การเปรียบเทียบวิธีการรวมตัวแบบการถดถอยเชิงเส้นพหุคูณ

นายธนาพันธ์ อัครเดชากร

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาสถิติศาสตรมหาบัณฑิต
สาขาวิชาสถิติ ภาควิชาสถิติ
คณะพาณิชยศาสตร์และการบัญชี จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2550

A COMPARISON OF MODEL COMBINING METHODS FOR MULTIPLE LINEAR REGRESSION MODELS

Mr. Thanaphan Akkhradechakon

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Statistics

Department of Statistics
Faculty of Commerce and Accountancy
Chulalongkorn University

Academic Year 2007

หัวข้อวิทยานิพนธ์	การเปรียบเทียบวิธีการรวมตัวแบบการถดถอยเชิงเส้นพหุคูณ
โดย	นายธนาพันธ์ อัครเดชากร
สาขาวิชา	สถิติ
อาจารย์ที่ปรึกษา	รองศาสตราจารย์ ร้อยเอก มานพ วราภักดิ์

คณะพาณิชยศาสตร์และการบัญชี จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

....คณบดีคณะพาณิชยศาสตร์และการบัญชี

(รองศาสตราจารย์ ดร.อรรณพ ตันละมัย)

คณะกรรมการสอบวิทยานิพนธ์

....ประธานกรรมการ

(รองศาสตราจารย์ ดร.กัลยา วานิชย์บัญชา)

....อาจารย์ที่ปรึกษา

(รองศาสตราจารย์ ร้อยเอก มานพ วราภักดิ์)

....กรรมการ

(รองศาสตราจารย์ ดร.ธีระพร วีระถาวร)

ธนาพันธ์ อัครเดชากร : การเปรียบเทียบวิธีการรวมตัวแบบการถดถอยเชิงเส้นพหุคูณ (A COMPARISON OF MODEL COMBINING METHODS FOR MULTIPLE LINEAR REGRESSION MODELS) อ. ที่ปรึกษา : รศ.ร.อ. มานพ วราภักดิ์, 149 หน้า.

การวิจัยครั้งนี้ มีวัตถุประสงค์เพื่อเปรียบเทียบวิธีหาค่าพยากรณ์ร่วมของตัวแบบการถดถอยเชิงเส้นพหุคูณ 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยตัวแบบที่นำมาหาค่าพยากรณ์ร่วมได้แก่ ตัวแบบที่ได้จากวิธีพิจารณาการถดถอยทุกรูปแบบ วิธีคัดเลือกตัวแปรแบบไปข้างหน้า วิธีกำจัดตัวแปรแบบถอยหลัง และวิธีการถดถอยขั้นบันได ซึ่งเกณฑ์ที่ใช้ในการตัดสินใจ คือ เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) จำนวนตัวแปรอิสระที่ศึกษาคือ 3, 5 และ 7 ตัว เมื่อจำนวนตัวแปรอิสระเท่ากับ 3 กำหนดระดับความสัมพันธ์ระหว่างตัวแปรอิสระ x_1 กับ x_2 เท่ากับ 0.3, 0.5 และ 0.8 โดยศึกษาเมื่อขนาดตัวอย่างเท่ากับ 14, 20, 30, 40 และ 50 เมื่อจำนวนตัวแปรอิสระเท่ากับ 5 กำหนดระดับความสัมพันธ์ระหว่าง x_1 กับ x_2 และ x_4 กับ x_5 เท่ากับ (0.3, 0.3), (0.4, 0.6) และ (0.7, 0.9) โดยศึกษาเมื่อขนาดตัวอย่างเท่ากับ 20, 30, 40 และ 50 เมื่อจำนวนตัวแปรอิสระเท่ากับ 7 กำหนดระดับความสัมพันธ์ระหว่าง x_1 กับ x_2, x_4 กับ x_5 และ x_6 กับ x_7 เท่ากับ (0.3, 0.3, 0.3), (0.4, 0.5, 0.6) และ (0.7, 0.8, 0.9) โดยศึกษาเมื่อขนาดตัวอย่างเท่ากับ 30, 40 และ 50 ความคลาดเคลื่อนมีการแจกแจงปกติที่มีค่าเฉลี่ยเท่ากับ 0 และส่วนเบี่ยงเบนมาตรฐานเท่ากับ 5 วิธีการวิจัยใช้การจำลองด้วยเทคนิคมอนติคาร์โลซึ่งกระทำซ้ำ 1,000 รอบในแต่ละสถานการณ์

ผลการวิจัยปรากฏว่าปัจจัยที่มีผลต่อค่าเฉลี่ยของ MAPE ของทุกวิธี คือ ระดับความสัมพันธ์ระหว่างตัวแปรอิสระและขนาดตัวอย่าง โดยค่าเฉลี่ยของ MAPE จะมีแนวโน้มเพิ่มขึ้นเมื่อระดับความสัมพันธ์สูงขึ้น และมีแนวโน้มลดลงเมื่อขนาดตัวอย่างเพิ่มขึ้น

จากการเปรียบเทียบค่าเฉลี่ยของ MAPE ของการหาค่าพยากรณ์ร่วม 3 วิธี พบว่า วิธี BO มีประสิทธิภาพมากที่สุดในทุกกรณีที่ศึกษา และโดยทั่วไปวิธีพยากรณ์เดี่ยวที่ได้รับน้ำหนักมากที่สุดจะขึ้นอยู่กับระดับพหุสัมพันธ์ระหว่างตัวแปรอิสระ ดังนี้

กรณีที่พหุสัมพันธ์ระหว่างตัวแปรอิสระอยู่ในระดับต่ำ วิธีพิจารณาการถดถอยทุกรูปแบบจะได้รับน้ำหนักมากที่สุด

กรณีที่พหุสัมพันธ์ระหว่างตัวแปรอิสระอยู่ในระดับปานกลาง วิธีกำจัดตัวแปรแบบถอยหลังจะได้รับน้ำหนักมากที่สุด

กรณีที่พหุสัมพันธ์ระหว่างตัวแปรอิสระอยู่ในระดับสูง วิธีการถดถอยขั้นบันไดจะได้รับน้ำหนักมากที่สุด

ภาควิชา......................สถิติ.................... ลายมือชื่อนิสิต ธนาพันธ์ อัครเดชากร

สาขาวิชา....................สถิติ.................... ลายมือชื่ออาจารย์ที่ปรึกษา ร.อ. [signature]

ปีการศึกษา.............. 2550................

4782251426 : MAJOR STATISTICS

KEY WORD : LINEAR REGRESSION / MODEL COMBINING / LEAST ABSOLUTE ERRORS / COMBINATION BY BOOTSTRAP / ADAPTIVE REGRESSION BY MIXING

THANAPHAN AKKHRADECHAKON : A COMPARISON OF MODEL COMBINING METHODS FOR MULTIPLE LINEAR REGRESSION MODELS. THESIS ADVISOR : ASSOC. PROF. CAPT. MANOP VARAPHAKDI, M.S.149 p.

The objective of this research is to compare model-combining methods for multiple linear regression models. Three combining methods are studied: least absolute errors (LAE), combination by bootstrap (BO) and adaptive regression by mixing (ARM). The models used in combining are built by the following procedures: all possible regressions, forward selection, backward elimination and stepwise regression. The mean absolute percentage error (MAPE) is used as the criterion in deciding which combining method is best. The number of independent variables are 3, 5 and 7. In 3-variable case, the correlations between independent variables x_1 and x_2 are 0.3, 0.5 and 0.8; the sample sizes are 14, 20, 30, 40 and 50. In 5-variable case, the correlations between x_1 and x_2, and between x_4 and x_5 are (0.3, 0.3), (0.4, 0.6) and (0.7, 0.9); the sample sizes are 20, 30, 40 and 50. In 7-variable case, the correlations between x_1 and x_2, between x_4 and x_5, and between x_6 and x_7 are (0.3, 0.3, 0.3), (0.4, 0.5, 0.6) and (0.7, 0.8, 0.9); the sample sizes are 30, 40 and 50. The random errors are normally distributed with mean 0 and standard deviation 5. This research used the Monte Carlo simulation, repeated 1,000 times in each situation.

The results of this research show that factors affecting the average of MAPE for all combining methods are correlations among the independent variables and sample sizes. The average of MAPE tends to increase when the correlation increases, and to fall when the sample size increases.

By comparing the average of MAPE for the three combining methods, the researcher concludes that BO method is the best in every case. Generally, the model-building procedure which receives the maximum weight depends on the degree of multicollinearity among the independent variables.

When multicollinearity is low, all possible regression receives the maximum weight.

When multicollinearity is medium, backward elimination receives the maximum weight.

When multicollinearity is high, stepwise regression receives the maximum weight.

Department............Statistics..........

Field of study..........Statistics..........

Academic year.........2007..............

Student's signature....ธนพันธ์ อัครเดชากร....

Advisor's signature.....

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยความช่วยเหลืออย่างดียิ่งของ รองศาสตราจารย์ ร้อยเอกมานพ วราภักดิ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่กรุณาให้คำแนะนำและข้อคิดเห็นต่าง ๆ ในการวิจัยมาด้วยดีตลอด ผู้วิจัยใคร่ขอกราบขอบพระคุณเป็นอย่างยิ่ง

ผู้วิจัยขอกราบขอบพระคุณคณะกรรมการสอบวิทยานิพนธ์ซึ่งประกอบด้วย รองศาสตราจารย์ ดร.กัลยา วานิชย์บัญชา ผู้เป็นประธานกรรมการ และรองศาสตราจารย์ ดร.ธีระพร วีระถาวร ผู้เป็นกรรมการ ที่ช่วยตรวจสอบและแก้ไขวิทยานิพนธ์ฉบับนี้ให้สมบูรณ์ยิ่งขึ้น

ท้ายนี้ ผู้วิจัยขอกราบขอบพระคุณ บิดา มารดา ตลอดจนทุกคนในครอบครัว ซึ่งสนับสนุนและให้กำลังใจเสมอมาจนสำเร็จการศึกษา และขอขอบคุณเพื่อน ๆ ที่คอยให้กำลังใจมาโดยตลอด

สารบัญ

หน้า

สารบัญตาราง

ตารางที่ หน้า

ตารางที่ หน้า

สารบัญภาพ

ภาพที่ หน้า

บทที่ 1

บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในการวิเคราะห์การถดถอย (regression analysis) มักมีตัวแบบที่ต้องพิจารณามากกว่าหนึ่งตัวแบบ โดยทั่วไปผู้วิเคราะห์จะทำการเลือกตัวแบบ (model selection) เพื่อเลือกตัวแบบที่ดีที่สุดออกมาเพียงตัวแบบเดียว และตัวแบบที่ได้จากกระบวนการดังกล่าวจะถูกนำไปใช้ในการประมาณค่าและการพยากรณ์ต่อไป

พิจารณาตัวแบบการถดถอยรูปทั่วไปที่มีค่าสังเกต n ค่า ดังต่อไปนี้

$$y_i = f(\mathbf{x}_i) + \varepsilon_i, \quad i = 1, \dots, n \tag{1.1}$$

เมื่อ y_i เป็นตัวแปรตาม

$\mathbf{x}_i = (x_{i1}, \dots, x_{ip})$ เป็นเวกเตอร์ของตัวแปรอิสระ p ตัว

$f(\cdot)$ เป็นฟังก์ชันการถดถอย

ε_i เป็นความคลาดเคลื่อนสุ่ม

ในกรณีที่ผู้วิเคราะห์มีสมมติฐานว่าตัวแปรอิสระแต่ละตัวและตัวแปรตามมีความสัมพันธ์กันในลักษณะเชิงเส้น ตัวแบบใน (1.1) จะอยู่ในรูปของตัวแบบการถดถอยเชิงเส้นพหุคูณ (multiple linear regression model) ซึ่งในกรณีนี้จะสามารถเขียนตัวแบบ (1.1) ใหม่ได้เป็น

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} + \varepsilon_i, \quad i = 1, \dots, n \tag{1.2}$$

เมื่อ $\beta_0, \beta_1, \beta_2, \dots, \beta_p$ เป็นสัมประสิทธิ์การถดถอย

หรือเขียนในรูปของเมทริกซ์ได้ดังนี้

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{1.3}$$

โดยที่ $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$, $\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1p} \\ 1 & x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}$,

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}, \qquad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

ภายใต้สมมติฐานนี้ ปัญหาการเลือกตัวแบบที่ดีที่สุดเพื่อใช้ในการพยากรณ์จะเหลือเพียงปัญหาการคัดเลือกตัวแปรอิสระ (variables selection) เท่านั้น ซึ่งโดยทั่วไปวิธีการคัดเลือกตัวแปรอิสระเข้าสมการการถดถอยสามารถจำแนกได้เป็น 2 แนวทางใหญ่ ๆ ดังนี้

1. วิธีพิจารณาสมการการถดถอยทุกรูปแบบ (all possible regressions) แล้วจึงใช้ค่าสถิติหรือเกณฑ์ในการเลือกตัวแบบที่เหมาะสมที่สุดออกมา ซึ่งค่าสถิติหรือเกณฑ์ที่นิยมใช้ในวิธีนี้ได้แก่ ค่าสัมประสิทธิ์การตัดสินใจ (R^2) ค่าสัมประสิทธิ์การตัดสินใจที่ปรับแล้ว (R^2_{adj}) ค่าเฉลี่ยของความคลาดเคลื่อนกำลังสอง (mean square error: MSE) ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (mean absolute percentage error: MAPE) ค่าสถิติของมอลโลวส์ (Mallows' C_p) เกณฑ์ข้อสนเทศของอะกะอิเกะ (Akaike information criterion: AIC) และเกณฑ์ข้อสนเทศของเบส์ (Bayes information criterion: BIC) เป็นต้น
2. การใช้วิธีคัดเลือกตัวแปรอิสระโดยอัตโนมัติ (automatic variable selection procedure) วิธีนี้จะทำการเพิ่มหรือลดตัวแปรอิสระจากตัวแบบการถดถอยตามขั้นตอนวิธีที่กำหนด ซึ่งวิธีที่นิยมใช้ได้แก่ วิธีคัดเลือกตัวแปรแบบไปข้างหน้า (forward selection) วิธีกำจัดตัวแปรแบบถอยหลัง (backward elimination) และวิธีการถดถอยขั้นบันได (stepwise regression) เป็นต้น

เมื่อนำข้อมูล $(\mathbf{x}_i, y_i), i = 1, 2, ..., n$ มาสร้างตัวแบบการถดถอยเชิงเส้นด้วยวิธีต่างๆ ข้างต้น บ่อยครั้งเราจะพบว่าวิธีการเหล่านี้ให้ตัวแบบที่ดีที่สุดออกมาต่างกัน สมมติว่าตัวแบบที่สร้างได้มีทั้งหมด m ตัวแบบ โดยมีตัวแบบที่ k คือ

$$\hat{\mathbf{y}}^{(k)} = \mathbf{X}^{(k)}\mathbf{b}^{(k)}; \quad k = 1, \ldots, m \tag{1.4}$$

เมื่อ $\hat{\mathbf{y}}^{(k)}$ เป็นเวกเตอร์ค่าพยากรณ์ของตัวแปรตามในตัวแบบที่ k

$\mathbf{X}^{(k)}$ เป็นเมทริกซ์ของตัวแปรอิสระในตัวแบบที่ k

$\mathbf{b}^{(k)}$ เป็นเวกเตอร์ตัวประมาณของสัมประสิทธิ์การถดถอยในตัวแบบที่ k ซึ่งประมาณได้จากข้อมูลตัวอย่าง

หากผู้วิเคราะห์นำเพียงตัวแบบใดตัวแบบหนึ่งจาก m ตัวแบบข้างต้นมาใช้ในการพยากรณ์ ค่าพยากรณ์ที่ได้อาจไม่มีเสถียรภาพ (unstable) กล่าวคือ หากข้อมูลมีการเปลี่ยนแปลงแม้เพียงเล็กน้อยก็อาจส่งผลให้วิธีการต่าง ๆ ที่กล่าวมานั้น สร้างตัวแบบออกมาต่างจากเดิมมาก นอกจากนี้ผู้วิเคราะห์อาจตัดสินใจไม่ได้ว่าจะเลือกตัวแบบใดจาก m ตัวแบบจึงจะให้ค่าพยากรณ์ที่ถูกต้องและแม่นยำที่สุด

โดยทั่วไปการขาดเสถียรภาพ (instability) หรือความไม่แน่นอน (uncertainty) ของค่าพยากรณ์ที่ได้จากการเลือกตัวแบบใดตัวแบบหนึ่งเพียงตัวแบบเดียวมักจะเกิดขึ้นในกรณีต่อไปนี้ (Yang 2003; Yuan and Yang, 2005)

1. ความแปรปรวนของความคลาดเคลื่อนมีค่าสูง
2. ข้อมูลตัวอย่างที่ใช้สร้างสมการการถดถอยมีจำนวนน้อย
3. เกิดปัญหาพหุสัมพันธ์ระหว่างตัวแปรอิสระ (multicollinearity)

ในกรณีที่จุดมุ่งหมายของการวิเคราะห์การถดถอยคือการพยากรณ์ค่าของตัวแปรตาม ผู้วิเคราะห์สามารถลดปัญหาการขาดเสถียรภาพนี้ลงได้ โดยใช้วิธีการรวมตัวแบบ (model-combining method) หรือเรียกอีกอย่างว่า การหาค่าพยากรณ์ร่วม ซึ่งจากงานวิจัยหลายๆ ชิ้นพบว่า ค่าพยากรณ์ร่วมที่ได้มักจะมีความแม่นยำกว่าค่าพยากรณ์จากตัวแบบเดียว ในที่นี้จะพิจารณาค่าพยากรณ์ร่วมซึ่งเกิดจากการรวมเชิงเส้นของค่าพยากรณ์จากแต่ละตัวแบบ ดังนี้

$$\hat{y}_i^{combined} = \sum_{k=1}^{m} w^{(k)} \hat{y}_i^{(k)} \quad , \quad i = 1, \ldots, n \tag{1.5}$$

เมื่อ $\hat{y}_i^{combined}$ คือค่าพยากรณ์ร่วมของตัวแปรตาม

$w^{(k)}$ คือค่าน้ำหนักที่ให้กับตัวแบบที่ $k,\ k = 1, \ldots, m$

$\hat{y}_i^{(k)}$ คือค่าพยากรณ์ของตัวแปรตามจากตัวแบบที่ $k,\ k = 1, \ldots, m$

จากสมการ (1.5) ปัญหาที่สำคัญคือ จะกำหนดค่าน้ำหนัก $w^{(k)}$ ให้กับแต่ละตัวแบบอย่างไรจึงจะได้ผลลัพธ์ที่ดีที่สุด วิธีการหนึ่งที่เป็นไปได้ คือ การวิเคราะห์การถดถอยระหว่าง y_i กับ $\hat{y}_i^{(k)}$, $k = 1, \ldots, m$ โดยใช้วิธีกำลังสองน้อยสุด (least squares) แต่ค่าน้ำหนักที่ได้จากวิธีนี้จะไม่ดีนัก เนื่องจากไม่ได้คำนึงถึงสหสัมพันธ์ระหว่าง $\hat{y}_i^{(k)}$, $k = 1, \ldots, m$ จากแต่ละตัวแบบ ซึ่งโดยทั่วไปจะมีค่าค่อนข้างสูง ทั้งนี้เพราะ $\hat{y}_i^{(k)}$ ของทุกตัวแบบต่างก็สร้างมาจากข้อมูลชุดเดียวกัน และยิ่งไปกว่านั้น ในกรณีที่ตัวแบบเชิงเส้นทั้งหมดที่พิจารณามีลักษณะติดกลุ่ม (nested set of linear models) การหาค่าพยากรณ์ร่วมโดยใช้วิธีกำลังสองน้อยสุดนี้จะให้ค่าน้ำหนัก $w^{(k)}$ เท่ากับ 1 สำหรับค่าพยากรณ์จากตัวแบบที่ใหญ่ที่สุด และเท่ากับ 0 สำหรับค่าพยากรณ์จากตัว

แบบอื่นๆ ผลสุดท้ายคือเราจะได้ค่าพยากรณ์ร่วมเป็นค่าพยากรณ์จากตัวแบบที่ใหญ่ที่สุดเท่านั้น (LaBlanc and Tibshirani, 1996)

ด้วยเหตุนี้จึงมีผู้เสนอวิธีต่างๆ เพื่อใช้หาน้ำหนักของการรวมตัวแบบให้มีประสิทธิภาพ เช่น

- วิธี stacking (Wolpert, 1992; Breiman 1996)
- วิธีบูตสแตรป (LeBlanc and Tibshirani, 1996)
- วิธี Baysian model averaging (BMA) (Hoeting et al., 1997)
- วิธี adaptive regression by mixing (ARM) (Yang 2001; 2003)
- วิธี frequentist model averaging (FMA) (Hjort and Claeskens, 2003)

ในปี ค.ศ. 1996 อมรรัตน์ ปรารมภ์ ได้ทำวิทยานิพนธ์เกี่ยวกับการหาค่าพยากรณ์ร่วม (combined forecasts) ในกรณีที่ข้อมูลเป็นแบบอนุกรมเวลา อมรรัตน์ได้เปรียบเทียบวิธีการหาค่าน้ำหนัก 3 วิธี ได้แก่ วิธีการการให้น้ำหนักที่เท่ากัน (simple averages method) วิธีการของ Bates, Ganger และ Newbold (BGN’s method) และวิธีค่าสัมบูรณ์ต่ำสุด (least absolute errors method) อมรรัตน์พบว่า การใช้ค่าพยากรณ์ร่วมจะให้ผลดีในกรณีที่ข้อมูลมีการเคลื่อนไหวในระดับค่าเฉลี่ยหรือในลักษณะแนวโน้มเชิงเส้น และจากการเปรียบเทียบวิธีการหาค่าน้ำหนักทั้ง 3 วิธี พบว่า วิธีหาค่าพยากรณ์ร่วมโดยวิธีค่าสัมบูรณ์ต่ำสุดเป็นวิธีที่เหมาะสมที่สุด

ในปี ค.ศ. 2003 Clarke ได้ทำการเปรียบเทียบวิธี BMA และวิธี stacking พบว่า ในกรณีส่วนใหญ่ วิธี stacking จะมีประสิทธิภาพมากกว่าวิธี BMA

ในปี ค.ศ. 2005 Yuan และ Yang ได้ทำการเปรียบเทียบวิธี ARM กับวิธี BMA และพบว่าวิธี ARM ให้ค่าพยากรณ์ที่ถูกต้องและแม่นยำกว่าวิธี BMA ในกรณีที่ตัวแบบมีขนาดปานกลางถึงใหญ่และความแปรปรวนของความคลาดเคลื่อนมีค่าปานกลางถึงสูง

ในการวิจัยครั้งนี้ ผู้วิจัยสนใจศึกษาการหาค่าพยากรณ์ร่วมของตัวแบบการถดถอยเชิงเส้นพหุคูณในกรณีที่ตัวแปรอิสระมีพหุสัมพันธ์กัน ณ ระดับต่างๆ โดยจะทำการเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ดังนี้

1. วิธีค่าสัมบูรณ์ต่ำสุด (least absolute errors)

วิธีนี้จะอาศัยเทคนิคการโปรแกรมเชิงเส้น (linear programming) โดยมีหลักการคือทำให้ผลรวมของค่าสัมบูรณ์ของความคลาดเคลื่อนมีค่าต่ำสุด

2. วิธีบูตสแตรป (combination by bootstrap)

วิธีนี้ใช้หลักการของวิธีกำลังสองน้อยสุด แต่จะนำวิธีบูตสแตรป เข้ามาช่วยในกระบวนการหาน้ำหนักด้วย เพื่อลดความเอนเอียงของการประมาณค่าน้ำหนัก

3. วิธี adaptive regression by mixing (ARM)

วิธีนี้เริ่มจากการแบ่งข้อมูลตัวอย่างเป็นสองส่วนเท่าๆ กัน โดยข้อมูลส่วนแรกจะถูกนำมาประมาณค่าพารามิเตอร์ของตัวแบบในกลุ่มที่พิจารณาใหม่ หลังจากนั้นจะนำตัวแบบที่ได้ใหม่นี้มาพยากรณ์ค่าตัวแปรตามของข้อมูลตัวอย่างส่วนที่เหลืออีกครึ่งหนึ่ง จากนั้นจะประเมินความแม่นยำของการพยากรณ์ที่ได้ (โดยการเปรียบเทียบค่าพยากรณ์กับค่าจริง) และนำผลการประเมินดังกล่าวมาใช้คำนวณหาค่าน้ำหนัก

เหตุที่ผู้วิจัยสนใจเปรียบเทียบวิธีการทั้ง 3 ที่กล่าวมาข้างต้น เนื่องจากที่ผ่านมา ยังไม่มีงานวิจัยที่ทำการศึกษาเปรียบเทียบวิธีบูตสแตรปกับวิธีอื่นๆ มาก่อน วิธี ARM เป็นวิธีการใหม่ที่มีคุณสมบัติที่ดีในทางทฤษฎี ส่วนวิธีค่าสัมบูรณ์ต่ำสุดเป็นวิธีที่ไม่ยุ่งยากเพราะสามารถใช้โปรแกรมสำเร็จรูปในการคำนวณหาน้ำหนักได้

1.2 วัตถุประสงค์ของการวิจัย

การวิจัยครั้งนี้มีวัตถุประสงค์ดังนี้

1. เพื่อศึกษาวิธีหาค่าพยากรณ์ร่วมของตัวแบบการถดถอยเชิงเส้นพหุคูณ 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด วิธีบูตสแตรป และวิธี adaptive regression by mixing
2. เพื่อหาข้อสรุปว่าวิธีหาค่าพยากรณ์ร่วมวิธีใดใน 3 วิธีข้างต้น เป็นวิธีการที่ดีที่สุด (ให้ค่าพยากรณ์ที่ถูกต้องที่สุด) ภายใต้สถานการณ์ต่างๆ

1.3 สมมติฐานการวิจัย

การวิจัยครั้งนี้ มีสมมติฐานคือ การหาค่าพยากรณ์ร่วมของตัวแบบการถดถอยเชิงเส้นพหุคูณโดยวิธีบูตสแตรป จะให้ค่าพยากรณ์ที่มีความถูกต้องและแม่นยำกว่าวิธีอื่นๆ

1.4 ข้อตกลงเบื้องต้น

การวิจัยครั้งนี้มีข้อตกลงดังนี้

1. ตัวแบบที่ศึกษาเป็นตัวแบบการถดถอยเชิงเส้นพหุคูณซึ่งเป็นเชิงเส้นทั้งในพารามิเตอร์และในตัวแปรอิสระ
2. ความคลาดเคลื่อนสุ่ม (ε_i) มีการแจกแจงแบบปกติที่มีค่าเฉลี่ยเป็นศูนย์และมีความแปรปรวนคงที่ และ ε_i แต่ละตัวเป็นอิสระต่อกัน

1.5 ขอบเขตของการวิจัย

ผู้วิจัยกำหนดขอบเขตของการวิจัยดังนี้

1. จำนวนตัวแปรอิสระ (p) ที่ใช้ในการวิจัยมี 3 ระดับ คือ 3, 5 และ 7 ตัว
2. กำหนดขนาดตัวอย่าง (n) ดังนี้

   กรณีที่มีตัวแปรอิสระ 3 ตัว ศึกษาเมื่อขนาดตัวอย่าง = 14, 20, 30, 40 และ 50

   กรณีที่มีตัวแปรอิสระ 5 ตัว ศึกษาเมื่อขนาดตัวอย่าง = 20, 30, 40 และ 50

   กรณีที่มีตัวแปรอิสระ 7 ตัว ศึกษาเมื่อขนาดตัวอย่าง = 30, 40 และ 50
3. ตัวแปรอิสระแต่ละตัวมีการแจกแจงปกติโดยมีค่าเฉลี่ยเท่ากับ 3, 4, 5, 6, 7, 8 และ 9 ตามลำดับ และส่วนเบี่ยงเบนมาตรฐานเท่ากับ 1.2, 2.4, 4.0, 6.0, 8.4, 11.2 และ 14.4 ตามลำดับ
4. ในงานวิจัยนี้ จะแบ่งระดับพหุสัมพันธ์ของตัวแปรอิสระเป็น 3 ระดับ โดยพิจารณาจากค่าสัมประสิทธิ์สหสัมพันธ์ (ρ) ดังนี้

   ระดับต่ำ ρ มีค่าตั้งแต่ 0.1 ถึง 0.3

   ระดับปานกลาง ρ มีค่าตั้งแต่ 0.4 ถึง 0.6

   ระดับสูง ρ มีค่าตั้งแต่ 0.7 ถึง 0.9

   และกำหนดระดับพหุสัมพันธ์ระหว่างตัวแปรอิสระในกรณีต่างๆ ดังนี้

   กรณีที่มีตัวแปรอิสระ 3 ตัว ผู้วิจัยจะศึกษาเมื่อตัวแปรอิสระ x_1 กับ x_2 มีความสัมพันธ์กัน โดยกำหนดสัมประสิทธิ์สหสัมพันธ์ ดังนี้

   ระดับต่ำ $\rho_{12} = 0.3$

   ระดับปานกลาง $\rho_{12} = 0.5$

   ระดับสูง $\rho_{12} = 0.8$

   (เมื่อ ρ_{ij} คือ สัมประสิทธิ์สหสัมพันธ์ระหว่าง x_i กับ x_j)

   กรณีที่มีตัวแปรอิสระ 5 ตัว ผู้วิจัยจะศึกษาเมื่อตัวแปรอิสระมีความสัมพันธ์กันเป็นคู่ๆ ได้แก่ x_1 กับ x_2 และ x_4 กับ x_5 โดยกำหนดสัมประสิทธิ์สหสัมพันธ์ ดังนี้

   ระดับต่ำ $\rho_{12} = 0.3$ และ $\rho_{45} = 0.3$

   ระดับปานกลาง $\rho_{12} = 0.4$ และ $\rho_{45} = 0.6$

   ระดับสูง $\rho_{12} = 0.7$ และ $\rho_{45} = 0.9$

กรณีที่มีตัวแปรอิสระ 7 ตัว ผู้วิจัยจะศึกษาเมื่อตัวแปรอิสระมีความสัมพันธ์กันเป็นคู่ๆ ได้แก่ x_1 กับ x_2 x_4 กับ x_5 และ x_6 กับ x_7 โดยกำหนดสัมประสิทธิ์สหสัมพันธ์ ดังนี้

ระดับต่ำ $\rho_{12} = 0.3,$ $\rho_{45} = 0.3$ และ $\rho_{67} = 0.3$

ระดับปานกลาง $\rho_{12} = 0.4,$ $\rho_{45} = 0.5$ และ $\rho_{67} = 0.6$

ระดับสูง $\rho_{12} = 0.7,$ $\rho_{45} = 0.8$ และ $\rho_{67} = 0.9$

5. กำหนดสัมประสิทธิ์ของการถดถอยพหุคูณ $(\boldsymbol{\beta})$ เป็นเวกเตอร์ของค่าคงที่ใดๆ ในที่นี้เลือกกำหนด $\beta_0 = 6, \beta_1 = 4, \beta_2 = 4, \beta_3 = 2, \beta_4 = 2, \beta_5 = 1, \beta_6 = 1$ และ $\beta_7 = 1$
6. ความคลาดเคลื่อนสุ่ม $(\boldsymbol{\varepsilon})$ มีการแจกแจงปกติที่มีค่าเฉลี่ยเท่ากับ 0 และส่วนเบี่ยงเบนมาตรฐานเท่ากับ 5
7. ตัวแบบการถดถอยที่จะนำมาใช้ในการหาค่าพยากรณ์ร่วมมีดังนี้

   7.1 ตัวแบบที่คัดเลือกจากวิธีพิจารณาสมการการถดถอยทุกรูปแบบ (all possible regressions)

   7.2 ตัวแบบที่ได้จากวิธีคัดเลือกตัวแปรแบบไปข้างหน้า (forward selection)

   7.3 ตัวแบบที่ได้จากวิธีกำจัดตัวแปรแบบถอยหลัง (backward elimination)

   7.4 ตัวแบบที่ได้จากวิธีการถดถอยขั้นบันได (stepwise regression)

1.6 เกณฑ์ที่ใช้ในการตัดสินใจ

เกณฑ์ที่ใช้ในการตัดสินใจว่าวิธีการหาค่าพยากรณ์ร่วมวิธีใดมีประสิทธิภาพมากที่สุด จะพิจารณาจากเกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (mean absolute percentage error: MAPE) ดังนี้

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100 \qquad (1.6)$$

โดยที่ y_i เป็นค่าสังเกตที่ i

$\hat{y}_i$ เป็นค่าพยากรณ์ที่ i

n เป็นขนาดตัวอย่าง

MAPE เป็นร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ของวิธีการที่พิจารณา

การวิจัยครั้งนี้จะทำการทดลองเป็นจำนวน 1000 รอบ แล้วหา**ค่าเฉลี่ยของ MAPE** ของแต่ละวิธี วิธีที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด จะถือเป็นวิธีที่ให้ค่าพยากรณ์ที่ถูกต้องและแม่นยำมากที่สุด หรือกล่าวได้ว่าเป็นวิธีที่มีประสิทธิภาพมากที่สุด

1.7 วิธีดำเนินการวิจัย

ในการศึกษาครั้งนี้ ผู้วิจัยมีขั้นตอนในการดำเนินการวิจัยดังนี้

1. สร้างข้อมูลตัวแปรอิสระโดยจำลองตามระดับความสัมพันธ์ต่างๆ ดังที่กำหนดไว้ในขอบเขตการวิจัย
2. สร้างค่าความคลาดเคลื่อนตามการแจกแจงและพารามิเตอร์ที่กำหนด
3. สร้างข้อมูลตัวแปรตาม จากค่าของตัวแปรอิสระและความคลาดเคลื่อน
4. สร้างตัวแบบด้วยวิธีต่อไปนี้
    - 4.1 วิธีพิจารณาสมการการถดถอยทุกรูปแบบ (โดยเลือกตัวแบบที่มีค่า MAPE ต่ำสุด)
    - 4.2 วิธีคัดเลือกตัวแปรแบบไปข้างหน้า
    - 4.3 วิธีกำจัดตัวแปรแบบถอยหลัง
    - 4.4 วิธีการถดถอยขั้นบันได
5. คำนวณหาน้ำหนักที่เหมาะสม เพื่อใช้ในการหาค่าพยากรณ์ร่วม โดยวิธีต่อไปนี้
    - 5.1 วิธีค่าสัมบูรณ์ต่ำสุด
    - 5.2 วิธีบูตสแตรป
    - 5.3 วิธี ARM
6. หาค่าพยากรณ์ร่วมของทุกตัวแบบในขั้นที่ 4 โดยใช้น้ำหนักที่ได้จากแต่ละวิธีในขั้นที่ 5
7. คำนวณค่า MAPE ของแต่ละวิธี
8. ทำซ้ำขั้นตอนที่ 2-7 เป็นจำนวน 1000 รอบ แล้วหาค่าเฉลี่ยของ MAPE ของแต่ละวิธี
9. สรุปและอภิปรายผล

1.8 ประโยชน์ที่คาดว่าจะได้รับ

ผู้วิจัยคาดหวังว่างานวิจัยนี้จะมีประโยชน์ดังนี้

1. เพื่อทราบประสิทธิภาพของวิธีหาค่าพยากรณ์ร่วมที่ศึกษาในแต่ละสถานการณ์ของการทดลอง
2. เพื่อเป็นแนวทางในการตัดสินใจเลือกวิธีหาค่าพยากรณ์ร่วมที่เหมาะสมสำหรับตัวแบบการถดถอยเชิงเส้นพหุคูณ
3. เพื่อเป็นแนวทางในการศึกษาเปรียบเทียบวิธีหาค่าพยากรณ์ร่วมวิธีอื่นๆ ต่อไป

บทที่ 2

ทฤษฎีและตัวสถิติที่เกี่ยวข้องกับการวิจัย

การศึกษาในครั้งนี้จะทำการเปรียบเทียบวิธีหาค่าพยากรณ์ร่วมของตัวแบบการถดถอยเชิงเส้นพหุคูณ เพื่อตอบคำถามว่าวิธีใดให้ค่าพยากรณ์ที่ถูกต้องและแม่นยำที่สุด โดยผู้วิจัยเสนอวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่

1. วิธีค่าสัมบูรณ์ต่ำสุด (least absolute errors: LAE)
2. วิธีบูตสแตรป (combination by bootstrap: BO)
3. วิธี adaptive regression by mixing (ARM)

โดยตัวแบบที่จะนำมาเข้าสู่กระบวนการหาค่าพยากรณ์ร่วม ได้แก่ ตัวแบบที่สร้างจากวิธีต่อไปนี้

1. ตัวแบบที่คัดเลือกจากวิธีพิจารณาสมการการถดถอยทุกรูปแบบ (all possible regressions)
2. ตัวแบบที่ได้จากวิธีคัดเลือกตัวแปรแบบไปข้างหน้า (forward selection)
3. ตัวแบบที่ได้จากวิธีกำจัดตัวแปรแบบถอยหลัง (backward elimination)
4. ตัวแบบที่ได้จากวิธีการถดถอยขั้นบันได (stepwise regression)

วิธีสร้างตัวแบบทั้ง 4 วิธีนี้จะต่างกันที่ขั้นตอนการเลือกตัวแปรอิสระเข้าหรือออกจากสมการการถดถอยเท่านั้น ส่วนการประมาณค่าสัมประสิทธิ์การถดถอยจะใช้วิธีเดียวกัน คือวิธีกำลังสองน้อยสุดแบบสามัญ (ordinary least squres: OLS)

การนำเสนอทฤษฎีและตัวสถิติที่เกี่ยวข้องจะเริ่มจาก แนวคิดเกี่ยวกับการถดถอยเชิงเส้นพหุคูณ จากนั้นจะกล่าวถึงการสร้างตัวแบบทั้ง 4 วิธี และท้ายสุดจะเสนอวิธีหาค่าพยากรณ์ร่วมทั้ง 3 วิธี

2.1 การถดถอยเชิงเส้นพหุคูณ (Multiple Linear Regression)

ตัวแบบที่ใช้ในการวิจัยครั้งนี้ คือ ตัวแบบการถดถอยเชิงเส้นพหุคูณซึ่งมีรูปแบบทั่วไปดังนี้

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} + \varepsilon_i, \quad i = 1, \dots, n \qquad (2.1)$$

เมื่อ y_i เป็นตัวแปรตาม

$x_{i1}, \dots, x_{ip}$ เป็นตัวแปรอิสระจำนวน p ตัว

$\beta_0, \beta_1, \beta_2, \dots, \beta_p$ เป็นสัมประสิทธิ์การถดถอย

ε_i เป็นความคลาดเคลื่อนสุ่ม

หรือเขียนในรูปของเมทริกซ์ได้ดังนี้

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{2.2}$$

โดยที่ $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$, $\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1p} \\ 1 & x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}$,

$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}$, $\boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$

ข้อสมมติของการวิเคราะห์การถดถอยเชิงเส้นพหุคูณมีดังนี้

1. $\boldsymbol{\varepsilon}$ มีค่าเฉลี่ยเป็นศูนย์ หรือเขียนได้ว่า $E(\boldsymbol{\varepsilon}) = \mathbf{0}$
2. $\boldsymbol{\varepsilon}$ มีความแปรปรวนคงที่ และ ε_i แต่ละตัวเป็นอิสระต่อกัน หรือเขียนได้ว่า $\mathrm{cov}(\boldsymbol{\varepsilon}) = \sigma^2 \mathbf{I}_n$

สำหรับการประมาณค่าเวกเตอร์พารามิเตอร์ $(\boldsymbol{\beta})$ ในการวิจัยครั้งนี้จะใช้วิธีกำลังสองน้อยที่สุดแบบสามัญ (ordinary least squares: OLS) ซึ่งมีหลักการคือ ทำให้ผลบวกกำลังสองของความคลาดเคลื่อนมีค่าต่ำสุด โดยตัวประมาณที่ได้จากวิธีนี้จะหาได้จากสูตร

$$\mathbf{b} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y} \tag{2.3}$$

เมื่อ $\mathbf{b}$ เป็นเวกเตอร์ตัวประมาณกำลังสองน้อยสุดของสัมประสิทธิ์การถดถอย

การหา $\mathbf{b}$ จากสมการ (2.3) มีเงื่อนไขว่า $\mathbf{X}^T\mathbf{X}$ ต้องเป็นเมทริกซ์ไม่เอกฐาน (nonsingular matrix) นั่นคือ สามารถหาเมทริกซ์ผกผันของ $\mathbf{X}^T\mathbf{X}$ ได้

2.2 วิธีสร้างสมการการถดถอย

วิธีสร้างสมการการถดถอยที่ใช้ในการวิจัยครั้งนี้ ได้แก่ วิธีพิจารณาสมการการถดถอยทุกรูปแบบ วิธีคัดเลือกตัวแปรแบบไปข้างหน้า วิธีกำจัดตัวแปรแบบถอยหลัง และวิธีการถดถอยขั้นบันได ซึ่งมีรายละเอียดดังต่อไปนี้

2.2.1 วิธีพิจารณาสมการการถดถอยทุกรูปแบบ (All Possible Regressions)

วิธีนี้จะทำการพิจารณาสมการการถดถอยทุกแบบที่เป็นไปได้ แล้วจึงใช้เกณฑ์ที่กำหนดในการเลือกตัวแบบที่ดีที่สุดออกมา

ให้ S แทนเซตของตัวแปรอิสระทั้งหมดที่คาดว่ามีความสัมพันธ์กับตัวแปรตาม กล่าวคือ $S = \{ x_1, x_2, ..., x_p \}$ การหาสมการการถดถอยทุกรูปแบบที่เป็นไปได้ จะพิจารณาจากเซตย่อยทั้งหมดของ S ดังนี้

กรณีที่ไม่มีตัวแปรอิสระอยู่ในสมการเลย จะได้สมการการถดถอย 1 สมการ คือ

$$\hat{y} = \bar{y}$$

กรณีที่มีตัวแปรอิสระ 1 ตัวอยู่ในสมการ จะได้สมการการถดถอย p สมการ ได้แก่

$$\begin{gathered} \hat{y} = f(x_1) \\ \hat{y} = f(x_2) \\ \vdots \\ \hat{y} = f(x_p) \end{gathered}$$

กรณีที่มีตัวแปรอิสระ 2 ตัวอยู่ในสมการ จะได้สมการการถดถอย $\binom{p}{2}$ สมการ ได้แก่

$$\begin{gathered} \hat{y} = f(x_1, x_2) \\ \hat{y} = f(x_1, x_3) \\ \vdots \\ \hat{y} = f(x_{p-1}, x_p) \end{gathered}$$

ทำเช่นนี้เรื่อยๆ ไปจนถึงกรณีสุดท้าย คือ
กรณีที่มีตัวแปรอิสระทั้ง p ตัว อยู่ในสมการ จะได้สมการการถดถอย 1 สมการ คือ

$$\hat{y} = f(x_1, x_2, ..., x_p)$$

โดยสรุปแล้ว สมการการถดถอยทั้งหมดที่เป็นไปได้ จะมี 2^p สมการ ซึ่งสมการเหล่านี้ จะถูกนำมาประเมินผลโดยใช้เกณฑ์ที่กำหนด เช่น R^2, MSE, Mallows' C_p เป็นต้น สำหรับการ

วิจัยครั้งนี้ ผู้วิจัยเลือกใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) ในการเลือกตัวแบบจากตัวแบบทั้งหมดที่เป็นไปได้ โดยจะเลือกตัวแบบที่มีค่า MAPE ต่ำสุด ตามสูตรดังนี้

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100$$

โดยที่ y_i เป็นค่าสังเกตที่ i

$\hat{y}_i$ เป็นค่าพยากรณ์ที่ i

n เป็นขนาดตัวอย่าง

2.2.2 วิธีคัดเลือกตัวแปรแบบไปข้างหน้า (Forward Selection)

วิธีการนี้จะเริ่มจากการนำตัวแปรอิสระที่มีความสัมพันธ์กับตัวแปรตามมากที่สุดเข้าสมการก่อน จากนั้นจะทำการเพิ่มตัวแปรอิสระเข้าไปในสมการครั้งละ 1 ตัว ตราบใดที่การเพิ่มตัวแปรอิสระเหล่านั้นมีนัยสำคัญ ซึ่งการพิจารณาว่าจะนำตัวแปรใดเข้าสมการจะใช้การทดสอบเอฟบางส่วน (partial F-test)

การคัดเลือกตัวแปรแบบไปข้างหน้า มีขั้นตอนดังนี้

1. เลือกตัวแปรอิสระที่มีสหสัมพันธ์กับตัวแปรตามสูงสุด แล้วสร้างสมการการถดถอยโดยมีตัวแปรดังกล่าวอยู่ในสมการเพียงตัวเดียว (พร้อมทั้งคำนวณค่าสัมประสิทธิ์การถดถอย)

2. ทดสอบว่าตัวแปรอิสระดังกล่าวมีนัยสำคัญหรือไม่ หากตัวแปรอิสระดังกล่าวไม่มีนัยสำคัญ จะจบขั้นตอน และได้สมการการถดถอยที่เหมาะสมคือ $\hat{y} = \bar{y}$ แต่ถ้าตัวแปรอิสระดังกล่าวมีนัยสำคัญ จะทำขั้นตอนต่อไป

3. คำนวณค่าสถิติทดสอบเอฟบางส่วน (partial F-test value) สำหรับตัวแปรอิสระทุกตัวที่ไม่อยู่ในสมการ

4. นำค่าสถิติทดสอบเอฟบางส่วนที่มากที่สุด (F_U) มาเปรียบเทียบกับค่าเอฟ ณ ระดับนัยสำคัญที่กำหนด (F_{in})

- ถ้า $F_U < F_{in}$ จะไม่นำตัวแปรอิสระที่มีค่าสถิติทดสอบเอฟบางส่วนมากที่สุดนั้นเข้าสู่สมการ และจบขั้นตอน ได้สมการการถดถอยที่คำนวณได้ครั้งล่าสุดเป็นสมการที่เหมาะสม
- ถ้า $F_U > F_{in}$ จะนำตัวแปรอิสระที่มีค่าสถิติทดสอบเอฟบางส่วนมากที่สุดนั้นเข้าสู่สมการ พร้อมทั้งคำนวณค่าสัมประสิทธิ์การถดถอย

