ่ม (Button) แล้วสร้างเม็ธธอร์ด (Methods) ให้ขบวนการเรียนรู้สิ้นสุดลง โดยการเขียนโปรแกรมแทรกในช่วงฝึกหัดโครงข่าย

6.4) ใช้ทั้ง 6.1 – 6.3

โดยสรุปขบวนการเรียนรู้จะมีขั้นตอนการทำงานดังนี้

1.) ตั้งค่าเริ่มต้นของค่าน้ำหนักไซแนปส์ทั้งหมด โดยใช้การสุ่มค่าแบบเกาส์เซียน

2.) เลือกคู่การฝึกหัด (X(k), T(k))

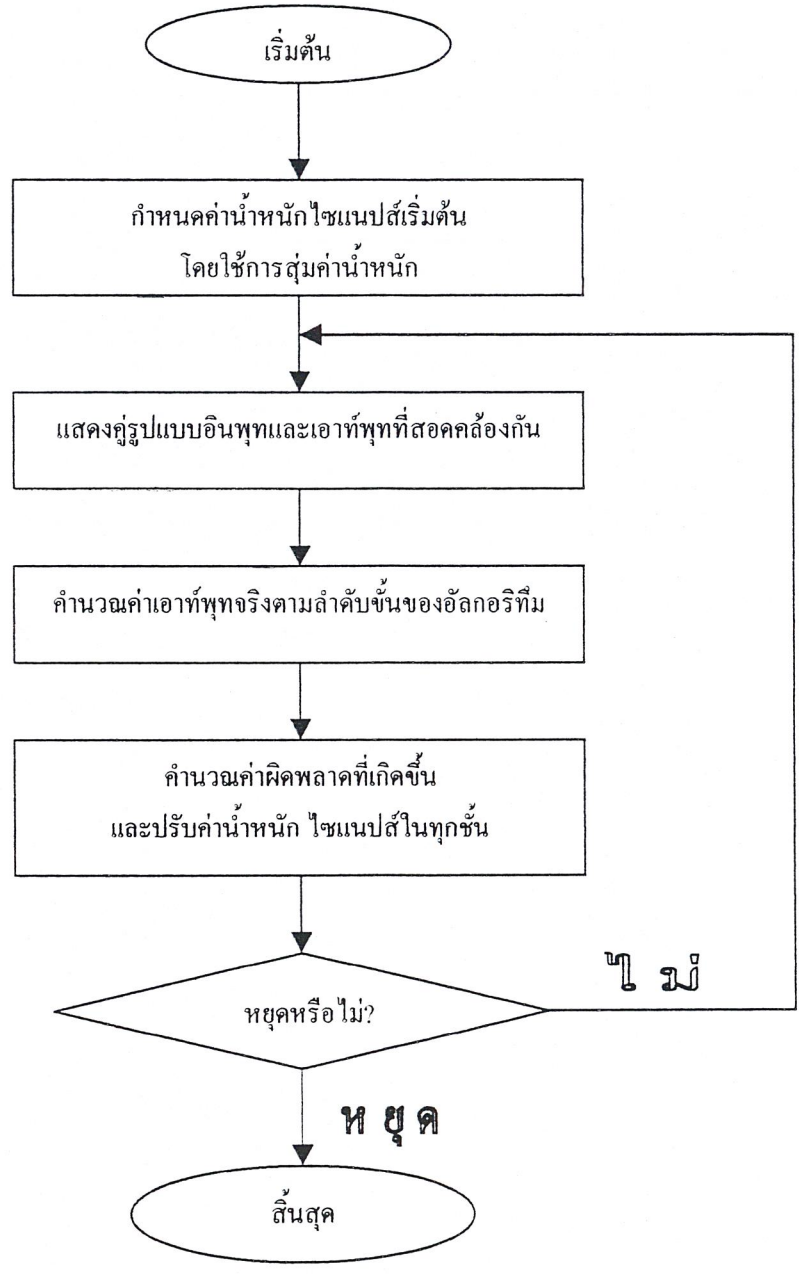
3.) คำนวณค่าเอาท์พุทจริงที่ได้จากโครงข่ายในชั้นเอาท์พุท โดยเริ่มต้นจากชั้นอินพุทและคำนวณไปที่ละชั้นๆ จนกระทั่งถึงชั้นเอาท์พุท L จากสมการ 2.13

$$O_i^l(k) = f\left(\sum_{m=0}^{N_{l-1}} W_{jm}^l O_m^{l-1}\right) \qquad (2.13)$$

4.) คำนวณค่าเกรเดียน δ_i และความแตกต่าง Δw_{ij} สำหรับแต่ละอินพุทของนิวรอลในแต่ละชั้น โดยเริ่มจากชั้นเอาท์พุทและปรับค่าไปทีละชั้นจนกระทั่งถึงชั้นอินพุท

5.) ปรับปรุงค่าน้ำหนักไซแนปส์

6.) กลับไปยังขั้นตอนที่ 2 - 5



รูปที่ 2-6 แสดง Flow chart ของโปรแกรม

บทที่ 3
การทำนายโหลด

ในปัจจุบันนี้ การทำนายโหลด (Load Forecasting) หรือการทำนายการใช้พลังงานไฟฟ้า (Eletric Demand Forecasting) นับว่าได้ว่าได้เข้ามามีบทบาทมากขึ้นทั้งในส่วนของความเสถียรภาพในระบบส่งจ่ายกำลังงานไฟฟ้า, การวางแผนการทำงานของโรงจักรไฟฟ้าและการจัดการบริหารทางด้านพลังงาน ทั้งนี้ก็เพื่อป้องกันอันตรายที่อาจจะเกิดขึ้นอันเนื่องมาจากการผลิตกำลังงานไฟฟ้าที่มีขนาดไม่เหมาะสมต่อความต้องการกำลังไฟฟ้าของโหลด อีกทั้งยังก่อให้เกิดประโยชน์จากการใช้ทรัพยากรที่มีอยู่อย่างจำกัดได้อย่างมีประสิทธิภาพและสามารถวางแผนการเพิ่มหรือขยายหน่วยผลิตเพื่อรองรับการขยายตัวของความต้องการกำลังไฟฟ้าที่นับวันจะมีความต้องการมากขึ้น

3.1 ประเภทของการทำนายโหลด

การจำแนกประเภทของการทำนายโหลด สามารถจำแนกออกเป็นประเภทใหญ่ๆ ได้ตามช่วงระยะเวลาของการทำนายได้ดังต่อไปนี้ [2, 4] คือ

3.1.1 SHORT-TERM LOAD FORECASTING

หมายถึง การทำนายโหลดหรือการทำนายความต้องการกำลังไฟฟ้าที่จะเกิดขึ้นภายในระยะเวลาตั้งแต่ 1 ชั่วโมงล่วงหน้าจนถึง 168 ชั่งโมงหรือ 1 สัปดาห์ล่วงหน้า ซึ่งโดยส่วนใหญ่แล้วผลของการทำนายโหลดประเภทนี้มักจะนำใช้ในส่วนของการวางแผนการทำงานของโรงจักรไฟฟ้า (Unit Commitment) หรือการวางแผนการผลิตของโรงจักรไฟฟ้า (Scheduleing and Operative Reserve) เป็นต้น

3.1.2 MEDIUM-TERM LOAD FORECASTING

หมายถึง การทำนายโหลดที่จะเกิดขึ้นตั้งแต่ระยะเวลา 1 สัปดาห์ล่วงหน้าจนถึง 3 ปีล่วงหน้า ซึ่งผลที่ได้จากการทำนายโหลดในช่วงเวลาดังกล่าวจะนำมาใช้ในส่วนของการวางแผนการบริหารเชื้อเพลิงการผลิต (Fuel Mamagement) หรือการวางแผนการซ่อมบำรุง (Maintenance Planning) เป็นต้น

3.1.3 LONG-TERM LOAD FORECASTING

หมายถึง การทำนายโหลดที่จะเกิดขึ้นในระยะเวลาตั้งแต่ 3 ปีขึ้นไปโดยส่วนมากแล้วผลที่ได้จากการทำนายในช่วงเวลาดังกล่าว จะนำมาใช้ในส่วนของการวางแผนการขยายหน่วยการผลิต (Capacity Expansion Planning) ในอนาคต

3.2 ประโยชน์ของการทำนายโหลด

การทำนายโหลดนั้นมีความสำคัญและมีความจำเป็นควบคู่ไปกับการผลิตกำลังไฟฟ้า ซึ่งโดยทั่วไปแล้วสามารถจำแนกประโยชน์ของการทำนายได้ดังต่อไปนี้ [2] คือ

1.) เพื่อใช้ในการวางแผนการผลิตกำลังไฟฟ้าให้เพียงพอต่อความต้องการของโหลด
2.) เพื่อใช้ในการวางแผนการทำงานของแต่ละโรงจักรไฟฟ้าให้สามารถทำการผลิตกำลังไฟฟ้าโดยที่ก่อให้เกิดค่าใช้จ่ายที่ต่ำ (Economic Load Dispatch)
3.) ทำให้ระบบส่งจ่ายกำลังไฟฟ้ามีความน่าเชื่อถือ (Reliability)
4.) เพื่อใช้ในส่วนของการวางแผนการซ่อมบำรุงของโรงจักรไฟฟ้า
5.) เพื่อใช้ในการวางแผนการสำรองเชื้อเพลิงการผลิตของโรงจักรไฟฟ้า
6.) ก่อให้เกิดการใช้ทรัพยากรที่มีอยู่อย่างมีประสิทธิภาพ
7.) เพื่อใช้ในการวางแผนการขยายหรือสร้างโรงจักรไฟฟ้าหรือเพิ่มหน่วยการผลิตเพื่อรองรับการขยายตัวของความต้องการกำลังไฟฟ้าที่จะเพิ่มขึ้นในอนาคต
8.) เพื่อใช้ในการวางแผนการแก้ปัญหาการขาดแคลนทรัพยากรหรือการทดแทนทรัพยากรในอนาคต

3.3 วิธีการทำนายโหลด

วิธีการทำนายโหลด สามารถจำแนกออกได้ตามลักษณะของวิธีการที่ใช้ดังต่อไปนี้ [4] คือ

3.3.1 SECTIONAL METHODS OR LOAD SURVEY METHODS

หมายถึง การทำนายโหลดโดยอาศัยการสำรวจความต้องการกำลังไฟฟ้าในแต่ละพื้นที่ โดยจะทำการแบ่งพื้นที่นั้นๆออกเป็นพื้นที่ย่อยหรือกริด (Grid) และทำการบันทึกค่าความต้องการกำลังไฟฟ้าและประเภทหรือลักษณะของโหลดในแต่ละพื้นที่ การทำนายโหลดด้วยวิธีนี้จะอาศัยการประเมิณถึงแนวโน้มของความต้องการกำลังไฟฟ้าที่จะมีเพิ่มขึ้นในอนาคต

3.3.2 METHODS OF EXTRAPOLATION

หมายถึง การทำนายโหลดโดยการอาศัยข้อมูลทางสถิติ กล่าวคือ อาศัยการจดบันทึกข้อมูลจัดเก็บในรูปแบบของข้อมูลทางสถิติ การทำนายโหลดจะใช้การนำข้อมูลที่ได้จากการบันทึกมาเขียนรูปกราฟและทำการทำนายถึงความต้องการกำลังไฟฟ้าในอนาคต โดยใช้การเปรียบเทียบกับความต้องการกำลังไฟฟ้าที่เพิ่มขึ้นในอดีต

3.3.3 MATHEMATICAL METHODS

หมายถึง วิธีการทำนายโหลดโดยอาศัยการใช้สมการทางคณิตศาสตร์แสดงความสัมพันธ์ระหว่างปัจจัยต่างๆที่มีผลกระทบต่อลักษณะของการเปลี่ยนแปลงของโหลด แล้วจึงใช้สมการทางคณิตศาสตร์นั้นๆทำนายถึงความต้องการกำลังไฟฟ้า เช่น วิธีการใช้สมการการถดถอยกำลังสองเคลื่อนที่ (Auto Regressive Moving Average หรือ ARMA) [7] (ภาคผนวก ก.) เป็นต้น

3.3.4 MATHEMATICAL METHODS CONSIDERING ECONOMIC PARAMETERS OR ENERGONOMIC

หมายถึง วิธีการทำนายโดยอาศัยสมการทางคณิตศาสตร์แสดงถึงความสัมพันธ์ระหว่างปัจจัยต่างๆกับลักษณะของการเปลี่ยนแปลงของโหลด (จากวิธี Mathematical Methods) ร่วมกันกับการพิจารณาถึงผลกระทบของสภาวะปัจจัยทางเศรษฐกิจที่มีอิทธิพลต่อการเปลี่ยนแปลงของความต้องการกำลังไฟฟ้า

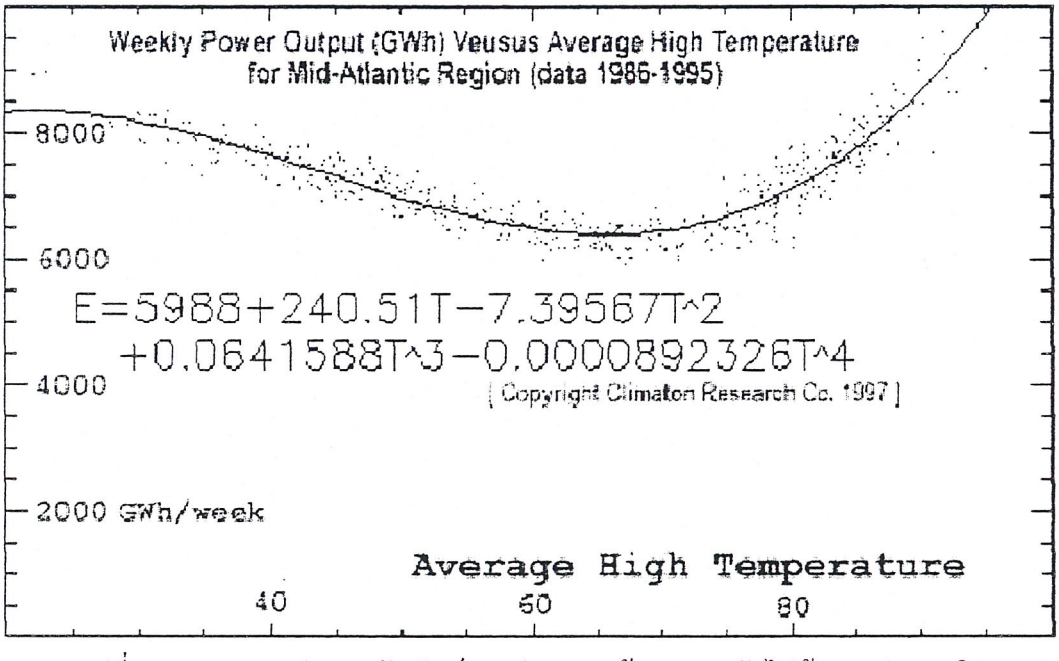
3.4 ปัจจัยที่มีอิทธิพลต่อการทำนายโหลด

ในการทำนายโหลดให้มีความถูกต้องแม่นยำ จำเป็นที่เราจะต้องคำนึงถึงปัจจัยต่างๆที่มีผลกระทบต่อการเปลี่ยนแปลงของโหลดมาพิจารณาในการจำลองลักษณะของสมการทางคณิตศาสตร์ หรือจำลองโมเดล (Model) ของระบบโครงข่ายประสาทเทียม (Artificial Neural Networks) ซึ่งปัจจัยต่างๆที่มีผลกระทบต่อการเปลี่ยนแปลงของโหลดนั้นประกอบด้วย [2, 4]

1.) โหลด (Load)

- โหลดสูงสุดในแต่ละวัน (Daily peak load)
- โหลด ณ เวลาใดๆ (Load at any time)

2.) อุณหภูมิ (Temperature)



รูปที่ 3-1 แสดงกราฟความสัมพันธ์ระหว่างความต้องการกำลังไฟฟ้าและอุณหภูมิ

(REPORTED BY EEI.)

3.) อุณหภูมิหยดน้ำค้าง (Dew-point temperature)

4.) ความชื้น (Humidity)

5.) ความเร็วลมและทิศทางลม (Wind speed and direction)

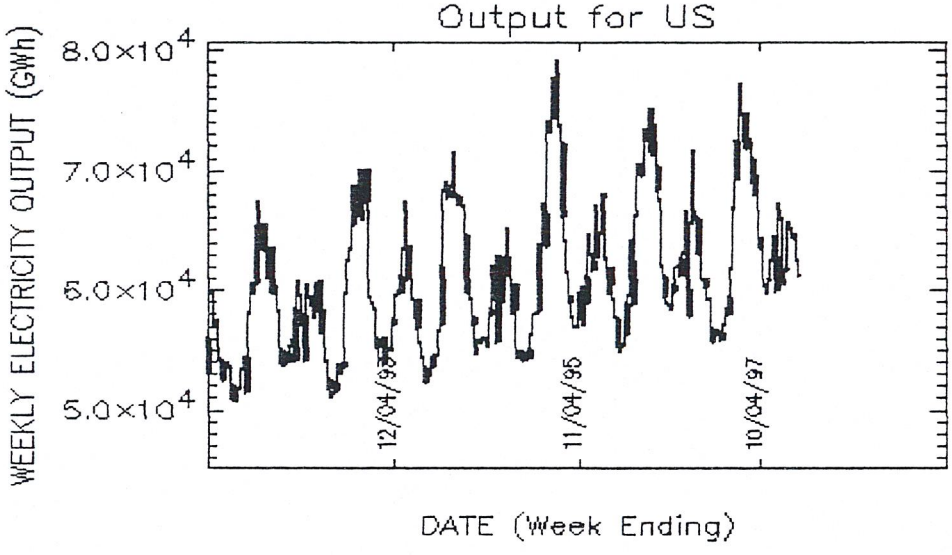
6.) ปริมาณเมฆ (Cloud)

7.) ดัชนีแสง (Light index)

8.) ช่วงระยะเวลาที่มีแสงแดด (Length of daylight)

9.) ปริมาณน้ำฝน (Rain)

10.) ฤดูกาล (Seasons)



รูปที่ 3-2 แสดงกราฟความสัมพันธ์ระหว่างความต้องการกำลังไฟฟ้าและฤดูกาล

(REPORT BY EEI.)

11. วันในสัปดาห์ (Day of week)

12. วันหยุด (Holidays)

13. กิจกรรมทางเศรษฐกิจ (Economic Activities)

- ผลผลิตทางอุตสาหกรรม (Industrial Production)
- ผลผลิตมวลรวมของประเทศ (Gross Nation Product)
- การบริโภคพลังงานดิบ (Raw Energy Consumption)
- การบริโภควัตถุดิบ (Material Consumption)

14.) ราคาเชื้อเพลิง (Cost of Energy)

15.) การขยายตัวของประชากร (Population Distribution)

16.) การขยายตัวของแรงงาน (Labor Distribution)

17.) รายได้ต่อบุคคล (Income per person)

18.) การบริโภคพลังงานไฟฟ้าต่อบุคคล (Electrical Energy Consumption per person)

บทที่ 4
การทำนายการใช้กำลังไฟฟ้าโดยใช้โครงข่ายประสาทเทียม

4.1 การจัดเตรียมโครงข่ายประสาทเทียม

โครงข่ายประสาทเทียมที่ได้ใช้ในการทดลองครั้งนี้ ได้ใช้โครงข่ายประสาทที่มีลักษณะการทำงานแบบคำนวณไปข้างหน้า (Feed–Forward Neural Networks) และฟังก์ชั่นกระตุ้นที่ใช้ได้ใช้ฟังก์ชั่นกระตุ้นแบบไม่เป็นเชิงเส้น (Nonlinear Activate Function) ได้แก่ ฟังก์ชั่นซิกมอยด์ (Sigmoid Function) ทั้งนี้ก็เนื่องมาจากฟังก์ชั่นกระตุ้นที่ไม่เป็นเชิงเส้นนั้นมีความสามารถในการสร้างขอบเขตของการตัดสินใจที่มีลักษณะโค้งมนได้ จึงทำให้สามารถแบ่งกลุ่มรูปแบบข้อมูลที่มีรูปแบบค่อนข้างซับซ้อนซึ่งมีความเหมาะสมกับการทำนายโหลด

ในการจัดเตรียมโครงข่ายประสาทเพื่อใช้ในการศึกษาการทำนายการใช้กำลังไฟฟ้าในครั้งนี้ ได้ทำการทดลองด้วยโครงข่ายประสาทเทียมแบบ 2 ชั้น ซึ่งประกอบไปด้วย 1 ชั้นอินพุท (Input Layer), 1 ชั้นซ่อน (Hidden Layer) และ 1 ชั้นเอาท์พุท (Output Layer) โดยที่แต่ละโครงข่ายจะมีความแตกต่างกันตามจำนวนโหนดภายในของโครงข่ายหรือตามรูปแบบของอินพุทที่ใช้ในโครงข่าย

เนื่องจากรูปแบบของอินพุทของโครงข่ายที่ได้ใช้ในการทดลองนั้นมีความแตกต่างกันทั้งในด้านของจำนวนและลักษณะของรูปแบบอินพุท ดังนั้นจำนวนโหนด (Node) ของชั้นต่างๆ ภายในโครงข่ายจึงมีความแตกต่างกันตามรูปแบบของอินพุท โดยที่จำนวนโหนดของชั้นอินพุทของแต่ละโครงข่ายจะมีจำนวนเท่ากับจำนวนอินพุทของโครงข่ายนั้นๆ ส่วนจำนวนของโหนดภายในชั้นซ่อนได้กำหนดตามทฤษฎีของ Kolmogorov [1] โดยได้กำหนดจำนวนโหนดในชั้นซ่อนเท่ากับ 2N+1 โหนด โดยที่ Nคือจำนวนอินพุทของโครงข่ายๆ ส่วนจำนวนของโหนดในชั้นเอาท์พุทจะมีจำนวน 3 โหนด เนื่องจากการทดลองในครั้งนี้ ได้กำหนดให้เป็นการทำนายโหลดรายชั่วโมงล่วงหน้าตั้งแต่ 1 ชั่วโมงล่วงหน้าจนกระทั่งถึง 3 ชั่วโมงล่วงหน้า (ทั้งนี้เนื่องจากข้อกำหนดของข้อมูลสภาวะอากาศจากกรมอุตุนิยมวิทยาที่มีลักษณะของข้อมูลเป็นราย 3 ชั่วโมง)

4.2 การจัดเตรียมข้อมูลที่ใช้ในการฝึกหัดโครงข่าย

จากที่ได้กล่าวมาแล้วในส่วนของการจัดเตรียมโครงข่ายประสาทเทียมว่า ในแต่ละโครงข่ายนั้นจะมีรูปแบบของอินพุทที่แตกต่างกัน ดังนั้นข้อมูลสำหรับการฝึกหัดโครงข่ายจะมีลักษณะแตกต่างกัน ซึ่งโดยรวมแล้วมีดังต่อไปนี้ คือ

- อุณหภูมิ
- ความชื้น
- ความเร็วลม
- โหลด
- เวลา

ซึ่งข้อมูลของสภาวะอากาศที่ได้ใช้ในการทดลองในครั้งนี้ อาทิเช่น อุณหภูมิ, ความชื้นและความเร็วลม ได้ใช้ค่าที่ได้จากการวัดในจังหวัดที่มีกำลังการใช้ไฟฟ้าค่อนข้างมากของแต่ละภูมิภาคของประเทศไทย โดยกำหนดให้ค่าของสภาวะอากาศที่ได้จากการวัดภายในจังหวัดนั้นๆ แทนสภาวะอากาศของภูมิภาคตามลักษณะทางภูมิศาสตร์ ซึ่งทั้งนี้ก็เพื่อเป็นการลดจำนวนของอินพุทที่จะใช้ในโครงข่ายลง

ซึ่งจังหวัดที่ได้กำหนดไว้ในการทดลองครั้งนี้ได้แก่

- กรุงเทพมหานคร
- นครสวรรค์
- เชียงใหม่
- ขอนแก่น
- ระยอง
- สงขลา
- กาญจนบุรี

จากจังหวัดที่ได้อ้างอิงถึงสภาวะอากาศนั้น จะเห็นได้ว่า กรุงเทพมหานครเป็นจังหวัดที่มีการใช้กำลังไฟฟ้าสูงสุดภายในประเทศ คือ มีอัตราการใช้กำลังไฟฟ้าถึงประมาณ 42.03 % หรือประมาณ 6185 เมกะวัตต์ในช่วงเวลาโหลดสูงสุด (ข้อมูลจากรายงานประจำเดือนของการไฟฟ้าฝ่ายผลิต ประจำเดือนสิงหาคม ปี 2541) ในขณะที่ภูมิภาคอื่นๆมีอัตราการใช้กำลังไฟฟ้าที่ต่ำกว่ามาก อาทิเช่น ภาคเหนือ มีอัตราการใช้กำลังไฟฟ้าประมาณ 9.48 % ในขณะที่ภาคตะวันออกเฉียงเหนือมีอัตราการใช้ไฟฟ้าประมาณ 10.02 % ดังนั้นเราควรคำนึงถึงปัจจัยทางสภาวะอากาศที่เกิดขึ้นในกรุงเทพมหานครเป็นสำคัญ

ทั้งนี้เนื่องจากการเปลี่ยนแปลงของสภาวะอากาศที่เกิดขึ้นในกรุงเทพมหานคร อาจมีผลให้โหลดเกิดการเปลี่ยนแปลงมากกว่าในภูมิภาคอื่นๆ

4.3 ค่าน้ำหนักไซแนปส์เริ่มต้นของโครงข่าย

ค่าน้ำหนักไซแนปส์เริ่มต้นของโครงข่ายในการทดลองในครั้งนี้ จะได้มาจากการสุ่มตัวเลขที่มีการกระจายแบบเกาซ์เซี่ยน (Gaussian) หรือเรียกว่าการสุ่มแบบนอร์มอล (Normal) ซึ่งเป็นการสุ่มตัวเลขที่มีการกระจายในทางสถิติโดยจะมีการกระจายความหนาแน่นตามฟังก์ชั่น ดังสมการที่ 4.1

$$f(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{(x-m)^2}{2\sigma^2}} \qquad (4.1)$$

โดยที่ m เป็นค่าเฉลี่ยเลขคณิต (Mean)

σ^2 เป็นค่าความแปรปรวน (Variance)

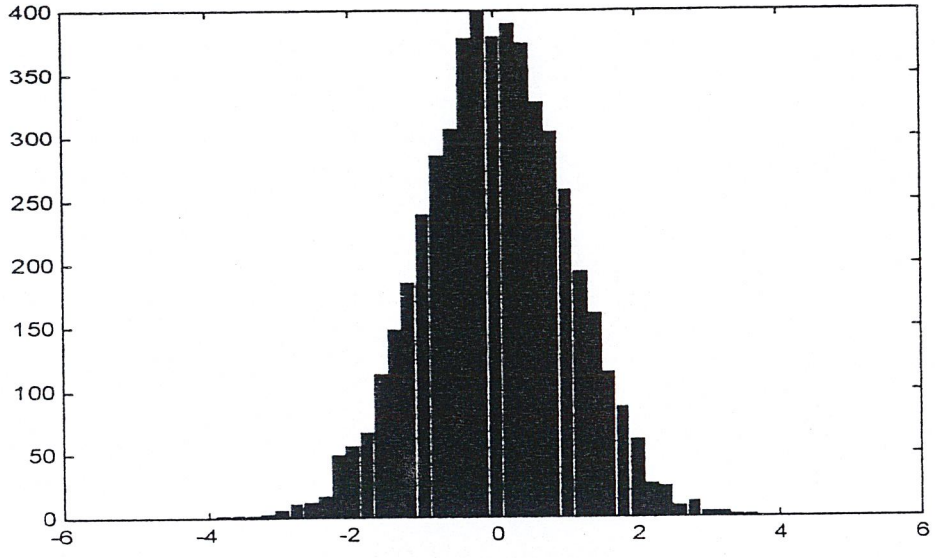
จะพบว่าการกระจายแบบเกาซ์เซี่ยนเมื่อให้ n มีค่ามากๆ ในขณะที่ให้ m เป็นค่าเฉลี่ยเลขคณิตศาสตร์ ที่มีค่าจำกัดที่แน่นอน และให้ค่าความแปรปรวนมีค่ามากกว่าหนึ่งมากๆ

การสุ่มแบบเกาซ์เซี่ยนโดยมากจะเกิดขึ้นในบริเวณแคบๆรอบค่าเฉลี่ย แต่บางครั้งอาจเกิดขึ้นห่างจากค่าเฉลี่ยก็ได้ เราสามารถใช้การสุ่มแบบเกาซ์เซี่ยนโดยใช้คำสั่งใน Matlab ดังนี้

```
randn(m,n);
```

เราจะได้ เมตริกซ์ขนาด m x n ที่ประกอบด้วยตัวเลขสุ่มแบบเกาซ์เซี่ยน

จากรูป 4.1 จะเห็นได้ว่าในการสุ่มตัวเลขแบบเกาซ์เซี่ยนจะได้ตัวเลขจากการสุ่มที่มีการกระจายในลักษณะของรูปทรงระฆัง (Bell-Curve) โดยจะพบว่ามีการกระจายของข้อมูลที่มีความหนาแน่นสูงอยู่ในช่วงใกล้ค่าเฉลี่ย (Mean)



รูปที่ 4-1 แสดงฮีสโตแกรม (Histogram) ของการสุ่มเลขแบบเกาซ์เซี่ยน จำนวน 5000

4.4 การฝึกหัดโครงข่าย

ในการทดลองครั้งนี้ ได้ทำการเปรียบเทียบความสามารถในการทำนายโหลดของโครงข่ายประสาทแต่ละโครงข่ายซึ่งมีความแตกต่างกันในส่วนของอินพุททั้งหมด 8 โครงข่าย โดยลักษณะทั่วไปของโครงข่ายประสาทจะใช้ลักษณะโครงข่ายแบบฟีดฟอร์เวิร์ดขนาด 2 ชั้นเช่นเดียวกัน คือ ประกอบด้วยชั้นของอินพุท, ชั้นซ่อนจำนวน 1 ชั้น และชั้นเอาท์พุท

4.4.1 โครงข่าย A

โครงข่ายประสาทเทียม A จะใช้อินพุทจำนวน 53 อินพุท ซึ่งประกอบไปด้วย อุณหภูมิขณะทำการทำนายของแต่ละพื้นที่ (7), อุณหภูมิเมื่อ 3 ชั่วโมงที่ผ่านมาของแต่ละพื้นที่ (7), อุณหภูมิขณะทำการทำนายของสัปดาห์ที่ผ่านมาของแต่ละพื้นที่ (7), อุณหภูมิเมื่อ 3 ชั่วโมงที่ผ่านมาของสัปดาห์ที่ผ่านมาของแต่ละพื้นที่(7), อุณหภูมิของ 3 ชั่วโมงล่วงหน้านับจากขณะทำการทำนายของสัปดาห์ที่ผ่านมาของแต่ละพื้นที่ (7),ความชื้นสัมพัทธ์ขณะทำนายของแต่ละพื้นที่ (7), ความเร็วลมขณะทำนายของแต่ละพื้นที่ (7), โหลดที่เกิดขึ้นขณะทำการทำนาย (1) และโหลดที่เกิดขึ้นของแต่ละชั่วโมงที่เกิดขึ้นภายใน 3 ชั่วโมงล่วงหน้าไปของสัปดาห์ที่ผ่านมา(3)

4.4.2 โครงข่าย B

โครงข่ายประสาทเทียม B จะใช้อินพุทจำนวน 32 อินพุท ซึ่งประกอบไปด้วย อุณหภูมิขณะทำการทำนายของแต่ละพื้นที่ (7), อุณหภูมิเมื่อ 3 ชั่วโมงที่ผ่านมาของแต่ละพื้นที่ (7), ความชื้นสัมพัทธ์ขณะทำนายของแต่ละพื้นที่ (7), ความเร็วลมขณะทำนายของแต่ละพื้นที่ (7), โหลดที่เกิดขึ้นขณะทำการทำนาย (1) และโหลดที่เกิดขึ้นของแต่ละชั่วโมงที่เกิดขึ้นภายใน 3 ชั่วโมงล่วงหน้าไปของสัปดาห์ที่ผ่านมา (3)

4.4.3 โครงข่าย C

โครงข่ายประสาทเทียม C จะใช้อินพุทจำนวน 35 อินพุท ซึ่งประกอบไปด้วย อุณหภูมิขณะทำการทำนายของแต่ละพื้นที่ (7), อุณหภูมิเมื่อ 3 ชั่วโมงที่ผ่านมาของแต่ละพื้นที่ (7), ความชื้นสัมพัทธ์ขณะทำนายของแต่ละพื้นที่ (7), ความเร็วลมขณะทำนายของแต่ละพื้นที่ (7), โหลดที่เกิดขึ้นขณะทำการทำนาย (1), โหลดที่เกิดขึ้นของแต่ละชั่วโมงที่เกิดขึ้นภายใน 3 ชั่วโมงล่วงหน้าไปของสัปดาห์ที่ผ่านมา (3) และเวลาขณะที่ทำการทำนาย (3 ; โดยอาศัยการ decode เป็น binary จากจำนวนการทำนายทั้งสิ้น 8 ครั้งใน 1 วัน)

4.4.4 โครงข่าย D

โครงข่ายประสาทเทียม D จะใช้อินพุทจำนวน 33 อินพุท ซึ่งประกอบไปด้วย อุณหภูมิขณะทำการทำนายของแต่ละพื้นที่ (7), อุณหภูมิเมื่อ 3 ชั่วโมงที่ผ่านมาของแต่ละพื้นที่ (7), ความชื้นสัมพัทธ์ขณะทำนายของแต่ละพื้นที่ (7), ความเร็วลมขณะทำนายของแต่ละพื้นที่ (7), โหลดที่เกิดขึ้นขณะทำการทำนาย (1), โหลดที่เกิดขึ้นของแต่ละชั่วโมงที่เกิดขึ้นภายใน 3 ชั่วโมงล่วงหน้าไปของสัปดาห์ที่ผ่านมา (3) และเวลาขณะที่ทำการทำนาย (1)

4.4.5 โครงข่าย E

โครงข่ายประสาทเทียม E จะใช้อินพุทจำนวน 36 อินพุท ซึ่งประกอบไปด้วย อุณหภูมิขณะทำการทำนายของแต่ละพื้นที่ (7), อุณหภูมิเมื่อ 3 ชั่วโมงที่ผ่านมาของแต่ละพื้นที่ (7), ความชื้นสัมพัทธ์ขณะทำนายของแต่ละพื้นที่ (7), ความเร็วลมขณะทำนายของแต่ละพื้นที่ (7), โหลดที่เกิดขึ้นขณะทำการทำนาย (1), โหลดที่เกิดขึ้นของแต่ละชั่วโมงที่เกิดขึ้นภายใน 3 ชั่วโมงล่วงหน้าไปของสัปดาห์ที่ผ่านมา (3), เวลาขณะที่ทำการทำนาย (1) และเวลาของโหลดที่ต้องการทำนาย (3)

4.4.6 โครงข่าย F

โครงข่ายประสาทเทียม F จะใช้อินพุทจำนวน 28 อินพุท ซึ่งประกอบไปด้วย อุณหภูมิขณะทำการทำนายของแต่ละพื้นที่ (7), อุณหภูมิเมื่อ 3 ชั่วโมงที่ผ่านมาของแต่ละพื้นที่ (7), ความชื้นสัมพัทธ์ขณะทำนายของแต่ละพื้นที่ (7), โหลดที่เกิดขึ้นขณะทำการทำนาย (1), โหลดที่เกิดขึ้นของแต่ละชั่วโมงที่เกิดขึ้นภายใน 3 ชั่วโมงล่วงหน้าไปของสัปดาห์ที่ผ่านมา (3) และเวลาขณะที่ทำการทำนาย (3 ; โดยอาศัยการ decode เป็น binary จากจำนวนการทำนายทั้งสิ้น 8 ครั้งใน 1 วัน)

4.4.7 โครงข่าย G

โครงข่ายประสาทเทียม G จะใช้อินพุทจำนวน 45 อินพุท ซึ่งประกอบไปด้วย อุณหภูมิขณะทำการทำนายของแต่ละพื้นที่ (7), อุณหภูมิเมื่อ 3 ชั่วโมงที่ผ่านมาของแต่ละพื้นที่ (7), ความชื้นสัมพัทธ์ขณะทำนายของแต่ละพื้นที่ (7), โหลดที่เกิดขึ้นขณะทำการทำนาย (1) โหลดที่เกิดขึ้นของแต่ละชั่วโมงที่เกิดขึ้นภายใน 3 ชั่วโมงล่วงหน้าไปของสัปดาห์ที่ผ่านมา(3), เวลาขณะที่ได้ทำการทำนาย (5 ; ใช้การ decode เป็น binary จากเวลาตั้งแต่ 0 ถึง 24) และเวลาของโหลดที่ต้องการทำนาย (15 ; โดยแต่ละค่าใช้การ decode เป็น binary จากเวลาตั้งแต่ 0 ถึง 24)

4.4.8 โครงข่าย H

โครงข่ายประสาทเทียม H จะใช้อินพุทจำนวน 38 อินพุท ซึ่งประกอบไปด้วย อุณหภูมิขณะทำการทำนายของแต่ละพื้นที่ (7), อุณหภูมิเมื่อ 3 ชั่วโมงที่ผ่านมาของแต่ละพื้นที่ (7), โหลดที่เกิดขึ้นขณะทำการทำนาย (1) โหลดที่เกิดขึ้นของแต่ละชั่วโมงที่เกิดขึ้นภายใน 3 ชั่วโมงล่วงหน้าไปของสัปดาห์ที่ผ่านมา(3), เวลาขณะที่ได้ทำการทำนาย (5 ; ใช้การ decode เป็น binary จากเวลาตั้งแต่ 0 ถึง 24) และเวลาของโหลดที่ต้องการทำนาย (15 ; โดยแต่ละค่าใช้การ decode เป็น binary จากเวลาตั้งแต่ 0 ถึง 24)

ตารางที่ 4.1 แสดงจำนวนและรูปแบบของอินพุทที่ใช้ในโครงข่ายประสาทแต่ละโครงข่าย

รูปแบบอินพุทของโครงข่าย	จำนวนอินพุทของโครงข่าย							
	โครงข่าย A	โครงข่าย B	โครงข่าย C	โครงข่าย D	โครงข่าย E	โครงข่าย F	โครงข่าย G	โครงข่าย H
อุณหภูมิขณะทำนายของแต่ละพื้นที่	7	7	7	7	7	7	7	7
อุณหภูมิเมื่อ 3 ชั่วโมงที่ผ่านมาของแต่ละพื้นที่	7	7	7	7	7	7	7	7
อุณหภูมิขณะทำนายของแต่ละพื้นที่เมื่อสัปดาห์ที่ผ่านมา	7	-	-	-	-	-	-	-
อุณหภูมิเมื่อ 3 ชั่วโมงที่ผ่านมาของแต่ละพื้นที่เมื่อสัปดาห์ที่ผ่านมา	7	-	-	-	-	-	-	-
ความชื้นสัมพัทธ์ขณะทำนายของแต่ละพื้นที่	7	7	7	7	7	7	7	-
ความเร็วลมขณะทำนายของแต่ละพื้นที่	7	7	7	7	7	-	-	-
โหลดขณะทำนาย	1	1	1	1	1	1	1	1
โหลด ณ เวลาที่ต้องการทำนายของสัปดาห์ที่ผ่านมา	3	3	3	3	3	3	3	3
เวลาขณะทำการทำนาย	-	-	3	1	1	3	5	5
เวลาของโหลดที่ต้องการทำนาย	-	-	-	-	3	3	15	15
Total	53	32	35	33	36	28	45	38

บทที่ 5
ผลการวิเคราะห์โครงข่ายประสาทเทียม

5.1 ผลการฝึกหัดโครงข่ายประสาทเทียม

จากการทดลอง "การทำนายการใช้กำลังไฟฟ้าของประเทศไทยโดยการใช้โครงข่ายประสาท" ในครั้งนี้ ได้ทำการทดสอบการทำงานโดยใช้การเขียนโปรแกรมด้วยโปรแกรม 'MATLAB' Version 5.1 โดยปฏิบัติงานด้วยเครื่องคอมพิวเตอร์ส่วนบุคคล ประสิทธิภาพ CPU Intel Pentium Processor 133 MHz หน่วยความจำ 80 MB บน OS (Operating System) Windows NT Workstation

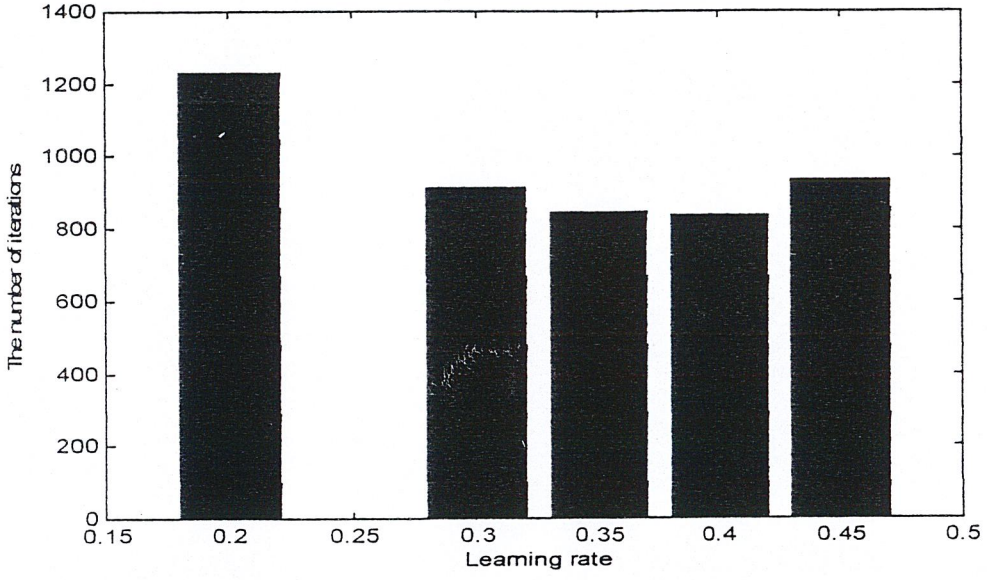
จากผลการทดลองในชั้นนี้ปรากฏว่า ค่าความผิดพลาดโดยรวมที่เกิดขึ้นของโครงข่ายจะมีลักษณะการลาดลงตามทฤษฎีของเกรเดียนเดสเซนต์ (ภาคผนวก ข.)

จากการพิจารณาการฝึกหัดของโครงข่ายแต่ละโครงข่ายนั้นพบว่า โครงข่ายจะมีการปรับปรุงค่าความผิดพลาดกำลังสองเฉลี่ย (Sum Square Error) ให้มีค่าลดลงเข้าสู่ค่าความผิดพลาดที่สามารถยอมรับได้ (Permissible Error) โดยที่ในช่วงเริ่มต้นของการฝึกหัดค่าความผิดพลาดกำลังสองเฉลี่ยจะมีการลดลงอย่างรวดเร็ว และจะค่อยๆปรับเรียบลงในรอบการเรียนรู้ต่อๆมา

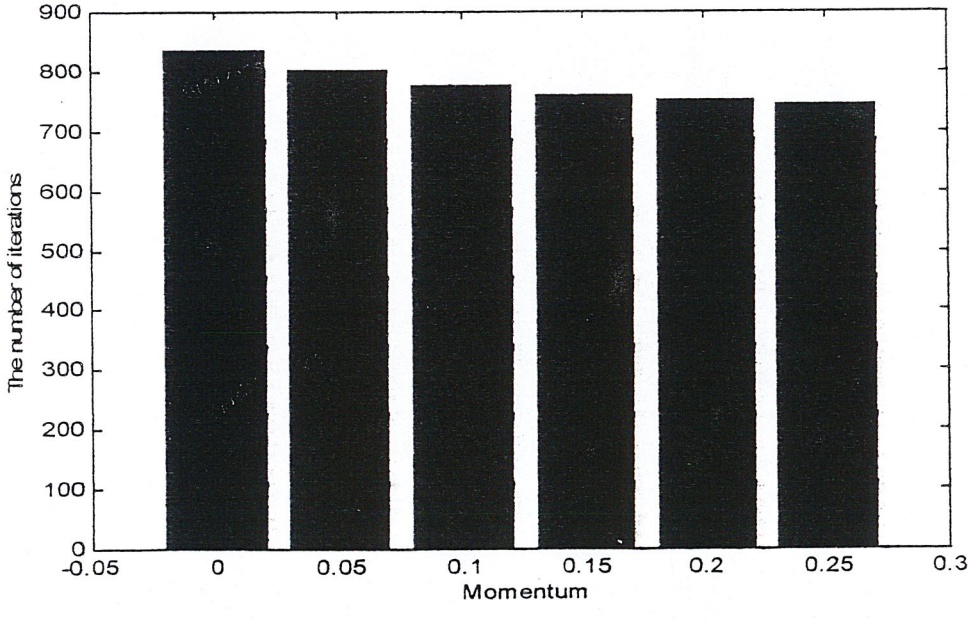
ในการฝึกหัดโครงข่ายในการทดลองครั้งนี้ ได้ใช้การปรับค่าพารามิเตอร์ที่สำคัญ คือ ค่าอัตราการเรียนรู้ (Learning rate) และค่าสัมประสิทธิ์โมเมนตัม (Momentum) เพื่อช่วยเร่งกระบวนการเรียนรู้ของโครงข่ายให้มีความรวดเร็วมากยิ่งขึ้น ซึ่งการปรับแต่งค่าพารามิเตอร์ดังกล่าว ก่อให้เกิดปัญหาบางประการตามมาซึ่งจะได้กล่าวถึงในภายหลัง

ผลการทดลองการฝึกหัดโครงข่ายประสาทที่ได้แสดงให้เห็นในบทนี้ ได้แสดงให้เห็นถึงผลของการฝึกหัดโครงข่ายประสาทต่างๆด้วยการทดลองการทำนายโหลดในวันจันทร์ ส่วนผลของการฝึกหัดโครงข่ายในการทดลองการทำนายโหลดในวันอังคาร, วันพุธ, วันพฤหัสและวันศุกร์จะไม่ได้แสดงให้เห็นแต่ขั้นตอนของการทดลองจะใช้หลักการเดียวกันกับการทดลองในวันจันทร์

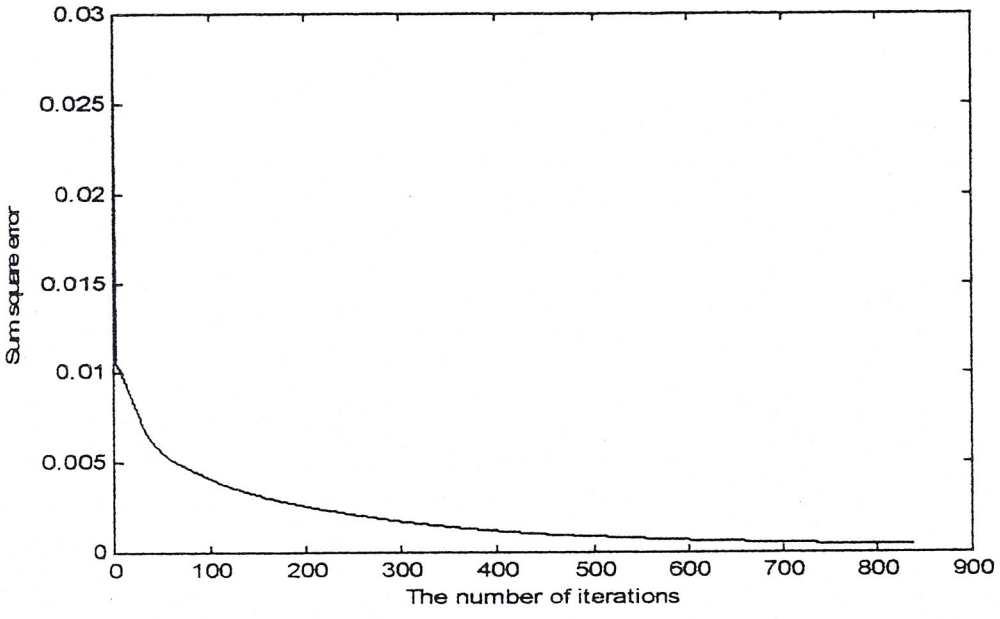
5.1.1 ผลการฝึกหัดโครงข่าย A



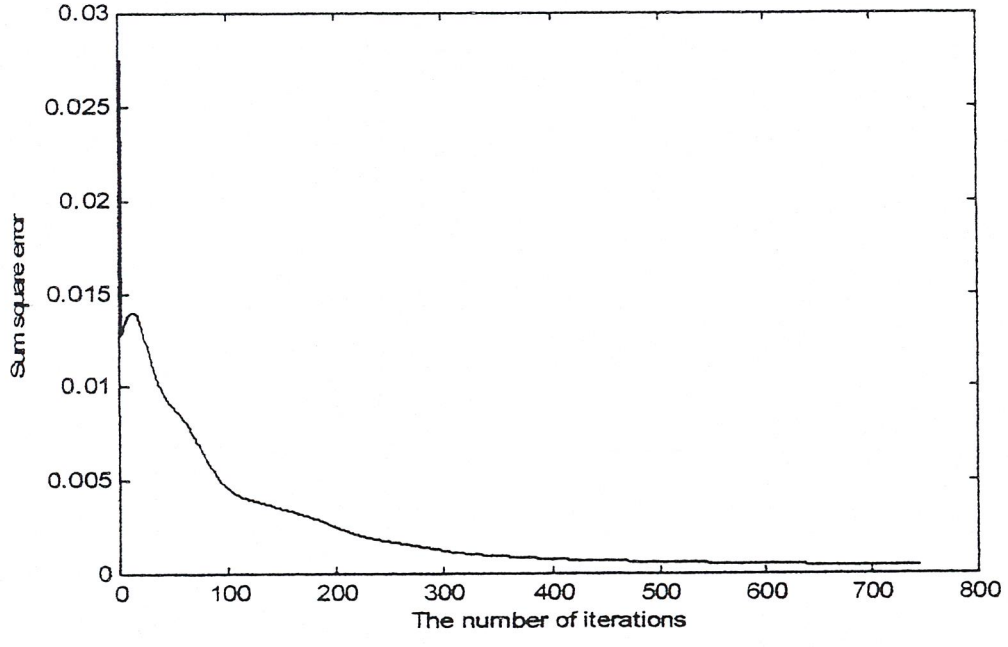
รูปที่ 5-1 แสดงความสัมพันธ์ของค่าอัตราการเรียนรู้และจำนวนรอบของการเรียนรู้ของโครงข่าย A ที่ค่าPermissible error = 0.0004



รูปที่ 5-2 แสดงความสัมพันธ์ของค่าสัมประสิทธิ์โมเมนตัมและจำนวนรอบของการเรียนรู้ของโครงข่าย A ที่ค่า Permissible error = 0.0004

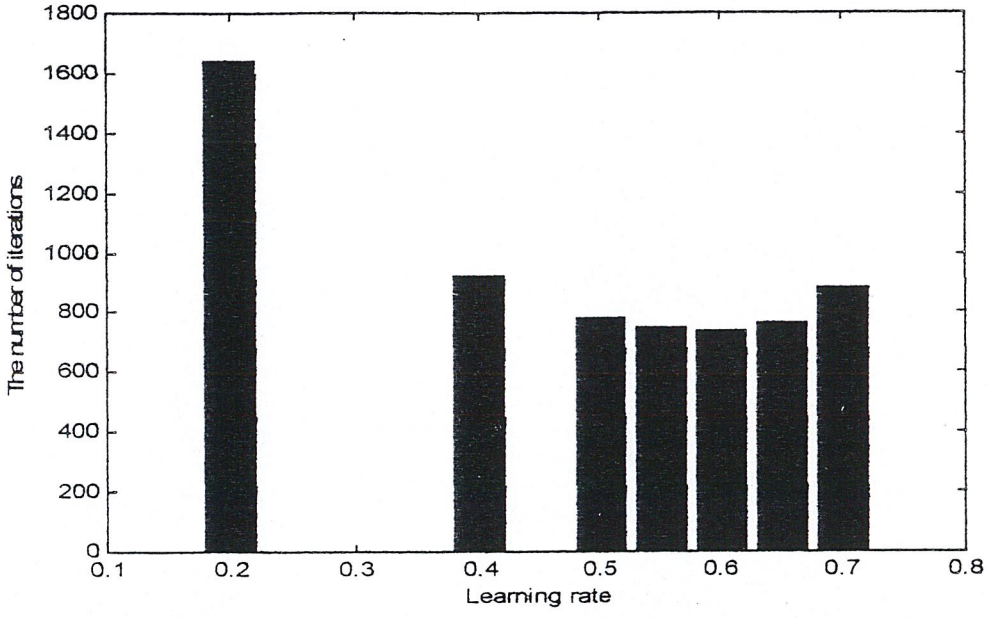


รูปที่ 5-3 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.4 และโมเมนตัม = 0 ของโครงข่าย A ที่ค่า Permissible error = 0.0004