5. ถ้าไม่มีตัวแปรใดจะนำเข้าสมการอีกแล้ว จะจบขั้นตอนและได้สมการการถดถอยที่คำนวณได้ครั้งล่าสุดเป็นสมการที่เหมาะสม แต่ถ้ายังมีตัวแปรอิสระเหลืออยู่ ให้กลับไปทำขั้นตอนที่ 3

2.2.3 วิธีกำจัดตัวแปรแบบถอยหลัง (Backward Elimination)

วิธีกำจัดตัวแปรแบบถอยหลัง มีขั้นตอนดังนี้

1. สร้างสมการการถดถอยที่มีตัวแปรอิสระทุกตัวอยู่ในสมการ พร้อมทั้งคำนวณค่าสัมประสิทธิ์การถดถอย
2. คำนวณค่าสถิติทดสอบเอฟบางส่วนของตัวแปรอิสระทุกตัวที่อยู่ในสมการการถดถอย
3. นำค่าสถิติทดสอบเอฟบางส่วนที่น้อยที่สุด (F_L) มาเปรียบเทียบกับค่าเอฟ ณ ระดับนัยสำคัญที่กำหนด (F_{out})
   - ถ้า $F_L < F_{out}$ จะนำตัวแปรอิสระที่มีค่าสถิติทดสอบเอฟบางส่วนที่น้อยที่สุดนั้นออกจากสมการการถดถอย แล้วคำนวณค่าสัมประสิทธิ์การถดถอยเมื่อตัดตัวแปรอิสระนั้นแล้ว จากนั้นกลับไปขั้นตอนที่ 2
   - ถ้า $F_L > F_{out}$ จะไม่นำตัวแปรอิสระที่มีค่าสถิติทดสอบเอฟบางส่วนน้อยที่สุดนั้นออกจากสมการ และจบขั้นตอน ได้สมการการถดถอยที่คำนวณได้ครั้งล่าสุดเป็นสมการที่เหมาะสม

2.2.4 วิธีการถดถอยขั้นบันได (Stepwise Regression)

วิธีการนี้เป็นวิธีการคัดเลือกตัวแปรอิสระแบบเลือกเข้า โดยในแต่ละขั้นตอนของการเลือกตัวแปรอิสระเข้าสู่สมการจะทำการตรวจสอบตัวแปรอิสระที่มีในสมการเสียก่อน (โดยวิธีกำจัดตัวแปรแบบถอยหลัง) ซึ่งสามารถแสดงขั้นตอนของการถดถอยขั้นบันไดได้ดังนี้

1. เลือกตัวแปรอิสระที่มีสหสัมพันธ์กับตัวแปรตามสูงสุด แล้วสร้างสมการการถดถอยโดยมีตัวแปรดังกล่าวอยู่ในสมการเพียงตัวเดียว (พร้อมทั้งคำนวณค่าสัมประสิทธิ์การถดถอย)
2. ทดสอบว่าตัวแปรอิสระดังกล่าวมีนัยสำคัญหรือไม่ หากตัวแปรอิสระดังกล่าวไม่มีนัยสำคัญ จะจบขั้นตอน และได้สมการการถดถอยที่เหมาะสมคือ $\hat{y} = \bar{y}$ แต่ถ้าตัวแปรอิสระดังกล่าวมีนัยสำคัญ จะทำขั้นตอนต่อไป

3. คำนวณค่าสถิติทดสอบเอฟบางส่วน (partial F-test value) สำหรับตัวแปรอิสระทุกตัวที่ไม่อยู่ในสมการ
4. นำค่าสถิติทดสอบเอฟบางส่วนที่มากที่สุด (F_U) มาเปรียบเทียบกับค่าเอฟ ณ ระดับนัยสำคัญที่กำหนด (F_{in})
   - ถ้า $F_U < F_{in}$ จะไม่นำตัวแปรอิสระที่มีค่าสถิติทดสอบเอฟบางส่วนมากที่สุดนั้นเข้าสู่สมการ
   - ถ้า $F_U > F_{in}$ จะนำตัวแปรอิสระที่มีค่าสถิติทดสอบเอฟบางส่วนมากที่สุดนั้นเข้าสู่สมการ พร้อมทั้งคำนวณค่าสัมประสิทธิ์การถดถอย
5. คำนวณค่าสถิติทดสอบเอฟบางส่วนของตัวแปรอิสระทุกตัวที่อยู่ในสมการการถดถอย
6. นำค่าสถิติทดสอบเอฟบางส่วนที่น้อยที่สุด (F_L) มาเปรียบเทียบกับค่าเอฟ ณ ระดับนัยสำคัญที่กำหนด (F_{out})
   - ถ้า $F_L < F_{out}$ จะนำตัวแปรอิสระที่มีค่าสถิติทดสอบเอฟบางส่วนที่น้อยที่สุดนั้นออกจากสมการการถดถอย แล้วคำนวณค่าสัมประสิทธิ์การถดถอยเมื่อตัดตัวแปรอิสระนั้นแล้ว
   - ถ้า $F_L > F_{out}$ จะไม่นำตัวแปรอิสระที่มีค่าสถิติทดสอบเอฟบางส่วนที่น้อยที่สุดนั้นออกจากสมการ
7. ทำขั้นตอนที่ 3-6 ซ้ำ จนกระทั่งไม่มีตัวแปรอิสระใดสามารถเข้าหรือออกจากสมการได้อีกแล้ว จึงสิ้นสุดขั้นตอน และได้สมการที่เหมาะสมคือสมการที่คำนวณได้ครั้งล่าสุด

2.3 การหาค่าพยากรณ์ร่วม

สมมติว่าตัวแบบการถดถอยเชิงเส้นพหุคูณในกลุ่มที่พิจารณามีอยู่ทั้งหมด m ตัวแบบ โดยมีตัวแบบที่ k คือ

$$\hat{\mathbf{y}}^{(k)} = \mathbf{X}^{(k)}\mathbf{b}^{(k)}; \quad k = 1, \ldots, m \tag{2.4}$$

เมื่อ $\hat{\mathbf{y}}^{(k)}$ เป็นเวกเตอร์ค่าพยากรณ์ของตัวแปรตามในตัวแบบที่ k

$\mathbf{X}^{(k)}$ เป็นเมทริกซ์ของตัวแปรอิสระในตัวแบบที่ k

$\mathbf{b}^{(k)}$ เป็นเวกเตอร์ตัวประมาณของสัมประสิทธิ์การถดถอยในตัวแบบที่ k ซึ่งประมาณได้จากข้อมูลตัวอย่าง

ในที่นี้จะพิจารณาค่าพยากรณ์ร่วม ซึ่งเกิดจากการรวมเชิงเส้นของค่าพยากรณ์จากแต่ละตัวแบบ ดังนี้

$$\hat{y}_i^{combined} = \sum_{k=1}^{m} w^{(k)} \hat{y}_i^{(k)} \quad , \quad i = 1, \ldots, n \tag{2.5}$$

เมื่อ $\hat{y}_i^{combined}$ คือค่าพยากรณ์ร่วมของตัวแปรตาม

$w^{(k)}$ คือค่าน้ำหนักที่ให้กับตัวแบบที่ k, $k = 1, \ldots, m$

$\hat{y}_i^{(k)}$ คือค่าพยากรณ์ของตัวแปรตามจากตัวแบบที่ k, $k = 1, \ldots, m$

ในงานวิจัยนี้ จะศึกษาวิธีการหาค่าน้ำหนัก $\left(w^{(k)}\right)$ 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด วิธีบูตสแตรป และวิธี adaptive regression by mixing

2.3.1 การหาค่าพยากรณ์ร่วมโดยวิธีค่าสัมบูรณ์ต่ำสุด (Combination by Least Absolute Errors)

วิธีนี้มีหลักการคือ ทำให้ผลรวมของค่าสัมบูรณ์ของความคลาดเคลื่อนมีค่าต่ำสุด โดยมีรายละเอียดดังนี้

ให้ e_i คือความคลาดเคลื่อนที่เกิดจากการพยากรณ์ y_i ด้วย $\hat{y}_i^{combined}$

จะได้ว่า
$$\begin{aligned} e_i &= y_i - \hat{y}_i^{combined} \\ &= y_i - \sum_{k=1}^{m} w^{(k)} \hat{y}_i^{(k)} , \quad i = 1, \ldots, n \end{aligned}$$

สามารถเขียนปัญหาโปรแกรมเชิงเส้นได้ดังนี้

หาค่าต่ำสุดของ $Z = \sum_{i=1}^{n} |e_i|$

ภายใต้ข้อจำกัดคือ $\sum_{k=1}^{m} w^{(k)} = 1$

$$w^{(k)} \geq 0 , \quad k = 1, \ldots, m$$

เนื่องจาก e_i ไม่มีข้อจำกัดด้านเครื่องหมาย จึงกำหนดให้

$$e_i = e_i^+ - e_i^- \text{ เมื่อ } e_i^+ \geq 0, e_i^- \geq 0 \text{ และ } e_i^+ \times e_i^- = 0$$

นั่นคือ อย่างน้อย 1 ตัวแปรใน e_i^+ และ e_i^- จะเท่ากับศูนย์เสมอ

ดังนั้น $|e_i| = e_i^+ + e_i^-$

เพื่อความสะดวก ให้ $u_i = e_i^+$ และ $v_i = e_i^-$

ดังนั้น เราสามารถแปลงปัญหาข้างต้น ได้เป็น

หาค่าต่ำสุดของ $$Z=\sum_{i=1}^{n}(u_i+v_i)$$

ภายใต้ข้อจำกัดคือ $$u_i-v_i+\sum_{k=1}^{m}w^{(k)}\hat{y}_i^{(k)}=y_i,\quad i=1,\dots,n$$

$$\sum_{k=1}^{m}w^{(k)}=1$$

$$u_i\ge 0\quad ,i=1,\dots,n$$

$$v_i\ge 0\quad ,i=1,\dots,n$$

$$w^{(k)}\ge 0\quad ,k=1,\dots,m$$

เขียนแบบแจกแจงรายละเอียดได้ดังนี้

หาค่าต่ำสุดของ $$Z=(u_1+v_1)+(u_2+v_2)+\dots+(u_n+v_n)$$

ภายใต้ข้อจำกัดคือ

$$\begin{array}{llllll}
u_1-v_1 & & & +\hat{y}_1^{(1)}w^{(1)}+\hat{y}_1^{(2)}w^{(2)}+\dots+\hat{y}_1^{(m)}w^{(m)} & = y_1 \\
 & u_2-v_2 & & +\hat{y}_2^{(1)}w^{(1)}+\hat{y}_2^{(2)}w^{(2)}+\dots+\hat{y}_2^{(m)}w^{(m)} & = y_2 \\
 & & \vdots & & \vdots \\
 & & & u_n-v_n+\hat{y}_n^{(1)}w^{(1)}+\hat{y}_n^{(2)}w^{(2)}+\dots+\hat{y}_n^{(m)}w^{(m)} & = y_n
\end{array}$$

$$w^{(1)}+w^{(2)}+\dots+w^{(m)}=1$$

$$u_i\ge 0\quad ,i=1,\dots,n$$

$$v_i\ge 0\quad ,i=1,\dots,n$$

$$w^{(k)}\ge 0\quad ,k=1,\dots,m$$

หรือเขียนในรูปเมทริกซ์ได้ดังนี้

หาค่าต่ำสุดของ

$$Z = \begin{bmatrix} \underbrace{1 \quad 1 \quad 1 \quad 1 \ \ldots \ 1 \quad 1}_{2n \text{ ตัว}} & \underbrace{0 \quad 0 \ \ldots \ 0}_{m \text{ ตัว}} \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ \vdots \\ u_n \\ v_n \\ w^{(1)} \\ w^{(2)} \\ \vdots \\ w^{(m)} \end{bmatrix}$$

ภายใต้ข้อจำกัดคือ

$$\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & \hat{y}_1^{(1)} & \hat{y}_1^{(2)} & \ldots & \hat{y}_1^{(m)} \\ 0 & 0 & 1 & -1 & 0 & 0 & \ldots & 0 & 0 & \hat{y}_2^{(1)} & \hat{y}_2^{(2)} & \ldots & \hat{y}_2^{(m)} \\ 0 & 0 & 0 & 0 & 1 & -1 & \ldots & 0 & 0 & \hat{y}_3^{(1)} & \hat{y}_3^{(2)} & \ldots & \hat{y}_3^{(m)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & \ldots & 1 & -1 & \hat{y}_n^{(1)} & \hat{y}_n^{(2)} & \ldots & \hat{y}_n^{(m)} \\ 0 & 0 & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 1 & 1 & \ldots & 1 \end{bmatrix}_{\substack{\underbrace{\hphantom{xxxxxxxxxxxx}}_{2n \text{ คอลัมน์}} \ \underbrace{\hphantom{xxxxxx}}_{m \text{ คอลัมน์}}}} \begin{bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ \vdots \\ u_n \\ v_n \\ w^{(1)} \\ w^{(2)} \\ \vdots \\ w^{(m)} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \\ 1 \end{bmatrix}$$

$$u_i \geq 0 \quad , i = 1, \ldots, n$$
$$v_i \geq 0 \quad , i = 1, \ldots, n$$
$$w^{(k)} \geq 0 \quad , k = 1, \ldots, m$$

ปัญหาโปรแกรมเชิงเส้นที่อยู่ในรูปแบบข้างต้น สามารถแก้ได้โดยใช้วิธีซิมเพลกซ์ (simplex method)

2.3.2 การหาค่าพยากรณ์ร่วมโดยวิธีบูตสแตรป (Combination by Bootstrap)

ให้ $\mathbf{Z}$ แทนเซตของข้อมูลตัวอย่าง กล่าวคือ

$$\mathbf{Z} = \{(\mathbf{x}_i, y_i) \mid i = 1, \ldots, n\} \qquad \text{เมื่อ } \mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$$

สมมติว่านำข้อมูล $\mathbf{Z}$ มาสร้างตัวแบบการถดถอยได้ทั้งหมด m ตัวแบบ โดยมีตัวแบบที่ k คือ

$$\hat{y}_i^{(k)} = f_{\mathbf{Z}}^{(k)}\left(\mathbf{x}_i; \mathbf{b}_{\mathbf{Z}}^{(k)}\right), \quad k = 1, \ldots, m, \quad i = 1, \ldots, n \tag{2.6}$$

เมื่อ $\hat{y}_i^{(k)}$ คือค่าพยากรณ์ของตัวแปรตามของตัวแบบที่ k

$f_{\mathbf{Z}}^{(k)}$ คือฟังก์ชันการถดถอยของตัวแบบที่ k ที่ประมาณจากข้อมูล $\mathbf{Z}$

$\mathbf{b}_{\mathbf{Z}}^{(k)}$ คือเวกเตอร์ตัวประมาณของสัมประสิทธิ์การถดถอยในตัวแบบที่ k ที่ประมาณจากข้อมูล $\mathbf{Z}$

ให้ $\mathbf{f}_{\mathbf{Z}}(\mathbf{x}_i) = \begin{bmatrix} f_{\mathbf{Z}}^{(1)}(\mathbf{x}_i; \mathbf{b}_{\mathbf{Z}}^{(1)}) \\ \vdots \\ f_{\mathbf{Z}}^{(m)}(\mathbf{x}_i; \mathbf{b}_{\mathbf{Z}}^{(m)}) \end{bmatrix}$ คือเวกเตอร์ของฟังก์ชันการถดถอยของ m ตัวแบบ ที่ประมาณจากข้อมูล $\mathbf{Z}$

ต้องการหาน้ำหนัก $\mathbf{w} = \begin{bmatrix} w^{(1)} \\ \vdots \\ w^{(k)} \end{bmatrix}$ ที่ทำให้ $\mathbf{f}_{\mathbf{Z}}^T(\mathbf{x}_i)\mathbf{w} = \sum_{k=1}^{m} w^{(k)} f_{\mathbf{Z}}^{(k)}\left(\mathbf{x}_i; \mathbf{b}_{\mathbf{Z}}^{(k)}\right)$ มีความผิดพลาดในการพยากรณ์ต่ำสุด

การหาค่า $\mathbf{w}$ ในที่นี้จะใช้เกณฑ์กำลังสองน้อยสุด กล่าวคือ

$$\text{หาค่าต่ำสุดของ} \qquad G = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \mathbf{f}_{\mathbf{Z}}^T(\mathbf{x}_i)\mathbf{w}\right)^2 \tag{2.7}$$

ซึ่งเวกเตอร์ $\mathbf{w}$ ที่จะทำให้ G มีค่าต่ำสุด สามารถหาได้จากสูตร

$$\mathbf{w}_{\mathrm{LS}} = \left[\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}}(\mathbf{x}_i)\mathbf{f}_{\mathbf{Z}}^T(\mathbf{x}_i)\right]^{-1}\left[\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}}(\mathbf{x}_i)y_i\right] \tag{2.8}$$

เมื่อ $\mathbf{w}_{\mathrm{LS}}$ คือเวกเตอร์ของค่าน้ำหนักที่ได้จากวิธีกำลังสองน้อยสุด

แต่ $\mathbf{w}_{\mathrm{LS}}$ ที่ได้จากสูตร (2.8) นี้จะเป็นตัวประมาณที่ไม่ดี ทั้งนี้เพราะข้อมูล $\mathbf{Z}=\{(\mathbf{x}_1, y_1),\ldots,(\mathbf{x}_n, y_n)\}$ ถูกนำมาใช้ในการประมาณค่าถึง 2 ครั้งด้วยกัน ซึ่งได้แก่

1. การสร้างตัวแบบใน (2.6)
2. การคำนวณความคลาดเคลื่อนใน (2.7)

ดังนั้นการประมาณค่าน้ำหนักด้วยวิธีกำลังสองน้อยสุดนี้ จะทำให้ค่า $\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}}(\mathbf{x}_i)\mathbf{f}_{\mathbf{Z}}^{T}(\mathbf{x}_i)$ และ $\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}}(\mathbf{x}_i)y_i$ ใน (2.8) มีความเอนเอียง

ด้วยเหตุนี้ LeBlanc และ Tibshirani จึงเสนอให้ใช้วิธีบูตสแตรป เข้ามาช่วยในการแก้ไขความเอนเอียง (bias-correction) ที่เกิดขึ้น ซึ่งค่าน้ำหนักที่ได้หลังจากแก้ไขความเอนเอียงแล้วจะเขียนได้ดังนี้

$$\mathbf{w}_{\mathrm{BO}}=\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}}(\mathbf{x}_i)\mathbf{f}_{\mathbf{Z}}^{T}(\mathbf{x}_i)+\mathbf{\Delta}_1\right]^{-1}\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}}(\mathbf{x}_i)y_i+\mathbf{\Delta}_2\right] \tag{2.9}$$

เมื่อ $\mathbf{w}_{\mathrm{BO}}$ คือเวกเตอร์ของน้ำหนักที่ได้จากวิธีบูตสแตรป

$\mathbf{\Delta}_1$ และ $\mathbf{\Delta}_2$ คือเทอมที่ใช้แก้ไขความเอนเอียง

ขั้นตอนวิธีของการหาน้ำหนักโดยวิธีบูตสแตรปมีดังนี้

1. สุ่มตัวอย่างโดยวิธีบูตสแตรปจากข้อมูล $\mathbf{Z}$ ออกมา N ชุด โดยให้ชุดที่ j แทนด้วย

$$\mathbf{Z}^{*j}=\left\{(\mathbf{x}_i^{*j}, y_i^{*j})\,\middle|\, i=1,\ldots,n\right\},\ j=1,\ldots,N$$

2. นำตัวอย่างแต่ละชุดไปประมาณค่าสัมประสิทธิ์การถดถอยของแต่ละตัวแบบใหม่ ซึ่งจะเขียนแทนด้วยสัญลักษณ์

$$\mathbf{b}_{\mathbf{Z}^{*j}}^{(1)},\mathbf{b}_{\mathbf{Z}^{*j}}^{(2)},\ldots,\mathbf{b}_{\mathbf{Z}^{*j}}^{(m)}$$

ซึ่งหมายถึงเวกเตอร์สัมประสิทธิ์การถดถอยของตัวแบบที่ 1,2, …, m ซึ่งประมาณขึ้นใหม่จากตัวอย่างบูตสแตรปชุดที่ j $(j=1,\ldots,N)$

และจะได้เวกเตอร์ฟังก์ชันการถดถอยของ m ตัวแบบใหม่ คือ

$$\mathbf{f}_{\mathbf{Z}^{*j}}=\begin{bmatrix} f_{\mathbf{Z}^{*j}}^{(1)} \\ f_{\mathbf{Z}^{*j}}^{(2)} \\ \vdots \\ f_{\mathbf{Z}^{*j}}^{(m)} \end{bmatrix},\ j=1,\ldots,N$$

3. คำนวณหาค่าของฟังก์ชันการถดถอยที่ได้ใหม่ ณ $\mathbf{x}_i^{*j}$ และ $\mathbf{x}_i$ ตามลำดับ ดังนี้

$$\mathbf{f}_{\mathbf{Z}^{*j}}\left(\mathbf{x}_i^{*j}\right)=\begin{bmatrix} f_{\mathbf{Z}^{*j}}^{(1)}\left(\mathbf{x}_i^{*j};\mathbf{b}_{\mathbf{Z}^{*j}}^{(1)}\right) \\ f_{\mathbf{Z}^{*j}}^{(2)}\left(\mathbf{x}_i^{*j};\mathbf{b}_{\mathbf{Z}^{*j}}^{(2)}\right) \\ \vdots \\ f_{\mathbf{Z}^{*j}}^{(m)}\left(\mathbf{x}_i^{*j};\mathbf{b}_{\mathbf{Z}^{*j}}^{(m)}\right) \end{bmatrix},\ j=1,\ldots,N;\ i=1,\ldots,n$$

และ

$$\mathbf{f}_{\mathbf{Z}^{*j}}\left(\mathbf{x}_i\right)=\begin{bmatrix} f_{\mathbf{Z}^{*j}}^{(1)}\left(\mathbf{x}_i;\mathbf{b}_{\mathbf{Z}^{*j}}^{(1)}\right) \\ f_{\mathbf{Z}^{*j}}^{(2)}\left(\mathbf{x}_i;\mathbf{b}_{\mathbf{Z}^{*j}}^{(2)}\right) \\ \vdots \\ f_{\mathbf{Z}^{*j}}^{(m)}\left(\mathbf{x}_i;\mathbf{b}_{\mathbf{Z}^{*j}}^{(m)}\right) \end{bmatrix},\ j=1,\ldots,N;\ i=1,\ldots,n$$

4. คำนวณหา $\mathbf{\Delta}_1$ และ $\mathbf{\Delta}_2$ จากสูตรต่อไปนี้

$$\mathbf{\Delta}_1=\frac{1}{N}\sum_{j=1}^{N}\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}^{*j}}\left(\mathbf{x}_i\right)\mathbf{f}_{\mathbf{Z}^{*j}}^{T}\left(\mathbf{x}_i\right)-\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}^{*j}}\left(\mathbf{x}_i^{*j}\right)\mathbf{f}_{\mathbf{Z}^{*j}}^{T}\left(\mathbf{x}_i^{*j}\right)\right]$$

$$\mathbf{\Delta}_2=\frac{1}{N}\sum_{j=1}^{N}\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}^{*j}}\left(\mathbf{x}_i\right)y_i-\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}^{*j}}\left(\mathbf{x}_i^{*j}\right)y_i^{*j}\right]$$

5. นำค่า $\mathbf{\Delta}_1$ และ $\mathbf{\Delta}_2$ แทนในสูตร (2.9) จะได้น้ำหนักจากวิธีบูตสแตรปตามต้องการ

$$\mathbf{w}_{\mathrm{BO}}=\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}}\left(\mathbf{x}_i\right)\mathbf{f}_{\mathbf{Z}}^{T}\left(\mathbf{x}_i\right)+\mathbf{\Delta}_1\right]^{-1}\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}}\left(\mathbf{x}_i\right)y_i+\mathbf{\Delta}_2\right]$$

2.3.3 การหาค่าพยากรณ์ร่วมโดยวิธี Adaptive Regression by Mixing (ARM)

ในปี 2001 Yang ได้เสนอวิธี ARM สำหรับหาค่าพยากรณ์ร่วมในกรณีที่ความคลาดเคลื่อนมีการแจกแจงปกติ โดยวิธีดังกล่าวมี 2 ขั้นตอนใหญ่ ๆ ดังนี้

ขั้นตอนที่ 1 ใช้ข้อมูลตัวอย่างจำนวนครึ่งหนึ่งมาประมาณค่าสัมประสิทธิ์การถดถอยของตัวแบบทั้ง m ตัวแบบใหม่ ซึ่งในขั้นตอนนี้จะได้ตัวแบบเพิ่มมาอีก m ตัวแบบ โดยแต่ละตัวแบบจะมีจำนวนตัวแปรอิสระเท่ากับตัวแบบเดิม แต่ค่าสัมประสิทธิ์การถดถอยจะเปลี่ยนแปลงไป

ขั้นตอนที่ 2 นำค่าข้อมูลตัวแปรอิสระจากข้อมูลตัวอย่างส่วนที่เหลือมาแทนค่าในตัวแบบใหม่ที่ได้จากขั้นตอนที่ 1 เพื่อพยากรณ์ค่าของตัวแปรตาม แล้วประเมินความแม่นยำของการพยากรณ์ที่ได้ (โดยการเปรียบเทียบค่าพยากรณ์กับค่าจริง) จากนั้นจะนำผลการประเมินที่ได้มาใช้คำนวณหาค่าน้ำหนัก

กำหนดข้อมูลตัวอย่าง คือ $(\mathbf{x}_i, y_i), i = 1, 2, ..., n$ โดยที่ $\mathbf{x}_i = (x_{i1}, ..., x_{ip})$
ให้ $\mathbf{Z}$ เป็นเซตของข้อมูลตัวอย่างทั้งหมด กล่าวคือ $\mathbf{Z} = \{(\mathbf{x}_i, y_i) \mid i = 1, 2, ..., n\}$
สมมติว่านำข้อมูล $\mathbf{Z}$ ไปสร้างตัวแบบการถดถอยได้ทั้งหมด m ตัวแบบ ดังนี้

$$\hat{y}_i^{(k)} = f_{\mathbf{Z}}^{(k)}(\mathbf{x}_i; \mathbf{b}_{\mathbf{Z}}^{(k)}) \quad , k = 1, \dots, m$$

เมื่อ $\hat{y}_i^{(k)}$ คือค่าพยากรณ์ของตัวแปรตามของตัวแบบที่ k
$f_{\mathbf{Z}}^{(k)}$ คือฟังก์ชันการถดถอยของตัวแบบที่ k ที่ประมาณจากข้อมูล $\mathbf{Z}$
$\mathbf{b}_{\mathbf{Z}}^{(k)}$ คือเวกเตอร์ตัวประมาณของสัมประสิทธิ์การถดถอยในตัวแบบที่ k ที่ประมาณจากข้อมูล $\mathbf{Z}$
และให้ $\left(\hat{\sigma}_{\mathbf{Z}}^{(k)}\right)^2$ คือตัวประมาณของความแปรปรวนในตัวแบบที่ k ที่ประมาณจากข้อมูล $\mathbf{Z}$

ขั้นตอนวิธีโดยละเอียดของ ARM สามารถแสดงได้ดังนี้ (เพื่อความสะดวกในที่นี้จะกำหนดให้ขนาดตัวอย่าง (n) เป็นจำนวนคู่)

1. แบ่งข้อมูลตัวอย่างออกเป็น 2 กลุ่ม กลุ่มละเท่า ๆ กัน โดยให้กลุ่มที่หนึ่งแทนด้วย

$$\mathbf{Z}_1 = \left\{ (\mathbf{x}_i, y_i) \;\middle|\; 1 \le i \le \frac{n}{2} \right\}$$

และกลุ่มที่สองแทนด้วย

$$\mathbf{Z}_2 = \left\{ (\mathbf{x}_i, y_i) \;\middle|\; \frac{n}{2} + 1 \le i \le n \right\}$$

2. นำข้อมูล $\mathbf{Z}_1$ มาประมาณค่าสัมประสิทธิ์การถดถอยของตัวแบบที่ k ใหม่ โดยใช้วิธีกำลังสองน้อยสุด ในขั้นนี้จะได้ตัวแบบใหม่ดังนี้

$$\hat{y}_i^{(k)} = f_{\mathbf{Z}_1}^{(k)}(\mathbf{x}_i; \mathbf{b}_{\mathbf{Z}_1}^{(k)}) \quad , k = 1, \ldots, m$$

เมื่อ $f_{\mathbf{Z}_1}^{(k)}$ คือฟังก์ชันการถดถอยของตัวแบบที่ k ที่ประมาณจากข้อมูล $\mathbf{Z}_1$

$\mathbf{b}_{\mathbf{Z}_1}^{(k)}$ คือเวกเตอร์ตัวประมาณของสัมประสิทธิ์การถดถอยในตัวแบบที่ k ที่ประมาณจากข้อมูล $\mathbf{Z}_1$

พร้อมทั้งคำนวณค่าประมาณของ σ^2 ของแต่ละตัวแบบ ซึ่งจะเขียนแทนด้วย $\left(\hat{\sigma}_{\mathbf{Z}_1}^{(k)}\right)^2$

3. ประเมินความแม่นยำของตัวแบบที่ได้ในขั้นที่ 2 โดยใช้ข้อมูลตัวอย่างที่เหลือคือ $\mathbf{Z}_2$ กล่าวคือ

แทนค่า $\mathbf{x}_i$ จากข้อมูล $\mathbf{Z}_2$ $\left(\frac{n}{2} + 1 \le i \le n\right)$ ลงในตัวแบบทั้ง m ตัวแบบในขั้นที่ 2 เพื่อพยากรณ์ค่า y_i จากนั้นคำนวณค่า overall measure of discrepancy ตามสูตรดังนี้

$$D^{(k)} = \sum_{i=n/2+1}^{n} \left(y_i - f_{\mathbf{Z}_1}^{(k)}(\mathbf{x}_i; \mathbf{b}_{\mathbf{Z}_1}^{(k)})\right)^2$$

4. คำนวณน้ำหนักสำหรับตัวแบบที่ k จากสูตร

$$w^{(k)} = \frac{\left(\hat{\sigma}_{\mathbf{Z}_1}^{(k)}\right)^{-n/2} \exp\left(\dfrac{-(\hat{\sigma}_{\mathbf{Z}_1}^{(k)})^{-2} D^{(k)}}{2}\right)}{\sum_{q=1}^{m} \left(\hat{\sigma}_{\mathbf{Z}_1}^{(q)}\right)^{-n/2} \exp\left(\dfrac{-(\hat{\sigma}_{\mathbf{Z}_1}^{(q)})^{-2} D^{(q)}}{2}\right)}$$

โดยที่ $\sum_{k=1}^{m} w^{(k)} = 1$

จะเห็นได้ว่า $w^{(k)}$ คำนวณที่ได้ในขั้นตอนที่ 4 จะขึ้นอยู่กับอันดับของข้อมูลตัวอย่างในตอนแรก ทั้งนี้เพราะมีการแบ่งครึ่งข้อมูลในขั้นตอนที่ 1 แต่เนื่องจากเรามีข้อสมมติว่าข้อมูล $(\mathbf{x}_i, y_i), i = 1, 2, ..., n$ แต่ละตัวเป็นอิสระกัน ดังนั้นอันดับของข้อมูลจึงไม่ควรมีผลต่อการประมาณค่าน้ำหนัก $w^{(k)}$

เราสามารถปรับปรุงค่าน้ำหนัก $w^{(k)}$ ได้ โดยการสับเปลี่ยนอันดับของข้อมูลใหม่ (permutation) แล้วคำนวณค่า $w^{(k)}$ จากทุก ๆ แบบของการเรียงสับเปลี่ยนของอันดับข้อมูลที่เป็นไปได้ จากนั้นจึงหาค่าเฉลี่ยของน้ำหนักที่ได้จากทุกแบบของการเรียงสับเปลี่ยน แต่วิธีนี้จะทำให้เสียเวลาในการคำนวณมาก (ตัวอย่างเช่น ถ้าขนาดตัวอย่างเท่ากับ 20 จะต้องคำนวณค่า $w^{(k)}$

ถึง 20! ครั้ง) ดังนั้นในทางปฏิบัติจึงใช้วิธีสับเปลี่ยนอันดับของข้อมูลอย่างสุ่ม (random permutation) เป็นจำนวนครั้งที่มากพอแทน

จากงานวิจัยของ Yang พบว่า จำนวนครั้งของการเรียงสับเปลี่ยนข้อมูลอย่างสุ่มที่เหมาะสมสำหรับตัวแบบการถดถอยเชิงเส้นทั่วๆ ไปคือ 250 ครั้ง ดังนั้นสามารถเขียนแสดงขั้นตอนต่อไปได้ดังนี้

5. ทำการสับเปลี่ยนลำดับของข้อมูลอย่างสุ่ม แล้วทำซ้ำขั้นที่ 1-4 เป็นจำนวน R-1 รอบ (เมื่อ R คือจำนวนครั้งของการเรียงสับเปลี่ยนข้อมูลทั้งหมดที่กำหนด) ซึ่งในการวิจัยครั้งนี้จะกำหนดให้ R = 250

6. หาค่าเฉลี่ยของน้ำหนักจากทั้ง R รอบ ค่าเฉลี่ยที่ได้จะเป็นน้ำหนักของวิธี ARM กล่าวคือ

$$w_{\mathrm{ARM}}^{(k)} = \frac{1}{R}\sum_{r=1}^{R} w_r^{(k)}$$

เมื่อ $w_r^{(k)}$ แทนน้ำหนักของตัวแบบที่ k ที่ได้จากการคำนวณรอบที่ r (r = 1, …, R)

7. จะได้ $\hat{y}_{i,\mathrm{ARM}} = \sum_{k=1}^{m} w_{\mathrm{ARM}}^{(k)} \hat{y}_i^{(k)}$ เป็นค่าพยากรณ์ร่วมของวีธี ARM ตามต้องการ ซึ่งค่าพยากรณ์ร่วมที่ได้จะเป็น ผลรวมแบบนูน (convex combination) ของค่าพยากรณ์จากทั้ง m ตัวแบบ

บทที่ 3

วิธีดำเนินการวิจัย

การวิจัยครั้งนี้เป็นการศึกษาประสิทธิภาพของวิธีหาค่าพยากรณ์ร่วมของตัวแบบการถดถอยเชิงเส้นพหุคูณ โดยวิธีการที่นำมาศึกษา ได้แก่

1. วิธีค่าสัมบูรณ์ต่ำสุด (least absolute errors: LAE)
2. วิธีบูตสแตรป (combination by bootstrap: BO)
3. วิธี adaptive regression by mixing (ARM)

การวิจัยครั้งนี้ ใช้เทคนิคการจำลองแบบมอนติคาร์โล (Monte Carlo simulation) โดยใช้โปรแกรมภาษาฟอร์แทรน 90 (Fortran 90) บนเครื่องคอมพิวเตอร์ PC ซึ่งแผนการทดลองและขั้นตอนในการวิจัยมีรายละเอียดดังนี้

3.1 แผนการทดลอง

การวิจัยครั้งนี้กำหนดสถานการณ์ต่าง ๆ ที่ต้องการศึกษาดังนี้

1. จำนวนตัวแปรอิสระ (p) ที่ใช้ในการวิจัยมี 3 ระดับ คือ 3, 5 และ 7 ตัว
2. กำหนดขนาดตัวอย่าง (n) ดังนี้

   กรณีที่มีตัวแปรอิสระ 3 ตัว ศึกษาเมื่อขนาดตัวอย่าง = 14, 20, 30, 40 และ 50

   กรณีที่มีตัวแปรอิสระ 5 ตัว ศึกษาเมื่อขนาดตัวอย่าง = 20, 30, 40 และ 50

   กรณีที่มีตัวแปรอิสระ 7 ตัว ศึกษาเมื่อขนาดตัวอย่าง = 30, 40 และ 50
3. ตัวแปรอิสระแต่ละตัวมีการแจกแจงปกติโดยมีค่าเฉลี่ยเท่ากับ 3, 4, 5, 6, 7, 8 และ 9 ตามลำดับ และส่วนเบี่ยงเบนมาตรฐานเท่ากับ 1.2, 2.4, 4.0, 6.0, 8.4, 11.2 และ 14.4 ตามลำดับ
4. ในงานวิจัยนี้ จะแบ่งระดับพหุสัมพันธ์ของตัวแปรอิสระเป็น 3 ระดับ โดยพิจารณาจากค่าสัมประสิทธิ์สหสัมพันธ์ (ρ) ดังนี้

   ระดับต่ำ ρ มีค่าตั้งแต่ 0.1 ถึง 0.3

   ระดับกลาง ρ มีค่าตั้งแต่ 0.4 ถึง 0.6

   ระดับสูง ρ มีค่าตั้งแต่ 0.7 ถึง 0.9

และกำหนดระดับพหุสัมพันธ์ระหว่างตัวแปรอิสระในกรณีต่างๆ ดังนี้

กรณีที่มีตัวแปรอิสระ 3 ตัว ผู้วิจัยจะศึกษาเมื่อตัวแปรอิสระ x_1 กับ x_2 มีความสัมพันธ์กัน โดยกำหนดสัมประสิทธิ์สหสัมพันธ์ ดังนี้

ระดับต่ำ $\rho_{12} = 0.3$

ระดับปานกลาง $\rho_{12} = 0.5$

ระดับสูง $\rho_{12} = 0.8$

(เมื่อ ρ_{ij} คือ สัมประสิทธิ์สหสัมพันธ์ระหว่าง x_i กับ x_j)

กรณีที่มีตัวแปรอิสระ 5 ตัว ผู้วิจัยจะศึกษาเมื่อตัวแปรอิสระมีความสัมพันธ์กันเป็นคู่ๆ ได้แก่ x_1 กับ x_2 และ x_4 กับ x_5 โดยกำหนดสัมประสิทธิ์สหสัมพันธ์ ดังนี้

ระดับต่ำ $\rho_{12} = 0.3$ และ $\rho_{45} = 0.3$

ระดับปานกลาง $\rho_{12} = 0.4$ และ $\rho_{45} = 0.6$

ระดับสูง $\rho_{12} = 0.7$ และ $\rho_{45} = 0.9$

กรณีที่มีตัวแปรอิสระ 7 ตัว ผู้วิจัยจะศึกษาเมื่อตัวแปรอิสระมีความสัมพันธ์กันเป็นคู่ๆ ได้แก่ x_1 กับ x_2 x_4 กับ x_5 และ x_6 กับ x_7 โดยกำหนดสัมประสิทธิ์สหสัมพันธ์ ดังนี้

ระดับต่ำ $\rho_{12} = 0.3,\ \rho_{45} = 0.3$ และ $\rho_{67} = 0.3$

ระดับปานกลาง $\rho_{12} = 0.4,\ \rho_{45} = 0.5$ และ $\rho_{67} = 0.6$

ระดับสูง $\rho_{12} = 0.7,\ \rho_{45} = 0.8$ และ $\rho_{67} = 0.9$

5. กำหนดสัมประสิทธิ์ของการถดถอยพหุคูณ ($\boldsymbol{\beta}$) เป็นเวกเตอร์ของค่าคงที่ใดๆ ในที่นี้เลือกกำหนด $\beta_0 = 6, \beta_1 = 4, \beta_2 = 4, \beta_3 = 2, \beta_4 = 2, \beta_5 = 1, \beta_6 = 1$ และ $\beta_7 = 1$
6. ความคลาดเคลื่อนสุ่ม ($\boldsymbol{\varepsilon}$) มีการแจกแจงปกติที่มีค่าเฉลี่ยเท่ากับ 0 และส่วนเบี่ยงเบนมาตรฐานเท่ากับ 5
7. ตัวแบบการถดถอยที่จะนำมาใช้ในการหาค่าพยากรณ์ร่วมมีดังนี้
   - 7.1 ตัวแบบที่คัดเลือกจากวิธีพิจารณาสมการการถดถอยทุกรูปแบบ (all possible regressions)
   - 7.2 ตัวแบบที่ได้จากวิธีคัดเลือกตัวแปรแบบไปข้างหน้า (forward selection)
   - 7.3 ตัวแบบที่ได้จากวิธีกำจัดตัวแปรแบบถอยหลัง (backward elimination)
   - 7.4 ตัวแบบที่ได้จากวิธีการถดถอยขั้นบันได (stepwise regression)
8. กำหนดการประมวลผลในแต่ละสถานการณ์เท่ากับ 1000 รอบ

3.2 ขั้นตอนในการวิจัย

ขั้นตอนในการวิจัยมีดังนี้

1. จำลองข้อมูลตัวแปรอิสระ ($\mathbf{X}$) ให้มีลักษณะตามที่กำหนดไว้ในแผนการทดลอง
2. กำหนดให้สัมประสิทธิ์ของการถดถอยพหุคูณ ($\boldsymbol{\beta}$) เป็นเวกเตอร์ของค่าคงที่ตามที่กำหนดไว้ในแผนการทดลอง
3. สร้างข้อมูลของความคลาดเคลื่อนสุ่ม ($\boldsymbol{\varepsilon}$) ให้มีการแจกแจงปกติที่มีค่าเฉลี่ยเท่ากับ 0 และส่วนเบี่ยงเบนมาตรฐานเท่ากับ 5
4. สร้างข้อมูลของตัวแปรตาม ($\mathbf{y}$) จากสมการ

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$$

5. สร้างสมการการถดถอยด้วยวิธีต่าง ๆ ดังนี้

   5.1 วิธีพิจารณาสมการการถดถอยทุกรูปแบบ (all possible regressions)

   สำหรับวิธีนี้จะใช้เกณฑ์ MAPE ในการเลือกตัวแบบจากตัวแบบทั้งหมดที่เป็นไปได้ โดยจะเลือกตัวแบบที่มีค่า MAPE ต่ำสุด ตามสูตรดังนี้

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100$$

   โดยที่ y_i เป็นค่าสังเกตที่ i

   $\hat{y}_i$ เป็นค่าพยากรณ์ที่ i

   n เป็นขนาดตัวอย่าง

   5.2 วิธีคัดเลือกตัวแปรแบบไปข้างหน้า (forward selection)

   5.3 วิธีกำจัดตัวแปรแบบถอยหลัง (backward elimination)

   5.4 วิธีการถดถอยขั้นบันได (stepwise regression)

   สำหรับวิธีการในข้อ 5.2 - 5.4 จะกำหนดระดับนัยสำคัญของการนำตัวแปรเข้าสมการ (α_{in}) เท่ากับ 0.05 และระดับนัยสำคัญของการนำตัวแปรออกจากสมการ (α_{out}) เท่ากับ 0.10

6. จากขั้นที่ 5 สมมติว่าได้ตัวแบบมาทั้งหมด m ตัวแบบ $(1 \le m \le 4)$ โดยมีตัวแบบที่ k คือ

$$\hat{\mathbf{y}}^{(k)} = \mathbf{X}^{(k)}\mathbf{b}^{(k)};\ \ k = 1, \ldots, m$$

   เมื่อ $\hat{\mathbf{y}}^{(k)}$ เป็นเวกเตอร์ค่าพยากรณ์ของตัวแปรตามในตัวแบบที่ k

   $\mathbf{X}^{(k)}$ เป็นเมทริกซ์ของตัวแปรอิสระในตัวแบบที่ k

$\mathbf{b}^{(k)}$ เป็นเวกเตอร์ตัวประมาณของสัมประสิทธิ์การถดถอยในตัวแบบที่ k ซึ่งประมาณได้จากข้อมูลตัวอย่าง

7. หาน้ำหนัก $w^{(k)}$ ที่เหมาะสม เพื่อใช้ในการหาค่าพยากรณ์ร่วม โดยวิธีการต่อไปนี้

7.1 วิธีค่าสัมบูรณ์ต่ำสุด

ให้น้ำหนักที่ได้จากวิธีนี้แทนด้วย $w_{\text{LAE}}^{(k)}$, $k = 1, \ldots, m$

7.2 วิธีบูตสแตรป

สำหรับวิธีนี้จะทำการสุ่มตัวอย่างบูตสแตรป (bootstrap sample) มาทั้งหมด 1000 ชุด และให้น้ำหนักที่ได้จากวิธีนี้แทนด้วย $w_{\text{BO}}^{(k)}$, $k = 1, \ldots, m$

7.3 วิธี ARM

สำหรับวิธีนี้จะกำหนดจำนวนครั้งของการสับเปลี่ยนลำดับของข้อมูลอย่างสุ่ม (random permutation) เท่ากับ 250 ครั้ง และให้น้ำหนักที่ได้จากวิธีนี้แทนด้วย $w_{\text{ARM}}^{(k)}$, $k = 1, \ldots, m$

8. คำนวณค่าพยากรณ์ร่วมที่ได้จากแต่ละวิธี ดังนี้

$$\hat{y}_{i,\text{LAE}} = \sum_{k=1}^{m} w_{\text{LAE}}^{(k)} \hat{y}_i^{(k)}$$

$$\hat{y}_{i,\text{BO}} = \sum_{k=1}^{m} w_{\text{BO}}^{(\text{k})} \hat{y}_i^{(k)}$$

$$\hat{y}_{i,\text{ARM}} = \sum_{k=1}^{m} w_{\text{ARM}}^{(k)} \hat{y}_i^{(k)}$$

เมื่อ $i = 1, \ldots, n$

9. คำนวณค่า MAPE ของแต่ละวิธี ดังนี้

$$\text{MAPE}_{\text{LAE}} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_{i,\text{LAE}}}{y_i} \right| \times 100$$

$$\text{MAPE}_{\text{BO}} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_{i,\text{BO}}}{y_i} \right| \times 100$$

$$\text{MAPE}_{\text{ARM}} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_{i,\text{ARM}}}{y_i}\right| \times 100$$

10. ทำซ้ำขั้นตอนที่ 3-9 เป็นจำนวน 1000 รอบ
11. คำนวณค่าเฉลี่ยและส่วนเบี่ยงเบนมาตรฐานของ MAPE สำหรับแต่ละวิธีที่ได้จากการทดลองทั้ง 1000 รอบ

รายละเอียดของวิธีการจำลองข้อมูลมีดังต่อไปนี้

3.2.1 การจำลองความคลาดเคลื่อนให้มีการแจกแจงแบบปกติ[1]

ในการสร้างความคลาดเคลื่อนให้มีการแจกแจงปกติ ต้องใช้เลขสุ่ม (random number) ซึ่งมีการแจกแจงเอกรูป (uniform distribution) ในช่วง $[0,1]$ เป็นพื้นฐาน ซึ่งมีรายละเอียดดังนี้

การสร้างเลขสุ่มที่เป็นอิสระต่อกันและมีการแจกแจงเอกรูปใน $[0,1]$

วิธีการคณิตศาสตร์ในการจำลองเลขสุ่ม (เทียม) มีหลายวิธีการ สำหรับวิธีการที่นิยมใช้กันมากในปัจจุบัน คือ วิธีสมภาค (congruential method) ซึ่งมีสูตรหรือตัวแบบหนึ่งที่ใช้กันมาก คือ

$$X_i = (c + aX_{i-1}) \quad \text{mod} \quad m \quad , \; i = 1,2,\ldots$$

โดยที่ค่า c, a และ m เป็นค่าคงที่จำนวนเต็มที่ไม่เป็นลบ คำว่า mod คือ modulus และความหมายของตัวแบบคือ X_i เป็นเศษเหลือที่เป็นจำนวนเต็มที่ได้จากการหาร $(c + aX_{i-1})$ ด้วย m นั่นคือ $X_i = c + aX_{i-1} - mk_i$ ซึ่ง $k_i = \lfloor (c + aX_{i-1})/m \rfloor$ (หมายถึง จำนวนเต็มใหญ่ที่สุดที่น้อยกว่าหรือเท่ากับผลหาร $(c + aX_{i-1})/m$ ดังนั้น ค่าเป็นไปได้ของ X_i คือ 0, 1, …, m-1 และก่อนที่จะได้ค่าของ $X_1, X_2, \ldots$ ต้องกำหนดค่าของ c, a, m และ X_0 เราเรียก X_0 ว่าซีด (seed) หรือ "ค่าเริ่มต้น" (starting value) จาก X_i ที่ได้จากการคำนวณ นำมาหาค่า R_i ซึ่ง

$$R_i = \frac{X_i}{m} \quad , \; i = 1, 2, \ldots$$

จะได้ R_i มีค่าอยู่ในช่วง $[0,1]$ เรียก $R_1, R_2, R_3, \ldots$ ว่า "เลขสุ่มเทียม" หรือ "เลขสุ่มคล้าย" ทั้งนี้เพราะเมื่อทราบค่าเริ่มต้น X_0 ค่าต่อไปจะมีค่าที่แน่นอนตามสูตร และทุกครั้งที่เริ่มด้วย X_0 ค่าเดิม (ขณะที่ค่า c, a และ m ไม่เปลี่ยนแปลง) จะได้ X_i เป็นเลขชุดเดิม

---

[1] ที่มา: มานพ วราภักดิ์, การจำลองเบื้องต้น (กรุงเทพ: ศูนย์ผลิตตำราเรียนสถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ, 2547)

ถ้ากำหนด $c > 0$ เราเรียกตัวแบบ $X_i = (c + aX_{i-1}) \mod m$, $i = 1,2,...$ ว่า “ตัวแบบจำลองสมภาคแบบผสม” (mixed congruential simulator) แต่ถ้ากำหนด $c = 0$ เราเรียกตัวแบบ $X_i = (c + aX_{i-1}) \mod m$, $i = 1,2,...$ ว่า “ตัวแบบจำลองสมภาคแบบผลคูณ” (multiplicative congruential simulator)

ตัวแบบจำลองสมภาคแบบผลคูณที่ใช้กันมากตัวแบบหนึ่ง ซึ่งได้ผ่านการตรวจสอบคุณสมบัติแล้วอย่างมาก คือ สำหรับคอมพิวเตอร์ 32 บิตต่อ 1 คำ (32 bit word) กำหนด $m = 2^{31} - 1 = 2147483647$, $a = 7^5 = 16807$ และ X_0 เป็นจำนวนเต็มบวกไม่เกิน m ซึ่งตัวแบบดังกล่าวจะเป็นตัวแบบที่ใช้จำลองเลขสุ่มในการวิจัยครั้งนี้

การแจกแจงปกติ

ตัวแปรสุ่ม X มีการแจกแจงปกติ (normal distribution) ด้วยพารามิเตอร์ μ และ σ^2 เขียนแทนด้วย $X \sim N(\mu,\sigma^2)$ ถ้า X มีฟังก์ชันความหนาแน่น

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \quad , \ -\infty < x < \infty , -\infty < \mu < \infty , 0 < \sigma^2 < \infty$$

สำหรับ $X \sim N(\mu,\sigma^2)$ พิสูจน์ได้ว่า

$$E(X) = \mu$$
$$\text{var}(X) = \sigma^2$$

ในการจำลองตัวแปรสุ่มปกติ X ที่มีค่าเฉลี่ย μ และความแปรปรวน σ^2 นั้น ถ้าเราจำลองตัวแปรสุ่มปกติมาตรฐาน (standard normal distribution) $Z \sim N(0,1)$ ได้แล้ว สามารถจำลอง X ได้ด้วยสูตร $X = \mu + \sigma Z$ ซึ่งจะได้ $X \sim N(\mu,\sigma^2)$ ดังนั้น วิธีการต่อไปนี้ จะเป็นวิธีการจำลอง $Z \sim N(0,1)$

การจำลองตัวแปรสุ่มปกติด้วยวิธีโพลาร์

วิธีโพลาร์ของ Marsaglia, MacLaren และ Bray (1964) จะเริ่มจากการ จำลอง V_1 จาก $U(-1,1)$ และจำลอง V_2 จาก $U(-1, 1)$ อย่างอิสระกัน และจะได้ตัวแบบจำลองตัวแปรสุ่ม $Z_1 \sim N(0,1)$ และ $Z_2 \sim N(0,1)$ อิสระกัน คือ

$$Z_1 = V_1\sqrt{\frac{-2\ell n(V_1^2+V_2^2)}{V_1^2+V_2^2}}$$

$$Z_2 = V_2\sqrt{\frac{-2\ell n(V_1^2+V_2^2)}{V_1^2+V_2^2}}$$

เขียนแสดงขั้นตอนวิธีได้ดังนี้

1. จำลองเลขสุ่ม R_1 และ R_2
2. $V_1 = 2R_1 - 1, V_2 = 2R_2 - 1$ (จำลอง V_1, V_2 จาก $U(-1,1)$)
3. $S = V_1^2 + V_2^2$
4. ถ้า $S > 1$ กลับไปขั้นตอน 1
5. $W = \sqrt{\frac{-2\ell nS}{S}}$
6. $Z_1 = V_1W$, $Z_2 = V_2W$

3.2.2 การสร้างตัวแปรอิสระให้มีความสัมพันธ์กันในระดับต่างๆ[2]

ในการวิจัยครั้งนี้จะสร้างตัวแปรอิสระ $\mathbf{X}$ ให้มีการแจกแจงปกติหลายตัวแปรและมีความสัมพันธ์กันในระดับต่างๆ โดยมีรายละเอียด ดังนี้

เวกเตอร์สุ่ม $\mathbf{X} = (X_1, X_2, ..., X_p)^T$ มีการแจกแจงปกติหลายตัวแปร (multivariate normal distribution) เขียนแทนด้วย $\mathbf{X} \sim N_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ ถ้า $\mathbf{X}$ มีฟังก์ชันความหนาแน่นร่วมคือ

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}|^{1/2}}\exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right] \quad , -\infty < x_i < \infty,\ i = 1, 2, ..., p$$

ซึ่ง $\mathbf{x} = (x_1, x_2, ..., x_p)^T$, $\boldsymbol{\mu}$ เป็นเวกเตอร์ของค่าเฉลี่ย $(\mu_1, \mu_2, ..., \mu_p)^T$ ซึ่ง $\mu_i = E(X_i)$, i = 1, 2, …, p และ $\boldsymbol{\Sigma}$ เป็นเมทริกซ์ความแปรปรวนร่วม (covariance matrix) ขนาด $p \times p$ และเป็นเมทริกซ์บวกแน่นอน (positive definite matrix)

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1p} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2p} \\ \vdots & & & \\ \sigma_{p1} & \sigma_{p2} & \cdots & \sigma_{pp} \end{bmatrix}$$

โดยที่ $\sigma_{ij} = \mathrm{cov}(X_i, X_j) = \sigma_{ji} = \mathrm{cov}(X_j, X_i)$ สำหรับ $i \neq j$ และ $\sigma_{ii} = \mathrm{var}(X_i)$, i = 1, 2, …, p; j = 1, 2, …, p

[2] ที่มา: มานพ วราภักดิ์, การจำลองเบื้องต้น (กรุงเทพ: ศูนย์ผลิตตำราเรียนสถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ, 2547)

เนื่องจาก $\mathbf{\Sigma}$ เป็นเมทริกซ์บวกแน่นอน ดังนั้น เขียน $\mathbf{\Sigma}$ ได้เป็น

$$\mathbf{\Sigma} = \mathbf{C}\mathbf{C}^T$$

โดยที่ $\mathbf{C}$ เป็นเมทริกซ์สามเหลี่ยมล่าง (lower triangular matrix):

$$\mathbf{C} = \begin{bmatrix} c_{11} & 0 & 0 & \ldots & 0 \\ c_{21} & c_{22} & 0 & \ldots & 0 \\ c_{31} & c_{32} & c_{33} & \cdots & 0 \\ \cdots & & & & \\ c_{p1} & c_{p2} & c_{p3} & \ldots & c_{pp} \end{bmatrix}$$

ฉะนั้น ถ้า $\mathbf{Z} = (Z_1, Z_2, ..., Z_p)^T$ โดยที่ $Z_1, Z_2, ..., Z_p$ เป็นอิสระกัน และต่างมีการแจกแจง $N(0,1)$ จะเขียน $\mathbf{X}$ ได้เป็น

$$\mathbf{X} = \mathbf{C}\mathbf{Z} + \boldsymbol{\mu}$$

ซึ่งได้ $\mathbf{X}$ มีการแจกแจงปกติหลายตัวแปร และ

$$\begin{aligned} E(\mathbf{X}) &= \boldsymbol{\mu} \\ \mathrm{var}(\mathbf{X}) &= \mathbf{C}\,\mathrm{var}(\mathbf{Z})\mathbf{C}^T = \mathbf{C}\mathbf{I}\mathbf{C}^T = \mathbf{\Sigma} \end{aligned}$$

เพราะฉะนั้น ในการจำลอง $\mathbf{X} \sim N_p(\boldsymbol{\mu}, \mathbf{\Sigma})$ จะจำลอง $\mathbf{Z}$ จากนั้น จำลอง $\mathbf{X}$ ด้วยสูตร $\mathbf{X} = \mathbf{C}\mathbf{Z} + \boldsymbol{\mu}$ ซึ่งต้องทราบค่า c_{ij} ใน $\mathbf{C}$ ด้วย ซึ่งมีวิธีการหาได้ โดยวิธีรากที่สอง (square root method) โดยสูตรการคำนวณ c_{ij} เป็นดังนี้

$$c_{ij} = \frac{\sigma_{ij} - \sum_{k=1}^{j=1} c_{ik} c_{jk}}{c_{jj}} \quad , \quad 1 \le j \le i \le p$$

โดยที่

$$c_{jj} = \left(\sigma_{jj} - \sum_{k=1}^{j=1} c_{jk}^2 \right)^{1/2} \quad , \quad j = 1, 2, ..., p$$

$$\sum_{k=1}^{0} c_{ik} c_{jk} = 0$$

เมื่อกำหนดค่าของ p, σ_{ij}, μ_i เป็นข้อมูลเข้า จะจำลอง $\mathbf{X} \sim N_p(\boldsymbol{\mu}, \mathbf{\Sigma})$ ได้ตามขั้นตอนวิธีดังนี้

1. $a = \sqrt{\sigma_{11}}$
2. สำหรับ $i = 1, 2, \ldots, p$ ให้

$$c_{i1} = \sigma_{1i} / a$$

3. $i = 2$
4. $c_{ii} = \left(\sigma_{ii} - \sum_{k=1}^{i-1} c_{ik}^2 \right)^{1/2}$
5. ถ้า $i = p$ ไปขั้นตอน 9
6. $i = i + 1$
7. สำหรับ $j = 2, 3, ..., i-1$ ให้

$$c_{ij} = \left(\sigma_{ij} - \sum_{k=1}^{j-1} c_{ik} c_{jk} \right) \Big/ c_{jj}$$

8. ไปขั้นตอน 4
9. สำหรับ $i = 1, 2, ..., p$

จำลอง $Z_i \sim N(0, 1)$

10. สำหรับ $i = 1, 2, ..., p$ ให้

$$X_i = \mu_i + \sum_{j=1}^{i} c_{ij} Z_j$$

11. จบขั้นตอน ข้อมูลออกคือค่าของ $X_1, X_2, ..., X_p$

3.2.3 วิธีบูตสแตรป[3]

เนื่องจากการหาน้ำหนักโดยวิธี BO ต้องอาศัยการสุ่มตัวอย่างโดยวิธีบูตสแตรป ดังนั้นในหัวข้อนี้จะกล่าวถึงรายละเอียดของวิธีบูตสแตรป

การใช้ข้อมูลหลายๆ ชุด เพื่อใช้ในการอนุมานเชิงสถิติ เพื่อให้ได้ผลการอนุมานที่ดีขึ้น โดยทั่วไปในทางปฏิบัติจะกระทำไม่ได้ (ปกติจะมีข้อมูลชุดเดียว) **"**วิธีบูตสแตรป**"** (bootstrapping) หรือ **"**วิธีการสุ่มตัวอย่างซ้ำ**"** (resampling procedures) เป็นวิธีการหนึ่งที่จะได้ตัวอย่างหลาย ๆ ชุด

จากตัวแบบการถดถอย

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} + \varepsilon_i, \quad i = 1,2,\ldots,n$$

ในการวิจัยนี้จะใช้วิธีการสุ่มซ้ำข้อมูล $(y_i, x_{i1}, x_{i2}, \cdots, x_{ip}),\quad i = 1,2,\ldots,n$ โดยมีขั้นตอนดังนี้

1. จากข้อมูลที่มีอยู่ 1 ชุดประกอบด้วย n แถว คือ $(y_i, x_{i1}, x_{i2}, \ldots, x_{ip}),\ i = 1,2,\ldots,n$ เลือกข้อมูลอย่างสุ่มแบบคืนที่จำนวน n แถว โดยให้แต่ละแถวมีความน่าจะเป็นที่จะถูกเลือกเท่ากัน เท่ากับ $\dfrac{1}{n}$ ซึ่งทำได้โดยการจำลอง I จากการแจกแจงเอกรูปแบบไม่ต่อเนื่องบนเซต

[3] ที่มา: มานพ วราภักดิ์, การจำลองเบื้องต้น (กรุงเทพ: ศูนย์ผลิตตำราเรียนสถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ, 2547)

$\{1, 2, \dots, n\}$ ได้ $I = \lceil nR \rceil$ โดยจำลอง R จาก $U(0,1)$ ค่า I ที่ได้จะบอกถึงข้อมูลแถวที่ I ถูกเลือก เช่น $I = 5$ หมายถึงข้อมูลแถวที่ 5 คือ $(y_5, x_{51}, x_{52}, \dots, x_{5p})$ ถูกเลือก ข้อมูลที่ถูกเลือกจะแทนด้วย $(y_i^*, x_{i1}^*, x_{i2}^*, \dots, x_{ip}^*)$, $\quad i = 1,2,\dots,n$

2. ทำขั้นตอน 1 N ครั้ง จะได้ข้อมูลตัวอย่างบูตสแตรป จำนวน N ชุด

3.2.4 การจัดเรียงอย่างสุ่ม (Random Permutation)[4]

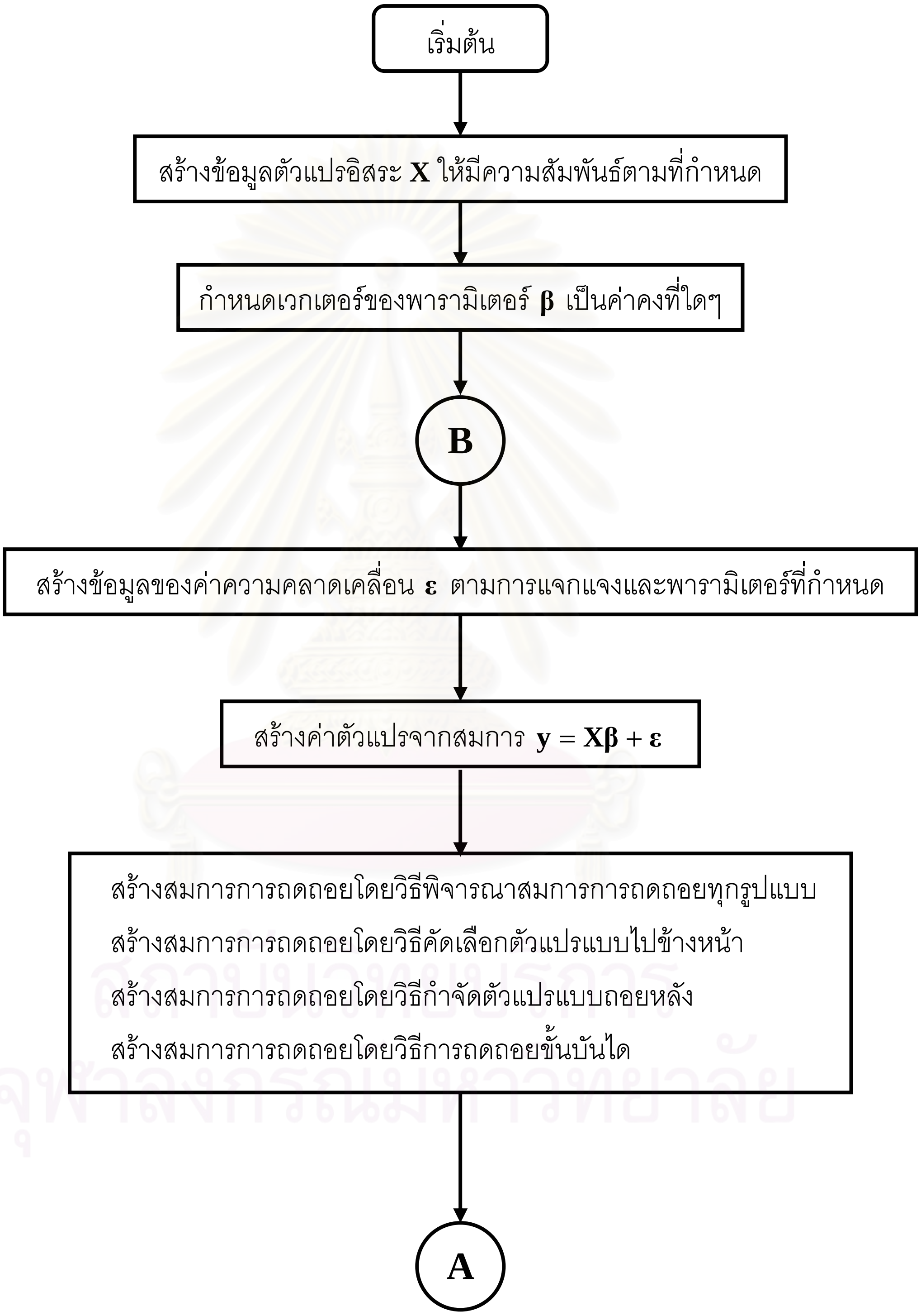
หัวข้อนี้จะกล่าวถึงการจัดเรียงอย่างสุ่ม ซึ่งจะต้องนำไปใช้ในวิธี ARM

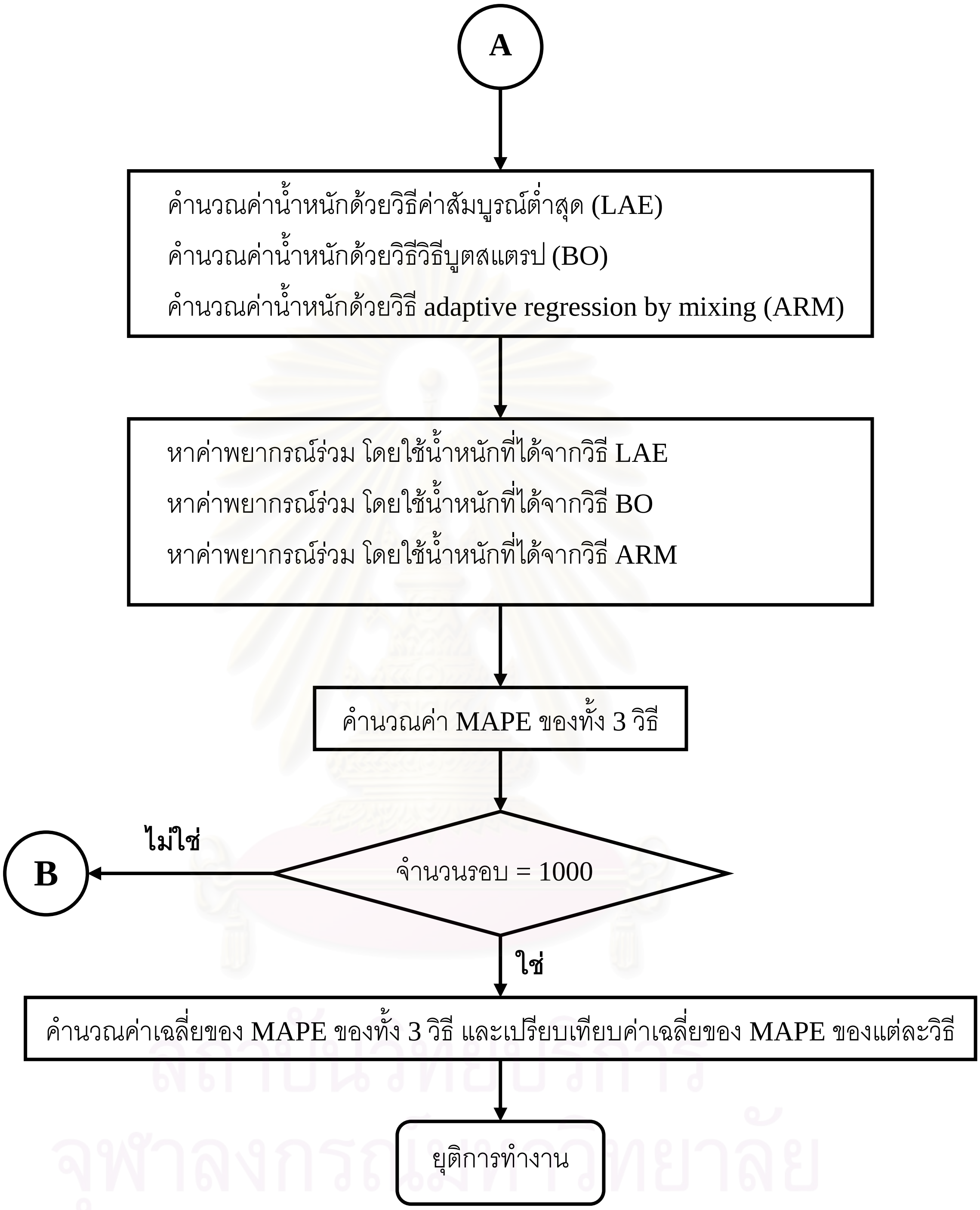
การจัดเรียงหมายเลข 1, 2, 3, …, n อย่างสุ่ม มีหนทางการจัดเรียงได้ $n!$ หนทาง เราต้องการหนึ่งการจัดเรียงอย่างสุ่มจากรูปแบบการจัดเรียงที่เป็นไปได้ทั้งหมด $n!$ วิธีการหนึ่งที่ทำได้คือ จะเริ่มด้วยรูปแบบการจัดเรียงรูปแบบหนึ่งให้เป็น 1, 2, 3, …, n (จะเริ่มด้วยการเรียงแบบอื่นก็ได้) จากนั้น สุ่มตำแหน่งของหมายเลขเหล่านี้จากตำแหน่งที่ 1 ถึง n สมมติได้ตำแหน่งที่ I จะสับเปลี่ยนตัวเลขในตำแหน่งที่ I กับตำแหน่งสุดท้ายคือตำแหน่งที่ n เช่นถ้าสุ่มได้ $I = 3$ จะสับเปลี่ยนตัวเลขระหว่าง 3 กับ n ให้เลข 3 ไปอยู่ตำแหน่งที่ n (ตัวสุดท้าย) และให้ n มาอยู่แทน 3 ในตำแหน่ง 3 จากนั้น สุ่มตำแหน่งต่อไปจากตำแหน่งที่เหลือ คือ ตำแหน่งที่ 1 ถึง $n-1$ เมื่อได้หมายเลขตำแหน่งก็สับเปลี่ยนตัวเลขในตำแหน่งที่ได้กับตัวเลขในตำแหน่งที่ $n-1$ ทำเช่นนี้เรื่อยไปจนถึงการสุ่มตำแหน่งที่เหลือสุดท้าย คือ ตำแหน่งที่ 1 และ 2 และทำการสับเปลี่ยนตัวเลขระหว่างตำแหน่งที่ 1 และ 2 ถ้าได้หมายเลขตำแหน่งที 1 เขียนเป็นขั้นตอนวิธีได้ดังนี้

1. สำหรับ $l = 1$ ถึง n (ค่า l เพิ่มขึ้นทีละหนึ่ง) ทำขั้นตอน 2
2. ให้ $X(l) = 1$
3. $j = 1$
4. ถ้า $j > n$ ไปขั้นตอน 13
5. $k = n + 1 - j$
6. จำลองเลขสุ่ม R
7. $I = \lceil kR \rceil$
8. $Y = X(I)$
9. $X(I) = X(k)$
10. $X(k) = Y$
11. $j = j + 1$
12. ไปขั้นตอน 4
13. จบขั้นตอน ขั้นมูลออกคือค่าของ $X(1), X(2), \dots, X(n)$ ตามลำดับ

[4] ที่มา: มานพ วราภักดิ์, การจำลองเบื้องต้น (กรุงเทพ: ศูนย์ผลิตตำราเรียนสถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ, 2547)

3.3 แผนผังขั้นตอนการทำงาน




A
คำนวณค่าน้ำหนักด้วยวิธีค่าสัมบูรณ์ต่ำสุด (LAE)
คำนวณค่าน้ำหนักด้วยวิธีวิธีบูตสแตรป (BO)
คำนวณค่าน้ำหนักด้วยวิธี adaptive regression by mixing (ARM)
หาค่าพยากรณ์ร่วม โดยใช้น้ำหนักที่ได้จากวิธี LAE
หาค่าพยากรณ์ร่วม โดยใช้น้ำหนักที่ได้จากวิธี BO
หาค่าพยากรณ์ร่วม โดยใช้น้ำหนักที่ได้จากวิธี ARM
คำนวณค่า MAPE ของทั้ง 3 วิธี
จำนวนรอบ = 1000
ไม่ใช่
B
ใช่
คำนวณค่าเฉลี่ยของ MAPE ของทั้ง 3 วิธี และเปรียบเทียบค่าเฉลี่ยของ MAPE ของแต่ละวิธี
ยุติการทำงาน

3.4 รายละเอียดของโปรแกรมที่ใช้ในการวิจัย

ในการวิจัยครั้งนี้ ผู้วิจัยเขียนโปรแกรมด้วยภาษาฟอร์แทรน 90 บนเครื่องคอมพิวเตอร์ PC ซึ่งรายละเอียดทั้งหมดของโปรแกรมจะแสดงในตารางที่ 3.1

ตารางที่ 3.1 โปรแกรมทั้งหมดที่ใช้ในการวิจัย

อันดับที่	ชื่อโปรแกรม	การทำงานของโปรแกรม	ชื่อโปรแกรมย่อยที่เรียกใช้
โปรแกรมหลัก	thesis_main	- อ่านขนาดตัวอย่าง, จำนวนตัวแปรอิสระ, ค่าสัมประสิทธิ์สหสัมพันธ์ระหว่างตัวแปรอิสระที่กำหนด	
		- คำนวณเมทริกซ์ความแปรปรวนร่วม	covariance
		- อ่านค่าส่วนเบี่ยงเบนมาตรฐานของความคลาดเคลื่อนสุ่มที่กำหนด	
		- อ่านค่าซีด (seed)	
		- สร้างเมทริกซ์ของตัวแปรอิสระ **X**	mnorm
		- อ่านจำนวนชุดของตัวอย่างบูตสแตรปที่ต้องการ	
		- อ่านจำนวนรอบของการจำลองที่ต้องการ	
		- สร้างความคลาดเคลื่อนสุ่ม	znorm
		- สร้างข้อมูลตัวแปรตาม	
		- สร้างตัวแบบ 4 วิธี และสรุปตัวแบบที่ได้ทั้งหมด	models_4methods, models_to_be_combined
		- หาค่าน้ำหนักโดยวิธี LAE	comb_lp
		- หาค่าน้ำหนักโดยวิธี BO	comb_bootstrap
		- หาค่าน้ำหนักโดยวิธี ARM	arm

ตารางที่ 3.1 (ต่อ)

อันดับที่	ชื่อโปรแกรม	การทำงานของโปรแกรม	ชื่อโปรแกรมย่อยที่เรียกใช้
		- คำนวณค่า $\hat{\mathbf{y}}$ ของแต่ละตัวแบบที่จะนำมาหาค่าพยากรณ์ร่วม - คำนวณค่าพยากรณ์ร่วม $(\hat{\mathbf{y}}^{combined})$ โดยใช้น้ำหนักที่ได้จาก 3 วิธี - คำนวณค่า MAPE ของแต่ละวิธี - คำนวณค่าเฉลี่ยของ MAPE ของแต่ละวิธี - คำนวณส่วนเบี่ยงเบนมาตรฐานของ MAPE ของแต่ละวิธี	
โปรแกรมย่อย			
1	arm	- หาน้ำหนักด้วยวิธี ARM	ols, perm_random2
2	comb_bootstrap	- หาน้ำหนักด้วยวิธี BO	bootstrap4models, ols, invm
3	bootstrap4models	- สุ่มตัวอย่างโดยวิธีบูตสแตรป	i_random
4	comb_lp	- หาน้ำหนักด้วยวิธี LAE	min_lp
5	min_lp	- แก้ปัญหาโปรแกรมเชิงเส้นในรูปแบบที่ศึกษาด้วยวิธีซิมเพลกซ์	invm
6	perm_random2	- การจัดเรียงอย่างสุ่ม	i_swap, i_random
7	i_random	- จำลองเลขสุ่มจากการแจกแจงเอกรูปแบบไม่ต่อเนื่องบนเซต $\{1,2,\ldots,n\}$	urand

38

ตารางที่ 3.1 (ต่อ)

อันดับที่	ชื่อโปรแกรม	การทำงานของโปรแกรม	ชื่อโปรแกรมย่อยที่เรียกใช้
8	i_swap	- สลับตำแหน่งของจำนวนเต็ม 2 จำนวน	
9	models_4methods	- สร้างตัวแบบจาก 4 วิธี	allpossible, partial_f, forward, backward, stepwise, ols
10	models_to_be_combined	- สรุปตัวแบบที่จะนำมาหาค่าพยากรณ์ร่วม	
11	allpossible	- สร้างตัวแบบโดยวิธีพิจารณาการถดถอยทุกรูปแบบ โดยเลือกตัวแบบที่มีค่า MAPE ต่ำสุด	ols
12	forward	- สร้างตัวแบบโดยวิธีคัดเลือกตัวแปรแบบไปข้างหน้า	ols
13	backward	- สร้างตัวแบบโดยวิธีกำจัดตัวแปรแบบถอยหลัง	ols, sortinteger
14	sortinteger	- เรียงตัวเลขจำนวนเต็มใน array จากน้อยไปมาก	
15	stepwise	- สร้างตัวแบบโดยวิธีการถดถอยขั้นบันได	ols
16	corr	- คำนวณค่าสหสัมพันธ์ระหว่างตัวแปร 2 ตัว	
17	finv	- ค่าสถิติเอฟ ณ ระดับนัยสำคัญและระดับขั้นความเสรีที่กำหนด	
18	partial_f	- คำนวณค่าสถิติทดสอบเอฟบางส่วน	ols
19	ols	คำนวณค่าสัมประสิทธิ์การถดถอยพร้อมค่าสถิติที่เกี่ยวข้องด้วยวิธีกำลังสองน้อยสุด	invm

ตารางที่ 3.1 (ต่อ)

อันดับที่	ชื่อโปรแกรม	การทำงานของโปรแกรม	ชื่อโปรแกรมย่อยที่เรียกใช้
20	invm	- คำนวณเมทริกซ์ผกผัน	
21	covariance	- คำนวณเมทริกซ์ความแปรปรวนร่วม	
22	mnorm	- สร้างเลขสุ่มให้มีการแจกแจงปกติหลายตัวแปร	znorm
23	znorm	- สร้างเลขสุ่มให้มีการแจกแจงปกติ	urand
24	urand	- สร้างเลขสุ่มให้มีการแจกแจงเอกรูปใน $[0,1]$	

บทที่ 4

ผลการวิจัย

การวิจัยครั้งนี้มีวัตถุประสงค์เพื่อเปรียบเทียบวิธีหาค่าพยากรณ์ร่วมของตัวแบบการถดถอยเชิงเส้นพหุคูณ โดยผู้วิจัยได้ทำการเปรียบเทียบวิธีการ 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์การเปรียบเทียบคือ ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ ซึ่งมีวิธีการคำนวณดังนี้

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100$$

โดยที่ y_i เป็นค่าสังเกตที่ i

$\hat{y}_i$ เป็นค่าพยากรณ์ที่ i

n เป็นขนาดตัวอย่าง

MAPE เป็นร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ของวิธีการที่พิจารณา

โดยในการวิจัยครั้งนี้จะทำการทดลองเป็นจำนวน 1000 รอบ และหา**ค่าเฉลี่ยของ MAPE** ของแต่ละวิธี

ผู้วิจัยจะเสนอผลการวิจัยโดยแบ่งออกเป็น 3 ส่วน ดังนี้

ส่วนที่ 1 ผลการเปรียบเทียบวิธีหาค่าพยากรณ์ร่วมในกรณีที่จำนวนตัวแปรอิสระเท่ากับ 3

ส่วนที่ 2 ผลการเปรียบเทียบวิธีหาค่าพยากรณ์ร่วมในกรณีที่จำนวนตัวแปรอิสระเท่ากับ 5

ส่วนที่ 3 ผลการเปรียบเทียบวิธีหาค่าพยากรณ์ร่วมในกรณีที่จำนวนตัวแปรอิสระเท่ากับ 7

สำหรับการนำเสนอผลการวิจัยจะนำเสนอในรูปแบบตารางและกราฟ เพื่อความสะดวกในการอธิบายจึงใช้สัญลักษณ์ต่อไปนี้เพื่อแทนความหมายต่าง ๆ

p หมายถึง จำนวนตัวแปรอิสระ

n หมายถึง ขนาดตัวอย่าง

ρ_{ij} หมายถึง ค่าสัมประสิทธิ์สหสัมพันธ์ระหว่างตัวแปรอิสระ x_i กับ x_j

LAE หมายถึง การหาค่าพยากรณ์ร่วมโดยวิธีค่าสัมบูรณ์ต่ำสุด

BO หมายถึง การหาค่าพยากรณ์ร่วมโดยวิธีบูตสแตรป

ARM หมายถึง การหาค่าพยากรณ์ร่วมโดยวิธี adaptive regression by mixing

MAPE หมายถึง ค่าร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์

S.D. หมายถึง ส่วนเบี่ยงเบนมาตรฐานของค่าร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์

4.1 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วมในกรณีที่จำนวนตัวแปรอิสระเท่ากับ 3

ในกรณีนี้ ผู้วิจัยทำการศึกษาเมื่อขนาดตัวอย่างเท่ากับ 14, 20, 30, 40 และ 50 ตามลำดับ โดยทำการศึกษาในกรณีต่าง ๆ ดังนี้

4.1.1 กรณีตัวแปรอิสระมีความสัมพันธ์ระดับต่ำ (ρ_{12} = 0.30) ซึ่งผลการวิจัยส่วนนี้จะนำเสนอในตารางที่ 4.1.1 และกราฟรูปที่ 4.1.1

4.1.2 กรณีตัวแปรอิสระมีความสัมพันธ์ระดับปานกลาง (ρ_{12} = 0.50) ซึ่งผลการวิจัยส่วนนี้จะนำเสนอในตารางที่ 4.1.2 และกราฟรูปที่ 4.1.2

4.1.3 กรณีตัวแปรอิสระมีความสัมพันธ์ระดับสูง (ρ_{12} = 0.80) ซึ่งผลการวิจัยส่วนนี้จะนำเสนอในตารางที่ 4.1.3 และกราฟรูปที่ 4.1.3

นอกจากนี้ผู้วิจัยยังได้เสนอผลการวิจัยในกราฟรูปที่ 4.1.4 - 4.1.9 โดยกราฟนี้จะแสดงค่าเฉลี่ยของ MAPE เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป ณ ขนาดตัวอย่างแต่ละระดับ

ตารางที่ 4.1.1 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) ในกรณีที่จำนวนตัวแปรอิสระ = 3 และตัวแปรอิสระมีความสัมพันธ์กันในระดับต่ำ

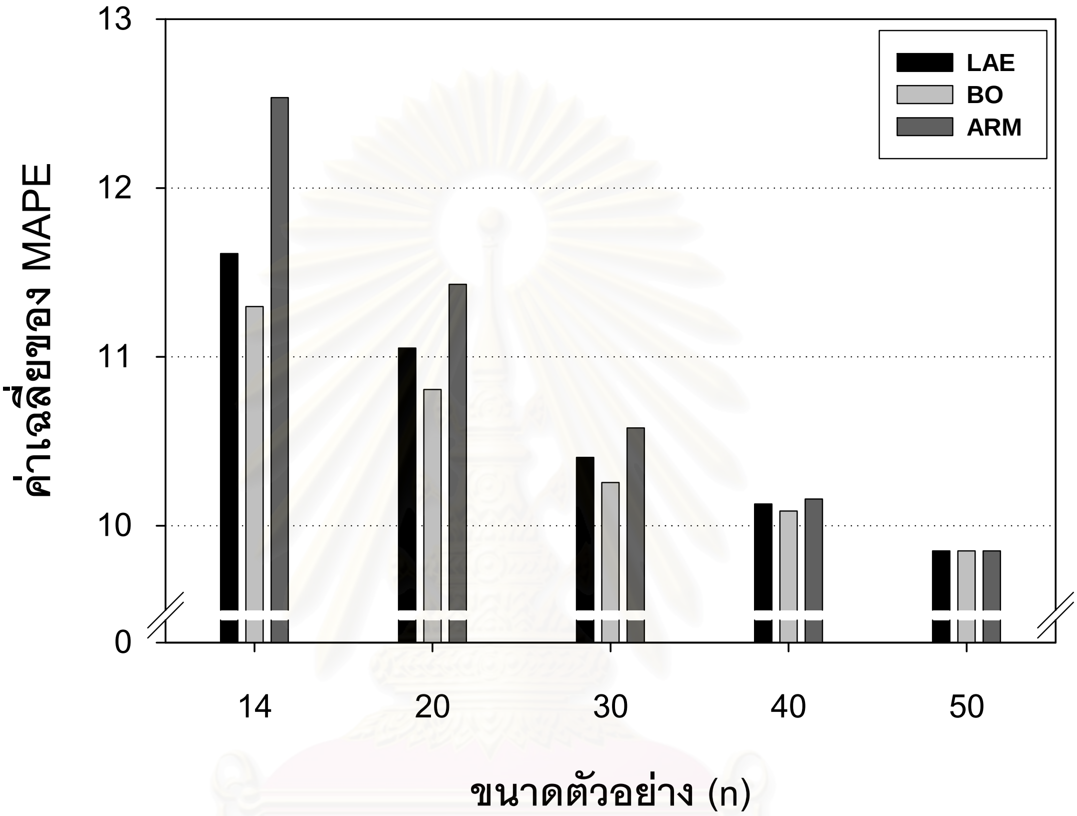
(ตัวเลขในวงเล็บคือ ส่วนเบี่ยงเบนมาตรฐาน (S.D.) ของ MAPE)

ระดับความสัมพันธ์ของตัวแปรอิสระ	ขนาดตัวอย่าง (n)	ค่าเฉลี่ยของ MAPE		
		LAE	BO	ARM
0.3	14	11.611008	11.297491 *	12.535362
		(3.256565)	(3.155663)	(3.744026)
	20	11.051885	10.806841*	11.429337
		(2.296547)	(2.162019)	(2.453322)
	30	10.404278	10.256296*	10.578956
		(1.740676)	(1.620157)	(1.836790)
	40	10.128608	10.087915*	10.158022
		(1.458207)	(1.444452)	(1.496710)
	50[†]	9.851171	9.851171	9.851171
		(1.393120)	(1.393120)	(1.393120)

* หมายถึง วิธีหาค่าพยากรณ์ร่วมที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด

[†]เมื่อ n = 50 ทุกวิธีให้ค่าเฉลี่ยของ MAPE เท่ากัน ทั้งนี้เพราะในทุกๆ รอบของการทดลองพบว่าวิธีสร้างตัวแบบ 4 วิธี ต่างก็ให้ตัวแบบออกมาเหมือนกัน 1 ตัวแบบ ทำให้ไม่มีการหาค่าพยากรณ์ร่วม

รูปที่ 4.1.1 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) กรณีที่ จำนวนตัวแปรอิสระ = 3 และตัวแปรอิสระมีความสัมพันธ์กันในระดับต่ำ



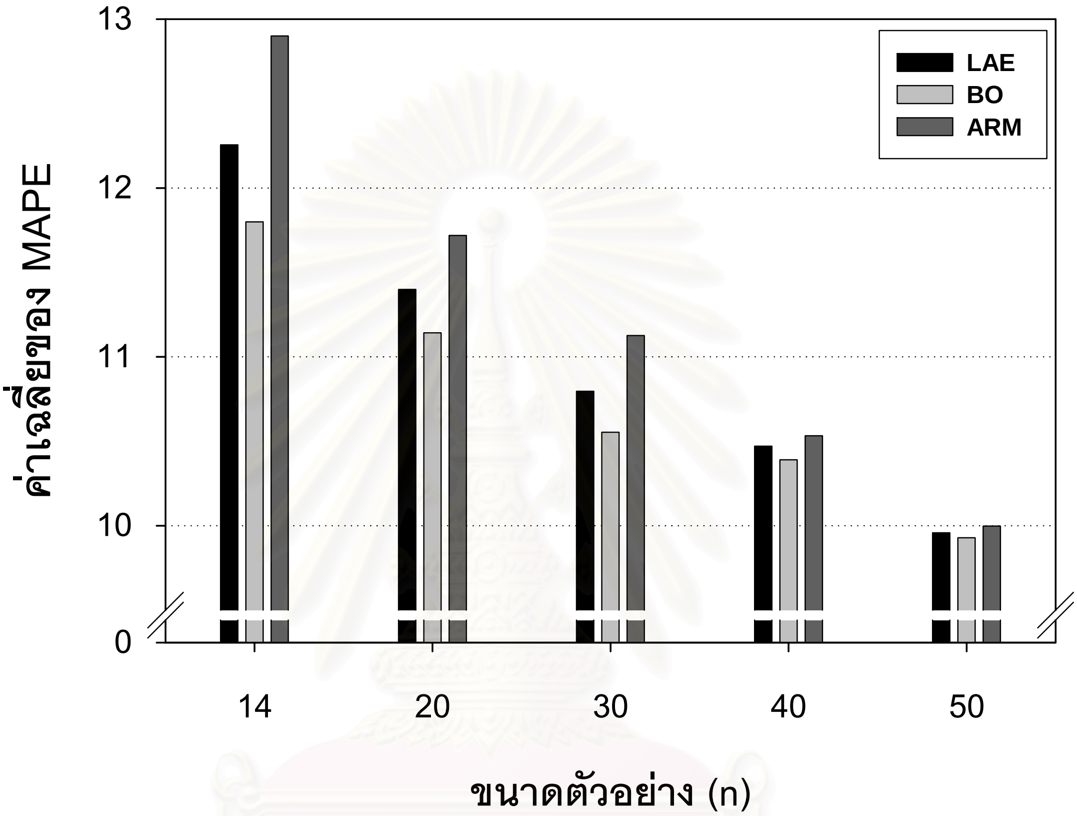
ตารางที่ 4.1.2 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) ในกรณีที่จำนวนตัวแปรอิสระ = 3 และตัวแปรอิสระมีความสัมพันธ์กันในระดับปานกลาง

(ตัวเลขในวงเล็บคือ ส่วนเบี่ยงเบนมาตรฐาน (S.D.) ของ MAPE)

ระดับความสัมพันธ์ของตัวแปรอิสระ	ขนาดตัวอย่าง (n)	ค่าเฉลี่ยของ MAPE		
		LAE	BO	ARM
0.5	14	12.255981	11.798019*	12.899991
		(3.225336)	(3.154775)	(3.404904)
	20	11.398889	11.142151*	11.717825
		(2.419001)	(2.283235)	(2.609334)
	30	10.796317	10.553008*	11.126207
		(1.818017)	(1.745989)	(1.895031)
	40	10.471069	10.389905*	10.532561
		(1.719982)	(1.620086)	(1.718063)
	50	9.958143	9.927339*	9.992432
		(1.412544)	(1.413867)	(1.418974)

* หมายถึง วิธีหาค่าพยากรณ์ร่วมที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด

รูปที่ 4.1.2 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) กรณีที่ จำนวนตัวแปรอิสระ = 3 และตัวแปรอิสระมีความสัมพันธ์กันในระดับปานกลาง