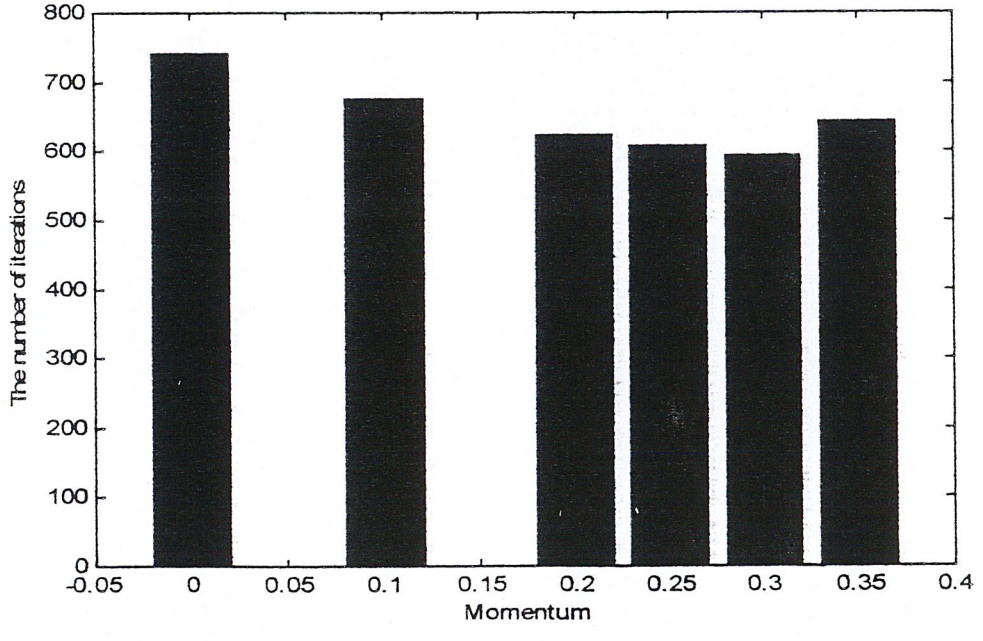


รูปที่ 5-4 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.4 และโมเมนตัม = 0.25 ของโครงข่าย A ที่ค่า Permissible error = 0.0004

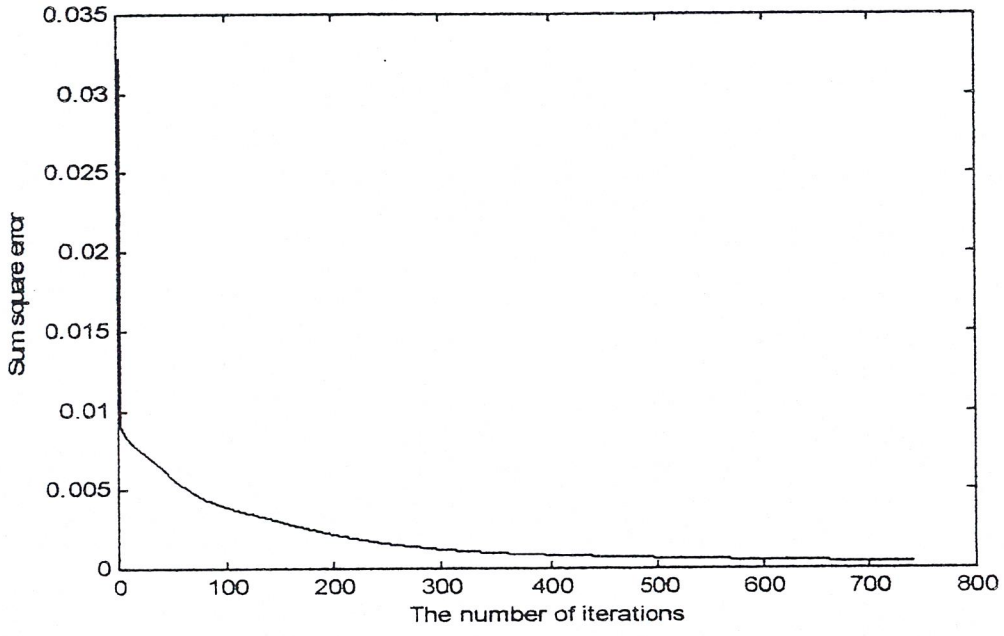
5.1.2 ผลการฝึกหัดโครงข่าย B



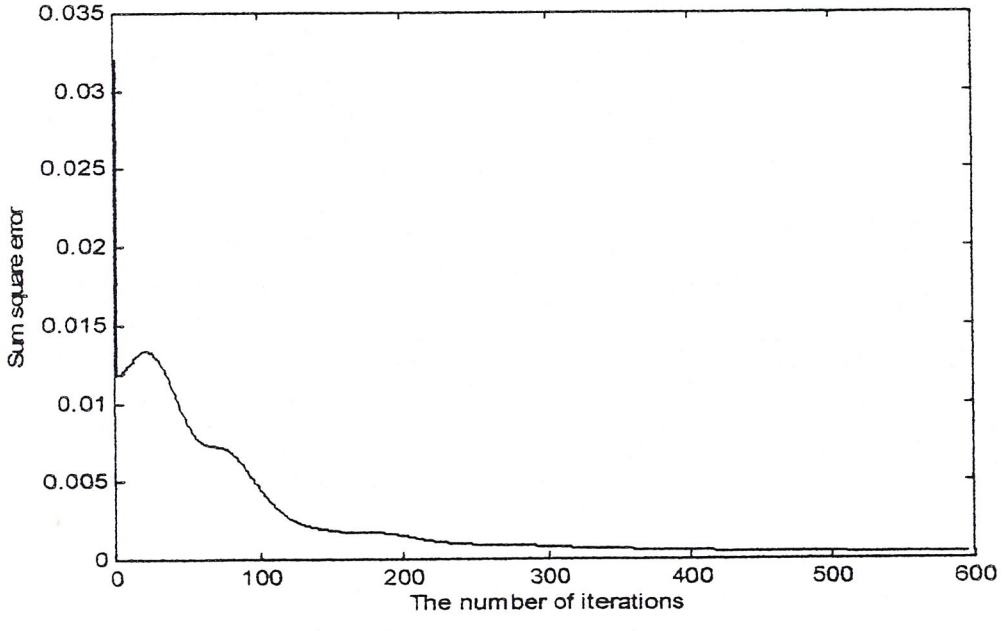
รูปที่ 5-5 แสดงความสัมพันธ์ของค่าอัตราการเรียนรู้และจำนวนรอบของการเรียนรู้ของโครงข่าย B ที่ค่า Permissible error = 0.0004



รูปที่ 5-6 แสดงความสัมพันธ์ของค่าสัมประสิทธิ์โมเมนตัมและจำนวนรอบของการเรียนรู้ของโครงข่าย B ที่ค่า Permissible error = 0.0004

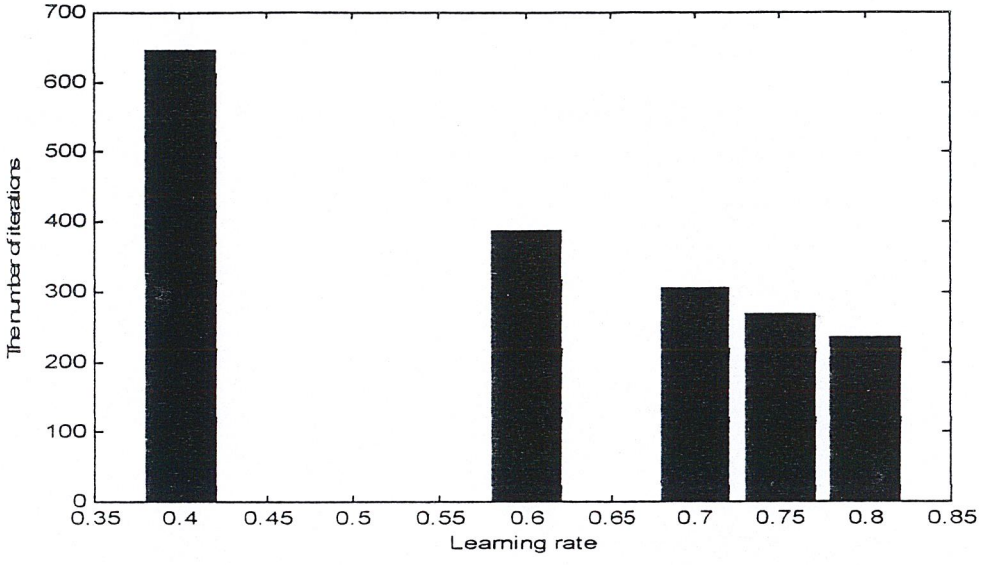


รูปที่ 5-7 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.6 และโมเมนตัม = 0 ของโครงข่าย B ที่ค่า Permissible error = 0.0004

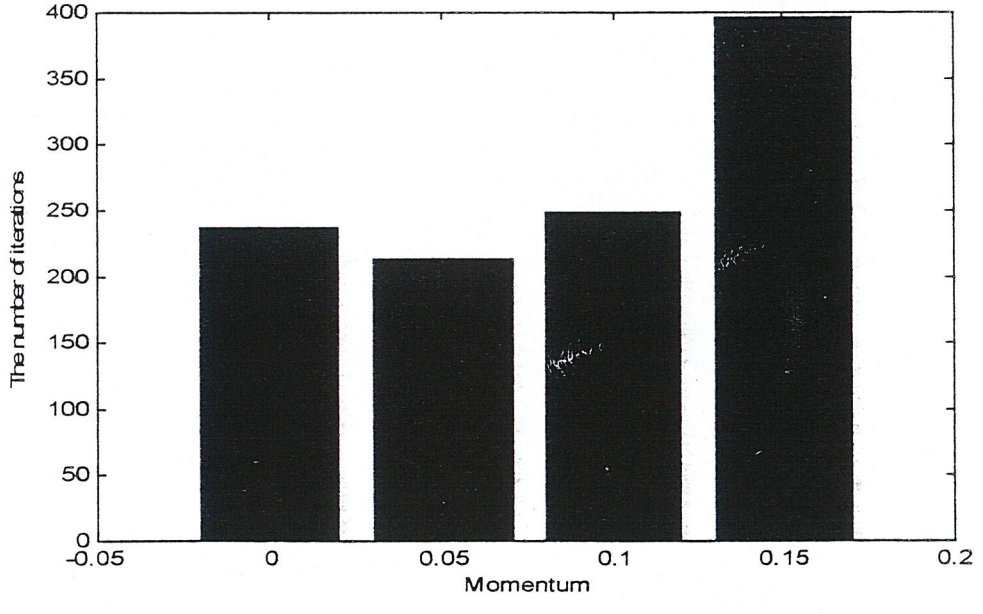


รูปที่ 5-8 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.6 และโมเมนตัม = 0.3 ของโครงข่าย B ที่ค่า Permissible error = 0.0004

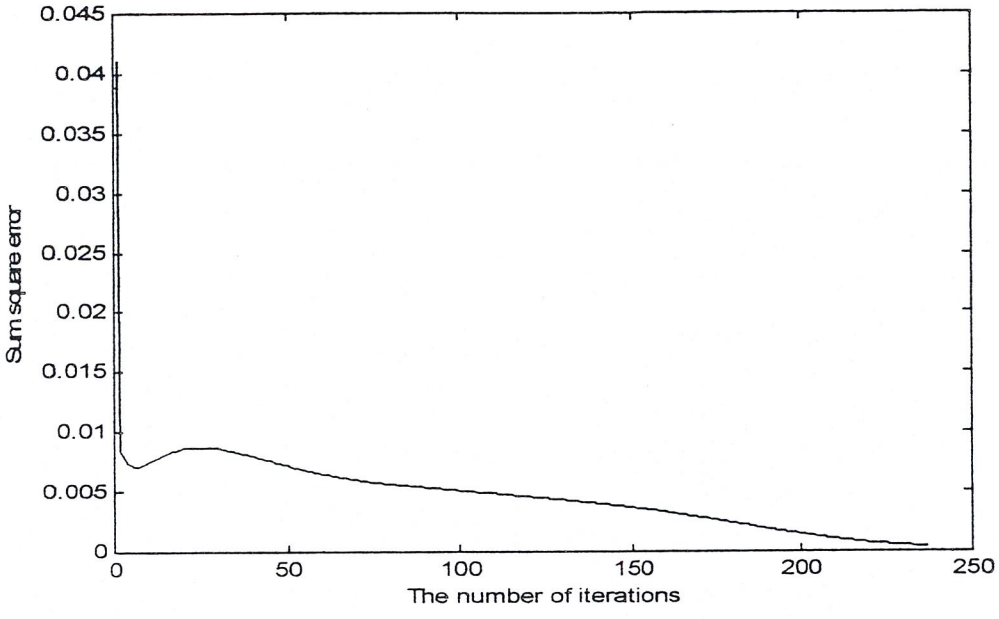
5.1.3 ผลการฝึกหัดโครงข่าย C



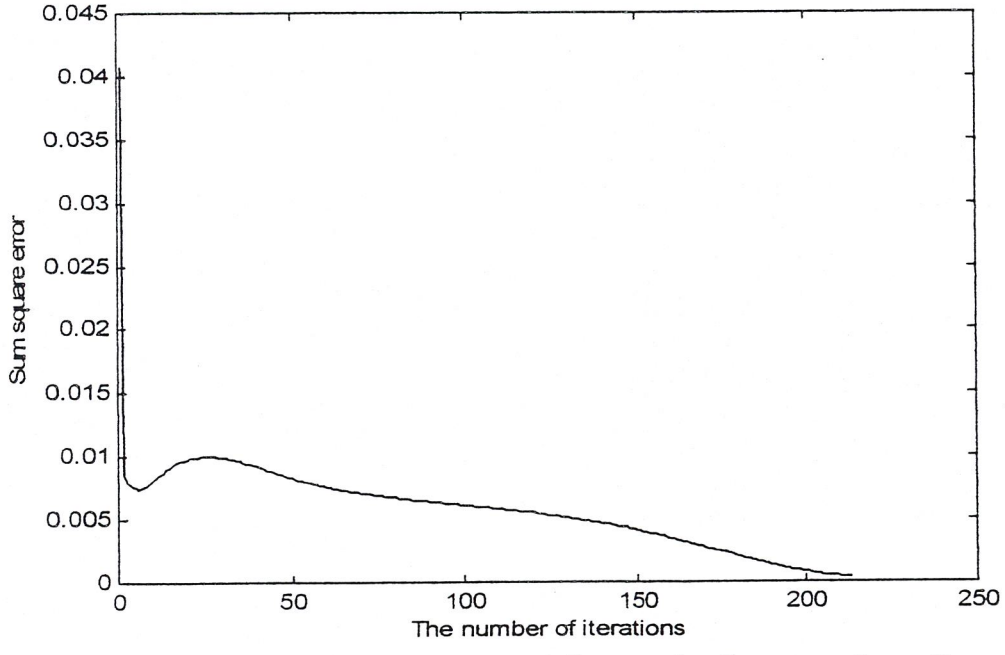
รูปที่ 5-9 แสดงความสัมพันธ์ของค่าอัตราการเรียนรู้และจำนวนรอบของการเรียนรู้ของโครงข่าย C ที่ค่า Permissible error = 0.0004



รูปที่ 5-10 แสดงความสัมพันธ์ของค่าสัมประสิทธิ์โมเมนตัมและจำนวนรอบของการเรียนรู้ของโครงข่าย C ที่ค่า Permissible error = 0.0004

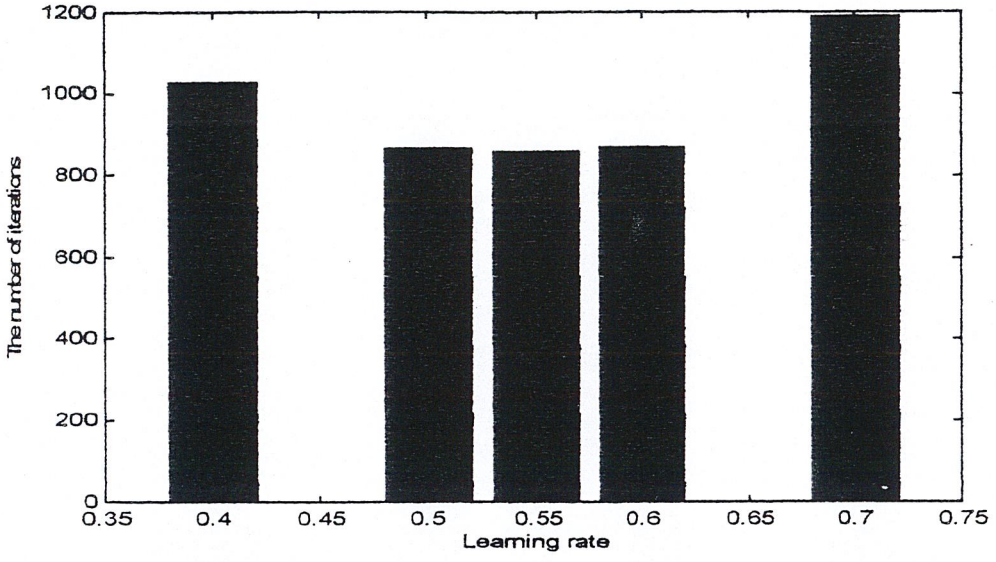


รูปที่ 5-11 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.8 และโมเมนตัม = 0 ของโครงข่าย C ที่ค่า Permissible error = 0.0004

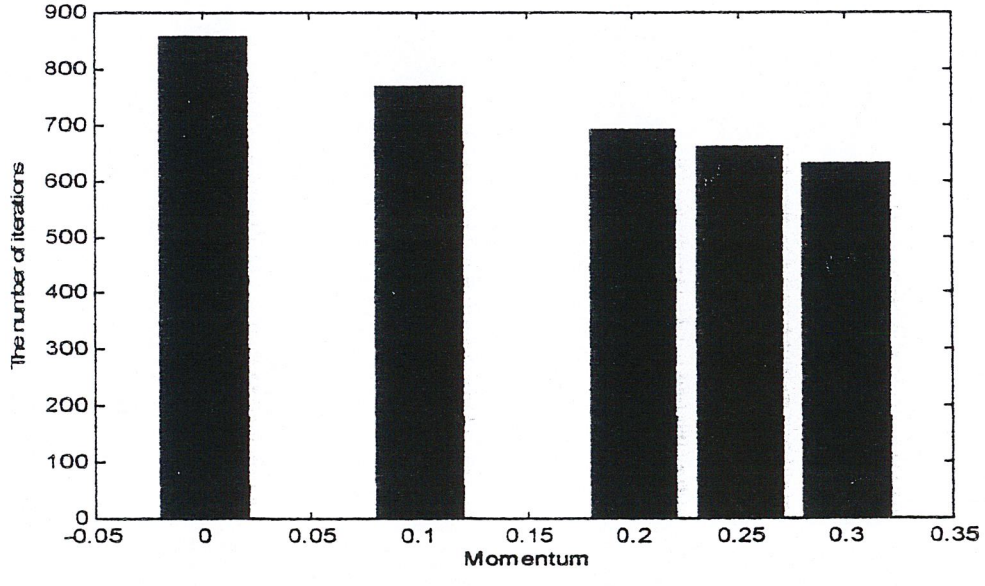


รูปที่ 5-12 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.8 และโมเมนตัม = 0.05 ของโครงข่าย C ที่ค่า Permissible error = 0.0004

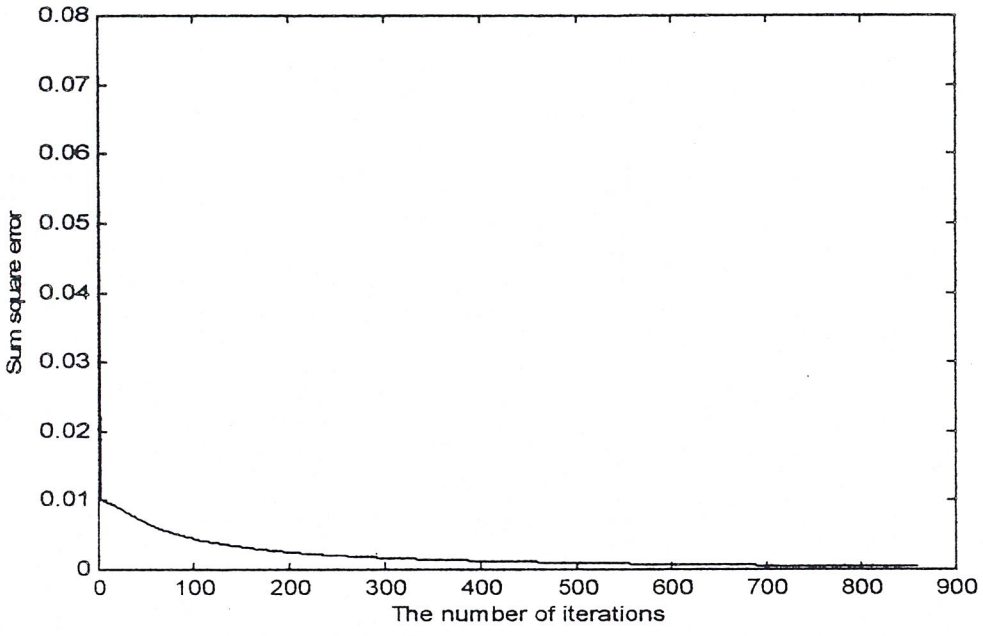
5.1.4 ผลการฝึกหัดโครงข่าย D



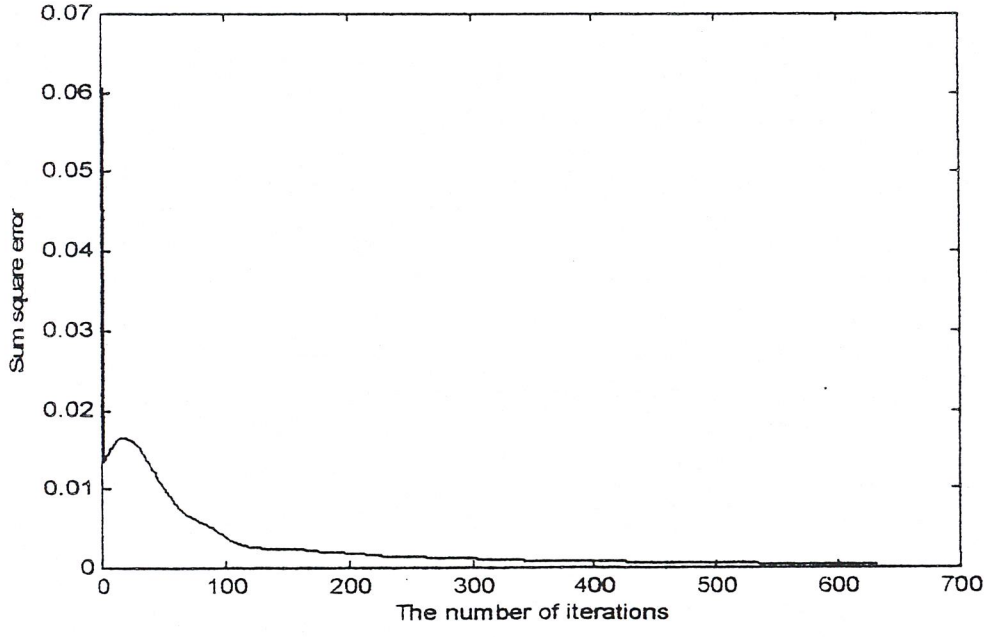
รูปที่ 5-13 แสดงความสัมพันธ์ของค่าอัตราการเรียนรู้และจำนวนรอบของการเรียนรู้ของโครงข่าย D ที่ค่า Permissible error = 0.0004



รูปที่ 5-14 แสดงความสัมพันธ์ของค่าสัมประสิทธิ์โมเมนตัมและจำนวนรอบของการเรียนรู้ของโครงข่าย D ที่ค่า Permissible error = 0.0004

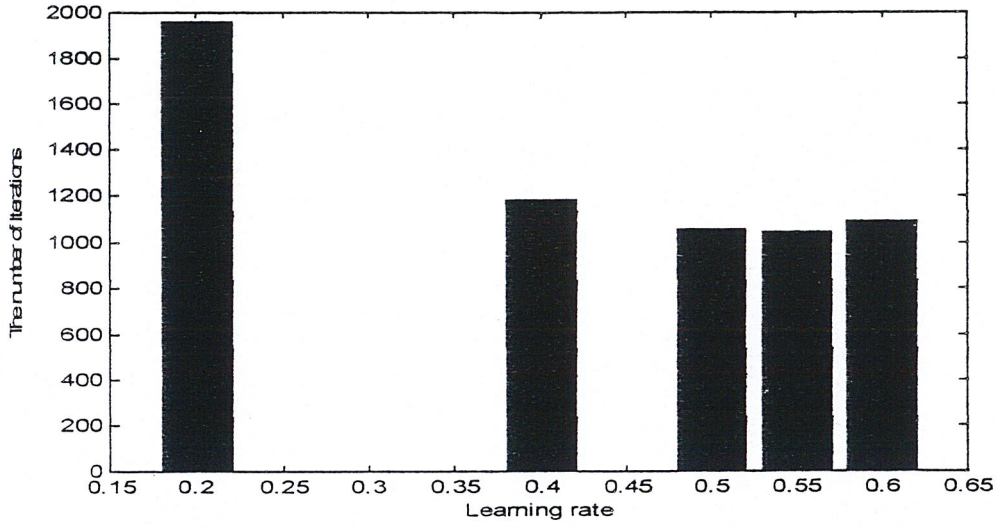


รูปที่ 5-15 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.55 และโมเมนตัม = 0 ของโครงข่าย D ที่ค่า Permissible error = 0.0004

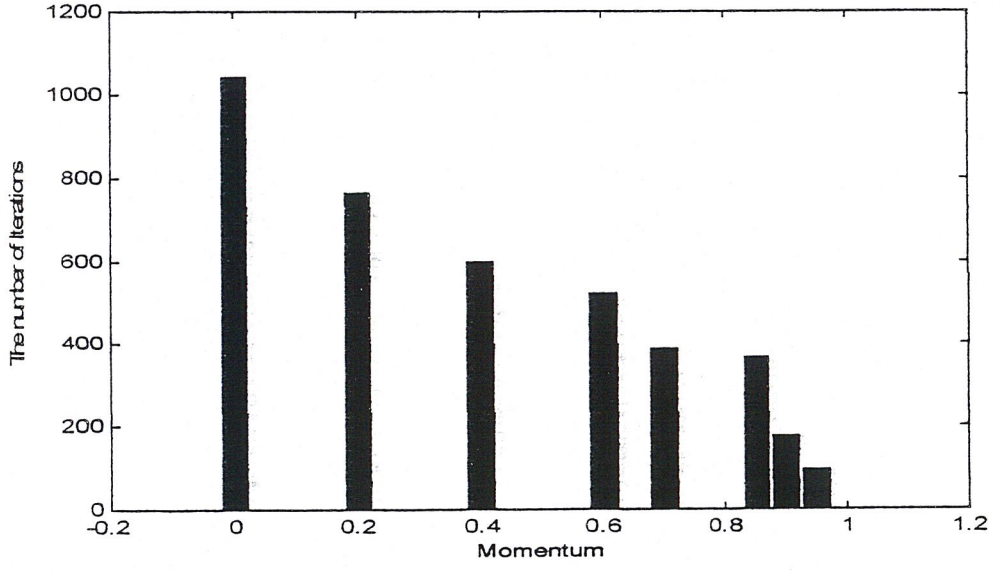


รูปที่ 5-16 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.55 และโมเมนตัม = 0.3 ของโครงข่าย D ที่ค่า Permissible error = 0.0004

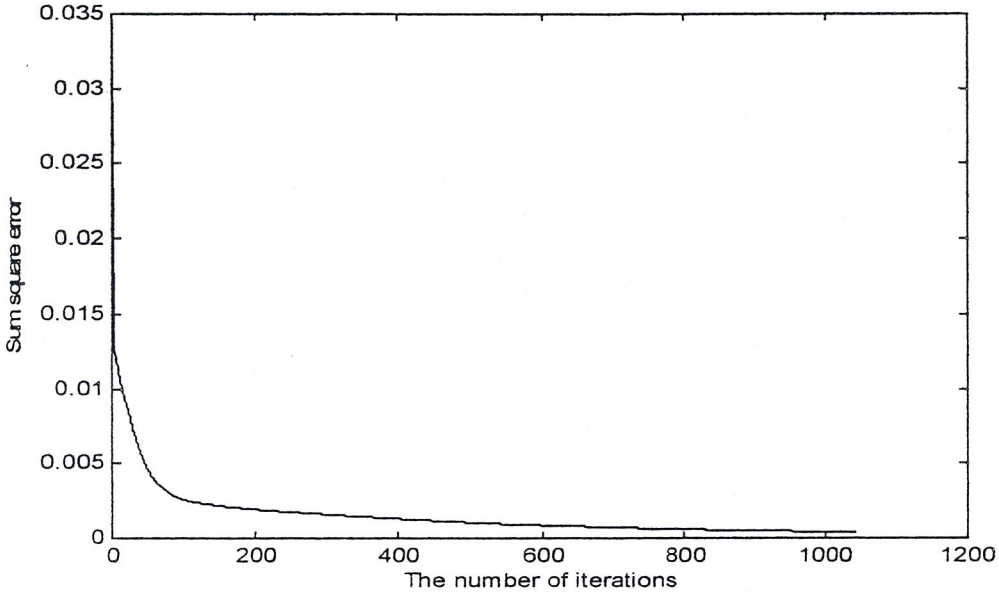
5.1.5 ผลการฝึกหัดโครงข่าย E



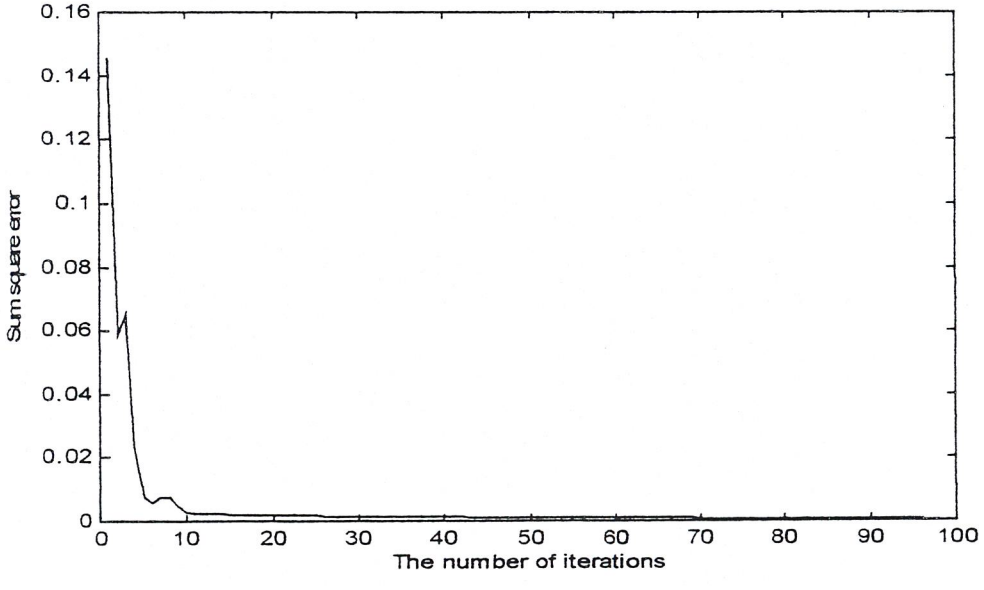
รูปที่ 5-17 แสดงความสัมพันธ์ของค่าอัตราการเรียนรู้และจำนวนรอบของการเรียนรู้ของโครงข่าย E ที่ค่า Permissible error = 0.0004



รูปที่ 5-18 แสดงความสัมพันธ์ของค่าสัมประสิทธิ์โมเมนตัมและจำนวนรอบของการเรียนรู้ของโครงข่าย E ที่ค่า Permissible error = 0.0004

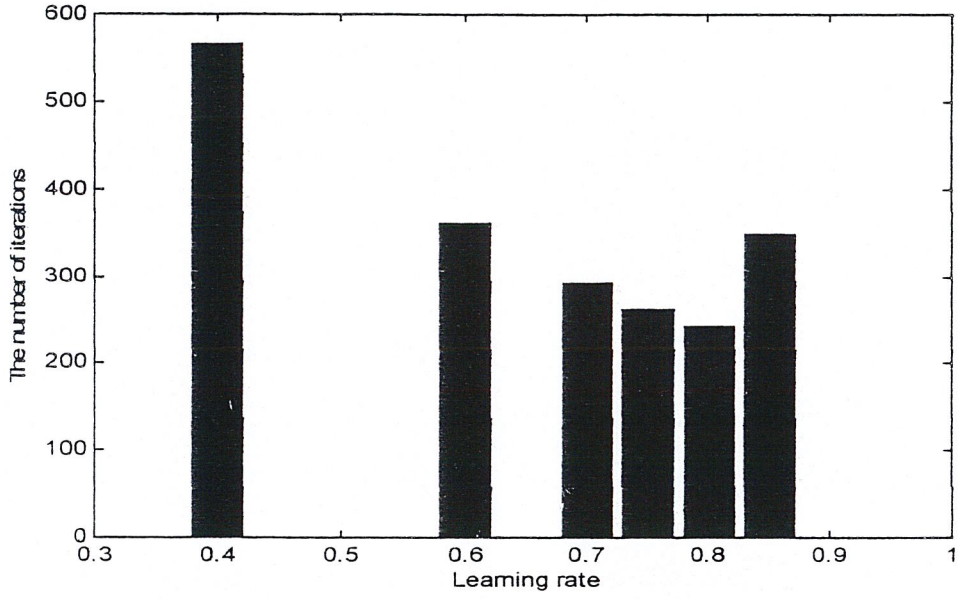


รูปที่ 5-19 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.55 และโมเมนตัม = 0 ของโครงข่าย E ที่ค่า Permissible error = 0.0004

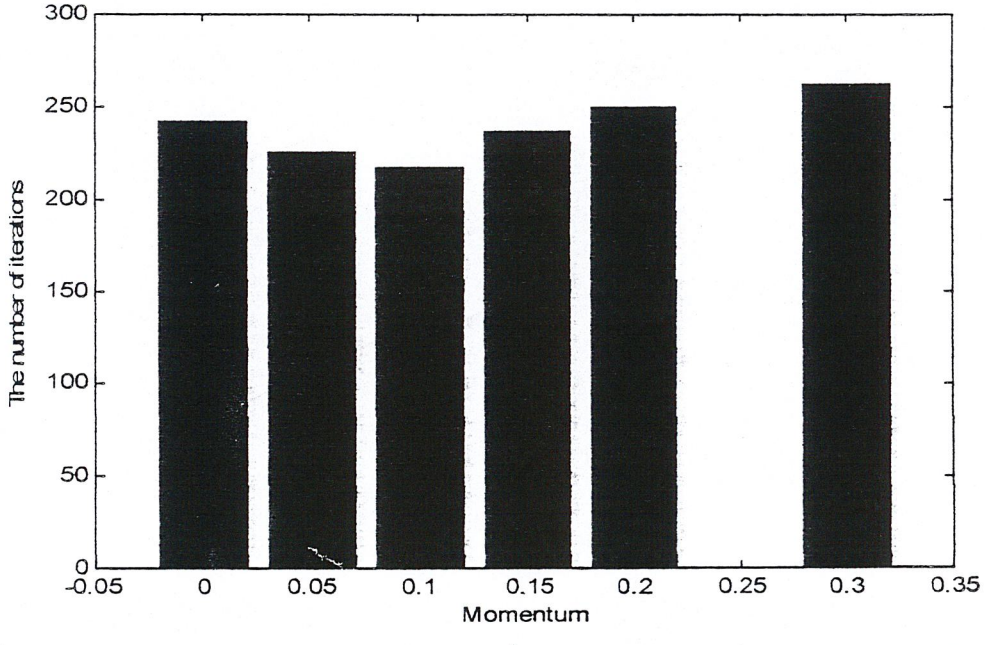


รูปที่ 5-20 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.55 และโมเมนตัม = 0.95 ของโครงข่าย E ที่ค่า Permissible error = 0.0004

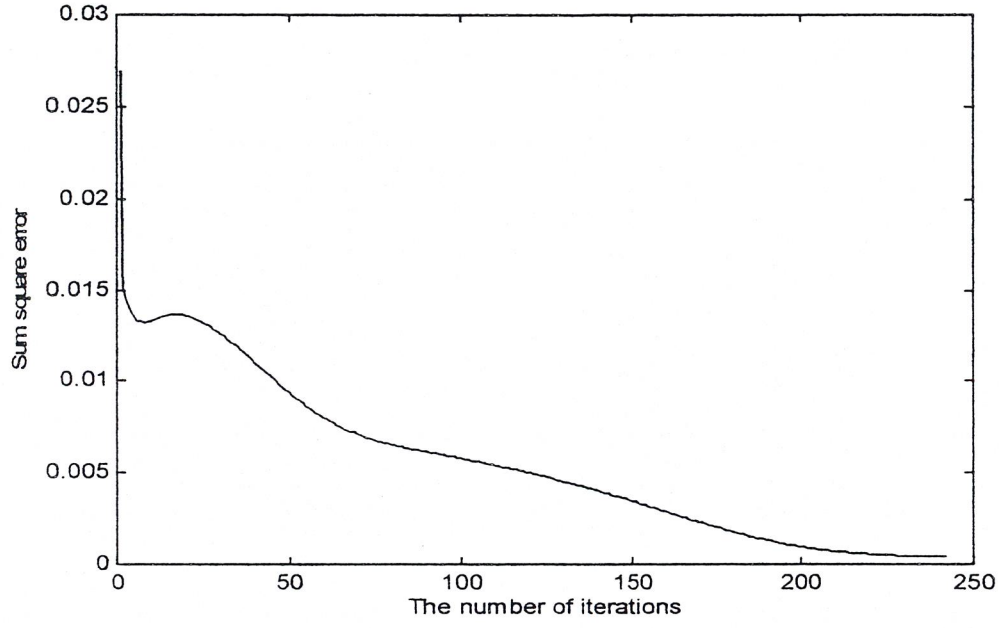
5.1.6 ผลการฝึกหัดโครงข่าย F



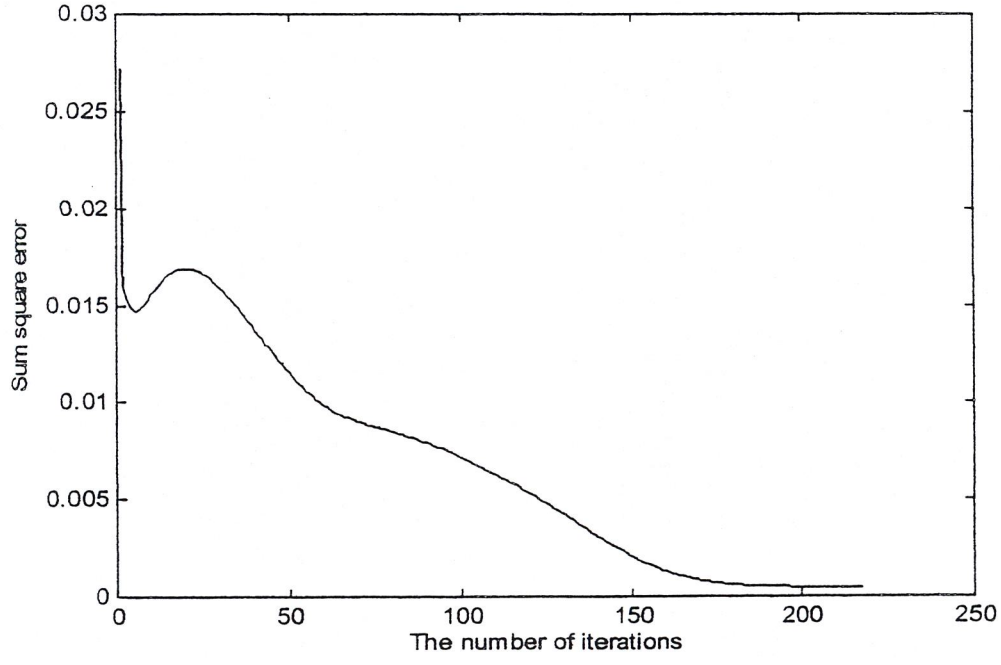
รูปที่ 5-21 แสดงความสัมพันธ์ของค่าอัตราการเรียนรู้และจำนวนรอบของการเรียนรู้ของโครงข่าย F ที่ค่าPermissible error = 0.0004



รูปที่ 5-22 แสดงความสัมพันธ์ของค่าสัมประสิทธิ์โมเมนต้มและจำนวนรอบของการเรียนรู้ของโครงข่าย F ที่ค่า Permissible error = 0.0004

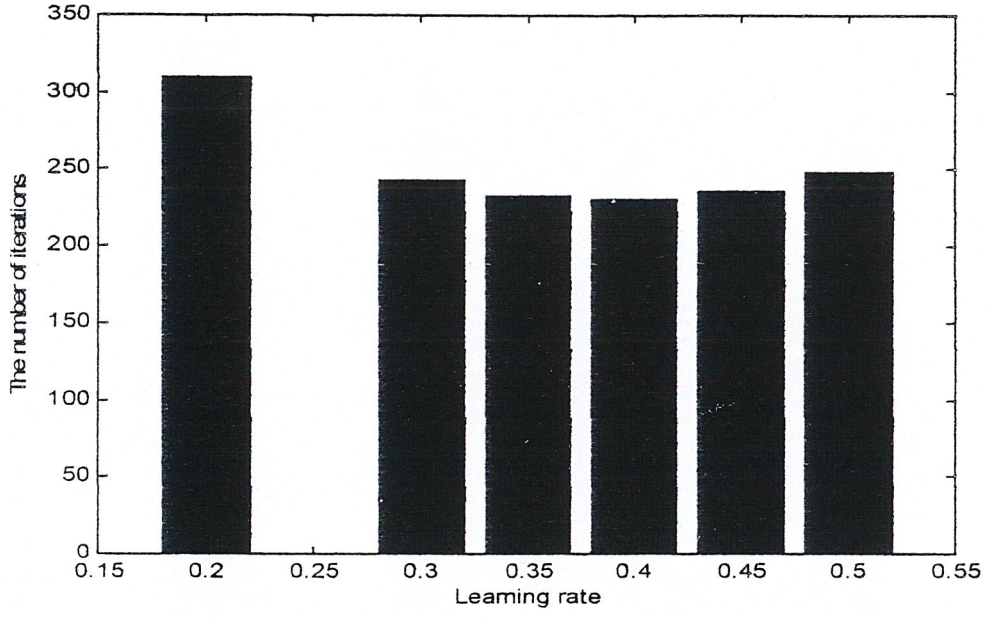


รูปที่ 5-23 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.8 และโมเมนตัม = 0 ของโครงข่าย F ที่ค่า Permissible error = 0.0004

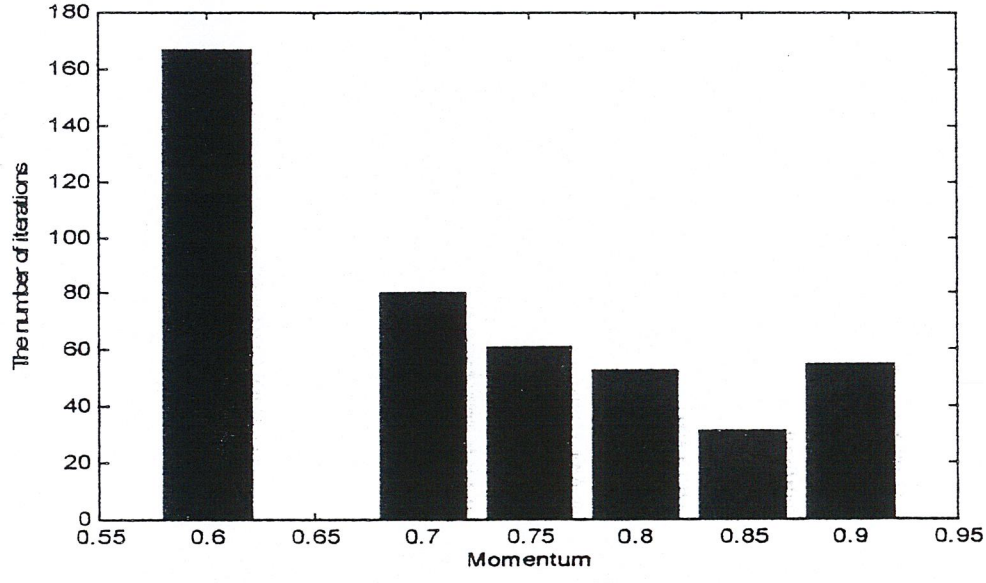


รูปที่ 5-24 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.8 และโมเมนตัม = 0.1 ของโครงข่าย F ที่ค่า Permissible error = 0.0004

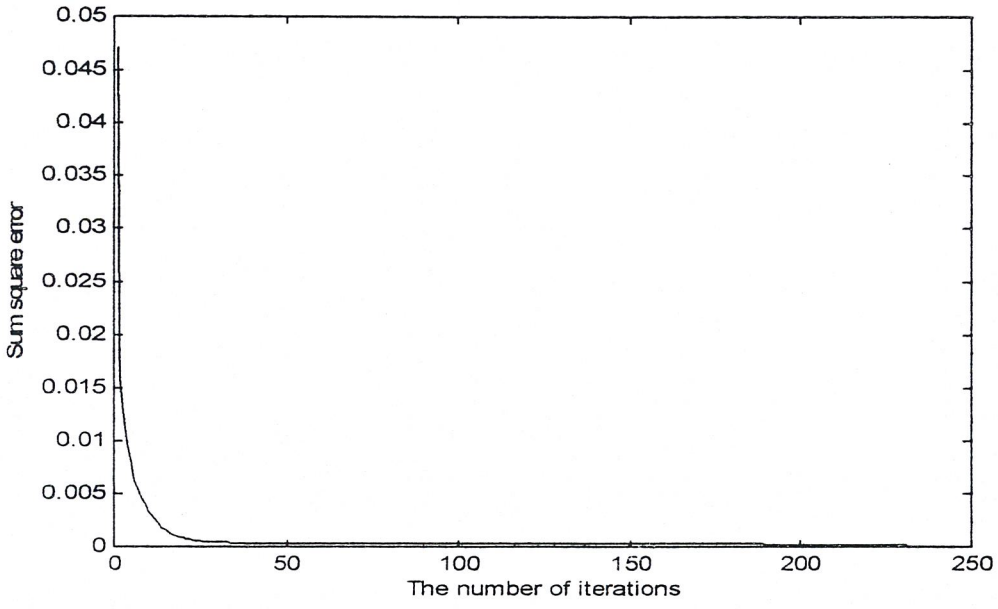
5.1.7 ผลการฝึกหัดโครงข่าย G



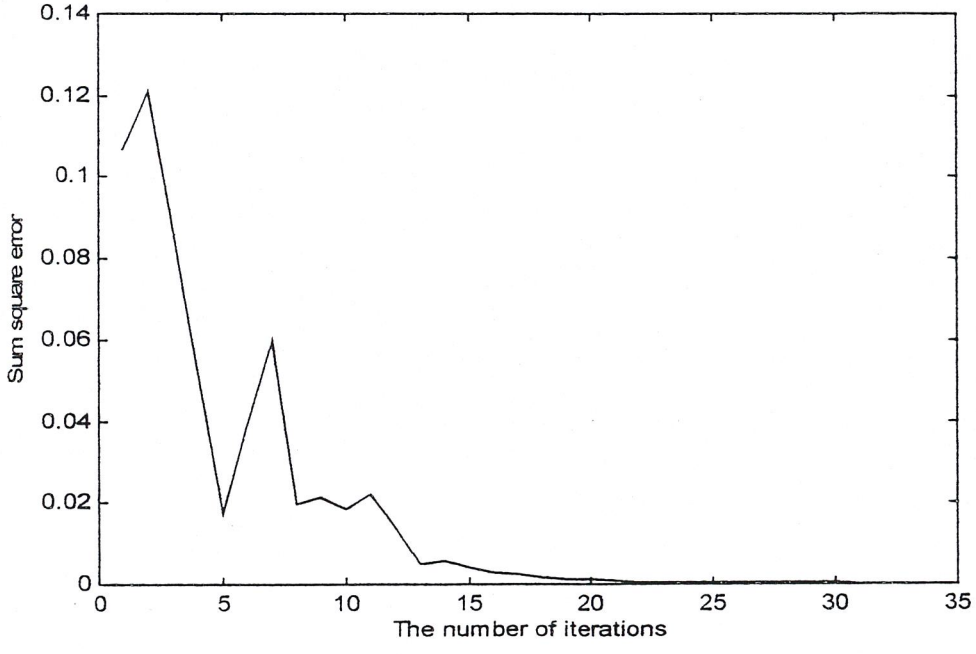
รูปที่ 5-25 แสดงความสัมพันธ์ของค่าอัตราการเรียนรู้และจำนวนรอบของการเรียนรู้ของโครงข่าย G ที่ค่า Permissible error = 0.0002



รูปที่ 5-26 แสดงความสัมพันธ์ของค่าสัมประสิทธิ์โมเมนตัมและจำนวนรอบของการเรียนรู้ของโครงข่าย G ที่ค่า Permissible error = 0.0002

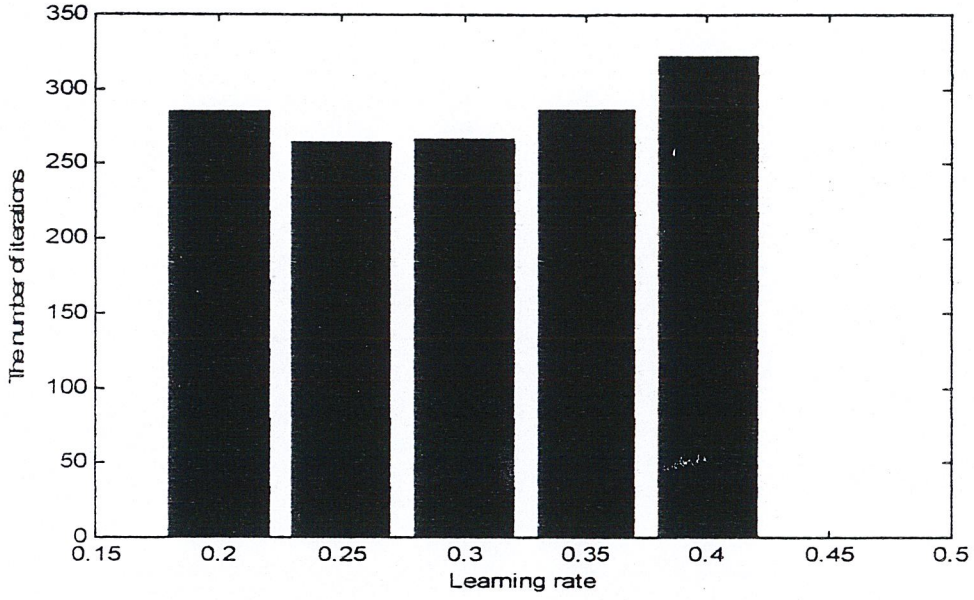


รูปที่ 5-27 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.4 และโมเมนตัม = 0 ของโครงข่าย G ที่ค่า Permissible error = 0.0002

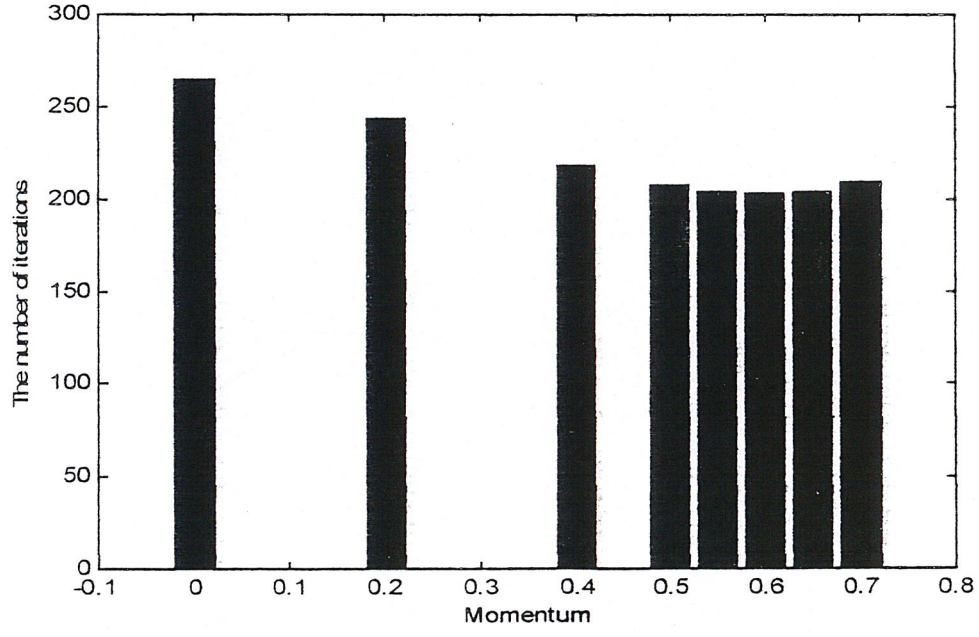


รูปที่ 5-28 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.4 และโมเมนตัม = 0.85 ของโครงข่าย G ที่ค่า Permissible error = 0.0002