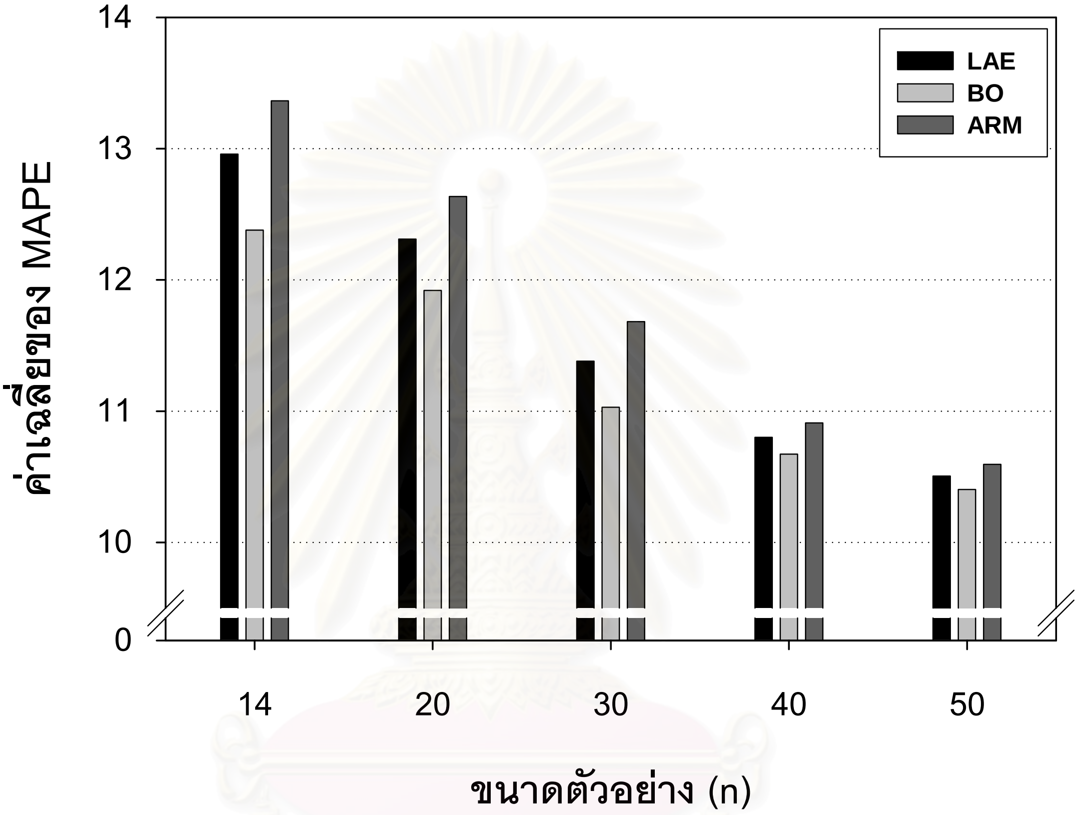
ตารางที่ 4.1.3 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) ในกรณีที่จำนวนตัวแปรอิสระ = 3 และตัวแปรอิสระมีความสัมพันธ์กันในระดับสูง

(ตัวเลขในวงเล็บคือ ส่วนเบี่ยงเบนมาตรฐาน (S.D.) ของ MAPE)

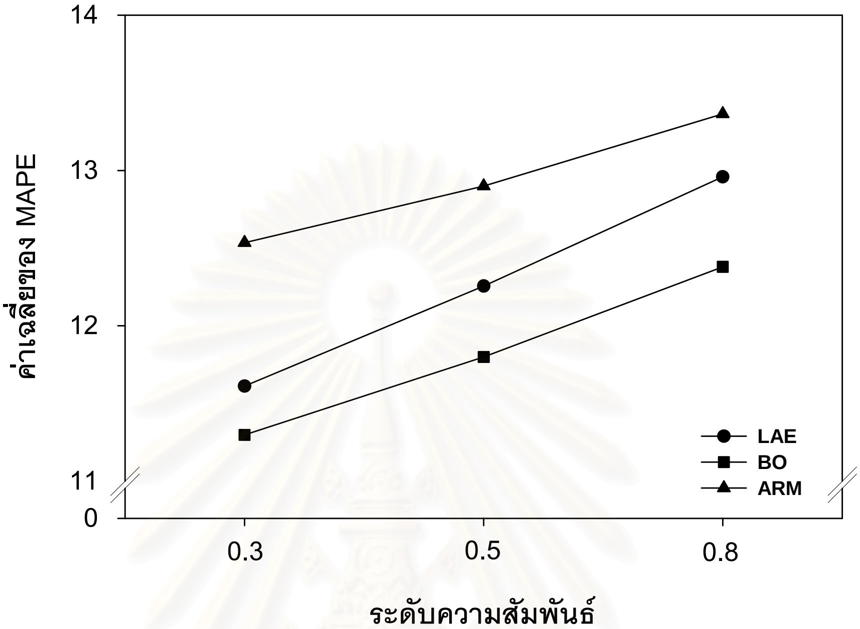
ระดับความสัมพันธ์ของตัวแปรอิสระ	ขนาดตัวอย่าง (n)	ค่าเฉลี่ยของ MAPE		
		LAE	BO	ARM
0.8	14	12.959306	12.379634*	13.364847
		(3.549879)	(3.383405)	(3.711074)
	20	12.310470	11.919763*	12.635369
		(3.334250)	(3.258728)	(3.433982)
	30	11.380243	11.029051*	11.681595
		(2.134116)	(2.010944)	(2.138259)
	40	10.799278	10.672130*	10.910117
		(1.770932)	(1.748090)	(1.813585)
	50	10.506131	10.402621*	10.594066
		(1.453087)	(1.420065)	(1.498479)

* หมายถึง วิธีหาค่าพยากรณ์ร่วมที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด

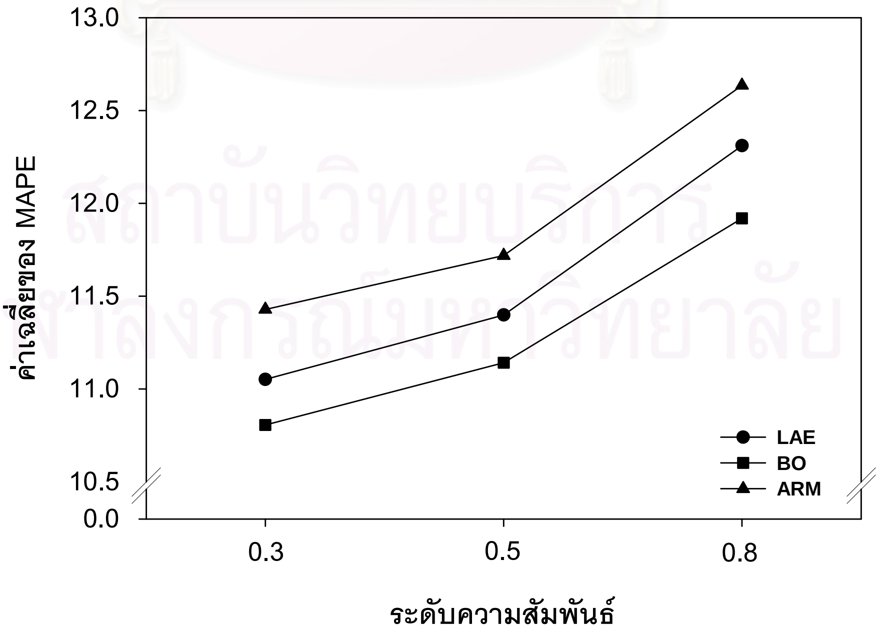
รูปที่ 4.1.3 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) กรณีที่ จำนวนตัวแปรอิสระ = 3 และตัวแปรอิสระมีความสัมพันธ์กันในระดับสูง



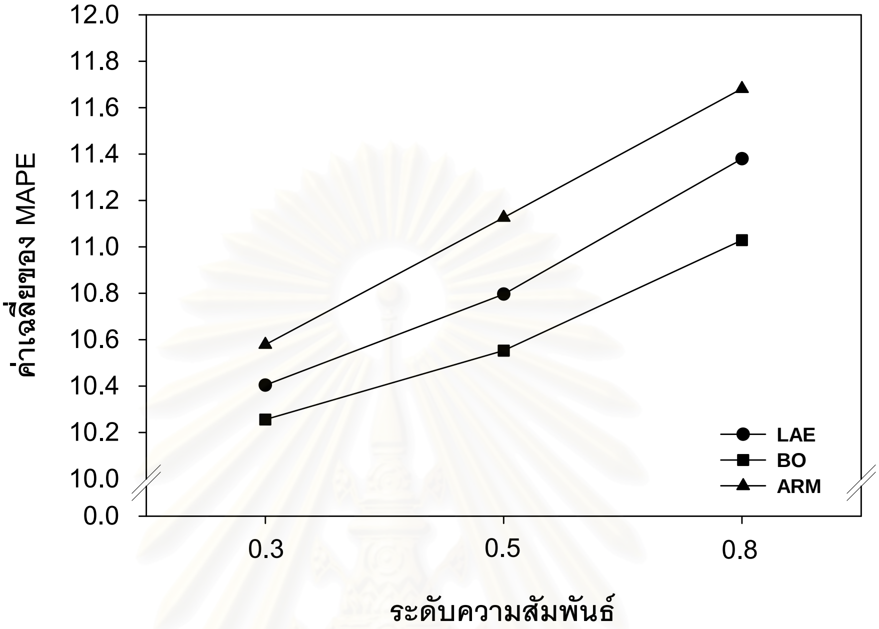
รูปที่ 4.1.4 แสดงการเปรียบเทียบค่า MAPE ของวิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป กรณีจำนวนตัวแปรอิสระ = 3 และขนาดตัวอย่าง = 14



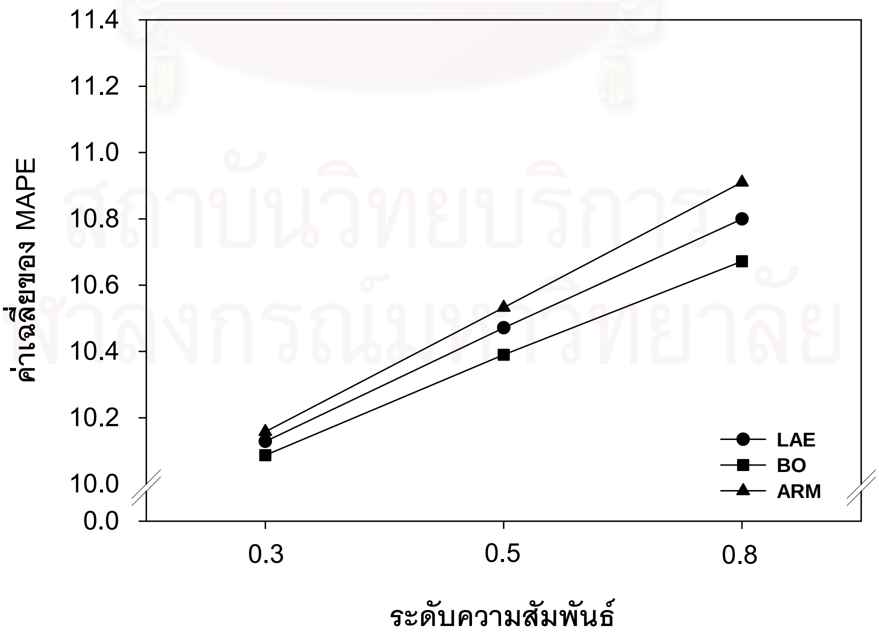
รูปที่ 4.1.5 แสดงการเปรียบเทียบค่า MAPE ของวิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป กรณีจำนวนตัวแปรอิสระ = 3 และขนาดตัวอย่าง = 20



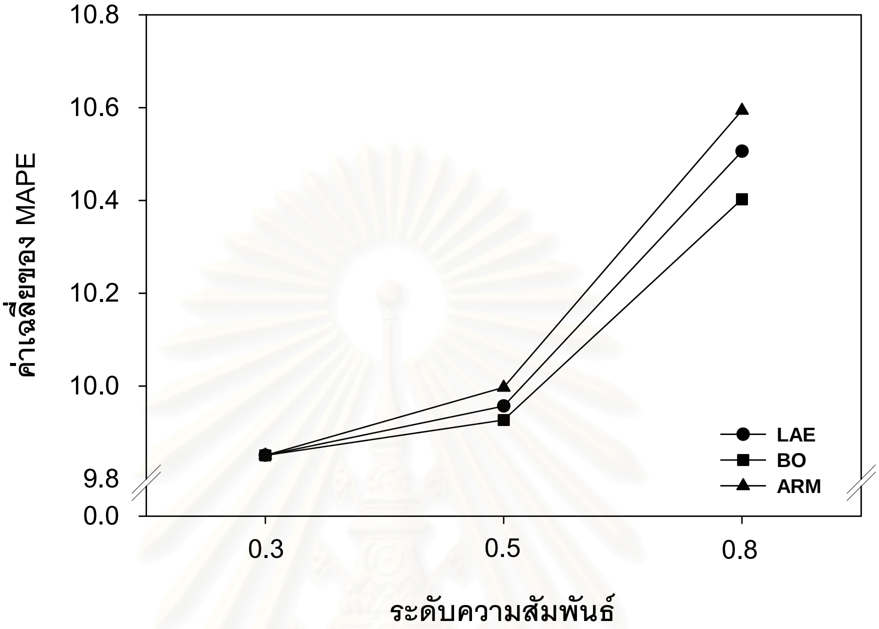
รูปที่ 4.1.6 แสดงการเปรียบเทียบค่า MAPE ของวิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป กรณีจำนวนตัวแปรอิสระ = 3 และขนาดตัวอย่าง = 30



รูปที่ 4.1.7 แสดงการเปรียบเทียบค่า MAPE ของวิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป กรณีจำนวนตัวแปรอิสระ = 3 และขนาดตัวอย่าง = 40



รูปที่ 4.1.8 แสดงการเปรียบเทียบค่า MAPE ของวิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป กรณีจำนวนตัวแปรอิสระ = 3 และขนาดตัวอย่าง = 50



จากตารางและรูปที่ 4.1.1 - 4.1.3 เราสามารถสรุปผลการเปรียบเทียบค่าเฉลี่ยของ MAPE ของวิธี LAE วิธี BO และวิธี ARM เมื่อจำนวนตัวแปรอิสระเท่ากับ 3 จำแนกตามระดับพหุสัมพันธ์ได้ดังนี้

1. ความสัมพันธ์ระดับต่ำ

สำหรับความสัมพันธ์ระดับต่ำ (ρ_{12} = 0.30) พบว่า เมื่อขนาดตัวอย่างเท่ากับ 14 ถึง 40 วิธี BO ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับ

ในกรณีนี้มีข้อสังเกตคือ เมื่อขนาดตัวอย่างเป็น 40 ค่าเฉลี่ยของ MAPE จะใกล้เคียงกันมาก สาเหตุที่เป็นเช่นนี้เพราะในขั้นตอนการสร้างตัวแบบ 4 วิธี ปรากฏว่าทั้ง 4 วิธีให้ตัวแบบออกมาเหมือนกันหรือใกล้เคียงกันในหลาย ๆ รอบของการทดลอง ส่วนในกรณีที่ขนาดตัวอย่างเท่ากับ 50 ปรากฏว่าวิธีสร้างตัวแบบทุกวิธีต่างก็ให้ตัวแบบออกมาเหมือนกันในทุก ๆ รอบของการทดลอง จึงได้ตัวแบบเพียง 1 ตัวแบบเท่านั้น ทำให้ไม่มีการหาค่าพยากรณ์ร่วม ส่งผลให้ค่าเฉลี่ยของ MAPE ของทุกวิธีมีค่าเท่ากัน

2. ความสัมพันธ์ระดับปานกลาง

สำหรับความสัมพันธ์ระดับปานกลาง (ρ_{12} = 0.50) พบว่าวิธี BO ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับ สำหรับทุกขนาดตัวอย่าง

ในกรณีนี้มีข้อสังเกตคือ เมื่อขนาดตัวอย่างเป็น 50 ค่าเฉลี่ยของ MAPE จะใกล้เคียงกันมาก สาเหตุที่เป็นเช่นนี้เพราะในขั้นตอนการสร้างตัวแบบ 4 วิธี ปรากฏว่าทั้ง 4 วิธีให้ตัวแบบออกมาเหมือนกันหรือใกล้เคียงกันในหลาย ๆ รอบของการทดลอง

3. ความสัมพันธ์ระดับสูง

สำหรับความสัมพันธ์ระดับสูง (ρ_{12} = 0.80) พบว่าวิธี BO ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับ สำหรับทุกขนาดตัวอย่าง

ในกรณีนี้มีข้อสังเกตคือ เมื่อขนาดตัวอย่างเป็น 50 ค่าเฉลี่ยของ MAPE จะใกล้เคียงกันมาก สาเหตุที่เป็นเช่นนี้เพราะในขั้นตอนการสร้างตัวแบบ 4 วิธี ปรากฏว่าทั้ง 4 วิธีให้ตัวแบบออกมาเหมือนกันหรือใกล้เคียงกันในหลาย ๆ รอบของการทดลอง

ข้อสรุป

ข้อสรุปสำหรับกรณีจำนวนตัวแปรอิสระเท่ากับ 3 (หัวข้อ 4.1) เป็นดังนี้

สำหรับทุกระดับความสัมพันธ์และขนาดตัวอย่าง พบว่าวิธี BO จะเป็นวิธีที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับ และจะสังเกตได้ว่าค่าเฉลี่ยของ MAPE มีแนวโน้มเพิ่มขึ้นเมื่อระดับความสัมพันธ์ของตัวแปรอิสระเพิ่มขึ้น ทั้งนี้เพราะเมื่อระดับความสัมพันธ์สูงขึ้นจะเกิดปัญหาพหุสัมพันธ์รุนแรงขึ้น ทำให้ตัวแบบแต่ละตัวแบบที่นำมาหาค่าพยากรณ์ร่วมมีความแปรปรวนของตัวประมาณสูงขึ้น ดังนั้น เมื่อนำค่าพยากรณ์จากแต่ละตัวแบบมาวิเคราะห์ร่วมกัน ย่อมทำให้ค่าเฉลี่ยของ MAPE ของการพยากรณ์ร่วมมีค่าสูงขึ้น นอกจากนี้ยังพบว่าค่าเฉลี่ยของ MAPE ของทุกวิธีจะมีแนวโน้มลดลงเมื่อขนาดตัวอย่างเพิ่มขึ้น ทั้งนี้เพราะเมื่อขนาดตัวอย่างเพิ่มขึ้น จะส่งผลให้ความแปรปรวนลดลง ทำให้สามารถแก้ปัญหาพหุสัมพันธ์ระหว่างตัวแปรอิสระลงได้บ้าง

4.2 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วมในกรณีที่จำนวนตัวแปรอิสระเท่ากับ 5

ในกรณีนี้ ผู้วิจัยทำการศึกษาเมื่อขนาดตัวอย่างเท่ากับ 20, 30, 40 และ 50 ตามลำดับ โดยทำการศึกษาในกรณีต่าง ๆ ดังนี้

4.2.1 กรณีตัวแปรอิสระมีความสัมพันธ์ระดับต่ำ (ρ_{12} = 0.30, ρ_{45} = 0.30) ซึ่งผลการวิจัยส่วนนี้จะนำเสนอในตารางที่ 4.2.1 และกราฟรูปที่ 4.2.1

4.2.2 กรณีตัวแปรอิสระมีความสัมพันธ์ระดับปานกลาง (ρ_{12} = 0.40, ρ_{45} = 0.60) ซึ่งผลการวิจัยส่วนนี้จะนำเสนอในตารางที่ 4.2.2 และกราฟรูปที่ 4.2.2

4.2.3 กรณีตัวแปรอิสระมีความสัมพันธ์ระดับสูง (ρ_{12} = 0.70, ρ_{45} = 0.90) ซึ่งผลการวิจัยส่วนนี้จะนำเสนอในตารางที่ 4.2.3 และกราฟรูปที่ 4.2.3

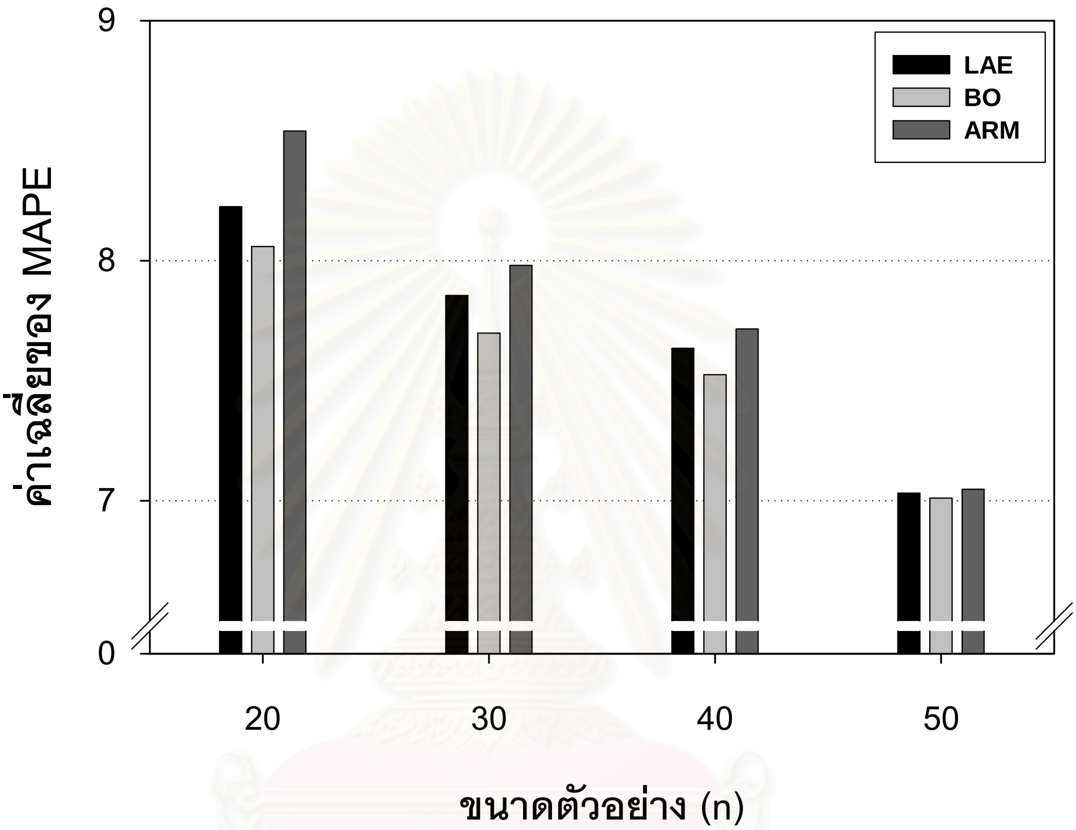
นอกจากนี้ผู้วิจัยยังได้เสนอผลการวิจัยในกราฟรูปที่ 4.2.4 - 4.2.9 โดยกราฟนี้จะแสดงค่าเฉลี่ยของ MAPE เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป ณ ขนาดตัวอย่างแต่ละระดับ

ตารางที่ 4.2.1 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) ในกรณีที่จำนวนตัวแปรอิสระ = 5 และตัวแปรอิสระมีความสัมพันธ์กันในระดับต่ำ
(ตัวเลขในวงเล็บคือ ส่วนเบี่ยงเบนมาตรฐาน (S.D.) ของ MAPE))

ระดับความสัมพันธ์ของตัวแปรอิสระ	ขนาดตัวอย่าง (n)	ค่าเฉลี่ยของ MAPE		
		LAE	BO	ARM
0.3, 0.3	20	8.225218	8.059649*	8.539927
		(1.812755)	(1.786824)	(2.057156)
	30	7.855328	7.698304*	7.980153
		(1.341018)	(1.326808)	(1.337432)
	40	7.634836	7.524965*	7.715374
		(1.196118)	(1.156120)	(1.185542)
	50	7.031257	7.011252*	7.048216
		(0.960815)	(0.960154)	(0.961455)

* หมายถึง วิธีหาค่าพยากรณ์ร่วมที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด

รูปที่ 4.2.1 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM)โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) กรณีที่ จำนวนตัวแปรอิสระ = 5 และตัวแปรอิสระมีความสัมพันธ์กันในระดับต่ำ

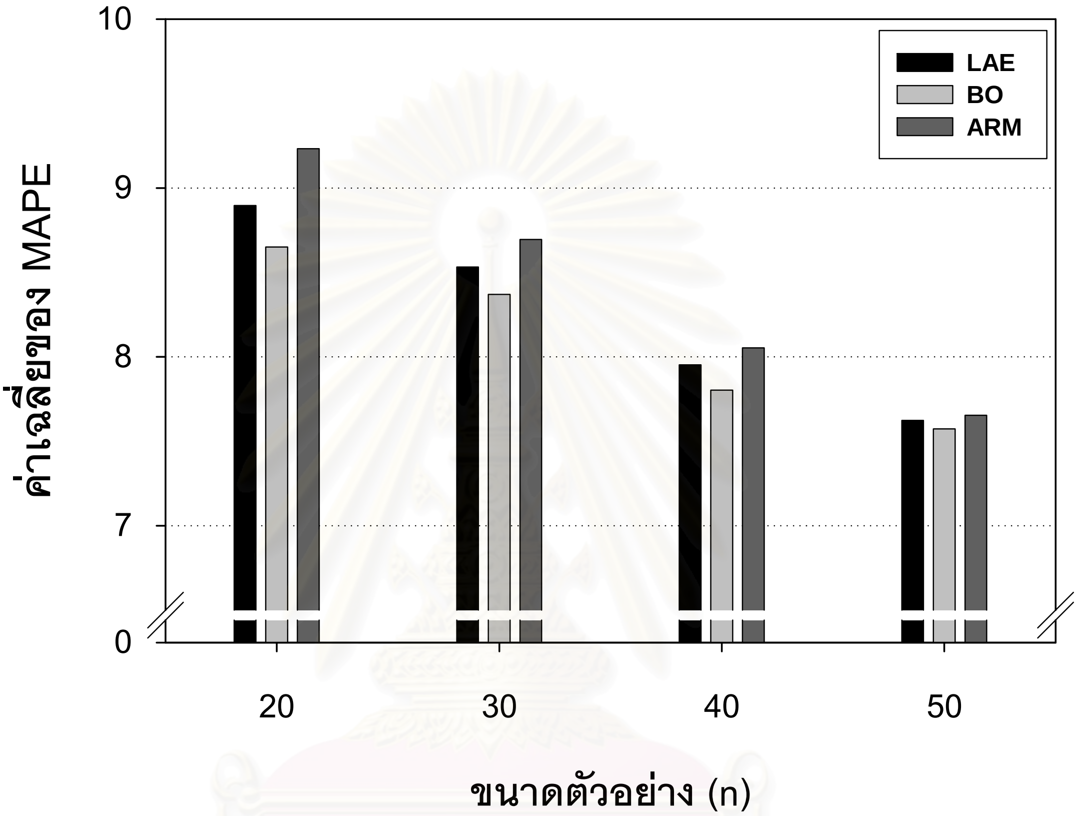


ตารางที่ 4.2.2 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) ในกรณีที่จำนวนตัวแปรอิสระ = 5 และตัวแปรอิสระมีความสัมพันธ์กันในระดับปานกลาง
(ตัวเลขในวงเล็บคือ ส่วนเบี่ยงเบนมาตรฐาน (S.D.) ของ MAPE)

ระดับความสัมพันธ์ของตัวแปรอิสระ	ขนาดตัวอย่าง (n)	ค่าเฉลี่ยของ MAPE		
		LAE	BO	ARM
0.4, 0.6	20	8.896202 (2.139924)	8.650221* (2.001070)	9.232244 (2.297605)
	30	8.532102 (1.699207)	8.369946* (1.617604)	8.695107 (1.719315)
	40	7.952439 (1.218755)	7.801985* (1.187191)	8.053237 (1.280838)
	50	7.623038 (0.966373)	7.573255* (0.973918)	7.653493 (0.974815)

* หมายถึง วิธีหาค่าพยากรณ์ร่วมที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด

รูปที่ 4.2.2 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) กรณีที่ จำนวนตัวแปรอิสระ = 5 และตัวแปรอิสระมีความสัมพันธ์กันในระดับปานกลาง

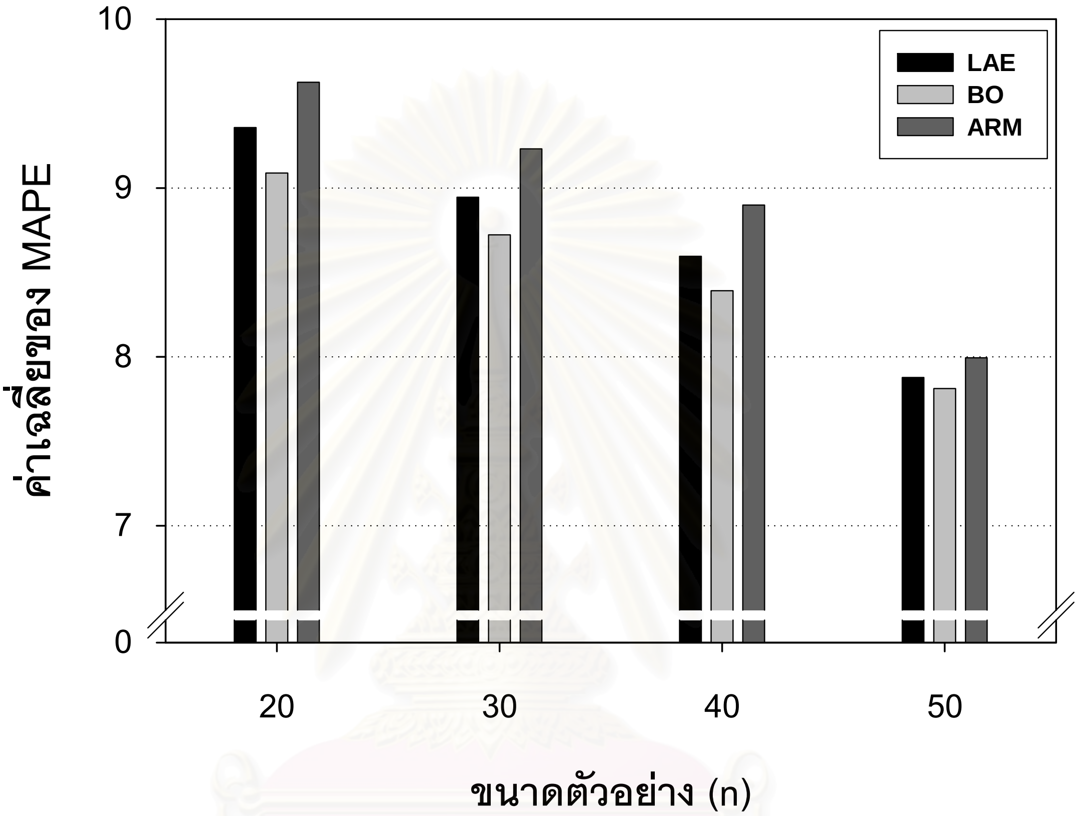


ตารางที่ 4.2.3 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) ในกรณีที่จำนวนตัวแปรอิสระ = 5 และตัวแปรอิสระมีความสัมพันธ์กันในระดับสูง
(ตัวเลขในวงเล็บคือ ส่วนเบี่ยงเบนมาตรฐาน (S.D.) ของ MAPE)

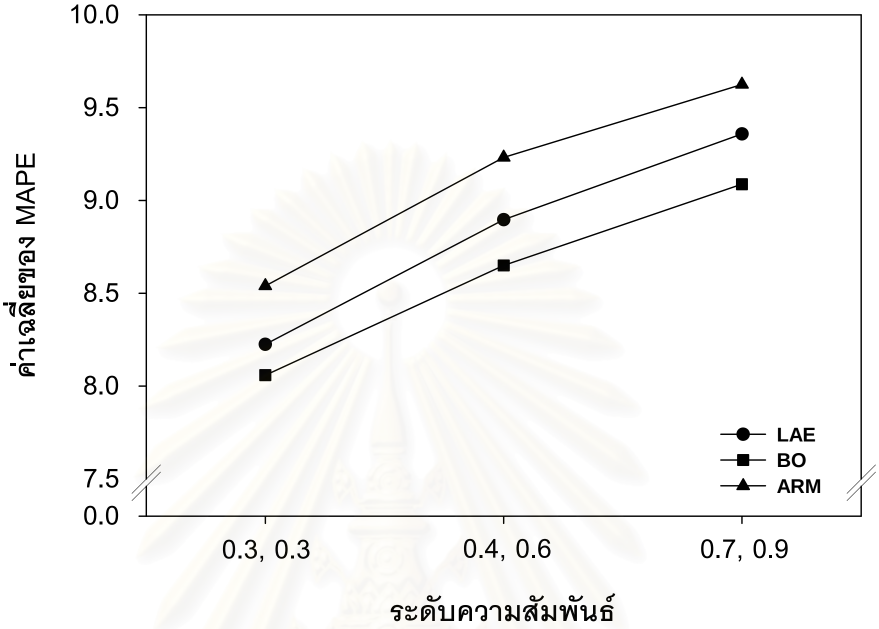
ระดับความสัมพันธ์ของตัวแปรอิสระ	ขนาดตัวอย่าง (n)	ค่าเฉลี่ยของ MAPE		
		LAE	BO	ARM
0.7, 0.9	20	9.358575	9.088102*	9.625632
		(2.120400)	(3.741372)	(2.288086)
	30	8.945754	8.722450*	9.231866
		(1.596882)	(1.563942)	(1.629181)
	40	8.595707	8.392031*	8.899076
		(1.355460)	(1.331613)	(1.367027)
	50	7.877534	7.811190*	7.994322
		(1.146616)	(1.131492)	(1.157631)

* หมายถึง วิธีหาค่าพยากรณ์ร่วมที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด

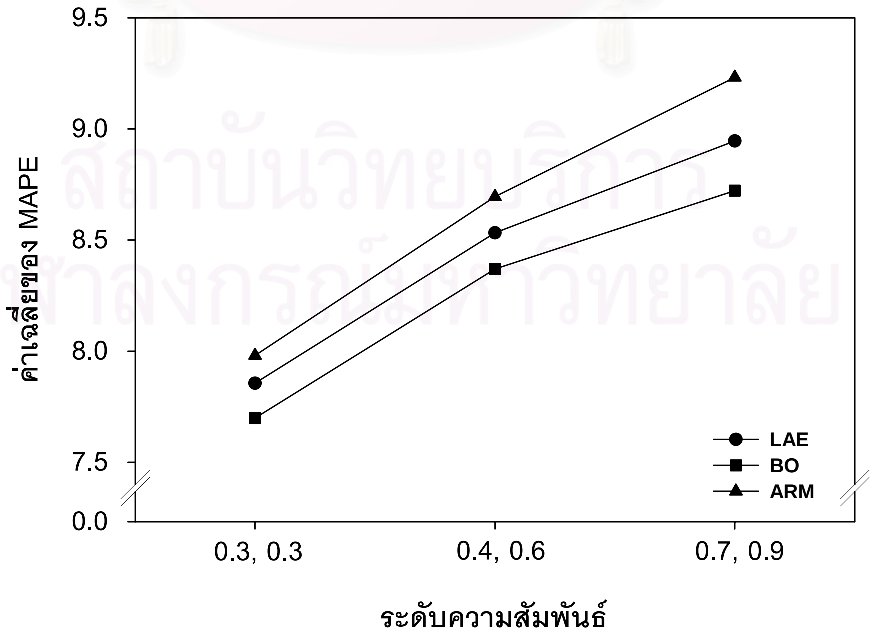
รูปที่ 4.2.3 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) กรณีที่ จำนวนตัวแปรอิสระ = 5 และตัวแปรอิสระมีความสัมพันธ์กันในระดับสูง



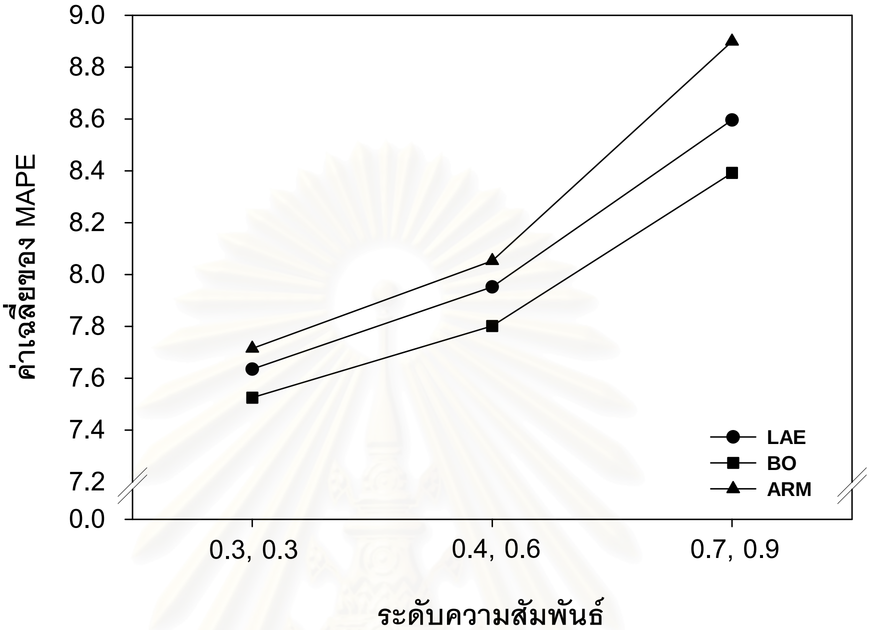
รูปที่ 4.2.4 แสดงการเปรียบเทียบค่า MAPE ของวิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป กรณีจำนวนตัวแปรอิสระ = 5 และขนาดตัวอย่าง = 20



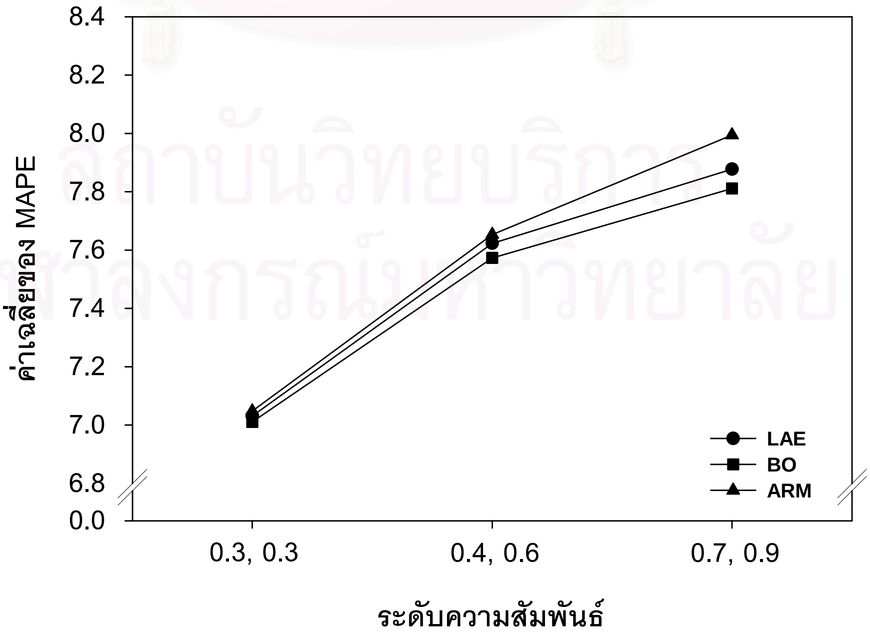
รูปที่ 4.2.5 แสดงการเปรียบเทียบค่า MAPE ของวิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป กรณีจำนวนตัวแปรอิสระ = 5 และขนาดตัวอย่าง = 30



รูปที่ 4.2.6 แสดงการเปรียบเทียบค่า MAPE ของวิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป กรณีจำนวนตัวแปรอิสระ = 5 และขนาดตัวอย่าง = 40



รูปที่ 4.2.7 แสดงการเปรียบเทียบค่า MAPE ของวิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป กรณีจำนวนตัวแปรอิสระ = 5 และขนาดตัวอย่าง = 50



จากตารางและรูปที่ 4.2.1 - 4.2.3 เราสามารถสรุปผลการเปรียบเทียบค่าเฉลี่ยของ MAPE ของวิธี LAE วิธี BO และวิธี ARM เมื่อจำนวนตัวแปรอิสระเท่ากับ 5 จำแนกตามระดับพหุสัมพันธ์ได้ดังนี้

1. ความสัมพันธ์ระดับต่ำ

สำหรับความสัมพันธ์ระดับต่ำ (ρ_{12} = 0.30, ρ_{45} = 0.30) พบว่าวิธี BO ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับ สำหรับทุกขนาดตัวอย่าง

ในกรณีนี้มีข้อสังเกตคือ เมื่อขนาดตัวอย่างเป็น 50 ค่าเฉลี่ยของ MAPE จะใกล้เคียงกันมาก สาเหตุที่เป็นเช่นนี้เพราะในขั้นตอนการสร้างตัวแบบ 4 วิธี ปรากฏว่าทั้ง 4 วิธีให้ตัวแบบออกมาเหมือนกันหรือใกล้เคียงกันในหลาย ๆ รอบของการทดลอง

2. ความสัมพันธ์ระดับปานกลาง

สำหรับความสัมพันธ์ระดับปานกลาง (ρ_{12} = 0.40, ρ_{45} = 0.60) พบว่าวิธี BO ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับ สำหรับทุกขนาดตัวอย่าง

ในกรณีนี้มีข้อสังเกตคือ เมื่อขนาดตัวอย่างเป็น 50 ค่าเฉลี่ยของ MAPE จะใกล้เคียงกันมาก สาเหตุที่เป็นเช่นนี้เพราะในขั้นตอนการสร้างตัวแบบ 4 วิธี ปรากฏว่าทั้ง 4 วิธีให้ตัวแบบออกมาเหมือนกันหรือใกล้เคียงกันในหลาย ๆ รอบของการทดลอง

3. ความสัมพันธ์ระดับสูง

สำหรับความสัมพันธ์ระดับสูง (ρ_{12} = 0.70, ρ_{45} = 0.90) พบว่าวิธี BO ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับ สำหรับทุกขนาดตัวอย่าง

ข้อสรุป

ข้อสรุปสำหรับกรณีจำนวนตัวแปรอิสระเท่ากับ 5 (หัวข้อ 4.2 เป็นดังนี้)

สำหรับทุกระดับความสัมพันธ์และขนาดตัวอย่าง พบว่าวิธี BO จะเป็นวิธีที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับ และจะสังเกตได้ว่าค่าเฉลี่ยของ MAPE มีแนวโน้มเพิ่มขึ้นเมื่อระดับความสัมพันธ์ของตัวแปรอิสระเพิ่มขึ้น ทั้งนี้เพราะเมื่อระดับความสัมพันธ์สูงขึ้นจะเกิดปัญหาพหุสัมพันธ์รุนแรงขึ้น ทำให้ตัวแบบแต่ละตัวแบบที่นำมาหาค่าพยากรณ์ร่วมมีความแปรปรวนของตัวประมาณสูงขึ้น ดังนั้น เมื่อนำค่าพยากรณ์จากแต่ละตัวแบบมาวิเคราะห์ร่วมกัน ย่อมทำให้ค่าเฉลี่ยของ MAPE ของการพยากรณ์ร่วมมีค่าสูงขึ้น นอกจากนี้ยังพบว่าค่าเฉลี่ยของ MAPE ของทุกวิธีจะมีแนวโน้มลดลงเมื่อขนาดตัวอย่างเพิ่มขึ้น ทั้งนี้เพราะเมื่อขนาดตัวอย่างเพิ่มขึ้น จะส่งผลให้ความแปรปรวนลดลง ทำให้สามารถแก้ปัญหาพหุสัมพันธ์ระหว่างตัวแปรอิสระลงได้บ้าง

4.3 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วมในกรณีที่จำนวนตัวแปรอิสระเท่ากับ 7

ในกรณีนี้ ผู้วิจัยทำการศึกษาเมื่อขนาดตัวอย่างเท่ากับ 30, 40 และ 50 ตามลำดับ โดยทำการศึกษาในกรณีต่าง ๆ ดังนี้

4.3.1 กรณีตัวแปรอิสระมีความสัมพันธ์ระดับต่ำ (ρ_{12} = 0.30, ρ_{45} = 0.30, ρ_{67} = 0.30) ซึ่งผลการวิจัยส่วนนี้จะนำเสนอในตารางที่ 4.3.1 และกราฟรูปที่ 4.3.1

4.3.2 กรณีตัวแปรอิสระมีความสัมพันธ์ระดับปานกลาง (ρ_{12} = 0.40, ρ_{45} = 0.50, ρ_{67} = 0.60) ซึ่งผลการวิจัยส่วนนี้จะนำเสนอในตารางที่ 4.3.2 และกราฟรูปที่ 4.3.2

4.3.3 กรณีตัวแปรอิสระมีความสัมพันธ์ระดับสูง (ρ_{12} = 0.70, ρ_{45} = 0.80, ρ_{67} = 0.90) ซึ่งผลการวิจัยส่วนนี้จะนำเสนอในตารางที่ 4.3.3 และกราฟรูปที่ 4.3.3

นอกจากนี้ผู้วิจัยยังได้เสนอผลการวิจัยในกราฟรูปที่ 4.3.4 - 4.3.9 โดยกราฟนี้จะแสดงค่าเฉลี่ยของ MAPE เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป ณ ขนาดตัวอย่างแต่ละระดับ

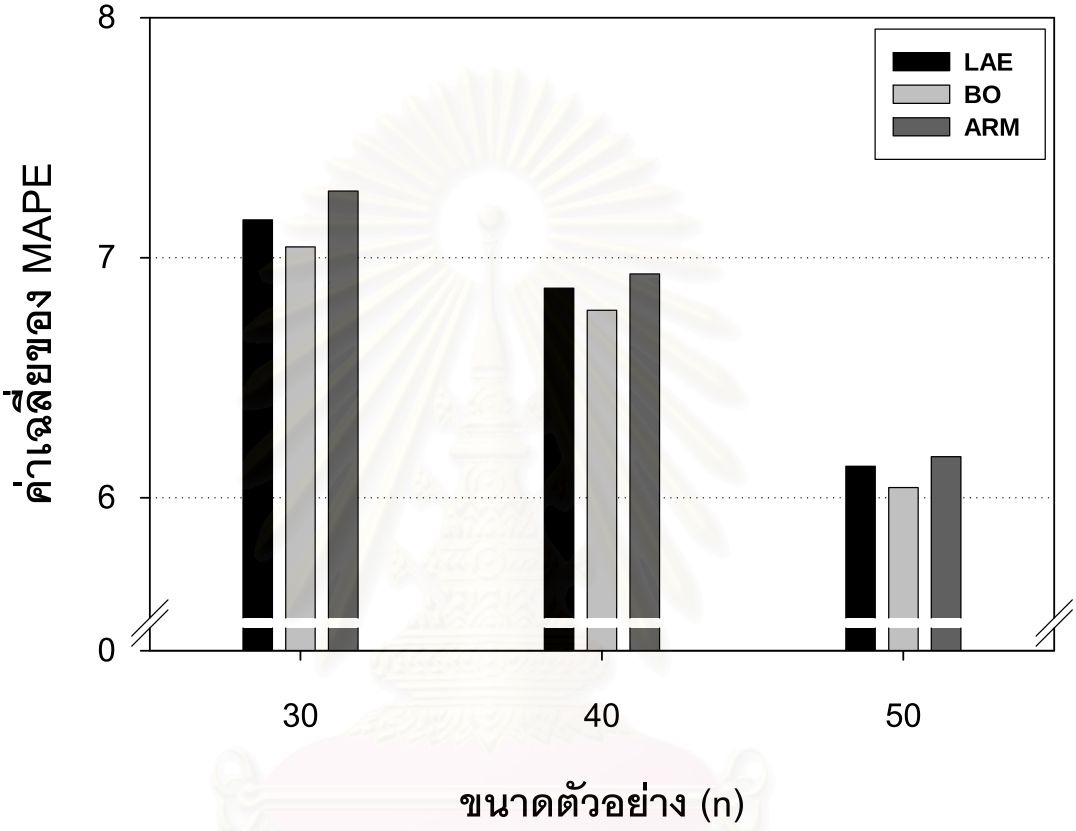
ตารางที่ 4.3.1 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) ในกรณีที่จำนวนตัวแปรอิสระ = 7 และตัวแปรอิสระมีความสัมพันธ์กันในระดับต่ำ

(ตัวเลขในวงเล็บคือ ส่วนเบี่ยงเบนมาตรฐาน (S.D.) ของ MAPE)

ระดับความสัมพันธ์ของตัวแปรอิสระ	ขนาดตัวอย่าง (n)	ค่าเฉลี่ยของ MAPE		
		LAE	BO	ARM
0.30, 0.30, 0.30	30	7.157643	7.045124*	7.278138
		(1.251232)	(1.280451)	(1.315318)
	40	6.873255	6.781580*	6.932578
		(0.966239)	(0.956389)	(0.976419)
	50	6.131056	6.042598*	6.171158
		(1.051155)	(1.044790)	(1.058952)

* หมายถึง วิธีหาค่าพยากรณ์ร่วมที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด

รูปที่ 4.3.1 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) กรณีที่ จำนวนตัวแปรอิสระ = 7 และตัวแปรอิสระมีความสัมพันธ์กันในระดับต่ำ

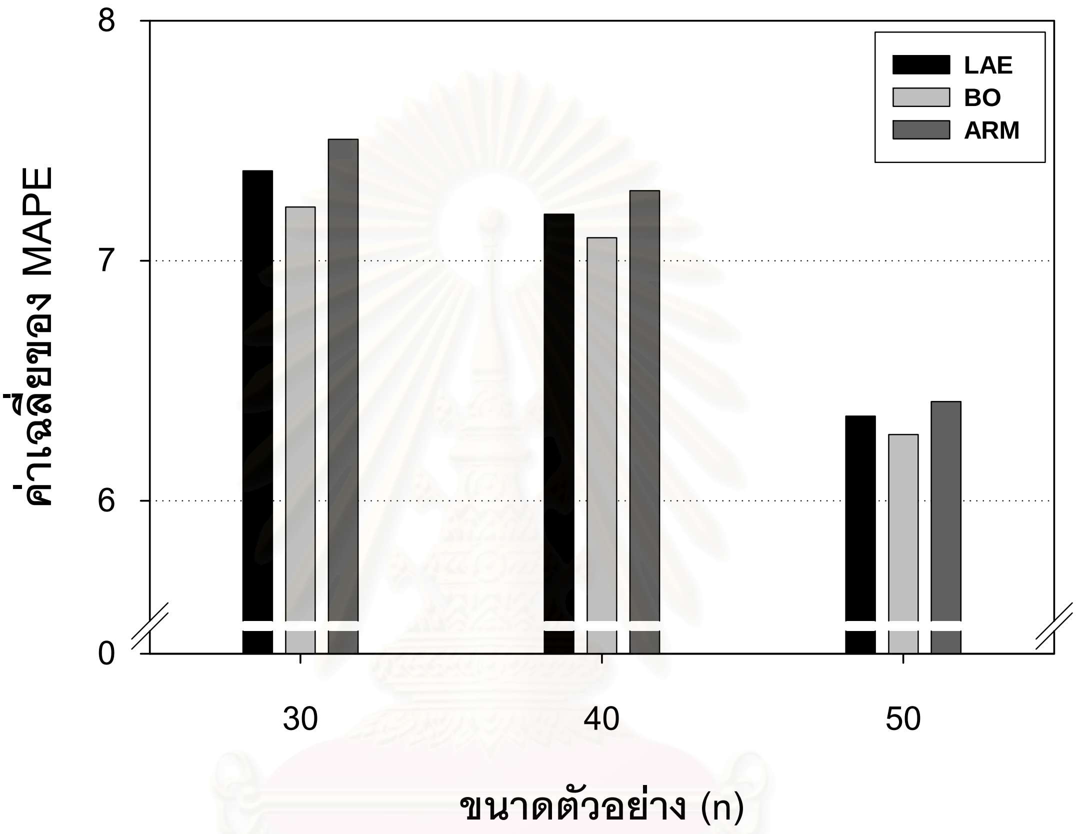


ตารางที่ 4.3.2 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) ในกรณีที่จำนวนตัวแปรอิสระ = 7 และตัวแปรอิสระมีความสัมพันธ์กันในระดับปานกลาง
(ตัวเลขในวงเล็บคือ ส่วนเบี่ยงเบนมาตรฐาน (S.D.) ของ MAPE)

ระดับความสัมพันธ์ของตัวแปรอิสระ	ขนาดตัวอย่าง (n)	ค่าเฉลี่ยของ MAPE		
		LAE	BO	ARM
0.40, 0.50, 0.60	30	7.374273	7.223721*	7.505656
		(1.394759)	(1.378781)	(1.418870)
	40	7.193154	7.095619*	7.291589
		(1.113510)	(1.094211)	(1.124650)
	50	6.352864	6.275786*	6.413159
		(0.791848)	(0.781913)	(0.801416)

* หมายถึง วิธีหาค่าพยากรณ์ร่วมที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด

รูปที่ 4.3.2 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) กรณีที่ จำนวนตัวแปรอิสระ = 7 และตัวแปรอิสระมีความสัมพันธ์กันในระดับปานกลาง



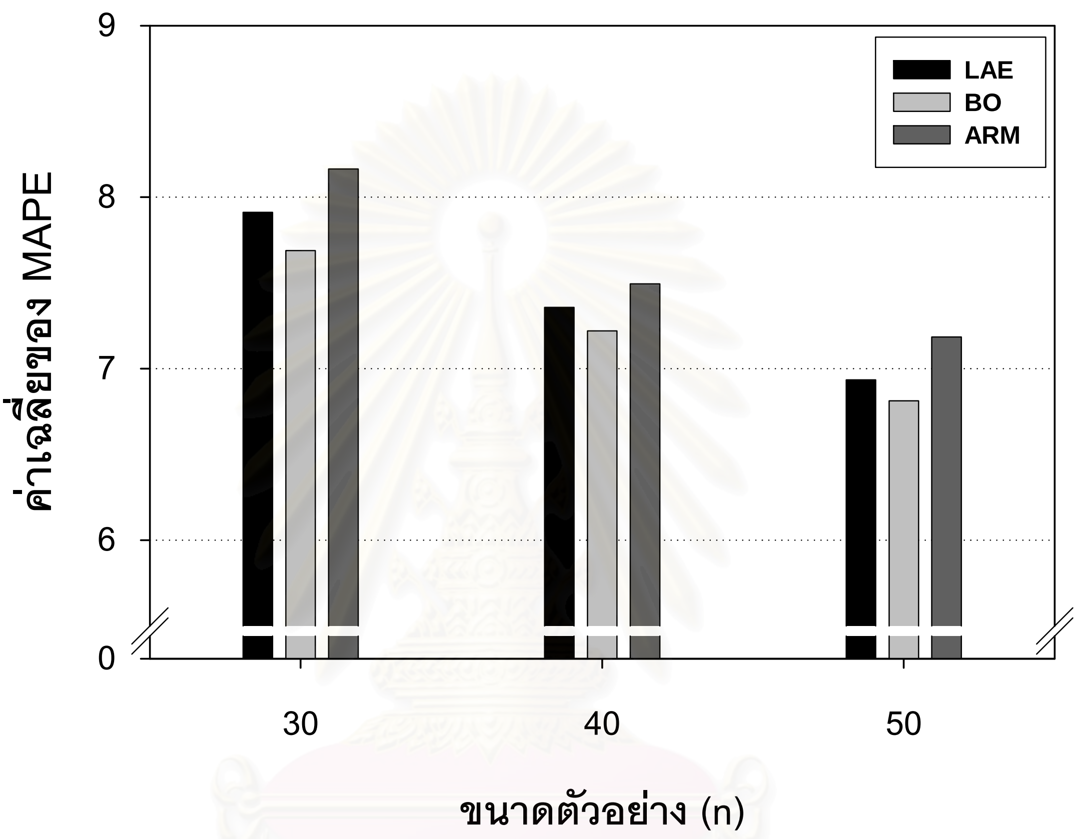
ตารางที่ 4.3.3 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) ในกรณีที่จำนวนตัวแปรอิสระ = **7** และตัวแปรอิสระมีความสัมพันธ์กันในระดับสูง

(ตัวเลขในวงเล็บคือ ส่วนเบี่ยงเบนมาตรฐาน (S.D.) ของ MAPE)

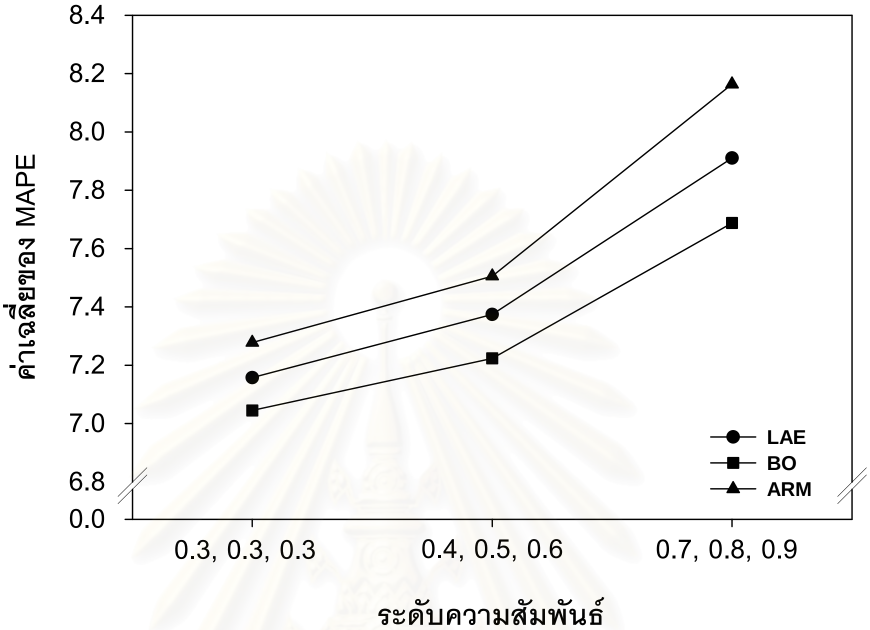
ระดับความสัมพันธ์ของตัวแปรอิสระ	ขนาดตัวอย่าง (n)	ค่าเฉลี่ยของ MAPE		
		LAE	BO	ARM
0.70, 0.80, 0.90	30	7.910403 (1.655110)	7.687837* (1.618479)	8.163797 (1.713122)
	40	7.357194 (1.397019)	7.219970* (1.356602)	7.494525 (1.411415)
	50	6.933598 (0.882729)	6.811594* (0.875247)	7.183651 (0.899264)

* หมายถึง วิธีหาค่าพยากรณ์ร่วมที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด

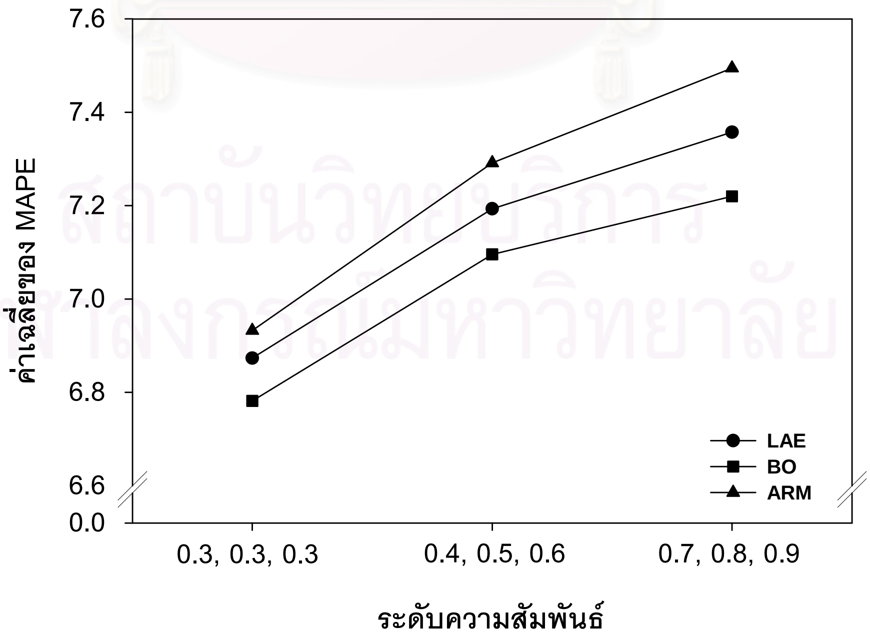
ตารางที่ 4.3.3 การเปรียบเทียบวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยใช้เกณฑ์ร้อยละของความคลาดเคลื่อนเฉลี่ยสัมบูรณ์ (MAPE) กรณีที่ จำนวนตัวแปรอิสระ = 7 และตัวแปรอิสระมีความสัมพันธ์กันในระดับสูง



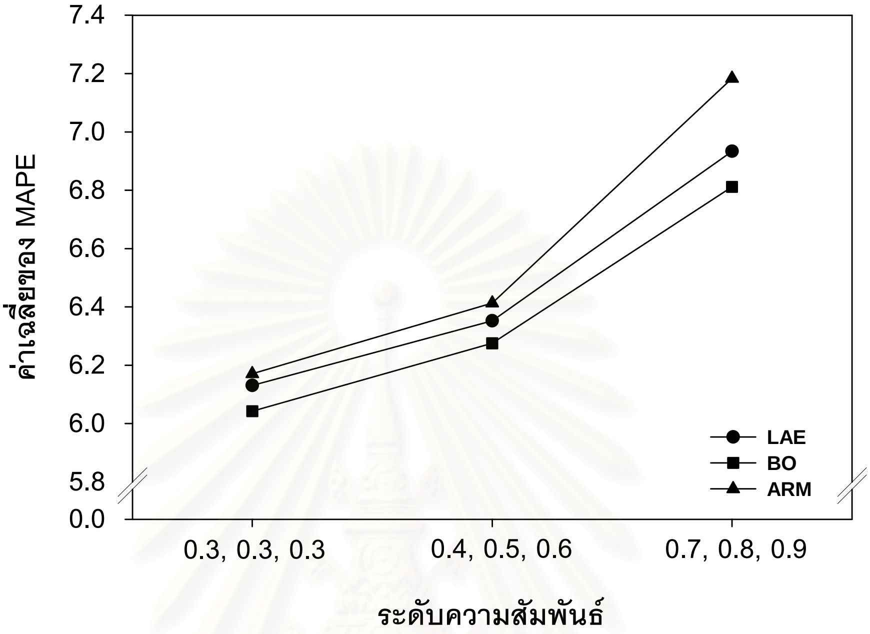
รูปที่ 4.3.4 แสดงการเปรียบเทียบค่า MAPE ของวิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป กรณีจำนวนตัวแปรอิสระ = 7 และขนาดตัวอย่าง = 30



รูปที่ 4.3.5 แสดงการเปรียบเทียบค่า MAPE ของวิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป กรณีจำนวนตัวแปรอิสระ = 7 และขนาดตัวอย่าง = 40



รูปที่ 4.3.6 แสดงการเปรียบเทียบค่า MAPE ของวิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) เมื่อระดับพหุสัมพันธ์เปลี่ยนแปลงไป กรณีจำนวนตัวแปรอิสระ = 7 และขนาดตัวอย่าง = 50



จากตารางและรูปที่ 4.3.1 - 4.3.3 เราสามารถสรุปผลการเปรียบเทียบค่าเฉลี่ยของ MAPE ของวิธี LAE วิธี BO และวิธี ARM เมื่อจำนวนตัวแปรอิสระเท่ากับ 7 จำแนกตามระดับพหุสัมพันธ์ได้ดังนี้

1. ความสัมพันธ์ระดับต่ำ

สำหรับความสัมพันธ์ระดับต่ำ (ρ_{12} = 0.30, ρ_{45} = 0.30, ρ_{67} = 0.30) พบว่าวิธี BO ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับ สำหรับทุกขนาดตัวอย่าง

2. ความสัมพันธ์ระดับปานกลาง

สำหรับความสัมพันธ์ระดับปานกลาง (ρ_{12} = 0.40, ρ_{45} = 0.50, ρ_{67} = 0.60) พบว่าวิธี BO ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับสำหรับทุกขนาดตัวอย่าง

3. ความสัมพันธ์ระดับสูง

สำหรับความสัมพันธ์ระดับสูง (ρ_{12} = 0.70, ρ_{45} = 0.80, ρ_{67} = 0.90) พบว่าวิธี BO ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับ สำหรับทุกขนาดตัวอย่าง

ข้อสรุป

ข้อสรุปสำหรับกรณีจำนวนตัวแปรอิสระเท่ากับ 7 (หัวข้อ 4.3) เป็นดังนี้

สำหรับทุกระดับความสัมพันธ์และขนาดตัวอย่าง พบว่าวิธี BO จะเป็นวิธีที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับ และจะสังเกตได้ว่าค่าเฉลี่ยของ MAPE มีแนวโน้มเพิ่มขึ้นเมื่อระดับความสัมพันธ์ของตัวแปรอิสระเพิ่มขึ้น ทั้งนี้เพราะเมื่อระดับความสัมพันธ์สูงขึ้นจะเกิดปัญหาพหุสัมพันธ์รุนแรงขึ้น ทำให้ตัวแบบแต่ละตัวแบบที่นำมาหาค่าพยากรณ์ร่วมมีความแปรปรวนของตัวประมาณสูงขึ้น ดังนั้น เมื่อนำค่าพยากรณ์จากแต่ละตัวแบบมาวิเคราะห์ร่วมกัน ย่อมทำให้ค่าเฉลี่ยของ MAPE ของการพยากรณ์ร่วมมีค่าสูงขึ้น นอกจากนี้ยังพบว่าค่าเฉลี่ยของ MAPE ของทุกวิธีจะมีแนวโน้มลดลงเมื่อขนาดตัวอย่างเพิ่มขึ้น ทั้งนี้เพราะเมื่อขนาดตัวอย่างเพิ่มขึ้น จะส่งผลให้ความแปรปรวนลดลง ทำให้สามารถแก้ปัญหาพหุสัมพันธ์ระหว่างตัวแปรอิสระลงได้บ้าง

4.4 ข้อสรุป

ผลสรุปโดยรวมของทุกกรณีเป็นดังนี้

จากผลการทดลองพบว่า วิธี BO เป็นวิธีที่ให้ค่าเฉลี่ยของ MAPE ต่ำสุด รองลงมาคือวิธี LAE และ ARM ตามลำดับ ทั้งนี้อาจเนื่องมาจากวิธี BO อาศัยการสุ่มตัวอย่างซ้ำ ๆ เสมือนสุ่มตัวอย่างมาจากประชากร ทำให้ความแปรปรวนลดลง

จากผลการวิจัย การเปลี่ยนแปลงของค่าเฉลี่ยของ MAPE มีลักษณะดังนี้

1. ค่าเฉลี่ยของ MAPE มีแนวโน้มเพิ่มขึ้นเมื่อระดับความสัมพันธ์ระหว่างตัวแปรอิสระสูงขึ้น สาเหตุที่เป็นเช่นนี้เพราะเมื่อเกิดปัญหาพหุสัมพันธ์รุนแรงขึ้น ทำให้ตัวแบบแต่ละตัวแบบที่นำมาหาค่าพยากรณ์ร่วมมีความแปรปรวนของตัวประมาณสูงขึ้น ดังนั้น เมื่อนำค่าพยากรณ์จากแต่ละตัวแบบมาวิเคราะห์ร่วมกัน ย่อมทำให้ค่าเฉลี่ยของ MAPE ของการพยากรณ์ร่วมสูงขึ้น

2. ค่าเฉลี่ยของ MAPE มีแนวโน้มลดลงเมื่อขนาดตัวอย่างใหญ่ขึ้น สาเหตุที่เป็นเช่นนี้เนื่องจากตัวแบบแต่ละตัวแบบที่นำมาหาค่าพยากรณ์ร่วมมีค่าสัมประสิทธิ์การถดถอยถูกต้องมากขึ้นเมื่อขนาดตัวอย่างใหญ่ขึ้น และในกรณีที่ตัวอย่างมีขนาดใหญ่มากจะสามารถแก้ปัญหาพหุสัมพันธ์ระหว่างตัวแปรอิสระได้มาก ดังนั้น เมื่อนำค่าพยากรณ์จากแต่ละตัวแบบมาวิเคราะห์ร่วมกัน ย่อมทำให้ค่าเฉลี่ยของ MAPE ของการพยากรณ์ร่วมมีค่าลดลง

บทที่ 5

สรุปผลการวิจัย และข้อเสนอแนะ

การวิจัยครั้งนี้เป็นการวิจัยเชิงทดลองเพื่อศึกษาและเปรียบเทียบประสิทธิภาพของวิธีหาค่าพยากรณ์ร่วมของตัวแบบการถดถอยเชิงเส้นพหุคูณ โดยผู้วิจัยได้ทำการเปรียบเทียบวิธีการ 3 วิธี ได้แก่ วิธีค่าสัมบูรณ์ต่ำสุด (LAE) วิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM) โดยสถานการณ์ที่ศึกษามีดังนี้

1. จำนวนตัวแปรอิสระที่ใช้ในการวิจัยมี 3 ระดับ คือ 3, 5 และ 7 ตัว
2. กำหนดขนาดตัวอย่างที่ใช้ในการวิจัย ดังนี้
   - กรณีที่มีตัวแปรอิสระ 3 ตัว ศึกษาเมื่อขนาดตัวอย่าง = 14, 20, 30, 40 และ 50
   - กรณีที่มีตัวแปรอิสระ 5 ตัว ศึกษาเมื่อขนาดตัวอย่าง = 20, 30, 40 และ 50
   - กรณีที่มีตัวแปรอิสระ 7 ตัว ศึกษาเมื่อขนาดตัวอย่าง = 30, 40 และ 50
3. กำหนดระดับความสัมพันธ์ระหว่างตัวแปรอิสระดังนี้
   - ระดับความสัมพันธ์ในกรณีจำนวนตัวแปรอิสระเท่ากับ 3 คือ 0.3, 0.5 และ 0.8
   - ระดับความสัมพันธ์ในกรณีจำนวนตัวแปรอิสระเท่ากับ 5 คือ (0.3, 0.3), (0.4, 0.6) และ (0.7, 0.9)
   - ระดับความสัมพันธ์ในกรณีจำนวนตัวแปรอิสระเท่ากับ 7 คือ (0.3, 0.3, 0.3), (0.4, 0.5, 0.6) และ (0.7, 0.8, 0.9)

การวิจัยครั้งนี้ผู้วิจัยศึกษาวิเคราะห์ด้วยเทคนิคการจำลองแบบมอนติคาร์โล โดยใช้โปรแกรมภาษาฟอร์แทรน 90 บนเครื่องคอมพิวเตอร์ PC เพื่อสร้างข้อมูลตามสถานการณ์ที่กำหนด โดยกระทำซ้ำ 1000 รอบ ในแต่ละสถานการณ์

5.1 สรุปผลการวิจัย

การวิจัยครั้งนี้ ผู้วิจัยต้องการเปรียบเทียบวิธีหาค่าพยากรณ์ร่วมโดยพิจารณาจากค่าเฉลี่ยของ MAPE ของแต่ละวิธี ซึ่งจากผลการวิจัยพบว่า ขนาดตัวอย่างและระดับความสัมพันธ์ระหว่างตัวแปรอิสระต่างก็มีผลต่อค่าเฉลี่ยของ MAPE ของวิธีหาค่าพยากรณ์ร่วมทั้ง 3 วิธี โดยค่าเฉลี่ยของ MAPE มีแนวโน้มเพิ่มขึ้นเมื่อระดับความสัมพันธ์ระหว่างตัวแปรอิสระเพิ่มขึ้น และมีแนวโน้มลดลงเมื่อขนาดตัวอย่างมากขึ้น ผู้วิจัยสรุปผลการวิจัยโดยแบ่งออกเป็น 2 ส่วนดังนี้

5.1.1 ผลการเปรียบเทียบค่าเฉลี่ยของ MAPE ของการหาค่าพยากรณ์ร่วมแต่ละวิธี

จากการเปรียบเทียบค่าเฉลี่ยของ MAPE ของทั้ง 3 วิธี พบว่า ค่าเฉลี่ยของ MAPE ของแต่ละวิธีเรียงลำดับจากน้อยไปมาก ได้แก่วิธี BO, LAE และ ARM ตามลำดับ ในทุกกรณีที่ศึกษา แสดงว่าวิธี BO ให้ค่าพยากรณ์ที่มีความถูกต้องมากที่สุด จึงสรุปได้ว่าวิธี BO มีประสิทธิภาพมากที่สุด รองลงมาคือวิธี LAE และวิธี ARM ตามลำดับ นอกจากนี้ยังมีข้อสังเกตว่า เมื่อขนาดตัวอย่างใหญ่ขึ้นเรื่อยๆ ค่าเฉลี่ยของ MAPE ของทั้ง 3 วิธีจะแตกต่างกันน้อยลงเรื่อยๆ ทั้งนี้เนื่องมาจากขนาดตัวอย่างที่เพิ่มขึ้นได้ช่วยแก้ปัญหาพหุสัมพันธ์ระหว่างตัวแปรอิสระ ทำให้การสร้างตัวแบบจาก 4 วิธีให้ตัวแบบออกมาเหมือนกันหรือใกล้เคียงกันมากขึ้นในหลายๆ รอบของการทดลอง จึงทำให้ค่าเฉลี่ยของ MAPE ของการหาค่าพยากรณ์ร่วมทั้ง 3 วิธีมีค่าใกล้เคียงกันมากขึ้น

5.1.2 ปัจจัยที่มีผลต่อค่าเฉลี่ยของ MAPE ของการหาค่าพยากรณ์ร่วมแต่ละวิธี

ปัจจัยที่มีผลต่อค่าเฉลี่ยของ MAPE มีดังนี้

1. ระดับความสัมพันธ์ระหว่างตัวแปรอิสระ

เมื่อระดับความสัมพันธ์ระหว่างตัวแปรอิสระสูงขึ้น ค่าเฉลี่ยของ MAPE ของทุกวิธีมีแนวโน้มเพิ่มขึ้น ทั้งนี้เพราะเมื่อเกิดปัญหาพหุสัมพันธ์รุนแรงขึ้น ทำให้ตัวแบบแต่ละตัวแบบที่นำมาหาค่าพยากรณ์ร่วมมีความแปรปรวนของตัวประมาณสูงขึ้น ดังนั้น เมื่อนำค่าพยากรณ์จากแต่ละตัวแบบมาวิเคราะห์ร่วมกัน ย่อมทำให้ค่าเฉลี่ยของ MAPE ของการพยากรณ์ร่วมมีค่าสูงขึ้น

2. ขนาดตัวอย่าง

เมื่อขนาดตัวอย่างเพิ่มขึ้น ค่าเฉลี่ยของ MAPE ของทุกวิธีมีแนวโน้มลดลง เพราะขนาดตัวอย่างที่เพิ่มขึ้นจะช่วยลดความแปรปรวนของการประมาณค่าลงได้

5.1.3 สรุปน้ำหนักของการพยากรณ์ร่วมที่ได้จากวิธี BO

เนื่องจากวิธี BO เป็นวิธีการหาน้ำหนักของการพยากรณ์ร่วมที่มีประสิทธิภาพมากที่สุด ผู้วิจัยจึงได้ทำการศึกษาเพิ่มเติมว่า ส่วนใหญ่วิธี BO จะให้น้ำหนักแก่วิธีพยากรณ์เดี่ยววิธีใดมากที่สุดภายใต้สถานการณ์ต่างๆ และพบว่าโดยทั่วไปการให้น้ำหนักจะขึ้นอยู่กับระดับพหุสัมพันธ์ของตัวแปรอิสระ ดังแสดงในตารางต่อไปนี้

ระดับพหุสัมพันธ์	วิธีพยากรณ์เดี่ยวที่ได้รับน้ำหนักมากที่สุด
ต่ำ	วิธีพิจารณาการถดถอยทุกรูปแบบ
ปานกลาง	วิธีกำจัดตัวแปรแบบถอยหลัง
สูง	วิธีการถดถอยขั้นบันได

5.2 ข้อเสนอแนะ

ผู้วิจัยเสนอแนะแนวทางในการเลือกวิธีหาค่าพยากรณ์ร่วมสำหรับตัวแบบการถดถอยเชิงเส้นพหุคูณ โดยแบ่งเป็น 2 ด้าน ดังนี้

5.2.1 ด้านการนำไปใช้ประโยชน์

ในการวิเคราะห์การถดถอยเชิงเส้นพหุคูณโดยทั่วไป หากมีตัวแบบในข่ายที่พิจารณามากกว่า 1 ตัวแบบ และจุดมุ่งหมายของการวิเคราะห์คือการพยากรณ์ค่าของตัวแปรตามให้ได้ถูกต้องและแม่นยำที่สุด ผู้วิเคราะห์อาจนำค่าพยากรณ์จากแต่ละตัวแบบมาหาค่าพยากรณ์ร่วม ซึ่งการวิจัยครั้งนี้ได้เสนอวิธีหาค่าพยากรณ์ร่วม 3 วิธี ได้แก่ วิธี LAE วิธี BO และวิธี ARM จากผลการทดลองพบว่า วิธี BO ให้ค่าเฉลี่ยของ MAPE ต่ำสุดในทุกกรณีที่ศึกษา ดังนั้น ผู้วิจัยจึงเสนอให้ใช้วิธี BO ในการหาค่าพยากรณ์ร่วม

นอกจากนี้ ผู้วิจัยเสนออีกหนทางหนึ่งที่จะนำผลการศึกษาครั้งนี้ไปใช้ประโยชน์ได้ง่ายขึ้น โดยไม่ต้องใช้วิธีถ่วงน้ำหนักซึ่งค่อนข้างยุ่งยาก โดยจะใช้ผลจากตารางในหัวข้อ 5.1.3 เป็นแนวทางในการเลือกวิธีพยากรณ์เดี่ยว กล่าวคือ ส่วนใหญ่วิธี BO ให้น้ำหนักแก่วิธีพยากรณ์เดี่ยววิธีใดมากที่สุด ก็จะเลือกใช้วิธีพยากรณ์เดี่ยววิธีนั้น ผู้วิจัยจึงสรุปแนวทางในการเลือกวิธีพยากรณ์เดี่ยวได้ดังนี้

- กรณีที่พหุสัมพันธ์ระหว่างตัวแปรอิสระอยู่ในระดับต่ำ ควรเลือกใช้วิธีพิจารณาการถดถอยทุกรูปแบบ
- กรณีที่พหุสัมพันธ์ระหว่างตัวแปรอิสระอยู่ในระดับปานกลาง ควรเลือกใช้วิธีกำจัดตัวแปรแบบถอยหลัง
- กรณีที่พหุสัมพันธ์ระหว่างตัวแปรอิสระอยู่ในระดับสูง ควรเลือกใช้วิธีการถดถอยขั้นบันได

5.2.2 ด้านการศึกษาวิจัย

เพื่อเป็นแนวทางให้ผู้ที่สนใจได้ศึกษาเพิ่มเติม และเป็นการขยายผลการวิจัยออกไปให้เกิดประโยชน์มากยิ่งขึ้น ผู้วิจัยมีข้อเสนอแนะดังต่อไปนี้

1. ผู้สนใจอาจทำการศึกษาในกรณีที่ตัวแบบในกลุ่มที่พิจารณามีรูปแบบอื่นๆ เช่น ตัวแบบการถดถอยพหุนาม เป็นต้น
2. ผู้สนใจอาจทำการศึกษาและเปรียบเทียบวิธีหาค่าพยากรณ์ร่วมวิธีอื่นๆ นอกเหนือจากวิธีที่เสนอในวิทยานิพนธ์ฉบับนี้ด้วย
3. ในการวิจัยครั้งนี้พบว่าวิธี ARM มีประสิทธิภาพต่ำสุดในทุกกรณีที่ศึกษา ทั้งนี้อาจเนื่องมาจากสถานการณ์ที่กำหนดขึ้นไม่เหมาะสมที่จะใช้วิธี ARM ผู้สนใจอาจทดลองทำการศึกษาในสถานการณ์อื่นๆ เช่น กรณีที่ตัวแปรอิสระมีจำนวนมาก (มากกว่า 7 ตัวขึ้นไป) และกรณีที่ตัวแบบในกลุ่มที่พิจารณามีจำนวนมากขึ้น (มากกว่า 4 ตัวแบบขึ้นไป) เป็นต้น

รายการอ้างอิง

ภาษาไทย

มานพ วราภักดิ์. **การจำลอง (Simulation).** กรุงเทพมหานคร: สำนักพิมพ์แห่งจุฬาลงกรณ์มหาวิทยาลัย, 2550.

อมรรัตน์ ปรารมภ์. **การหาค่าพยากรณ์ร่วมโดยการให้ค่าเฉลี่ยถ่วงน้ำหนัก**. วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาสถิติ บัณฑิตวิทยาลัย จุฬาลงกรณ์มหาวิทยาลัย, 2539.

ภาษาต่างประเทศ

Breimen, L. Stacked Regressions. **Machine Learning** 24 (July 1996): 49-64.

Clarke, B. Comparing Bayes Model Averaging and Stacking When Model Approximation Error Cannot Be Ignored. **Journal of Machine Learning Research** 4 (December 2003): 683-712.

Draper, N.R. and Smith, H. **Applied Regression Analysis**. 3rd Edition. New York: Wiley-Interscience, 1998.

LeBlanc, M. and Tibshirani, R. Combining Estimates in Regression and Classification. **Journal of the American Statistical Association** 91 (December 1996): 1641-1650.

Yang, Y. Adaptive regression by mixing. **Journal of American Statistical Association** 96 (June 2001): 574-588.

Yang, Y. Regression with multiple candidate models: selecting or mixing? **Statistica Sinica** 13 (July 2003): 783-809.

Yuan, Z. and Yang, Y. Combining linear regression models: When and how? **Journal of American Statistical Association** 100 (December 2005): 1202-1214.

ภาคผนวก

ภาคผนวก ก.

ตัวอย่างการหาน้ำหนักของการพยากรณ์ร่วมโดยวิธีบูตสแตรป (BO) และวิธี adaptive regression by mixing (ARM)

ในที่นี้จะยกตัวอย่างการหาน้ำหนักโดยวิธี BO และวิธี ARM ในกรณีที่ขนาดตัวอย่าง (n) เท่ากับ 14 จำนวนตัวแปรอิสระเท่ากับ 3 และสัมประสิทธิ์สหสัมพันธ์ระหว่าง x_1 และ x_2 (ρ_{12}) เท่ากับ 0.5 ซึ่งข้อมูลที่ใช้ในการวิเคราะห์แสดงในตารางต่อไปนี้

ค่าสังเกตที่	x_1	x_2	x_3	y
1	2.73	5.19	6.97	45.85
2	3.50	3.18	7.64	44.55
3	2.78	3.42	8.14	46.42
4	3.02	4.59	8.79	52.45
5	2.95	4.50	-5.56	26.42
6	2.53	4.87	4.36	40.26
7	2.30	4.85	2.25	51.24
8	2.16	3.11	0.61	28.10
9	0.83	1.91	3.92	28.55
10	2.13	5.15	-2.90	26.14
11	2.18	2.24	9.34	54.10
12	3.01	3.26	5.69	32.13
13	2.38	3.40	0.27	37.26
14	1.28	2.85	5.16	36.34

จะเรียกข้อมูลในตารางข้างต้นว่าข้อมูล **Z** กล่าวคือ $\mathbf{Z} = \{(\mathbf{x}_i, y_i) \mid i = 1, \ldots, 14\}$ เมื่อ $\mathbf{x}_i = (x_{i1}, x_{i2}, x_{i3})$

เมื่อนำข้อมูล **Z** มาสร้างตัวแบบ 4 วิธี ได้แก่วิธีพิจารณาการถดถอยทุกรูปแบบ (APR) โดยใช้เกณฑ์ MAPE ต่ำสุด วิธีคัดเลือกตัวแปรแบบไปข้างหน้า (Forward) วิธีกำจัดตัวแปรแบบถอยหลัง (Backward) และวิธีการถดถอยขั้นบันได (Stepwise) ได้ผลดังนี้

APR $\quad \hat{y}^{(1)} = 25.58 + 3.10x_1 + 1.58x_3 \quad = f_{\mathbf{Z}}^{(1)}(\mathbf{x})$

Forward $\quad \hat{y}^{(2)} = 32.80 + 1.65x_3 \quad = f_{\mathbf{Z}}^{(2)}(\mathbf{x})$

Backward $\quad \hat{y}^{(3)} = 19.62 + 3.25x_2 + 1.90x_3 \quad = f_{\mathbf{Z}}^{(3)}(\mathbf{x})$

Stepwise $\quad$ ได้สมการเดียวกับ Forward

ดังนั้น จะได้เวกเตอร์ของฟังก์ชันการถดถอยของ 3 ตัวแบบ ดังนี้

$$\mathbf{f_z}(\mathbf{x}) = \begin{bmatrix} 25.58 + 3.10x_1 + 1.58x_3 \\ 32.80 + 1.65x_3 \\ 19.62 + 3.25x_2 + 1.90x_3 \end{bmatrix}$$

ในการคำนวณจะกำหนดสัญลักษณ์ต่าง ๆ ดังต่อไปนี้

กำหนดให้ **y** คือเวกเตอร์ของข้อมูลตัวแปรตาม กล่าวคือ

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{14} \end{bmatrix} = \begin{bmatrix} 45.85 \\ 44.55 \\ \vdots \\ 36.34 \end{bmatrix}$$

กำหนดให้ **X** คือเมทริกซ์ของข้อมูลตัวแปรอิสระ กล่าวคือ

$$\mathbf{X} = \begin{bmatrix} 1 & x_{1,1} & x_{1,2} & x_{1,3} \\ 1 & x_{2,1} & x_{2,2} & x_{2,3} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{14,1} & x_{14,2} & x_{14,3} \end{bmatrix} = \begin{bmatrix} 1 & 2.73 & 5.19 & 6.97 \\ 1 & 3.50 & 3.18 & 7.64 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1.28 & 2.85 & 5.16 \end{bmatrix}$$

สำหรับตัวแบบที่ 1 จะได้

$$\mathbf{b_z}^{(1)} = \begin{bmatrix} b_0 \\ b_1 \\ b_3 \end{bmatrix} = \begin{bmatrix} 25.58 \\ 3.10 \\ 1.58 \end{bmatrix}$$ ซึ่งจะกำหนดให้คู่กับเมทริกซ์

$$\mathbf{X}^{(1)} = \begin{bmatrix} 1 & x_{1,1} & x_{1,3} \\ 1 & x_{2,1} & x_{2,3} \\ \vdots & \vdots & \vdots \\ 1 & x_{14,1} & x_{14,3} \end{bmatrix} = \begin{bmatrix} 1 & 2.73 & 6.97 \\ 1 & 3.50 & 7.64 \\ \vdots & \vdots & \vdots \\ 1 & 1.28 & 5.16 \end{bmatrix}$$

สำหรับตัวแบบที่ 2 จะได้

$$\mathbf{b}_{\mathbf{Z}}^{(2)} = \begin{bmatrix} b_0 \\ b_3 \end{bmatrix} = \begin{bmatrix} 32.80 \\ 1.65 \end{bmatrix}$$ ซึ่งจะกำหนดให้คู่กับเมทริกซ์

$$\mathbf{X}^{(2)} = \begin{bmatrix} 1 & x_{1,3} \\ 1 & x_{2,3} \\ \vdots & \vdots \\ 1 & x_{14,3} \end{bmatrix} = \begin{bmatrix} 1 & 6.97 \\ 1 & 7.64 \\ \vdots & \vdots \\ 1 & 5.16 \end{bmatrix}$$

สำหรับตัวแบบที่ 3 จะได้

$$\mathbf{b}_{\mathbf{Z}}^{(3)} = \begin{bmatrix} b_0 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 19.62 \\ 13.25 \\ 1.90 \end{bmatrix}$$ ซึ่งจะกำหนดให้คู่กับเมทริกซ์

$$\mathbf{X}^{(3)} = \begin{bmatrix} 1 & x_{1,2} & x_{1,3} \\ 1 & x_{2,2} & x_{2,3} \\ \vdots & \vdots & \vdots \\ 1 & x_{14,2} & x_{14,3} \end{bmatrix} = \begin{bmatrix} 1 & 5.19 & 6.97 \\ 1 & 3.18 & 7.64 \\ \vdots & \vdots & \vdots \\ 1 & 2.85 & 5.16 \end{bmatrix}$$

ขั้นตอนการหาน้ำหนักโดยวิธี BO

1. สุ่มตัวอย่างโดยวิธีบูตสแตรปจากข้อมูล $\mathbf{Z}$ ออกมา 1000 ชุด โดยให้ชุดที่ j แทนด้วย $\mathbf{Z}^{*j} = \{(\mathbf{x}_i^{*j}, y_i^{*j}) | i = 1,...,14\},\ j = 1, ..., 1000$ ดังแสดงในตารางต่อไปนี้

$\mathbf{Z}^{*j}$	**แถวที่ถูกเลือก**			
$\mathbf{Z}^{*1}$	9, 10, 11, 12, 3, 11, 1, 6, 7, 14, 7, 5, 11, 10			
$\mathbf{Z}^{*2}$	13, 13, 10, 14, 9, 9, 9, 7, 13, 8, 5, 10, 1, 13			
$\mathbf{Z}^{*3}$	9, 4, 5, 1, 1, 4, 4, 4, 6, 13, 14, 12, 6, 7			
$\mathbf{Z}^{*4}$	9, 10, 4, 3, 6, 5, 6, 14, 14, 11, 2, 9, 14, 8			
$\mathbf{Z}^{*5}$	14, 4, 13, 2, 9 ,7 ,8, 7, 8, 7, 11, 2, 3, 2			
$\vdots$				
$\mathbf{Z}^{*1000}$				

2. นำตัวอย่างแต่ละชุดไปประมาณค่าสัมประสิทธิ์การถดถอยของแต่ละตัวแบบใหม่ ดังนี้

$\mathbf{Z}^{*j}$	แถวที่ถูกเลือก	$\mathbf{b}^{(1)}_{\mathbf{Z}^{*j}}$	$\mathbf{b}^{(2)}_{\mathbf{Z}^{*j}}$	$\mathbf{b}^{(3)}_{\mathbf{Z}^{*j}}$
$\mathbf{Z}^{*1}$	9, 10, 11, 12, 3, 11, 1, 6, 7, 14, 7, 5, 11, 10	$(26.15, 3.46, 1.76)^T$	$(34.06, 1.73)^T$	$(20.50, 3.05, 2.25)^T$
$\mathbf{Z}^{*2}$	13, 13, 10, 14, 9, 9, 9, 7, 13, 8, 5, 10, 1, 13	$(14.37, 8.72, 1.96)^T$	$(32.72, 1.01)^T$	$(18.20, 3.85, 1.61)^T$
$\mathbf{Z}^{*3}$	9, 4, 5, 1, 1, 4, 4, 4, 6, 13, 14, 12, 6, 7	$(22.91, 4.80, 1.48)^T$	$(32.24, 1.64)^T$	$(14.83, 4.84, 1.42)^T$
$\mathbf{Z}^{*4}$	9, 10, 4, 3, 6, 5, 6, 14, 14, 11, 2, 9, 14, 8	$(20.10, 4.70, 1.81)^T$	$(29.79, 1.85)^T$	$(16.91, 3.33, 2.18)^T$
$\mathbf{Z}^{*5}$	14, 4, 13, 2, 9 ,7 ,8, 7, 8, 7, 11, 2, 3, 2	$(29.25, 3.50, 1.04)^T$	$(36.00, 1.42)^T$	$(8.38, 7.27, 1.90)^T$
$\vdots$				
$\mathbf{Z}^{*1000}$				

การคำนวณสัมประสิทธิ์การถดถอยในตารางข้างต้น ใช้สูตรดังนี้

$$\mathbf{b}^{(1)}_{\mathbf{Z}^{*j}} = \left[(\mathbf{X}^{(1)*j})^T(\mathbf{X}^{(1)*j})\right]^{-1}(\mathbf{X}^{(1)*j})^T\mathbf{y}^{*j}$$

$$\mathbf{b}^{(2)}_{\mathbf{Z}^{*j}} = \left[(\mathbf{X}^{(2)*j})^T(\mathbf{X}^{(2)*j})\right]^{-1}(\mathbf{X}^{(2)*j})^T\mathbf{y}^{*j}$$

$$\mathbf{b}^{(3)}_{\mathbf{Z}^{*j}} = \left[(\mathbf{X}^{(3)*j})^T(\mathbf{X}^{(3)*j})\right]^{-1}(\mathbf{X}^{(3)*j})^T\mathbf{y}^{*j}$$

เมื่อ $\mathbf{X}^{(1)*j}, \mathbf{X}^{(2)*j}, \mathbf{X}^{(3)*j}$ คือ เมทริกซ์ของข้อมูลตัวแปรอิสระสำหรับตัวแบบที่ 1, 2 และ 3 ซึ่งได้จากการทำบูตสแตรปรอบที่ j