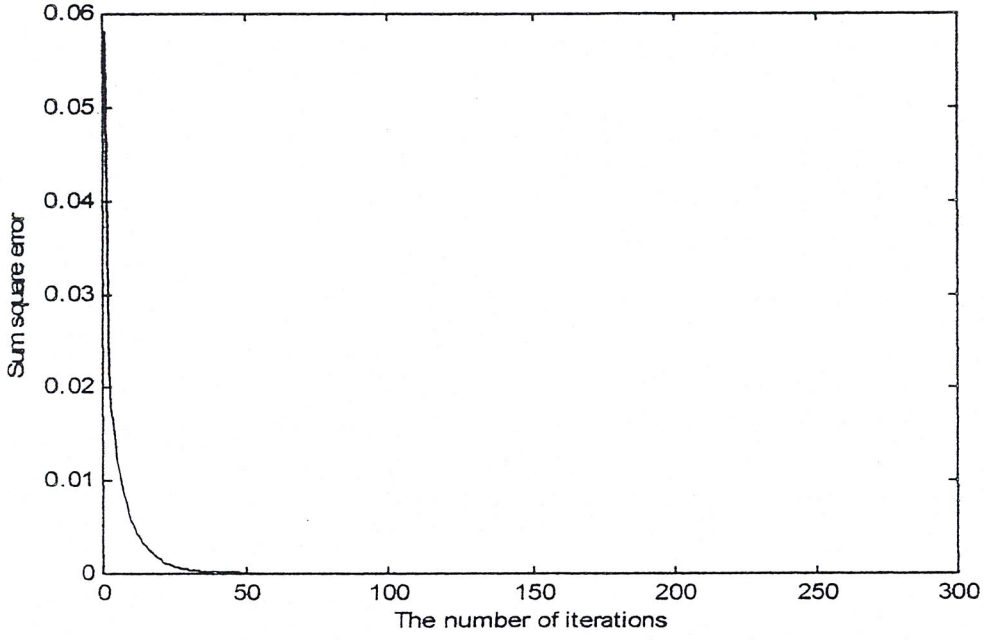
5.1.8 ผลการฝึกหัดโครงข่าย H



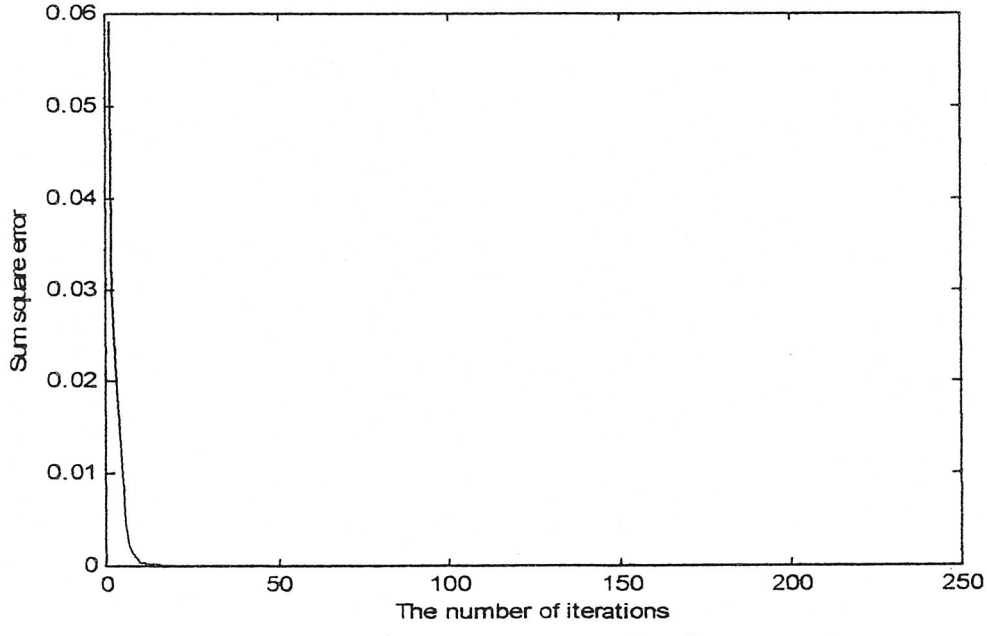
รูปที่ 5-29 แสดงความสัมพันธ์ของค่าอัตราการเรียนรู้และจำนวนรอบของการเรียนรู้ของโครงข่าย H ที่ค่า Permissible error 0.00005



รูปที่ 5-30 แสดงความสัมพันธ์ของค่าสัมประสิทธิ์โมเมนตัมและจำนวนรอบของการเรียนรู้ของโครงข่าย H ที่ค่า Permissible error = 0.00005



รูปที่ 5-31 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.25 และโมเมนตัม = 0 ของโครงข่าย H ที่ค่า Permissible error = 0.00005



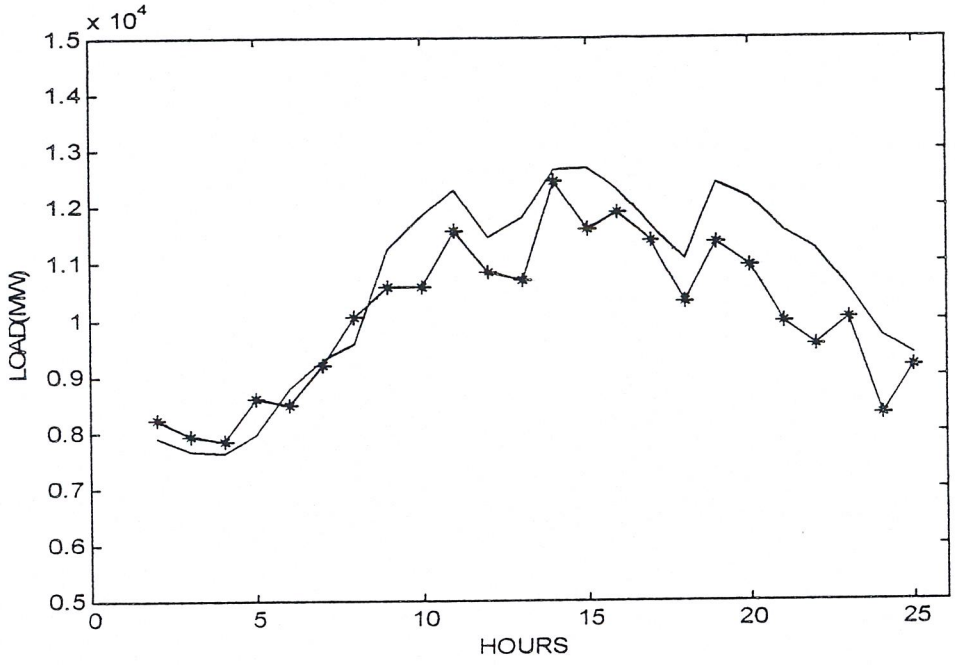
รูปที่ 5-32 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.25 และโมเมนตัม = 0.60 ของโครงข่าย H ที่ค่า Permissible error = 0.00005

5.2 ผลการทำนายโหลดในวันจันทร์โดยการใช้โครงข่ายประสาทเทียม

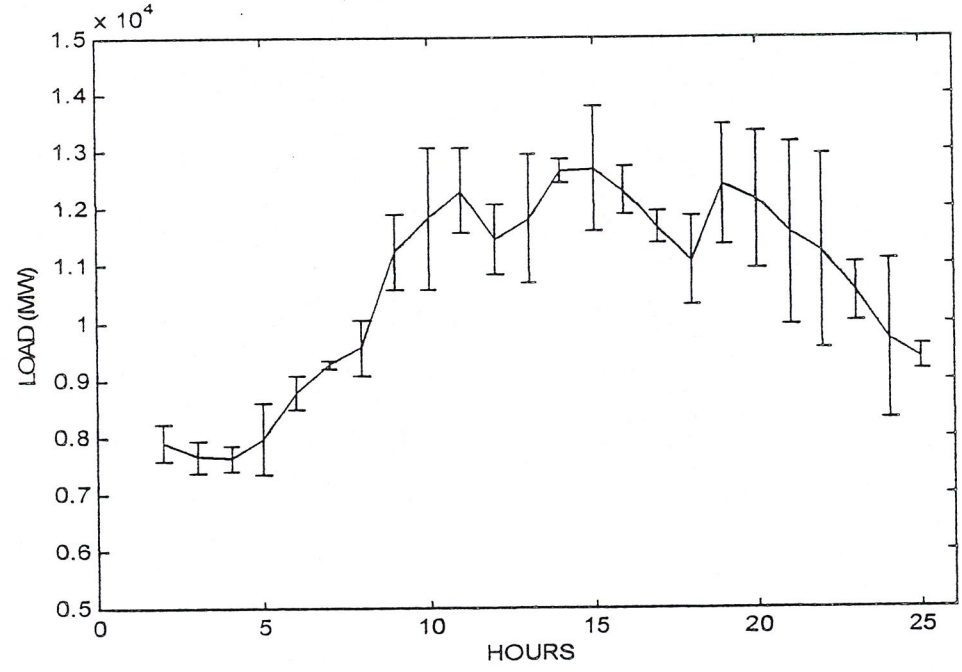
5.2.1 โครงข่ายประสาท A

ตารางที่ 5-1 แสดงผลการทำนายโหลดในวันจันทร์ของโครงข่าย Aที่ค่า Permissible error = 0.0001 โดยมีค่าความผิดพลาดเฉลี่ย 6.5618 % และมีค่าความผิดพลาดสูงสุด 15.1102 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	7900.3	8214.1721	313.8721	3.9729
03.00	7660.7	7946.7893	286.0893	3.7345
04.00	7630.6	7858.5898	227.9898	2.9878
05.00	7971.4	8597.9616	626.5616	7.8601
06.00	8782.1	8488.3145	-293.7860	-3.3453
07.00	9277.0	9206.2037	-70.7963	-0.7631
08.00	9563.6	10050.1898	486.5898	5.0879
09.00	11241.0	10578.3306	-662.6694	-5.8951
10.00	11820.8	10568.7979	-1252.0021	-10.5915
11.00	12315.8	11557.9851	-757.8149	-6.1532
12.00	11450.0	10835.3998	-614.6002	-5.3677
13.00	11812.9	10674.5481	-1138.3519	-9.6365
14.00	12655.8	12456.1888	-199.6112	-1.5772
15.00	12694.7	11588.3816	-1106.3184	-8.7148
16.00	12308.0	11882.6481	-425.3519	-3.4559
17.00	11671.0	11381.3351	-289.6649	-2.4819
18.00	11076.1	10304.5147	-771.5853	-6.9662
19.00	12427.0	11364.7750	-1062.2250	-8.5477
20.00	12146.3	10944.7915	-1201.5085	-9.8919
21.00	11561.2	9965.9693	-1595.2307	-13.7981
22.00	11245.4	9546.1990	-1699.2010	-15.1102
23.00	10539.4	10026.2755	-513.1245	-4.8686
24.00	9705.6	8305.0356	-1400.5644	-14.4305
01.00	9380.7	9170.2219	-210.4781	-2.2437



รูปที่ 5-33 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันจันทร์โดยโครงข่าย A (*)

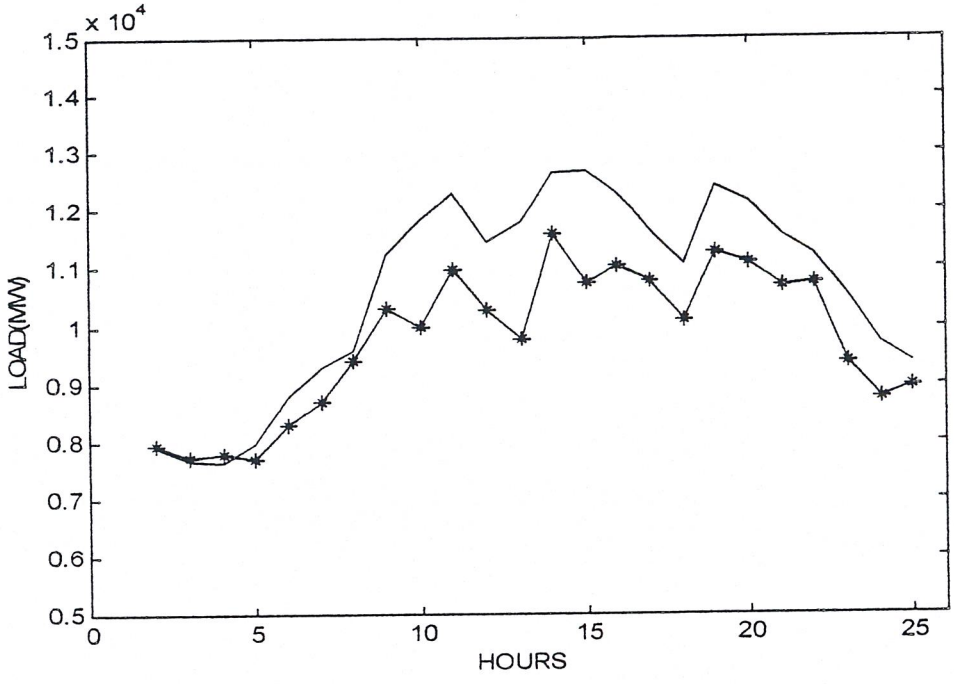


รูปที่ 5-34 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันจันทร์โดยโครงข่าย A

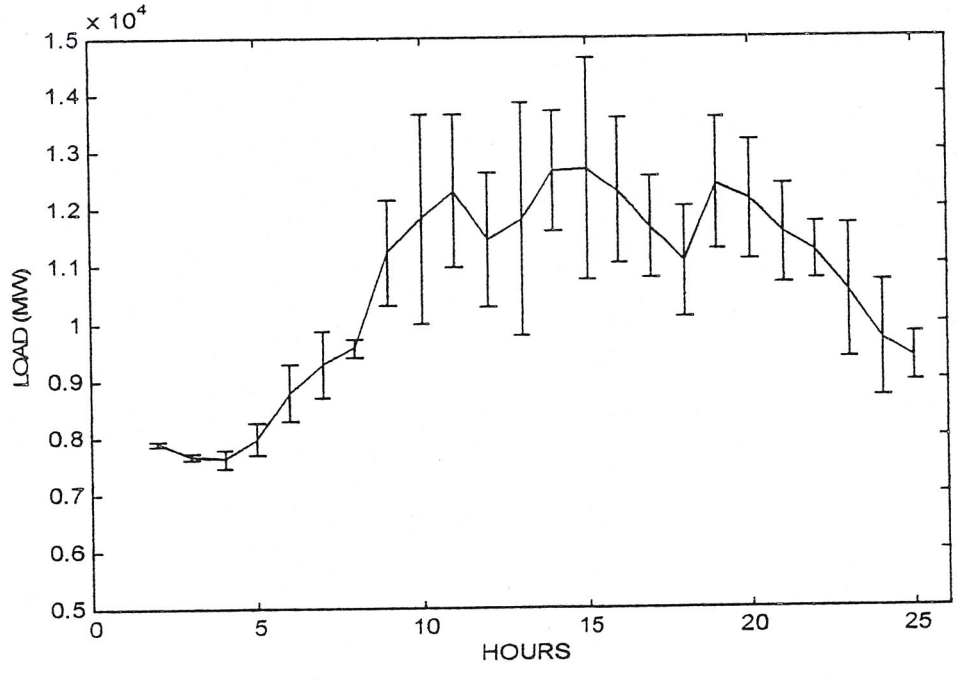
5.2.2 โครงข่ายประสาท B

ตารางที่ 5-2 แสดงผลการทำนายโหลดในวันจันทร์ของโครงข่าย B ที่ค่า Permissible error = 0.0004 โดยมีค่าความผิดพลาดเฉลี่ย 7.8707 %และมีค่าความผิดพลาดสูงสุด 17.2067 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	7900.3	7943.576	43.276	0.5478
03.00	7660.7	7724.755	64.055	0.8362
04.00	7630.6	7799.387	168.787	2.2120
05.00	7971.4	7697.902	-273.498	-3.4310
06.00	8782.1	8277.431	-504.669	-5.7466
07.00	9277.0	8697.110	-579.890	-6.2508
08.00	9563.6	9400.425	-163.175	-1.7062
09.00	11241.0	10317.150	-923.850	-8.2186
10.00	11820.8	9996.046	-1824.754	-15.4379
11.00	12315.8	10972.340	-1343.460	-10.9084
12.00	11450.0	10288.160	-1161.840	-10.1471
13.00	11812.9	9780.295	-2032.605	-17.2067
14.00	12655.8	11608.950	-1046.850	-8.2717
15.00	12694.7	10751.390	-1943.310	-15.3080
16.00	12308.0	11035.460	-1272.540	-10.3391
17.00	11671.0	10775.880	-915.120	-7.8410
18.00	11076.1	10100.720	-975.380	-8.8062
19.00	12427.0	11267.620	-1159.380	-9.3295
20.00	12146.3	11112.690	-1033.610	-8.5097
21.00	11561.2	10703.020	-858.180	-7.4229
22.00	11245.4	10750.310	-495.090	-4.4026
23.00	10539.4	9370.197	-1169.203	-11.0936
24.00	9705.6	8705.848	-999.752	-10.3008
01.00	9380.7	8947.048	-433.652	-4.6228



รูปที่ 5-35 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันจันทร์โดยโครงข่าย B (*)

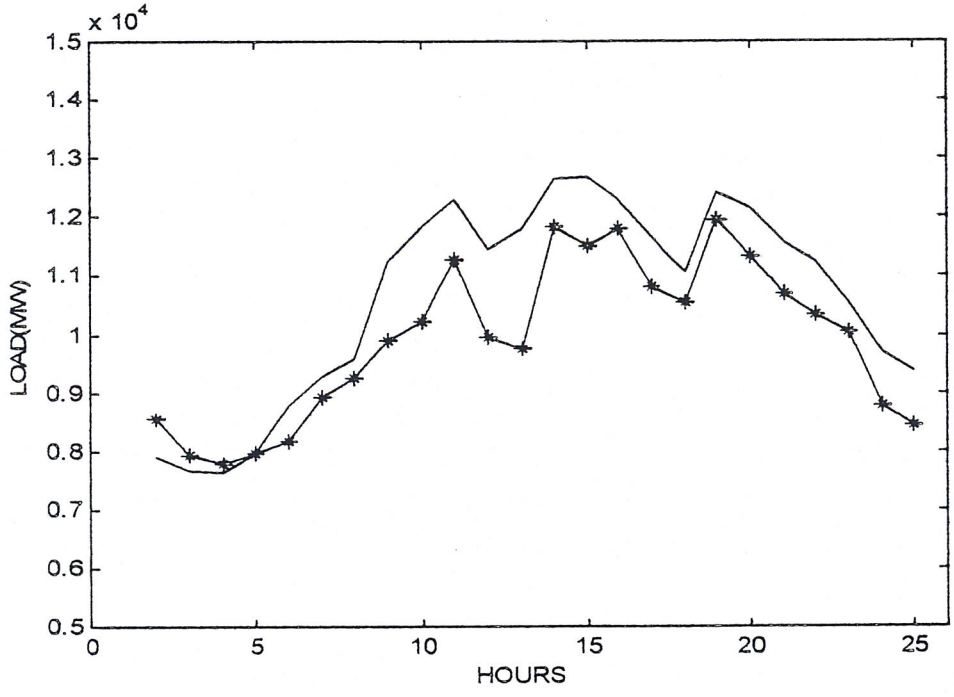


รูปที่ 5-36 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันจันทร์โดยโครงข่าย B

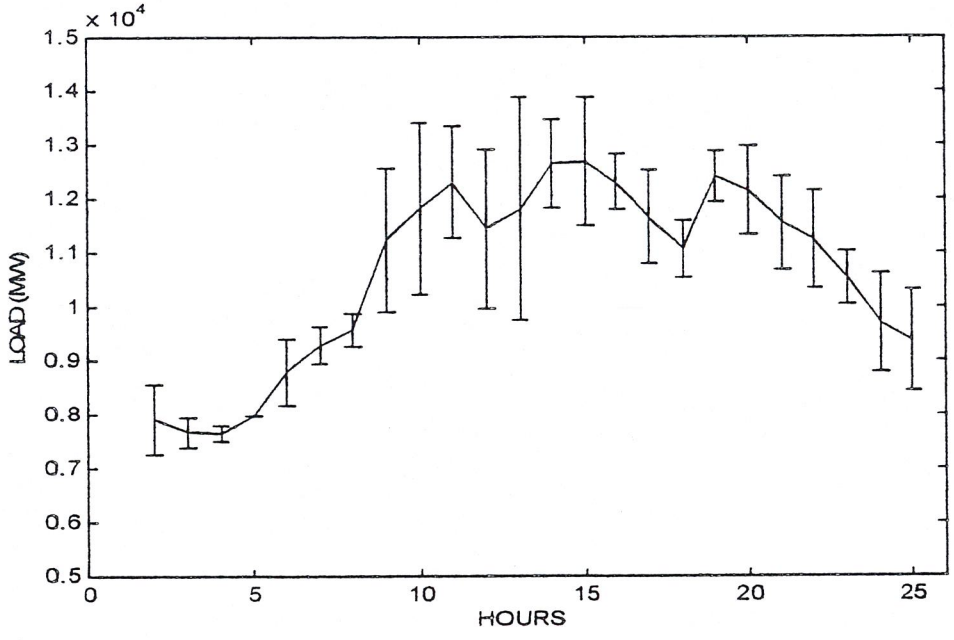
5.2.3 โครงข่ายประสาท C

ตารางที่ 5-3 แสดงผลการทำนายโหลดในวันจันทร์ของโครงข่าย C ที่ค่า Permissible error = 0.0004 โดยมีค่าความผิดพลาดเฉลี่ย 7.2838 % และมีค่าความผิดพลาดสูงสุด 17.4343 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	7900.3	8553.908	653.608	8.2732
03.00	7660.7	7940.142	279.442	3.6477
04.00	7630.6	7778.806	148.206	1.9423
05.00	7971.4	7968.660	-2.740	-0.0344
06.00	8782.1	8161.047	-621.053	-7.0718
07.00	9277.0	8927.237	-349.763	-3.7702
08.00	9563.6	9249.306	-314.294	-3.2864
09.00	11241.0	9905.464	-1335.536	-11.8809
10.00	11820.8	10227.690	-1593.110	-13.4772
11.00	12315.8	11275.080	-1040.720	-8.4503
12.00	11450.0	9967.693	-1482.307	-12.9459
13.00	11812.9	9753.408	-2059.492	-17.4343
14.00	12655.8	11837.160	-818.640	-6.4685
15.00	12694.7	11509.270	-1185.430	-9.3380
16.00	12308.0	11791.000	-517.000	-4.2005
17.00	11671.0	10808.280	-862.720	-7.3920
18.00	11076.1	10539.780	-536.320	-4.8421
19.00	12427.0	11959.100	-467.900	-3.7652
20.00	12146.3	11320.870	-825.430	-6.7957
21.00	11561.2	10695.800	-865.400	-7.4854
22.00	11245.4	10344.280	-901.120	-8.0132
23.00	10539.4	10031.710	-507.690	-4.8171
24.00	9705.6	8784.232	-921.368	-9.4932
01.00	9380.7	8443.989	-936.711	-9.9855



รูปที่ 5-37 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันจันทร์โดยโครงข่าย C (*)

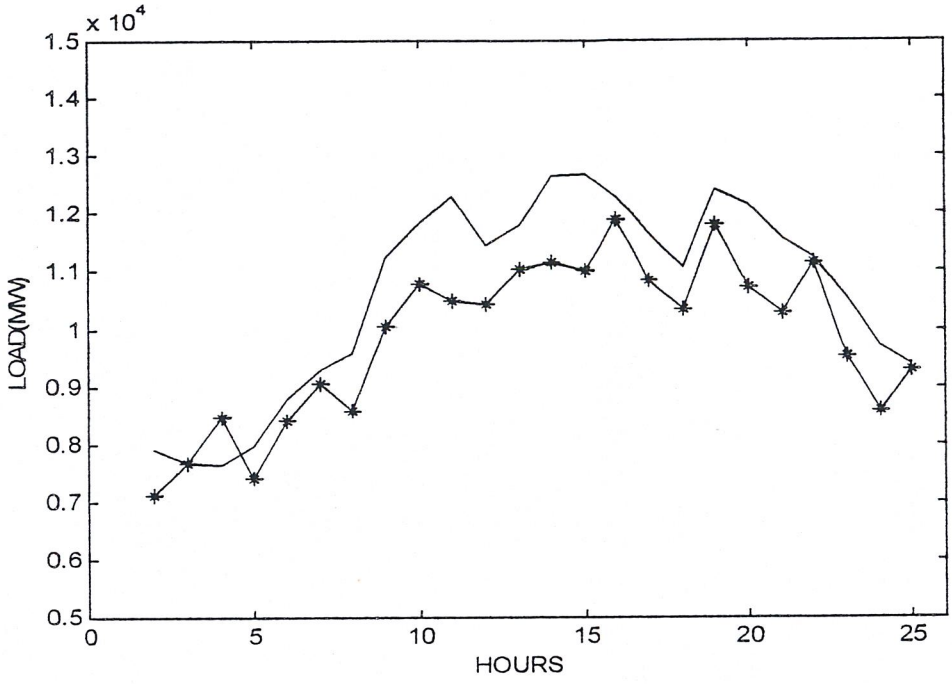


รูปที่ 5-38 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันจันทร์โดยโครงข่าย C

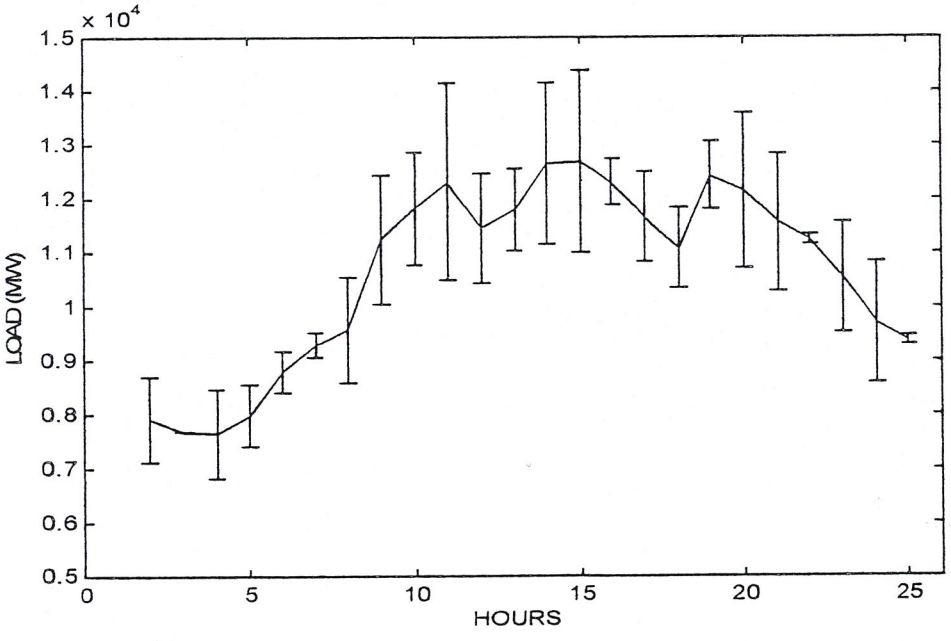
5.2.4 โครงข่ายประสาท D

ตารางที่ 5-4 แสดงผลการทำนายโหลดในวันจันทร์ของโครงข่าย D ที่ค่า Permissible error = 0.0004 โดยมีค่าความผิดพลาดเฉลี่ย 7.8606 % และมีค่าความผิดพลาดสูงสุด 14.7877 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	7900.3	7116.635	-783.665	-9.9194
03.00	7660.7	7655.263	-5.437	-0.0710
04.00	7630.6	8453.379	822.779	10.7826
05.00	7971.4	7394.272	-577.128	-7.2400
06.00	8782.1	8394.285	-387.815	-4.4160
07.00	9277.0	9040.659	-236.341	-2.5476
08.00	9563.6	8585.913	-977.687	-10.2230
09.00	11241.0	10044.930	-1196.070	-10.6402
10.00	11820.8	10789.270	-1031.530	-8.7264
11.00	12315.8	10494.580	-1821.220	-14.7877
12.00	11450.0	10414.170	-1035.830	-9.0466
13.00	11812.9	11047.500	-765.400	-6.4794
14.00	12655.8	11170.850	-1484.950	-11.7334
15.00	12694.7	11006.800	-1687.900	-13.2961
16.00	12308.0	11887.130	-420.870	-3.3996
17.00	11671.0	10826.910	-844.090	-7.2324
18.00	11076.1	10322.890	-753.210	-6.8003
19.00	12427.0	11790.030	-636.970	-5.1257
20.00	12146.3	10710.300	-1436.000	-11.8225
21.00	11561.2	10279.320	-1281.880	-11.0878
22.00	11245.4	11150.000	-95.400	-0.8483
23.00	10539.4	9503.385	-1036.015	-9.8299
24.00	9705.6	8573.954	-1131.646	-11.6597
01.00	9380.7	9292.551	-88.149	-0.9397



รูปที่ 5-39 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันจันทร์โดยโครงข่าย D (*)

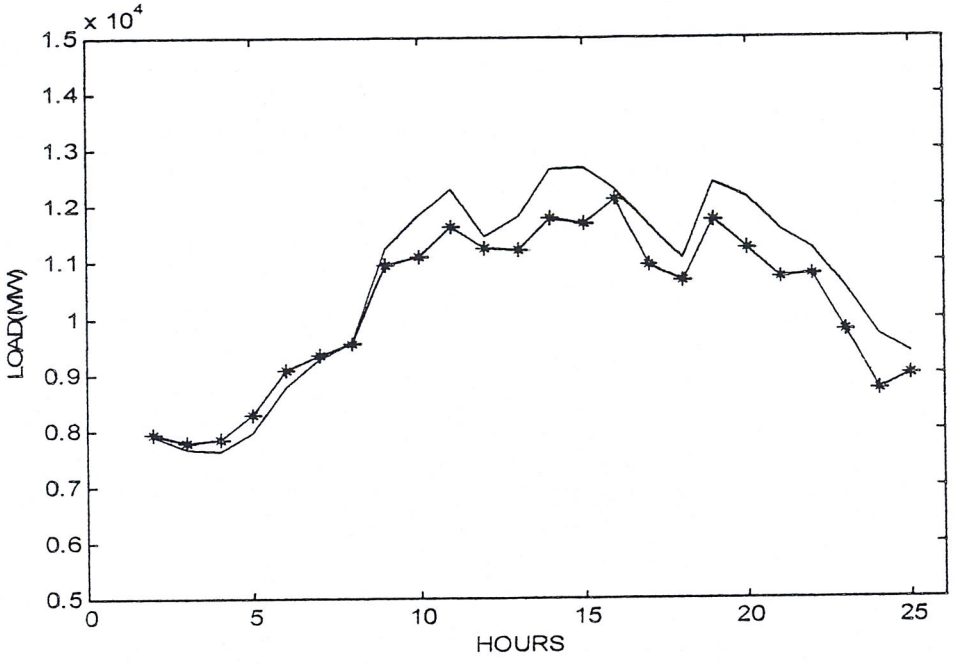


รูปที่ 5-40 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันจันทร์โดยโครงข่าย D

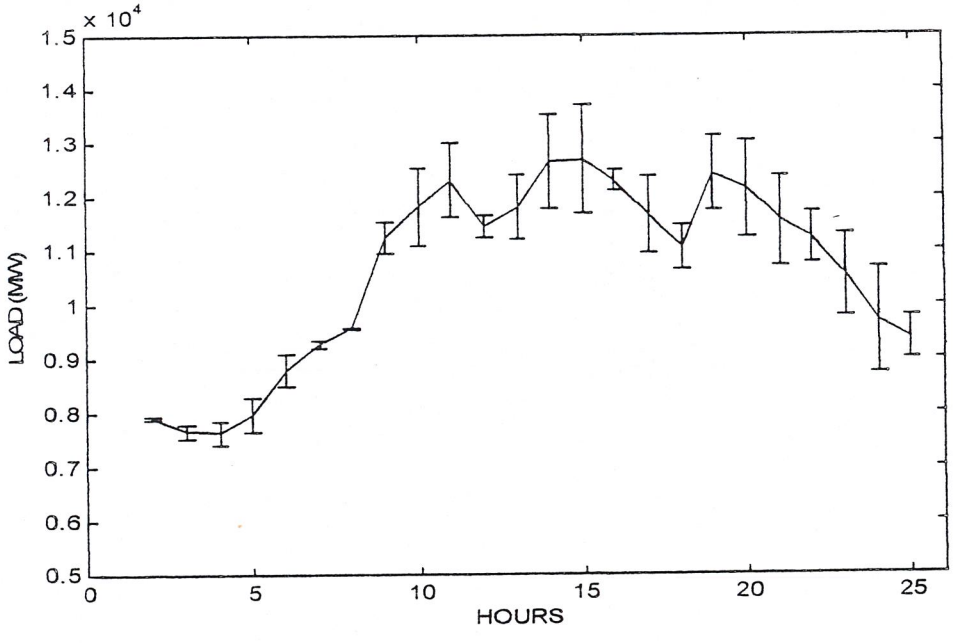
5.2.5 โครงข่ายประสาท E

ตารางที่ 5-5 แสดงผลการทำนายโหลดในวันจันทร์ของโครงข่าย E ที่ค่า Permissible error = 0.0001 โดยมีค่าความผิดพลาดเฉลี่ย 4.4331 % และมีค่าความผิดพลาดสูงสุด 9.9886 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	7900.3	7923.938	23.638	0.2992
03.00	7660.7	7790.834	130.134	1.6987
04.00	7630.6	7849.917	219.317	2.8742
05.00	7971.4	8289.895	318.495	3.9955
06.00	8782.1	9070.996	288.896	3.2896
07.00	9277.0	9350.749	73.749	0.7950
08.00	9563.6	9551.717	-11.883	-0.1243
09.00	11241.0	10942.050	-298.950	-2.6595
10.00	11820.8	11100.960	-719.840	-6.0896
11.00	12315.8	11613.650	-702.150	-5.7012
12.00	11450.0	11257.390	-192.610	-1.6822
13.00	11812.9	11211.370	-601.530	-5.0921
14.00	12655.8	11783.710	-872.090	-6.8908
15.00	12694.7	11677.510	-1017.190	-8.0127
16.00	12308.0	12113.800	-194.200	-1.5778
17.00	11671.0	10965.760	-705.240	-6.0427
18.00	11076.1	10670.810	-405.290	-3.6591
19.00	12427.0	11738.410	-688.590	-5.5411
20.00	12146.3	11245.890	-900.410	-7.4130
21.00	11561.2	10722.170	-839.030	-7.2573
22.00	11245.4	10784.560	-460.840	-4.0980
23.00	10539.4	9771.055	-768.345	-7.2902
24.00	9705.6	8736.150	-969.450	-9.9886
01.00	9380.7	8975.292	-405.408	-4.3217



รูปที่ 5-41 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันจันทร์โดยโครงข่าย E (*)

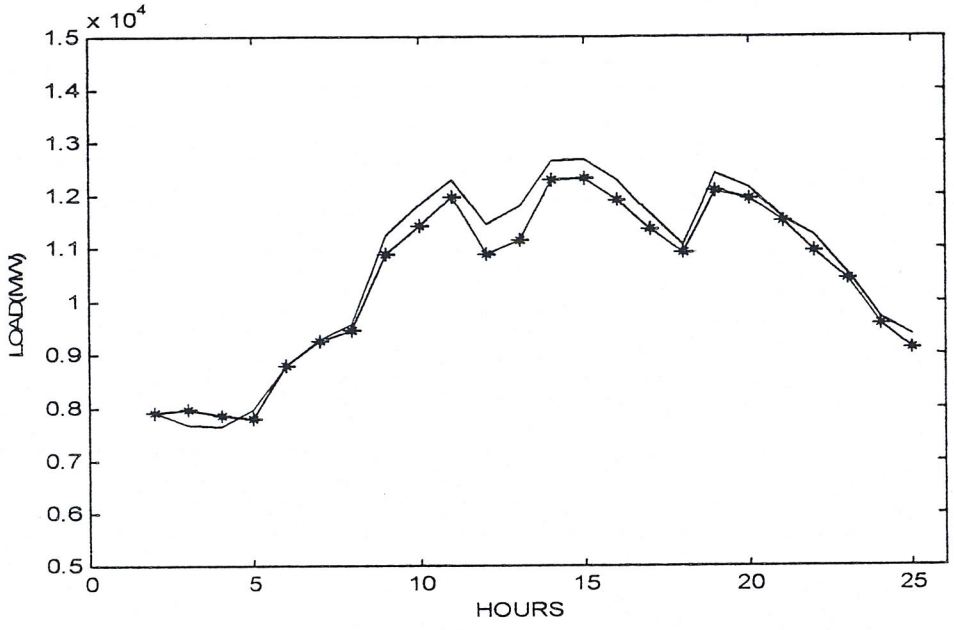


รูปที่ 5-42 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันจันทร์โดยโครงข่าย E

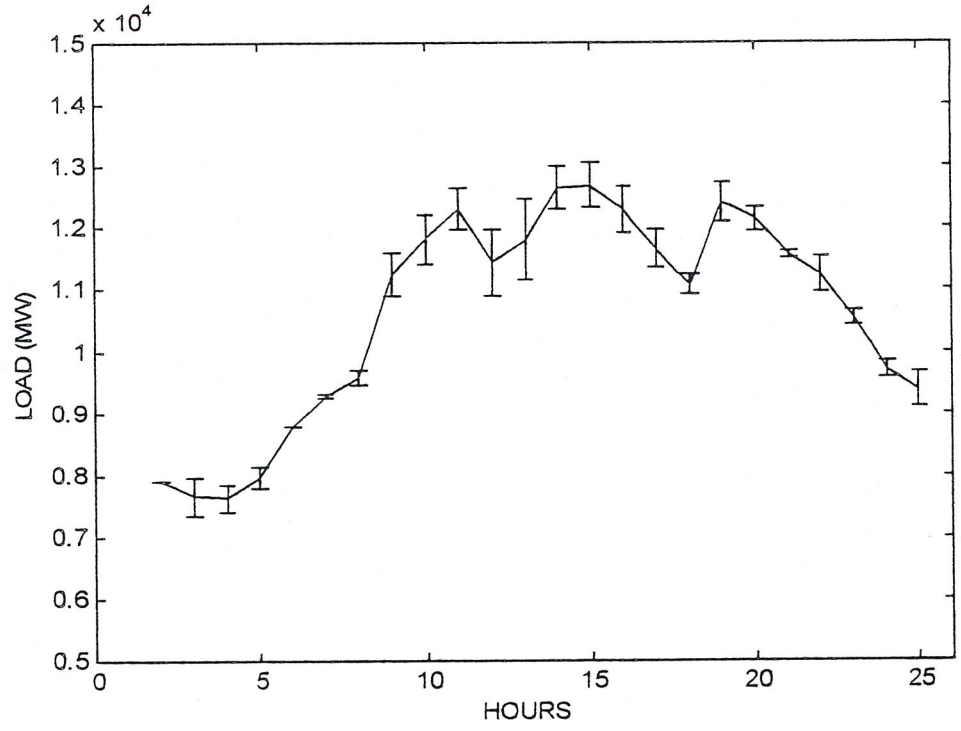
5.2.6 โครงข่ายประสาท F

ตารางที่ 5-6 แสดงผลการทำนายโหลดในวันจันทร์ของโครงข่าย F ที่ค่า Permissible error = 0.0001 โดยมีค่าความผิดพลาดเฉลี่ย 2.3335 % และมีค่าความผิดพลาดสูงสุดประมาณ 5.6603 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	7900.3	7891.070	-9.230	-0.1168
03.00	7660.7	7965.483	304.783	3.9785
04.00	7630.6	7845.784	215.184	2.8200
05.00	7971.4	7800.332	-171.068	-2.1460
06.00	8782.1	8770.558	-11.542	-0.1314
07.00	9277.0	9251.464	-25.536	-0.2753
08.00	9563.6	9445.954	-117.646	-1.2301
09.00	11241.0	10885.930	-355.070	-3.1587
10.00	11820.8	11435.110	-385.690	-3.2628
11.00	12315.8	11982.600	-333.200	-2.7055
12.00	11450.0	10907.370	-542.630	-4.7391
13.00	11812.9	11144.260	-668.640	-5.6603
14.00	12655.8	12299.250	-356.550	-2.8173
15.00	12694.7	12324.700	-370.000	-2.9146
16.00	12308.0	11927.110	-380.890	-3.0947
17.00	11671.0	11349.000	-322.000	-2.7590
18.00	11076.1	10916.340	-159.760	-1.4424
19.00	12427.0	12110.140	-316.860	-2.5498
20.00	12146.3	11952.590	-193.710	-1.5948
21.00	11561.2	11504.710	-56.490	-0.4886
22.00	11245.4	10945.640	-299.760	-2.6656
23.00	10539.4	10431.100	-108.300	-1.0276
24.00	9705.6	9563.998	-141.602	-1.4590
01.00	9380.7	9102.553	-278.147	-2.9651



รูปที่ 5-43 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันจันทร์โดยโครงข่าย F (*)

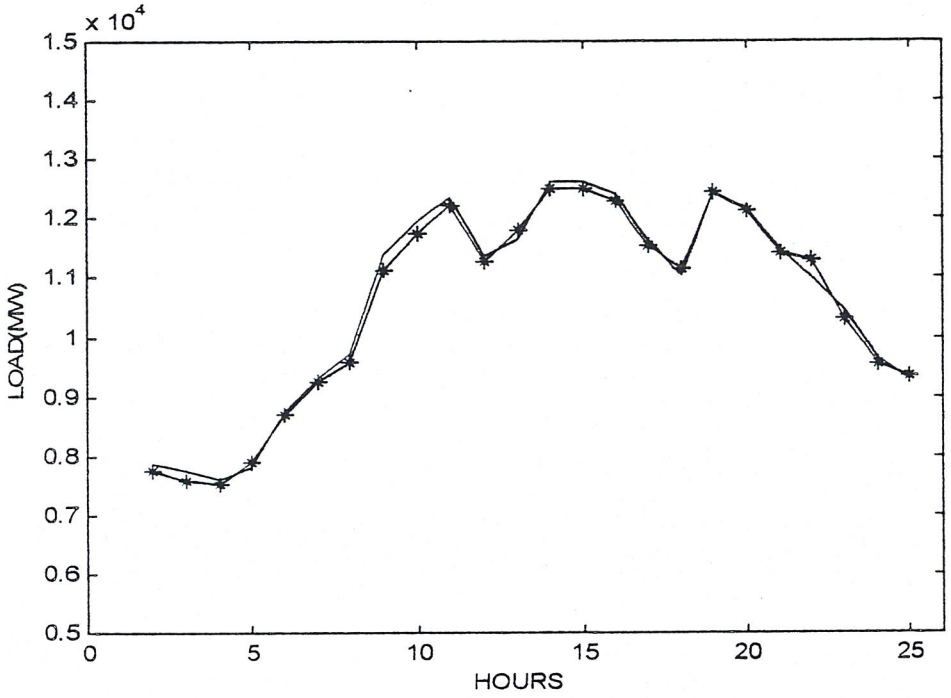


รูปที่ 5-44 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันจันทร์โดยโครงข่าย F

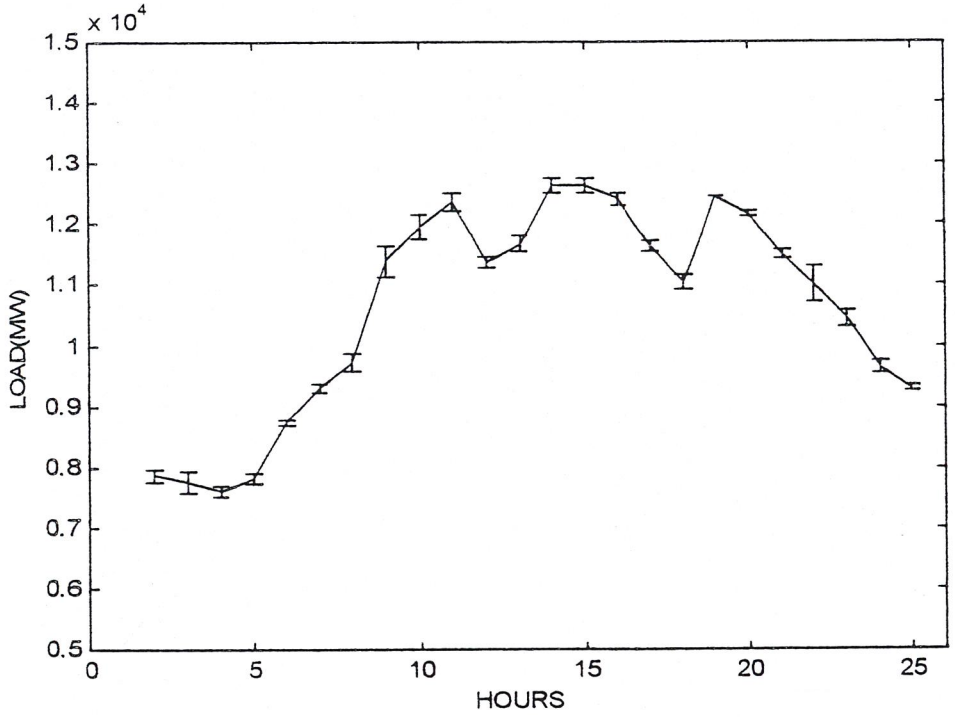
5.2.7 โครงข่ายประสาท G

ตารางที่ 5-7 แสดงผลการทำนายโหลดในวันจันทร์ของโครงข่าย G ที่ค่า Permissible error = 0.000001 โดยมีค่าความผิดพลาดเฉลี่ย 1.1005 % และมีค่าความผิดพลาดสูงสุด 2.5564 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	7866.9	7762.725	-104.175	-1.3242
03.00	7756.5	7568.112	-188.388	-2.4288
04.00	7603.2	7514.065	-89.135	-1.1723
05.00	7813.6	7900.645	87.045	1.1140
06.00	8743.7	8701.071	-42.629	-0.4875
07.00	9301.3	9239.436	-61.864	-0.6651
08.00	9722.3	9579.046	-143.254	-1.4735
09.00	11379.1	11128.610	-250.490	-2.2013
10.00	11951.3	11743.340	-207.960	-1.7401
11.00	12349.8	12200.700	-149.100	-1.2073
12.00	11351.9	11265.620	-86.280	-0.7600
13.00	11670.3	11790.910	120.610	1.0335
14.00	12618.0	12504.030	-113.970	-0.9032
15.00	12622.2	12505.190	-117.010	-0.9270
16.00	12408.9	12315.690	-93.210	-0.7512
17.00	11627.0	11539.740	-87.260	-0.7505
18.00	11034.6	11153.290	118.690	1.0756
19.00	12446.0	12447.160	1.160	0.0093
20.00	12166.8	12123.190	-43.610	-0.3585
21.00	11489.8	11424.410	-65.390	-0.5691
22.00	11012.8	11294.330	281.530	2.5564
23.00	10443.6	10300.930	-142.670	-1.3661
24.00	9638.5	9537.232	-101.268	-1.0507
01.00	9283.1	9328.309	45.209	0.4870



รูปที่ 5-45 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันจันทร์โดยโครงข่าย G (*)

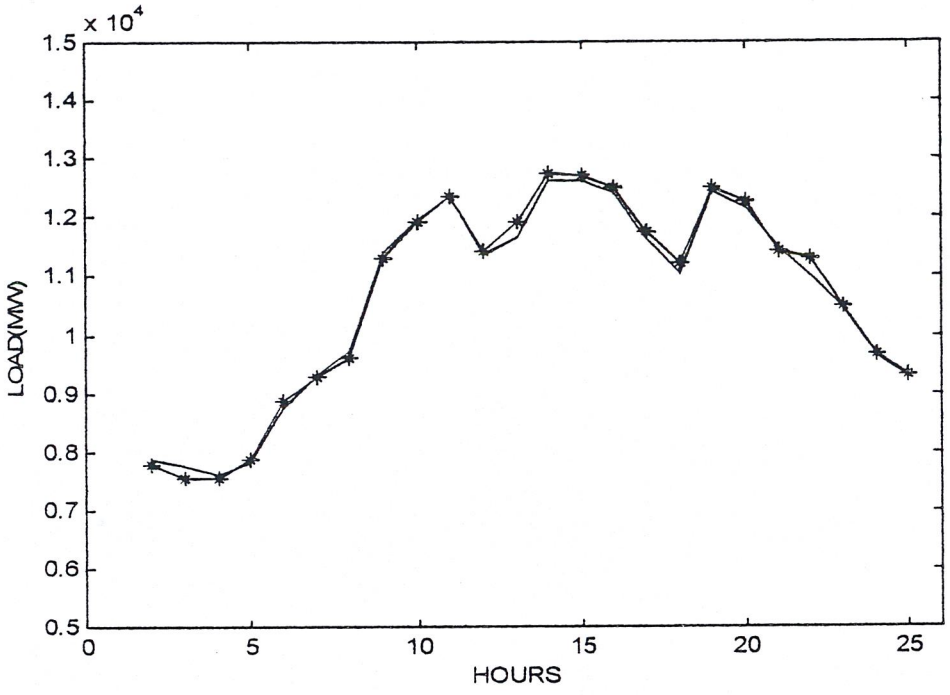


รูปที่ 5-46 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันจันทร์โดยโครงข่าย G

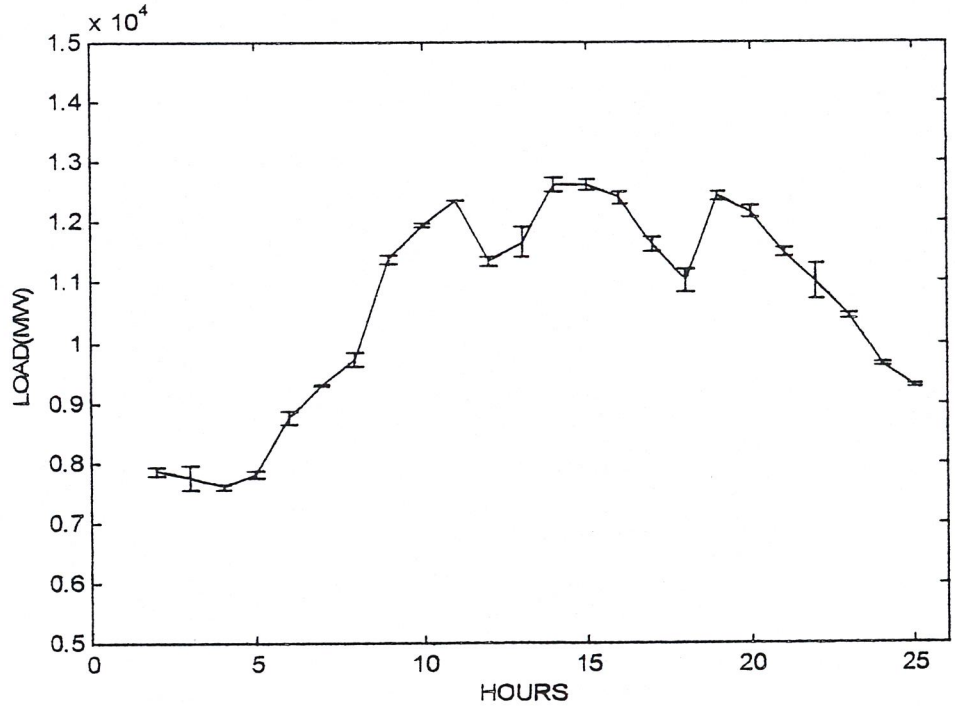
5.2.8 โครงข่ายประสาท H

ตารางที่ 5-8 แสดงผลการทำนายโหลดในวันจันทร์ของโครงข่าย H ที่ค่า Permissible error = 0.00005 โดยมีค่าความผิดพลาดเฉลี่ย 0.9220 % และมีค่าความผิดพลาดสูงสุด 2.7503 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	7866.9	7790.328	-76.572	-0.9733
03.00	7756.5	7561.943	-194.557	-2.5083
04.00	7603.2	7564.553	-38.647	-0.5083
05.00	7813.6	7867.363	53.763	0.6881
06.00	8743.7	8863.489	119.789	1.3700
07.00	9301.3	9288.777	-12.523	-0.1346
08.00	9722.3	9598.790	-123.510	-1.2704
09.00	11379.1	11301.170	-77.930	-0.6849
10.00	11951.3	11917.800	-33.500	-0.2803
11.00	12349.8	12350.110	0.310	0.0025
12.00	11351.9	11420.210	68.310	0.6017
13.00	11670.3	11913.570	243.270	2.0845
14.00	12618.0	12730.710	112.710	0.8932
15.00	12622.2	12713.120	90.920	0.7203
16.00	12408.9	12520.600	111.700	0.9002
17.00	11627.0	11751.310	124.310	1.0691
18.00	11034.6	11226.160	191.560	1.7360
19.00	12446.0	12521.180	75.180	0.6040
20.00	12166.8	12258.560	91.760	0.7542
21.00	11489.8	11421.320	-68.480	-0.5960
22.00	11012.8	11315.680	302.880	2.7503
23.00	10443.6	10486.940	43.340	0.4150
24.00	9638.5	9659.227	20.727	0.2150
01.00	9283.1	9317.130	34.030	0.3666



รูปที่ 5-47 แสดงกราฟโหลด (-) จริงเทียบกับโหลดที่ได้จากการทำนายในวันจันทร์โดยโครงข่าย H (*)



รูปที่ 5.48 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันจันทร์โดยโครงข่าย H

5.3 ผลการทำนายโหลดในวันอื่นๆโดยโครงข่ายประสาทเทียม

จากผลของการทำนายโหลดที่เกิดขึ้นภายในวันจันทร์ด้วยโครงข่ายประสาทเทียมรูปแบบต่างๆนั้น ปรากฏว่า ผลของการทำนายโหลดในวันจันทร์ด้วยโครงข่ายประสาท H นั้นให้ผลการทำนายที่ให้ค่าความผิดพลาดเฉลี่ยต่ำที่สุด คือ ประมาณ 0.9220 เปอร์เซ็นต์ ในขณะที่ผลของการทำนายโหลดในวันจันทร์ด้วยโครงข่ายประสาท G นั้นให้ค่าความผิดพลาดสูงสุดต่ำที่สุด คือ ประมาณ 2.5564 เปอร์เซ็นต์ หรือ 281.530 เมกะวัตต์ ดังนั้นการทดลองการทำนายโหลดของวันอังคารถึงวันศุกร์ เราจึงได้ใช้การทำนายด้วยโครงข่ายประสาท G และโครงข่ายประสาท H เพื่อทำการเปรียบเทียบผลของการทำนายโหลดด้วยโครงข่ายทั้งสอง

โดยรูปแบบหรือขั้นตอนของการฝึกหัดโครงข่ายในการทำนายโหลดในวันที่เหลือนั้น ยังคงใช้รูปแบบขั้นตอนเดียวกันกับการทำนายโหลดวันจันทร์เป็นเกณฑ์การทดลอง

ในการทำนายโหลดของวันศุกร์ เราได้แบ่งการทำนายโหลดออกเป็น 2 ช่วง คือ

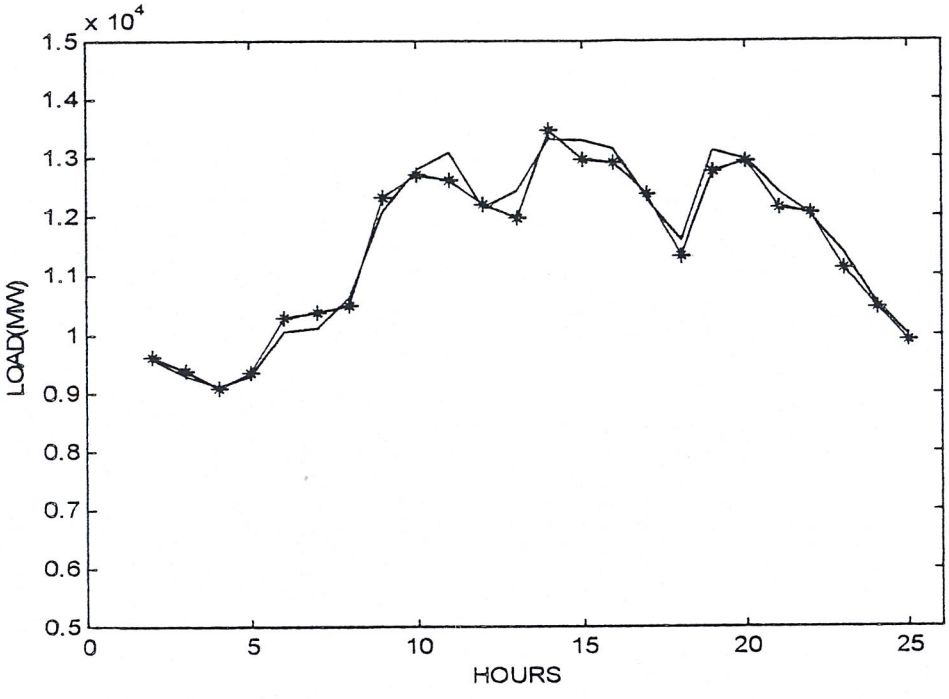
- ช่วงแรก จะทำนายโหลดตั้งแต่เวลา 02.00 น. จนกระทั่งเวลา 22.00 น.
- ช่วงที่สอง จะทำนายโหลดตั้งแต่เวลา 23.00 น. จนกระทั่งเวลา 24.00 น.