$\mathbf{y}^{*j}$ คือ เวกเตอร์ของข้อมูลตัวแปรตามซึ่งได้จากการทำบูตสแตรป รอบที่ j

ตัวอย่างการคำนวณหาเวกเตอร์ $\mathbf{b}^{(1)}_{\mathbf{Z}^{*1}}$, $\mathbf{b}^{(2)}_{\mathbf{Z}^{*1}}$ และ $\mathbf{b}^{(3)}_{\mathbf{Z}^{*1}}$ $(j=1)$

จากตารางข้างต้นจะได้ว่า

$$\mathbf{y}^{*1} = \begin{bmatrix} 28.55 \\ 26.14 \\ 54.10 \\ \vdots \\ 26.14 \end{bmatrix}_{14\times1}, \quad \mathbf{X}^{(1)*1} = \begin{bmatrix} 1 & 0.83 & 3.92 \\ 1 & 2.13 & -2.90 \\ 1 & 2.18 & 9.34 \\ \vdots & \vdots & \vdots \\ 1 & 2.13 & -2.90 \end{bmatrix}_{14\times3}, \quad \mathbf{X}^{(2)*1} = \begin{bmatrix} 1 & 3.92 \\ 1 & -2.90 \\ 1 & 9.34 \\ \vdots & \vdots \\ 1 & -2.90 \end{bmatrix}_{14\times2} \quad \text{และ}$$

$$\mathbf{X}^{(3)*1} = \begin{bmatrix} 1 & 1.91 & 3.92 \\ 1 & 5.15 & -2.90 \\ 1 & 2.24 & 9.34 \\ \vdots & \vdots & \vdots \\ 1 & 5.15 & -2.90 \end{bmatrix}_{14\times3}$$

ดังนั้น $\mathbf{b}^{(1)}_{\mathbf{Z}^{*j}} = \left[(\mathbf{X}^{(1)*j})^T(\mathbf{X}^{(1)*j})\right]^{-1}(\mathbf{X}^{(1)*j})^T\mathbf{y}^{*j} = \begin{bmatrix}26.15\\3.46\\1.76\end{bmatrix}$

$$\mathbf{b}^{(2)}_{\mathbf{Z}^{*j}} = \left[(\mathbf{X}^{(2)*j})^T(\mathbf{X}^{(2)*j})\right]^{-1}(\mathbf{X}^{(2)*j})^T\mathbf{y}^{*j} = \begin{bmatrix}34.06\\1.73\end{bmatrix}$$

$$\mathbf{b}^{(3)}_{\mathbf{Z}^{*j}} = \left[(\mathbf{X}^{(3)*j})^T(\mathbf{X}^{(3)*j})\right]^{-1}(\mathbf{X}^{(3)*j})^T\mathbf{y}^{*j} = \begin{bmatrix}20.50\\3.05\\2.25\end{bmatrix}$$

3. คำนวณค่าของฟังก์ชันการถดถอยที่ได้ใหม่ ณ $\mathbf{x}_i^{*j}$ และ $\mathbf{x}_i$ ตามลำดับ

$\mathbf{Z}^{*j}$	$\mathbf{f}_{\mathbf{Z}^{*j}}(\mathbf{x})$	i	$\mathbf{f}_{\mathbf{Z}^{*j}}(\mathbf{x}_i^{*j})$	$\mathbf{f}_{\mathbf{Z}^{*j}}(\mathbf{x}_i)$
$\mathbf{Z}^{*1}$	$\begin{bmatrix}26.15+3.46x_1+1.76x_3\\34.06+1.73x_3\\20.50+3.05x_2+2.25x_3\end{bmatrix}$	1	$\begin{bmatrix}35.93\\40.86\\35.18\end{bmatrix}$	$\begin{bmatrix}47.90\\46.16\\52.10\end{bmatrix}$
		2	$\begin{bmatrix}28.42\\29.03\\29.65\end{bmatrix}$	$\begin{bmatrix}51.75\\47.33\\47.47\end{bmatrix}$
		$\vdots$	$\vdots$	$\vdots$
		10	$\begin{bmatrix}28.42\\29.03\\29.65\end{bmatrix}$	$\begin{bmatrix}39.68\\43.01\\40.84\end{bmatrix}$
$\mathbf{Z}^{*2}$ $\vdots$ $\mathbf{Z}^{*1000}$				

4. คำนวณหา $\mathbf{\Delta}_1$ และ $\mathbf{\Delta}_2$ จากสูตรต่อไปนี้

$$\mathbf{\Delta}_1 = \frac{1}{1000}\sum_{j=1}^{1000}\left[\frac{1}{14}\sum_{i=1}^{14}\mathbf{f}_{\mathbf{Z}^{*j}}(\mathbf{x}_i)\mathbf{f}^T_{\mathbf{Z}^{*j}}(\mathbf{x}_i) - \frac{1}{14}\sum_{i=1}^{14}\mathbf{f}_{\mathbf{Z}^{*j}}\left(\mathbf{x}_i^{*j}\right)\mathbf{f}^T_{\mathbf{Z}^{*j}}\left(\mathbf{x}_i^{*j}\right)\right]$$

$$\mathbf{\Delta}_2 = \frac{1}{1000}\sum_{j=1}^{1000}\left[\frac{1}{14}\sum_{i=1}^{14}\mathbf{f}_{\mathbf{Z}^{*j}}(\mathbf{x}_i)y_i - \frac{1}{14}\sum_{i=1}^{14}\mathbf{f}_{\mathbf{Z}^{*j}}\left(\mathbf{x}_i^{*j}\right)y_i^{*j}\right]$$

5. คำนวณหาน้ำหนักจากสูตร

$$\mathbf{w}_{\mathrm{BO}} = \left[\frac{1}{14}\sum_{i=1}^{14}\mathbf{f}_{\mathbf{Z}}(\mathbf{x}_i)\mathbf{f}^T_{\mathbf{Z}}(\mathbf{x}_i) + \mathbf{\Delta}_1\right]^{-1}\left[\frac{1}{14}\sum_{i=1}^{14}\mathbf{f}_{\mathbf{Z}}(\mathbf{x}_i)y_i + \mathbf{\Delta}_2\right]$$

ซึ่งในที่นี้ได้

$$\mathbf{w}_{\mathrm{BO}} = \begin{bmatrix} 0.2614 \\ 0.1903 \\ 0.5483 \end{bmatrix}$$

ผังงานการหาน้ำหนักด้วยวิธี BO

ข้อมูลเริ่มต้น คือ $\mathbf{Z} = \{(\mathbf{x}_i, y_i) \mid i = 1, ..., n\}$
ซึ่งเขียนในรูปเมทริกซ์ได้ดังนี้

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1p} \\ 1 & x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}, \ \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

↓

สร้างตัวแบบจาก 4 วิธี

↓

ตัดตัวแบบที่ซ้ำออก สมมติว่าเหลือ m ตัวแบบ

↓

ตัวแบบที่ (1) มีสัมประสิทธิ์การถดถอยคือ $\begin{bmatrix} b_0^{(1)} & b_1^{(1)} & \cdots & b_p^{(1)} \end{bmatrix}^T$
ตัวแบบที่ (2) มีสัมประสิทธิ์การถดถอยคือ $\begin{bmatrix} b_0^{(2)} & b_1^{(2)} & \cdots & b_p^{(2)} \end{bmatrix}^T$
$\vdots$
ตัวแบบที่ (m) มีสัมประสิทธิ์การถดถอยคือ $\begin{bmatrix} b_0^{(m)} & b_1^{(m)} & \cdots & b_p^{(m)} \end{bmatrix}^T$

↓

สร้างเมทริกซ์ $\mathbf{X}^{(1)}$ ซึ่งเกิดจากการตัดคอลัมน์ของ $\mathbf{X}$ ที่ตรงกับตำแหน่งที่สัมประสิทธิ์การถดถอยของตัวแบบที่ (1) เป็น 0 ออก
สร้างเวกเตอร์ $\mathbf{b}_{\mathbf{Z}}^{(1)}$ ซึ่งเกิดจากการตัดแถวที่เป็น 0 ออกจากเวกเตอร์สัมประสิทธิ์การถดถอยของตัวแบบที่ (1)

สร้างเมทริกซ์ $\mathbf{X}^{(2)}$ ซึ่งเกิดจากการตัดคอลัมน์ของ $\mathbf{X}$ ที่ตรงกับตำแหน่งที่สัมประสิทธิ์การถดถอยของตัวแบบที่ (2) เป็น 0 ออก
สร้างเวกเตอร์ $\mathbf{b}_{\mathbf{Z}}^{(2)}$ ซึ่งเกิดจากการตัดแถวที่เป็น 0 ออกจากเวกเตอร์สัมประสิทธิ์การถดถอยของตัวแบบที่ (2)
$\vdots$
สร้างเมทริกซ์ $\mathbf{X}^{(m)}$ ซึ่งเกิดจากการตัดคอลัมน์ของ $\mathbf{X}$ ที่ตรงกับตำแหน่งที่สัมประสิทธิ์การถดถอยของตัวแบบที่ (m) เป็น 0 ออก
สร้างเวกเตอร์ $\mathbf{b}_{\mathbf{Z}}^{(m)}$ ซึ่งเกิดจากการตัดแถวที่เป็น 0 ออกจากเวกเตอร์สัมประสิทธิ์การถดถอยของตัวแบบที่ (m)

↓

1

1

DO 2
j =1,1000

สุ่มตัวอย่างโดยวิธีบูตสแตรปจากข้อมูล $\mathbf{Z}$ โดยให้ชุดที่ j แทนด้วย

$$\mathbf{Z}^{*j} = \left\{(\mathbf{x}_i^{*j}, y_i^{*j}) \middle| i = 1,\ldots,n\right\}$$

ซึ่งข้อมูลชุดที่ j ในรูปของเมทริกซ์ คือ $\mathbf{X}^{(1)*j}, \mathbf{X}^{(2)*j}, \ldots, \mathbf{X}^{(m)*j}$ และ $\mathbf{y}^{*j}$

นำข้อมูลชุดที่ j ไปประมาณค่าสัมประสิทธิ์การถดถอยของแต่ละตัวแบบใหม่ ดังนี้

$$\begin{aligned}
\mathbf{b}_{\mathbf{Z}^{*j}}^{(1)} &= \left[(\mathbf{X}^{(1)*j})^T(\mathbf{X}^{(1)*j})\right]^{-1}(\mathbf{X}^{(1)*j})^T\mathbf{y}^{*j} \\
\mathbf{b}_{\mathbf{Z}^{*j}}^{(2)} &= \left[(\mathbf{X}^{(2)*j})^T(\mathbf{X}^{(2)*j})\right]^{-1}(\mathbf{X}^{(2)*j})^T\mathbf{y}^{*j} \\
&\vdots \\
\mathbf{b}_{\mathbf{Z}^{*j}}^{(m)} &= \left[(\mathbf{X}^{(m)*j})^T(\mathbf{X}^{(m)*j})\right]^{-1}(\mathbf{X}^{(m)*j})^T\mathbf{y}^{*j}
\end{aligned}$$

คำนวณค่าพยากรณ์ของแต่ละตัวแบบใหม่ ณ $\mathbf{x}_i^{*j}$

$$\left.\begin{aligned}
\hat{\mathbf{y}}_A^{(1)*j} &= \mathbf{X}^{(1)*j}\mathbf{b}_{\mathbf{Z}^{*j}}^{(1)} \\
\hat{\mathbf{y}}_A^{(2)*j} &= \mathbf{X}^{(2)*j}\mathbf{b}_{\mathbf{Z}^{*j}}^{(2)} \\
&\vdots \\
\hat{\mathbf{y}}_A^{(m)*j} &= \mathbf{X}^{(m)*j}\mathbf{b}_{\mathbf{Z}^{*j}}^{(m)}
\end{aligned}\right\} \text{ จะได้ } \mathbf{f}_{\mathbf{Z}^{*j}}(\mathbf{x}_i^{*j}) = \begin{bmatrix} i\text{th row of } \hat{\mathbf{y}}_A^{(1)*j} \\ i\text{th row of } \hat{\mathbf{y}}_A^{(2)*j} \\ \vdots \\ i\text{th row of } \hat{\mathbf{y}}_A^{(m)*j} \end{bmatrix}, i = 1,\ldots,n$$

คำนวณค่าพยากรณ์ของแต่ละตัวแบบใหม่ ณ $\mathbf{x}_i$

$$\left.\begin{aligned}
\hat{\mathbf{y}}_B^{(1)*j} &= \mathbf{X}^{(1)}\mathbf{b}_{\mathbf{Z}^{*j}}^{(1)} \\
\hat{\mathbf{y}}_B^{(2)*j} &= \mathbf{X}^{(2)}\mathbf{b}_{\mathbf{Z}^{*j}}^{(2)} \\
&\vdots \\
\hat{\mathbf{y}}_B^{(m)*j} &= \mathbf{X}^{(m)}\mathbf{b}_{\mathbf{Z}^{*j}}^{(m)}
\end{aligned}\right\} \text{ จะได้ } \mathbf{f}_{\mathbf{Z}^{*j}}(\mathbf{x}_i) = \begin{bmatrix} i\text{th row of } \hat{\mathbf{y}}_B^{(1)*j} \\ i\text{th row of } \hat{\mathbf{y}}_B^{(2)*j} \\ \vdots \\ i\text{th row of } \hat{\mathbf{y}}_B^{(m)*j} \end{bmatrix}, i = 1,\ldots,n$$

2

3

3

คำนวณ

$$\boldsymbol{\Delta}_1 = \frac{1}{1000}\sum_{j=1}^{1000}\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}^{*j}}(\mathbf{x}_i)\,\mathbf{f}^T_{\mathbf{Z}^{*j}}(\mathbf{x}_i) - \frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}^{*j}}\left(\mathbf{x}_i^{*j}\right)\mathbf{f}^T_{\mathbf{Z}^{*j}}\left(\mathbf{x}_i^{*j}\right)\right]$$

$$\boldsymbol{\Delta}_2 = \frac{1}{1000}\sum_{j=1}^{1000}\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}^{*j}}(\mathbf{x}_i)y_i - \frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}^{*j}}\left(\mathbf{x}_i^{*j}\right)y_i^{*j}\right]$$

คำนวณค่าพยากรณ์ของ m ตัวแบบเดิม ณ $\mathbf{x}_i$

$$\left.\begin{aligned}\hat{\mathbf{y}}^{(1)} &= \mathbf{X}^{(1)}\mathbf{b}^{(1)}\\ \hat{\mathbf{y}}^{(2)} &= \mathbf{X}^{(2)}\mathbf{b}^{(2)}\\ &\vdots\\ \hat{\mathbf{y}}^{(m)} &= \mathbf{X}^{(m)}\mathbf{b}^{(m)}\end{aligned}\right\} \text{จะได้ } \mathbf{f}_{\mathbf{Z}}(\mathbf{x}_i) = \begin{bmatrix} i\text{th row of } \hat{\mathbf{y}}^{(1)} \\ i\text{th row of } \hat{\mathbf{y}}^{(2)} \\ \vdots \\ i\text{th row of } \hat{\mathbf{y}}^{(m)} \end{bmatrix}, i = 1,\ldots,n$$

คำนวณหาน้ำหนัก

$$\mathbf{w}_{\mathrm{BO}} = \left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}}(\mathbf{x}_i)\mathbf{f}^T_{\mathbf{Z}}(\mathbf{x}_i) + \boldsymbol{\Delta}_1\right]^{-1}\left[\frac{1}{n}\sum_{i=1}^{n}\mathbf{f}_{\mathbf{Z}}(\mathbf{x}_i)y_i + \boldsymbol{\Delta}_2\right]$$

RETURN

ขั้นตอนการหาน้ำหนักโดยวิธี ARM

1. แบ่งข้อมูลตัวอย่างออกเป็น 2 กลุ่ม กลุ่มละเท่า ๆ กัน โดยให้กลุ่มที่หนึ่งแทนด้วย

$$\mathbf{Z}_1 = \left\{ (\mathbf{x}_i, y_i) \;\middle|\; 1 \le i \le 7 \right\}$$

และกลุ่มที่สองแทนด้วย

$$\mathbf{Z}_2 = \left\{ (\mathbf{x}_i, y_i) \;\middle|\; 8 \le i \le 14 \right\}$$

ดังแสดงในตารางต่อไปนี้

	ค่าสังเกตที่	x_1	x_2	x_3	y
$\mathbf{Z}_1$	1	2.73	5.19	6.97	45.85
	2	3.50	3.18	7.64	44.55
	3	2.78	3.42	8.14	46.42
	4	3.02	4.59	8.79	52.45
	5	2.95	4.50	-5.56	26.42
	6	2.53	4.87	4.36	40.26
	7	2.30	4.85	2.25	51.24
$\mathbf{Z}_2$	8	2.16	3.11	0.61	28.10
	9	0.83	1.91	3.92	28.55
	10	2.13	5.15	-2.90	26.14
	11	2.18	2.24	9.34	54.10
	12	3.01	3.26	5.69	32.13
	13	2.38	3.40	0.27	37.26
	14	1.28	2.85	5.16	36.34

2. นำข้อมูล $\mathbf{Z}_1$ มาประมาณค่าสัมประสิทธิ์การถดถอยของตัวแบบแต่ละตัวแบบใหม่ โดยใช้วิธีกำลังสองน้อยสุด ในขั้นนี้จะได้ฟังก์ชันการถดถอยใหม่ดังนี้

$$f_{\mathbf{Z}_1}^{(1)}(\mathbf{x}) = 59.31 - 7.93x_1 + 1.50x_3 \qquad \left(\hat{\sigma}_{\mathbf{Z}_1}^{(1)}\right)^2 = 27.39$$

$$f_{\mathbf{Z}_1}^{(2)}(\mathbf{x}) = 37.44 + 1.38x_3 \qquad \left(\hat{\sigma}_{\mathbf{Z}_1}^{(2)}\right)^2 = 32.50$$

$$f_{\mathbf{Z}_1}^{(3)}(\mathbf{x}) = 24.75 + 2.77x_2 + 1.50x_3 \qquad \left(\hat{\sigma}_{\mathbf{Z}_1}^{(3)}\right)^2 = 34.39$$

3. ประเมินความแม่นยำของตัวแบบที่ได้ในขั้นที่ 2 โดยใช้ข้อมูลตัวอย่างที่เหลือคือ $\mathbf{Z}_2$ กล่าวคือ

   แทนค่า $\mathbf{x}_i$ จากข้อมูล $\mathbf{Z}_2$ $(8 \le i \le 14)$ ลงในตัวแบบทั้ง 3 ตัวแบบในขั้นที่ 2 เพื่อพยากรณ์ค่า y_i จากนั้นคำนวณค่า overall measure of discrepancy ดังนี้

$$D^{(1)} = \sum_{i=8}^{14}\left(y_i - f_{\mathbf{Z}_1}^{(1)}(\mathbf{x}_i)\right)^2 = 1853.08$$
$$D^{(2)} = \sum_{i=8}^{14}\left(y_i - f_{\mathbf{Z}_1}^{(2)}(\mathbf{x}_i)\right)^2 = 617.61$$
$$D^{(3)} = \sum_{i=8}^{14}\left(y_i - f_{\mathbf{Z}_1}^{(3)}(\mathbf{x}_i)\right)^2 = 377.10$$

4. คำนวณน้ำหนักสำหรับแต่ละตัวแบบ จากสูตร

$$w^{(k)} = \frac{\left(\hat{\sigma}_{\mathbf{Z}_1}^{(k)}\right)^{-n/2} \exp\left(\dfrac{-(\hat{\sigma}_{\mathbf{Z}_1}^{(k)})^{-2} D^{(k)}}{2}\right)}{\sum_{q=1}^{3}\left(\hat{\sigma}_{\mathbf{Z}_1}^{(q)}\right)^{-n/2} \exp\left(\dfrac{-(\hat{\sigma}_{\mathbf{Z}_1}^{(q)})^{-2} D^{(q)}}{2}\right)} \quad , \quad k = 1, 2, 3$$

   ได้น้ำหนักดังนี้

$$w^{(1)} = 0.0000$$
$$w^{(2)} = 0.2629$$
$$w^{(3)} = 0.7371$$

5. ทำการสับเปลี่ยนลำดับของข้อมูลอย่างสุ่ม แล้วทำซ้ำขั้นที่ 1-4 เป็นจำนวน R-1 รอบ (เมื่อ R คือจำนวนครั้งของการเรียงสับเปลี่ยนข้อมูลทั้งหมดที่กำหนด) ซึ่งในการวิจัยครั้งนี้จะกำหนดให้ R = 250

6. หาค่าเฉลี่ยของน้ำหนักจากทั้ง 250 รอบ จะได้น้ำหนักของวิธี ARM ดังต่อไปนี้

$$w_{\mathrm{ARM}}^{(1)} = 0.2253$$
$$w_{\mathrm{ARM}}^{(2)} = 0.3716$$
$$w_{\mathrm{ARM}}^{(3)} = 0.4031$$

ภาคผนวก ข.