ทั้งนี้ก็เพื่อให้การทดลองทำนายโหลดในการทดลองครั้งนี้ครอบคลุมการทำนายโหลดในช่วงเวลาของวันทำงานปกติ โดยในการทดลองครั้งนี้จะไม่ได้ทำการทดลองการทำนายโหลดของวันหยุดเทศกาลและวันหยุดสุดสัปดาห์ ทั้งนี้เนื่องจากลักษณะของโหลดในวันหยุดสุดสัปดาห์และวันหยุดเทศกาลนั้นมีความแตกต่างจากลักษณะของโหลดในวันทำงานปกติ ซึ่งมีสาเหตุมาจากปัจจัยอื่นๆที่มีส่วนเกี่ยวข้องนอกเหนือจากปัจจัยทางสภาวะอากาศ

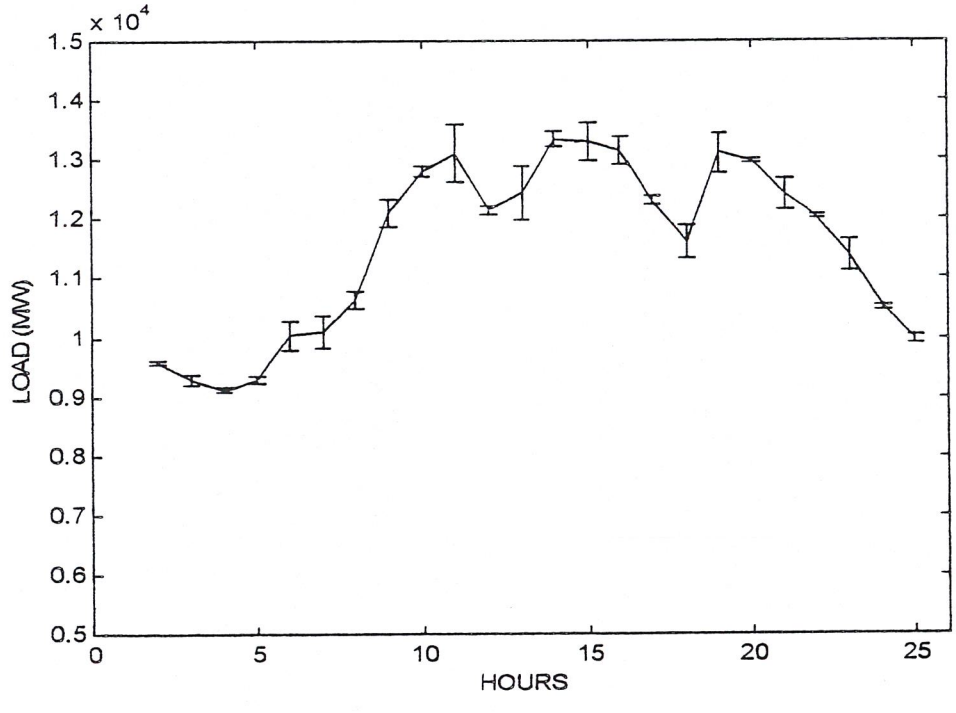
5.3.1 การทำนายโหลดวันอังคาร

ตารางที่ 5-9 แสดงผลการทำนายโหลดในวันอังคารของโครงข่าย G ที่ค่า Permissible = 0.000145 โดยมีค่าความผิดพลาดเฉลี่ย 1.5144 % และมีค่าความผิดพลาดสูงสุด 3.6952 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	9583.9	9611.569	27.669	0.2887
03.00	9276.4	9367.128	90.728	0.9781
04.00	9120.0	9062.502	-57.498	-0.6305
05.00	9283.5	9354.061	70.561	0.7599
06.00	10036.5	10283.460	246.960	2.4606
07.00	10093.4	10364.950	271.550	2.6904
08.00	10633.1	10491.600	-141.500	-1.3308
09.00	12098.1	12334.780	236.680	1.9563
10.00	12791.7	12707.780	-83.920	-0.6561
11.00	13097.9	12613.900	-484.000	-3.6952
12.00	12148.7	12216.270	67.570	0.5562
13.00	12445.2	11987.320	-457.880	-3.6792
14.00	13341.1	13461.140	120.040	0.8998
15.00	13296.5	12982.690	-313.810	-2.3601
16.00	13154.8	12910.950	-243.850	-1.8537
17.00	12316.5	12400.910	84.410	0.6853
18.00	11609.9	11333.080	-276.820	-2.3843
19.00	13114.5	12785.140	-329.360	-2.5114
20.00	12968.1	12940.910	-27.190	-0.2097
21.00	12422.3	12158.910	-263.390	-2.1203
22.00	12042.7	12065.850	23.150	0.1922
23.00	11394.5	11131.020	-263.480	-2.3123
24.00	10506.0	10465.440	-40.560	-0.3861
01.00	9981.7	9907.082	-74.618	-0.7475



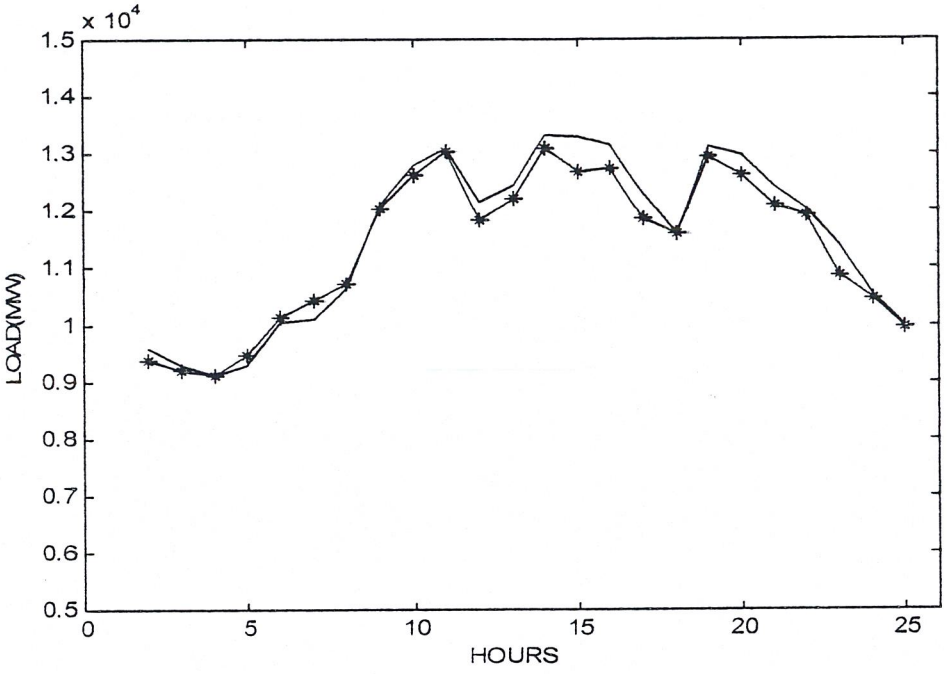
รูปที่ 5-49 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันอังคาร โดยโครงข่าย G (*)



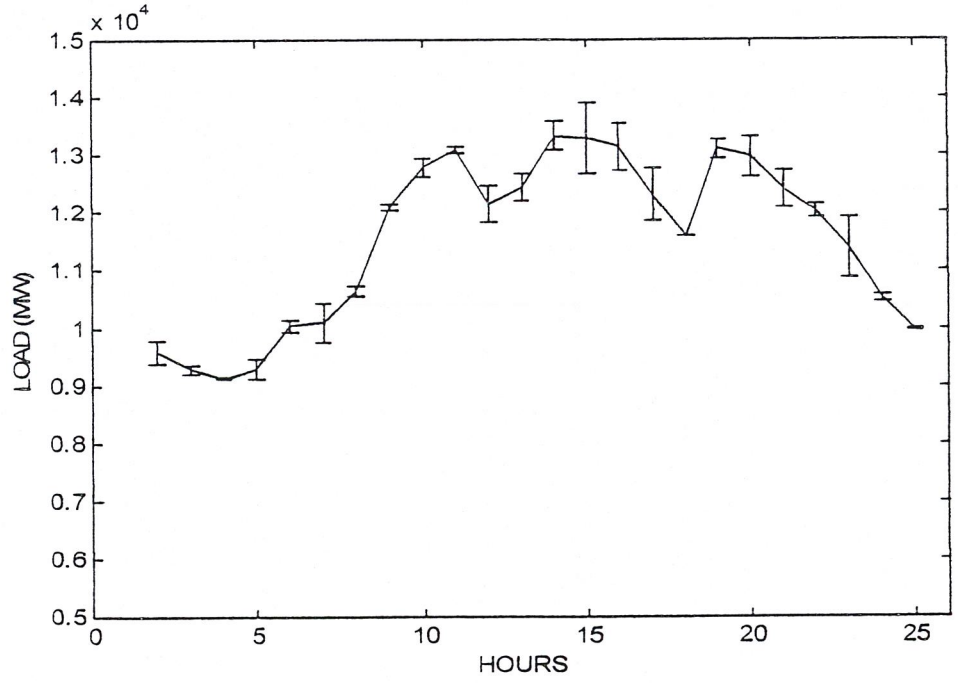
รูปที่ 5.50 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันอังคาร โดยโครงข่าย G

ตารางที่ 5-10 แสดงผลการทำนายโหลดในวันอังคารของโครงข่าย H ที่ค่า Permissible = 0.000135 โดยมีค่าความผิดพลาดเฉลี่ย 1.7944 % และมีค่าความผิดพลาดสูงสุด 4.7194 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	9583.9	9377.180	-206.720	-2.1570
03.00	9276.4	9204.109	-72.291	-0.7793
04.00	9120.0	9100.716	-19.284	-0.2114
05.00	9283.5	9469.600	186.100	2.0046
06.00	10036.5	10134.390	97.890	0.9754
07.00	10093.4	10432.650	339.250	3.3611
08.00	10633.1	10713.570	80.470	0.7568
09.00	12098.1	12034.160	-63.940	-0.5285
10.00	12791.7	12635.620	-156.080	-1.2202
11.00	13097.9	13036.750	-61.150	-0.4669
12.00	12148.7	11826.720	-321.980	-2.6503
13.00	12445.2	12211.380	-233.820	-1.8788
14.00	13341.1	13088.290	-252.810	-1.8950
15.00	13296.5	12688.340	-608.160	-4.5738
16.00	13154.8	12753.700	-401.100	-3.0491
17.00	12316.5	11856.320	-460.180	-3.7363
18.00	11609.9	11610.810	0.910	0.0078
19.00	13114.5	12958.620	-155.880	-1.1886
20.00	12968.1	12615.540	-352.560	-2.7187
21.00	12422.3	12097.930	-324.370	-2.6112
22.00	12042.7	11934.440	-108.260	-0.8990
23.00	11394.5	10856.750	-537.750	-4.7194
24.00	10506.0	10453.740	-52.260	-0.4974
01.00	9981.7	9963.913	-17.787	-0.1782



รูปที่ 5-51 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันอังคาร โดยโครงข่าย H (*)

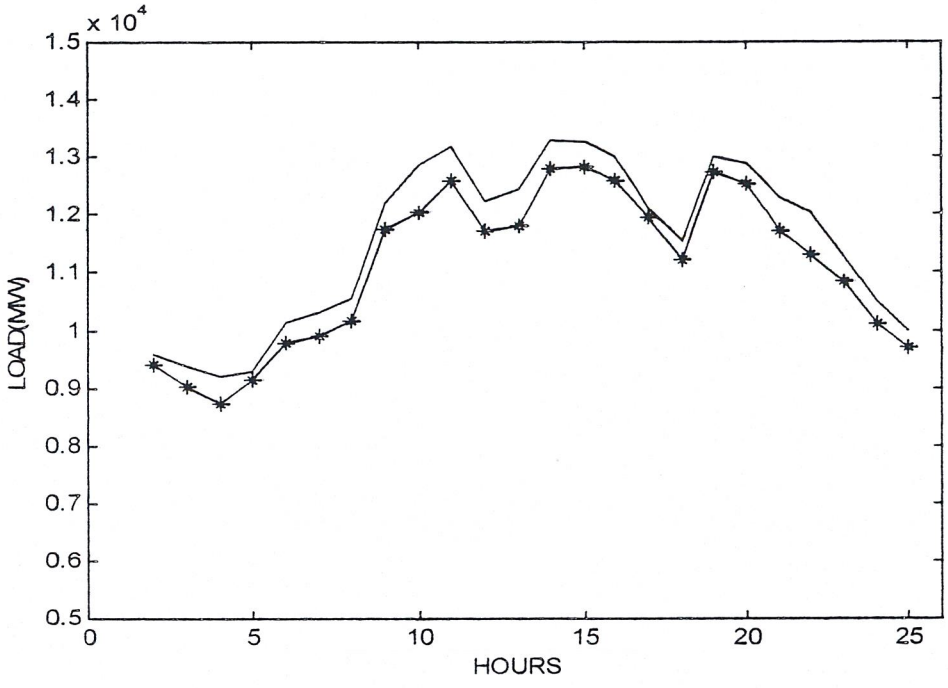


รูปที่ 5.52 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันอังคาร โดยโครงข่าย H

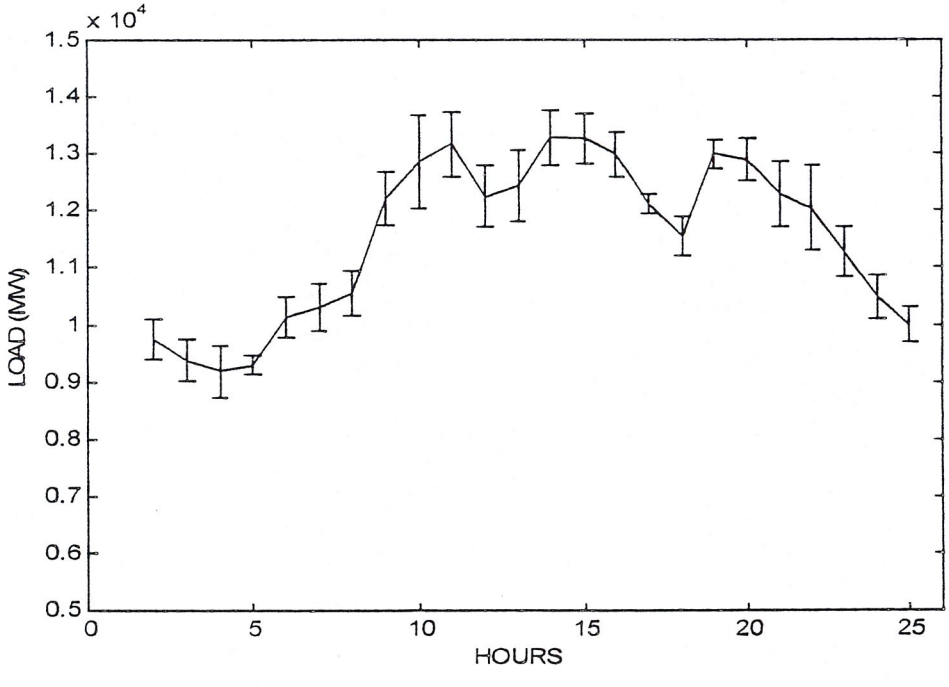
5.3.2 การทำนายโหลดวันพุธ

ตารางที่ 5-11 แสดงผลการทำนายโหลดในวันพุธของโครงข่าย G ที่ค่า Permissible = 0.000005 โดยมีค่าความผิดพลาดเฉลี่ย 3.6932 % และมีค่าความผิดพลาดสูงสุด 6.3926 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	9574.7	9392.355	-182.345	-1.9044
03.00	9380.4	9007.031	-373.369	-3.9803
04.00	9184.8	8738.279	-446.521	-4.8615
05.00	9278.7	9123.331	-155.369	-1.6745
06.00	10127.5	9770.184	-357.316	-3.5282
07.00	10314.8	9907.098	-407.702	-3.9526
08.00	10544.4	10149.750	-394.650	-3.7427
09.00	12205.6	11733.380	-472.220	-3.8689
10.00	12848.2	12026.870	-821.330	-6.3926
11.00	13167.4	12583.740	-583.660	-4.4326
12.00	12253.7	11705.380	-548.320	-4.4747
13.00	12436.4	11813.760	-622.640	-5.0066
14.00	13289.7	12810.390	-479.310	-3.6066
15.00	13273.4	12828.370	-445.030	-3.3528
16.00	12991.5	12586.980	-404.520	-3.1137
17.00	12129.9	11955.150	-174.750	-1.4407
18.00	11553.7	11210.000	-343.700	-2.9748
19.00	12993.0	12732.430	-260.570	-2.0055
20.00	12896.6	12534.890	-361.710	-2.8047
21.00	12288.2	11729.580	-558.620	-4.5460
22.00	12040.6	11292.310	-748.290	-6.2147
23.00	11278.4	10830.750	-447.650	-3.9691
24.00	10476.0	10092.200	-383.800	-3.6636
01.00	9989.9	9677.755	-312.145	-3.1246



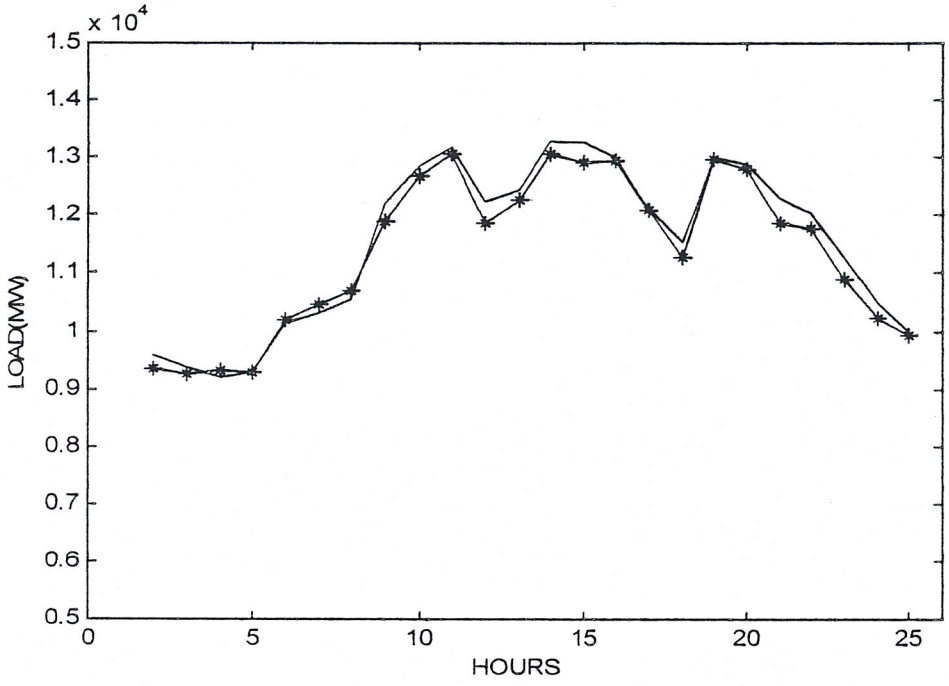
รูปที่ 5-53 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันพุธโดยโครงข่าย G (*)



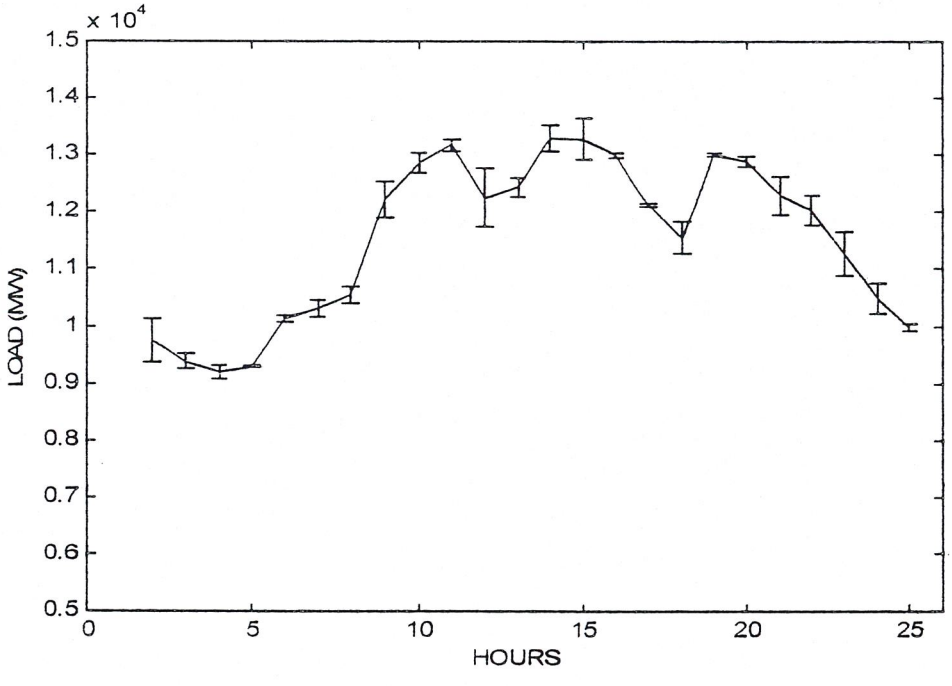
รูปที่ 5.54 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันพุธโดยโครงข่าย G

ตารางที่ 5-12 แสดงผลการทำนายโหลดในวันพุธของโครงข่าย H ที่ค่า Permissible = 0.0001 โดยมีค่าความผิดพลาดเฉลี่ย 1.5732 % และมีค่าความผิดพลาดสูงสุด 3.3172 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	9574.7	9347.232	-227.468	-2.3755
03.00	9380.4	9250.320	-130.080	-1.3867
04.00	9184.8	9299.518	114.718	1.2490
05.00	9278.7	9266.928	-11.772	-0.1269
06.00	10127.5	10177.780	50.280	0.4965
07.00	10314.8	10468.890	154.090	1.4939
08.00	10544.4	10698.690	154.290	1.4632
09.00	12205.6	11882.040	-323.560	-2.6509
10.00	12848.2	12670.750	-177.450	-1.3811
11.00	13167.4	13069.780	-97.620	-0.7414
12.00	12253.7	11847.810	-405.890	-3.3124
13.00	12436.4	12285.740	-150.660	-1.2114
14.00	13289.7	13056.110	-233.590	-1.7577
15.00	13273.4	12905.670	-367.730	-2.7704
16.00	12991.5	12952.730	-38.770	-0.2984
17.00	12129.9	12098.960	-30.940	-0.2551
18.00	11553.7	11268.160	-285.540	-2.4714
19.00	12993.0	12963.480	-29.520	-0.2272
20.00	12896.6	12803.820	-92.780	-0.7194
21.00	12288.2	11953.390	-334.810	-2.7246
22.00	12040.6	11770.750	-269.850	-2.2412
23.00	11278.4	10904.270	-374.130	-3.3172
24.00	10476.0	10221.900	-254.100	-2.4255
01.00	9989.9	9923.916	-65.984	-0.6605



รูปที่ 5-55 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันพุธโดยโครงข่าย H (*)

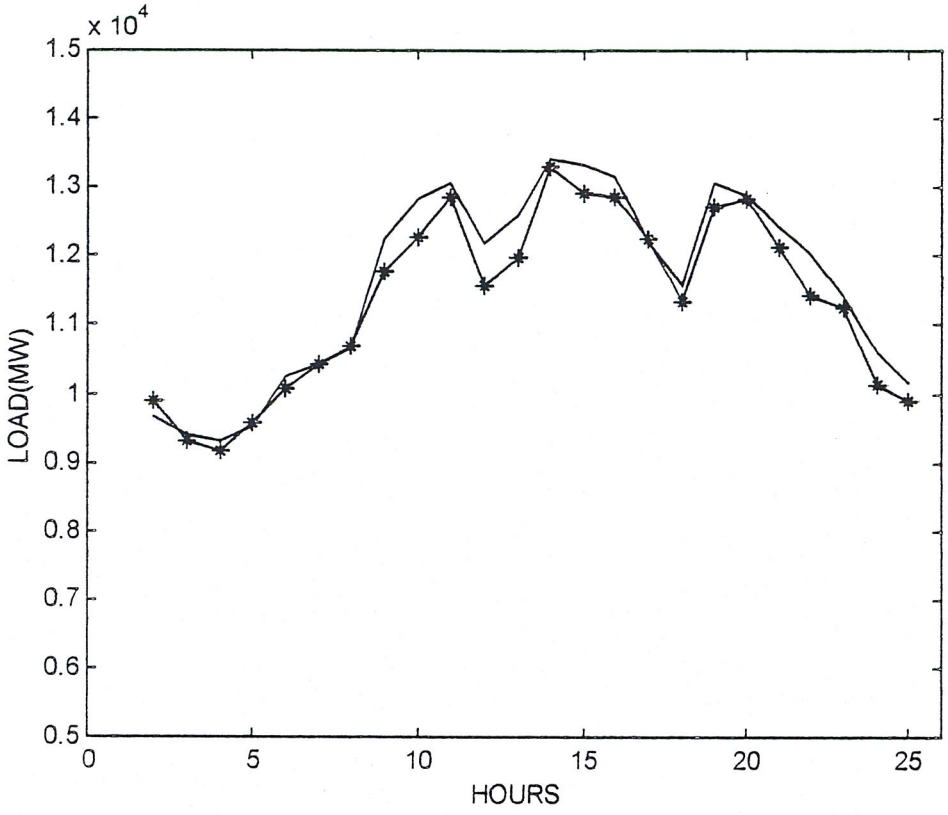


รูปที่ 5.56 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันพุธโดยโครงข่าย H

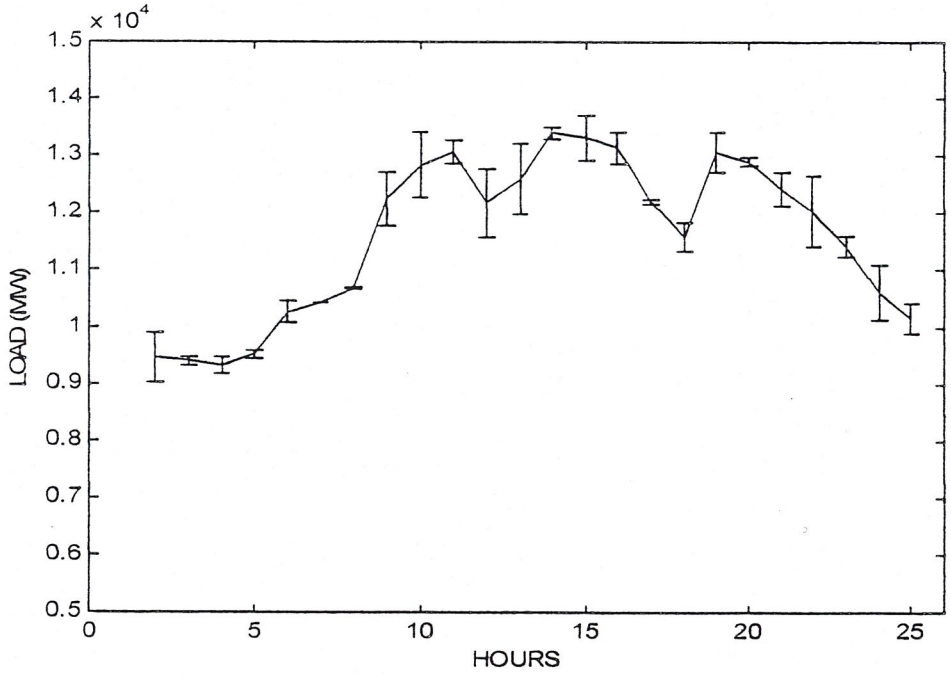
5.3.3 การทำนายโหลดวันพฤหัสบดี

ตารางที่ 5-13 แสดงผลการทำนายโหลดในวันพฤหัสของโครงข่าย G ที่ค่า Permissible = 0.00015 โดยมีค่าความผิดพลาดเฉลี่ย 2.3042 % และมีค่าความผิดพลาดสูงสุด 5.1336 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	9648.1	9910.650	262.550	2.7213
03.00	9385.8	9317.252	-68.548	-0.7303
04.00	9308.9	9174.488	-134.412	-1.4439
05.00	9504.7	9567.477	62.777	0.6605
06.00	10261.0	10074.320	-186.680	-1.8193
07.00	10438.6	10438.140	-0.460	-0.0044
08.00	10670.7	10683.940	13.240	0.1241
09.00	12240.1	11768.340	-471.760	-3.8542
10.00	12834.9	12260.390	-574.510	-4.4762
11.00	13076.5	12868.740	-207.760	-1.5888
12.00	12171.4	11569.270	-602.130	-4.9471
13.00	12594.4	11967.960	-626.440	-4.9740
14.00	13402.9	13297.120	-105.780	-0.7892
15.00	13321.6	12919.050	-402.550	-3.0218
16.00	13139.8	12856.720	-283.080	-2.1544
17.00	12204.4	12251.190	46.790	0.3834
18.00	11578.3	11320.940	-257.360	-2.2228
19.00	13062.1	12714.980	-347.120	-2.6575
20.00	12901.6	12827.070	-74.530	-0.5777
21.00	12411.5	12116.610	-294.890	-2.3759
22.00	12044.8	11426.470	-618.330	-5.1336
23.00	11421.4	11242.650	-178.750	-1.5650
24.00	10605.0	10121.200	-483.800	-4.5620
01.00	10159.1	9903.821	-255.279	-2.5128



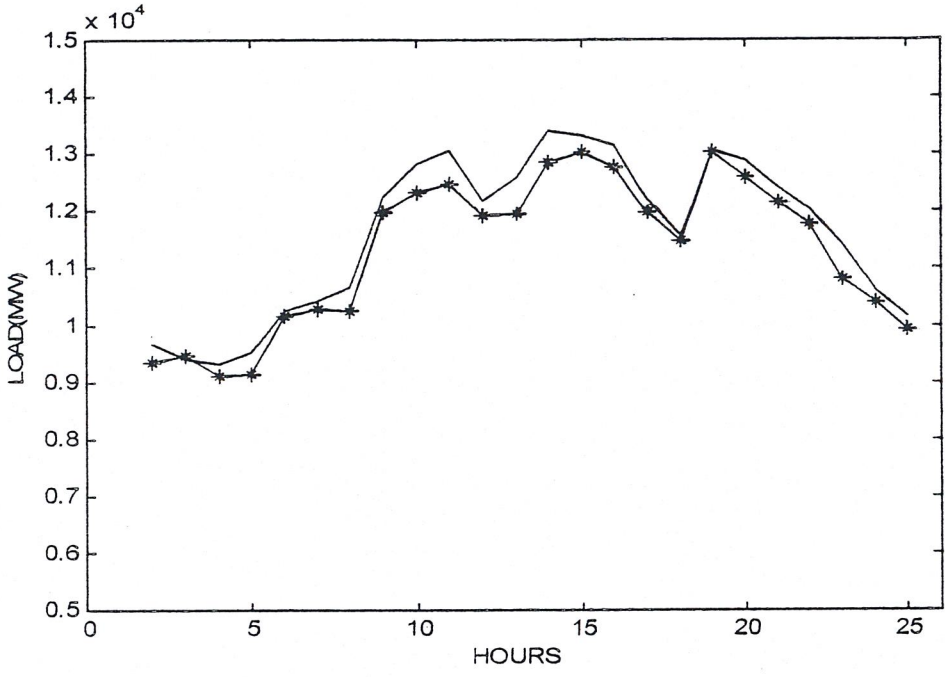
รูปที่ 5-57 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันพฤหัสโดยโครงข่าย G (*)



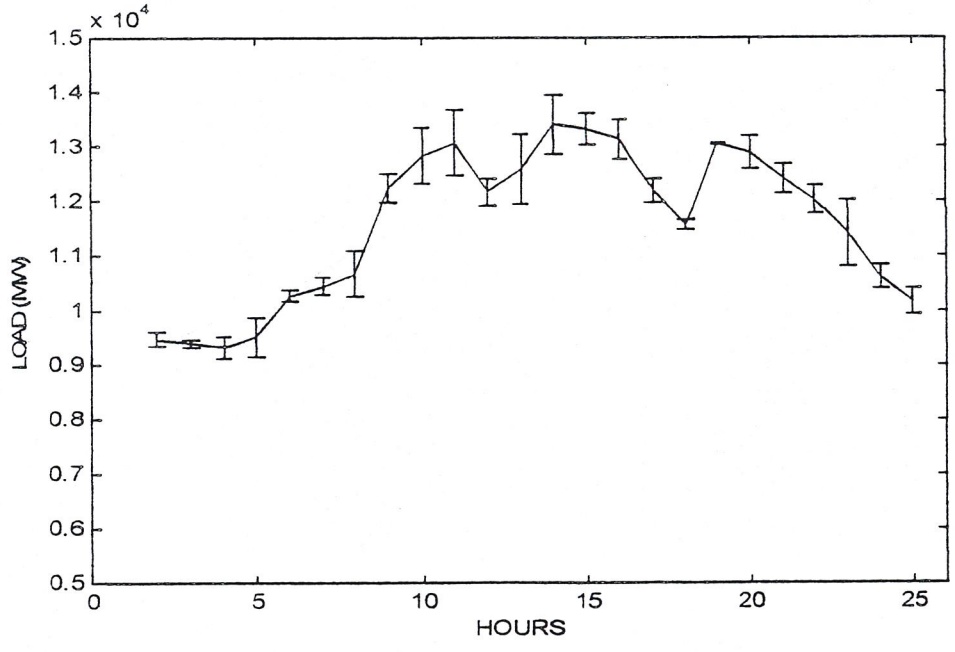
รูปที่ 5.58 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันพฤหัสโดยโครงข่าย G

ตารางที่ 5-14 แสดงผลการทำนายโหลดในวันพฤหัสของโครงข่าย H ที่ค่า Permissible = 0.00001 โดยมีค่าความผิดพลาดเฉลี่ย 2.6048 % และมีค่าความผิดพลาดสูงสุด 5.4186 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	9648.1	9330.807	-317.293	-3.2887
03.00	9385.8	9468.262	82.462	0.8786
04.00	9308.9	9108.527	-200.373	-2.1525
05.00	9504.7	9135.821	-368.879	-3.8810
06.00	10261.0	10163.310	-97.690	-0.9521
07.00	10438.6	10280.370	-158.230	-1.5158
08.00	10670.7	10253.830	-416.870	-3.9067
09.00	12240.1	11984.890	-255.210	-2.0850
10.00	12834.9	12326.680	-508.220	-3.9597
11.00	13076.5	12477.520	-598.980	-4.5806
12.00	12171.4	11922.430	-248.970	-2.0455
13.00	12594.4	11962.350	-632.050	-5.0185
14.00	13402.9	12854.220	-548.680	-4.0937
15.00	13321.6	13029.410	-292.190	-2.1934
16.00	13139.8	12768.850	-370.950	-2.8231
17.00	12204.4	11977.420	-226.980	-1.8598
18.00	11578.3	11486.380	-91.920	-0.7939
19.00	13062.1	13049.720	-12.380	-0.0948
20.00	12901.6	12594.460	-307.140	-2.3806
21.00	12411.5	12151.740	-259.760	-2.0929
22.00	12044.8	11778.570	-266.230	-2.2103
23.00	11421.4	10802.530	-618.870	-5.4185
24.00	10605.0	10384.350	-220.650	-2.0806
01.00	10159.1	9934.822	-224.278	-2.2077



รูปที่ 5-59 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันพฤหัสโดยโครงข่าย H (*)

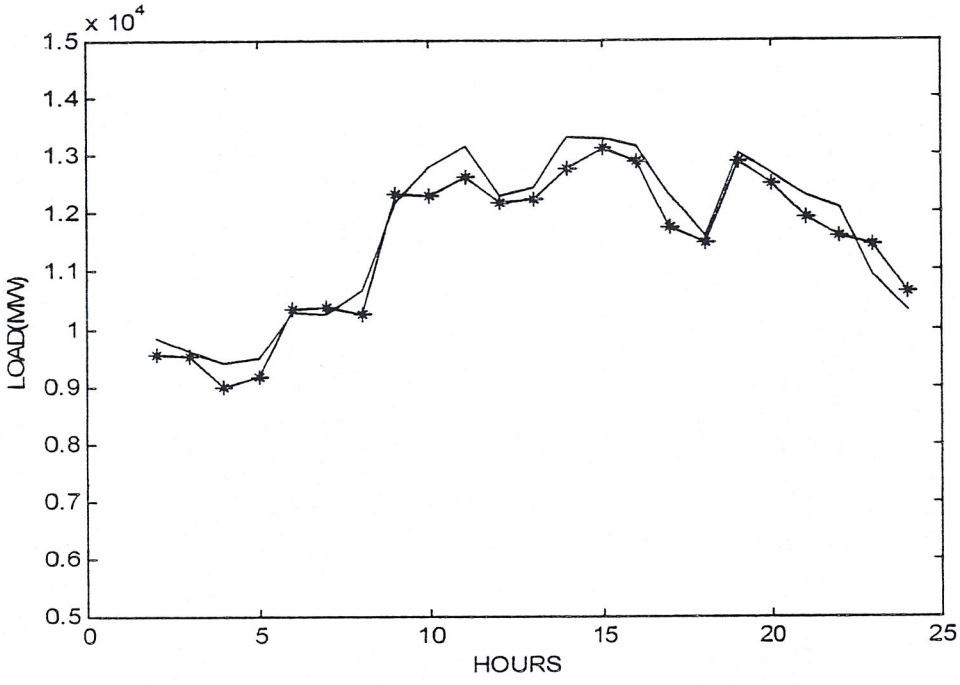


รูปที่ 5.60 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันพฤหัสโดยโครงข่าย H

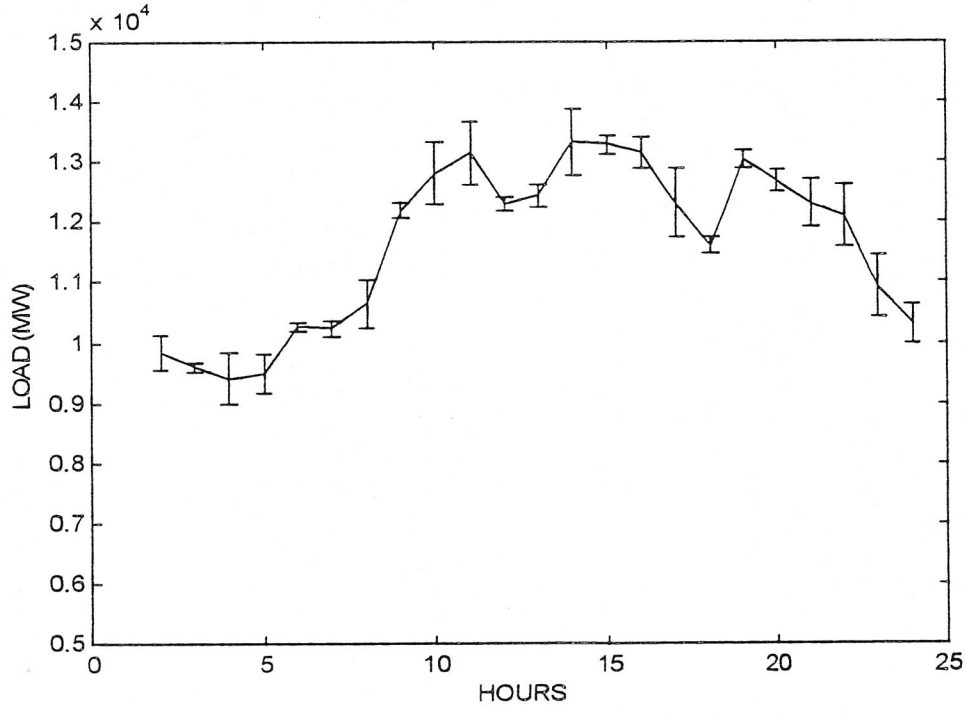
5.3.4 การทำนายโหลดวันศุกร์

ตารางที่ 5-15 แสดงผลการทำนายโหลดในวันศุกร์ของโครงข่าย G ที่ค่า Permissible = 0.0001 โดยมีค่าความผิดพลาดเฉลี่ย 2.6048 % และมีค่าความผิดพลาดสูงสุด 4.6900 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	9839.9	9543.346	-296.554	-3.0138
03.00	9597.2	9517.914	-79.286	-0.8261
04.00	9411.6	8989.190	-422.410	-4.4882
05.00	9486.8	9156.294	-330.506	-3.4839
06.00	10271.8	10343.190	71.390	0.6950
07.00	10237.2	10377.520	140.320	1.3707
08.00	10649.8	10263.720	-386.080	-3.6252
09.00	12193.3	12320.490	127.190	1.0431
10.00	12811.2	12289.710	-521.490	-4.0706
11.00	13152.1	12626.720	-525.380	-3.9946
12.00	12305.4	12193.000	-112.400	-0.9134
13.00	12435.3	12245.290	-190.010	-1.5280
14.00	13327.3	12771.040	-556.260	-4.1738
15.00	13292.1	13130.430	-161.670	-1.2163
16.00	13158.1	12902.260	-255.840	-1.9444
17.00	12311.6	11743.430	-568.170	-4.6149
18.00	11606.5	11471.950	-134.550	-1.1593
19.00	13040.8	12899.220	-141.580	-1.0857
20.00	12684.0	12497.350	-186.650	-1.4715
21.00	12308.0	11908.460	-399.540	-3.2462
22.00	12110.2	11609.790	-500.410	-4.1321
23.00	10931.8	11444.500	512.700	4.6900
24.00	10307.8	10629.800	322.000	3.1238



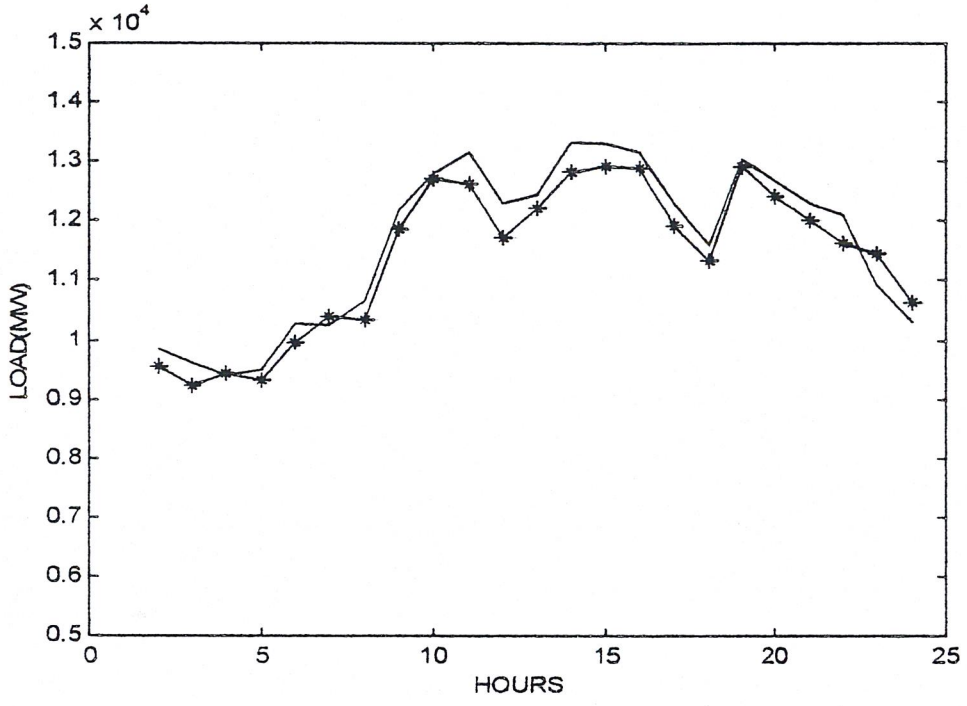
รูปที่ 5-61 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันศุกร์โดยโครงข่าย G (*)



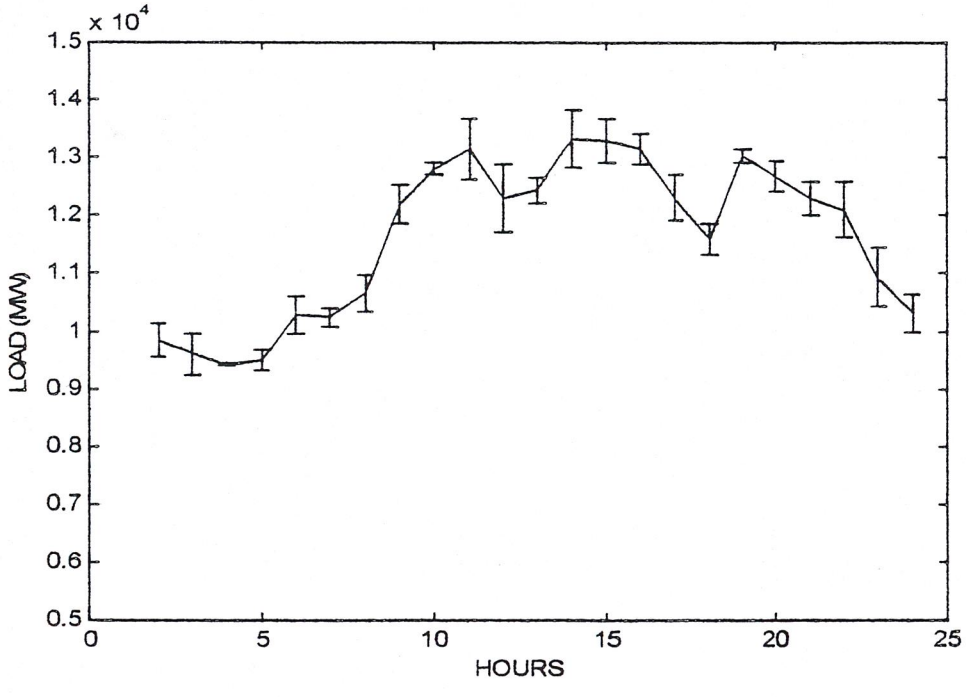
รูปที่ 5.62 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันศุกร์โดยโครงข่าย G

ตารางที่ 5-16 แสดงผลการทำนายโหลดในวันศุกร์ของโครงข่าย H ที่ค่า Permissible = 0.0001 โดยมีค่าความผิดพลาดเฉลี่ย 2.6822 % และมีค่าความผิดพลาดสูงสุด 4.6907 %

เวลา	โหลดจริง (MW)	โหลดทำนาย (MW)	Error (MW)	Error (%)
02.00	9839.9	9558.285	-281.615	-2.8620
03.00	9597.2	9227.118	-370.082	-3.8561
04.00	9411.6	9419.020	7.420	0.0788
05.00	9486.8	9296.497	-190.303	-2.0060
06.00	10271.8	9942.478	-329.322	-3.2061
07.00	10237.2	10400.880	163.680	1.5989
08.00	10649.8	10329.540	-320.260	-3.0072
09.00	12193.3	11857.910	-335.390	-2.7506
10.00	12811.2	12705.490	-105.710	-0.8251
11.00	13152.1	12626.590	-525.510	-3.9956
12.00	12305.4	11728.190	-577.210	-4.6907
13.00	12435.3	12223.700	-211.600	-1.7016
14.00	13327.3	12824.100	-503.200	-3.7757
15.00	13292.1	12916.400	-375.700	-2.8265
16.00	13158.1	12894.630	-263.470	-2.0023
17.00	12311.6	11917.900	-393.700	-3.1978
18.00	11606.5	11347.700	-258.800	-2.2298
19.00	13040.8	12927.130	-113.670	-0.8716
20.00	12684.0	12423.090	-260.910	-2.0570
21.00	12308.0	12011.340	-296.660	-2.4103
22.00	12110.2	11634.620	-475.580	-3.9271
23.00	10931.8	11444.506	512.706	4.6900
24.00	10307.8	10629.800	322.000	3.1238



รูปที่ 5-63 แสดงกราฟโหลดจริง (-) เทียบกับโหลดที่ได้จากการทำนายในวันศุกร์โดยโครงข่าย H (*)



รูปที่ 5.64 แสดงกราฟค่าความผิดพลาดจากการทำนายโหลดในวันศุกร์โดยโครงข่าย H

5.4 ผลการวิเคราะห์โครงข่ายประสาทเทียม

5.4.1 ผลการวิเคราะห์โครงข่ายประสาท

จากผลของการทดลองการทำนายโหลดในวันต่างๆนั้นจะเห็นได้ว่า โครงข่ายประสาทเทียมที่ได้นำมาใช้ในการทดลองจะมีความแตกต่างกันทางด้านลักษณะโครงสร้างของโครงข่าย โดยแต่ละโครงข่ายก็จะมีโครงสร้างที่แตกต่างกันออกไปตามจำนวนและลักษณะรูปแบบของอินพุทที่ใช้ ดังนั้นคุณลักษณะเฉพาะของโครงข่ายแต่ละโครงข่ายจึงมีความแตกต่างกันออกไป ดังเช่น ค่าอัตราการเรียนรู้และค่าสัมประสิทธิ์โมเมนตัมของโครงข่าย ดังที่ได้แสดงไว้ในตารางที่ 5.17 และตารางที่ 5.18

ตารางที่ 5.17 แสดงค่าอัตราการเรียนรู้ของแต่ละโครงข่าย G และ H ในการทำนายโหลดในวันต่างๆ

โครงข่ายประสาท	ค่าอัตราการเรียนรู้				
	วันจันทร์	วันอังคาร	วันพุธ	วันพฤหัส	วันศุกร์
G	0.40	0.20	0.40	0.40	0.50
H	0.25	0.40	0.30	0.40	0.80

ตารางที่ 5.18 แสดงค่าสัมประสิทธิ์โมเมนตัมของโครงข่าย G และ H ในการทำนายโหลดในวันต่างๆ

โครงข่ายประสาท	ค่าสัมประสิทธิ์โมเมนตัม				
	วันจันทร์	วันอังคาร	วันพุธ	วันพฤหัส	วันศุกร์
G	0.85	0.00	0.35	0.00	0.00
H	0.60	0.35	0.00	0.15	0.00

ค่าอัตราการเรียนรู้และค่าสัมประสิทธิ์โมเมนตัมของโครงข่ายที่เกิดขึ้น เราสามารถอธิบายได้ด้วยสมการของทางคณิตศาสตร์ตามทฤษฎีของ เกรเดียน เดสเซนต์ คือ

$$w_{ij}^{l}(t+1) = w_{ij}^{l}(t) + \eta\delta_{i}^{l}o_{j}^{l-1} + \alpha\left(w_{ij}^{l}(t) - w_{ij}^{l}(t-1)\right)$$

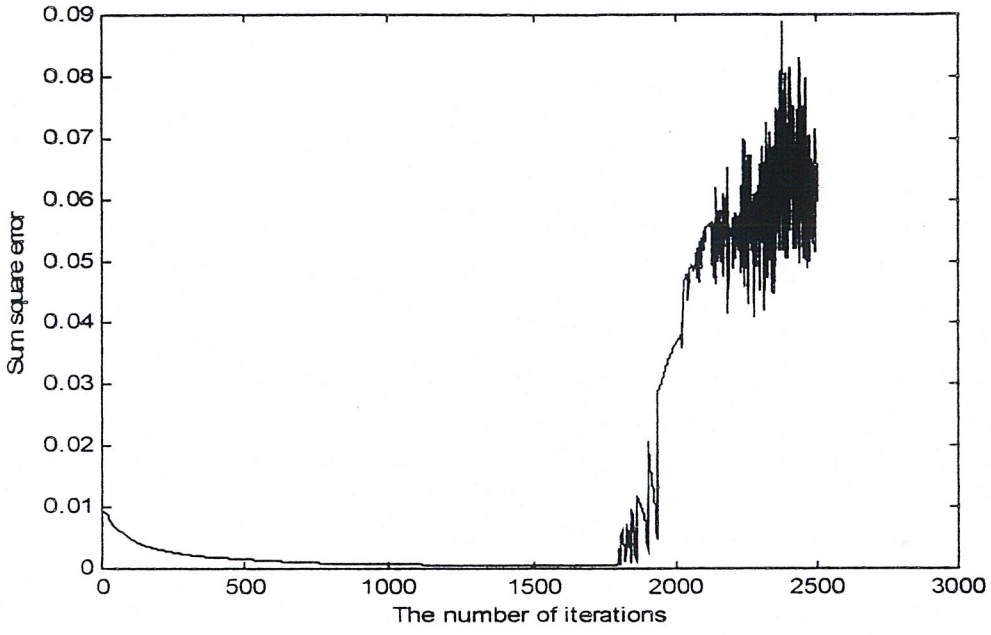
$$\delta_{i}^{L} = (T_{i} - O_{i}^{L})O_{i}^{L}(1 - O_{i}^{L})$$

$$\delta_{i}^{l} = (\sum_{r=1}^{N_{l}} \delta_{r}^{l+1} w_{ri}^{l+1})O_{i}^{l}(1 - O_{i}^{l})$$

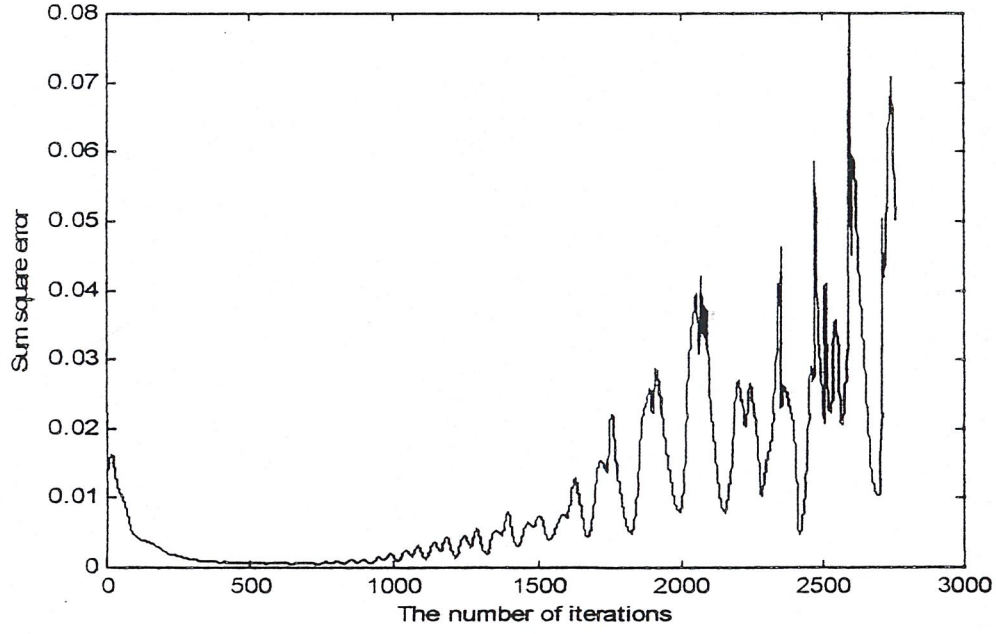
จากผลการทดลอง จะเห็นได้ว่า ค่าสัมประสิทธิ์โมเมนตัมของโครงข่ายประสาทบางโครงข่ายจะให้ค่าเป็น 0.00 ทั้งนี้เนื่องจาก ในกระบวนการการเรียนรู้ของโครงข่ายโครงข่ายจะทำการปรับค่าความผิดพลาดที่เกิดขึ้น โดยในการทดลองครั้งนี้ ได้ใช้วิธีการปรับค่าความผิดพลาดแบบแพร่ย้อนกลับ ซึ่งหากว่าค่าความต่างของน้ำหนักของการไซแนปส์ที่เกิดขึ้นก่อนและหลังการเรียนรู้ในรอบนั้นๆ มีค่าแตกต่างกันมาก เมื่อคูณกับค่าสัมประสิทธิ์โมเมนตัมแล้วจึงทำให้ผลลัพธ์ของเทอมนี้มีค่ามากขึ้น ซึ่งโดยปกติเรียกว่าค่าน้ำหนักเกรเดียน ซึ่งจะมีผลทำให้การเรียนรู้ของโครงข่ายประสาทอาจลู่เข้าสู่จุด Grobal Minimum และเลยออกไปจากจุดดังกล่าว จึงทำให้โครงข่ายประสาทต้องอาศัยจำนวนรอบของการเรียนรู้เพิ่มขึ้นเพื่อให้โครงข่ายสามารถกลับเข้าสู่จุด Grobal Minimum ได้ตามเดิม

ส่วนค่าอัตราการเรียนรู้ที่เกิดขึ้นในแต่ละโครงข่ายประสาทที่มีค่าแตกต่างกัน ทั้งนี้ก็เป็นผลเนื่องมาจากคุณลักษณะเฉพาะของโครงข่ายประสาทที่แตกต่างกันดังที่ได้กล่าวไว้ในข้างต้น

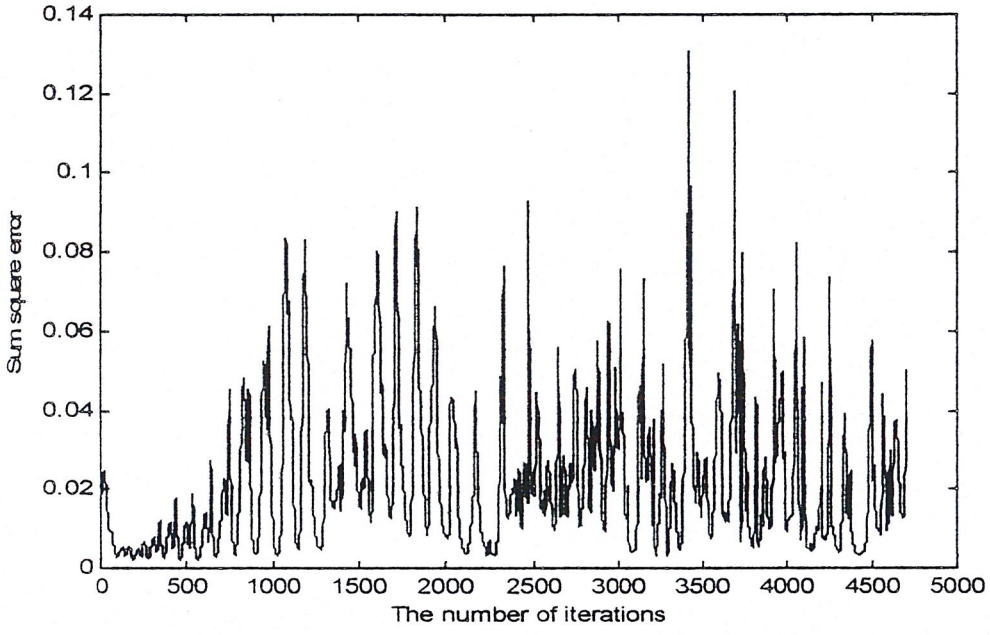
จากการนำทฤษฎีของ เกรเดียน–เดสเซนต์ มาประยุกต์ใช้ในกระบวนการฝึกหัดโครงข่ายประสาทเพื่อหลีกเลี่ยงการติดอยู่ที่ Local Minimum จำเป็นที่เราจะต้องทำการหาค่าอัตราการเรียนรู้และค่าสัมประสิทธิ์โมเมนตัมให้มีความเหมาะสมกับโครงข่ายนั้นๆ ทั้งนี้เนื่องจากหากค่าที่ได้ไม่มีความเหมาะสมแล้วจะทำให้การเรียนรู้ของโครงข่ายขาดเสถียรภาพหรือเกิดการแกว่งขึ้นในช่วงของการฝึกหัด ทำให้โครงข่ายไม่สามารถทำการเรียนรู้ได้



รูปที่ 5-65 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.5 และโมเมนตัม = 0 ของโครงข่าย A

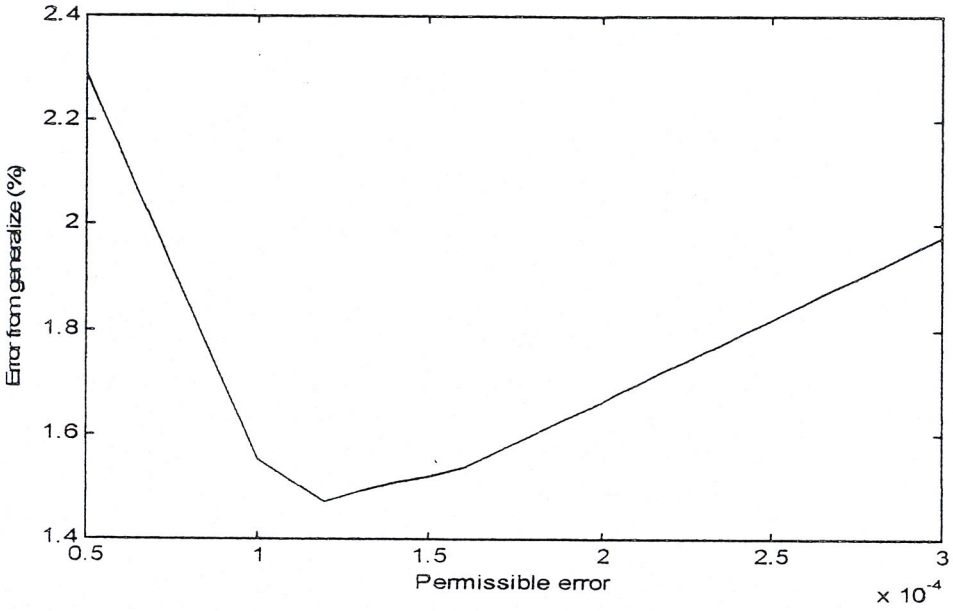


รูปที่ 5-66 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.40 และโมเมนตัม = 0.30 ของโครงข่าย B



รูปที่ 5-67 แสดงค่าความผิดพลาดของระบบต่อค่าอัตราการเรียนรู้ = 0.55 และโมเมนตัม = 0.40 ของโครงข่าย D

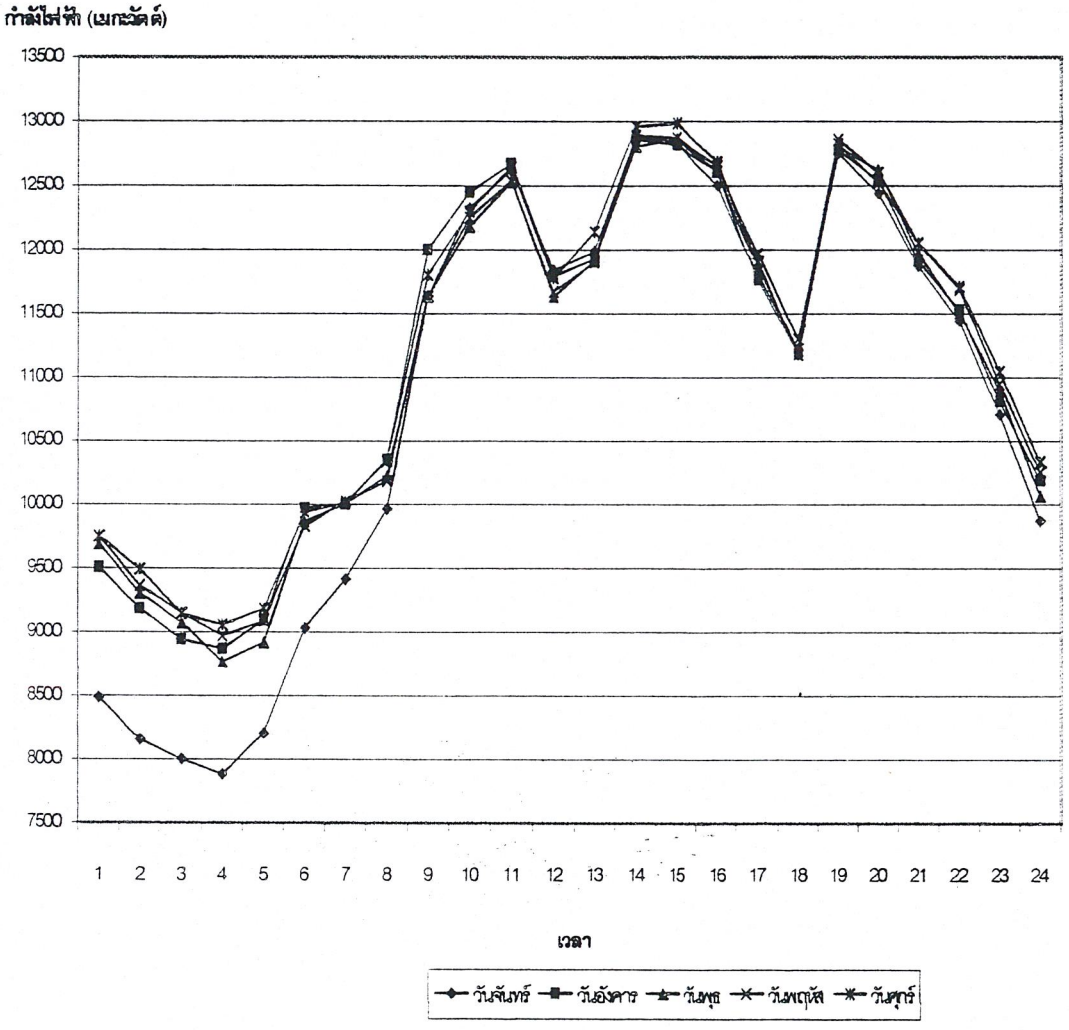
จากการทดลองการทำนายโหลดด้วยโครงข่ายประสาท G โดยอาศัยการปรับค่าความผิดพลาดที่ยอมรับได้ปรากฏว่า โครงข่ายประสาทเทียมสามารถประมวลผลข้อมูลได้อย่างมีประสิทธิภาพที่สุดที่ค่าความผิดพลาดจากการเรียนรู้ค่าหนึ่งๆ ซึ่งหากค่าความผิดพลาดที่เกิดจากการเรียนรู้ของโครงข่ายนั้นมีค่าลดลงจากจุดดังกล่าวแล้ว โครงข่ายประสาทเทียมก็จะมีประสิทธิภาพการประมวลผลที่ลดลง ทั้งนี้เนื่องจากในสภาวะดังกล่าว โครงข่ายประสาทเทียมจะทำงานในลักษณะของการจดจำคู่รูปแบบของอินพุทและเอาท์พุทหรือทำงานในลักษณะของการเมมโมไรเซชั่น (Memorization) ในทางกลับกันหากค่าความผิดพลาดที่เกิดจากขั้นตอนของการเรียนรู้มีค่าสูงกว่าจุดดังกล่าว โครงข่ายประสาทก็จะทำงานในสภาวะของการประมวลผลหรือทำงานในลักษณะของการเจนเนอรอลไลเซชั่น (Generalization) และมีประสิทธิภาพของการประมวลผลเป็นสัดส่วนโดยตรงกับค่าความผิดพลาดที่เกิดจากกระบวนการเรียนรู้



รูปที่ 5-68 แสดงความสัมพันธ์ของค่าความผิดพลาดที่เกิดจากกระบวนการการเรียนรู้ของโครงข่ายประสาทกับค่าความผิดพลาดที่เกิดจากการทำนายโหลดวันอังคาร ด้วยโครงข่ายโครงข่าย G

5.4.2 ผลการวิเคราะห์การทำนายโหลด

จากข้อมูลโหลดที่เกิดขึ้นในวันทำงานปกติ เราจะเห็นได้ว่า ลักษณะของเส้นกราฟโหลดของวันจันทร์ ณ เวลา 01.00 น. จนกระทั่งเวลา 08.00 น. มีกำลังการใช้ไฟฟ้าที่ต่ำกว่าในวันอื่นๆ ดังนั้น ในการทำการทดลองทำนายโหลดที่เกิดขึ้นในวันจันทร์จะมีลักษณะเฉพาะ จึงจำเป็นต้องใช้โครงข่ายประสาทเทียมแยกออกจากวันอื่นๆ โดยต่างหาก ส่วนในวันทำงานปกติที่เหลือจะใช้โครงข่ายประสาทเดียวกันในการทำนาย แต่ทว่า หากเราทำการพิจารณาถึงลักษณะพฤติกรรมของเส้นกราฟโหลดอย่างจริงจังแล้ว เราจะพบว่า ลักษณะของเส้นกราฟโหลดในส่วนของวันอังคารถึงวันศุกร์นั้นไม่ได้เหมือนกันเลยทีเดียว ดังนั้นเพื่อประสิทธิภาพของการทำนายโหลดในวันต่างๆ จึงได้ทำการทดลองการทำนายโหลดในวันต่างๆ โดยการใช้โครงข่ายประสาทเทียมเฉพาะวัน



รูปที่ 5-69 แสดงลักษณะของเส้นกราฟโหลดในวันต่างๆ

จากผลของการทดลองในครั้งนี้ จะเห็นได้ว่าในการทำนายโหลดด้วยโครงข่ายประสาทเทียมในแบบต่างๆที่มีจำนวนและลักษณะรูปแบบของอินพุทที่มีความแตกต่างกัน ทำให้เราสามารถจำแนกปัจจัยที่มีผลกระทบต่อกำลังการใช้ไฟฟ้าของประเทศไทยได้ดังต่อไปนี้ คือ

1) โหลด จากลักษณะของข้อมูลโหลดที่ได้ใช้ในการทดลองครั้งนี้ สามารถจำแนกออกได้เป็น 2 ประเภทใหญ่ๆด้วยกัน คือ ข้อมูลโหลดที่เกิดขึ้นในขณะทำนายโหลดและข้อมูลโหลดที่เกิดจริงในช่วงเวลาที่ต้องการทำนายโหลดในอดีต ซึ่งจากการทดลองจะเห็นได้ว่าหากข้อมูลโหลดมีการเปลี่ยนแปลงค่าไปผลของการทำนายโหลดโดยโครงข่ายประสาทเทียมก็จะมีแนวโน้มเปลี่ยนแปลงไปในทิศทางเดียวกันกับการเปลี่ยนแปลงของข้อมูลโหลด ทั้งนี้และทั้งนั้นโครงข่ายประสาทเทียมจะไม่ได้รับรู้ถึงความสำคัญมากน้อยของข้อมูลนั้นๆ แต่โครงข่ายประสาทเทียมจะทำงานโดยอาศัยการหาความสัมพันธ์ของคู่รูปแบบของอินพุทและเอาท์พุท ดังนั้นข้อมูลโหลดจึงเป็นคล้ายสิ่งที่แสดงถึงระดับของโหลดอ้างอิงที่สภาวะแวดล้อมนั้นๆ

2) อุณหภูมิ อุณหภูมิเป็นปัจจัยที่สำคัญอีกประการหนึ่งที่ส่งผลกระทบต่ออัตราการใช้กำลังไฟฟ้าของประเทศ โดยผลกระทบที่เกิดจากอุณหภูมินั้นเราสามารถจำแนกออกได้เป็นประเภทดังต่อไปนี้ คือ

- ผลกระทบที่มีต่อระบบการส่งจ่ายกำลังไฟฟ้า กำลังการส่งจ่ายกำลังไฟฟ้าผ่านระบบสายส่ง (Transmossion line) นั้นจะถูกจำกัดด้วยอุณหภูมิของสายตัวนำหรือที่เรียกกันว่า เทอร์มอลลิมิตของสายตัวนำและอุณหภูมิจะมีผลโดยตรงต่อค่าของการสูญเสียที่เกิดขึ้นในระบบสายส่งกำลัง

- ผลกระทบที่เกิดขึ้นต่อโหลดทางเศรษฐกิจและเขตที่พักอาศัย อุณหภูมิได้เป็นปัจจัยที่สำคัญต่อการใช้กำลังไฟฟ้าของโหลดประเภทนี้ ทั้งนี้เนื่องจากแหล่งธุรกิจการค้าหรือที่พักอาศัยล้วนแล้วแต่ต้องการอำนวยความสะดวกสบายให้กับลูกค้าหรือผู้พักอาศัย ดังจะเห็นได้จากการติดเครื่องปรับอากาศในแหล่งธุรกิจหรือที่พักอาศัย และโดยเฉพาะในฤดูร้อนแล้วจะเห็นได้ว่า การใช้กำลังไฟฟ้าของโหลดประเภทนี้จะมีอัตราการเพิ่มขึ้นเป็นอย่างมาก

- ผลกระทบที่เกิดขึ้นต่อโหลดทางอุตสาหกรรม จะเห็นได้โดยทั่วไปว่า การทำงานของเครื่องจักรต่างๆ ส่วนใหญ่แล้วต้องการการระบายความร้อนทั้งสิ้น โดยเฉพาะเครื่องจักรที่มีขนาดใหญ่หรือมีพิกัดกำลังสูงด้วยแล้ว การระบายความร้อนเป็นสิ่งที่หลีกเลี่ยงไม่ได้ ไม่ว่าจะเป็นการระบายความร้อนด้วยอากาศหรือการระบายความร้อนด้วยน้ำก็ตาม ทั้งนี้เนื่องจากอุณหภูมิที่สูงเกินไปจะเป็นอันตรายต่อการทำงานของเครื่องจักรและทำให้เครื่องจักรนั้นมีอายุการใช้งานสั้นลง ดังนั้น หากสภาวะแวดล้อมโดยรอบมีอุณหภูมิสูงขึ้นแล้วย่อมทำให้ระบบการระบายความร้อนของเครื่องจักรต้องทำงานมากขึ้น เพื่อระบายความร้อนออกจากตัวเครื่องจักรได้ในอัตราเท่าเดิม

3) ความชื้น ความชื้นในอากาศเกิดจากโมเลกุลของ H_2O ในอากาศมีลักษณะเป็นก๊าซไฟฟ้าลบอ่อนๆ ทำให้อากาศที่มีความชื้นมีความคงทนต่อแรงดันสูงขึ้น ตราบเท่าที่ความความชื้นยังไม่กลั่นตัวเป็นหยดน้ำหรือยังไม่ถึงจุดน้ำค้าง โดยสามารถเขียนแสดงความสัมพันธ์ได้ดังนี้ คือ [8]

$$U_b = (a + bd) \cdot \delta \cdot \sqrt{5.1 * 10^{-2} (h_a + 8.65)} \qquad (5.1)$$

a = 20 kV (สำหรับ DC +), 15kV(สำหรับ DC -)

b = 5.1 kV/cm , d = ระยะแกปเป็น cm

$h_a = 4 \ldots 40$ g/m^3

δ = ความหนาแน่นของอากาศ

แต่จากผลของการทำนายโหลดด้วยโครงข่ายประสาท G และ H ปรากฏว่า โครงข่ายประสาทเทียมแบบ H นั้นให้มีความสามารถในการทำนายที่แม่นยำกว่าโครงข่ายประสาท G ทั้งที่ไม่มีอินพุทที่เป็นความชื้นสัมพัทธ์อยู่เลย ทั้งนี้อาจเป็นสาเหตุมาจากการที่ค่าความชื้นสัมพัทธ์ที่ได้จากการวัดค่าตามจังหวัดต่างๆที่ได้กำหนดไว้มีค่าที่สูงหรือต่ำเกินกว่าระดับเฉลี่ยหรือระดับมาตรฐานที่สามารถอ้างอิงรวมถึงจังหวัดต่างๆรอบข้างได้หรืออาจเป็นผลมาจากการเปลี่ยนแปลงความชื้นสัมพัทธ์ที่เกิดขึ้นอย่างรวดเร็ว เช่น ในกรณีที่เกิดฝนตกจะทำให้ค่าความชื้นสัมพัทธ์ของอากาศเปลี่ยนแปลงไปอย่างรวดเร็ว ทำให้การใช้ค่าความชื้นสัมพัทธ์ในช่วงของเหตุการณ์ดังกล่าวนั้นเกิดค่าความผิดพลาดขึ้นในตัวของอินพุทเอง จึงทำให้เกิดความผิดพลาดในการทำนายขึ้น

4) ลม จะเห็นได้ว่าในประเทศที่มีการพัฒนาแล้ว การผลิตกระแสไฟฟ้าในบางประเทศได้ใช้การผลิตกำลังไฟฟ้าโดยอาศัยแรงลม ซึ่งจะเป็นผลทำให้บางครัวเรือนนั้นสามารถผลิตกระแสไฟฟ้าใช้ขึ้นเองได้ ทำให้ความต้องการโหลดจากระบบโดยส่วนรวมนั้นลดลง โดยเฉพาะอย่างยิ่งในประเทศที่มีลมแรง

แต่สำหรับข้อมูลของระดับความเร็วลมในประเทศนั้นมีค่าที่ไม่ต่อเนื่องเท่าใดและมีความแตกต่างกันของข้อมูลมาก จึงทำให้เกิดการการบีบตัวอยู่ในช่วงใดช่วงหนึ่งของฟังก์ชั่นซิกมอยด์เมื่อทำการกระตุ้นอินพุทในขณะที่ทำการจัดรูปแบบข้อมูลของโครงข่าย ทำให้เกิดค่าความผิดพลาดขึ้นในตัวของข้อมูลซึ่งจะส่งผลต่อการทำนายโหลดด้วยต่อไป ทำให้ผลการทำนายโหลดในวันนั้นๆที่เกิดขึ้นมีค่าความผิดพลาดสูง

5) เวลา จากลักษณะของเส้นกราฟโหลดของวันต่างๆนั้นปรากฏว่า ลักษณะของเส้นกราฟงโหลดจะมีการเปลี่ยนแปลงตามค่าเวลาเช่นกัน ดังนั้น ในการฝึกหัดโครงข่ายที่มีรูปแบบของอินพุทที่แสดงถึงเวลาจะทำให้โครงข่ายประสาทสามารถเรียนรู้ได้ถึงลักษณะของของเส้นกราฟโหลดที่เกิดขึ้น ณ เวลาต่างๆได้ ซึ่งเราสามารถสังเกตุได้จากผลการทดลองในโครงข่ายที่มีรูปแบบอินพุทที่แสดงถึงเวลา ได้แก่ โครงข่ายประสาท C จนถึงโครงข่ายประสาท H จะเห็นได้ว่า โครงข่ายประสาทสามารถให้ผลการทำนายที่มีประ-สิทธิภาพและความรวดเร็วในการฝึกหัดโครงข่ายมากกว่าโครงข่ายที่ไม่มีรูปแบบของอินพุทเวลา

6) วันหยุด ลักษณะของเส้นกราฟการใช้กำลังไฟฟ้าที่เกิดขึ้นในขณะวันหยุดจะมีลักษณะเฉพาะตัวและมีความแตกต่างออกไปจากลักษณะของกราฟโหลดในวันปกติและวันหยุดอื่นๆด้วยกัน ทำให้การทำนายโหลดในวันหยุดจำเป็นต้องอาศัยโครงข่ายประสาทเฉพาะวันหยุดนั้นๆ ทั้งนี้เราไม่สามารถที่จะนำโครงข่ายประสาทแบบอื่นๆมาใช้ในการทำนายโหลดในวันหยุดนั้นๆได้

บทที่ 6
บทวิจารณ์และสรุป

เนื่องจากการทำนายการใช้กำลังไฟฟ้านั้นเราไม่สามารถแสดงได้ด้วยสมการทางคณิตศาสตร์ใดๆ ทั้งนี้เนื่องจากโหลดจะมีการเปลี่ยนแปลงอยู่ตลอดเวลาตามสภาวะแวดล้อมในขณะนั้น ไม่ว่าจะเป็นสภาวะอากาศหรือสภาวะทางเศรษฐกิจก็ตาม ดังนั้นการนำโครงข่ายประสาทเทียมเข้ามาประยุกต์ใช้ในการทำนายโหลดจึงเป็นอีกวิธีหนึ่งในการหาความสัมพันธ์ของการเปลี่ยนแปลงของโหลดกับสภาวะแวดล้อมในขณะนั้นๆ โดยอาศัยคุณลักษณะสำคัญของโครงข่ายประสาทเทียมที่มีความสามารถในด้านการประมวลผลข้อมูลที่มีความซับซ้อน

จากลักษณะของข้อมูลที่ใช้ในการฝึกหัดโครงข่ายและการทำนายกำลังไฟฟ้า จะเห็นได้ว่า เราต้องใช้ข้อมูลจำนวนมากโดยเฉพาะข้อมูลทางสภาวะอากาศ ซึ่งในการทดลองครั้งนี้ได้ใช้ข้อมูลสภาวะอากาศใน 7 จังหวัดอ้างอิงแทนสภาวะอากาศทั้งประเทศ (ดังที่ได้กล่าวไว้ในบทที่ 4) ดังนั้นการเชื่อมต่อของระบบเครือข่ายจึงเป็นสิ่งที่หลีกเลี่ยงไม่ได้ในการรับส่งข้อมูลเข้าสู่ระบบฐานข้อมูลเพื่อใช้ในการฝึกหัดโครงข่ายและการทำนายผลที่เกิดขึ้น ในโครงงานครั้งนี้ได้นำคุณสมบัติอันโดดเด่นของระบบอินเตอร์เน็ตเข้ามาใช้ในส่วนของการรับส่งข้อมูลจากสถานที่ต่างๆเข้ามาเก็บในระบบฐานข้อมูล ซึ่งในการทดลองได้ผลเป็นที่น่าพอใจ คือ ข้อมูลจากเครื่องลูกข่ายสามารถส่งข้อมูลเข้ามาเก็บยังระบบฐานข้อมูลในเครื่องเซิร์ฟเวอร์ได้เป็นอย่างดีไม่ว่าเครื่องลูกข่ายจะตั้งอยู่ที่ตำแหน่งใดก็ตาม

จากผลของการทำนายการใช้กำลังไฟฟ้าโดยผ่านระบบเครือข่ายในการทำโครงงานในครั้งนี้ ได้ผลการทำนายโหลดในวันปกติเป็นที่น่าพอใจในระดับหนึ่ง ทั้งนี้เนื่องจาก ค่าความผิดพลาดเฉลี่ยที่เกิดขึ้นจากการทำนายยังมีค่าค่อนข้างสูง คือ ประมาณ 2.6048 % และโครงข่ายประสาทต้องใช้ระยะเวลาในการฝึกหัดที่ค่อนข้างมาก

จากขอบเขตของการทำโครงงานในครั้งนี้ ได้ศึกษาถึงผลกระทบของการเปลี่ยนแปลงของโหลดจากสภาวะอากาศเป็นหลักโดยไม่ได้คำนึงถึงผลจากสภาวะทางเศรษฐกิจ ดังนั้นอาจเป็นสาเหตุหนึ่งที่ทำให้การทำนายโหลดในโครงงานครั้งนี้ได้ผลที่มีค่าความผิดพลาดที่สูงพอสมควร เนื่องจากในขณะที่ทำการทดลองในครั้งนี้ ประเทศไทยได้ประสบปัญหาเกี่ยวกับสภาวะทางเศรษฐกิจเป็นอย่างมาก ซึ่งย่อมส่งผลกระทบต่อพฤติกรรมของโหลดที่เกิดขึ้นในช่วงเวลาดังกล่าว

ปัญหาที่เกิดขึ้นและแนวทางในการพัฒนา

1) ผลการทำนาย

จากผลของการทำนายโหลดในการทดลองครั้งนี้เกิดค่าความผิดพลาดที่ค่อนข้างสูงดังที่ได้กล่าวไว้ในข้างต้น ซึ่งอาจเป็นผลมาจากรูปแบบของข้อมูลที่ใช้ในการฝึกหัดโครงข่ายมีจำนวนที่ค่อนข้างน้อยมาก คือ มีจำนวนทั้งสิ้น 16 รูปแบบต่อการทำนาย 1 ครั้ง ทั้งนี้ก็เพื่อหลีกเลี่ยงผลกระทบของการเปลี่ยนแปลงโหลดอันเนื่องมาจากการเปลี่ยนแปลงของฤดูกาล (Seasons change) ที่มีอิทธิพลต่อการใช้กำลังไฟฟ้าค่อนข้างมาก จึงอาจเป็นส่วนหนึ่งที่ทำให้โครงข่ายประสาทให้ผลตอบสนองที่มีความผิดพลาดที่ค่อนข้างสูง

จากลักษณะการทำงานของโครงข่ายในโครงงานนี้ มีลักษณะการทำงานในลักษณะเป็นระบบเปิด (Open-loop system) ดังนั้นหากเรานำค่าความผิดพลาดที่เกิดขึ้นจากการทำนายในของโครงข่ายมาเป็นอินพุทย้อนกลับของโครงข่ายประสาทหรือกล่าวอีกนัยหนึ่งก็คือ การทำงานในลักษณะระบบปิด (Close-loop system) นั่นเอง อาจทำให้โครงข่ายมีประสิทธิภาพในการทำนายที่ดีขึ้น ทั้งนี้เนื่องจากโครงข่ายมี Feedback ย้อนกลับเป็นตัวควบคุมการประมวลผลของโครงข่ายตลอดเวลา

จากลักษณะของโครงข่ายประสาท ที่จะให้ผลตอบสนองที่มีประสิทธิภาพที่สุดที่ค่าความผิดพลาดที่เกิดจากการเรียนรู้ที่ค่าหนึ่งๆแล้ว ดังนั้นการทำนายโหลดของโครงข่ายจะให้ผลถูกต้องที่สุดที่ตำแหน่งนั้นๆด้วยเช่นกัน แต่ผลที่ได้จากการทำนายในบางส่วนในการทดลองครั้งนี้ไม่สามารถนำค่าน้ำหนักของการไซแนปส์ของโครงข่ายจากตำแหน่งมาใช้ในการทำนายได้ ทั้งนี้เนื่องจากระยะเวลาในการฝึกหัดโครงข่ายที่ใช้เวลานานมาก จึงเป็นผลให้ผลการทำนายบางส่วนมีค่าความผิดพลาดที่ค่อนข้างสูง ดังนั้นเราควรหาวิธีหาตำแหน่งที่ทำให้โครงข่ายตอบสนองได้ดีและใช้ค่าน้ำหนักของโครงข่ายที่ตำแหน่งนั้นๆในการทำนาย เช่น วิธีการใส่สิ่งรบกวนในโครงข่าย (Noise Methods) เป็นต้น

2) ระยะเวลาในการฝึกหัดโครงข่าย

เนื่องจากในการทำโครงงานในครั้งนี้ได้ใช้โครงข่ายประสาทเทียมแบบฟีดฟอร์เวิร์ด โดยมีการเรียนรู้แบบแพร่ย้อนกลับและอาศัยการปรับค่าน้ำหนักของการไซแนปส์ตามทฤษฎีของ เกรเดียน-เดสเซนต์ จะเห็นได้ว่าโครงข่ายนั้นใช้ระยะเวลาในขั้นตอนการฝึกหัดค่อนข้างมาก ดังนั้นเราอาจปรับปรุงรูปแบบของกระบวนการปรับค่าน้ำหนักของโครงข่ายประสาทเทียมด้วยวิธีการอื่นๆ อาทิเช่น การใช้เทคนิค Fast Back-Propagation, การใช้คอนจูเกตเกรเดียน (Conjugate Gradient) หรือใช้วิธีนิวตันเม็ทธอด (Newton's Method) เป็นต้น

3) เสถียรภาพในการฝึกหัดโครงข่าย

จากผลการทดลองจะเห็นได้ว่า ในการปรับค่าพารามิเตอร์ต่างๆของโครงข่ายในกระบวนการเรียนรู้ของโครงข่ายนั้นอาจทำให้เสถียรภาพของโครงข่ายเสียไป ดังจะเห็นได้จากการแกว่ง (Oscillate) ในขั้นตอนการปรับค่าน้ำหนักของการไซแนปส์ของโครงข่ายประสาท และจากการทดลองจะพบว่าโครงข่ายประสาทแต่ละโครงข่ายนั้นจะมีค่าพารามิเตอร์ที่แตกต่างกัน ดังนั้นเราอาจนำวิธีอื่นๆเข้ามาประยุกต์ใช้ในการหาค่าพารามิเตอร์ที่เหมาะสมของโครงข่ายประสาทเทียมแต่ละโครงข่าย เช่น เจเนติค อัลกอริธึ่ม (Genetic Algorithm) เป็นต้น

ภาคผนวก ก.

การทำนายโหลดโดยสมการทางคณิตศาสตร์

(Load Forecasting by Mathematical Methods)

การทำนายโหลดด้วยวิธีนี้เป็นการทำนายโหลดโดยการใช้สมการทางคณิตศาสตร์หาความสัมพันธ์ระหว่างสภาพอากาศต่อการเปลี่ยนแปลงของโหลด ซึ่งมีรายละเอียดดังนี้ [5, 7] คือ

1.1) PEAK LOAD MODEL

หาได้จากสมการ $P = B + F(W)$; P = PEAK LOAD

B = BASE LOAD

F(W) = WEATHER-DEPENDENT COMPONENT

1.2) LOAD SHAPE

ในการพิจารณาถึง LOAD SHAPE นั้น มี TECHNIQUE อยู่ 2 ประเภท คือ

1.2.1) TIME OF DAY

- SUMMATION OF EXPLICIT TIME FUNCTION MODEL

วิธีนี้จะทำการหา LOAD SHAPE โดยใช้การ SAMPLING ที่เวลาใดๆตลอดช่วงเวลา T โดยการใช้อนุกรมเวลา (Time Series)

$$\{z(t), t = 1, 2, \ldots, T\}$$

$$Z(t) = \sum_{i=1}^{N} \alpha_i f_i(t) + v(t), t \in \tau$$

โดยที่ Z(t) คือ ค่า LOAD ที่เวลา t ใดๆ

α คือ ค่า Coefficient

$f_i(t)$ คือ Time Function

- SPECTRAL DECOMPOSITION MODEL

วิธีนี้จะใช้สมการเดียวกันกับวิธี Time of day model แต่จะใช้ $f_i(.)$ แสดงถึง EIGENFUNCTION ซึ่งเป็นผลมาจาก Time Series ซึ่งวิธีนี้ค่า $f_i(.)$ จะได้จากการประมาณค่าของกระบวนการ Autocorrelation Matrix และจากแก้สมการหาค่า EIGENVALUE ดังนั้นวิธีการนี้จึงไม่เหมาะสมที่จะนำมาใช้ในการปฏิบัติแบบ REAL TIME

1.2.2) DYNAMIC

- AUTOMATIC REGRESSTION MOVING AVERAGE(ARMA) [4, 5, 8]

$$Z(t) = Y_p(t) + Y(t)$$

; $Y_p(t)$ คือ Component ที่สภาวะอากาศปกติ

Y (t) คือ Component ที่ขึ้นอยู่กับสภาวะของอากาศและ random effect

$$y(t) = \sum_{i=1}^{n} a_i y(t-i) + \sum_{i=1}^{n_u} \sum_{j_k=0}^{m_k} b_{j_k} u_k(t-j_k) + \sum_{h=1}^{H} c_h w(t-h)$$

โดยที่ $U_k(t)$, $k = 1,2,\ldots,n_u$ คือ Weather Dependent Input

w(t) คือ random load

$$Z'(t) = Z(t) - Z(t - t_p)$$

ในการหา LOAD COMPONENT ในลักษณะของช่วงเวลาโดยการใช้ Pre – Filtering สามารถหาได้จากสมการ

; t_p คือ Period time of day component

- STATE-SPACE MODEL วิธีนี้มีรากฐานมาจากสมการที่ (1) เช่นเดียวกันกับ ARMA Methods

$$Z'(t) = C^T x(t)$$

; x(t+1) = Ax(t) + Bu(t) + W(t)

x(t) คือ state vector ที่เวลา t

u(t) คือ vector ของอากาศในสภาวะปกติ

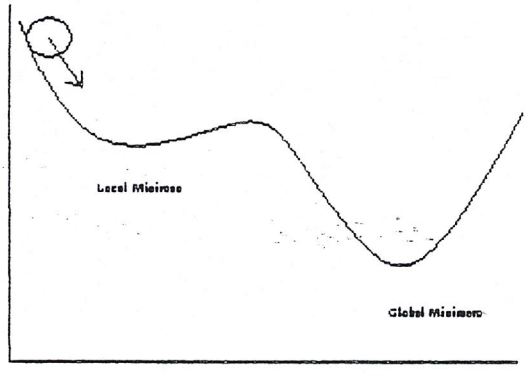
W(t) คือ vector of random

ภาคผนวก ข.

กฎของเกรเดียนเดสเซนต์

(Gradient Descent Rule)

ค่าน้ำหนัก (Weights) ของไซแนปส์ (Synaps) จะมีการปรับปรุงในแต่ละรอบของกระบวนการเรียนรู้ (Training Process) ซึ่งจะมีค่าเป็นสัดส่วนกับค่าดิฟเฟอร์เรนเชียล (Differential) หรือค่าเกรเดียน (The Gradient) ของความผิดพลาดระหว่างค่าเอาท์พุทตอบสนองต่อเป้าหมาย (Target Response) กับค่าเอาท์พุทจริง (Actual Target) โดยมีเป้าหมาย ให้ความผิดพลาดที่เกิดขึ้นมีค่าลดลงดังแสดงไว้ในรูปที่ ผ. ข-1 [6]



รูปที่ ผ. ข-1 แสดงการลาดลงของค่าความผิดพลาดไปสู่ตำแหน่งที่ต่ำที่สุด

ภาคผนวก ค.

ฐานข้อมูลและการจัดการข้อมูล

(Databases and Information Management)

1. ดาตาเบสเซิร์ฟเวอร์

ดาตาเบสเซิร์ฟเวอร์ (Database Server) เป็นกุญแจสำคัญที่สามารถแก้ปัญหาต่างๆของการจัดเก็บ ข้อมูลหรือกล่าวได้ว่าเซิร์ฟเวอร์ (Server) ต้องจัดเก็บข้อมูลที่มีขนาดใหญ่ในลักษณะที่มีผู้ใช้หลายคน(Multiuser) และต้องการความน่าเชื่อถือด้านข้อมูล ดังนั้นผู้ใช้ต่างๆสามารถเข้าถึงข้อมูลเดียวกันได้ในเวลาเดียวกัน ซึ่งทั้งหมดนี้ต้องใช้งานในขณะที่ระบบฐานข้อมูลนั้นมีประสิทธิภาพสูงสุด ดาตาเบสเซิร์ฟเวอร์ต้องป้องกันการเข้าถึงข้อมูลของบุคคลที่ไม่ได้รับอนุญาตและจัดการหาวิธีการแก้ไขที่มี ประสิทธิภาพเมื่อเกิดความผิดพลาดขึ้นซึ่งออราเคิล (Oracle Server) มีศักยภาพและประสิทธิผลในการแก้ปัญหาเหล่านี้

ออราเคิลเซิร์ฟเวอร์

(1) ไคลเอนท์ / เซิร์ฟเวอร์ (Client/Server distributed processing environments)

เพื่อที่จะเป็นตัวให้และตัวรับในระบบคอมพิวเตอร์หรือเครือข่ายคอมพิวเตอร์ ออราเคิลยอมให้มีการประมวลผลแยกได้ระหว่างดาตาเบสเซิร์ฟเวอร์และไคลเอนท์แอพพลิเคชั่นโปรแกรม (Client application programs) ขณะที่คอมพิวเตอร์ทำงานระบบจัดการข้อมูลจะจัดการระบบดาตาเบสเซิร์ฟเวอร์ข้อมูลทั้งหมดความสามารถในการตอบสนองในขณะที่เครื่องลูกข่าย (Workstations) ทำงานดาตาเบสแอพพลิเค- ชั่น (Database application) ก็ทำการแปลความหมายและแสดงผลของข้อมูล (Database application) ในที่นี้คือ โปรแกรมต่างๆในออราเคิล

(2) สนับสนุนผู้ใช้หลายคน (Many concurrent database users)

ออราเคิลสามารถสนับสนุนผู้ใช้จำนวนมากที่ปฏิบัติงานในเวลาเดียวกันและจัดการข้อมูลอย่างเดียวกันซึ่งจะทำการจัดข้อมูลให้มีขนาดเล็กลง รวมทั้งประกันการเข้ามาพร้อมกันของข้อมูลซึ่งนั่นหมายถึงความมีประสิทธิภาพในการจัดการข้อมูล และออราเคิลสนใจกับลำดับความสำคัญของข้อมูลว่ามีขนาดเล็กหรือใหญ่รวมทั้งมีระบบจัดการข้อมูลที่มากๆและมีประสิทธิภาพสูง

2 ภาษาเอสคิวแอล (SQL: Structured Query Language)

เอสคิวแอล คือ ภาษาที่เป็นคำสั่งซึ่งใช้ในการเข้าถึงข้อมูลในระบบฐานข้อมูลออราเคิล (Oracle database) เมื่อผู้ใช้ต้องการที่จัดการข้อมูลในระบบฐานข้อมูล โปรแกรมใช้งานและเครื่องมือของออราเคิลจะไม่ยอมให้ผู้ใช้เข้าถึงตัวข้อมูลในฐานข้อมูลได้โดยตรงแต่จะให้ผู้ใช้เข้าถึงโดยทางเอสคิวแอล

2.1) คำสั่งของภาษาเอสคิวแอล

ภาษาเอสคิวแอลเป็นภาษาที่ง่ายต่อการใช้งานโดยมีลักษณะคล้ายคำสั่งภาษาอังกฤษ ซึ่งรายละเอียดและรูปแบบทางไวยากรณ์ของคำสั่งเอสคิวแอลต่างๆ ช่วยให้สามารถค้นหาคำสั่งหารูปแบบทางไวยากรณ์และคำอธิบายโดยย่อว่าทำงานอย่างไร ในส่วนที่ใช้ในการเก็บข้อมูลของโครงงานในครั้งนี้การเก็บข้อมูลในรูปของตาราง ดังนั้นประการแรก เราต้องสร้างตารางก่อนโดยใช้คำสั่ง CREATE TABLE ซึ่งมีรูปแบบทางไวยกรณ์ดังนี้

CREATE TABLE

ตาราง ค-1 แสดงไวยกรณ์คำสั่งของอนุประโยค CREATE TABLE

CREATE TABLE *<table name>*
({*<column name>* *<data type>* [*<size>*}]
[*<colunstraint>*...]
[*<defvalue>*]}.,*<tabconstraint >*.,...);

ตาราง ค-2 แสดงองค์ประกอบและคำนิยามของคำสั่งของอนุประโยค CREATE TABLE

องค์ประกอบ	คำนิยาม
<table name >	ชื่อของตารางที่สร้างขึ้นด้วยคำสั่งนี้
<column name>	ชื่อคอลัมน์ของตาราง
<data type>	ชนิดของข้อมูลที่จะอยู่ในคอลัมน์อาจเป็นชนิดใดก็ได้ดังต่อไปนี้ INTEGER, CHARACTER, DECIMAL, NUMERIC, FLOAT, SMALLINT, REAL, DOUBLE, PRECISION, LONG*, VARCHAR, DATE, TIME * (* แสดงว่าไม่มาตรฐาน)
<size>	ความหมายขององค์ประกอบนี้ขึ้นกับ*<datatype>*
<colconstrnt>	อาจเป็นอย่างหนึ่งอย่างใดดังต่อไปนี้NOTNULL, UNIQUE, PRIMARYKEY, CHECK(<predicate>), DEFUALT=*<valueexpression>*, REFERENCES<table name>[(*<column name >*)]
<tabconstrnt>	อาจเป็นอย่างหนึ่งอย่างใดดังต่อไปนี้ UNIQUE,PRIMARY KEY,(*<column name>*.,...) REFERENCE<table name >[(*<column name>*)]

INSERT

คำสั่ง INSERT เป็นคำสั่งป้อนค่าต่างๆลงในตารางที่ได้สร้างขึ้นมาแล้วโดยป้อนเป็นรูปแบบแถว โดยมีรูปแบบทางไวยกรณ์ดังนี้

ตาราง ค-3 แสดงไวยากรณ์คำสั่งของอนุประโยค INSERT INTO

```
INSERT INTO <table name>[(<column list>)]
    VALUES (<values list>) | <query>;
```

INSERT เป็นคำสั่งในการป้อนแถวใหม่แถวหนึ่งหรือหลายแถวใน *<table name>* ถ้าใช้อนุประโยค VALUES ก็ใส่ค่าเหล่านั้นลงไปใน *<table name >* ถ้ากำหนด *<query>* ไว้ แต่ละแถวของเอาท์พุทของการสอบถามข้อมูลนี้ก็จะใส่ลงไปใน *<table name>* ตามลำดับของคอลัมน์ที่ได้กำหนดไว้ ถ้าไม่มี *<column list>* คอลัมน์ทั้งหมดของ *<table name>* จะทำการเรียงลำดับตามที่สมมติให้

SELECT

คำสั่ง SELECT เป็นคำสั่งที่ใช้สำหรับเลือกดูข้อมูลในตารางที่ได้สร้างไว้ โดยมีรูปแบบไวยากรณ์คำสั่งดังต่อไปนี้

ตาราง ค-4 แสดงไวยากรณ์คำสั่งของอนุประโยค SELECT

```
SELECT | {[DISTINCT| ALL] <value expression>,} | *
    [INTO <host variable list>(*embeded only*)
        FROM <table reference>.,...
    [WHERE <predicate>]
        [GROUP BY <grouping column>,...]
    [HAVING <predicate>]
    [ORDER BY <ordering column > [ANSI|DESC],..];
```

คำสั่งนี้จะประกอบด้วยการสอบถามข้อมูลและค่าของเอาท์พุทจากฐานข้อมูลโดยใช้กฎและข้อกำหนดดังต่อไปนี้

ถ้าไม่มีการกำหนด ALL หรือ DISTINCT แล้วจะสมมติว่าใช้ ALL

<values expression> จะประกอบด้วย *<column spec>*, *<aggregate func>*, *<nonstandard function>*, *<constant>* หรืออนุประโยคดังกล่าวผสมกันพร้อมด้วยโอเปอเรเตอร์ต่างๆที่เป็นนิพจน์ที่ใช้ได้

<table reference> นั้นจะประกอบด้วยชื่อของตารางรวมทั้งชื่อของเจ้าของตาราง

ถ้าใช้ GROUP BY ค่าของ *<column spec>* ที่ใช้ในอนุประโยค SELECT จะต้องถูกใช้เป็นค่าของ *<grouping column>*

ถ้าใช้ HAVING แล้ว ทุกแถวของเอาท์พุทที่เกิดจากอนุประโยค GROUP BY จะใช้ *<predicate>* และแถวต่างๆที่ทำให้ *<predicate>* เป็นจริงเป็นเอาท์พุท

ถ้าใช้ ORDER BY เอาท์พุทจะมีการเรียงลำดับที่แน่นอน แต่ละ *<column identifier>* จะอ้างถึง *<value expression>* เฉพาะในอนุประโยคของ SELECT ถ้า *<value expression>* นั้นเป็น *<column spec >* แล้ว *<column identifier>* อาจเป็น *<column spec>* เดียวกันได้ มิฉะนั้นแล้ว *<column identifier >* จะเป็นจำนวนเต็มบวกแสดงตำแหน่งของ *<values expression>* ในลำดับของอนุประโยค SELECT เอาท์พุทจะถูกจัดวางตำแหน่งตามค่าที่ของ *<column identifier>* ตามลำดับจากน้อยไปหามาก นอกเสียจากว่าเราจะกำหนดเป็น DESC (จากมากไปหาน้อย)

<column identifier> ที่ให้ชื่อไว้ก่อนในอนุประโยค ORDER BY จะมีลำดับความสำคัญเหนือคอลัมน์ที่ให้ชื่อไว้ทีหลังในการพิจารณาลำดับของเอาท์พุท

ตาราง ค-5 แสดงองค์ประกอบและคำนิยามของคำสั่งของอนุประโยค SELECT

องค์ประกอบ	คำนิยาม
<value expression>	ประกอบด้วย *<column spec>*, *<aggregate func>*, *<nonstandard function>*, *<constant>* หรืออนุประโยคเหล่านี้ผสมกันพร้อมด้วยโอเปอเรเตอร์ต่างๆเป็นนิพจน์ที่ใช้ได้
<table name>	ชื่อหรือชื่อที่มีความหมายเหมือนของตารางหรือวิว
<alias>	ชื่อที่มีความหมายเหมือนชั่วคราวของ<table name>
<predicate>	เงื่อนไขที่เป็นจริงหรือเท็จของแต่ละแถวหรือการผสมกันของแถวต่างๆจากตาราง(ต่างๆ)ในอนุระโยค FROM
<grouping column>	ชื่อของคอลัมน์ที่ถูกเรียกโดยอนุประโยคย่อย GROUP BY
<ordering column>	ชื่อของคอลัมน์ที่ถูกเรียกโดยอนุประโยคย่อย ORDER BY
<table reference>.	ประกอบด้วยชื่อของตาราง รวมทั้งชื่อเจ้าของตารางเป็นชื่อนำหน้าถ้าผู้ใช้ในขณะนั้นไม่ได้เป็นเจ้าของ หรือชื่อที่มีความหมายเหมือนกัน(*nonstandard*)ของตารางนั้น

UPDATE

คำสั่ง UPDATE เป็นคำสั่งที่ใช้ในการเปลี่ยนค่าของข้อมูลต่างๆในแต่ละ *<column name>* ให้เป็น *<value expression>* ที่ตรงกัน ถ้าอนุประโยค WHERE ใช้ *<predicate>* เฉพาะแถวต่างๆในตารางทั้งหลาย ซึ่งค่าต่างๆของตารางเหล่านั้นทำให้ *<predicate>* นั้นเป็นจริงเท่านั้นจึงจะถูกเปลี่ยนค่าไปถ้า ส่วนคำสั่ง WHERE CURRENT OF นี้ใช้ได้ใน embeded SQL เท่านั้นและใช้กับเคอร์เซอร์ที่แก้ไขได้เท่านั้น ถ้าขาดอนุประโยค WHERE ไป แถวทั้งหมดจะถูกเปลี่ยนแปลง โดยมีรูปแบบไวยากรณ์คำสั่ง UPDATE ดังต่อไปนี้

ตาราง ค-6 แสดงไวยากรณ์คำสั่งของอนุประโยค UPDATE

```
UPDATE <table name>
    SET {<column name> = < values expression >}.,...
    {[WHERE <predicate>];}
    {[WHERE CURRENT OF <curcor name>] <SQL term>}
```

DELETE

เป็นคำสั่งที่ใช้ลบข้อมูลต่างๆในตาราง โดยมีรูปแบบของอนุประโยคดังต่อไปนี้

ตาราง ค-7 แสดงไวยากรณ์คำสั่งของอนุประโยค DELETE

```
DELETE FROM <table name>
    [WHERE <predicate>
    | WHERE CURRENT OF <cursor name>
        (*embeded only*)];
```

DROP

เป็นคำสั่งที่ใช้สำหรับลบตารางโดยที่ข้อมูลต่างๆก็สูญหายไปด้วย ซึ่งมีรูปแบบไวยากรณ์ของอนุประโยคดังนี้

ตาราง ค-8 แสดงไวยากรณ์คำสั่งของอนุประโยค DROP

DROP TABLE <*table name*>

ตาราง ค-9 แสดงสัญลักษณ์ต่างๆที่ใช้ในรูปแบบทางไวยากรณ์ของคำสั่ง DROP

สัญลักษณ์	คำอธิบาย
|	• สิ่งใดก็ตามที่อยู่ข้างหน้าสัญลักษณ์นี้อาจเลือกให้แทนด้วยสิ่งที่ตามหลังสัญลักษณ์นี้ได้ นี่เป็นวิธีพูดว่า "หรือ"ด้วยสัญลักษณ์นั่นเอง
{ }	• ทุกอย่างที่อยู่ภายในปีกกานี้จะได้รับการพิจารณาว่าเป็นหน่วยเดียวในการประเมินผล | , ,....หรือสัญลักษณ์อื่นๆ
[ ]	• ทุกอย่างที่อยู่ภายในวงเล็บก้ามปูนี้เป็นหนทางเลือก
...	• สิ่งใดก็ตามที่อยู่หน้าเครื่องหมายนี้อาจทำซ้ำกี่ครั้งก็ได้
,....	• สิ่งใดก็ตามที่อยู่ข้างหน้าสัญลักษณ์นี้อาจทำซ้ำกี่ครั้งก็ได้โดยมีจำนวนที่เกิดขึ้นแยกกันด้วยคอมม่า

โดยต่อไปจะเป็นตัวอย่างของการสร้างตารางเพื่อใช้ในการเก็บข้อมูล ซึ่งจะยกตัวอย่างการเก็บข้อมูลอุณหภูมิของจังหวัดนครสวรรค์ ประจำเดือนมกราคม ปี พศ. 2541 (ข้อมูลโดยกรมอุตุนิยมวิทยา)

ตาราง ค-10 แสดงตัวอย่างข้อมูลอุณหภูมิของจังหวัดนครสวรรค์ เดือนมกราคม พ.ศ. 2541

วันที่	เวลา							
	1	4	7	10	13	16	19	22
1/1/98	220	200	190	271	325	342	283	235
2/1/98	215	202	195	265	325	335	285	245
3/1/98	227	210	204	274	330	344	292	245
4/1/98	230	205	205	275	330	345	291	245
5/1/98	230	220	220	288	334	340	286	245
6/1/98	220	204	200	283	320	328	285	230
7/1/98	215	201	191	270	325	335	282	240
8/1/98	216	205	198	273	338	345	286	245
9/1/98	222	212	196	274	337	360	288	244
10/1/98	224	210	204	260	315	350	298	258
11/1/98	244	225	220	288	355	359	296	264
12/1/98	258	250	238	280	330	345	305	287
13/1/98	275	255	250	290	338	351	317	295
14/1/98	273	255	244	293	336	345	295	255
15/1/98	215	215	190	271	327	330	260	221
16/1/98	188	169	160	265	324	345	280	264
17/1/98	222	185	171	268	330	347	285	230
18/1/98	210	180	166	265	321	344	273	222
19/1/98	203	173	164	250	314	340	290	255
20/1/98	264	235	220	277	318	336	295	255
21/1/98	250	265	237	294	325	340	304	295
22/1/98	277	255	255	294	335	345	315	295
23/1/98	275	255	252	295	345	348	307	266
24/1/98	255	245	240	290	330	352	308	285
25/1/98	288	260	255	237	287	294	270	245

26/1/98	230	225	215	275	320	335	295	256
27/1/98	240	230	222	300	345	353	308	267
28/1/98	241	227	220	307	350	357	314	270
29/1/98	260	230	223	268	315	328	285	235
30/1/98	215	200	193	282	325	340	300	250
31/1/98	235	210	207	292	346	355	314	265

เมื่อเราต้องการทำการจัดเก็บข้อมูลในรูปของตาราง เราต้องทำการกำหนดการเก็บค่าตัวแปรของแต่ละหลักและแถวของตารางเสียก่อน คือ

ชื่อตาราง คือ temp_cc

วัน เดือน ปี แทนด้วย dmy เก็บในรูปของ dd-mmm-yyyy เช่น 30/5/98 ก็เป็น 30-May-1998 โดยกำหนด ชนิดข้อมูล (data type) และขนาด (size) เป็น varchar2(15)

ค่าตัวเลขของอุณหภูมิราย 3 ชั่วโมง เป็น HR1 ถึง HR8 โดยกำหนดชนิดข้อมูล(data type)และขนาด (size) เป็น NUMBER(5)

จากหลักไวยากรณ์ข้างต้นเปิดโปรแกรม Oracle SQL WorkSheet

```
CREATE TABLE temp_cc
        (dmy            varchar2(15),
         hr1             number(5),
         hr2             number(5),
         hr3             number(5),
         hr4             number(5),
         hr5             number(5),
         hr6             number(5),
         hr7             number(5),
         hr8             number(5));
```

รันโปรแกรม Oracle SQL WorkSheet

```
SQLWKS>     CREATE TABLE   temp_cc
     2> (dmy              varchar2(15),
     3> hr1               number(5),
     4> hr2               number(5),
     5> hr3               number(5),
     6> hr4               number(5),
     7> hr5               number(5),
     8> hr6               number(5),
     9> hr7               number(5),
    10> hr8               number(5));
Statement processed.
```

```
SQLWKS> select * from temp_cc
     2>
```

DMY	HR1	HR2	HR3	HR4	HR5	HR6	HR7	HR8

0 rows selected.