โปรแกรมทั้งหมดที่ใช้ในการวิจัย

```
program thesis_main
implicit none
integer::ix,n,n_test,p,p1,p2,p3,p4,i,j,r,nboot,i_sim, i_sim_max
! For regressions & combination
double precision, allocatable:: b_apr(:,:), b_forward(:,:), b_backward(:,:), b_stepwise(:,:), bmodel1(:,:),
bmodel2(:,:), bmodel3(:,:), bmodel4(:,:), xmodel1(:,:), xmodel2(:,:), xmodel3(:,:), xmodel4(:,:),
yhatmodel1(:,:), yhatmodel2(:,:) ,yhatmodel3(:,:), yhatmodel4(:,:), b_1(:,:), b_2(:,:), b_3(:,:), b_4(:,:),
w_arm(:,:), w_bt(:,:), w_lp(:,:)
! For generating true model
double precision, allocatable:: xm(:), z(:), mu(:), sigma(:,:), y(:,:), x(:,:), beta(:,:), xbeta(:,:), eps(:,:),
corr_matrix(:,:), stdev_vec(:)
double precision:: sigma_eps, rho0, znorm
!
double precision, allocatable:: y_test(:,:), x_test(:,:), eps_test(:,:), yhat_apr_test(:,:),
yhat_forward_test(:,:), yhat_backward_test(:,:), yhat_stepwise_test(:,:), yhatmodel1_test(:,:),
yhatmodel2_test(:,:), yhatmodel3_test(:,:), yhatmodel4_test(:,:), yhat_arm_test(:,:), yhat_bt_test(:,:),
yhat_lp_test(:,:), mape_apr_test(:), mape_forward_test(:), mape_backward_test(:),
mape_stepwise_test(:), mape_arm_test(:), mape_bt_test(:), mape_lp_test(:)
double precision:: mape_apr_test_average, mape_apr_test_stdev, mape_forward_test_average,
mape_forward_test_stdev, mape_backward_test_average, mape_backward_test_stdev,
mape_stepwise_test_average, mape_stepwise_test_stdev, mape_arm_test_average,
mape_arm_test_stdev, mape_bt_test_average, mape_bt_test_stdev, mape_lp_test_average,
mape_lp_test_stdev
!
integer,allocatable:: numb_of_models(:)
!
  open(25,file='C:/Documents and Settings/tee/Desktop/small_tee_thesis/trial_case.txt')
  !
  ! First, generate data from true model...
  !
  write(*,*) 'n=?, p=?'
  read(*,*) n,p
  write(25,*) 'n=', n
  write(25,*) 'p=', p
  allocate(mu(p), sigma(p,p), beta(p+1,1), corr_matrix(p,p), stdev_vec(p), xm(p), x(n,p+1), xbeta(n,1))
  do i=1,p
    do j=1,p
      if(i==j) then
        corr_matrix(i,j) = 1.0D0
       else
        corr_matrix(i,j) = 0.0D0
      end if
    end do
  end do
  if(p>=3) then
    write(*,*) 'corr_matrix(1,2) =?'
    read(*,*) corr_matrix(1,2)
    write(25,*) 'corr_matrix(1,2) is', corr_matrix(1,2)
    corr_matrix(2,1) = corr_matrix(1,2)
  end if
  if(p>=5) then
    write(*,*) 'corr_matrix(4,5) =?'
    read(*,*) corr_matrix(4,5)
           write(25,*) 'corr_matrix(4,5) is', corr_matrix(4,5)
    corr_matrix(5,4) = corr_matrix(4,5)
```

```
end if
if(p==7) then
   write(*,*) 'corr_matrix(6,7) =?'
   read(*,*) corr_matrix(6,7)
   write(25,*) 'corr_matrix(6,7) is', corr_matrix(6,7)
   corr_matrix(7,6) = corr_matrix(6,7)
end if
write(*,*) 'correlation matrix is', ((corr_matrix(i,j), j=1,p), i=1,p)
write(25,*) 'correlation matrix is', ((corr_matrix(i,j), j=1,p), i=1,p)
if(p>0) stdev_vec(1) = 1.2D0
if(p>1) stdev_vec(2) = 2.4D0
if(p>2) stdev_vec(3) = 4.0D0
if(p>3) stdev_vec(4) = 6.0D0
if(p>4) stdev_vec(5) = 8.4D0
if(p>5) stdev_vec(6) = 11.2D0
if(p>6) stdev_vec(7) = 14.4D0
write(*,*) 'stdev_vec is', stdev_vec
write(25,*) 'stdev_vec is', stdev_vec
call covariance(corr_matrix, stdev_vec, sigma, p)
write(*,*) 'covariance matrix is', ((sigma(i,j), j=1,p), i=1,p)
write(25,*) 'covariance matrix is', ((sigma(i,j), j=1,p), i=1,p)
if(p>0) mu(1) = 3.0D0
if(p>1) mu(2) = 4.0D0
if(p>2) mu(3) = 5.0D0
if(p>3) mu(4) = 6.0D0
if(p>4) mu(5) = 7.0D0
if(p>5) mu(6) = 8.0D0
if(p>6) mu(7) = 9.0D0
write(*,*) 'mu is', mu
write(25,*) 'mu is', mu
beta(1,1) = 6.0D0
if(p>0) beta(2,1) = 4.0D0
if(p>1) beta(3,1) = 4.0D0
if(p>2) beta(4,1) = 2.0D0
if(p>3) beta(5,1) = 2.0D0
if(p>4) beta(6,1) = 1.0D0
if(p>5) beta(7,1) = 1.0D0
if(p>6) beta(8,1) = 1.0D0
write(*,*) 'beta is', beta
write(25,*) 'beta is', beta
write(*,*) 'Enter sigma_eps'
read(*,*) sigma_eps
write(25,*) 'sigma_eps is', sigma_eps
write(*,*) 'Enter SEED'
read(*,*) ix
write(25,*) 'SEED ix is', ix
x(:,1) = 1.0D0
do i=1,n
   call mnorm(ix,mu,sigma,p,xm)
   do j=2,p+1
      x(i,j) = xm(j-1)
   end do
end do
xbeta = matmul(x,beta)
17 format(f22.15)
write(*,*) '1 x1 x2...xp is'
write(*,17) x
write(25,*) '1 x1 x2...xp is'
write(25,17) x
write(*,*) 'xbeta is'
```

```
   write(*,17) xbeta
   write(25,*) 'xbeta is'
   write(25,17) xbeta
   write(*,*) 'nboot=?'
   read(*,*) nboot
   write(25,*) 'nboot is', nboot
   write(*,*) 'i_sim_max =?'
   read(*,*) i_sim_max
   write(25,*) 'i_sim_max is', i_sim_max

allocate(numb_of_models(i_sim_max),mape_apr_test(i_sim_max),mape_forward_test(i_sim_max),ma
pe_backward_test(i_sim_max),mape_stepwise_test(i_sim_max), mape_arm_test(i_sim_max),
mape_bt_test(i_sim_max), mape_lp_test(i_sim_max))
   do i_sim = 1, i_sim_max
            write(25,*) 'ix is', ix
     allocate(y(n,1), eps(n,1), z(n))
     !
     ! simulate eps ~ N(0, sigma_eps square)
     !
     do i=1,n
       z(i) = znorm(ix)
       eps(i,1) = sigma_eps*z(i)
     end do
     y = xbeta+eps
     allocate(b_apr(p+1,1), b_forward(p+1,1), b_backward(p+1,1), b_stepwise(p+1,1), b_1(p+1,1),
b_2(p+1,1), b_3(p+1,1), b_4(p+1,1), yhatmodel1(n,1), yhatmodel2(n,1) ,yhatmodel3(n,1),
yhatmodel4(n,1))
     call
models_4methods(y,x,n,p,b_apr,b_forward,b_backward,b_stepwise,p1,p2,p3,p4,b_1,b_2,b_3,b_4)
     allocate(bmodel1(p1+1,1), bmodel2(p2+1,1), bmodel3(p3+1,1), bmodel4(p4+1,1),
xmodel1(n,p1+1), xmodel2(n,p2+1), xmodel3(n,p3+1), xmodel4(n,p4+1))
     call
models_to_be_combined(x,n,p,r,p1,p2,p3,p4,b_1,b_2,b_3,b_4,bmodel1,bmodel2,bmodel3,bmodel4,xm
odel1,xmodel2,xmodel3,xmodel4,yhatmodel1,yhatmodel2,yhatmodel3,yhatmodel4)
     allocate(w_arm(r,1), w_bt(r,1), w_lp(r,1))
     write(25,*) 'For i_sim', i_sim,',', 'r is', r
     write(*,*) 'For i_sim', i_sim,',', 'r is', r
     numb_of_models(i_sim) = r
     call comb_lp(n,r,p1,p2,p3,p4,y,yhatmodel1,yhatmodel2,yhatmodel3,yhatmodel4,w_lp)
     write(25,*) 'w_lp is', w_lp
     write(25,*) 'sum(w_lp) =', sum(w_lp)
     call
comb_bootstrap(ix,n,p1,p2,p3,p4,r,y,xmodel1,xmodel2,xmodel3,xmodel4,yhatmodel1,yhatmodel2,yha
tmodel3,yhatmodel4,nboot,w_bt)
     write(25,*) 'w_bt is', w_bt
     write(25,*) 'sum(w_bt) =', sum(w_bt)
     call arm(ix,y,xmodel1,xmodel2,xmodel3,xmodel4,n,p1,p2,p3,p4,r,w_arm)
     write(25,*) 'w_arm is', w_arm
     write(25,*) 'sum(w_arm) =', sum(w_arm)
     n_test = n
     allocate(y_test(n_test,1), x_test(n_test,p+1), eps_test(n_test,1), yhat_apr_test(n_test,1),
yhat_forward_test(n_test,1), yhat_backward_test(n_test,1), yhat_stepwise_test(n_test,1),
yhatmodel1_test(n_test,1), yhatmodel2_test(n_test,1), yhatmodel3_test(n_test,1),
yhatmodel4_test(n_test,1), yhat_arm_test(n_test,1), yhat_bt_test(n_test,1), yhat_lp_test(n_test,1))
     x_test = x
     eps_test = eps
     y_test = y
     !
     yhat_apr_test = matmul(x_test, b_apr)
     yhat_forward_test = matmul(x_test, b_forward)
```

```
      yhat_backward_test = matmul(x_test, b_backward)
      yhat_stepwise_test = matmul(x_test, b_stepwise)
      if(p1/=-1) yhatmodel1_test=matmul(x_test,b_1)
      if(p2/=-1) yhatmodel2_test=matmul(x_test,b_2)
      if(p3/=-1) yhatmodel3_test=matmul(x_test,b_3)
      if(p4/=-1) yhatmodel4_test=matmul(x_test,b_4)
      if(r==1) then
         yhat_arm_test = w_arm(1,1)*yhatmodel1_test
         yhat_bt_test = w_bt(1,1)*yhatmodel1_test
         yhat_lp_test = w_lp(1,1)*yhatmodel1_test
        else if(r==2) then
         yhat_arm_test = w_arm(1,1)*yhatmodel1_test+ w_arm(2,1)*yhatmodel2_test
         yhat_bt_test = w_bt(1,1)*yhatmodel1_test+ w_bt(2,1)*yhatmodel2_test
         yhat_lp_test = w_lp(1,1)*yhatmodel1_test+ w_lp(2,1)*yhatmodel2_test
        else if(r==3) then
         yhat_arm_test = w_arm(1,1)*yhatmodel1_test+ w_arm(2,1)*yhatmodel2_test+
w_arm(3,1)*yhatmodel3_test
         yhat_bt_test = w_bt(1,1)*yhatmodel1_test+ w_bt(2,1)*yhatmodel2_test+
w_bt(3,1)*yhatmodel3_test
         yhat_lp_test = w_lp(1,1)*yhatmodel1_test+ w_lp(2,1)*yhatmodel2_test+
w_lp(3,1)*yhatmodel3_test
        else if(r==4) then
         yhat_arm_test = w_arm(1,1)*yhatmodel1_test+ w_arm(2,1)*yhatmodel2_test+
w_arm(3,1)*yhatmodel3_test+w_arm(4,1)*yhatmodel4_test
         yhat_bt_test = w_bt(1,1)*yhatmodel1_test+ w_bt(2,1)*yhatmodel2_test+
w_bt(3,1)*yhatmodel3_test+w_bt(4,1)*yhatmodel4_test
         yhat_lp_test = w_lp(1,1)*yhatmodel1_test+ w_lp(2,1)*yhatmodel2_test+
w_lp(3,1)*yhatmodel3_test+w_lp(4,1)*yhatmodel4_test
      end if
      mape_apr_test(i_sim) = 100.0D0*sum(abs((y_test-yhat_apr_test)/y_test))/dble(n_test)
      mape_forward_test(i_sim) = 100.0D0*sum(abs((y_test-yhat_forward_test)/y_test))/dble(n_test)
      mape_backward_test(i_sim) = 100.0D0*sum(abs((y_test-yhat_backward_test)/y_test))/dble(n_test)
      mape_stepwise_test(i_sim) = 100.0D0*sum(abs((y_test-yhat_stepwise_test)/y_test))/dble(n_test)
      !
      mape_arm_test(i_sim) = 100.0D0*sum(abs((y_test-yhat_arm_test)/y_test))/dble(n_test)
      mape_bt_test(i_sim) = 100.0D0*sum(abs((y_test-yhat_bt_test)/y_test))/dble(n_test)
      mape_lp_test(i_sim) = 100.0D0*sum(abs((y_test-yhat_lp_test)/y_test))/dble(n_test)
      write(25,*) 'MAPE for APR is', mape_apr_test(i_sim)
      write(25,*) 'MAPE for FORWARD is', mape_forward_test(i_sim)
      write(25,*) 'MAPE for BACKWARD is', mape_backward_test(i_sim)
      write(25,*) 'MAPE for STEPWISE is', mape_stepwise_test(i_sim)
      write(25,*) 'MAPE for ARM is', mape_arm_test(i_sim)
      write(25,*) 'MAPE for BT is', mape_bt_test(i_sim)
      write(25,*) 'MAPE for LP is', mape_lp_test(i_sim)

deallocate(bmodel1,bmodel2,bmodel3,bmodel4,xmodel1,xmodel2,xmodel3,xmodel4,w_arm,w_bt,w_l
p)
      deallocate(b_apr, b_forward, b_backward, b_stepwise, b_1, b_2, b_3, b_4, yhatmodel1,
yhatmodel2, yhatmodel3, yhatmodel4)
      deallocate(y_test, x_test, eps_test, yhatmodel1_test, yhatmodel2_test, yhatmodel3_test,
yhatmodel4_test, yhat_apr_test, yhat_forward_test, yhat_backward_test, yhat_stepwise_test,
yhat_arm_test, yhat_bt_test, yhat_lp_test)
      deallocate(y,z,eps)
   end do
   mape_apr_test_average = sum(mape_apr_test)/dble(i_sim_max)
   mape_apr_test_stdev = sqrt((sum(mape_apr_test**2.0D0)-
sum(mape_apr_test)**2.0D0/dble(i_sim_max))/dble(i_sim_max))
   !
   mape_forward_test_average = sum(mape_forward_test)/dble(i_sim_max)
```

```
  mape_forward_test_stdev = sqrt((sum(mape_forward_test**2.0D0)-
sum(mape_forward_test)**2.0D0/dble(i_sim_max))/dble(i_sim_max))
  !
  mape_backward_test_average = sum(mape_backward_test)/dble(i_sim_max)
  mape_backward_test_stdev = sqrt((sum(mape_backward_test**2.0D0)-
sum(mape_backward_test)**2.0D0/dble(i_sim_max))/dble(i_sim_max))
  !
  mape_stepwise_test_average = sum(mape_stepwise_test)/dble(i_sim_max)
  mape_stepwise_test_stdev = sqrt((sum(mape_stepwise_test**2.0D0)-
sum(mape_stepwise_test)**2.0D0/dble(i_sim_max))/dble(i_sim_max))
  !
  mape_arm_test_average = sum(mape_arm_test)/dble(i_sim_max)
  mape_arm_test_stdev = sqrt((sum(mape_arm_test**2.0D0)-
sum(mape_arm_test)**2.0D0/dble(i_sim_max))/dble(i_sim_max))
  !
  mape_bt_test_average = sum(mape_bt_test)/dble(i_sim_max)
  mape_bt_test_stdev = sqrt((sum(mape_bt_test**2.0D0)-
sum(mape_bt_test)**2.0D0/dble(i_sim_max))/dble(i_sim_max))
  !
  mape_lp_test_average = sum(mape_lp_test)/dble(i_sim_max)
  mape_lp_test_stdev = sqrt((sum(mape_lp_test**2.0D0)-
sum(mape_lp_test)**2.0D0/dble(i_sim_max))/dble(i_sim_max))
  !
  write(25,*) '      '
  write(25,*) 'The number of 1-model cases =', count(numb_of_models==1)
  write(25,*) 'The number of 2-model cases =', count(numb_of_models==2)
  write(25,*) 'The number of 3-model cases =', count(numb_of_models==3)
  write(25,*) 'The number of 4-model cases =', count(numb_of_models==4)
  write(25,*) '      '
  !
  write(25,*) 'Average MAPE for APR is     ', mape_apr_test_average
  write(25,*) 'SD of MAPE for APR is       ', mape_apr_test_stdev
  write(25,*) '  '
  !
  write(25,*) 'Average MAPE for FORWARD is ', mape_forward_test_average
  write(25,*) 'SD of MAPE for FORWARD is   ', mape_forward_test_stdev
  write(25,*) '  '
  !
  write(25,*) 'Average MAPE for BACKWARD is', mape_backward_test_average
  write(25,*) 'SD. of MAPE for BACKWARD is ', mape_backward_test_stdev
  write(25,*) '  '
  !
  write(25,*) 'Average MAPE for STEPWISE is', mape_stepwise_test_average
  write(25,*) 'SD of MAPE for STEPWISE is  ', mape_stepwise_test_stdev
  write(25,*) '  '
  !
  write(25,*) 'Average MAPE for ARM is     ', mape_arm_test_average
  write(25,*) 'SD of MAPE for ARM is       ', mape_arm_test_stdev
  write(25,*) '  '
  !
  write(25,*) 'Average MAPE for BT is      ', mape_bt_test_average
  write(25,*) 'SD of MAPE for BT is        ', mape_bt_test_stdev
  write(25,*) '  '
  !
  write(25,*) 'Average MAPE for LP is      ', mape_lp_test_average
  write(25,*) 'SD of MAPE for LP is        ', mape_lp_test_stdev
  write(25,*) '  '
  !
```

```
deallocate(numb_of_models,mape_apr_test,mape_forward_test,mape_backward_test,mape_stepwise_t
est, mape_arm_test, mape_bt_test, mape_lp_test)
  deallocate(mu,sigma,beta,corr_matrix,stdev_vec,xm,x,xbeta)
  close(25)
stop
end program thesis_main
!
!
!
subroutine arm(ix,y,xmodel1,xmodel2,xmodel3,xmodel4,n,p1,p2,p3,p4,r,w_arm)
implicit none
integer:: ix,n,p1,p2,p3,p4,r,i,j0,j1,im
double precision:: y(n,1), xmodel1(n,p1+1), xmodel2(n,p2+1), xmodel3(n,p3+1), xmodel4(n,p4+1),
y_perm(n,1), xmodel1_perm(n,p1+1), xmodel2_perm(n,p2+1), xmodel3_perm(n,p3+1),
xmodel4_perm(n,p4+1), y_data1(n/2,1), y_data2(n/2,1), xmodel1_data1(n/2,p1+1),
xmodel1_data2(n/2,p1+1), xmodel2_data1(n/2,p2+1), xmodel2_data2(n/2,p2+1),
xmodel3_data1(n/2,p3+1), xmodel3_data2(n/2,p3+1),
xmodel4_data1(n/2,p4+1),xmodel4_data2(n/2,p4+1), bmodel1_data1(p1+1,1), bmodel2_data1(p2+1,1),
bmodel3_data1(p3+1,1), bmodel4_data1(p4+1,1), yhat(n/2,1), e(n/2,1),
yhat_model1_data2(n/2,1),yhat_model2_data2(n/2,1), yhat_model3_data2(n/2,1),
yhat_model4_data2(n/2,1), w1(250), w2(250), w3(250), w4(250), w_arm(r,1)
integer:: v1ton (n)
double precision:: s_sq_model1_data1, s_sq_model2_data1, s_sq_model3_data1,s_sq_model4_data1,
ss_reg,ms_reg,ss_res,f_cal,d1,d2,d3,d4,w1hat,w2hat,w3hat,w4hat,numer1,numer2,numer3,numer4
!
! n MUST be EVEN
!
outermost: do
  if(r==1) exit outermost
  do j0=1,n
    v1ton(j0)=j0
  end do
  !
  !
  ! Before permutation, set:
  y_perm=y
  if(p1/=-1) xmodel1_perm=xmodel1
  if(p2/=-1) xmodel2_perm=xmodel2
  if(p3/=-1) xmodel3_perm=xmodel3
  if(p4/=-1) xmodel4_perm=xmodel4
  !
  !
  im=0
  do i=1,250
    !
    !STEP 1
    !
    y_data1 = y_perm(1:n/2,:)
    y_data2 = y_perm(n/2+1:n,:)
    !
    if(p1/=-1) xmodel1_data1 = xmodel1_perm(1:n/2,:)
    if(p2/=-1) xmodel2_data1 = xmodel2_perm(1:n/2,:)
    if(p3/=-1) xmodel3_data1 = xmodel3_perm(1:n/2,:)
    if(p4/=-1) xmodel4_data1 = xmodel4_perm(1:n/2,:)
    if(p1/=-1) xmodel1_data2 = xmodel1_perm(n/2+1:n,:)
    if(p2/=-1) xmodel2_data2 = xmodel2_perm(n/2+1:n,:)
    if(p3/=-1) xmodel3_data2 = xmodel3_perm(n/2+1:n,:)
    if(p4/=-1) xmodel4_data2 = xmodel4_perm(n/2+1:n,:)
    !
```

```
     ! STEP 2
     !
     if(p1/=-1) call
ols(y_data1,xmodel1_data1,bmodel1_data1,n/2,p1,yhat,e,ss_reg,ms_reg,ss_res,s_sq_model1_data1,f_c
al)
     if(p2/=-1) call
ols(y_data1,xmodel2_data1,bmodel2_data1,n/2,p2,yhat,e,ss_reg,ms_reg,ss_res,s_sq_model2_data1,f_c
al)
     if(p3/=-1) call
ols(y_data1,xmodel3_data1,bmodel3_data1,n/2,p3,yhat,e,ss_reg,ms_reg,ss_res,s_sq_model3_data1,f_c
al)
     if(p4/=-1) call
ols(y_data1,xmodel4_data1,bmodel4_data1,n/2,p4,yhat,e,ss_reg,ms_reg,ss_res,s_sq_model4_data1,f_c
al)
     !
     ! STEP 3
     !
     if(p1/=-1) yhat_model1_data2 = matmul(xmodel1_data2,bmodel1_data1)
     if(p2/=-1) yhat_model2_data2 = matmul(xmodel2_data2,bmodel2_data1)
     if(p3/=-1) yhat_model3_data2 = matmul(xmodel3_data2,bmodel3_data1)
     if(p4/=-1) yhat_model4_data2 = matmul(xmodel4_data2,bmodel4_data1)
     if(p1/=-1) d1 = sum((y_data2-yhat_model1_data2)**2.0D0)
     if(p2/=-1) d2 = sum((y_data2-yhat_model2_data2)**2.0D0)
     if(p3/=-1) d3 = sum((y_data2-yhat_model3_data2)**2.0D0)
     if(p4/=-1) d4 = sum((y_data2-yhat_model4_data2)**2.0D0)
     !
     ! STEP 4
     !
     if(p1/=-1) then
       numer1 = ((s_sq_model1_data1)**(-dble(n)/4.0D0))*exp(-((s_sq_model1_data1)**(-
1.0D0))*d1/2.0D0)
      else
       numer1 = 0.0D0
     end if
     if(p2/=-1) then
       numer2 = ((s_sq_model2_data1)**(-dble(n)/4.0D0))*exp(-((s_sq_model2_data1)**(-
1.0D0))*d2/2.0D0)
      else
       numer2 = 0.0D0
     end if
     if(p3/=-1) then
       numer3 = ((s_sq_model3_data1)**(-dble(n)/4.0D0))*exp(-((s_sq_model3_data1)**(-
1.0D0))*d3/2.0D0)
      else
       numer3 = 0.0D0
     end if
     if(p4/=-1) then
       numer4 = ((s_sq_model4_data1)**(-dble(n)/4.0D0))*exp(-((s_sq_model4_data1)**(-
1.0D0))*d4/2.0D0)
      else
       numer4 = 0.0D0
     end if
     do
       if(numer1+numer2+numer3+numer4==0.0D0) then
                      im=im+1
                              w1(i) = 0.0D0
                              w2(i) = 0.0D0
                              w3(i) = 0.0D0
                              w4(i) = 0.0D0
                              exit
```

```
        else
         w1(i) = numer1/(numer1+numer2+numer3+numer4)
         w2(i) = numer2/(numer1+numer2+numer3+numer4)
         w3(i) = numer3/(numer1+numer2+numer3+numer4)
         w4(i) = numer4/(numer1+numer2+numer3+numer4)
         exit
       end if
    end do
    !
    ! STEP 5
    !
    call perm_random2(n,v1ton,ix)
    do j1=1,n
       y_perm(j1,:) = y(v1ton(j1),:)
       if(p1/=-1) xmodel1_perm(j1,:) = xmodel1(v1ton(j1),:)
       if(p2/=-1) xmodel2_perm(j1,:) = xmodel2(v1ton(j1),:)
       if(p3/=-1) xmodel3_perm(j1,:) = xmodel3(v1ton(j1),:)
       if(p4/=-1) xmodel4_perm(j1,:) = xmodel4(v1ton(j1),:)
    end do
  end do
  w1hat = sum(w1)/dble(250-im)
  w2hat = sum(w2)/dble(250-im)
  w3hat = sum(w3)/dble(250-im)
  w4hat = sum(w4)/dble(250-im)
  !
  if(p1/=-1) w_arm(1,1) = w1hat
  if(p2/=-1) w_arm(2,1) = w2hat
  if(p3/=-1) w_arm(3,1) = w3hat
  if(p4/=-1) w_arm(4,1) = w4hat
  exit outermost
end do outermost
if(r==1) w_arm = 1.0D0
return
end subroutine arm
!
!
!
subroutine
comb_bootstrap(ix,n,p1,p2,p3,p4,r,y,xmodel1,xmodel2,xmodel3,xmodel4,yhatmodel1,yhatmodel2,yha
tmodel3,yhatmodel4,nboot,w_bt)
implicit none
integer:: ix,n,p1,p2,p3,p4,r,i,j,nboot
double precision::  w_bt(r,1), y(n,1), e(n,1), yst(n,1), xmodel1(n,p1+1), xmodel2(n,p2+1),
xmodel3(n,p3+1), xmodel4(n,p4+1), yhatmodel1(n,1), yhatmodel2(n,1) ,yhatmodel3(n,1),
yhatmodel4(n,1), xst_model1(n,p1+1), xst_model2(n,p2+1), xst_model3(n,p3+1), xst_model4(n,p4+1),
xst_model1s(nboot,n,p1+1), xst_model2s(nboot,n,p2+1), xst_model3s(nboot,n,p3+1),
xst_model4s(nboot,n,p4+1), yst_s(nboot,n,1), bst_model1s(nboot,p1+1,1), bst_model2s(nboot,p2+1,1),
bst_model3s(nboot,p3+1,1), bst_model4s(nboot,p4+1,1), yhatst_model1s_xst(nboot,n,1),
yhatst_model2s_xst(nboot,n,1), yhatst_model3s_xst(nboot,n,1), yhatst_model4s_xst(nboot,n,1),
yhatst_model1s_x(nboot,n,1), yhatst_model2s_x(nboot,n,1), yhatst_model3s_x(nboot,n,1),
yhatst_model4s_x(nboot,n,1), yhatst_s_x(nboot,r,1), yhatst_s_xst(nboot,r,1), sumfordel1hat(r,r),
sumfordel2hat(r,1), product11(r,r), sum11(r,r), product12(r,r), sum12(r,r), product21(r,1), sum21(r,1),
product22(r,1), sum22(r,1), sum1112(r,r), sum2122(r,1), del1hat(r,r), del2hat(r,1), yhat(r,n,1),
product1(r,r), product2(r,1), sum1(r,r), sum2(r,1), mcc(r,r), mcy(r,1), mcc_inv(r,r)
double precision:: ss_reg,ms_reg,ss_res,ms_res,f_cal
outermost: do
  if(r==1) exit outermost
  !
  !
  do i=1, nboot
```

```
    call
bootstrap4models(ix,n,p1,p2,p3,p4,y,xmodel1,xmodel2,xmodel3,xmodel4,yst,xst_model1,xst_model2,
xst_model3,xst_model4)
    if(p1/=-1) xst_model1s(i,:,:) = xst_model1(:,:)
    if(p2/=-1) xst_model2s(i,:,:) = xst_model2(:,:)
    if(p3/=-1) xst_model3s(i,:,:) = xst_model3(:,:)
    if(p4/=-1) xst_model4s(i,:,:) = xst_model4(:,:)
    yst_s(i,:,:) = yst(:,:)
    if(p1/=-1) call ols(yst_s(i,:,:), xst_model1s(i,:,:), bst_model1s(i,:,:), n, p1,
yhatst_model1s_xst(i,:,:),e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    if(p2/=-1) call ols(yst_s(i,:,:), xst_model2s(i,:,:), bst_model2s(i,:,:), n, p2,
yhatst_model2s_xst(i,:,:),e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    if(p3/=-1) call ols(yst_s(i,:,:), xst_model3s(i,:,:), bst_model3s(i,:,:), n, p3,
yhatst_model3s_xst(i,:,:),e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    if(p4/=-1) call ols(yst_s(i,:,:), xst_model4s(i,:,:), bst_model4s(i,:,:), n, p4,
yhatst_model4s_xst(i,:,:),e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
  end do
  !
  do i=1,nboot
    if(p1/=-1) yhatst_model1s_x(i,:,:) = matmul(xmodel1, bst_model1s(i,:,:))
    if(p2/=-1) yhatst_model2s_x(i,:,:) = matmul(xmodel2, bst_model2s(i,:,:))
    if(p3/=-1) yhatst_model3s_x(i,:,:) = matmul(xmodel3, bst_model3s(i,:,:))
    if(p4/=-1) yhatst_model4s_x(i,:,:) = matmul(xmodel4, bst_model4s(i,:,:))
  end do
  !
  sumfordel1hat=0.0D0
  sumfordel2hat=0.0D0
  do j=1,nboot
    sum11=0.0D0
    sum21=0.0D0
    do i=1,n
      !
      ! yhatst_s_x(sample,model,:)
      !
      if(p1/=-1) yhatst_s_x(j,1,:) = yhatst_model1s_x(j,i,:)
      if(p2/=-1) yhatst_s_x(j,2,:) = yhatst_model2s_x(j,i,:)
      if(p3/=-1) yhatst_s_x(j,3,:) = yhatst_model3s_x(j,i,:)
      if (p4/=-1) yhatst_s_x(j,4,:) = yhatst_model4s_x(j,i,:)
      product11 = matmul(yhatst_s_x(j,:,:), transpose(yhatst_s_x(j,:,:)))
      sum11 = sum11+product11(1,1)
      product21=(yhatst_s_x(j,:,:))*y(i,1)
      sum21 = sum21+product21(1,1)
    end do
    sum12=0.0D0
    sum22=0.0D0
    do i=1,n
      if(p1/=-1) yhatst_s_xst(j,1,:) = yhatst_model1s_xst(j,i,:)
      if(p2/=-1) yhatst_s_xst(j,2,:) = yhatst_model2s_xst(j,i,:)
      if(p3/=-1) yhatst_s_xst(j,3,:) = yhatst_model3s_xst(j,i,:)
      if(p4/=-1) yhatst_s_xst(j,4,:) = yhatst_model4s_xst(j,i,:)
      product12 = matmul(yhatst_s_xst(j,:,:), transpose(yhatst_s_xst(j,:,:)))
      sum12 = sum12+product12(1,1)
      product22=(yhatst_s_xst(j,:,:))*yst_s(j,i,1)
      sum22 = sum22+product22(1,1)
    end do
    sum1112=sum11/dble(n) - sum12/dble(n)
    sumfordel1hat = sumfordel1hat + sum1112
    sum2122=sum21/dble(n) - sum22/dble(n)
    sumfordel2hat = sumfordel2hat + sum2122
  end do
```

```
  del1hat = sumfordel1hat/dble(nboot)
  del2hat = sumfordel2hat/dble(nboot)
  !
  !
  do i=1,n
    !
    ! yhat(model,i,:)
    !
    if(p1/=-1) yhat(1,i,:) = yhatmodel1(i,:)
    if(p2/=-1) yhat(2,i,:) = yhatmodel2(i,:)
    if(p3/=-1) yhat(3,i,:) = yhatmodel3(i,:)
    if(p4/=-1) yhat(4,i,:) = yhatmodel4(i,:)
  end do
  sum1=0.0D0
  sum2=0.0D0
  do i=1,n
    product1 = matmul(yhat(:,i,:), transpose(yhat(:,i,:)))
    sum1 = sum1 + product1
    product2 = (yhat(:,i,:))*y(i,1)
    sum2 = sum2 + product2
  end do
  mcc = sum1/dble(n) + del1hat
  mcy = sum2/dble(n) + del2hat
  !
  !
  call invm(r,mcc,mcc_inv)
  w_bt = matmul(mcc_inv, mcy)
  exit outermost
end do outermost
if(r==1) w_bt = 1.0D0
return
end subroutine comb_bootstrap
!
!
!
subroutine
bootstrap4models(ix,n,p1,p2,p3,p4,y,xmodel1,xmodel2,xmodel3,xmodel4,yst,xst_model1,xst_model2,
xst_model3,xst_model4)
implicit none
integer:: n,p1,p2,p3,p4,i_random,irand,i,ix
double precision:: y(n,1), yst(n,1), xmodel1(n,p1+1), xmodel2(n,p2+1), xmodel3(n,p3+1),
xmodel4(n,p4+1), xst_model1(n,p1+1), xst_model2(n,p2+1), xst_model3(n,p3+1), xst_model4(n,p4+1)
do i=1,n
  irand = i_random(1,n,ix)
  yst(i,:) = y(irand,:)
  if(p1/=-1) xst_model1(i,:) = xmodel1(irand,:)
  if(p2/=-1) xst_model2(i,:) = xmodel2(irand,:)
  if(p3/=-1) xst_model3(i,:) = xmodel3(irand,:)
  if(p4/=-1) xst_model4(i,:) = xmodel4(irand,:)
end do
return
end subroutine bootstrap4models
!
!
!
!
!
subroutine comb_lp(n,r,p1,p2,p3,p4,y,yhatmodel1,yhatmodel2,yhatmodel3,yhatmodel4,w_lp)
implicit none
integer:: n,r,i,j,p1,p2,p3,p4
```

```
double precision:: zmin
double precision:: y(n,1), yhatmodel1(n,1), yhatmodel2(n,1) ,yhatmodel3(n,1), yhatmodel4(n,1),
c0(1,(2*n+r)), a((n+1),(2*n+r)), b((n+1),1), xopt((2*n+r)), w_lp(r)
outermost: do
  if(r==1) exit outermost
  !
  do i=1,2*n
    c0(1,i) = 1.0D0
  end do
  !
  do i=(2*n+1),(2*n+r)
   c0(1,i) = 0.0D0
  end do
  !
  do i=1,n+1
    do j=1,2*n+r
      a(i,j)=0.0D0
    end do
  end do
  !
  do i=1,2*n
   a((n+1),i) = 0.0D0
  end do
  !
  do i=1,r
    a((n+1),2*n+i) = 1.0D0
  end do
  !
  b((n+1),1) = 1.0D0
  !
  j=1
  do i=1,n
    a(i,j) = 1.0D0
    j=j+2
  end do
  !
  j=2
  do i=1,n
    a(i,j) = -1.0D0
    j=j+2
  end do
  !
  where(a(1:n,1:2*n)/=1.0D0.and.a(1:n,1:2*n)/=-1.0D0)
    a(1:n,1:2*n)=0.0D0
  end where
  !
  do i=1,n
    if(p1/=-1) a(i,2*n+1) = yhatmodel1(i,1)
    if(p2/=-1) a(i,2*n+2) = yhatmodel2(i,1)
    if(p3/=-1) a(i,2*n+3) = yhatmodel3(i,1)
    if(p4/=-1) a(i,2*n+4) = yhatmodel4(i,1)
    b(i,1) = y(i,1)
  end do
  !write(*,*) 'c0 is', c0
  !write(*,*) 'a is', a
  !write(*,*) 'b is', b
  !write(*,*) '2*n+r is', 2*n+r
  !write(*,*) 'n+1 is', n+1
  call min_lp(c0,a,b,(2*n+r),(n+1),zmin,xopt)
  !write(*,*) 'zmin is', zmin
```

```
  !
  do i=1,r
     w_lp(i) = xopt(2*n+i)
  end do
  exit outermost
end do outermost
if(r==1) w_lp = 1.0D0
return
end subroutine comb_lp
!
!
!
!
!
!
!
subroutine min_lp(c0,a,b,n,m,zmin,xopt)
implicit none
integer:: n,m,i,j,e,l
integer:: xb(m), minl_minus_c1(2), minl_ratio(1), minl_ls1(1), minl_q21(2)
double precision:: zmin
double precision:: c0(1,n), c1(1,n), c1_ext(1,n+m), c2(1,n), c2_ext(1,n+m), cb(1,m), a(m,n),
identity(m,m), ai(m,n+m), b(m,1), minus_c1(1,n), ratio(m), basis(m,m), basis_inv(m,m), q1(1,n),
q2(1,m), q3(m,n), q5(1,1), q6(m,1), ls1(1+m,1+n+m), rs1(1+m), h(m,n), k(m,1), q21(1,n), q22(1,1),
ls2(m+1,n+1), rs2(1+m), minl_ls2(1), xopt(n)
!
!
do i=1,m
  do j=1,m
     if(i==j) then
       identity(i,j)=1.0D0
      else
       identity(i,j)=0.0D0
     end if
  end do
end do
!
ai(:,1:n)=a
ai(:,n+1:n+m)=identity
!
!
! PHASE 1
!
!
do i=1,n
  c1(1,i) = sum(a(:,i))
end do
c1_ext(:,1:n) =c1
c1_ext(:,n+1:n+m) = 0.0D0
!
! Iteration 0
!
do i=1,m
  xb(i) = n+i
end do
minus_c1 = (-1)*c1
if(all(minus_c1>=0.0D0)) write(25,*) 'huge problem in PH1: It.0'
minl_minus_c1 = minloc(minus_c1)
e=minl_minus_c1(2)
where(a(:,e)>0.0D0)
```

```
   ratio=b(:,1)/a(:,e)
  elsewhere
   ratio=1.0D+10
end where
minl_ratio=minloc(ratio)
l=minl_ratio(1)
xb(l)=e
!
! Iteration "Any"
!
loop1: do
   do i=1,m
      basis(:,i) = ai(:,xb(i))
      cb(1,i) = c1_ext(1,xb(i))
   end do
   ls1(1,1) = -1.0D0
   do i=2,m+1
      ls1(i,1) = 0.0D0
   end do
   call invm(m,basis,basis_inv)
   q3 = matmul(basis_inv,a)
   q1 = matmul(cb,q3) - c1
   q2 = matmul(cb,basis_inv)
   ls1(1,2:n+1) = q1(1,:)
   ls1(1,n+2:n+m+1) = q2(1,:)
   ls1(2:m+1,2:n+1) = q3
   ls1(2:m+1,n+2:n+m+1) = basis_inv
   q5 = matmul(q2,b)-sum(b)
   q6 = matmul(basis_inv,b)
   rs1(1) = q5(1,1)
   rs1(2:m+1) = q6(:,1)
   if(all(ls1(1,2:n+m+1) > -1.0D-6)) exit loop1
   minl_ls1 = minloc(ls1(1,2:n+m+1))
   e = minl_ls1(1)+1
   where(ls1(2:m+1,e)>0.0D0)
      ratio = rs1(2:m+1)/ls1(2:m+1,e)
     elsewhere
      ratio = 1.0D+10
   endwhere
   minl_ratio = minloc(ratio)
   l = minl_ratio(1)
   xb(l) = e-1
end do loop1
!
!
! PHASE 2
!
!
c2 = (-1)*c0
!
!Iteration 0
!
h = ls1(2:m+1,2:n+1)
k(:,1) = rs1(2:m+1)
ls2(1,1) = -1.0D0
do i=2,m+1
   ls2(i,1) = 0.0D0
end do
c2_ext(:,1:n) = c2
c2_ext(:,n+1:n+m) = 0.0D0
```

```
do i=1,m
   cb(1,i) = c2_ext(1,xb(i))
end do
q21 = matmul(cb,h) - c2
ls2(1,2:n+1) = q21(1,1:n)
ls2(2:m+1,2:n+1) = h
q22 = matmul(cb,k)
rs2(1)=q22(1,1)
rs2(2:m+1) = k(:,1)
!
loopspecial: do
   if(all(q21>-1.0D-6)) exit loopspecial
   minl_q21 = minloc(q21)
   e = minl_q21(2)
   where(h(:,e)>0.0D0)
      ratio = k(:,1)/h(:,e)
     elsewhere
      ratio = 1.0D+10
   end where
   minl_ratio = minloc(ratio)
   l = minl_ratio(1)
   xb(l) = e
   !
   ! Iteration "Any"
   !
   loop2: do
      do i=1,m
         basis(:,i) = ai(:,xb(i))
                           cb(1,i) = c2_ext(1,xb(i))
      end do
      ls2(1,1) = -1.0D0
      do i=2,m+1
         ls2(i,1) = 0.0D0
      end do
      call invm(m,basis,basis_inv)
      q3 = matmul(basis_inv,a)
      q1 = matmul(cb,q3)-c2
      ls2(1,2:n+1) = q1(1,:)
      ls2(2:m+1,2:n+1) = q3
      q2 = matmul(cb, basis_inv)
      q5 = matmul(q2,b)
      q6 = matmul(basis_inv,b)
      rs2(1) = q5(1,1)
      rs2(2:m+1) = q6(:,1)
      if(all(ls2(1,2:n+1)>-1.0D-6)) exit loopspecial
      minl_ls2 = minloc(ls2(1,2:n+1))
      e = minl_ls2(1)+1
      where(ls2(2:m+1,e)>0.0D0)
         ratio = rs2(2:m+1)/ls2(2:m+1,e)
        elsewhere
         ratio = 1.0D+10
      endwhere
      minl_ratio = minloc(ratio)
      l = minl_ratio(1)
      xb(l) = e-1
   end do loop2
end do loopspecial
!
!
zmin = -rs2(1)
```

```
do i=1,n
   xopt(i) = 0.0D0
end do
do i=1,m
   xopt(xb(i)) = rs2(i+1)
end do
end subroutine min_lp
!
!
!
!
!
!
subroutine perm_random2(n,iarray,ix)
implicit none
integer:: n,i,j,i_random,ix
integer:: iarray(n)
do i=1,n
   j=i_random(i,n,ix)
   call i_swap(iarray(i),iarray(j))
end do
return
end subroutine perm_random2
!
!
function i_random(ilo,ihi,ix)
implicit none
integer:: i_random,ilo,ihi,ix
double precision:: urand,r
r=urand(ix)
i_random=ilo+int(r*dble(ihi+1-ilo))
i_random=max(i_random,ilo)
i_random=min(i_random,ihi)
return
end function i_random
!
!
subroutine i_swap(i,j)
implicit none
integer::i,j,k
k=i
i=j
j=k
return
end subroutine i_swap
!
!
!
!
!
subroutine models_4methods(y,x,n,p,b1,b2,b3,b4,p1,p2,p3,p4,b_1,b_2,b_3,b_4)
implicit none
integer:: n,p,p1,p2,p3,p4
double precision:: y(n,1), x(n,p+1), b1(p+1,1), b2(p+1,1), b3(p+1,1), b4(p+1,1), yhat_apr(n,1),
mape_minv(1), b_1(p+1,1), b_2(p+1,1) ,b_3(p+1,1), b_4(p+1,1), f0(p), f0x(p,p), f0xx(p,p,p),
f0xxx(p,p,p,p), f0xxxx(p,p,p,p,p), f0xxxxx(p,p,p,p,p,p), f0xxxxxx(p,p,p,p,p,p,p)
integer:: b1d(p+1,1), b2d(p+1,1), b3d(p+1,1), b4d(p+1,1)
!
call allpossible(y,x,n,p,b1,yhat_apr,mape_minv)
!
```

```
call partial_f(y,x,n,p,f0,f0x,f0xx,f0xxx,f0xxxx,f0xxxxx,f0xxxxxx)
!
call forward(y,x,b2,n,p,f0,f0x,f0xx,f0xxx,f0xxxx,f0xxxxx,f0xxxxxx)
call backward(y,x,b3,n,p,f0,f0x,f0xx,f0xxx,f0xxxx,f0xxxxx,f0xxxxxx)
call stepwise(y,x,b4,n,p,f0,f0x,f0xx,f0xxx,f0xxxx,f0xxxxx,f0xxxxxx)
!
where(b1==0.0D0)
   b1d=0
  elsewhere
   b1d=1
end where
!
where(b2==0.0D0)
   b2d=0
  elsewhere
   b2d=1
end where
!
where(b3==0.0D0)
   b3d=0
  elsewhere
   b3d=1
end where
!
where(b4==0.0D0)
   b4d=0
  elsewhere
   b4d=1
end where
!
!
! CASE0
!
if(any(b1d/=b2d).and.any(b1d/=b3d).and.any(b1d/=b4d).and.any(b2d/=b3d).and.any(b2d/=b4d).and.an
y(b3d/=b4d)) then
   b_1=b1
   b_2=b2
   b_3=b3
   b_4=b4
end if
!
! CASE1
!
if(all(b1d==b2d).and.any(b1d/=b3d).and.any(b1d/=b4d).and.any(b2d/=b3d).and.any(b2d/=b4d).and.an
y(b3d/=b4d)) then
   b_1=b1
   b_2=b3
   b_3=b4
   b_4=0.0D0
end if
if(any(b1d/=b2d).and.all(b1d==b3d).and.any(b1d/=b4d).and.any(b2d/=b3d).and.any(b2d/=b4d).and.an
y(b3d/=b4d)) then
   b_1=b1
   b_2=b2
   b_3=b4
   b_4=0.0D0
end if
if(any(b1d/=b2d).and.any(b1d/=b3d).and.all(b1d==b4d).and.any(b2d/=b3d).and.any(b2d/=b4d).and.an
y(b3d/=b4d)) then
   b_1=b1
```

```
  b_2=b2
  b_3=b3
  b_4=0.0D0
end if
if(any(b1d/=b2d).and.any(b1d/=b3d).and.any(b1d/=b4d).and.all(b2d==b3d).and.any(b2d/=b4d).and.an
y(b3d/=b4d)) then
  b_1=b1
  b_2=b2
  b_3=b4
  b_4=0.0D0
end if
if(any(b1d/=b2d).and.any(b1d/=b3d).and.any(b1d/=b4d).and.any(b2d/=b3d).and.all(b2d==b4d).and.an
y(b3d/=b4d)) then
  b_1=b1
  b_2=b2
  b_3=b3
  b_4=0.0D0
end if
if(any(b1d/=b2d).and.any(b1d/=b3d).and.any(b1d/=b4d).and.any(b2d/=b3d).and.any(b2d/=b4d).and.all
(b3d==b4d)) then
  b_1=b1
  b_2=b2
  b_3=b3
  b_4=0.0D0
end if
!
! CASE 2
!
if(all(b1d==b2d).and.all(b1d==b3d).and.any(b1d/=b4d)) then
  b_1=b1
  b_2=b4
  b_3=0.0D0
  b_4=0.0D0
end if
if(all(b1d==b2d).and.all(b1d==b4d).and.any(b1d/=b3d)) then
  b_1=b1
  b_2=b3
  b_3=0.0D0
  b_4=0.0D0
end if
if(all(b1d==b3d).and.all(b1d==b4d).and.any(b1d/=b2d)) then
  b_1=b1
  b_2=b2
  b_3=0.0D0
  b_4=0.0D0
end if
if(all(b2d==b3d).and.all(b2d==b4d).and.any(b1d/=b2d)) then
  b_1=b1
  b_2=b2
  b_3=0.0D0
  b_4=0.0D0
end if
!
! CASE 3
!
if(all(b1d==b2d).and.all(b3d==b4d).and.any(b2d/=b3d)) then
  b_1=b1
  b_2=b3
  b_3=0.0D0
  b_4=0.0D0
```

```
end if
if(all(b1d==b3d).and.all(b2d==b4d).and.any(b3d/=b2d)) then
  b_1=b1
  b_2=b2
  b_3=0.0D0
  b_4=0.0D0
end if
if(all(b1d==b4d).and.all(b2d==b3d).and.any(b4d/=b2d)) then
  b_1=b1
  b_2=b2
  b_3=0.0D0
  b_4=0.0D0
end if
!
! CASE 4
!
if(all(b1d==b2d).and.all(b1d==b3d).and.all(b1d==b4d)) then
  b_1=b1
  b_2=0.0D0
  b_3=0.0D0
  b_4=0.0D0
end if
!
!
!
p1 = count(b_1/=0.0D0)-1
p2 = count(b_2/=0.0D0)-1
p3 = count(b_3/=0.0D0)-1
p4 = count(b_4/=0.0D0)-1
!
!
return
end subroutine models_4methods
!
!
!
subroutine
models_to_be_combined(x,n,p,r,p1,p2,p3,p4,b_1,b_2,b_3,b_4,bmodel1,bmodel2,bmodel3,bmodel4,xm
odel1,xmodel2,xmodel3,xmodel4,yhatmodel1,yhatmodel2,yhatmodel3,yhatmodel4)
implicit none
integer:: n,p,r,p1,p2,p3,p4,i,j,k
double precision:: x(n,p+1), b_1(p+1,1), b_2(p+1,1), b_3(p+1,1), b_4(p+1,1), bmodel1(p1+1,1),
bmodel2(p2+1,1), bmodel3(p3+1,1), bmodel4(p4+1,1), xmodel1(n,p1+1), xmodel2(n,p2+1),
xmodel3(n,p3+1), xmodel4(n,p4+1), yhatmodel1(n,1), yhatmodel2(n,1) ,yhatmodel3(n,1),
yhatmodel4(n,1)
!
if(p4/=-1.and.p3/=-1.and.p2/=-1.and.p1/=-1) r=4
if(p4==-1.and.p3/=-1.and.p2/=-1.and.p1/=-1) r=3
if(p4==-1.and.p3==-1.and.p2/=-1.and.p1/=-1) r=2
if(p4==-1.and.p3==-1.and.p2==-1.and.p1/=-1) r=1
if(p4==-1.and.p3==-1.and.p2==-1.and.p1==-1) r=0
!
! MODEL 1
!
if(p1/=-1) then
  bmodel1(1,1) = b_1(1,1)
  xmodel1(:,1) = x(:,1)
  if(p1>0) then
    k=2
    outer1: do i=2,p1+1
```

```
        inner1: do j=k,p+1
          if(b_1(j,1) == 0.0D0) then
            k=k+1
            cycle inner1
          end if
          bmodel1(i,1) = b_1(j,1)
          xmodel1(:,i) = x(:,j)
          k=k+1
          cycle outer1
        end do inner1
      end do outer1
    end if
    yhatmodel1=matmul(xmodel1,bmodel1)
  end if
  !
  ! MODEL 2
  !
  if(p2/=-1) then
    bmodel2(1,1) = b_2(1,1)
    xmodel2(:,1) = x(:,1)
    if(p2>0) then
      k=2
      outer2: do i=2,p2+1
        inner2: do j=k,p+1
          if(b_2(j,1) == 0.0D0) then
            k=k+1
            cycle inner2
          end if
          bmodel2(i,1) = b_2(j,1)
          xmodel2(:,i) = x(:,j)
          k=k+1
          cycle outer2
        end do inner2
      end do outer2
    end if
    yhatmodel2=matmul(xmodel2,bmodel2)
  end if
  !
  ! MODEL 3
  !
  if(p3/=-1) then
    bmodel3(1,1) = b_3(1,1)
    xmodel3(:,1) = x(:,1)
    if(p3>0) then
      k=2
      outer3: do i=2,p3+1
        inner3: do j=k,p+1
          if(b_3(j,1) == 0.0D0) then
            k=k+1
            cycle inner3
          end if
          bmodel3(i,1) = b_3(j,1)
          xmodel3(:,i) = x(:,j)
          k=k+1
          cycle outer3
        end do inner3
      end do outer3
    end if
    yhatmodel3=matmul(xmodel3,bmodel3)
  end if
```

```
!
! MODEL 4
!
if(p4/=-1) then
  bmodel4(1,1) = b_4(1,1)
  xmodel4(:,1) = x(:,1)
  if(p4>0) then
    k=2
    outer4: do i=2,p4+1
      inner4: do j=k,p+1
        if(b_4(j,1) == 0.0D0) then
          k=k+1
          cycle inner4
        end if
        bmodel4(i,1) = b_4(j,1)
        xmodel4(:,i) = x(:,j)
        k=k+1
        cycle outer4
      end do inner4
    end do outer4
  end if
  yhatmodel4=matmul(xmodel4,bmodel4)
end if
return
end subroutine models_to_be_combined
!
!
!
!
!
subroutine allpossible(y,x,n,p,b_best,yhat_best,mape_minv)
implicit none
integer:: n,p,m,i1,i2,i3,i4,i5,i6,i7,j
double precision:: y(n,1), x(n,p+1), b1(2,1), x1(n,2), b2(3,1), x2(n,3), b3(4,1), x3(n,4), b4(5,1), x4(n,5),
b5(6,1), x5(n,6), b6(7,1), x6(n,7), b7(8,1), x7(n,8), yhat(n,1), e(n,1), b(p+1,2**p), mape(2**p),
mape_minv(1), mape_minl(1), b_best(p+1,1), yhat_best(n,1)
double precision:: f_cal, ms_reg, ms_res, ss_reg, ss_res
outermost: do
  j=1
  !
  ! SUBSET 0: y = beta0 + eps
  !
  yhat = sum(y)/dble(n)
  mape(1) = 100.0D0*sum(abs((y-yhat)/y))/dble(n)
  b(1,j) = yhat(1,1)
  do m=2,p+1
    b(m,j) = 0.0D0
  end do
  j=j+1
  !
  ! SUBSET 1: y = beta0 + betax + eps
  !
  do i1=2,p+1
    x1(:,1) = 1.0D0
    x1(:,2) = x(:,i1)
    call ols(y,x1,b1,n,1,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    mape(j) = 100.0D0*sum(abs((y-yhat)/y))/dble(n)
    b(1,j) = b1(1,1)
    b(i1,j) = b1(2,1)
    do m=2,p+1
```

```
    if(m/=i1) b(m,j)=0.0D0
  end do
  j=j+1
end do
if(p<2) exit outermost
!
! SUBSET 2: y = beta0+betax+betax+eps
!
loop2a: do i1=2,p
  x2(:,1) = 1.0D0
  x2(:,2) = x(:,i1)
  loop2b: do i2=3,p+1
    if(i2<=i1) cycle loop2b
    x2(:,3)=x(:,i2)
    call ols(y,x2,b2,n,2,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    mape(j) = 100.0D0*sum(abs((y-yhat)/y))/dble(n)
    b(1,j) = b2(1,1)
    b(i1,j) = b2(2,1)
    b(i2,j) = b2(3,1)
    do m=2,p+1
      if(m/=i1.and.m/=i2) b(m,j)=0.0D0
    end do
    j=j+1
  end do loop2b
end do loop2a
if(p<3) exit outermost
!
! SUBSET 3: y = beta0 + betax + betax + betax +eps
!
loop3a: do i1=2,p-1
  x3(:,1) = 1.0D0
  x3(:,2) = x(:,i1)
  loop3b: do i2=3,p
    if(i2<=i1) cycle loop3b
    x3(:,3)=x(:,i2)
    loop3c: do i3=4,p+1
      if(i3<=i2) cycle loop3c
      x3(:,4) = x(:,i3)
      call ols(y,x3,b3,n,3,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
      mape(j) = 100.0D0*sum(abs((y-yhat)/y))/dble(n)
      b(1,j) = b3(1,1)
      b(i1,j) = b3(2,1)
      b(i2,j) = b3(3,1)
      b(i3,j) = b3(4,1)
      do m=2,p+1
        if(m/=i1.and.m/=i2.and.m/=i3) b(m,j)=0.0D0
      end do
      j=j+1
    end do loop3c
  end do loop3b
end do loop3a
if(p<4) exit outermost
!
! SUBSET 4: y = beta0 + betax + betax + betax + betax + eps
!
loop4a: do i1=2,p-2
  x4(:,1) = 1.0D0
  x4(:,2) = x(:,i1)
  loop4b: do i2=3,p-1
    if(i2<=i1) cycle loop4b
```

```
      x4(:,3)=x(:,i2)
      loop4c: do i3=4,p
        if(i3<=i2) cycle loop4c
        x4(:,4) = x(:,i3)
        loop4d: do i4=5,p+1
          if(i4<=i3) cycle loop4d
          x4(:,5) = x(:,i4)
          call ols(y,x4,b4,n,4,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
          mape(j) = 100.0D0*sum(abs((y-yhat)/y))/dble(n)
          b(1,j) = b4(1,1)
          b(i1,j) = b4(2,1)
          b(i2,j) = b4(3,1)
          b(i3,j) = b4(4,1)
          b(i4,j) = b4(5,1)
          do m=2,p+1
            if(m/=i1.and.m/=i2.and.m/=i3.and.m/=i4) b(m,j)=0.0D0
          end do
          j=j+1
        end do loop4d
      end do loop4c
    end do loop4b
  end do loop4a
  if(p<5) exit outermost
  !
  ! SUBSET 5: y = beta0 + betax + betax + betax + betax + betax + eps
  !
  loop5a: do i1=2,p-3
    x5(:,1) = 1.0D0
    x5(:,2) = x(:,i1)
    loop5b: do i2=3,p-2
      if(i2<=i1) cycle loop5b
      x5(:,3)=x(:,i2)
      loop5c: do i3=4,p-1
        if(i3<=i2) cycle loop5c
        x5(:,4) = x(:,i3)
        loop5d: do i4=5,p
          if(i4<=i3) cycle loop5d
          x5(:,5) = x(:,i4)
          loop5e: do i5=6,p+1
            if(i5<=i4) cycle loop5e
            x5(:,6) = x(:,i5)
            call ols(y,x5,b5,n,5,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
            mape(j) = 100.0D0*sum(abs((y-yhat)/y))/dble(n)
            b(1,j) = b5(1,1)
            b(i1,j) = b5(2,1)
            b(i2,j) = b5(3,1)
            b(i3,j) = b5(4,1)
            b(i4,j) = b5(5,1)
            b(i5,j) = b5(6,1)
            do m=2,p+1
              if(m/=i1.and.m/=i2.and.m/=i3.and.m/=i4.and.m/=i5) b(m,j)=0.0D0
            end do
            j=j+1
          end do loop5e
        end do loop5d
      end do loop5c
    end do loop5b
  end do loop5a
  if(p<6) exit outermost
  !
```

```
! SUBSET 6: y = beta0 + betax + betax + betax + betax + betax + betax + eps
!
loop6a: do i1=2,p-4
  x6(:,1) = 1.0D0
  x6(:,2) = x(:,i1)
  loop6b: do i2=3,p-3
    if(i2<=i1) cycle loop6b
    x6(:,3)=x(:,i2)
    loop6c: do i3=4,p-2
      if(i3<=i2) cycle loop6c
      x6(:,4) = x(:,i3)
      loop6d: do i4=5,p-1
        if(i4<=i3) cycle loop6d
        x6(:,5) = x(:,i4)
        loop6e: do i5=6,p
          if(i5<=i4) cycle loop6e
          x6(:,6) = x(:,i5)
          loop6f: do i6=7,p+1
            if(i6<=i5) cycle loop6f
            x6(:,7) = x(:,i6)
            call ols(y,x6,b6,n,6,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