จากนั้นก็นำข้อมูลที่ได้เก็บในตารางโดยใช้คำสั่ง INSERT

```
insert into temp_cc values('1-Oct-1997' ,250 , 250 , 250 ,314 , 325 ,328 ,292 , 276 );
insert into temp_cc values('2-Oct-1997' ,269 , 260 , 260 ,305 ,330 ,330 ,300 ,280 );
insert into temp_cc values('3-Oct-1997' ,274 , 270 , 264 ,305 , 320 ,333 ,306 ,250 );
insert into temp_cc values('4-Oct-1997' ,250 , 247 , 245 ,290 ,306 ,320 , 290 ,270 );
insert into temp_cc values('5-Oct-1997' ,265 , 250 , 245 ,248 , 265 ,284 ,270 ,257 );
insert into temp_cc values('6-Oct-1997' ,237 , 245 , 244 ,275 , 298 ,298 ,275 ,268 );
insert into temp_cc values('7-Oct-1997' ,255 , 245 , 250 ,284 , 315 ,320 ,288 ,267 );
insert into temp_cc values('8-Oct-1997' ,257 , 250 , 252 ,300 , 330 ,305 ,290 ,270 );
insert into temp_cc values('9-Oct-1997' ,255 , 252 , 255 , 320 , 340 ,343 ,308 ,278 );
insert into temp_cc values('10-Oct-1997' ,270 , 262 , 260,310 , 320 ,318 ,295 ,284 );
```

เสร็จแล้วก็รันโปรแกรม ออราเคิล เอสคิวแอล เวิร์คชีต (Oracle SQL WorkSheet)

```
SQLWKS> insert into temp_cc  values('1-Oct-1997',250,250,250,314,325,328,292,276);
1 row processed.
SQLWKS> insert into  temp_cc  values('2-Oct-1997'269,260,260,305,330,330,300,280);
1 row processed.
SQLWKS> insert into  temp_cc  values('3-Oct1997',274,270,264,305,320,333,306,250);
1 row processed.
SQLWKS> insert into  temp_cc  values('4-Oct1997',250,247,245,290,306,320,290,270);
1 row processed.
SQLWKS> insert into  temp_cc  values('5-Oct1997',265,250,245,248,265,284,270,257);
1 row processed.
SQLWKS> insert into  temp_cc  values('6-Oct1997',237,245,244,275,298,298,275,268);
1 row processed.
SQLWKS> insert into  temp_cc  values('7-Oct1997',255,245,250,284,315,320,288,267);
1 row processed.
SQLWKS> insert into  temp_cc  values('8-Oct1997',257,250,252,300,330,305,290,270);
1 row processed.
SQLWKS> insert into  temp_cc  values('9-Oct1997',255,252,255,320,340,343,308,278);
1 row processed.
SQLWKS> insert into temp_cc values('10-Oct1997',270,262,260,310,320,318,295,284);
1 row processed.
```

หลังจากนั้นลองตรวจสอบดูว่ามีอยู่ในตารางหรือไม่ โดยใช้คำสั่ง SELECT

```
SQLWKS> select * from temp_cc
  2>
```

DMY	HR1	HR2	HR3	HR4	HR5	HR6	HR7	HR8
1-Oct-1997	250	250	250	314	325	328	292	276
2-Oct-1997	269	260	260	305	330	330	300	280
3-Oct-1997	274	270	264	305	320	333	306	250
4-Oct-1997	250	247	245	290	306	320	290	270
5-Oct-1997	265	250	245	248	265	284	270	257
6-Oct-1997	237	245	244	275	298	298	275	268

7-Oct-1997	255	245	250	284	315	320	288	267
8-Oct-1997	257	250	252	300	330	305	290	270
9-Oct-1997	255	252	255	320	340	343	308	278
10-Oct-1997	270	262	260	310	320	318	295	284

10 rows selected.

จากนั้นก็ป้อนข้อมูลที่เหลือซึ่งข้อมูลทั้งหมดของ temp_cc แล้วลองเลือกดูข้อมูลในวัน เดือน ปี ต่างๆ ได้ตามที่ต้องการ เช่น

```
SQLWKS> select * from  temp_cc where month='January'
2>
```

DMY	HR1	HR2	HR3	HR4	HR5	HR6	HR7	HR8
1-Jan-1998	220	200	190	271	325	342	283	235
2-Jan-1998	215	202	195	265	325	335	285	245
3-Jan-1998	227	210	204	274	330	344	292	245
4-Jan-1998	230	205	205	275	330	345	291	245
5-Jan-1998	230	220	220	288	334	340	286	245
6-Jan-1998	220	204	200	283	320	328	285	230
7-Jan-1998	215	201	191	270	325	335	282	240
8-Jan-1998	216	205	198	273	338	345	286	245
9-Jan-1998	222	212	196	274	337	360	288	244
10-Jan-1998	224	210	204	260	315	350	298	258
11-Jan-1998	244	225	220	288	355	359	296	264
12-Jan-1998	258	250	238	280	330	345	305	287
13-Jan-1998	275	255	250	290	338	351	317	295
14-Jan-1998	273	255	244	293	336	345	295	255
15-Jan-1998	215	215	190	271	327	330	260	221
16-Jan-1998	188	169	160	265	324	345	280	264
17-Jan-1998	222	185	171	268	330	347	285	230

18-Jan-1998	210	180	166	265	321	344	273	222
19-Jan-1998	203	173	164	250	314	340	290	255
20-Jan-1998	264	235	220	277	318	336	295	255
21-Jan-1998	250	265	237	294	325	340	304	295
22-Jan-1998	277	255	255	294	335	345	315	295
23-Jan-1998	275	255	252	295	345	348	307	266
24-Jan-1998	255	245	240	290	330	352	308	285
25-Jan-1998	288	260	255	237	287	294	270	245
26-Jan-1998	230	225	215	275	320	335	295	256
27-Jan-1998	240	230	222	300	345	353	308	267
28-Jan-1998	241	227	220	307	350	357	314	270
29-Jan-1998	260	230	223	268	315	328	285	235
30-Jan-1998	215	200	193	282	325	340	300	250
31-Jan-1998	235	210	207	292	346	355	314	265

31 rows selected.

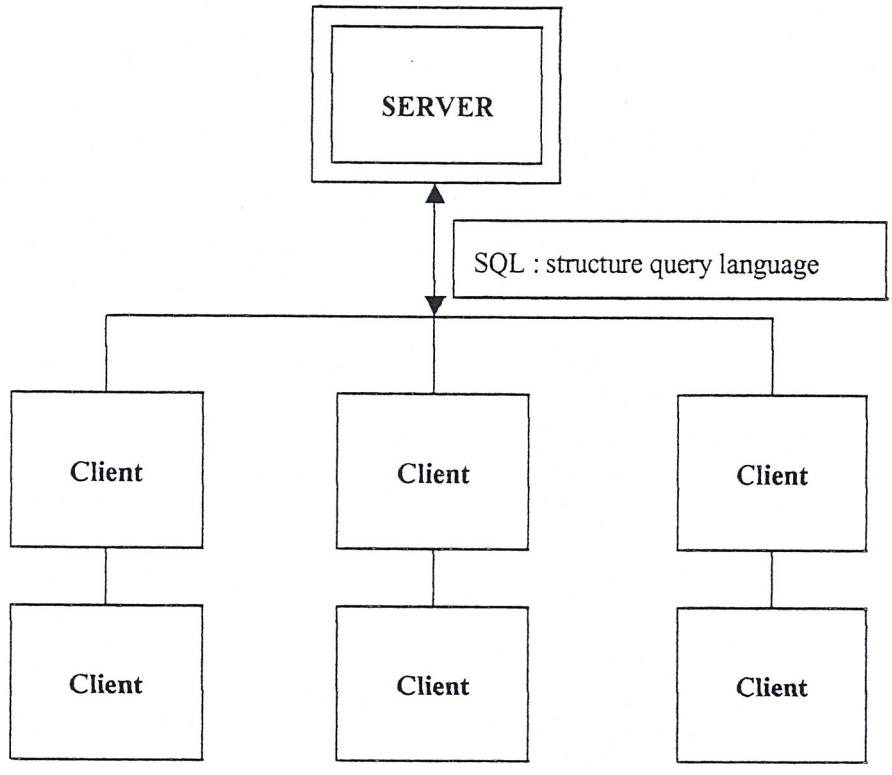
ข้อมูลที่เหลือก็ทำในลักษณะเดียวกันกับ temp_cc(นครสวรรค์), temp_bk(กรุงเทพฯ), temp_ee (ระยอง), temp_ne(ขอนแก่น), temp_nn(เชียงใหม่), temp_se(สงขลา) และ temp_we(กาญจนบุรี) รวมทั้งข้อมูลโหลดและความชื้นสัมพัทธ์

ข้อมูลที่เก็บใน ออราเคิล นี้จะนำไปใช้ในการป้อนโปรแกรมเพื่อทำนายการใช้ไฟฟ้าในช่วงเวลาสั้นๆ (Short-Term LoadForecasting) เมื่อได้ผลจากการทำนายแล้วก็จะนำมาเก็บในออราเคิล เพื่อเป็นฐานข้อมูลที่จะใช้ในการเปรียบเทียบผลการทำนายในครั้งต่อๆไป

3.) การติดต่อกับระบบฐานข้อมูลออราเคิล (Oracle)

การติดต่อกับระบบฐานข้อมูลออราเคิลสามารถทำการติดต่อได้หลายทางและมีรูปแบบการติดต่อหลายรูปแบบ แต่รูปแบบที่ใช้ในการทำโครงงานครั้งนี้ คือ การติดต่อในลักษณะเซิร์ฟเวอร์-ไคลเอนท์ (Server-Client) โดยใช้เครื่องที่มีระบบปฏิบัติการ(OS.) วินโดว์เอ็นทีเซิร์ฟเวอร์ (Windows-NT Server) และวินโดว์เอ็นทีไคลเอนท์ (Windows-NT Client) โดยโปรแกรมออราเคิลที่ติดตั้งในแต่ละเครื่องก็จะมีลักษณะและการติดตั้งที่แตกต่างกันออกไปตามเครื่องที่ทำการติดตั้ง กล่าวคือ ทางด้านเครื่องเซิร์ฟเวอร์ก็จะติดตั้งโปรแกรมในลักษณะเอนเทอร์ไพร์ส (Enterprise) และเครื่องไคลเอนท์จะติดตั้งโปรแกรมในลักษณะไคลเอนท์ (Client) ธรรมดา ซึ่งข้อแตกต่างระหว่างการติดตั้งแบบเอน-

เทอร์ไพร์สกับการติดตั้งแบบไคลเอนท์ก็คือ เครื่องเซิร์ฟเวอร์ที่ได้ทำการติดตั้งแบบเอนเทอร์ไพร์สจะใช้เป็นระบบจัดการฐานข้อมูลที่ได้รับการพัฒนาจากเครื่องเซิร์ฟเวอร์เองหรือจากเครื่องไคลเอนท์แต่ฐานข้อมูลได้รับการออกแบบจะสามารถเก็บได้ในเครื่องเซิร์ฟเวอร์เท่านั้น ส่วนที่ทำการติดตั้งโปรแกรมออราเคิลแบบไคลเอ็นจะไม่สามารถเก็บระบบฐานข้อมูลในเครื่องได้ แต่จะสามารถพัฒนาข้อมูลเข้าสู่เครื่องเซิร์ฟเวอร์ได้โดยตรง



รูปที่ ค.1 แสดงการติดต่อระหว่าง Windows-NT Server – Windows-NT Client ของโปรแกรมออราเคิล

3.1) จุดประสงค์ในการติดต่อกับระบบฐานข้อมูล

การติดต่อกับระบบฐานข้อมูลมีสองจุดประสงค์หลักคือ

3.1.1) เพื่อทำการบันทึกหรือเปลี่ยนแปลงข้อมูลต่างๆในระบบฐานข้อมูล

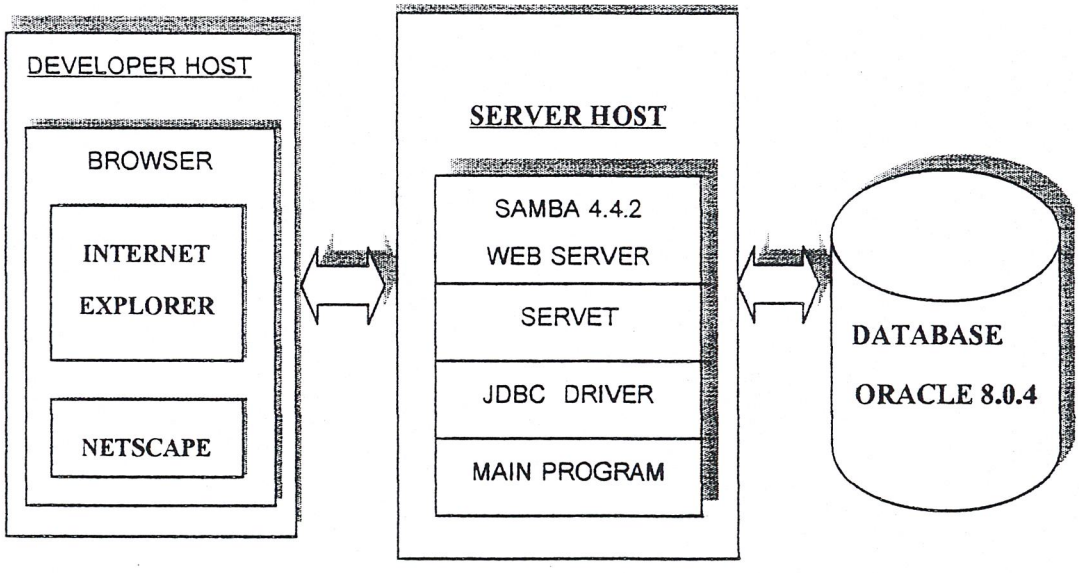
3.1.2) เพื่อนำข้อมูลในฐานข้อมูลมาใช้ในการประมวลผลในส่วนของโปรแกรมหลัก

การติดต่อในสองลักษณะข้างต้นจะมีการติดต่อในลักษณะและรูปแบบของการติดต่อที่แตกต่างกันออกไปแล้วแต่จุดประสงค์ในการติดต่อ

การติดต่อกับฐานข้อมูลเพื่อทำการบันทึกหรือเปลี่ยนแปลงฐานข้อมูล

สามารถที่จะติดต่อได้โดยใช้ภาษาเอสคิวแอล (SQL : Structure Query Language) ผ่านทางเครื่องไคลเอนท์หรือเครื่องเซิร์ฟเวอร์ โดยสามารถทำการสร้างระบบฐานข้อมูลใหม่, การนำข้อมูลเข้าไปเก็บในระบบฐานข้อมูล, การเปลี่ยนแปลงข้อมูลหรือเพิ่มเติมระบบฐานข้อมูลรวมทั้งสามารถกำหนดความสามารถในการเข้าถึงระบบฐานข้อมูลข้อผู้ใช้แต่ละบุคคล รวมทั้งติดตั้งระบบความปลอดภัย (Security) ให้กับข้อมูลด้วยโดยใช้เครื่องมือช่วย (Wizard) ร่วมด้วย การติดต่อทั้งหมดจะสามารถติดต่อเข้าสู่ระบบฐานข้อมูลได้โดยตรงซึ่งเป็นวิธีที่ใช้สร้างระบบฐานข้อมูลเริ่มต้น

การติดต่อกับระบบฐานข้อมูลอีกรูปแบบหนึ่งก็คือ การใช้ความสามารถด้านอินเตอร์เน็ต (Internet) เข้าช่วยในการติดต่อกับระบบฐานข้อมูล โดยการเขียนเว็บไซต์ (Web Site) ขึ้นมาแล้วให้กรอกข้อมูลที่จำเป็นที่ต้องใช้ในการประมวลผลโปรแกรมเก็บไว้ กล่าวคือ จะทำการสร้างฐานข้อมูลเพื่อรองรับการกรอกข้อมูลผ่านทางเว็บไซต์



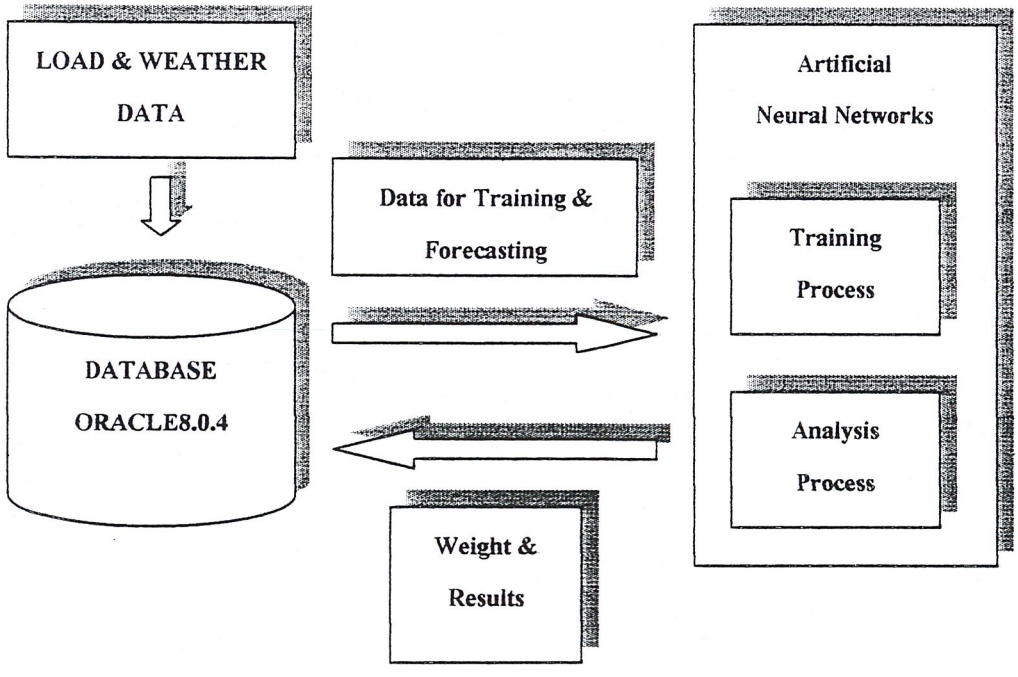
รูปที่ ค.2 แสดงการติดต่อกับระบบฐานข้อมูลโดยใช้ความสามารถทางอินเตอร์เน็ต

การติดต่อกับระบบฐานข้อมูลโดยใช้ความสามารถทางอินเตอร์เน็ต มีหลักการทำงานคือ ที่เครื่องเซิร์ฟเวอร์จะต้องมีเว็บเซิร์ฟเวอร์ (Web Server) เพื่อรองรับการให้บริการด้านเว็บเพจ (Web page) ซึ่งในโครงงานครั้งนี้ได้ใช้โปรแกรมแซมบาเว็บเซิร์ฟเวอร์ (Samba Web Server version 4.4.2) โดยจะเขียนเว็บเพจด้วยภาษาเอชทีเอ็มแอล (HTML: HyperText Markup Language) ไว้รองรับการกรอกข้อมูลเมื่อมีการกรอกข้อมูลบนเว็บเพจ ข้อมูลนั้นจะถูกส่งผ่านเข้าไปเก็บไปยังระบบฐานข้อมูลโดยการสนับสนุนของเซิร์ฟเล็ท (Servlet) เข้าช่วย

จะเห็นว่าการใช้ความสามารถทางอินเตอร์เน็ตเข้าช่วยจะทำให้การรับข้อมูลทำได้ง่ายขึ้น กล่าวคือเราสามารถกรอกข้อมูลเข้าสู่ระบบฐานข้อมูลได้จากเครื่องคอมพิวเตอร์ทั่วโลกที่มีบราวเซอร์ (Browser)ไม่ว่าจะเป็นเน็ตสเคป (Netscape) หรืออินเตอร์เน็ตเอ็กซ์พลอเรอร์ (Internet Explorer)

การติดต่อกับระบบฐานข้อมูลเพื่อนำข้อมูลในระบบฐานข้อมูลมาใช้ในการประมวลผล

ในการประมวลผลของตัวโปรแกรมหลัก จำเป็นต้องใช้ข้อมูลจากระบบฐานข้อมูล จึงต้องการการติดต่อเพื่อนำข้อมูลจากระบบฐานข้อมูลมาใช้ โดยในการติดต่อดังกล่าวจะใช้ไดร์ฟเวอร์ช่วยในการติดต่อ (ในที่นี้ใช้ JDBC Driver) รูปแบบภาษาที่เขียนเพื่อใช้ในการติดต่อจะต้องเป็นภาษาจาวาเช่นเดียวกัน



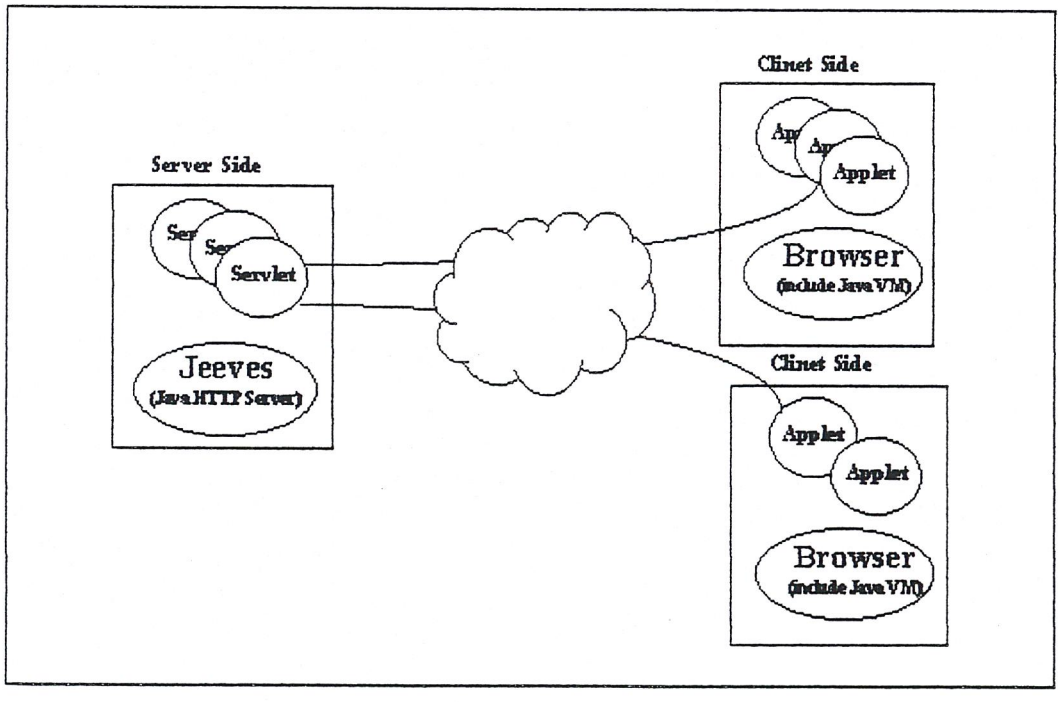
รูปที่ ค.3 แสดงการติดต่อกับระบบฐานข้อมูลเพื่อนำข้อมูลมาใช้ในการโปรแกรมหลัก

จากรูปแบบการติดต่อกับฐานข้อมูลจะเห็นว่า ในการติดต่อจะใช้โปรแกรมช่วยในการติดต่อโปรแกรมหลักๆ คือ เซิร์ฟเล็ท (Servlet)

3.2) โปรแกรมการติดต่อฐานข้อมูล (Java Servlet)

เซิร์ฟเล็ท (Servlet) คือโปรแกรมจาวา (Java Program) ที่ถูกสร้างมาเพื่อให้ทำงานได้บนเอชทีทีพีเซิร์ฟเวอร์ (HTTP Server) ซึ่งมีประโยชน์ในการควบคุมให้เอชทีทีพีเซิร์ฟเวอร์ทำงานหรือโต้ตอบกับโปรแกรมของเครื่องไคลเอนท์ ซึ่งมีลักษณะคล้ายการทำงานของซีจีไอ (CGI.: Common Gateway Interface)

จาวาเซิร์ฟเล็ท จะทำงานบนจาวาเอชทีทีพีเซิร์ฟเวอร์ (Java HTTP Server) ของเครื่องเซิร์ฟเวอร์ (Server) คล้ายกับที่จาวาแอพเพล็ท (Java Applet) ทำงานบนบราวเซอร์ (Browser) ของเครื่องไคลเอนท์จาวา (Client Java) เอชทีทีพีเซิร์ฟเวอร์ในปัจจุบันที่มีอยู่ก็ชื่อจีฟ (Jeeves) ซึ่งพัฒนาขึ้นโดยบริษัทซัน (SUN) และกำลังมีของบริษัท 3th Parties พัฒนาตามออกมา



รูปที่ ค.4 แสดงการทำงานของเซิร์ฟเล็ท

ตาราง ค-11 เปรียบเทียบความแตกต่างระหว่างจาวาเซิร์ฟเล็ทกับจาวาแอพเพ็ท

Java Servlet	Java Applets
ทำงานบน Java HTTP Server (เช่น Jeeves)	ทำงานบน Browser (Client)
ไม่มีส่วนติดต่อกับผู้ใช้โดยตรง(User Interface)	มี AWT สำหรับทำส่วนติดต่อกับผู้ใช้
อนุญาติให้สามารถ access local file และ Network ได้	ไม่อนุญาติให้สามารถ access local file และ Network ได้ นอกจากติดต่อกลับไปหาผู้เรียก

เนื่องจากจาวาเซิร์ฟเล็ทถูกสร้างมาเพื่อจะให้แทนที่ซีจีไอจึงมีจุดเด่นหลายๆอย่าง ที่ซีจีไอไม่มีหรือด้อยกว่า อาทิเช่น

1) เซิร์ฟเล็ทสามารถที่จะฝังตัวทำงานอยู่บนเครื่องเซิร์ฟเวอร์เมื่อทำการให้บริการไคลเอนท์เสร็จสิ้นแล้ว เพื่อจะคอยรับคำสั่ง (Request) ใหม่ที่อาจจะมีมาจากเครื่องไคลเอนท์อีก โดยไม่ต้องถูกสร้างขึ้นมาใหม่และตายไปทุกๆครั้งที่มีคำสั่งจากไคลเอนท์เหมือนซีจีไอสคริปท์ (CGI. Script) ทำให้ เซิร์ฟเล็ทรบกวนทรัพยากรของเครื่องเซิร์ฟเวอร์น้อยกว่าซีจีไอสคริปท์
2) เซิร์ฟเล็ทสามารถจัดการกับงานประเภทการติดต่อแบบหลายทาง (Multiple Connection) ได้ดี โดยอาศัยความสามารถของเทรด (Thread) ที่มีอยู่ในตัวจาวา ตัวอย่างเช่น การประกาศข่าว (Broadcast) ข้อมูลไปทุกการติดต่อ (Connection) พร้อมๆกัน
3) เซิร์ฟเล็ทสามารถติดต่อกับแอพเพล็ทบนเครื่องไคลเอนท์ได้อย่างต่อเนื่อง ทำให้สามารถทำการรับส่งข้อมูลระหว่างกันอย่างต่อเนื่อง แต่ทำได้ไม่สะดวกกับซีจีไอสคริปท์ การรับส่งข้อมูลอย่างต่อเนื่องต้องอาศัยการบันทึกเหตุการณ์ (State) ก่อนหน้าเอาไว้เนื่องจากซีจีไอสคริปท์จะสิ้นสุดเมื่อให้บริการหนึ่งคำสั่งเท่านั้น
4) เซิร์ฟเล็ทสามารถถูกพัฒนาหรือเปลี่ยนแปลง (Upload) จากเครื่องไคลเอนท์ เพื่อสั่งให้ไปทำงานบนเครื่องเซิร์ฟเวอร์ใดๆบนระบบเครือข่ายได้ ประโยชน์เช่น เราสามารถเขียนโปรแกรมเพื่อค้นหาข้อมูลในเครื่องให้บริการเครือข่าย (Wep Host) ใดๆ โดยไม่ต้องส่งข้อมูลมาประมวลที่เครื่องต้นทางแต่สามารถทำได้ที่ตัวเครือข่ายต้นทาง (Host) นั้นๆเลย เสร็จแล้วค่อยส่งผลกลับคืนจึงช่วยลดเวลาและการติดขัดของข้อมูล (Traffi) บนระบบเครือข่ายลงอย่างมาก

ตาราง ค-12 เปรียบเทียบความแตกต่างระหว่างจาวาเซิร์ฟเล็ทกับซีจีไอ

	Java Servlet	CGI
ภาษาที่ใช้	JAVA	ภาษาทั่วๆไปเช่น Perl, C
การเรียกโปรแกรม	โดยการเรียก Methode Call	โดยการ รันโปรแกรม
การส่งผ่านค่า Parameters	ผ่าน ServletRequest methodes	ผ่าน Evironment Variables
ระบบความปลอดภัย	JAVA Security Manager	OS นั้นๆ
ความสิ้นเปลืองทรัพยากรบน Server	น้อย	สูง

3.3) การติดต่อกับฐานข้อมูลโดยใช้เจดีบีซี (JDBC)

เจดีบีซี (JDBC : Java Database Connectivity) คือจาจาเอพีไอ (Java API) ที่ใช้สำหรับตีความคำสั่งภาษาเอสคิวแอล โดยประกอบด้วยชุดของคลาส (Classes) และอินเตอร์เฟส (Interfaces)ที่เขียนด้วยภาษาจาวา ตัวเจดีบีซีนี้เป็นมาตรฐานสำหรับการพัฒนาระบบฐานข้อมูลและทำให้มีความเป็นไปได้ในการเขียนโปรแกรมใช้งานเกี่ยวกับฐานข้อมูลด้วยจาวาเอพีไอเพียงอย่างเดียว

การใช้เจดีบีซีนี้ทำได้ง่ายในการส่งคำสั่งด้วยภาษาเอสคิวแอลให้กับระบบฐานข้อมูลต่างๆในรูปแบบเดียวกัน นั่นคือเจดีบีซีเอพีไอจะไม่บังคับให้เขียนโปรแกรมตัวหนึ่งเพื่อใช้กับฐานข้อมูลของไซส์เบส (Sysbase), อีกโปรแกรมหนึ่งสำหรับข้อมูลของออราเคิล, อีกหนึ่งโปรแกรมสำหรับฐานข้อมูลของอินฟอร์มิกส์ (Informix) และอีกหลายโปรแกรมสำหรับฐานข้อมูลชนิดต่างๆ เราสามารถเขียนโปรแกรมโดยใช้จาวาเอพีไอเพียงโปรแกรมเดียวและสามารถส่งคำสั่งเอสคิวแอลไปยังฐานข้อมูลชนิดต่างๆนั้นได้ การเขียนโปรแกรมโดยใช้ภาษาจาวานี้ไม่ต้องกังวลว่าจะต้องเขียนโปรแกรมอีกหลายตัวเพื่อนำไปใช้กับคอมพิวเตอร์ที่อยู่บนระบบปฏิบัติการชนิดต่างๆ ในการรวมตัวกันระหว่างจาวาและเจดีบีซีนี้ ทำให้ผู้เขียนโปรแกรมสามารถเขียนโปรแกรมเพียงหนึ่งโปรแกรมแล้วสามารถนำไปใช้ได้ในทุกๆแห่งที่ต้องการ

จาวาเป็นโปรแกรมที่กำลังได้รับความนิยมสูง ทั้งนี้เนื่องจากมีความปลอดภัยในภาษา, ความง่ายต่อการใช้ และสามารถถ่ายโอนข้อมูลได้อย่างอัตโนมัติบนระบบเครือข่าย จาวาเป็นโปรแกรมภาษาที่มีความฉลาด เราจึงนำมาใช้ในการเขียนโปรแกรมในการจัดการระบบฐานข้อมูล และเจดีบีซี

ก็เป็นทางเลือกที่ต้องการเพื่อที่จะให้โปรแกรมตัวนั้นสามารถติดต่อกับฐานข้อมูล ไม่ว่าฐานข้อมูลชนิดนั้นจะเป็นชนิดใดก็ตาม

เจดีบีซีเป็นส่วนขยายของความสามารถของจาวา นั่นคือเมื่อใช้จาวาและเจดีบีซีเอพีไอ เราสามารถเผยแพร่เว็บเพจที่มีจาวาแอพเพลตที่ใช้ข้อมูลจากฐานข้อมูลที่อยู่ที่แห่งอื่นได้ อาทิเช่น ในบริษัทต่างๆก็สามารถใช้เจดีบีซีเพื่อติดต่อพนักงานเข้ากับฐานข้อมูลต่างๆไม่ว่าพวกเขาจะใช้เครื่องบนระบบปฏิบัติการวินโดวส์ (Windows), แมคอินทอชท์ (Macintosh) หรือยูนิกซ์ (UNIX) ก็ตาม บนระบบอินทราเน็ตและในขณะนี้ความต้องการของการเข้าถึงฐานข้อมูลที่ง่ายจากจาวากำลังมีเพิ่มสูงขึ้นเรื่อยๆกับผู้เขียนโปรแกรมจาวาจำนวนมาก

ผู้ที่จัดการเอ็มไอเอส (MIS) ชอบที่จะรวมจาวาและเจดีบีซีเข้าด้วยกันเพราะทำให้การเผยแพร่ข้อมูลทำได้ง่ายและประหยัด ในบริษัทสามารถใช้ฐานข้อมูลที่ได้ติดตั้งไว้ก่อนแล้วและการเข้าถึงข้อมูลก็ทำได้ง่ายแม้ว่ามันจะอยู่ในระบบจัดการฐานข้อมูลต่างชนิดกันก็ตาม ในการพัฒนาโปรแกรมตัวใหม่จะใช้เวลาน้อยลง, การติดตั้งและการดูแลเรื่องเวอร์ชันก็ง่ายมาก โดยผู้เขียนโปรแกรมสามารถเขียนหรือแก้ไขโปรแกรมเพียงโปรแกรมเดียวแล้วนำไปไว้ที่เครื่องเซิร์ฟเวอร์ (Server) จากนั้นทุกๆคนก็จะสามารถเข้าถึงข้อมูลแบบเวอร์ชันล่าสุดได้ทันที สำหรับธุรกิจบริการข้อมูล การใช้จาวาและเจดีบีซีเป็นหนทางที่ดีที่สุดที่จะนำเสนอข้อมูลที่แก้ไขล่าสุดให้กับลูกค้าข้างนอกของตน

โดยการใช้ทั่วไปเจดีบีซีสามารถทำได้ 3 อย่างดังนี้

1) สร้างการเชื่อมต่อกับฐานข้อมูล
2) ส่งคำสั่งเอสคิวแอล (SQL)
3) จัดการกับผลลัพธ์ที่ได้มา

ตัวอย่างโปรแกรมแสดงการทำงานทั้ง 3 อย่างข้างต้น

```
Connection con = DriverManager.getConnection (
                        "jdbc:odbc:wombat", "login", "password");
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery("SELECT a, b, c FROM Table1");
while (rs.next()) {
    int x = getInt("a");
    String s = getString("b");
    float f = getFloat("c");
}
```

การรักษาความปลอดภัยของเจดีบีซีผู้พัฒนาเจดีบีซีนั้นควรจะพิจารณา

1) ความน่าเชื่อถือของโปรแกรม ความน่าเชื่อถือของโปรแกรมจาวา (Java program) เป็นไปตามจาวาแอพพลิเคชัน (Java application) และจาวาแอพเพล็ท (Java applet) จากแหล่งข้อมูลที่เกี่ยวข้องนั้นความจะเป็นเซิร์ฟเวอร์ (Server) ที่เก็บรายระเอียดของการให้บริการของบริษัทของคุณ หรือ เซิร์ฟเวอร์หลักที่แอพเพล็ท (Applet) สามารถที่จะถูกกำหนดด้วยรหัสลับที่แน่ใจได้ว่าสามารถที่จะดาวน์โหลด (Download) ได้อย่างสมบูรณ์
2) ความไม่น่าเชื่อถือของโปรแกรม ความไม่น่าเชื่อถือของโปรแกรมนั้นคือจาวาแอพเพลตที่เข้าได้ทางอินเตอร์เน็ต ซึ่งความไม่น่าเชื่อถือของแอพเพล็ทก็คือ จะต้องไม่ยอมให้เครื่องลูกข่าย (Client) เข้าถึงเช่นแฟ้มข้อมูลของระบบ (File system) ได้เป็นต้น ในอนาคตความไม่น่าเชื่อถือของแอพเพล็ท เป็นเพียงแต่การยอมรับการเชื่อมโยงได้กับระบบเครือข่าย (Networks) กลับไปยังเซิร์ฟเวอร์จากการดาวน์โหลด (Download)

การใช้งานเจดีบีซีและตั้งชื่อฐานข้อมูลบนระบบเครือข่าย

ทางจาวาซอร์ฟ (Javasoft) นั้นได้ออกแบบการใช้งานมาตรฐานการตั้งชื่อไว้เป็นแบบยูอาแอล (URL:Uniform Resource Location) รูปแบบประโยคพื้นฐานที่จะอธิบายถึงฐานข้อมูลก็คือ jdbc::jdbc:mSQL://161.246.24.245:8080/mydatabase

ตัวอย่างฐานข้อมูล mydatabase บนโฮส (Host) 161.246.24.245,port 8080, โดยใช้ mSQL สำหรับการเข้าถึงทางระบบเครือข่าย (Networks)

การประมวลผลการค้นหาข้อมูลและการได้มาซึ่งผลลัพธ์ เมื่อได้ผลของการค้นหาฐานข้อมูลโดยเอสคิวแอลจากการทำงานของ java.sql.Statement class แล้วหลังจากการสถาปนาการเชื่อมโยงฐานข้อมูลจึงสร้างประโยค (Statement) และประมวลผล

ตัวอย่าง โปรแกรมแสดงการทำงานของเจดีบีซี

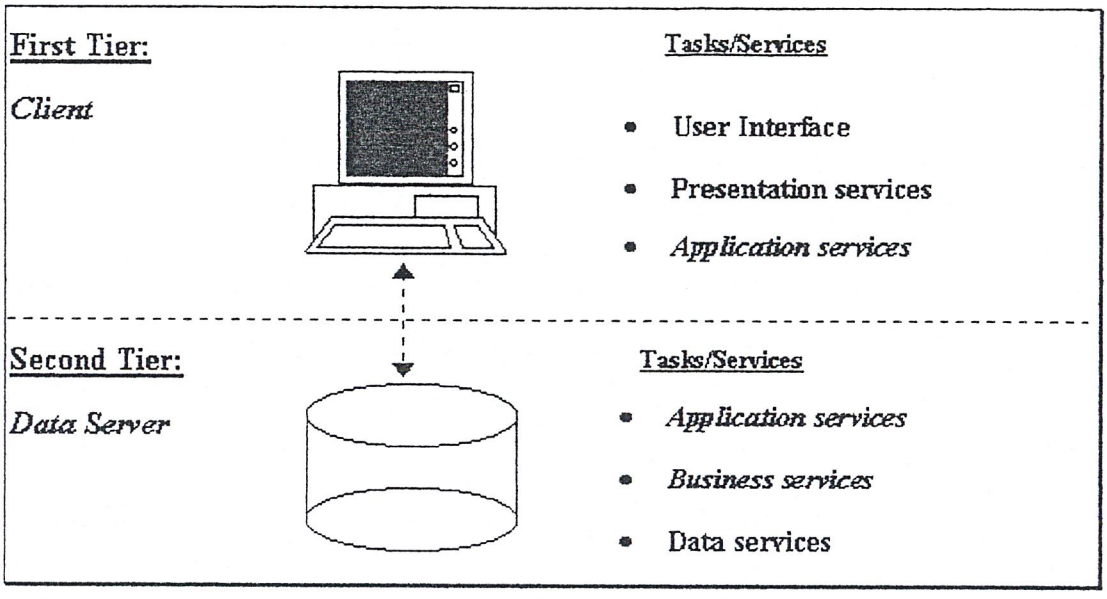
```
java.sql.Statement stmt = conn.creatStatement();
ResultSet r = stmt.executeQuery("SELECT a,b,c FROM Table1");
ResultSet object r
while (r.next) {
        int I = r.getInt("a");
        String s = r.getString("b");
        byte b[] = r.getByte("c");
        System.out.println("ROW = "+ I + " " + s + " " + b[0]); }
```

(จะได้รับข้อมูลจากการค้นหาไว้ การกำหนดค่าให้จาก Field ใน r โดยใช้JDBC getxxx)

การใช้งานแอพพลิเคชันเมื่อทำงานในระบบไคลเอนท์-เซิร์ฟเวอร์ (Client/Server) นั้นสามารถเลือกใช้ได้หลายรูปแบบขึ้นอยู่กับลักษณะเฉพาะของงานแต่ละชนิดและขนาดของงานที่เลือกใช้ งานบางชนิดจำเป็นต้องติดต่อฐานข้อมูลขนาดใหญ่ แต่บางชนิดไม่จำเป็นต้องใช้ฐานข้อมูลขนาดใหญ่ องค์ประกอบเหล่านี้มีส่วนเกี่ยวข้องกับประสิทธิภาพของแอพพลิเคชั่นและผลลัพธ์ของงานที่ได้มา มีวิธีการแก้ไขปัญหาโดยแบ่งแอพพลิเคชั่นออกเป็นโมดูลย่อยให้ทำงานร่วมกัน โดยที่แต่ละโมดูลก็จะทำงานตามหน้าที่ของตนเป็นอิสระจากโมดูลอื่นๆ ทำให้ง่ายต่อการเปลี่ยนแปลงแก้ไขและบำรุงรักษา เรียกการทำงานแบบนี้ว่า "มัลติเทียร์" ซึ่งพัฒนามาจาก "ทู-เทียร์" และใช้วิธีการเข้าถึงฐานข้อมูลโดยใช้ เจดีบีซีไดร์ฟเวอร์ (Java Database Connectivity Driver) ซึ่งแบ่งตามลักษณะการทำงานออกได้เป็น 4 แบบ ขึ้นอยู่กับว่าผู้ใช้จะเลือกใช้งานอย่างไรเพื่อประสิทธิภาพที่ดีที่สุด

การทำงานแบบไคลเอนท์-เซิร์ฟเวอร์ (Client/Server)

สถาปัตยกรรมของแอพพลิเคชั่นแบบไคลเอนท์-เซิร์ฟเวอร์ จะแบ่งการประมวลผลออกเป็นสองโปรแกรม โดยทั่วไปจะทำงานบนเครื่องสองเครื่องขึ้นไป แอพพลิเคชันที่ทำงานกับฐานข้อมูลแบบไคลเอนท์-เซิร์ฟเวอร์ จะรับผิดชอบในการจัดเก็บข้อมูล, การประมวลผลข้อมูลและการโอนย้ายข้อมูลไปแสดงผลที่อื่น โดยเครื่องเซิร์ฟเวอร์จะเป็นตัวเก็บรวบรวมข้อมูลไว้ ส่วนเครื่องไคลเอนท์จะประมวลผลข้อมูลที่ได้มาหรือสร้างเป็นข้อมูลใหม่ ซึ่งวิธีการทำงานโดยใช้สถาปัตยกรรมแบบไคลเอนท์-เซิร์ฟเวอร์นี้ทำให้สามารถติดต่อใช้งานข้อมูลได้จากผู้ใช้หลายแห่ง



รูปที่ ค.5 แสดงการทำงานไคลเอนท์-เซิร์ฟเวอร์ แบบ ทูเทียร์ (Two-Tier Application)

รูปแบบธรรมดาทั่วไปของสถาปัตยกรรมไคลเอนท์-เซิร์ฟเวอร์ เป็นทูเทียร์ (two-tier) ซึ่งมาจากการแบ่งการทำงานของแอพพลิเคชั่นออกเป็นส่วนไคลเอนท์-เซิร์ฟเวอร์ ซึ่งยอมรับการติดต่อจากหลายๆที่เข้าคู่ส่วนให้บริการซึ่งเก็บข้อมูลไว้ ส่วนแสดงผลจะอยู่ที่ไคลเอนท์และส่วนเก็บรวบรวมข้อมูลจะอยู่ที่เซิร์ฟเวอร์ แอพพลิเคชันทั่วไปส่วนใหญ่บนอินเตอร์เน็ต เช่น จดหมายอิเล็กทรอนิกส์ (E-mail), การขอใช้บริการเครือข่าย (Telnet), การเคลื่อนย้ายข้อมูล (Ftp), การขอใช้การบริการระบบสืบค้นข็อมูล (Gopher) หรือเวบเพจ (Web page) เป็นแอพพลิเคชั่นแบบ 2 ระดับซึ่งทำงานโดยไม่ต้องประมวลข้อมูลขนาดใหญ่ โดยทั่วไปจะทำงานติดต่อใช้ข้อมูลภายในอินเตอร์เน็ต

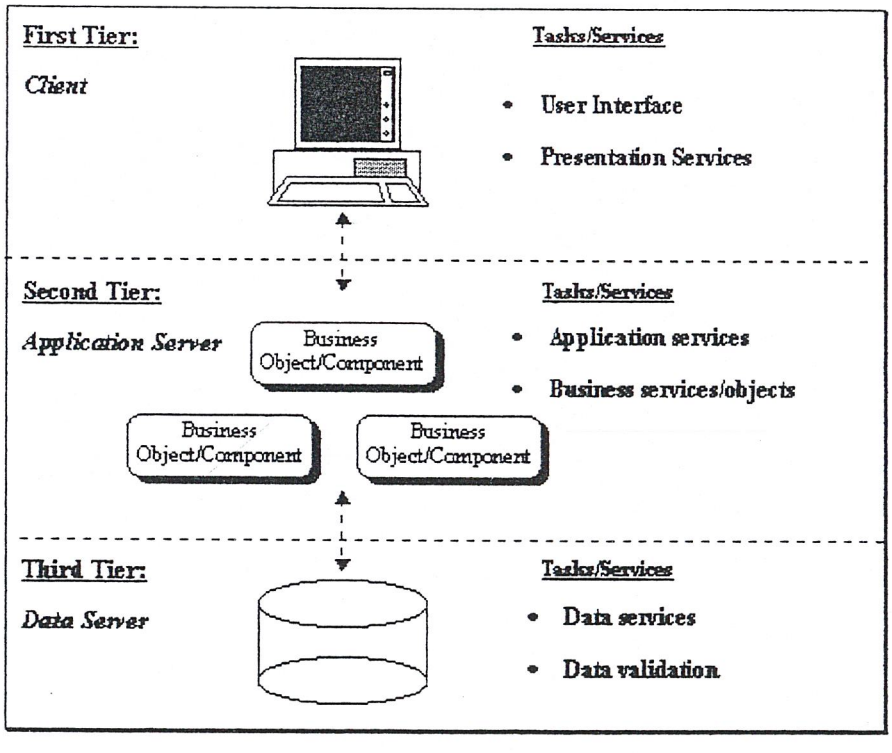
ข้อดีของแอพพลิเคชั่นแบบทูเทียร์ คือ เป็นแอพพลิเคชั่นง่ายๆธรรมดาที่ไม่ต้องการการดูแลบำรุงรักษามาก สามารถพิจารณาเลือกใช้ว่าเหมาะกับแอพพลิชั่นแบบทูเทียร์หรือไม่ควรขึ้นกับเงื่อนไขดังนี้

- เป็นแอพพลิเคชันที่ใช้ ฐานข้อมูลเดี่ยว
- ฐานข้อมูลบรรจุอยู่ภายในหน่วยประมวลผลกลาง (CPU) เครื่องเดียว
- ฐานข้อมูลมีขนาดเดิมไม่เปลี่ยนแปลงบ่อยๆ
- ผู้ใช้ (User Base) ไม่มีการเปลี่ยนแปลงบ่อย
- ความต้องการ (Requirement) ไม่มีการเปลี่ยนแปลง หรือ เปลี่ยนแปลงน้อยมาก
- แอพพลิเคชันที่สมบูรณ์แล้ว ไม่จำเป็นต้องดูแลบำรุงรักษามาก

ข้อเสียของทูเทียร์

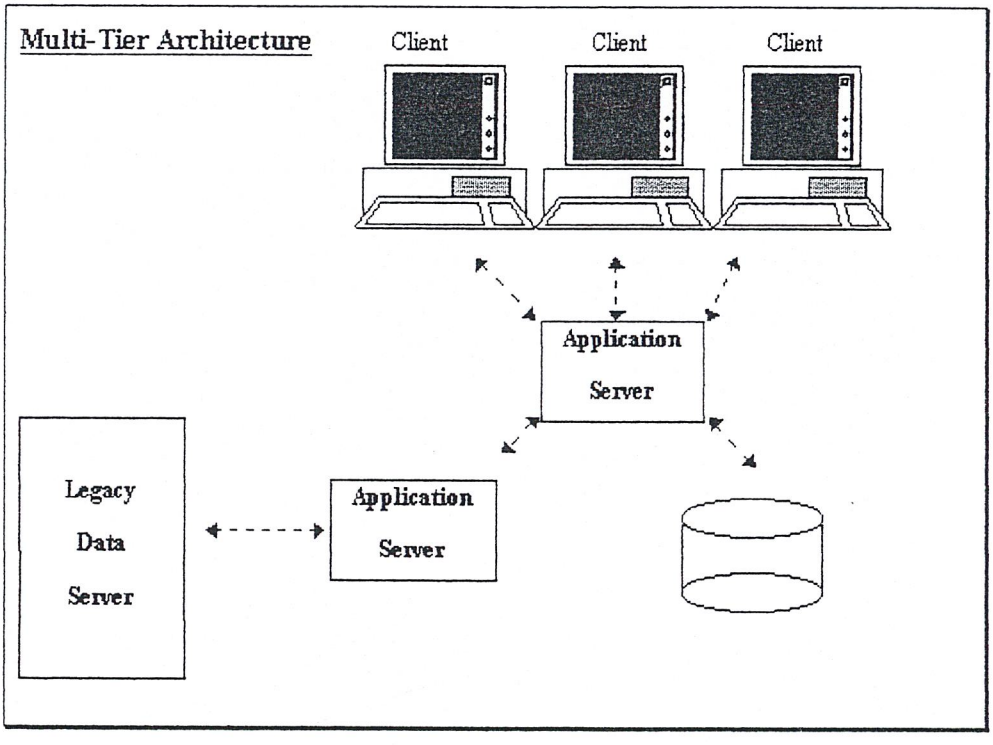
ความต้องการของผู้ใช้เพิ่มมากขึ้นดังนั้นความซับซ้อนของแอพพลิเคชันจึงต้องมากตามไปด้วย จากการที่เครื่องไคลเอนท์มีประสิทธิภาพและมีความซับซ้อนขึ้นเรื่อยๆ ในขณะที่เครื่องเซิร์ฟเวอร์มีขนาดเล็กลงเพื่อให้ราคาถูกลงและความสามารถในการจัดการกับฐานข้อมูลที่ซับซ้อนต่ำลง เช่น ในปัจจุบันเครื่องคอมพิวเตอร์เมนเฟรมได้ถูกเปลี่ยนมาใช้เครื่องคอมพิวเตอร์ขนาดเล็กจำนวนมากมาทำงานแทนและงานบางส่วนจะถูกผลักภาระไปที่เครื่องไคลเอนท์ เพื่อเป็นการลดค่าใช้จ่าย แต่การทำเช่นนี้ทำให้เกิดปัญหาขึ้น

"fat client" เครื่องไคลเอนท์ที่มีปัญหา fat client นี้เกิดจากการที่ไคลเอนท์ไม่สามารถรองรับขนาดของข้อมูลและงานของผู้ใช้ที่มีจำนวนมากขึ้นได้ เพราะว่างานของเครื่องไคลเอนท์ไม่ได้มีแค่แสดงข้อมูลให้เห็นเท่านั้นแต่ยังมีการดึงข้อมูลอื่นๆจำนวนมากที่ไม่เกี่ยวข้องเลย กับงานนั้นๆ มาด้วย และในกรณีที่มีการเปลี่ยนแปลงฟังก์ชันการทำงานบางส่วน ผู้ใช้จำเป็นต้องมีการเปลี่ยนแปลง, ทดสอบและแจกจ่ายโปรแกรมในส่วนของไคลเอนท์ที่ได้ปรับปรุงแล้วไปยังไคลเอนท์ทุกเครื่อง



รูปที่ ค.6 แสดงการทำงานไคลเอนท์-เซิร์ฟเวอร์ แบบ ทรีเทียร์ (Three-Tier Application)

เพื่อแก้ปัญหาของทูเทียร์เราจึงเพิ่มจากทูเทียร์เป็นทรีเทียร์ โดยในแบบทูเทียร์เดิมไคลเอนท์จะติดต่อโดยตรงกับฐานข้อมูล หากมีการเปลี่ยนแปลงใดๆเกิดขึ้นในฐานข้อมูลการแสดงผลทางด้านไคลเอนท์จำเป็นต้องเปลี่ยนแปลงตามไปด้วย ในการแก้ปัญหานี้เราจะเพิ่มเทียร์ใหม่เข้ามาขั้นระหว่างเครื่องไคลเอนท์และเครื่องเซิร์ฟเวอร์ โดยไคลเอนท์จะติดต่อกับเซิร์ฟเวอร์โดยผ่านทางออบเจ็คที่อยู่บนมิดเดิลเทียร์ จากนั้นมิดเดิลเทียร์จะติดต่อกับเซิร์ฟเวอร์ โดยไคลเอนท์จะเห็นเฉพาะออบเจ็คในมิดเดิลเทียร์เท่านั้น การเปลี่ยนแปลงใดๆ จะต้องทำผ่านมิดเดิลเทียร์เท่านั้น



รูปที่ ค.7 แสดงการทำงานไคลเอนท์-เซิร์ฟเวอร์ แบบ มัลติเทียร์ (Multi-Tier Application)

โปรแกรมแอพพลิเคชันโดยทั่วไปที่ใช้งานอยู่ จะประกอบด้วยส่วนที่ติดต่อกับผู้ใช้ (userInterface) สำหรับแสดงผลและเก็บรวบรวมข้อมูลเข้ามา กลุ่มของฟังก์ชันต่างๆที่ทำหน้าที่ประมวลผลข้อมูลและแบ่งงานต่างๆรวมถึงวิธีการเก็บรักษาข้อมูล ถึงแม้ว่าฟังก์ชันที่ใช้ในการเก็บรักษาข้อมูลโดยทั่วไปจะทำงานอยู่ภายใต้เครื่องเซิร์ฟเวอร์ของฐานข้อมูลส่วนกลาง บางครั้งเราเรียกรูปแบบลักษณะการทำงานแบบนี้ว่า เป็นโมเดลแอพพลิเคชันแบบ 2 ระดับ (two-tier application model), ซึ่งโปรแกรมแอพพลิเคชันแบบเก่า คือ จะเป็นโปรแกรมเดียวซึ่งทำงานบนเครื่องของผู้ใช้

เนื่องจากโปรแกรมแอพพลิเคชันที่ทำงานเดี่ยวๆนั้นมีขนาดใหญ่มากจึงพัฒนาได้ช้า, และบำรุงรักษามากอีกทั้งยังใช้เนื้อที่ฮาร์ดดิสค์ (Hard disk) สูงมาก เพียงแค่มีการเปลี่ยนแปลงเล็กน้อยก็จะต้องมีการเรขียนโปรแกรมทับลงไปใหม่, คอมไพล์ใหม่ และเนื่องจากโปรแกรมแอพพลิเคชันเหล่านี้เขียนขึ้นมาเพื่อใช้งานกับระบบที่มีลักษณะต่างกันจึงไม่สามารถที่จะเปลี่ยนไปใช้งานบนระบบที่แตกต่างไปได้วิธีการแก้ปัญหาดังกล่าวมาทำได้โดยการแบ่งโปรแกรมแอพพลิเคชันเดี่ยวๆนี้ออกเป็นโมดูลย่อยๆที่ทำงานร่วมกัน การแยกส่วนที่ติดต่อกับผู้ใช้ออกมาจากฟังก์ชันอื่นๆในโปรแกรมแอพพลิเคชันทำให้เราสามารถสร้างโปรแกรมไคลเอนท์เล็กๆซึ่งไม่ซับซ้อนและไม่ต้องทำงานมากเกินไปบนเครื่องของผู้ใช้โดยในโมดูลนี้จะใช้ฮาร์ดดิสค์บนเครื่องผู้ใช้น้อยกว่า และสามารถพัฒนา, บำรุงรักษาได้ง่ายกว่า ตัวอย่างเช่น ส่วนที่ติดต่อกับผู้ใช้สามารถเปลี่ยนแปลงได้โดยไม่ต้องเขียนโปรแกรมแอพพลิเคชั่นใหม่ทั้งหมด ในทางทฤษฎีแล้วส่วนโมดูลที่ติดต่อกับผู้ใช้นี้สามารถเขียนได้โดยใช้ภาษาที่ต่างกัน เช่น จาวา จึงทำให้ใช้ได้บนเครื่องที่แตกต่างกันไคลเอนท์โมดูลออกแบบโดยใช้จาวาแอพเพลตจึงไม่ต้องการเนื้อที่บนฮาร์ดดิสค์เพื่อการติดตั้ง java IDL สนับสนุนซอฟท์แวร์ที่ออกแบบเป็นโมดูลแต่ละโมดูลออกแบบให้รองรับแอพพลิเคชันได้หลายแบบแบ่งลักษณะของโมดูลดังกล่าวได้ 3 ประเภทใหญ่ๆ คือ

1. ยูสเซอร์อินเตอร์เฟสไคลเอนท์เทียร์ (User – Interface Client Tier)

ส่วนที่ติดต่อกับผู้ใช้ในโมเดลแอพพลิเคชันแบบหลายระดับรวมไปถึงส่วนที่ติดต่อกับผู้ใช้แบบกราฟฟิก (GUI-Graphics User Interface) สำหรับแอพพลิเคชันทั้งแบบดั้งเดิมและแบบพื้นฐานสร้างมาเพื่อให้ทำงานกับผู้ใช้ได้เร่งและได้ผลลัพธ์ที่ถูกต้อง โดยรวมฟังก์ชันที่ทำงานเกี่ยวกับกราฟฟิกออกจากส่วนที่ให้บริการทางเครือข่าย การทำงานได้สอดคล้องกันของส่วนที่ติดต่อกับผู้ใช้จะช่วยลดปัญหาในการเรียนรู้เพื่อใช้งานแอพพลิเคชั่นไหม่ๆ, ทำงานร่วมกันกับแอพพลิเคชันได้ดีและให้ผลลัพธ์ที่มีคุณภาพสูงขึ้นแอพพลิเคชันแบบกราฟฟิกนี้สามารถใช้ได้สำหรับงานทั่วไปของผู้ใช้ เช่น บนเครื่องในระบบเครือข่าย, หรือบนอินเตอร์เน็ต

2. เซิร์ฟเวอร์เทียร์ (Server Tier)

ส่วนที่ให้บริการหรือส่วนเซิร์ฟเวอร์นี้เป็นส่วนสำคัญของแอพพลิเคชัน เป็นส่วนกลางซึ่งคอยให้บริการการใช้แอพพลิชันและการสร้างแอพพลิเคชัน ซึ่งการให้บริการนี้มีอยู่ในเครือข่าย และสามารถเข้าใช้ได้จากแอพพลิเคชันทุกระดับ

3. ดาตาสตอร์เทียร์ (Data Store (Database) Tier)

แอพพลิเคชันแบบหลายระดับ (Multi-tier) นี้จะแยกการติดต่อเข้าใช้ข้อมูลออกมาจากส่วนเซิร์ฟเวอร์ และเรียกส่วนที่แยกออกมานี้ว่า "data store tier " มีออปชั่น (Option) หลายแบบที่ใช้เก็บและติดต่อใช้ข้อมูลเพื่อช่วยให้ผู้พิจารณาสามารถเลือกใช้กลุ่มข้อมูลที่มีความสำคัญเป็นอันดับแรกสุด ออบเจกที่สร้างขึ้นมาจะใช้เพื่อสนองความต้องการในการใช้ข้อมูลต่างๆประกอบด้วย ความ

สามารถในการเรียกใช้ข้อมูลในอาร์ดีเอ็มเอส (RDBMS: Relation Database Management System) หรือ โอ โอดีบีเอ็มเอส (OODBMS: Object Oriented Database Management Systems)

ชนิดของ JDBC Driver

เจดีบีซีไดร์ฟเวอร์ (JDBC drivers) ที่เราทราบในเวลานี้มีอยู่ 4 แบบด้วยกันคือ

1) JDBC/ODBC bridge.

2.) Native-API, partly Java driver.

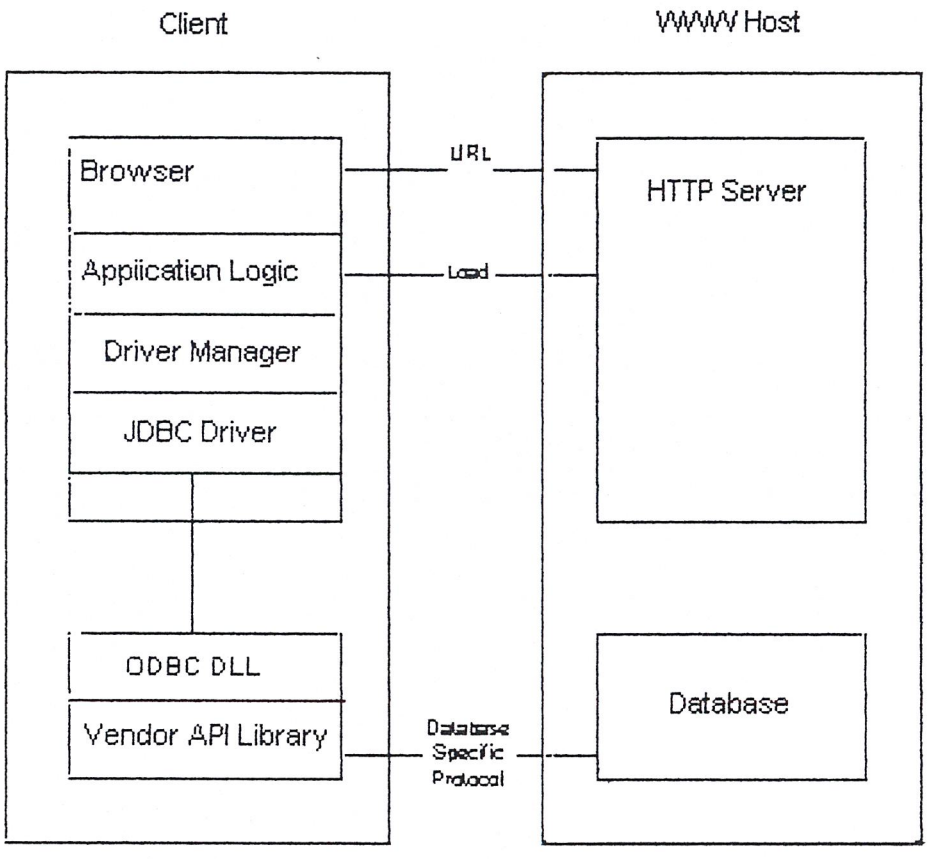
3.) Network-protocol, all-Java driver.

4.) Native-protocol, all-Java driver.

1 JDBC/ODBC bridge

JDBC/ODBC bridge ได้ถูกพัฒนาขึ้นโดยการร่วมมือกันระหว่างจาวาซอร์ฟ (JavaSoft) และอินเตอร์โซลว์ (Intersolv) เพื่อที่จะเพิ่มความสามารถให้กับฐานข้อมูลจำนวนมากที่ใช้โอดีบีซี (ODBC) ในส่วนของไคลเอนท์, จาวาแอพเพล็ทหรือโปรแกรมอื่นจะถูกเขียนขึ้นด้วยเจดีบีซีเอพีไอ โดยบริดจ์ (Bridge) ตัวนี้จะทำการแปลงคำสั่งจากเจดีบีซีไปเป็นคำสั่งของโอดีบีซีแล้วส่งคำสั่งนั้นไปยังตัวโอดีบีซีไดร์ฟเวอร์ (ODBC driver) เพื่อจัดการกับฐานข้อมูล ข้อได้เปรียบหลักของบริดจ์ตัวนี้คือ โปรแกรมที่เขียนขึ้นนั้นจะง่ายต่อการเข้าถึงข้อมูลจากระบบฐานข้อมูลของผู้ผลิตต่างๆ โดยการเลือกโอดีบีซีไดร์ฟเวอร์ที่เหมาะสม แต่อย่างไรก็ตาม ตัวติดต่อกับฐานข้อมูลชนิดนี้ทำให้ต้องนึกถึงผลข้างเคียงที่อาจเกิดขึ้นและความยุ่งยาก เพราะว่าคำสั่งต้องส่งจากเจดีบีซีไปยังบริดจ์ที่เชื่อมไปยังโอดีบีซีไดร์ฟเวอร์และสุดท้ายจากโอดีบีซีก็ส่งไปยังเนทีพไคลเอนเอพีไอ

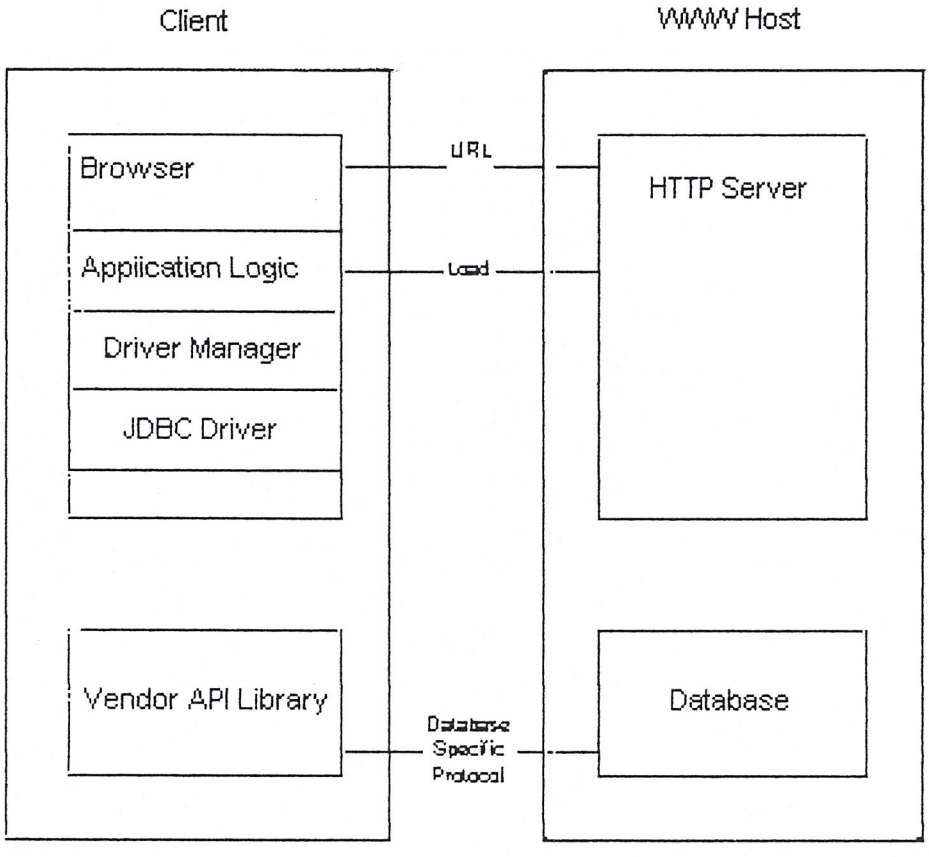
ตัวไดร์ฟเวอร์ชนิดนี้ทำให้จาวาแอพเพลตส่งข้อมูลไปไม่ทันทีทันใดกับที่เรียกไป เพราะระหัสคำสั่งหลักๆ (Code) ต้องถูกติดตั้งไว้ก่อนแล้วบนทุกๆเครื่องของไคลเอนท์ที่ต้องการใช้การเชื่อมต่อด้วย JDBC/ODBC bridge เพื่อช่วยเหลือคำสั่งของตัวเอพีไอจากการที่ต้องติดตั้งโปรแกรมก่อนนั้นทำให้การจัดการงานด้านการติดต่อของไคลเอนท์เซร์ฟเวอร์เป็นภาระที่หนักมาก ดังนั้น JDBC/ODBC Bridge จึงไม่ได้แก้ปัญหาการเปลี่ยนแปลงของไคลเอนท์โปรแกรม (Client program)



รูปที่ ค.8 แสดงการติดต่อโดยใช้ ไดรฟ์เวอร์ Type I.

2 Native-API, Partly Java Driver

เป็นไดรฟ์เวอร์แบบทูเทียร์นั่นคือ เจดีบีซีไดร์ฟเวอร์ต้องการไลบรารี (Library) เพื่อแปลงฟังก์ชันของเจดีบีซีไปเป็นภาษาของระบบฐานข้อมูลต่างๆ (Query language) ของ DBMS (เช่น library สำหรับ Sybase คือ dblib, สำหรับ Oracle คือ ocilib และอื่นๆ) ไดร์ฟเวอร์เหล่านี้โดยปกติจะเขียนขึ้นโดยภาษา Java และ C/C++ เนื่องจากไดร์ฟเวอร์ต้องใช้เส้นใย (Layer) ของ C ในการเรียกไปยังไลบรารีไดร์ฟเวอร์ชนิดที่ 2 เช่น JDBC/ODBC Bridge ต้องการให้ภาษาที่ใช้เขียน (Code ซึ่งคือ library ของแต่ละผู้ผลิต) ติดตั้งบนแต่ละ client ดังนั้นจึงมีปัญหาทางด้านการดูแลซอร์ฟแวร์เช่นเดียวกับ bridge อย่างไรก็ตามไดร์ฟเวอร์ชนิดที่ 2 นี้จะเร็วกว่าชนิดที่ 1 เพราะเส้นใยพิเศษของการแปลงเป็นโอดีบีซีถูกเอาออกไป



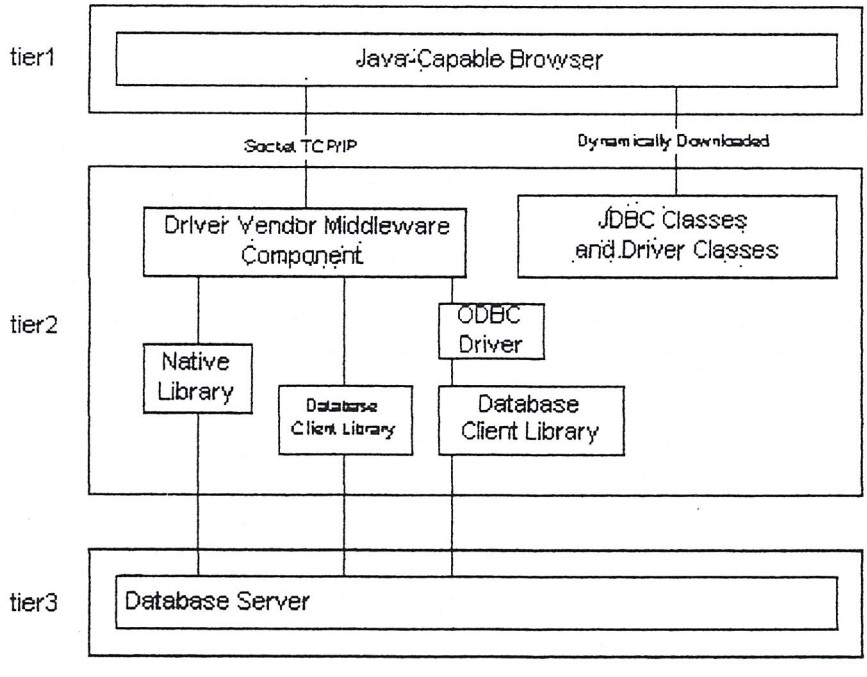
รูปที่ ค.9 แสดงภาพแสดงการติดต่อโดยใช้ ไดรฟ์เวอร์ Type II.

3 Network-Protocol , AII-Java Driver

ไดร์ฟเวอร์ชนิดที่สามนี้แปลงการเรียกเจดีบีซีไปเป็นโปรโตคอลเครือข่ายฐานข้อมูลอิสระ ซึ่งจะแปลงไปเป็นการเรียกดาตาเบสสเป็คสิฟิกเอพีไอ (Database-specific API) โดยมิดเดิลเทียร์เซิร์ฟเวอร์ (Middle-tier server) อาจใช้ไดร์ฟเวอร์ชนิดที่ 1 หรือชนิดที่ 2 ถ้าเขียนโดยจาวาสถาปัตยกรรมโดยรวมประกอบด้วยทรีเทียร์ คือ JDBC client และ driver, middleware และฐานข้อมูลที่ถูกเข้าถึงเจดีบีซีไดร์ฟเวอร์ขนาดเล็ก (200 KB หรือน้อยกว่า) ทำงานบนไคลเอนท์และ มีการใช้ตรรกะ (Logic) ในการส่งผ่านคำสั่งเอสคิวแอลในเครือข่ายไปยังเจดีบีซีเซิร์ฟเวอร์, รับข้อมูลกลับจากเซิร์ฟเวอร์และจัดการการติดต่อ โดยไดร์ฟเวอร์ชนิดที่สามนี้จะมีลักษณะเป็นการติดต่อเพียงชั่วขณะ (Just-in-time client deployment)

เจดีบีซีเซิร์ฟเวอร์จัดการการติดต่อหลายอย่างกับฐานข้อมูลรวมทั้งการยกเว้นและสถานะของเหตุการณ์ที่มีผลมาจากการทำคำสั่งเอสคิวแอล เจดีบีซีเซิร์ฟเวอร์ยังจัดรูปแบบของข้อมูลสำหรับการส่งในเครือข่ายไปยังเจดีบีซีไคลเอนท์

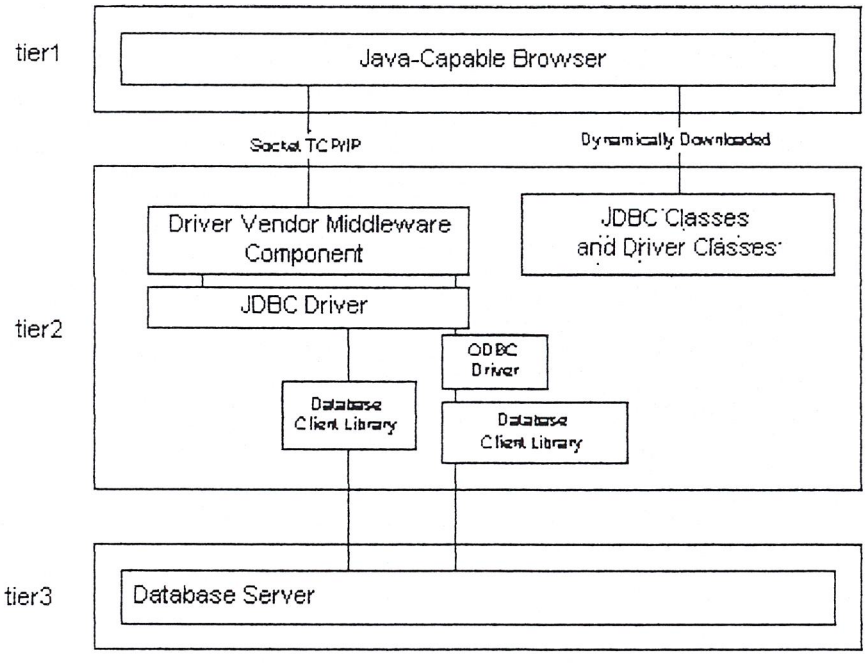
มิดเดิลแวร์เซิร์ฟเวอร์สามารถอิมพลีเมนต์ (Implement) เป็นส่วนประกอบดั้งเดิมหรือเขียนโดยจาวา การใช้แบบดั้งเดิมติดต่อกับเซิร์ฟเวอร์ฐานข้อมูลใช้ไคลเอนท์ไลบราลี (Client library) ของผู้ผลิตหรือโอดีบีซี เช่น dbAnywhere ของซิมแมนเทค (Symantec) และซีคิวลิงค์ (SequeLink) ของอินโฟซอร์ฟ (Intersoft) ถึงแม้ว่าซีคิวลิงค์ไม่ต้องการไคลเอนท์ไลบราลีของฐานข้อมูลติดตั้งบนเซิร์ฟเวอร์แต่จะใช้ไลบราลีของตัวเอง เซิร์ฟเวอร์จะต้องตั้งค่าสำหรับฐานข้อมูลที่จะเข้าถึงซึ่งอาจเกี่ยวข้องกับการตั้งเลขพอร์ต, ตัวแปรสภาพแวดล้อมต่างๆของฐานข้อมูล (เช่น DSQuery กับ Sybase), พารามิเตอร์เฉพาะของฐานข้อมูล (การ log, การแปลง) และพารามิเตอร์อื่นๆที่เซิร์ฟเวอร์ต้องการ ถ้ามิดเดิลแวร์เขียนโดยจาวา, จะสามารถใช้เจดีบีซีคอมเพลนท์เซิร์ฟเวอร์ (JDBC-compliant server) ในการสื่อสารกับ DBMS โดยผ่านโปรโตคอลฐานข้อมูลของผู้ผลิต



รูปที่ ค.10 แสดงการติดต่อโดยใช้ไดรฟ์เวอร์ JDBC แบบ three-tierType III

Type III driver เหมาะสำหรับแอพพลิเคชันที่มีหลายผู้ใช้บนอินเตอร์เน็ต / อินทราเน็ตมากที่สุด ที่ซึ่งการกระทำของข้อมูลต่อเนื่องจำนวนมาก เช่น queries, searches และอื่นๆถูกคาดหมายประสิทธิภาพเป็นสิ่งสำคัญ Server สามารถจัดการฐานข้อมูลจำนวนมากรวมกันได้, สามารถให้การตรวจสอบและดูแลข้อมูล, สามารถทำโหลดบาลานซิ่ง (Load balancing) และสนับสนุนรูปแบบ (Catalog) และคูอะรี่แคช (query caches) และอย่างที่ได้กล่าวไปแล้ว แอบพลิเคชันฐานข้อมูลบนเว็บเพจแบบทรีเทียร์เกี่ยวข้องกับความปลอดภัย, ไฟร์วอล (Firewalls) และพรอกซี่ (Proxies) ซึ่ง Type III driver สนับสนุนสิ่งเหล่านี้

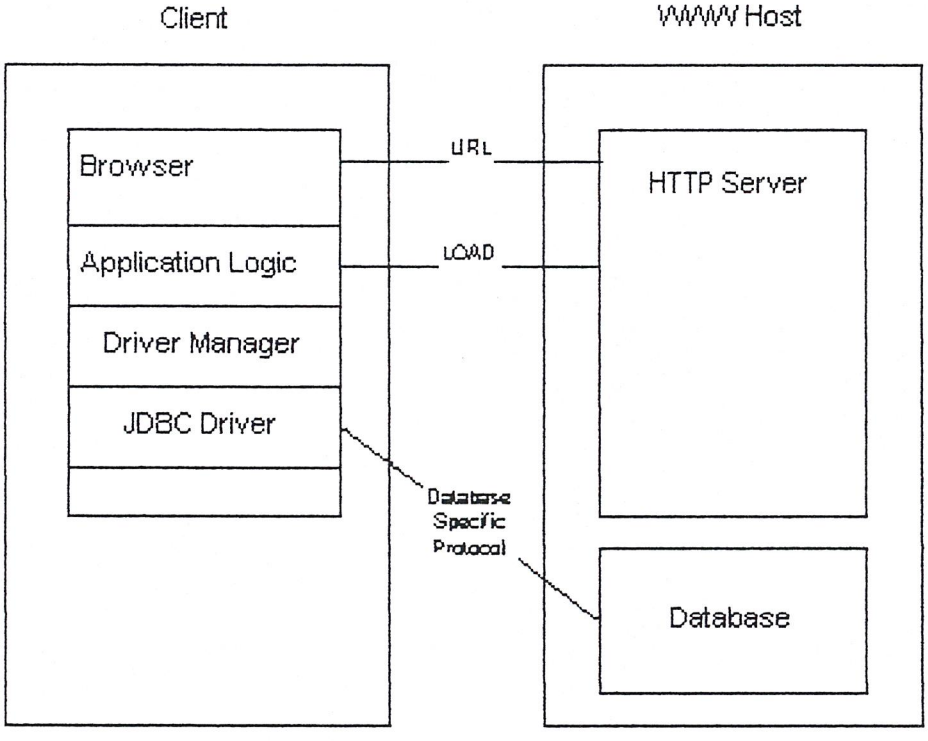
สิ่งที่เป็นข้อเสียของเน็ตเวิร์คเซ็นทริคไดร์ฟเวอร์ (Network-centric driver) ก็คือส่วนประกอบของเซิร์ฟเวอร์เป็นมิดเดิลแวร์ ผู้ผลิตแต่ละรายใช้มิดเดิลแวร์ของตนเองสำหรับการติดต่อในเครือข่าย



รูปที่ ค.11 แสดงการติดต่อโดยใช้ไดรฟ์เวอร์ JDBC แบบ three-tier Type III (Weblogic)

4 Native-Protocol , All-Java Driver (Type IV)

จะแปลงการเรียกเจดีบีซีไปเป็นโปรโตคอลเครือข่ายโดยตรงโดยใช้ไดร์ฟเวอร์ซึ่งเขียนขึ้นโดยเฉพาะ โดยไดร์ฟเวอร์เหล่านี้สามารถเขียนโดยจาวาทั้งหมดและสามารถให้การส่งข้อมูลแบบ just-in-time ของแอพเพล็ท (เหมือน Type III) เนื่องจากไดร์ฟเวอร์เหล่านี้แปลงเจดีบีซีไปเป็นโปรโตคอลดั้งเดิมโดยตรงโดยไม่มีการใช้โอดีบีซีหรือเอพีไอดั้งเดิม จึงสามารถให้การเข้าถึงฐานข้อมูลที่มีประสิทธิภาพสูง ไดร์ฟเวอร์เหล่านี้ทำขึ้นจากผู้ผลิตดีบีเอ็มเอส (DBMS) เท่านั้น จากความจริงที่ว่าความรู้ในเรื่องโปรโตคอลเป็นของผู้ผลิตในปัจจุบันยังมี Type IV ใช้อยู่น้อย แต่จำนวนน่าจะมากขึ้นในเดือนต่อๆมา



รูปที่ ค.12 แสดงการติดต่อโดยใช้ไดรฟ์เวอร์ Type IV.

ตาราง ค-13 แสดงข้อดีข้อเสียของเจดีบีซีไดร์ฟเวอร์ชนิดต่างๆ

ชนิดของ JDBC	ข้อดี	ข้อเสีย
JDBC/ODBC Driver	สามารถใช้ในการเข้าถึงฐานข้อมูลได้หลายรูปแบบโดยเลือกใช้ODBCที่เหมาะสมกับฐานข้อมูลนั้นๆ	มีผลข้างเคียงที่อาจเกิดขึ้นและความยุ่งยาก เพราะว่าคำสั่งต้องส่งจาก JDBC ไปยัง bridge ที่เชื่อมไปยังODBC driver และสุดท้ายจาก ODBC ก็ส่งไปยัง native client-API เพื่อไปยัง ฐานข้อมูลทำให้การส่งข้อมูลทำได้ช้าหรือมีความเร็วในการส่งข้อมูลต่ำนั่นเองชนิดที่ 1 นี้ไม่สามารถรองรับการทำงานแบบ just-in-time deliveryได้ เนื่องจาก native code ที่ใช้จะต้องมีการติดตั้งบนเครื่องclient ก่อนแล้วถึงจะสามารถใช้ JDBC/ODBC bridge ในการเรียกใช้ API ได้ในการที่จะต้องทำการติดตั้งโปรแกรมที่ส่วน client ก่อนนี้ก็สามารถเปรียบเทียบได้กับระบบ client-server แบบเก่าๆที่ไม่สามารถแก้ไขปัญหา program deployment ในส่วนของ client ได้
Native-API, partly Java driver	มีความเร็วในการใช้งานมากกว่าชนิดที่ 1 เพราะ ส่วนที่ใช้ในการเปลี่ยนJDBCไปเป็นODBCนั้นได้ถูกนำออกไปทำให้การใช้งานหรือการทำงานมีความเร็วมากขึ้น	ต้องการ code ติดตั้งที่ส่วนclient ก่อน ดังนั้น ทั้งสองชนิดจึงมีปัญหาเกี่ยวกับการบำรุงรักษา solfwareอยู่เสมอๆ

Network-protocol, All-Java driver	เหมาะสมกับระบบเครือข่าย internet/intranet-based, multiuser data-intensive applications ซึ่งมีการทำ operation หลายๆอย่างพร้อมๆ กันเช่น การquery ,การค้นหาข้อมูล และอื่นๆ ซึ่งจะแบ่งงานกันได้อย่างมีประสิทธิภาพ server สามารถรองรับการจัดการฐานข้อมูลหลายๆแบบ	ในกรณีที่ middle tier เป็นแบบที่ใช้ภาษาC หรือC++เขียนขึ้นถ้าเกิดมีการเปลี่ยนชนิดของฐานข้อมูลจำเป็นที่จะต้องเปลี่ยน code ภายใน middle tier
Native-Protocol, All Java driver	สามารถทำ just-in-time delivery ได้ มีประสิทธิภาพในการเข้าถึงฐานข้อมูลสูง	เขียนยากและในปัจจุบันมีผู้ใช้น้อย

ภาคผนวก ง.

Source Code

1. Servlet

```
/*
*SubServlet.java
*Neural Networks Project*/

import java.io.*;
import java.util.*;
import java.math.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import oracle.jdbc.driver.*;

public class ProjectServlet extends HttpServlet{
   public void service(HttpServletRequest req,HttpServletResponse res)throws IOException{
      Enumeration keys;
      String key;
      String value;
      PrintWriter out=new PrintWriter(res.getOutputStream());
      String line=null;
      res.setContentType("text/html");
      out.println("<p>");
      out.println("<center>");
      String date_temp=req.getParameter("date_temp");
      String month_temp=req.getParameter("month_temp");
      String year_temp=req.getParameter("year_temp");
      String hour_temp=req.getParameter("hour_temp");
      String datatemp1_1=req.getParameter("datatemp1_1");
```

```
boolean succeed=false;
try{
   Class.forName("oracle.jdbc.driver.OracleDriver");
   Connection conn=DriverManager.getConnection
      ("jdbc:oracle:thin:@161.246.24.245:1521:orcl","project2","tamon");
   Statement stmt=conn.createStatement();

   //oracle
   ((OracleStatement)stmt).defineColumnType(1,Types.CHAR);
   ((OracleStatement)stmt).defineColumnType(2,Types.CHAR);
   ((OracleStatement)stmt).defineColumnType(3,Types.CHAR);
   ((OracleStatement)stmt).defineColumnType(4,Types.CHAR);
   ((OracleStatement)stmt).defineColumnType(5,Types.CHAR);
   ((OracleStatement)stmt).defineColumnType(6,Types.CHAR);
   ((OracleStatement)stmt).defineColumnType(7,Types.CHAR);
   ((OracleStatement)stmt).defineColumnType(8,Types.CHAR);
   ((OracleStatement)stmt).defineColumnType(9,Types.VARCHAR);
   ((OracleStatement)stmt).defineColumnType(10,Types.VARCHAR);
   ((OracleStatement)stmt).defineColumnType(11,Types.VARCHAR);

   String add_guery1=date_temp;
   out.println("<font color=#0000ff size=\"4\"><b><u>You selected No_day="+add_guery1+"</u>
      </b></font>");

   String add_guery2=month_temp;
   out.println("<font color=#0000ff size=\"4\"><b><u>month="+add_guery2+"</u></b></font>");
   String add_guery3=year_temp;
   out.println("<font color=#0000ff size=\"4\"><b><u>year="+add_guery3+"</u></b></font>");
   String add_guery4=hour_temp;
   out.println("<font color=#0000ff size=\"4\"><b><u>hour="+add_guery4+"</u></b></font>");
   String add_guery5=datatemp1_1;
   out.println("<font color=#0000ff size=\"4\"><b><u>values="+add_guery5+"</u></b></font>");
   String query="insert into tempnnweb(hr"+add_guery4+",No_day,month,year)
      values("+add_guery5+","+add_guery1+",'"+add_guery2+"',"+add_guery3+")";
   ResultSet rset=stmt.executeQuery(query);
```

```
            conn.close();
        }
        catch(Exception e){
            e.printStackTrace();
            out.println("<br>");
            out.println("<center>");
            out.println("connect -\n");
            postfail(out);
            postamble(out);
            return;
        }
        postamble(out);
    }

    public String getServletInfo(){
        return "Neural Networks Project ;)";
    }

    private void postamble(PrintWriter out){
        out.println("<center>");
        out.println("<br>");
        out.println("<br>");
        out.println("<br>");
        out.println("<br>");
        out.println("<br>");
        out.println("<br>");
        out.println("<br>");
        out.println(" <font color=#ff0000 size=5><b>Thank you </b></font> ");
        out.println("<br>");
        out.println(" <font color=#ff0000 size=5><b>Please connect again next time</b></font> ");
        out.println("<br>");
        out.println("<br>");
        out.println("<a href=\"http://161.246.24.245/welcomeload.html\">[LOAD]</a>");
        out.println("<a href=\"http://161.246.24.245/welcometemp.html\">[TEMPERATURE]</a>");
        out.println("<a href=\"http://161.246.24.245/welcomerain.html\">[RAIN]</a>");
```

```
        out.println("<a href=\"http://161.246.24.245/welcomesun.html\">[SUN]</a>");
        out.println("<a href=\"http://161.246.24.245/welcomerh.html\">[HUMUDITY]</a>");
        out.println("<a href=\"http://161.246.24.245/viewdata.html\">[VIEWDATA]</a>");
        out.println("</center>");
        out.println("<br>");
        out.println("</body>");
        out.println("</html>");
        out.flush();
    }

    private void postfail(PrintWriter out){
        out.println("<center>");
        out.println("<br>");
        out.println("<img src=http://161.246.10.21/~ksyspro/sis/image/ball_red.gif width=14 height=14
        ↳ alt=\"Red Ball\">");
        out.println("<img src=http://161.246.10.21/~ksyspro/sis/image/ball_red.gif width=14 height=14
        ↳ alt=\"Red Ball\">");
        out.println("<img src=http://161.246.10.21/~ksyspro/sis/image/ball_red.gif width=14 height=14
        ↳ alt=\"Red Ball\">");
        out.println(" <font color=#ff0000 size=5><b>HAVE ERROR</b></font> ");
        out.println("<img src=http://161.246.10.21/~ksyspro/sis/image/ball_red.gif width=14 height=14
        ↳ alt=\"Green Ball\">");
        out.println("<img src=http://161.246.10.21/~ksyspro/sis/image/ball_red.gif width=14 height=14
        ↳ alt=\"Green Ball\">");
        out.println("<img src=http://161.246.10.21/~ksyspro/sis/image/ball_red.gif width=14 height=14
        ↳ alt=\"Green Ball\">");
        out.println("<img src=http://161.246.10.21/~ksyspro/sis/image/ball_red.gif width=14 height=14
        ↳ alt=\"Green Ball\">");
        out.println("<p>");
    }
}
```

2. HTML

```
<html>
<head>
<title>Post data</title>
<script language="JavaScript">
<!– begin
// global variables
var max=0;
function textlist(){
   max=textlist.arguments.length;
      for (i=0; i<max; i++)
      this[i]=textlist.arguments[i];
}
tl=new textlist
(
   "..... Electric Load Forecasting in Thailand .....",
   "****    Very short term load forecasting    ****",
   "         by Artificial Neural Networks          ",
   "......  Electrical Engineering Faculture  ......",
   "King Mongkut's Institute of Technology Ladkrabang",
   "========= Please Input Temperature Data =========",
   "****   You can put other data and submit   ****",
   "    Load, Temperature, Rain, Sun and Humidity    ",
   "....I wish you will be happying with our homepage....",
   "        ****$$$$ THANK YOU $$$$****            "
);

var x=0; pos=0;
var l=tl[0].length;
function textticker(){
   document.tickform.tickfield.value=tl[x].substring(0,pos)+"_";
   if(pos++==l){
      pos=0;
      setTimeout("textticker()",1000);
```

```
        x++;
        if(x==max)
            x=0;
        l=tl[x].length;
    }
    else
        setTimeout("textticker()",50);
}
// end -->

<!-- begin code
var timerID = null;
var timerRunning = false;
function stopclock (){
    if(timerRunning)
        clearTimeout(timerID);
    timerRunning = false;
}
function startclock () {
stopclock();
showtime();
}

function showtime (){
    var now = new Date();
    document.clock.IN.value = now;
    timerID = setTimeout("showtime()",1000);
    timerRunning = true;
}
// end code -->
```

```
</script>
</head>
<body onLoad="textticker();startclock()" background="graph.gif" text=#ffffff link=#0000ff
  ↳ vlink=#800080 alink=#ff0000>
<center>
<table border=1 width=480 cellpadding=2 cellspacing=0>
<tr><td bgcolor=#ffe0e0 align=center>
<font color=#0000a0 size="4"><b>Temperature at North</b></font><br>
 </td>
</tr>
</table><BR>
<center><!img src="/cgi-bin/Count.cgi?dd=C|df=sc390211.dat"></center>
<form name="clock" onSubmit="0"><input type="text" name="IN" size=35 value=""></form>
<center><form name="tickform"><input type=text name="tickfield" size=28></form><p></center>
<hr width=75%>
<form method="post" action="http://161.246.24.245:8080/servlet/Jojo61Servlet">
select date :
<select name="date_temp">
<option value="1">1
<option value="2">2
<option value="3">3
<option value="4">4
<option value="5">5
<option value="6">6
<option value="7">7
<option value="8">8
<option value="9">9
<option value="10">10
<option value="11">11
<option value="12">12
<option value="13">13
<option value="14">14
<option value="15">15
<option value="16">16
<option value="17">17
```

```
<option value="18">18
<option value="19">19
<option value="20">20
<option value="21">21
<option value="22">22
<option value="23">23
<option value="24">24
<option value="25">25
<option value="26">26
<option value="27">27
<option value="28">28
<option value="29">29
<option value="30">30
<option value="31">31
</SELECT>       
select month :
<select name="month_temp">
<option value="January">January
<option value="Febuary">Febuary
<option value="March">March
<option value="April">April
<option value="May">May
<option value="June">June
<option value="July">July
<option value="August">August
<option value="September">September
<option value="October">October
<option value="November">November
<option value="December">December
</select>       
select year :
<select name="year_temp">
<option value="1998">1998
<option value="1999">1999
<option value="2000">2000
```

```
</select>        
<br><br>select hours :
<select name="hour_temp">
<option value="1">1
<option value="2">2
<option value="3">3
<option value="4">4
<option value="5">5
<option value="6">6
<option value="7">7
<option value="8">8
</select>        
Input Data :
<input type="text" name="datatemp1_1" size=10 maxlength=20 >
<br><br>
<input type="submit" value="submit">
<input type="reset" value="reset form">
</form>
<a href="welcomeload.html">[LOAD]</a>
<a href="welcometemp.html">[TEMPERATURE]</a>
<a href="welcomerain.html">[RAIN]</a>
<a href="welcomesun.html">[SUN]</a>
<a href="welcomerh.html">[HUMUDITY]</a>
<a href="viewdata.html">[VIEWDATA]</a>
</body>
</html>
```

3. Neural Networks โดยโปรแกรม MatLab

```
clc;
clear all;
close all;

nb_in          =    38;          % The number of input neurals.
nb_out         =    3;           % The number of output neurals.
nb_hid         =    1;           % The number of hidden layers.
nb_pattern     =    16;          % The number of training patterns.
nb_analysis    =    8;           % The number of analysis patterns.
nb_hid_st      =    77;          % The number of first hidden layer neurals.
Bias           =    1;           % Set Bias on each neural.
Permissible    =    0.0000005;   % Permissible error.
Learning_rate  =    0.40;        % Used in Back-propagation training algorithm.
Momentum       =    0.20;        % Used in Back-propagation training algorithm.
load f:\Train\Wednesday\Ptrain\We1Tr3H102_2.mat; % Load data to used in training algorithm.
global Stop;

% Bias acts like a weight on a connection from a unit with
% a constant activation of 1.
in_mt          =    zeros([1, nb_in+1]);    % Input matrix.
in_mt(1, nb_in+1)  =    Bias;
out_mt         =    zeros([1, nb_out]);     % Output matrix.

% First hidden layer output matrix.
out_hid_st_mt = zeros([1, nb_hid_st+1]);
out_hid_st_mt(1, nb_hid_st+1) = Bias;
```

```
% Initialize weight and bias , select from menu
WRandom = [...
   'set([BPTrain_H Analysis_H Load_H Save_H Exit_H],"Enable","off");'...
   'menu1 = menu("Initialize The Connection Strengths","Random values");'...
   'disp(" ");'...
   'w1_mt = randn([nb_in+1, nb_hid_st]);'...
   'w2_mt = randn([nb_hid_st+1, nb_out]);'...
   'if max(max(w1_mt)) >= abs(min(min(w1_mt))),'...
      'w1_mt = w1_mt / max(max(w1_mt));'...
   'else,'...
      'w1_mt = w1_mt / min(min(w1_mt));'...
   'end;'...
   'if max(max(w2_mt)) >= abs(min(min(w2_mt))),'...
      'w2_mt = w2_mt / max(max(w2_mt));'...
   'else,'...
      'w2_mt = w2_mt / min(min(w2_mt));'...
   'end;'...
   'if menu1 == 1,'...
      'w1_mt = 1*w1_mt;'...
      'w2_mt = 1*w2_mt;'...
   'end;'...
   'set([BPTrain_H Analysis_H Load_H Save_H Exit_H],"Enable","on");'];

BPTrain = [...
   'set(Stop_H,"Enable","on");'...
   'set([WRandom_H BPTrain_H Analysis_H Load_H ],"Enable","off");'...
   'figure(figC); clf;'...
   'uicontrol("Style","text","Position",[15 5 300 20],"Horiz","left","String","Training Process ... loops.","Fore",
      [1 .1 .1],"Back",[0 0 0]); '...
   'set(figE,"Visible","on");'...
   'Stop = 0;'...
   'loop = 0;'...
   'last = 1;'...
   'Iteration = 0;'...
   'w1_old = w1_mt;'...
```

```
'w2_old = w2_mt;'...
'SumSqrErr(1,1) = Permissible+0.000001;'...
'tic;'...
'while SumSqrErr(1,last) > Permissible & Stop == 0 ,'...
    'for pat = 1:nb_pattern;'...
        'if Stop == 1;'...
            'break;'...
        'end;'...
        'in_mt(1:nb_in) = Pattern_mt(pat, :);'...
        'out_hid_st_mt(1:nb_hid_st) = logsig(in_mt*w1_mt);'...
        'out_mt = logsig(out_hid_st_mt*w2_mt);'...
        'Out_pattern = Target_mt(pat, :);'...
        'NetsError = Out_pattern - out_mt;'...
        'Error_Term2 = out_mt.*(1-out_mt).*NetsError;'...
        'DeltaW2 = Learning_rate.*Error_Term2;'...
        'DeltaW2 = out_hid_st_mt"*DeltaW2;'...
        'w2_new = w2_mt + DeltaW2 + (Momentum*(w2_mt - w2_old));'...
        'w2_old = w2_mt;'...
        'w2_mt = w2_new;'...
        'Error_Term1 = out_hid_st_mt(1:nb_hid_st).*(1-out_hid_st_mt(1:nb_hid_st)).*
          ↳ (Error_Term2*w2_mt(1:nb_hid_st,1:nb_out)");'...
        'DeltaW1 = Learning_rate.*Error_Term1;'...
        'DeltaW1 = in_mt"*DeltaW1;'...
        'w1_new = w1_mt + DeltaW1 + (Momentum*(w1_mt - w1_old));'...
        'w1_old = w1_mt;'...
        'w1_mt = w1_new;'...
        'loop = loop + 1;'...
        'SqrErr(1, pat) = sum(sum(NetsError.*NetsError));'...
        'pause(0.000001);'...
    'end;'...
    'last = last+1;'...
    'Iteration = Iteration + 1;'...
    'SumSqrErr(1,last) = (sum(SqrErr))/(2*nb_pattern);'...
    'SumSqrError = SumSqrErr(1,2:length(SumSqrErr));'...
```

```
    'figure(figC); clf;'...
      'uicontrol("Style","text","Position",[ 15 5 300 20],"Horiz","left","String","Training Process ...","Fore",
        ↳ [0 1 0],"Back",[0 0 0]); '...
      'uicontrol("Style","text","Position",[180 5 300 20],"Horiz","left","String",Iteration,"Fore",[0 1 0],"Back",
        ↳ [0 0 0]); '...
      'uicontrol("Style","text","Position",[250 5 300-20],"Horiz","left","String","Iterations.","Fore",
        ↳ [0 1 0],"Back",[0 0 0]); '...
      'figure(figB); clf;'...
      'uicontrol("Style","text","Position",[ 15 5 300 20],"Horiz","left","String","SumSquareError ...","Fore",
        ↳ [0 1 0],"Back",[0 0 0]); '...
      'uicontrol("Style","text","Position",[180 5 300 20],"Horiz","left","String",SumSqrErr(1,last),"Fore",
        ↳ [0 1 0],"Back",[0 0 0]); '...
  'end;'...
    'save f:\Train\Wednesday\Weigth\10fwg40m00p0005r100_1_2.mat w1_mt w2_mt;'...
    'save f:\Train\Wednesday\Error\10sqg40m00p0005r100_1_2.mat SumSqrError;'...
  'figure(figD); clf;'...
    'uicontrol("Style","text","Position",[ 15 5 300 20],"Horiz","left","String","Time in Process ...","Fore",
      ↳ [0 1 0],"Back",[0 0 0]); '...
    'uicontrol("Style","text","Position",[180 5 300 20],"Horiz","left","String",toc,"Fore",[0 1 0],"Back",
      ↳ [0 0 0]); '...
  'figure(figE);'...
    'plot(SumSqrError,"b-.");'...
  'set(Stop_H,"Enable","off");'...
  'set([WRandom_H BPTrain_H Analysis_H Load_H Exit_H],"Enable","on");'];

Analysis = [...
  'set(Stop_H,"Enable","on");'...
  'set([WRandom_H Load_H BPTrain_H Analysis_H Exit_H],"Enable","off");'...
  'Stop = 0; loop = 0;'...
  'figure(figC);'...
    'uicontrol("Style","text","Position",[15 5 320 20],"Horiz","left","String","No-Operation.","Fore",
      ↳ [.1 .1 1],"Back",[0 0 0]);'...
  'figure(figB); clf;'...
  'menu1 = menu("Choose The Pattern","LF for Next 3 Hours");'...
  'disp(" ");'...
```

```
'if menu1 == 1,'...
    'load f:\Train\Wednesday\Analysis\We2A3H102_2.mat;'...
    'load f:\Train\Wednesday\Error\10sqg40m00p0005r100_1_2.mat;'...
    'fid = fopen( "f:\Train\Wednesday\TestResult\10We2g40m00p0005r100_1_2.txt ", "w");'...
'end;'...
'set([figH figI ],"Visible","on");'...
'[x Iteration] = size(SumSqrError);'...
'SQR = SumSqrError(1,Iteration);'...
'TError = 0;AError = 0;'...
'for pat = 1: nb_analysis;'...
    'if Stop == 1;'...
        'break;'...
    'end;'...
    'in_mt(1:nb_in) = Analysis_mt(pat,:);'...
    'out_hid_st_mt(1:nb_hid_st) = logsig(in_mt*w1_mt);'...
    'out_mt = logsig(out_hid_st_mt*w2_mt);'...
    'Out_pattern = ATarget_mt(pat,:);'...
    'FLoad1 = (out_mt(1,1) * 20000);'...
    'FText1 = "Load at next Hr";'...
    'RLoad1 = (Out_pattern(1,1)*20000);'...
    'Error1 = (FLoad1-RLoad1);'...
    'PError1 = ((FLoad1-RLoad1)/(RLoad1))*100;'...
    'fprintf(fid,"\n %d\t %s\t %.4f\t %.4f\t %.4f\t %.4f", pat ,FText1,RLoad1,FLoad1,Error1,PError1);'...
    'figure(figH);clf;'...
        'uicontrol("Style","text","Position",[ 15 85 157 20],"Horiz","left","String",FText1,"Fore",
            ↳ [.1 1 .1],"Back",[0 0 0]);'...
        'uicontrol("Style","text","Position",[170 85  50 20],"Horiz","left","String",FLoad1,"Fore",
            ↳ [1 .1 .1],"Back",[0 0 0]);'...
        'uicontrol("Style","text","Position",[215 85  22 20],"Horiz","left","String","MW","Fore",
            ↳ [.1 1 .1],"Back",[0 0 0]);'...
    'figure(figI);clf;'...
        'uicontrol("Style","text","Position",[ 15 85 157 20],"Horiz","left","String",FText1,"Fore",
            ↳ [.1 1 .1],"Back",[0 0 0]);'...
        'uicontrol("Style","text","Position",[170 85  50 20],"Horiz","left","String",RLoad1,"Fore",
            ↳ [1 .1 .1],"Back",[0 0 0]);'...
```

```
    'uicontrol("Style","text","Position",[215 85  22 20],"Horiz","left","String","MW","Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[250 85 120 20],"Horiz","left","String","Error","Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[290 85  50 20],"Horiz","left","String",Error1,"Fore",
      ↳ [1 .1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[345 85  22 20],"Horiz","left","String","MW","Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
'FText2 = "Load at next 2 Hrs";'...
'FLoad2 = (out_mt(1,2) * 20000);'...
'RLoad2 = (Out_pattern(1,2)*20000);'...
'Error2 = (FLoad2-RLoad2);'...
'PError2 = ((FLoad2-RLoad2)/(RLoad2))*100;'...
'fprintf(fid,"\n\t %s\t %.4f\t %.4f\t %.4f\t %.4f",FText2,RLoad2,FLoad2,Error2,PError2);'...
'figure(figH);'...
    'uicontrol("Style","text","Position",[ 15 45 157 20],"Horiz","left","String",FText2,"Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[170 45  50 20],"Horiz","left","String",FLoad2,"Fore",
      ↳ [1 .1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[215 45  22 20],"Horiz","left","String","MW","Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
'figure(figI);'...
    'uicontrol("Style","text","Position",[ 15 45 157 20],"Horiz","left","String",FText2,"Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[170 45  50 20],"Horiz","left","String",RLoad2,"Fore",
      ↳ [1 .1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[215 45  22 20],"Horiz","left","String","MW","Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[250 45 120 20],"Horiz","left","String","Error","Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[290 45  50 20],"Horiz","left","String",Error2,"Fore",
      ↳ [1 .1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[345 45  22 20],"Horiz","left","String","MW","Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
```

```
'FText3 = "Load at next 3 Hrs";'...
'FLoad3 = (out_mt(1,3) * 20000);'...
'RLoad3 = (Out_pattern(1,3)*20000);'...
'Error3 = (FLoad3-RLoad3);'...
'PError3 = ((FLoad3-RLoad3)/(RLoad3))*100;'...
'fprintf(fid,"\n\t %s\t %.4f\t %.4f\t %.4f\t %.4f",FText3,RLoad3,FLoad3,Error3,PError3);'...
'figure(figH);'...
    'uicontrol("Style","text","Position",[ 15 5 157 20],"Horiz","left","String",FText3,"Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[170 5  50 20],"Horiz","left","String",FLoad3,"Fore",
      ↳ [1 .1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[215 5  22 20],"Horiz","left","String","MW","Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
'figure(figI);'...
    'uicontrol("Style","text","Position",[ 15 5 157 20],"Horiz","left","String",FText3,"Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[170 5  50 20],"Horiz","left","String",RLoad3,"Fore",
      ↳ [1 .1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[215 5  22 20],"Horiz","left","String","MW","Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[250 5 120 20],"Horiz","left","String","Error","Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[290 5  50 20],"Horiz","left","String",Error3,"Fore",
      ↳ [1 .1 .1],"Back",[0 0 0]);'...
    'uicontrol("Style","text","Position",[345 5  22 20],"Horiz","left","String","MW","Fore",
      ↳ [.1 1 .1],"Back",[0 0 0]);'...
  'pause(0.0001);'...
  'TError = abs(PError1)+abs(PError2)+abs(PError3)+TError;'...
'end;'...
'pause(0.0001);'...
'AError = TError/(8*3);'...
'FText4 = "Average Error is ";'...
'fprintf(fid,"\n\t %s\t %.6f",FText4,AError);'...
'FText5 = "Number of Iteration =";'...
'fprintf(fid,"\n\t %s\t %d",FText5,Iteration);'...
```

```
    'FText6 = "Sum-Square-Error =";'...
    'fprintf(fid,"\n\t %s\t %.15f",FText6,SQR);'...
    'fclose(fid);'...
    'set([figH figI],"Visible","off");'...
    'figure(figD);'...
        'uicontrol("Style","text","Position",[15 5 290 20],"Horiz","left","String",Aerror,"Fore",
        ↳ [.1 .1 1],"Back",[0 0 0]);'...
    'set(Stop_H,"Enable","off");'...
    'set([WRandom_H BPTrain_H Analysis_H Load_H Save_H Exit_H],"Enable","on");' ];

LoadW = [...
    set([WRandom_H BPTrain_H Analysis_H Save_H],"Enable","off");'...
    'menu1 = menu("Load Menu","Load Weigths ","Load Initial weigths");'...
    'disp(" ");'...
    'if menu1 == 1,'...
        'load f:\Train\Wednesday\Weigth\10fwg40m00p00075r100_1_2.mat;'...
    'elseif menu1 == 2,'...
        'load f:\Train\Wednesday\Weigth\10swinir100.mat;'...
    'end;'...
    'set([WRandom_H BPTrain_H Analysis_H Save_H],"Enable","on");'...
    'clear menu1;'];

SaveW = [...
    'set([BPTrain_H Analysis_H Load_H Save_H Exit_H],"Enable","off");'...
    'menu1 = menu("Save Menu","Save to backup weights","Save to initial weigth");'...
    'disp(" ");'...
    'if menu1 == 1,'...
        'save f:\Train\Wednesday\Weigth\072bwg50m00p100r100_1_2.mat w1_mt w2_mt;'...
    'elseif menu1 == 2,'...
        'save f:\Train\Wednesday\Weigth\10swinir100 w1_mt w2_mt;'...
    'end;'...
    'set([BPTrain_H Analysis_H Load_H Save_H Exit_H],"Enable","on");'...
    'clear menu1;'];
```

```
Exit = [...
   'close([figA figB figC figD figE figF figG figH figI]);'...
   'clear all;'];

% Display error plot windows with control menu
% Create Active Windows
figA = figure('Position',[5 5 700 265],'Name','Mean-Square-Error Surface And Control ...',...
       'NumberTitle','off','Resize','off','Interruptible','no');
clf;
uicontrol('Style','frame','Position',[5 5 140 250],'BackgroundColor',[0.5 0.5 0.5]);

% Create user interface control.
WRandom_H = uicontrol('Style','push','Position',[10 220 130 30],...
                      'String','Random Weights','Back',[0 0 0],...
                      'Interruptible','yes','CallBack',WRandom);

BPTrain_H = uicontrol('Style','push','Position',[10 185 130 30],...
                      'String','Back-P training','Back',[0 0 0],...
                      'Interruptible','yes','CallBack',BPTrain,'Enable','off');

Analysis_H = uicontrol('Style','push','Position',[10 150 130 30],...
                      'String','Pattern Analysis','Back',[0 0 0],...
                      'Interruptible','yes','CallBack',Analysis,'Enable','off');

Load_H = uicontrol('Style','push','Position',[10 115 130 30],...
                      'String','Load Weights','Back',[0 0 0],...
                      'Interruptible','yes','CallBack',LoadW);

Save_H = uicontrol('Style','push','Position',[10 80 130 30],...
                      'String','Save Weights','Back',[0 0 0],...
                      'Interruptible','yes','CallBack',SaveW,'Enable','off');

Stop_H = uicontrol('Style','push','Position',[10 45 130 30],...
                      'String','Stop !','Back',[0 0 0],...
                      'Interruptible','yes','CallBack','Stop = 1;','Enable','off');
```

```
Exit_H = uicontrol('Style','push','Position',[10 10 130 30],...
                'String','Exit','Back',[0 0 0],...
                'Interruptible','yes','CallBack',Exit,'Enable','off');

% Create and display Sum-Square-Error values.
figB = figure('Position',[714 182 300 40],'Name','Sum-Square-Error values ...',...
                'NumberTitle','off','Resize','off','MenuBar','none');
clf;
FText = 'No Operation.';
uicontrol('Style','text','Position',[15 5 320 20],'Horiz','left',...
                'String',FText,'Fore',[.1 .1 1],'Back',[0 0 0]);

% Create Train display.
figC = figure('Position',[714 250 300 40],'Name','Train Status ...',...
                'NumberTitle','off','Resize','off','MenuBar','none');
clf;
uicontrol('Style','text','Position',[15 5 320 20],'Horiz','left',...
                'String',FText,'Fore',[.1 .1 1],'Back',[0 0 0]);

% Create Analysis display.
figD = figure('Position',[714 115 300 40],'Name','Alarm Messages Analysis Status ...',...
                'NumberTitle','off','Resize','off','MenuBar','none');
clf;
uicontrol('Style','text','Position',[15 5 320 20],'Horiz','left',...
                'String',FText,'Fore',[.1 .1 1],'Back',[0 0 0]);

% Create and display Back-Propagation Training Method SumSqrErr versus Number of Iteration.
figE = figure('Position',[5 315 1020 200],'Name','Sum-Square-Errors of Back-Propagation Training Method ...',...
                'NumberTitle','off','Resize','off','Visible','off');

% Create and display Analysis Forecasted Forecasting.
figH = figure('Position',[5 320 255 120],'Name','Load Forecast ...',...
                'NumberTitle','off','Resize','off','MenuBar','none','Visible','off');
clf;
uicontrol('Style','text','Position',[15 85 320 20],'Horiz','left',...
                'String',FText,'Fore',[.1 .1 1],'Back',[0 0 0]);
```

```
% Create and display Actual Load & Error from Forecasting.
figI = figure('Position',[270 320 390 120],'Name','Actual Load & Error from Forecasting ...',...
                'NumberTitle','off','Resize','off','MenuBar','none','Visible','off');
clf;
uicontrol('Style','text','Position',[15 85 320 20],'Horiz','left',...
                'String',FText,'Fore',[.1 .1 1],'Back',[0 0 0]);
```

3.1 FUNCTION LOGSIG

```
function a = logsig(n,b)

%LOGSIG  Logistic Sigmoid Activation Transfer Function.Returns elements of N squashed
%        between 0 and 1 using the log-sigmoid function shifted by a bias.
%        (See LEARNBP, NWLOG, TANSIG)
%LOGSIG(N)  N - SxQ matrix of net input vectors.
%          Returns the log-sigmoid with bias of 0.
%LOGSIG(N,B)  Used when Batching. B - Sx1 vector of biases. Returns the log-sigmoid
%          using the biases in each row of B for elements in corresponding rows of
%          N.
%WARNING: LOGSIG is intended for use with real valued arguments.  Use complex
%        arguments at your own risk! M.H. Beale & H.B. Demuth, 1-31-92
%        Copyright (c) 1992-93 by the MathWorks, Inc.

if nargin < 1 | nargin > 2
        error('Wrong number of arguments.');
end

if nargin==1
        z = n;
else
[nr,nc] = size(n);
z = n + b*ones(1,nc);
pend

a = 1 ./ (1+exp(-z));

i = find(~finite(a));
a(i) = sign(z(i))*0.5 + 0.5;
```

4 Neural Networks โดยภาษา Java

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import java.io.*;
import java.sql.*;
import java.net.*;
import oracle.jdbc.driver.*;
import netscape.security.*;
import Activate;
import Back_Prop;
import InitSetting;
import Utility;

public class ANNs extends Applet{
   // Create panel, textfield and button
   Panel  MainUIPanel          = new Panel();
   Panel  UIPanel1             = new Panel();
   Panel  UIPanel2             = new Panel();
   Panel  UIPanel3             = new Panel();
   Panel  UIPanel4             = new Panel();
   TextField  Status1Field     = new TextField(15);
   TextField  Status2Field     = new TextField(15);
   TextField  Status3Field     = new TextField(15);
   TextField  Status4Field     = new TextField(15);
   TextField  Status5Field     = new TextField(15);
   TextField  Output1Field     = new TextField(15);
   TextField  Output2Field     = new TextField(15);
   TextField  Output3Field     = new TextField(15);
   TextField  Output4Field     = new TextField(15);
   TextField  Output5Field     = new TextField(15);
   TextField  Output6Field     = new TextField(15);
   TextField  Output7Field     = new TextField(15);
```

```
TextField Output8Field      = new TextField(15);
TextField Output9Field      = new TextField(15);
TextField Output10Field     = new TextField(15);
Button loadButton           = new Button("Load Pattern");
Button loadwButton          = new Button("Load Weight");
Button savewButton          = new Button("Save Weight");
Button trainButton          = new Button(" Training ");
Button forecastButton       = new Button(" Forecast ");
Button nextButton           = new Button("  Next  ");
Button resetButton          = new Button("  Reset  ");

// Variable declaration for neural networks.
short Nb_In        = 38;          // The number of input neurals.
short Nb_Hid       = 77;          // The number of neurals in first hidden layer.
short Nb_Out       = 3;           // The number of output neurals.
short Nb_Train     = 16;          // The number of training patterns.
short Nb_Analysis  = 8;           // The number of analysis patterns.
double Permissible = 0.004000;    // Used in Back-Propagation training algorithm.
double SumSquareError;            // Used in Back-Propagation training algorithm.
int  Iteration;                   // Used in Back-Propagation training algorithm.
double In_mt[][]          = new double[1][Nb_In+1];          // Input matrix.
double Out_Hid_st_mt[][]  = new double[1][Nb_Hid+1];         // Hidden layer output matrix.
double Out_Hid_mt[][]     = new double[1][Nb_Hid];           // Hidden layer node matrix.
double Out_mt[][]         = new double[1][Nb_Out];           // Output matrix.
double Train_mt[][]       = new double[Nb_Train][Nb_In+Nb_Out];// Pattern train matrix.
double Analysis_mt[][]    = new double[Nb_Analysis][Nb_In+Nb_Out];// Pattern analysis matrix.
double Target[][]         = new double[1][Nb_Out];           // Output matrix.
double Actual[][]         = new double[Nb_Analysis][Nb_Out];// Actual Load matrix.
double Forecast[][]       = new double[Nb_Analysis][Nb_Out];// Farecasting output matrix.
double Error[][]          = new double[Nb_Analysis][Nb_Out];// Eror matrix.
double W1_mt[][]          = new double[Nb_In+1][Nb_Hid];     // Weigths or Threshole matrix
                              between input layer and first hidden layer.
double W2_mt[][]    = new double[Nb_Hid+1][Nb_Out];          // Weigths or Threshole matrix
                              between first hidden layer and output layer.
```

```
int   XRes          = 500;                  // Y-axis resolution in plotting graph.
int   YRes          = 250;                  // Y-axis resolution in plotting graph.
int   XOri          =  50;                  // X-Origin transfer to plotting graph.
int   YOri          = 280;                  // Y-Origin transfer to plotting graph.

InitSetting  ISetting  = new InitSetting();      // Used InitSetting objects.
Utility      GetUtil   = new Utility();          // Used Utility objects.
Activate     Factivate = new Activate();         // Used Activate objects.
Back_Prop    BP        = new Back_Prop();        // Used Back_Prop objects.
PatTrain     PT        = new PatTrain();         // Used PatTrain objects.
PatAnalysis  PA        = new PatAnalysis();      // Used PatAnalysis objects.
PlotCurve    PC        = new PlotCurve(XOri,YOri,XRes,YRes);

static final short SDefault     = 0;    // To assign Status to Default Status.
static final short SPLoading    = 1;    // To assign Status to Data-Loading Status.
static final short STraining    = 2;    // To assign Status to Training Status.
static final short Sanalysis    = 3;    // To assign Status to Analysis Status.
static final short SWLoading    = 4;    // To assign Status to Weigth Loading Status.
static final short SWSaving     = 5;    // To assign Status to Weigth Saving Status.
short Status = SDefault;                // To assign Variable to select the Status.
static final short ADefault     = 0;    // To assign Active to Default Active.
static final short AStart       = 1;    // To assign Active to Start Active.
static final short AFinish      = 2;    // To assign Active to Finish Active.
short Active = ADefault;                // To assign Variable to select the Active.
static final short PLDefault    = 0;    // To assign Loading Status to Default Status.
static final short PLPass       = 1;    // To assign Loading Pattern to passed Status.
short PLoading = PLDefault;             // To assign Variable to select the Loading Status.
static final short WLDefault    = 0;    // To assign Loading Status to Default Status.
static final short WLPass       = 1;    // To assign Loading Weigth to passed Status.
short WLoading = WLDefault              // To assign Variable to select the Loading Status.
static final short WDefault     = 0;    // To assign Saving Status to Default Status.
static final short WSPass       = 1;    // To assign Saving Weigth to passed Status.
short WSaving = SDefault;               // To assign Variable to select the Loading Status.
short No_Pat = 0;                       // To assign the Default pattern to analysis.
short NumPat = No_Pat;                  // To assign Variable to select the pattern to analysis.
```

```
public void init(){
    // Add handler actionlistener to Button.
    loadPatButton.addActionListener(new ButtonHandler());
    loadwButton.addActionListener(new ButtonHandler());
    savewButton.addActionListener(new ButtonHandler());
    trainButton.addActionListener(new ButtonHandler());
    forecastButton.addActionListener(new ButtonHandler());
    nextButton.addActionListener(new ButtonHandler());
    resetButton.addActionListener(new ButtonHandler());

    // Add Component & Setlayout to MainUIPanel.
    setLayout(new BorderLayout());
    add("South",MainUIPanel);
    MainUIPanel.setLayout(new GridLayout(4,1));

    // Add Component & Setlayout to UIPanel.
    UIPanel1.add(Status1Field);
    UIPanel1.add(Status2Field);
    UIPanel1.add(Status3Field);
    UIPanel1.add(Status4Field);
    UIPanel1.add(Status5Field);
    MainUIPanel.add(UIPanel1);
    UIPanel2.add(Output1Field);
    UIPanel2.add(Output2Field);
    UIPanel2.add(Output3Field);
    UIPanel2.add(Output4Field);
    UIPanel2.add(Output5Field);
    MainUIPanel.add(UIPanel2);
    UIPanel3.add(Output6Field);
    UIPanel3.add(Output7Field);
    UIPanel3.add(Output8Field);
    UIPanel3.add(Output9Field);
    UIPanel3.add(Output10Field);
    MainUIPanel.add(UIPanel3);
```

```
    UIPanel4.add(loadPatButton);
    UIPanel4.add(trainButton);
    UIPanel4.add(forecastButton);
    UIPanel4.add(nextButton);
    UIPanel4.add(resetButton);
    MainUIPanel.add(UIPanel4);

    // Set background color to white.
    setBackground(Color.white);

    // To set cann't edit in the TextField.
    Status1Field.setEditable(false);
    Status2Field.setEditable(false);
    Status3Field.setEditable(false);
    Status4Field.setEditable(false);
    Status5Field.setEditable(false);
    Output1Field.setEditable(false);
    Output2Field.setEditable(false);
    Output3Field.setEditable(false);
    Output4Field.setEditable(false);
    Output5Field.setEditable(false);
    Output6Field.setEditable(false);
    Output7Field.setEditable(false);
    Output8Field.setEditable(false);
    Output9Field.setEditable(false);
    Output10Field.setEditable(false);

    showstatus();       // Show status with all default status.
}

void showstatus(){
    if (Status == SDefault || Status == STraining){
        Status1Field.setText("          Status  ");
        Status2Field.setText("Numbers of Iteration ");
        Status3Field.setText("  Sum Square Error  ");
```

```
    Status4Field.setText("          Times   ");
    if (Active == AStart)
        Output1Field.setText("      Training....");
    else if (Active == AFinish)
        Output1Field.setText("        Finish   ");
}
else if (Status == SPLoading){
    Status1Field.setText("Connect to database  ");
    Output1Field.setText("      Connecting ! ");
    Output6Field.setText("         Loading! ");
    Status2Field.setText("  Pattern train data");
    Status3Field.setText("Pattern forecast data");
}
else if (Status == SAnalysis){
    Status1Field.setText("          Time     ");
    Status2Field.setText("       Next hour   ");
    Status3Field.setText("  Next 2 hours (MW)   ");
    Status4Field.setText("  Next 3 hours (MW)   ");
    Status5Field.setText("       Error (%)   ");
    Output1Field.setText("   Forecasted Load    ");
    Output6Field.setText("     Actual Load    ");
}
else if (Status == SWLoading){
    Status1Field.setText("      Weigth1 mtrix");
    Status2Field.setText("Saving Weigth1 status");
    Status3Field.setText("      Weigth2 mtrix");
    Status4Field.setText("Saving Weigth2 status");
}
else if (Status == SWSaving){
    Status1Field.setText("     Weigth Matrix");
    Status2Field.setText("        Element ");
    Status3Field.setText("      Connecting");
    Status4Field.setText("        Saving  ");
    Output1Field.setText("        Weigth 1 ");
    Output6Field.setText("        Weigth 2 ");
```

```
        }
    }

    void showwarning(){
        clearstatus();
        Status1Field.setText("        Warning !!! ");
        Output1Field.setText("      Please select   ");
        Output2Field.setText(" Load Pattern Button   ");
        Output3Field.setText("          before     ");
        if (Status == STraining){
            Output4Field.setText("         Training     ");
        else if (Status == SAnalysis){
            Output4Field.setText("       Forecasting    ");
        Status = SDefault;
}

    void showiteration(long value){
        String Value = "\t    "+value;
        Output2Field.setText(Value);
    }

    void showerror(double value1,double value2){
        String Value1 = "  Avg. =  "+value1;
        Output5Field.setText(Value1);
        String Value2 = "  Max. =  "+value2;
        Output10Field.setText(Value2);
    }

    void showsumsquareerror(double value){
        String Value = ""+value;
        Output3Field.setText(Value);
    }
```

```
    void clearstatus(){
        Status1Field.setText("");
        Status2Field.setText("");
        Status3Field.setText("");
        Status4Field.setText("");
        Status5Field.setText("");
        Output1Field.setText("");
        Output2Field.setText("");
        Output3Field.setText("");
        Output4Field.setText("");
        Output5Field.setText("");
        Output6Field.setText("");
        Output7Field.setText("");
        Output8Field.setText("");
        Output9Field.setText("");
        Output10Field.setText("");
    }

    void shownext1hour(double value1,double value2){
        String Value1 = "  "+value1;
        Output2Field.setText(Value1);
        String Value2 = "  "+value2;
        Output7Field.setText(Value2);
}

    void shownext2hour(double value3,double value4){
        String Value3 = "  "+value3;
        Output3Field.setText(Value3);
        String Value4 = "  "+value4;
        Output8Field.setText(Value4);
    }
```

```
void shownext3hour(double value5,double value6){
    String Value5 = "  "+value5;
    Output4Field.setText(Value5);
    String Value6 = "  "+value6;
    Output9Field.setText(Value6);
}

void showtime(int Tim){
    String TIM = "          "+Tim+"   Sec.";
    Output4Field.setText(TIM);
}

void train(){
    // Variable declaration.
    double W1_old[][]  = new double[Nb_In+1][Nb_Hid];
    double W2_old[][]  = new double[Nb_Hid+1][Nb_Out];
    double W1_new[][] = new double[Nb_In+1][Nb_Hid];
    double W2_new[][] = new double[Nb_Hid+1][Nb_Out];
    double NetError[][] = new double[1][Nb_Out];
    double DeltaW1[][] = new double[Nb_In+1][Nb_Hid];
    double DeltaW2[][] = new double[Nb_Hid+1][Nb_Out];
    double W2Target[][]= new double[Nb_Hid][Nb_Out];
    double SumSquare[][]  = new double[1][Nb_Train];
    showstatus();
    // Get time when start training.
    java.util.Date starttrain = new java.util.Date();
    int  Sdate   = starttrain.getDate();
    int  Shour   = starttrain.getHours();
    int  Sminute = starttrain.getMinutes();
    int  Ssecond = starttrain.getSeconds();
    // Networks training with Back-Propagation method.
    // Bias acts like a weigth on the connection from a unit with a constant with a constant of 1.
    In_mt  = ISetting.setbias(1,Nb_In+1);
    Out_Hid_st_mt = ISetting.setbias(1,Nb_Hid+1);
```

```
// Setting the initial weigth with Gaussian random methods.
W1_mt = ISetting.setweigth(Nb_In+1,Nb_Hid);
W2_mt = ISetting.setweigth(Nb_Hid+1,Nb_Out);
W1_old = W1_mt;
W2_old = W2_mt;
Iteration = 0;
SumSquareError    = Permissible+0.01;
// To compute and adjust the weigth connecttion to each layers.
while (SumSquareError > Permissible){
   for (short pat=0;pat<Nb_Train;pat++){
      System.arraycopy(Train_mt[pat],0,In_mt[0],0,Nb_In);
      System.arraycopy(Train_mt[pat],Nb_In,Target[0],0,Nb_Out);
      System.arraycopy(FActivate.sigmoid(GetUtil.matrixoperate(In_mt,W1_mt,'*'))[0],0,Out_Hid_st_mt
        ↳ [0],0,Nb_Hid);
      System.arraycopy(FActivate.sigmoid(GetUtil.matrixoperate(Out_Hid_st_mt,W2_mt,'*'))[0],0,
        ↳ Out_mt[0],0,Nb_Out);
      NetError = BP.neterror(Target,Out_mt);
      DeltaW2  = BP.deltaweigth2(Target,Out_mt,Out_Hid_st_mt);
      W2_new   = BP.weigthadjust(W2_mt,W2_old,DeltaW2);
      W2_old   = W2_mt;
      W2_mt    = W2_new;
      System.arraycopy(Out_Hid_st_mt[0],0,Out_Hid_mt[0],0,Nb_Hid);
      W2Target = GetUtil.copymatrix(W2_mt,W2Target);
      DeltaW1  = BP.deltaweigth1(W2Target,Out_Hid_mt,In_mt,Target,Out_mt);
      W1_new   = BP.weigthadjust(W1_mt,W1_old,DeltaW1);
      W1_old   = W1_mt;
      W1_mt    = W1_new;
      SumSquare[0][pat] = BP.sumsquare(NetError);
   }
  SumSquareError = BP.sumsquareerror(SumSquare,Nb_Train);
  showiteration(Iteration);
  showsumsquareerror(SumSquareError);
  Iteration = Iteration+1;
}
```

```
    // Get time when finish training.
    java.util.Date finishtrain = new java.util.Date();
    int Fdate  = finishtrain.getDate();
    int Fhour  = finishtrain.getHours();
    int Fminute = finishtrain.getMinutes();
    int Fsecond = finishtrain.getSeconds();
    // Time in training process in second.
    int Ddate  = Fdate-Sdate;
    int Dhour  = Fhour-Shour;
    int Dminute = Fminute-Sminute;
    int Dsecond = Fsecond-Ssecond;
    int TotalTime = Dsecond+(Dminute*60)+(Dhour*3600)+(Ddate*86400);
    showtime(TotalTime);
    Active = AFinish;
    showstatus();
}

void analysis(short NoPat){
    // Method to analysis or forecasting.
    double TForecast[] = new double[Nb_Out];
    double TActual[] = new double[Nb_Out];
    double TError[] = new double[Nb_Out];
    double TMax = 0.00;
    double TAvg = 0.00;
    In_mt = ISetting.setbias(1,Nb_In+1);
    Out_Hid_st_mt    = ISetting.setbias(1,Nb_Hid+1);
    for (short pat=0;pat<Nb_Analysis;pat++){
        System.arraycopy(Analysis_mt[pat],0,In_mt[0],0,Nb_In);
        System.arraycopy(Analysis_mt[pat],Nb_In,Actual[pat],0,Nb_Out);
        System.arraycopy(FActivate.sigmoid(GetUtil.matrixoperate(In_mt,W1_mt,'*'))[0],0,
          Out_Hid_st_mt[0],0,Nb_Hid);
        System.arraycopy(FActivate.sigmoid(GetUtil.matrixoperate(Out_Hid_st_mt,W2_mt,'*'))[0],0,
          Forecast[pat],0,Nb_Out);
    }
```

```
    Actual   = GetUtil.denormalize(Actual);
    Forecast = GetUtil.denormalize(Forecast);
    Error    = GetUtil.matrixoperate(Actual,Forecast,'-');
    shownext1hour(Forecast[NoPat][0],Actual[NoPat][0]);
    shownext2hour(Forecast[NoPat][1],Actual[NoPat][1]);
    shownext3hour(Forecast[NoPat][2],Actual[NoPat][2]);
    System.arraycopy(Forecast[NoPat],0,TForecast,0,Nb_Out);
    System.arraycopy(Actual[NoPat],0,TActual,0,Nb_Out);
    System.arraycopy(Error[NoPat],0,TError,0,Nb_Out);
    for (short i=0;i<Nb_Out;i++){
       TError[i] = Math.abs(TError[i]/Actual[NoPat][i])*100;
       if (TError[i] > TMax)
          TMax = TError[i];
    }
    TAvg = GetUtil.summatrix(TError)/3;
    showerror(TAvg,TMax);
    PC.updateplot(TForecast,TActual,NoPat);
    repaint();
    Status = AFinish;
}

class ButtonHandler implements ActionListener{
    public void actionPerformed(ActionEvent e){
       String s=e.getActionCommand();
       if(s == "Load Pattern"){
          Status = SPLoading;
          Active = ADefault;
          clearstatus();
          showstatus();
          Train_mt     = PT.pattotrain(Train_mt);
          Output2Field.setText("          OK!  ");
          Analysis_mt = PA.pattoanalysis(Analysis_mt);
          Output3Field.setText("          OK!  ");
          PLoading = PLPass;
       }
```

```
else if(s == " Training "){
    Status = STraining;
    if (PLoading == PLPass){
        Active = AStart;
        clearstatus();
        showstatus();
        train();
    }
    else
        showwarning();
}
else if(s == " Forecast "){
    Status  = SAnalysis;
    if (PLoading == PLPass){
        clearstatus();
        showstatus();
        analysis(NumPat);
    }
    else
        showwarning();
}
else if(s == "  Next  "){
    Status = SAnalysis;
    if (PLoading == PLPass){
        clearstatus();
        showstatus();
        if (NumPat < 7){
            NumPat++;
            analysis(NumPat);
        }
        else
            showwarning();
    }
```

```
        else if(s == "  Reset  "){
            Status = SDefault;
            Active = ADefault;
            NumPat = No_Pat;
            Actual = new double[Nb_Analysis][Nb_Out];
            Forecast = new double[Nb_Analysis][Nb_Out];
            PC.updateplot();
            clearstatus();
            showstatus();
            repaint();
        }
        else if(s == "Load Weight"){
            clearstatus();
            Status = SWLoading;
            showstatus();
            ToLoadWeight();
        }
        else if(s == "Save Weight"){
            clearstatus();
            Status = SWSaving;
            showstatus();
            ToSaveWeight();
        }
    }
}

ToLoadWeight(){
    try{
        for (short i=1; i<40; i++){
            for ( short j=1;j<78;j+){
                Output2Field.setText("    "+i+" , "+j);
                // Load Oracle driver
                DriverManager.registerDriver (new oracle.jdbc.driver.OracleDriver());
                // Connect to the local database
                Output3Field.setText(" Connecting ");
```

```
            Connection conn =  DriverManager.getConnection ("jdbc:oracle:thin:@161.246.24.245:1521:
              ↳  orcl", "project2", "tamon");
            Output3Field.setText("  Connected  ");
            Output4Field.setText(null);
            Statement stmt = conn.createStatement ();
            ResultSet rset = stmt.executeQuery("select J"+j+" from W1_mt where i = "+i+"");
            W1_mt[i-1][j-1] = rset.getDouble(J"+j+");
            Output4Field.setText("  Loaded  ");
        }
    }
    for (short i=1; i<79; i++){
        for ( short j=1;j<4;j+){
            Output7Field.setText("      "+i+" , "+j);
            // Load Oracle driver
            DriverManager.registerDriver (new oracle.jdbc.driver.OracleDriver());
            // Connect to the local database
            Output9Field.setText(null);
            Output8Field.setText("  Connecting  ");
            Connection conn =  DriverManager.getConnection ("jdbc:oracle:thin:@161.246.24.245:1521:
              ↳  orcl", "project2", "tamon");
            Output8Field.setText("  Connecting  ");
            Statement stmt = conn.createStatement ();
            ResultSet rset = stmt.executeQuery("select J"+j+" from W2_mt where i = "+i+"");
            W2_mt[i-1][j-1] = rset.getDouble(J"+j+");
            Output9Field.setText("  Loaded  ");
        }
    }
    catch(Exception e){
        Output5Field.setText(ex.toString());}
}
```

```
ToSaveWeight(){
   try{
      for (short i=1; i<40; i++){
         for ( short j=1;j<78;j+){
            Output2Field.setText("     "+i+" , "+j);
            // Load Oracle driver
            DriverManager.registerDriver (new oracle.jdbc.driver.OracleDriver());
            // Connect to the local database
            Output4Field.setText(null);
            Output3Field.setText("  Connecting  ");
            Connection conn =  DriverManager.getConnection ("jdbc:oracle:thin:@161.246.24.245:1521:
               ↳  orcl", "project2", "tamon");
            Output3Field.setText("  Connecting  ");
            Statement stmt = conn.createStatement ();
            ResultSet rset = stmt.executeQuery ("update w1 set j"+j+" ="+W1_mt[i-1][j-1]+"where i
               ↳  ="+i+"");
            Output4Field.setText("  Saved  ");
         }
      }
      for (short i=1; i<79; i++){
         for ( short j=1;j<4;j+){
            Output7Field.setText("     "+i+" , "+j);
            // Load Oracle driver
            DriverManager.registerDriver (new oracle.jdbc.driver.OracleDriver());
            // Connect to the local database
            Output9Field.setText(null);
            Output8Field.setText("  Connecting  ");
            Connection conn =  DriverManager.getConnection ("jdbc:oracle:thin:@161.246.24.245:1521:
               ↳  orcl", "project2", "tamon");
            Output8Field.setText("  Connecting  ");
            Statement stmt = conn.createStatement ();
            ResultSet rset = stmt.executeQuery ("update w2 set j"+j+" ="+W2_mt[i-1][j-1]+"where i
               ↳  ="+i+"");
            Output9Field.setText("  Saved  ");
         }
```

```
            }
        catch(Exception e){
            Output5Field.setText(ex.toString());}
    }
    public void paint(Graphics g){
        PC.paint(g);
    }
}

class PlotCurve{
    int Ox,Oy,minxdisp,maxxdisp,minydisp,maxydisp,resX,resY,width,height;
    double  Load[][] = new double[2][3];
    int  TPat;
    int  Curve[][][]  = new int[2][24][2];
    int  Point[][][];
    static final short Pdefault  = 0;   // To assign Active to Default Plotting.
    static final short Pplot  = 1;      // To assign Active to Start Plotting.
    short Plotting = PDefault;          // To assign Variable to select the Plotting Active.

    public PlotCurve(int x_orig,int y_orig,int x_res,int y_res){
        super();
        Ox   = x_orig;      // X-origin transfer 0 to  50
        Oy   = y_orig;      // Y-origin transfer 0 to 280.
        resX  = x_res;      // X-Resolution. = 500
        resY  = y_res;      // Y-Resolution. = 250
        minxdisp = Ox;
        maxxdisp = Ox+resX;
        minydisp = Oy-resY;
        maxydisp = Oy;
        updateplot();
    }

    public void updateplot(){
        Plotting = PDefault;
    }
```

```
public void updateplot(double Fore[],double Actl[],short ToPat){
    Load[0] = Actl;
    Load[1] = Fore;
    TPat  = ToPat+1;
    Point = new int[2][3][2];
    for (short i=0;i<3;i++){
       Point[1][i][0] = topointx(ToPat,i);
       Point[1][i][1] = topointy(Load[1][i]);
       Point[0][i][1] = topointy(Load[0][i]);
    }
    int TempPat = (ToPat*3);
    for (int j=TempPat;j<TempPat+3;j++){
       Curve[1][j] = Point[1][j-TempPat];
       Curve[0][j] = Point[0][j-TempPat];
    }
    Plotting = PPlot;
}

int topointx(short XPat,short XNum){
    int  PointX;
    PointX  = (XPat*3)+XNum+2;
    int Dx = resX / 25;
    PointX  = (PointX*Dx)+Ox;
    return PointX;
}

int topointy(double ToY){
    double  MaxLoad = 15000;
    double    TempY;
    double    dY;
    long      TempLongY;
    int       PointY;
    dY        = (resY/MaxLoad);
    TempY   = dY*ToY;
    Double DOUB      = new Double(TempY);
```

```
    TempLongY       = DOUB.longValue();
    Long  LONG      = new Long(TempLongY);
    PointY          = LONG.intValue();
    PointY          = Oy-PointY;
    return PointY;
}

public void paint(Graphics g){
    // Area to plot length.x = 500
    // Area to plot length.y = 250
    // Oringin(0,0) transfer to 50,30
    g.setColor(Color.black);
    g.drawLine( 50,280,650,280);          // Draw X axis.
    g.drawLine( 50, 30, 50,280); // Draw Y axis.
    g.drawString("Time",350,310);         // To mark X-Axis.
    g.drawString("Load (GW)",18,15);// To mark Y-Axis.
    g.drawString("0",35,295);             // To mark origin point.
    for (short i=1;i<26;i++)
       g.drawLine((50+(i*24)),277,(50+(i*24)),283);// Mark scale x axis.
    for (short i=1;i<13;i++){
       int j = i*2;
       if (j < 10)
          g.drawString(""+j,(47+(j*24)),295);      // Mark scale x axis.
       else
          g.drawString(""+j,(44+(j*24)),295);      // Mark scale x axis.
    }
    g.drawLine( 47, 30, 53, 30);          // Mark scale y axis.
    g.drawLine( 47,155, 53,155);          // Mark scale y axis.
    g.drawString("15",30,35);             // Mark scale x axis.
    g.drawString("7.5",30,160);           // Mark scale x axis.
    g.setColor(Color.red);
    g.drawLine(520,245,530,245);
    g.setColor(Color.blue);
    g.drawLine(520,265,530,265);
    g.setColor(Color.black);
```

```
        g.drawString(" Forecasted Load ",535,250);
        g.drawString(" Actual Load ",535,270);
        int TempCounter = 0;
        if (Plotting == PPlot){
          while (TempCounter < ((TPat-1)*3)+2){
            g.setColor(Color.red);
            g.drawLine(Curve[1][TempCounter][0],Curve[1][TempCounter][1],Curve[1][TempCounter+1]
              [0],Curve[1][TempCounter+1][1]);
            g.setColor(Color.blue);
            g.drawLine(Curve[1][TempCounter][0],Curve[0][TempCounter][1],Curve[1][TempCounter+1]
              [0],Curve[0][TempCounter+1][1]);
            TempCounter++;
          }
        }
      }
    }
```

4.1 Class Activate

```
public class Activate{
   // Activate function that act in the networks.
   // Variable declaration.
   double F_X;          // Function f(x).
   double Gain;         // Used in set up the Sigmoidal function.
   double Matrix[][];

   public Activate(){
      // Initial setting.
      Gain = 1.00;
   }

   double sigmoid(double X){
      // Sigmoidal function.
      F_X = 1/(1+(Math.exp(-X*Gain)));
      return F_X;
   }

   double[][] sigmoid(double ToActivate[][]){
      // Sigmoidal function.
      Matrix = new double[ToActivate.length][ToActivate[0].length];
      for (int i=0;i<ToActivate.length;i++){
         for (int j=0;j<ToActivate[0].length;j++){
            Matrix[i][j] = sigmoid(ToActivate[i][j]);
         }
      }
      return Matrix;
   }
}
```

4.2 Class Back_Prop

```
import Utility;

public class Back_Prop{
   double  LearningRate;          // Used in Back-Propagation training algorithm.
   double  Momentum;              // Used in Back-Propagation training algorithm.
   double  SumSquare;
   double  SumSquareError;
   double  NetError[][];
   double  Error[][];
   double  Delta[][];
   double  Weigth[][];
   Utility  BPUtil;

   public Back_Prop(){
      LearningRate  = 0.25;
      Momentum      = 0.40;
      BPUtil = new Utility();
   }

   double[][] neterror(double Target[][],double Output[][]){
      NetError = BPUtil.matrixoperate(Target,Output,'-');
      return NetError;
   }

   double[][] deltaweigth1(double WeigthTarget[][],double Out_Hid[][],double Input[][],double Target[][],
     ↳ double Output[][]){
      Delta = BPUtil.matrixoperate(BPUtil.transposematrix(Input),BPUtil.matrixoperate
        ↳ (BPUtil.matrix(Out_Hid,LearningRate),errorterm1(WeigthTarget,Out_Hid,Target,Output),'.'),'*');
      return Delta;
   }
```

```
double[][] deltaweigth2(double Target[][],double Output[][],double OutHidden[][]){
    Delta = BPUtil.matrixoperate(BPUtil.transposematrix(OutHidden),BPUtil.matrixoperate
      ↳ (BPUtil.matrix(Target,LearningRate),errorterm2(Target,Output),'.'),'*');
    return Delta;
}

double[][] errorterm1(double WeigthTarget[][],double Out_Hid[][],double Target[][],double Output[][]){
    Error = BPUtil.matrixoperate(BPUtil.matrixoperate(Out_Hid,BPUtil.matrixoperate
      ↳ (BPUtil.matrix(Out_Hid,1),Out_Hid,'-'),'.'),BPUtil.matrixoperate(errorterm2(Target,Output)
      ↳ ,BPUtil.transposematrix(WeigthTarget),'*'),'.');
    return Error;
}

double[][] errorterm2(double Target[][],double Output[][]){
    Error = BPUtil.matrixoperate(BPUtil.matrixoperate(Output,BPUtil.matrixoperate
      ↳ ((BPUtil.matrix(Output,1)),Output,'-'),'.'),neterror(Target,Output),'.');
    return Error;
}

double[][] weigthadjust(double Weigth_mt[][],double Weigth_old[][],double Delta_mt[][]){
    Weigth = new double[Weigth_mt.length][Weigth_mt[0].length];
    Weigth = BPUtil.matrixoperate((BPUtil.matrixoperate(Weigth_mt,Delta_mt,'+')),(BPUtil.matrixoperate
      ↳ ((BPUtil.matrix(Weigth,Momentum)),(BPUtil.matrixoperate(Weigth_mt,Weigth_old,'-')),'.')),'+');
    return Weigth;
}

double sumsquare(double NetError[][]){
    SumSquare = BPUtil.summatrix(BPUtil.matrixoperate(NetError,NetError,'.'));
    return SumSquare;
}

double sumsquareerror(double SumSquare[][],short NumOfPattern){
    SumSquareError = BPUtil.summatrix(SumSquare)/(2*NumOfPattern);
    return SumSquareError;
}
}
```

4.3 Class InitSetting

```
import java.util.*;

public class InitSetting{
   // Initial setting the networks.
   // Variable declaration.
   double Bias[][];          // Bias setting matrix.
   double Bias_Act;
   double Weigth[][];        // Initial weigth random matrix.
   double Temp;
   double Max;
   double Multiple;          // Used in deviation the initial weigth random.
   Random GRandom;

   public InitSetting(){
      // Initial setting.
      Bias_Act = 1;
      Multiple = 1;
      GRandom  = new Random();
   }

   double[][] setbias(int Msizes,int Nsizes){
      // Bias setting to the networks.
      Bias = new double[Msizes][Nsizes];
      for (short i=0;i<Msizes;i++){
         for (short j=0;j<Nsizes;j++){
            Bias[i][j]  = Bias_Act;
         }
      }
      return Bias;
   }
```

```
double[][] setweigth(int Msizes,int Nsizes){
    // Random the initial weigth with Gaussian random method.
    Weigth = new double[Msizes][Nsizes];
    for (short i=0;i<Msizes;i++){
        for (short j=0;j<Nsizes;j++){
            Weigth[i][j] = Multiple*(GRandom.nextGaussian());
            Temp = Math.abs(Weigth[i][j]);
            if (Temp > Max)
                Max = Temp;
        }
    }
    for (short i=0;i<Msizes;i++){
        for (short j=0;j<Nsizes;j++)
            Weigth[i][j] = Weigth[i][j]/Max;
    }
    Max = 0;
    return Weigth;
  }
}
```

4.4 Class Utility

```
public class Utility{
   double MaxTemp;      // To lomit maximun temperature in normalization.
   double MaxLoad;      // To lomit maximun load in normalization.
   double Normal;       // Used in normalization.
   double Denormal[][]; // Used in denormalization.
   double C_mt[][];
   double D_mt[];
   double Temp;

   public Utility(){
      MaxTemp = 500;
      MaxLoad = 20000;
   }

   double[][] matrix(double A_mt[][],double Value){
      C_mt = new double[A_mt.length][A_mt[0].length];
      for (short i=0;i<A_mt.length;i++){
         for (short j=0;j<A_mt[0].length;j++)
            C_mt[i][j] = Value;
      }
      return C_mt;
   }

   double[][] matrixoperate(double A_mt[][],double B_mt[][],char Type){
      if (Type != '*'){
         C_mt = new double[A_mt.length][A_mt[0].length];
         for (short i=0;i<A_mt.length;i++){
            for (short j=0;j<A_mt[0].length;j++){
               if (Type == '.')
                  C_mt[i][j] = A_mt[i][j]*B_mt[i][j];
               else if (Type == '+')
                  C_mt[i][j] = A_mt[i][j]+B_mt[i][j];
               else if (Type == '-')
                  C_mt[i][j] = A_mt[i][j]-B_mt[i][j];
```

```
            }
        }
    }
    else if (Type == '*'){
        C_mt = new double[A_mt.length][B_mt[0].length];
        Temp  = 0;
        for (short i=0;i<A_mt.length;i++){
            for (short j=0;j<B_mt[0].length;j++){
                for (short k=0;k<B_mt.length;k++){
                    C_mt[i][j] = (A_mt[i][k]*B_mt[k][j])+Temp;
                    Temp = C_mt[i][j];
                }
                Temp = 0;
            }
        }
    }
    return C_mt;
}

double[][] transposematrix(double A_mt[][]){
    C_mt = new double[A_mt[0].length][A_mt.length];
    for (short i=0;i<A_mt.length;i++){
        for (short j=0;j<A_mt[0].length;j++)
            C_mt[j][i] = A_mt[i][j];
    }
    return C_mt;
}

double summatrix(double A_mt[][]){
    Temp = 0;
    for (short i=0;i<A_mt.length;i++){
        for (short j=0;j<A_mt[0].length;j++)
            Temp = A_mt[i][j]+Temp;;
    }
    return Temp;
}
```

```
double summatrix(double A_mt[]){
    Temp = 0;
    for (short i=0;i<A_mt.length;i++)
        Temp = A_mt[i]+Temp;;
    return Temp;
}

double[][] copymatrix(double A_mt[][],double B_mt[][]){
    C_mt = new double[B_mt.length][B_mt[0].length];
    for (short i=0;i<B_mt.length;i++)
        System.arraycopy(A_mt[i],0,C_mt[i],0,B_mt[0].length);
    return C_mt;
}

double normalize(double Input,short Number){
    if (Number < 4 || Number >= 38)
        Normal = Input/MaxLoad;
    else if (Number >= 4 & Number < 18)
        Normal = Input/MaxTemp;
    else if (Number >= 18 & Number < 38)
        Normal = Input;
    else if (Number >= 38)
        Normal = Input/MaxLoad;
    return Normal;
}

double[][] denormalize(double Output[][]){
    Denormal = new double[Output.length][Output[0].length];
    for (short i=0;i<Output.length;i++){
        for (short j=0;j<Output[0].length;j++)
            Denormal[i][j] = Output[i][j]*20000;
    }
    return Denormal;
}
}
```

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สามารถสำเร็จลุล่วงไปได้ด้วยดี ทั้งนี้เนื่องจาก
รศ. ศิริวัฒน์ โพธิเวชกุล อาจารย์ที่ปรึกษาโครงงาน ที่ได้ให้ความช่วยเหลือมาโดยตลอด
คุณพ่อและคุณแม่ ที่ได้ให้กำลังใจแก่ลูกเสมอมา
อ. คุปต์ โพธ์แก้ว ที่ได้แนะแนวทางในการทำโครงงานและเอื้อเฟื้อข้อมูล
คุณฉันทกร จำศิลป์ ที่ได้ให้คำปรึกษาและให้กำลังใจมาโดยตลอด
คุณวิสันต์ ตั้งวงษ์เจริญ ที่ได้ให้คำแนะนำและคำปรึกษาในเรื่องระบบฐานข้อมูล
เพื่อนๆ ที่คอยเป็นกำลังใจ
กรมอุตุนิยมวิทยา ที่ได้เอื้อเฟื้อข้อมูล
การไฟฟ้าฝ่ายผลิต ที่ได้เอื้อเฟื้อข้อมูล
ซึ่งทางคณะผู้จัดทำขอขอบคุณมา ณ. ที่นี้

คณะผู้จัดทำ

บรรณานุกรม

[1] Anne–Johan Annema, " Feed-Forward Neural Networks; Vector Decomposition Analysis and Analog Implementation ", Kluwer Academic Publishers, 1995

[2] Kevin Warwick, Raj Aggarwal and Arthur Ekwue, " Artificial Intelligence Techniques " , IEEE Power Engineering Series 22, London; Institution of Electrical Engineers, 1997

[3] Sullivan, R.L " Power System Planning ", McGraw-Hill, New York, 1977

[4] M.V. Deshpande, " Element of Electrical Power Station Design ", Wheeler Publishing, 1986

[5] George Gross and Francisco D. Galiana, " Short – Term Load Forecasting " , Proceeding of the IEEE, Vol. 75, No. 12, pp. 1558 - 1570 , 1987

[6] เถลิง วัฒนมงคลและฉันทกร จำศิลป์, " การวิเคราะห์หาตำแหน่งที่เกิดความผิดปกติในระบบไฟฟ้ากำลังโดยใช้โครงข่ายประสาทเทียม ", ปริญญานิพนธ์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2539

[7] นิพนธ์ สุทธิภูพันธ์และปฐมรัฐ สูตะบุตร, " การประยุกต์คอมพิวเตอร์ในการทำนายความต้องการทางไฟฟ้าของประเทศไทย ", ปริญญานิพนธ์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2531

[8] ดร. สำรวย สังข์สะอาด, " วิศวกรรมไฟฟ้าแรงสูง", ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย, 2528