            mape(j) = 100.0D0*sum(abs((y-yhat)/y))/dble(n)
            b(1,j) = b6(1,1)
            b(i1,j) = b6(2,1)
            b(i2,j) = b6(3,1)
            b(i3,j) = b6(4,1)
            b(i4,j) = b6(5,1)
            b(i5,j) = b6(6,1)
            b(i6,j) = b6(7,1)
            do m=2,p+1
              if(m/=i1.and.m/=i2.and.m/=i3.and.m/=i4.and.m/=i5.and.m/=i6) b(m,j)=0.0D0
            end do
            j=j+1
          end do loop6f
        end do loop6e
      end do loop6d
    end do loop6c
  end do loop6b
end do loop6a
if(p<7) exit outermost
!
! SUBSET 7: y = beta0 + betax + betax + betax + betax + betax + betax + betax + eps
!
loop7a: do i1=2,p-5
  x7(:,1) = 1.0D0
  x7(:,2) = x(:,i1)
  loop7b: do i2=3,p-4
    if(i2<=i1) cycle loop7b
    x7(:,3)=x(:,i2)
    loop7c: do i3=4,p-3
      if(i3<=i2) cycle loop7c
      x7(:,4) = x(:,i3)
      loop7d: do i4=5,p-2
        if(i4<=i3) cycle loop7d
        x7(:,5) = x(:,i4)
        loop7e: do i5=6,p-1
          if(i5<=i4) cycle loop7e
          x7(:,6) = x(:,i5)
          loop7f: do i6=7,p
            if(i6<=i5) cycle loop7f
```

```
                    x7(:,7) = x(:,i6)
                    loop7g: do i7=8,p+1
                      if(i7<=i6) cycle loop7g
                      x7(:,8) = x(:,i7)
                      call ols(y,x7,b7,n,7,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
                      mape(j) = 100.0D0*sum(abs((y-yhat)/y))/dble(n)
                      b(1,j) = b7(1,1)
                      b(i1,j) = b7(2,1)
                      b(i2,j) = b7(3,1)
                      b(i3,j) = b7(4,1)
                      b(i4,j) = b7(5,1)
                      b(i5,j) = b7(6,1)
                      b(i6,j) = b7(7,1)
                      b(i7,j) = b7(8,1)
                      do m=2,p+1
                        if(m/=i1.and.m/=i2.and.m/=i3.and.m/=i4.and.m/=i5.and.m/=i6.and.m/=i7)
b(m,j)=0.0D0
                      end do
                      j=j+1
                    end do loop7g
                  end do loop7f
                end do loop7e
              end do loop7d
            end do loop7c
          end do loop7b
  end do loop7a
  if(p<8) exit outermost
end do outermost
mape_minv = minval(mape)
mape_minl = minloc(mape)
b_best(:,1) = b(:,mape_minl(1))
yhat_best = matmul(x,b_best)
write(25,*) 'b_best is', b_best
write(25,*) 'mape_minv is', mape_minv
return
end subroutine allpossible
!
!
!
subroutine forward(y,x,b,n,p,f0,f0x,f0xx,f0xxx,f0xxxx,f0xxxxx,f0xxxxxx)
implicit none
integer:: n,p,i,z1,z2,z3,z4,z5,z6,z7
double precision:: y(n,1), x(n,p+1), b(p+1,1), r(p), abs_r(p), bz1(2,1), bz1z2(3,1), bz1z2z3(4,1),
bz1z2z3z4(5,1), bz1z2z3z4z5(6,1), bz1z2z3z4z5z6(7,1), xz1(n,2), xz1z2(n,3), xz1z2z3(n,4),
xz1z2z3z4(n,5), xz1z2z3z4z5(n,6), xz1z2z3z4z5z6(n,7), yhat(n,1), e(n,1),f0(p), f0x(p,p), f0xx(p,p,p),
f0xxx(p,p,p,p), f0xxxx(p,p,p,p,p), f0xxxxx(p,p,p,p,p,p), f0xxxxxx(p,p,p,p,p,p,p)
double precision:: ss_reg,ms_reg,ss_res,ms_res,f_cal,finv,corr
integer:: abs_r_maxl(1), f0x_maxl(1), f0xx_maxl(1), f0xxx_maxl(1), f0xxxx_maxl(1),
f0xxxxx_maxl(1), f0xxxxxx_maxl(1)
outer: do
  inner: do
    !
    ! STEP 1: Seek z1
    !
    do i=1,p
      r(i) = corr(x(:,i+1), y(:,1),n)
    end do
    abs_r = abs(r)
    abs_r_maxl = maxloc(abs_r)
    z1 = abs_r_maxl(1)
```

```
!
! STEP 2: Check z1
!
if (f0(z1) <= finv(1,n-2,0.95D0)) then
   b(1,1) = sum(y)/dble(n)
   do i=2,p+1
      b(i,1) = 0.0D0
   end do
   write(25,*) 'No Xs are in Eq.'
   write(25,*) 'Coeff. b =', b
   if(.true.) exit outer
  else
   if(p==1) exit inner
   !
   ! STEP 3: Seek z2
   !
   f0x_maxl = maxloc(f0x(:,z1))
   z2 = f0x_maxl(1)
   !
   ! STEP 4: Check z2
   !
   if (f0x(z2,z1) <= finv(1,n-3,0.95D0)) then
      xz1(:,1) = x(:,1)
      xz1(:,2) = x(:,z1+1)
      call ols(y,xz1,bz1,n,1,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
      b(1,1) = bz1(1,1)
      b(z1+1,1) = bz1(2,1)
      do i=1,p+1
         if(i==1.or.i==z1+1) cycle
         b(i,1) = 0.0D0
      end do
      write(25,*) 'Order of Xs entering Eq:', z1
      write(25,*) 'Coeff. b =', b
      if(.true.) exit outer
     else
      if(p==2) exit inner
      !
      ! STEP 5: Seek z3
      !
      f0xx_maxl = maxloc(f0xx(:,z1,z2))
      z3 = f0xx_maxl(1)
      !
      ! STEP 6: Check z3
      !
      if (f0xx(z3,z1,z2) <= finv(1,n-4,0.95D0)) then
         xz1z2(:,1) = x(:,1)
         xz1z2(:,2) = x(:,z1+1)
         xz1z2(:,3) = x(:,z2+1)
         call ols(y,xz1z2,bz1z2,n,2,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
         b(1,1) = bz1z2(1,1)
         b(z1+1,1) = bz1z2(2,1)
         b(z2+1,1) = bz1z2(3,1)
         do i=1, p+1
            if(i==1.or.i==z1+1.or.i==z2+1) cycle
            b(i,1) = 0.0D0
         end do
         write(25,*) 'Order of Xs entering Eq:', z1, z2
         write(25,*) 'Coef. b =', b
         if(.true.) exit outer
        else
```

```
if(p==3) exit inner
!
! STEP 7: Seek z4
!
f0xxx_maxl = maxloc(f0xxx(:,z1,z2,z3))
z4 = f0xxx_maxl(1)
!
! STEP 8: Check z4
!
if (f0xxx(z4,z1,z2,z3) <= finv(1,n-5,0.95D0)) then
  xz1z2z3(:,1) = x(:,1)
  xz1z2z3(:,2) = x(:,z1+1)
  xz1z2z3(:,3) = x(:,z2+1)
  xz1z2z3(:,4) = x(:,z3+1)
  call ols(y,xz1z2z3,bz1z2z3,n,3,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
  b(1,1) = bz1z2z3(1,1)
  b(z1+1,1) = bz1z2z3(2,1)
  b(z2+1,1) = bz1z2z3(3,1)
  b(z3+1,1) = bz1z2z3(4,1)
  do i=1,p+1
    if(i==1.or.i==z1+1.or.i==z2+1.or.i==z3+1) cycle
    b(i,1) = 0.0D0
  end do
  write(25,*) 'Order of Xs entering Eq:', z1, z2, z3
  write(25,*) 'Coef. b =', b
  if(.true.) exit outer
 else
  if(p==4) exit inner
  !
  ! STEP 9: Seek z5
  !
  f0xxxx_maxl = maxloc(f0xxxx(:,z1,z2,z3,z4))
  z5 = f0xxxx_maxl(1)
  !
  ! STEP 10: Check z5
  !
  if (f0xxxx(z5,z1,z2,z3,z4) <= finv(1,n-6,0.95D0)) then
    xz1z2z3z4(:,1) = x(:,1)
    xz1z2z3z4(:,2) = x(:,z1+1)
    xz1z2z3z4(:,3) = x(:,z2+1)
    xz1z2z3z4(:,4) = x(:,z3+1)
    xz1z2z3z4(:,5) = x(:,z4+1)
    call ols(y,xz1z2z3z4,bz1z2z3z4,n,4,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    b(1,1) = bz1z2z3z4(1,1)
    b(z1+1,1) = bz1z2z3z4(2,1)
    b(z2+1,1) = bz1z2z3z4(3,1)
    b(z3+1,1) = bz1z2z3z4(4,1)
    b(z4+1,1) = bz1z2z3z4(5,1)
    do i=1,p+1
      if(i==1.or.i==z1+1.or.i==z2+1.or.i==z3+1.or.i==z4+1) cycle
      b(i,1) = 0.0D0
    end do
    write(25,*) 'Order of Xs entering Eq:', z1, z2, z3, z4
    write(25,*) 'Coef. b =', b
    if(.true.) exit outer
   else
    if(p==5) exit inner
    !
    ! STEP 11: Seek z6
    !
```

```
f0xxxxx_maxl = maxloc(f0xxxxx(:,z1,z2,z3,z4,z5))
z6 = f0xxxxx_maxl(1)
!
! STEP 12: Check z6
!
if (f0xxxxx(z6,z1,z2,z3,z4,z5) <= finv(1,n-7,0.95D0)) then
  xz1z2z3z4z5(:,1) = x(:,1)
  xz1z2z3z4z5(:,2) = x(:,z1+1)
  xz1z2z3z4z5(:,3) = x(:,z2+1)
  xz1z2z3z4z5(:,4) = x(:,z3+1)
  xz1z2z3z4z5(:,5) = x(:,z4+1)
  xz1z2z3z4z5(:,6) = x(:,z5+1)
  call ols(y,xz1z2z3z4z5,bz1z2z3z4z5,n,5,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
  b(1,1) = bz1z2z3z4z5(1,1)
  b(z1+1,1) = bz1z2z3z4z5(2,1)
  b(z2+1,1) = bz1z2z3z4z5(3,1)
  b(z3+1,1) = bz1z2z3z4z5(4,1)
  b(z4+1,1) = bz1z2z3z4z5(5,1)
  b(z5+1,1) = bz1z2z3z4z5(6,1)
  do i=1,p+1
    if(i==1.or.i==z1+1.or.i==z2+1.or.i==z3+1.or.i==z4+1.or.i==z5+1) cycle
    b(i,1) = 0.0D0
  end do
  write(25,*) 'Order of Xs entering Eq:', z1, z2, z3, z4, z5
  write(25,*) 'Coef. b =', b
  if(.true.) exit outer
 else
  if(p==6) exit inner
  !
  ! STEP 13: Seek z7
  !
  f0xxxxxx_maxl = maxloc(f0xxxxxx(:,z1,z2,z3,z4,z5,z6))
  z7 = f0xxxxxx_maxl(1)
  !
  ! STEP 14: Check z7
  !
  if (f0xxxxxx(z7,z1,z2,z3,z4,z5,z6) <= finv(1,n-8,0.95D0)) then
    xz1z2z3z4z5z6(:,1) = x(:,1)
    xz1z2z3z4z5z6(:,2) = x(:,z1+1)
    xz1z2z3z4z5z6(:,3) = x(:,z2+1)
    xz1z2z3z4z5z6(:,4) = x(:,z3+1)
    xz1z2z3z4z5z6(:,5) = x(:,z4+1)
    xz1z2z3z4z5z6(:,6) = x(:,z5+1)
    xz1z2z3z4z5z6(:,7) = x(:,z6+1)
    call ols(y,xz1z2z3z4z5z6,bz1z2z3z4z5z6,n,6,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    b(1,1) = bz1z2z3z4z5z6(1,1)
    b(z1+1,1) = bz1z2z3z4z5z6(2,1)
    b(z2+1,1) = bz1z2z3z4z5z6(3,1)
    b(z3+1,1) = bz1z2z3z4z5z6(4,1)
    b(z4+1,1) = bz1z2z3z4z5z6(5,1)
    b(z5+1,1) = bz1z2z3z4z5z6(6,1)
    b(z6+1,1) = bz1z2z3z4z5z6(7,1)
    do i=1,p+1
      if(i==1.or.i==z1+1.or.i==z2+1.or.i==z3+1.or.i==z4+1.or.i==z5+1.or.i==z6+1) cycle
      b(i,1) = 0.0D0
    end do
    write(25,*) 'Order of Xs entering Eq:', z1, z2, z3, z4, z5, z6
    write(25,*) 'Coef. b =', b
    if(.true.) exit outer
   else
```

```
                    if(p==7) exit inner
                  end if
                end if
              end if
            end if
          end if
        end if
      end if
    end do inner
    call ols(y,x,b,n,p,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    write(25,*) 'All Xs are in Eq.'
    if(p==1) write(25,*) 'Order of Xs entering Eq:', z1
    if(p==2) write(25,*) 'Order of Xs entering Eq:', z1, z2
    if(p==3) write(25,*) 'Order of Xs entering Eq:', z1, z2, z3
    if(p==4) write(25,*) 'Order of Xs entering Eq:', z1, z2, z3, z4
    if(p==5) write(25,*) 'Order of Xs entering Eq:', z1, z2, z3, z4, z5
    if(p==6) write(25,*) 'Order of Xs entering Eq:', z1, z2, z3, z4, z5, z6
    if(p==7) write(25,*) 'Order of Xs entering Eq:', z1, z2, z3, z4, z5, z6, z7
    write(25,*) 'Coef. b =', b
    if(.true.) exit outer
  end do outer
  return
  end subroutine forward
!
!
!
subroutine backward(y,x,b,n,p,f0,f0x,f0xx,f0xxx,f0xxxx,f0xxxxx,f0xxxxxx)
implicit none
integer:: n,p,i,j,z1,z2,z3,z4,z5,z6,z7
double precision:: y(n,1), x(n,p+1), b(p+1,1), b_noz1(7,1), b_noz1z2(6,1), b_noz1z2z3(5,1),
b_noz1z2z3z4(4,1), b_noz1z2z3z4z5(3,1), b_noz1z2z3z4z5z6(2,1), x_noz1(n,7), x_noz1z2(n,6),
x_noz1z2z3(n,5), x_noz1z2z3z4(n,4), x_noz1z2z3z4z5(n,3), x_noz1z2z3z4z5z6(n,2), yhat(n,1),
e(n,1),f0(p), f0x(p,p), f0xx(p,p,p), f0xxx(p,p,p,p), f0xxxx(p,p,p,p,p), f0xxxxx(p,p,p,p,p,p),
f0xxxxxx(p,p,p,p,p,p,p), f0xxxxxx_minv(1), f0xxxxx_noz1(p,p,p,p,p,p), f0xxxxx_noz1_minv(1),
f0xxxx_noz1z2(p,p,p,p,p), f0xxxx_noz1z2_minv(1), f0xxx_noz1z2z3(p,p,p,p),
f0xxx_noz1z2z3_minv(1), f0xx_noz1z2z3z4(p,p,p), f0xx_noz1z2z3z4_minv(1),
f0x_noz1z2z3z4z5(p,p), f0x_noz1z2z3z4z5_minv(1), f0_noz1z2z3z4z5z6(p),
f0_noz1z2z3z4z5z6_minv(1)
double precision:: ss_reg,ms_reg,ss_res,ms_res,f_cal,finv
integer:: f0xxxxxx_minl(7), f0xxxxx_noz1_minl(6), f0xxxx_noz1z2_minl(5),
f0xxx_noz1z2z3_minl(4), f0xx_noz1z2z3z4_minl(3), f0x_noz1z2z3z4z5_minl(2),
f0_noz1z2z3z4z5z6_minl(1), i_array(p), i_array_sorted(p)
!
!
outermost: do
  loop1: do
    if(p<7) exit loop1
    f0xxxxxx_minl = minloc(f0xxxxxx,f0xxxxxx>=0.0D0)
    z1 = f0xxxxxx_minl(1)
    f0xxxxxx_minv = minval(f0xxxxxx,mask=f0xxxxxx>=0.0D0)
    if(f0xxxxxx_minv(1) > finv(1,n-8,0.90D0)) then
      call ols(y,x,b,n,7,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
      write(25,*) 'All Xs are retained.'
      write(25,*) 'Coef. b =', b
      if(.true.) exit outermost
    end if
    if(.true.) exit loop1
  end do loop1
  !
  ! z1 removed
```

```
!
loop2: do
  if(p<6) exit loop2
  if(p==6) z1=999
  f0xxxxx_noz1 = f0xxxxx
  if(z1/=999) then
    f0xxxxx_noz1(z1,:,:,:,:,:)=-9.0D0
    f0xxxxx_noz1(:,z1,:,:,:,:)=-9.0D0
    f0xxxxx_noz1(:,:,z1,:,:,:)=-9.0D0
    f0xxxxx_noz1(:,:,:,z1,:,:)=-9.0D0
    f0xxxxx_noz1(:,:,:,:,z1,:)=-9.0D0
    f0xxxxx_noz1(:,:,:,:,:,z1)=-9.0D0
  end if
  f0xxxxx_noz1_minl = minloc(f0xxxxx_noz1, f0xxxxx_noz1>=0.0D0)
  z2 = f0xxxxx_noz1_minl(1)
  f0xxxxx_noz1_minv = minval(f0xxxxx_noz1, mask=f0xxxxx_noz1>=0.0D0)
  if(f0xxxxx_noz1_minv(1) > finv(1,n-7,0.90D0)) then
    do i=1,p
      if(i==z1) then
        i_array(i) = 777
       else
        i_array(i) = i
      end if
    end do
    call sortinteger(i_array,i_array_sorted,p)
    x_noz1(:,1) = x(:,1)
    do i=2,7
      x_noz1(:,i) = x(:,i_array_sorted(i-1)+1)
    end do
    call ols(y,x_noz1,b_noz1,n,6,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    b(1,1) = b_noz1(1,1)
    j=2
    do i=2,p+1
      if(i_array(i-1)==777) then
        b(i,1) = 0.0D0
       else
        b(i_array(i-1)+1,1) = b_noz1(j,1)
        j=j+1
      end if
    end do
    write(25,*) 'Order of Xs removed from Eq:', z1
    write(25,*) 'Coef. b =', b
    if(.true.) exit outermost
  end if
  if(.true.) exit loop2
end do loop2
!
! z2 removed
!
loop3: do
  if(p<5) exit loop3
  if(p==5) then
    z1=999
    z2=999
  end if
  f0xxxx_noz1z2=f0xxxx
  if(z1/=999) then
    f0xxxx_noz1z2(z1,:,:,:,:) = -9.0D0
    f0xxxx_noz1z2(:,z1,:,:,:) = -9.0D0
    f0xxxx_noz1z2(:,:,z1,:,:) = -9.0D0
```

```
    f0xxxx_noz1z2(:,:,:,z1,:) = -9.0D0
    f0xxxx_noz1z2(:,:,:,:,z1) = -9.0D0
  end if
  if(z2/=999) then
    f0xxxx_noz1z2(z2,:,:,:,:) = -9.0D0
    f0xxxx_noz1z2(:,z2,:,:,:) = -9.0D0
    f0xxxx_noz1z2(:,:,z2,:,:) = -9.0D0
    f0xxxx_noz1z2(:,:,:,z2,:) = -9.0D0
    f0xxxx_noz1z2(:,:,:,:,z2) = -9.0D0
  end if
  f0xxxx_noz1z2_minl = minloc(f0xxxx_noz1z2, f0xxxx_noz1z2>=0.0D0)
  z3 = f0xxxx_noz1z2_minl(1)
  f0xxxx_noz1z2_minv = minval(f0xxxx_noz1z2, mask=f0xxxx_noz1z2>=0.0D0)
  if (f0xxxx_noz1z2_minv(1) > finv(1,n-6,0.90D0)) then
    do i=1,p
      if(i==z1.or.i==z2) then
        i_array(i) = 777
       else
        i_array(i) = i
      end if
    end do
    call sortinteger(i_array,i_array_sorted,p)
    x_noz1z2(:,1) = x(:,1)
    do i=2,6
      x_noz1z2(:,i) = x(:,i_array_sorted(i-1)+1)
    end do
    call ols(y,x_noz1z2,b_noz1z2,n,5,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    b(1,1) = b_noz1z2(1,1)
    j=2
    do i=2,p+1
      if(i_array(i-1)==777) then
        b(i,1) = 0.0D0
       else
        b(i_array(i-1)+1,1) = b_noz1z2(j,1)
        j=j+1
      end if
    end do
    write(25,*)'Order of Xs removed from Eq:', z1,z2
    write(25,*) 'Coef. b =', b
    if(.true.) exit outermost
  end if
  if(.true.) exit loop3
end do loop3
!
! z3 removed
!
loop4: do
  if(p<4) exit loop4
  if(p==4) then
    z1=999
    z2=999
    z3=999
  end if
  f0xxx_noz1z2z3 = f0xxx
  if(z1/=999) then
    f0xxx_noz1z2z3(z1,:,:,:) = -9.0D0
    f0xxx_noz1z2z3(:,z1,:,:) = -9.0D0
    f0xxx_noz1z2z3(:,:,z1,:) = -9.0D0
    f0xxx_noz1z2z3(:,:,:,z1) = -9.0D0
  end if
```

```
  if(z2/=999) then
    f0xxx_noz1z2z3(z2,:,:,:) = -9.0D0
    f0xxx_noz1z2z3(:,z2,:,:) = -9.0D0
    f0xxx_noz1z2z3(:,:,z2,:) = -9.0D0
    f0xxx_noz1z2z3(:,:,:,z2) = -9.0D0
  end if
  if(z3/=999) then
    f0xxx_noz1z2z3(z3,:,:,:) = -9.0D0
    f0xxx_noz1z2z3(:,z3,:,:) = -9.0D0
    f0xxx_noz1z2z3(:,:,z3,:) = -9.0D0
    f0xxx_noz1z2z3(:,:,:,z3) = -9.0D0
  end if
  f0xxx_noz1z2z3_minl = minloc(f0xxx_noz1z2z3,f0xxx_noz1z2z3>=0.0D0)
  z4 = f0xxx_noz1z2z3_minl(1)
  f0xxx_noz1z2z3_minv = minval(f0xxx_noz1z2z3,mask=f0xxx_noz1z2z3>=0.0D0)
  if(f0xxx_noz1z2z3_minv(1) > finv(1,n-5,0.90D0)) then
    do i=1,p
      if(i==z1.or.i==z2.or.i==z3) then
        i_array(i) = 777
       else
        i_array(i) = i
      end if
    end do
    call sortinteger(i_array,i_array_sorted,p)
    x_noz1z2z3(:,1) = x(:,1)
    do i=2,5
      x_noz1z2z3(:,i) = x(:,i_array_sorted(i-1)+1)
    end do
    call ols(y,x_noz1z2z3,b_noz1z2z3,n,4,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    b(1,1) = b_noz1z2z3(1,1)
    j=2
    do i=2,p+1
      if(i_array(i-1)==777) then
        b(i,1) = 0.0D0
       else
        b(i_array(i-1)+1,1) = b_noz1z2z3(j,1)
        j=j+1
      end if
    end do
    write(25,*)'Order of Xs removed from Eq:', z1,z2,z3
    write(25,*) 'Coef. b =', b
    if(.true.) exit outermost
  end if
  if(.true.) exit loop4
end do loop4
!
! z4 removed
!
loop5: do
  if(p<3) exit loop5
  if(p==3) then
    z1=999
    z2=999
    z3=999
    z4=999
  end if
    f0xx_noz1z2z3z4 = f0xx
    if(z1/=999) then
      f0xx_noz1z2z3z4(z1,:,:) = -9.0D0
      f0xx_noz1z2z3z4(:,z1,:) = -9.0D0
```

```
      f0xx_noz1z2z3z4(:,:,z1) = -9.0D0
    end if
    if(z2/=999) then
      f0xx_noz1z2z3z4(z2,:,:) = -9.0D0
      f0xx_noz1z2z3z4(:,z2,:) = -9.0D0
      f0xx_noz1z2z3z4(:,:,z2) = -9.0D0
    end if
    if(z3/=999) then
      f0xx_noz1z2z3z4(z3,:,:) = -9.0D0
      f0xx_noz1z2z3z4(:,z3,:) = -9.0D0
      f0xx_noz1z2z3z4(:,:,z3) = -9.0D0
    end if
    if(z4/=999) then
      f0xx_noz1z2z3z4(z4,:,:) = -9.0D0
      f0xx_noz1z2z3z4(:,z4,:) = -9.0D0
      f0xx_noz1z2z3z4(:,:,z4) = -9.0D0
    end if
    f0xx_noz1z2z3z4_minl = minloc(f0xx_noz1z2z3z4,f0xx_noz1z2z3z4>=0.0D0)
    z5 = f0xx_noz1z2z3z4_minl(1)
    f0xx_noz1z2z3z4_minv = minval(f0xx_noz1z2z3z4,mask=f0xx_noz1z2z3z4>=0.0D0)
    if(f0xx_noz1z2z3z4_minv(1) > finv(1,n-4,0.90D0)) then
      do i=1,p
        if(i==z1.or.i==z2.or.i==z3.or.i==z4) then
          i_array(i) = 777
         else
          i_array(i) = i
        end if
      end do
      call sortinteger(i_array,i_array_sorted,p)
      x_noz1z2z3z4(:,1) = x(:,1)
      do i=2,4
        x_noz1z2z3z4(:,i) = x(:,i_array_sorted(i-1)+1)
      end do
      call ols(y,x_noz1z2z3z4,b_noz1z2z3z4,n,3,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
      b(1,1) = b_noz1z2z3z4(1,1)
      j=2
      do i=2,p+1
        if(i_array(i-1)==777) then
          b(i,1) = 0.0D0
         else
          b(i_array(i-1)+1,1) = b_noz1z2z3z4(j,1)
          j=j+1
        end if
      end do
      write(25,*)'Order of Xs removed from Eq:', z1,z2,z3,z4
      write(25,*) 'Coef. b =', b
    if(.true.) exit outermost
  end if
  if(.true.) exit loop5
end do loop5
!
! z5 removed
!
loop6: do
  if(p<2) exit loop6
  if(p==2) then
    z1=999
    z2=999
    z3=999
    z4=999
```

```
    z5=999
  end if
  f0x_noz1z2z3z4z5 = f0x
  if(z1/=999) then
    f0x_noz1z2z3z4z5(z1,:) = -9.0D0
    f0x_noz1z2z3z4z5(:,z1) = -9.0D0
  end if
  if(z2/=999) then
    f0x_noz1z2z3z4z5(z2,:) = -9.0D0
    f0x_noz1z2z3z4z5(:,z2) = -9.0D0
  end if
  if(z3/=999) then
    f0x_noz1z2z3z4z5(z3,:) = -9.0D0
    f0x_noz1z2z3z4z5(:,z3) = -9.0D0
  end if
  if(z4/=999) then
    f0x_noz1z2z3z4z5(z4,:) = -9.0D0
    f0x_noz1z2z3z4z5(:,z4) = -9.0D0
  end if
  if(z5/=999) then
    f0x_noz1z2z3z4z5(z5,:) = -9.0D0
    f0x_noz1z2z3z4z5(:,z5) = -9.0D0
  end if
  f0x_noz1z2z3z4z5_minl = minloc(f0x_noz1z2z3z4z5,f0x_noz1z2z3z4z5>=0.0D0)
  z6 = f0x_noz1z2z3z4z5_minl(1)
  f0x_noz1z2z3z4z5_minv = minval(f0x_noz1z2z3z4z5,mask=f0x_noz1z2z3z4z5>=0.0D0)
  if(f0x_noz1z2z3z4z5_minv(1) > finv(1,n-3,0.90D0)) then
    do i=1,p
      if(i==z1.or.i==z2.or.i==z3.or.i==z4.or.i==z5) then
        i_array(i) = 777
       else
        i_array(i) = i
      end if
    end do
    call sortinteger(i_array,i_array_sorted,p)
    x_noz1z2z3z4z5(:,1) = x(:,1)
    do i=2,3
      x_noz1z2z3z4z5(:,i) = x(:,i_array_sorted(i-1)+1)
    end do
    call ols(y,x_noz1z2z3z4z5,b_noz1z2z3z4z5,n,2,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    b(1,1) = b_noz1z2z3z4z5(1,1)
    j=2
    do i=2,p+1
      if(i_array(i-1)==777) then
        b(i,1) = 0.0D0
       else
        b(i_array(i-1)+1,1) = b_noz1z2z3z4z5(j,1)
        j=j+1
      end if
    end do
    write(25,*)'Order of Xs removed from Eq:', z1,z2,z3,z4,z5
    write(25,*) 'Coef. b =', b
    if(.true.) exit outermost
  end if
  if(.true.) exit loop6
end do loop6
!
! z6 removed
!
loop7: do
```

```
    if(p<1) exit loop7
    if(p==1) then
      z1=999
      z2=999
      z3=999
      z4=999
      z5=999
      z6=999
    end if
    f0_noz1z2z3z4z5z6 = f0
    if(z1/=999) f0_noz1z2z3z4z5z6(z1) = -9.0D0
    if(z2/=999) f0_noz1z2z3z4z5z6(z2) = -9.0D0
    if(z3/=999) f0_noz1z2z3z4z5z6(z3) = -9.0D0
    if(z4/=999) f0_noz1z2z3z4z5z6(z4) = -9.0D0
    if(z5/=999) f0_noz1z2z3z4z5z6(z5) = -9.0D0
    if(z6/=999) f0_noz1z2z3z4z5z6(z6) = -9.0D0
    f0_noz1z2z3z4z5z6_minl = minloc(f0_noz1z2z3z4z5z6,f0_noz1z2z3z4z5z6>=0.0D0)
    z7 = f0_noz1z2z3z4z5z6_minl(1)
    f0_noz1z2z3z4z5z6_minv = minval(f0_noz1z2z3z4z5z6,mask=f0_noz1z2z3z4z5z6>=0.0D0)
    if(f0_noz1z2z3z4z5z6_minv(1) > finv(1,n-2,0.90D0)) then
      do i=1,p
        if(i==z1.or.i==z2.or.i==z3.or.i==z4.or.i==z5.or.i==z6) then
          i_array(i) = 777
         else
          i_array(i) = i
        end if
      end do
      call sortinteger(i_array,i_array_sorted,p)
      x_noz1z2z3z4z5z6(:,1) = x(:,1)
      do i=2,2
        x_noz1z2z3z4z5z6(:,i) = x(:,i_array_sorted(i-1)+1)
      end do
      call ols(y,x_noz1z2z3z4z5z6,b_noz1z2z3z4z5z6,n,1,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
      b(1,1) = b_noz1z2z3z4z5z6(1,1)
      j=2
      do i=2,p+1
        if(i_array(i-1)==777) then
          b(i,1) = 0.0D0
         else
          b(i_array(i-1)+1,1) = b_noz1z2z3z4z5z6(j,1)
          j=j+1
        end if
      end do
      write(25,*)'Order of Xs removed from Eq:', z1,z2,z3,z4,z5,z6
      write(25,*) 'Coef. b =', b
      if(.true.) exit outermost
    end if
    if(.true.) exit loop7
  end do loop7
  !
  ! z7 removed
  !
  b(1,1) = sum(y)/dble(n)
  do i=2,p+1
    b(i,1) = 0.0D0
  end do
  write(25,*) 'Order of Xs removed from Eq:', z1,z2,z3,z4,z5,z6,z7
  write(25,*) 'Coef. b =', b
  if(.true.) exit outermost
end do outermost
```

```
return
end subroutine backward
!
!
!
subroutine sortinteger(array_orig,array_sorted,m)
! Integers to be sorted must be less than 9999999
implicit none
integer:: i,m
integer:: array_orig(m), array_interm(m), array_sorted(m), array_interm_minv(1), array_interm_minl(1)
array_interm = array_orig
do i=1,m
  array_interm_minv = minval(array_interm)
  array_interm_minl = minloc(array_interm)
  array_sorted(i) = array_interm_minv(1)
  array_interm(array_interm_minl(1)) = 9999999
end do
return
end subroutine sortinteger
!
!
!
subroutine stepwise(y,x,b,n,p,f0,f0x,f0xx,f0xxx,f0xxxx,f0xxxxx,f0xxxxxx)
implicit none
integer::
n,p,i,z1,z2,z3,z4,z5,z6,z7,za,zb,zc,zd,ze,zf,zg,zh,zi,zj,zk,zl,zm,zn,zo,zp,zq,zr,zs,zt,zu,zv,zw,zx,zy,zz,za
1
double precision:: y(n,1), x(n,p+1), b(p+1,1), r(p), abs_r(p), bz1(2,1), bzazb(3,1), bzczdze(4,1),
bzfzgzhzi(5,1), bzjzkzlzmzn(6,1), bzozpzqzrzszt(7,1), xz1(n,2), xzazb(n,3), xzczdze(n,4),
xzfzgzhzi(n,5), xzjzkzlzmzn(n,6), xzozpzqzrzszt(n,7), yhat(n,1), e(n,1),f0(p), f0x(p,p), f0xx(p,p,p),
f0xxx(p,p,p,p), f0xxxx(p,p,p,p,p), f0xxxxx(p,p,p,p,p,p), f0xxxxxx(p,p,p,p,p,p,p)
double precision::
ss_reg,ms_reg,ss_res,ms_res,f_cal,finv,corr,fza_zb,fzc_zdze,fzf_zgzhzi,fzj_zkzlzmzn,fzo_zpzqzrzszt,f
zu_zvzwzxzyzzza1
integer:: abs_r_maxl(1), f0x_maxl(1), f0xx_maxl(1), f0xxx_maxl(1), f0xxxx_maxl(1),
f0xxxxx_maxl(1), f0xxxxxx_maxl(1)
outer1: do
  outer2: do
    !
    ! STEP 1: Seek z1
    !
    do i=1,p
      r(i) = corr(x(:,i+1), y(:,1),n)
    end do
    abs_r = abs(r)
    abs_r_maxl = maxloc(abs_r)
    z1 = abs_r_maxl(1)
    !
    ! STEP 2: Check z1
    !
    loop1: do
      if (f0(z1) <= finv(1,n-2,0.95D0)) then
        !
        ! z1 is not entered into Eq.
        !
        b(1,1) = sum(y)/dble(n)
        do i=2,p+1
          b(i,1) = 0.0D0
        end do
        write(25,*) 'No Xs are in Eq.'
```

```
 write(25,*) 'Coeff. b =', b
 if(.true.) exit outer1
else
 !
 ! Enter z1 into Eq.
 !
 write(25,*) 'First y = f(_):', z1
 if(p<2) exit outer2
 !
 ! STEP 3: Seek z2
 !
 f0x_maxl = maxloc(f0x(:,z1))
 z2 = f0x_maxl(1)
 if(f0x(z2,z1)<=finv(1,n-3,0.95D0)) then
   !
   ! Only z1 will be in Eq.
   !
   xz1(:,1) = x(:,1)
   xz1(:,2) = x(:,z1+1)
   call ols(y,xz1,bz1,n,1,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
   b(1,1) = bz1(1,1)
   b(z1+1,1) = bz1(2,1)
   do i=1,p+1
     if(i==1.or.i==z1+1) cycle
     b(i,1) = 0.0D0
   end do
   write(25,*) 'Coeff. b =', b
   if(.true.) exit outer1
  else
   !
   ! Enter z2
   !
   !write(25,*) 'Next y = f(_,_):', z1,z2
   !
   ! STEP 4: Check exit
   !
   loop2: do
     if(f0x(z1,z2)<f0x(z2,z1)) then
       za=z1
       zb=z2
      else
       za=z2
       zb=z1
     end if
     fza_zb=f0x(za,zb)
     if (fza_zb<=finv(1,n-3,0.90D0)) then
       !
       ! Remove za
       !
       write(25,*) 'Next y = f(_):', zb
       z1=zb
       !
       ! Back to STEP 2
       !
       if(.true.) cycle loop1
      else
       !
       ! Retain za (and zb)
       !
       write(25,*) 'Next y = f(_,_):', za,zb
```

```
if(p<3) exit outer2
!
! STEP 5: Seek z3
!
f0xx_maxl = maxloc(f0xx(:,za,zb))
z3 = f0xx_maxl(1)
if(f0xx(z3,za,zb)<=finv(1,n-4,0.95D0)) then
  !
  ! Only za, zb will be in Eq.
  !
  xzazb(:,1) = x(:,1)
  xzazb(:,2) = x(:,za+1)
  xzazb(:,3) = x(:,zb+1)
  call ols(y,xzazb,bzazb,n,2,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
  b(1,1) = bzazb(1,1)
  b(za+1,1) = bzazb(2,1)
  b(zb+1,1) = bzazb(3,1)
  do i=1, p+1
    if(i==1.or.i==za+1.or.i==zb+1) cycle
    b(i,1) = 0.0D0
  end do
  write(25,*) 'Coef. b =', b
  if(.true.) exit outer1
 else
  !
  ! Enter z3
  !
  write(25,*) 'Next y = f(_,_,_):', za,zb,z3
  !
  ! STEP 6: Check exit
  !
  loop3: do
    if(f0xx(za,zb,z3)==min(f0xx(za,zb,z3),f0xx(zb,za,z3),f0xx(z3,za,zb))) then
      zc=za
      zd=zb
      ze=z3
     else if(f0xx(zb,za,z3)==min(f0xx(za,zb,z3),f0xx(zb,za,z3),f0xx(z3,za,zb))) then
      zc=zb
      zd=za
      ze=z3
     else if(f0xx(z3,za,zb)==min(f0xx(za,zb,z3),f0xx(zb,za,z3),f0xx(z3,za,zb))) then
      zc=z3
      zd=za
      ze=zb
    end if
    fzc_zdze=f0xx(zc,zd,ze)
    if(fzc_zdze <=finv(1,n-4,0.90D0)) then
      !
      ! Remove zc
      !
      write(25,*) 'Next y = f(_,_):', zd,ze
      z1=zd
      z2=ze
      !
      ! Back to STEP 4
      !
      if(.true.) cycle loop2
     else
      !
      ! Retain zc (and zd,ze)
```

```
!
write(25,*) 'Next y = f(_,_,_):', zc,zd,ze
if(p<4) exit outer2
!
! STEP 7: Seek z4
!
f0xxx_maxl=maxloc(f0xxx(:,zc,zd,ze))
z4=f0xxx_maxl(1)
if(f0xxx(z4,zc,zd,ze)<=finv(1,n-5,0.95D0)) then
  !
  ! Only zc, zd, ze will be in Eq.
  !
  xzczdze(:,1) = x(:,1)
  xzczdze(:,2) = x(:,zc+1)
  xzczdze(:,3) = x(:,zd+1)
  xzczdze(:,4) = x(:,ze+1)
  call ols(y,xzczdze,bzczdze,n,3,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
  b(1,1) = bzczdze(1,1)
  b(zc+1,1) = bzczdze(2,1)
  b(zd+1,1) = bzczdze(3,1)
  b(ze+1,1) = bzczdze(4,1)
  do i=1,p+1
    if(i==1.or.i==zc+1.or.i==zd+1.or.i==ze+1) cycle
    b(i,1) = 0.0D0
  end do
  write(25,*) 'Coef. b =', b
  if(.true.) exit outer1
 else
  !
  ! Enter z4
  !
  write(25,*) 'Next y = f(_,_,_,_):', zc,zd,ze,z4
  !
  ! STEP 8: Check exit
  !
  loop4: do

if(f0xxx(z4,zc,zd,ze)==min(f0xxx(z4,zc,zd,ze),f0xxx(zc,z4,zd,ze),f0xxx(zd,z4,zc,ze),f0xxx(ze,z4,zc,zd
))) then
      zf=z4
      zg=zc
      zh=zd
      zi=ze
     else
if(f0xxx(zc,z4,zd,ze)==min(f0xxx(z4,zc,zd,ze),f0xxx(zc,z4,zd,ze),f0xxx(zd,z4,zc,ze),f0xxx(ze,z4,zc,zd
))) then
      zf=zc
      zg=z4
      zh=zd
      zi=ze
     else
if(f0xxx(zd,z4,zc,ze)==min(f0xxx(z4,zc,zd,ze),f0xxx(zc,z4,zd,ze),f0xxx(zd,z4,zc,ze),f0xxx(ze,z4,zc,zd
))) then
      zf=zd
      zg=z4
      zh=zc
      zi=ze
     else
if(f0xxx(ze,z4,zc,zd)==min(f0xxx(z4,zc,zd,ze),f0xxx(zc,z4,zd,ze),f0xxx(zd,z4,zc,ze),f0xxx(ze,z4,zc,zd
))) then
```

```
  zf=ze
  zg=z4
  zh=zc
  zi=zd
end if
fzf_zgzhzi=f0xxx(zf,zg,zh,zi)
if(fzf_zgzhzi<=finv(1,n-5,0.90D0)) then
  !
  ! Remove zf
  !
  write(25,*) 'Next y = f(_,_,_):', zg,zh,zi
  za=zg
  zb=zh
  z3=zi
  !
  ! Back to STEP 6
  !
  if(.true.) cycle loop3
 else
  !
  ! Retain zf (and zg,zh,zi)
  !
  write(25,*) 'Next y = (_,_,_,_):', zf,zg,zh,zi
  if(p<5) exit outer2
  !
  ! STEP 9: Seek z5
  !
  f0xxxx_maxl=maxloc(f0xxxx(:,zf,zg,zh,zi))
  z5=f0xxxx_maxl(1)
  if(f0xxxx(z5,zf,zg,zh,zi)<=finv(1,n-6,0.95D0)) then
    !
    ! Only zf, zg, zh, zi will be in Eq.
    !
    xzfzgzhzi(:,1) = x(:,1)
    xzfzgzhzi(:,2) = x(:,zf+1)
    xzfzgzhzi(:,3) = x(:,zg+1)
    xzfzgzhzi(:,4) = x(:,zh+1)
    xzfzgzhzi(:,5) = x(:,zi+1)
    call ols(y,xzfzgzhzi,bzfzgzhzi,n,4,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    b(1,1) = bzfzgzhzi(1,1)
    b(zf+1,1) = bzfzgzhzi(2,1)
    b(zg+1,1) = bzfzgzhzi(3,1)
    b(zh+1,1) = bzfzgzhzi(4,1)
    b(zi+1,1) = bzfzgzhzi(5,1)
    do i=1,p+1
      if(i==1.or.i==zf+1.or.i==zg+1.or.i==zh+1.or.i==zi+1) cycle
      b(i,1) = 0.0D0
    end do
    write(25,*) 'Coef. b =', b
    if(.true.) exit outer1
   else
    !
    ! Enter z5
    !
    write(25,*) 'Next y = f(_,_,_,_):', zf,zg,zh,zi,z5
    !
    ! STEP 10: Check exit
    !
    loop5: do
```

```
if(f0xxxx(z5,zf,zg,zh,zi)==min(f0xxxx(z5,zf,zg,zh,zi),f0xxxx(zf,z5,zg,zh,zi),f0xxxx(zg,z5,zf,zh,zi),f0x
xxx(zh,z5,zf,zg,zi),f0xxxx(zi,z5,zf,zg,zh))) then
     zj=z5
     zk=zf
     zl=zg
     zm=zh
     zn=zi
    else
if(f0xxxx(zf,z5,zg,zh,zi)==min(f0xxxx(z5,zf,zg,zh,zi),f0xxxx(zf,z5,zg,zh,zi),f0xxxx(zg,z5,zf,zh,zi),f0x
xxx(zh,z5,zf,zg,zi),f0xxxx(zi,z5,zf,zg,zh))) then
     zj=zf
     zk=z5
     zl=zg
     zm=zh
     zn=zi
    else
if(f0xxxx(zg,z5,zf,zh,zi)==min(f0xxxx(z5,zf,zg,zh,zi),f0xxxx(zf,z5,zg,zh,zi),f0xxxx(zg,z5,zf,zh,zi),f0x
xxx(zh,z5,zf,zg,zi),f0xxxx(zi,z5,zf,zg,zh))) then
     zj=zg
     zk=z5
     zl=zf
     zm=zh
     zn=zi
    else
if(f0xxxx(zh,z5,zf,zg,zi)==min(f0xxxx(z5,zf,zg,zh,zi),f0xxxx(zf,z5,zg,zh,zi),f0xxxx(zg,z5,zf,zh,zi),f0x
xxx(zh,z5,zf,zg,zi),f0xxxx(zi,z5,zf,zg,zh))) then
     zj=zh
     zk=z5
     zl=zf
     zm=zg
     zn=zi
    else
if(f0xxxx(zi,z5,zf,zg,zh)==min(f0xxxx(z5,zf,zg,zh,zi),f0xxxx(zf,z5,zg,zh,zi),f0xxxx(zg,z5,zf,zh,zi),f0x
xxx(zh,z5,zf,zg,zi),f0xxxx(zi,z5,zf,zg,zh))) then
     zj=zi
     zk=z5
     zl=zf
     zm=zg
     zn=zh
   end if
   fzj_zkzlzmzn=f0xxxx(zj,zk,zl,zm,zn)
   if(fzj_zkzlzmzn<=finv(1,n-6,0.90D0)) then
     !
     ! Remove zj
     !
     write(25,*) 'Next y = f(_,_,_,_):', zk,zl,zm,zn
     zc=zk
     zd=zl
     ze=zm
     z4=zn
     !
     ! Back to STEP 8
     !
     if(.true.) cycle loop4
    else
     !
     ! Retain zj (and zk,zl,zm,zn)
     !
     !write(25,*) 'Next y = (_,_,_,_,_):', zj,zk,zl,zm,zn
```

```
                                        if(p<6) exit outer2
                                        !
                                        ! STEP 11: Seek z6
                                        !
                                        f0xxxxx_maxl=maxloc(f0xxxxx(:,zj,zk,zl,zm,zn))
                                        z6=f0xxxxx_maxl(1)
                                        if(f0xxxxx(z6,zj,zk,zl,zm,zn)<=finv(1,n-7,0.95D0)) then
                                          !
                                          ! Only zj, zk, zl, zm, zn will be in Eq.
                                          !
                                          xzjzkzlzmzn(:,1) = x(:,1)
                                          xzjzkzlzmzn(:,2) = x(:,zj+1)
                                          xzjzkzlzmzn(:,3) = x(:,zk+1)
                                          xzjzkzlzmzn(:,4) = x(:,zl+1)
                                          xzjzkzlzmzn(:,5) = x(:,zm+1)
                                          xzjzkzlzmzn(:,6) = x(:,zn+1)
                                          call
ols(y,xzjzkzlzmzn,bzjzkzlzmzn,n,5,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
                                          b(1,1) = bzjzkzlzmzn(1,1)
                                          b(zj+1,1) = bzjzkzlzmzn(2,1)
                                          b(zk+1,1) = bzjzkzlzmzn(3,1)
                                          b(zl+1,1) = bzjzkzlzmzn(4,1)
                                          b(zm+1,1) = bzjzkzlzmzn(5,1)
                                          b(zn+1,1) = bzjzkzlzmzn(6,1)
                                          do i=1,p+1
                                            if(i==1.or.i==zj+1.or.i==zk+1.or.i==zl+1.or.i==zm+1.or.i==zn+1)
cycle
                                            b(i,1) = 0.0D0
                                          end do
                                          write(25,*) 'Coef. b =', b
                                          if(.true.) exit outer1
                                         else
                                          !
                                          ! Enter z6
                                          !
                                          write(25,*) 'Next y = f(_,_,_,_,_,_):', zj,zk,zl,zm,zn,z6
                                          !
                                          ! STEP 12: Check exit
                                          !
                                          loop6: do

if(f0xxxxx(z6,zj,zk,zl,zm,zn)==min(f0xxxxx(z6,zj,zk,zl,zm,zn),f0xxxxx(zj,z6,zk,zl,zm,zn),f0xxxxx(zk
,z6,zj,zl,zm,zn),f0xxxxx(zl,z6,zj,zk,zm,zn), f0xxxxx(zm,z6,zj,zk,zl,zn), f0xxxxx(zn,z6,zj,zk,zl,zm)))
then
                                              zo=z6
                                              zp=zj
                                              zq=zk
                                              zr=zl
                                              zs=zm
                                              zt=zn
                                             else
if(f0xxxxx(zj,z6,zk,zl,zm,zn)==min(f0xxxxx(z6,zj,zk,zl,zm,zn),f0xxxxx(zj,z6,zk,zl,zm,zn),f0xxxxx(zk
,z6,zj,zl,zm,zn),f0xxxxx(zl,z6,zj,zk,zm,zn),f0xxxxx(zm,z6,zj,zk,zl,zn), f0xxxxx(zn,z6,zj,zk,zl,zm)))
then
                                              zo=zj
                                              zp=z6
                                              zq=zk
                                              zr=zl
                                              zs=zm
                                              zt=zn
```

```
                                    else
if(f0xxxxx(zk,z6,zj,zl,zm,zn)==min(f0xxxxx(z6,zj,zk,zl,zm,zn),f0xxxxx(zj,z6,zk,zl,zm,zn),f0xxxxx(zk
,z6,zj,zl,zm,zn),f0xxxxx(zl,z6,zj,zk,zm,zn),f0xxxxx(zm,z6,zj,zk,zl,zn), f0xxxxx(zn,z6,zj,zk,zl,zm)))
then
                                     zo=zk
                                     zp=z6
                                     zq=zj
                                     zr=zl
                                     zs=zm
                                     zt=zn
                                    else
if(f0xxxxx(zl,z6,zj,zk,zm,zn)==min(f0xxxxx(z6,zj,zk,zl,zm,zn),f0xxxxx(zj,z6,zk,zl,zm,zn),f0xxxxx(zk
,z6,zj,zl,zm,zn),f0xxxxx(zl,z6,zj,zk,zm,zn),f0xxxxx(zm,z6,zj,zk,zl,zn), f0xxxxx(zn,z6,zj,zk,zl,zm)))
then
                                     zo=zl
                                     zp=z6
                                     zq=zj
                                     zr=zk
                                     zs=zm
                                     zt=zn
                                    else
if(f0xxxxx(zm,z6,zj,zk,zl,zn)==min(f0xxxxx(z6,zj,zk,zl,zm,zn),f0xxxxx(zj,z6,zk,zl,zm,zn),f0xxxxx(zk
,z6,zj,zl,zm,zn),f0xxxxx(zl,z6,zj,zk,zm,zn),f0xxxxx(zm,z6,zj,zk,zl,zn), f0xxxxx(zn,z6,zj,zk,zl,zm)))
then
                                     zo=zm
                                     zp=z6
                                     zq=zj
                                     zr=zk
                                     zs=zl
                                     zt=zn
                                    else
if(f0xxxxx(zn,z6,zj,zk,zl,zm)==min(f0xxxxx(z6,zj,zk,zl,zm,zn),f0xxxxx(zj,z6,zk,zl,zm,zn),f0xxxxx(zk
,z6,zj,zl,zm,zn),f0xxxxx(zl,z6,zj,zk,zm,zn),f0xxxxx(zm,z6,zj,zk,zl,zn), f0xxxxx(zn,z6,zj,zk,zl,zm)))
then
                                     zo=zn
                                     zp=z6
                                     zq=zj
                                     zr=zk
                                     zs=zl
                                     zt=zm
                                   end if
                                   fzo_zpzqzrzszt=f0xxxxx(zo,zp,zq,zr,zs,zt)
                                   if(fzo_zpzqzrzszt<=finv(1,n-7,0.90D0)) then
                                     !
                                     ! Remove zo
                                     !
                                     !write(25,*) 'Next y = f(_,_,_,_,_):', zp,zq,zr,zs,zt
                                     zf=zp
                                     zg=zq
                                     zh=zr
                                     zi=zs
                                     z5=zt
                                     !
                                     ! Back to STEP 10
                                     !
                                     if(.true.) cycle loop5
                                    else
                                     !
                                     ! Retain zo (and zp,zq,zr,zs,zt)
                                     !
```

```
                    !write(25,*) 'Next y = (_,_,_,_,_,_):', zo,zp,zq,zr,zs,zt
                    if(p<7) exit outer2
                    !
                    ! STEP 13: Seek z7
                    !
                    f0xxxxxx_maxl=maxloc(f0xxxxxx(:,zo,zp,zq,zr,zs,zt))
                    z7=f0xxxxxx_maxl(1)
                    if(f0xxxxxx(z7,zo,zp,zq,zr,zs,zt)<=finv(1,n-8,0.95D0)) then
                     !
                     ! Only zo, zp, zq, zr, zs, zt will be in Eq.
                     !
                     xzozpzqzrzszt(:,1) = x(:,1)
                     xzozpzqzrzszt(:,2) = x(:,zo+1)
                     xzozpzqzrzszt(:,3) = x(:,zp+1)
                     xzozpzqzrzszt(:,4) = x(:,zq+1)
                     xzozpzqzrzszt(:,5) = x(:,zr+1)
                     xzozpzqzrzszt(:,6) = x(:,zs+1)
                     xzozpzqzrzszt(:,7) = x(:,zt+1)
                     call
ols(y,xzozpzqzrzszt,bzozpzqzrzszt,n,6,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
                     b(1,1) = bzozpzqzrzszt(1,1)
                     b(zo+1,1) = bzozpzqzrzszt(2,1)
                     b(zp+1,1) = bzozpzqzrzszt(3,1)
                     b(zq+1,1) = bzozpzqzrzszt(4,1)
                     b(zr+1,1) = bzozpzqzrzszt(5,1)
                     b(zs+1,1) = bzozpzqzrzszt(6,1)
                     b(zt+1,1) = bzozpzqzrzszt(7,1)
                     do i=1,p+1

if(i==1.or.i==zo+1.or.i==zp+1.or.i==zq+1.or.i==zr+1.or.i==zs+1.or.i==zt+1) cycle
                       b(i,1) = 0.0D0
                     end do
                     write(25,*) 'Coef. b =', b
                     if(.true.) exit outer1
                    else
                     !
                     ! Enter z7
                     !
                     !write(25,*) 'Next y = f(_,_,_,_,_,_,_):', zo, zp, zq, zr, zs, zt, z7
                     !
                     ! STEP 14: Check exit
                     !
                     loop7: do

if(f0xxxxxx(z7,zo,zp,zq,zr,zs,zt)==min(f0xxxxxx(z7,zo,zp,zq,zr,zs,zt),f0xxxxxx(zo,z7,zp,zq,zr,zs,zt),f
0xxxxxx(zp,z7,zo,zq,zr,zs,zt),f0xxxxxx(zq,z7,zo,zp,zr,zs,zt), f0xxxxxx(zr,z7,zo,zp,zq,zs,zt),
f0xxxxxx(zs,z7,zo,zp,zq,zr,zt), f0xxxxxx(zt,z7,zo,zp,zq,zr,zs))) then
                       zu=z7
                       zv=zo
                       zw=zp
                       zx=zq
                       zy=zr
                       zz=zs
                       za1=zt
                      else
if(f0xxxxxx(zo,z7,zp,zq,zr,zs,zt)==min(f0xxxxxx(z7,zo,zp,zq,zr,zs,zt),f0xxxxxx(zo,z7,zp,zq,zr,zs,zt),f
0xxxxxx(zp,z7,zo,zq,zr,zs,zt),f0xxxxxx(zq,z7,zo,zp,zr,zs,zt), f0xxxxxx(zr,z7,zo,zp,zq,zs,zt),
f0xxxxxx(zs,z7,zo,zp,zq,zr,zt), f0xxxxxx(zt,z7,zo,zp,zq,zr,zs))) then
                       zu=zo
                       zv=z7
```

```
zw=zp
zx=zq
zy=zr
zz=zs
za1=zt
else
if(f0xxxxxx(zp,z7,zo,zq,zr,zs,zt)==min(f0xxxxxx(z7,zo,zp,zq,zr,zs,zt),f0xxxxxx(zo,z7,zp,zq,zr,zs,zt),f
0xxxxxx(zp,z7,zo,zq,zr,zs,zt),f0xxxxxx(zq,z7,zo,zp,zr,zs,zt), f0xxxxxx(zr,z7,zo,zp,zq,zs,zt),
f0xxxxxx(zs,z7,zo,zp,zq,zr,zt), f0xxxxxx(zt,z7,zo,zp,zq,zr,zs))) then
zu=zp
zv=z7
zw=zo
zx=zq
zy=zr
zz=zs
za1=zt
else
if(f0xxxxxx(zq,z7,zo,zp,zr,zs,zt)==min(f0xxxxxx(z7,zo,zp,zq,zr,zs,zt),f0xxxxxx(zo,z7,zp,zq,zr,zs,zt),f
0xxxxxx(zp,z7,zo,zq,zr,zs,zt),f0xxxxxx(zq,z7,zo,zp,zr,zs,zt), f0xxxxxx(zr,z7,zo,zp,zq,zs,zt),
f0xxxxxx(zs,z7,zo,zp,zq,zr,zt), f0xxxxxx(zt,z7,zo,zp,zq,zr,zs))) then
zu=zq
zv=z7
zw=zo
zx=zp
zy=zr
zz=zs
za1=zt
else
if(f0xxxxxx(zr,z7,zo,zp,zq,zs,zt)==min(f0xxxxxx(z7,zo,zp,zq,zr,zs,zt),f0xxxxxx(zo,z7,zp,zq,zr,zs,zt),f
0xxxxxx(zp,z7,zo,zq,zr,zs,zt),f0xxxxxx(zq,z7,zo,zp,zr,zs,zt), f0xxxxxx(zr,z7,zo,zp,zq,zs,zt),
f0xxxxxx(zs,z7,zo,zp,zq,zr,zt), f0xxxxxx(zt,z7,zo,zp,zq,zr,zs))) then
zu=zr
zv=z7
zw=zo
zx=zp
zy=zq
zz=zs
za1=zt
else
if(f0xxxxxx(zs,z7,zo,zp,zq,zr,zt)==min(f0xxxxxx(z7,zo,zp,zq,zr,zs,zt),f0xxxxxx(zo,z7,zp,zq,zr,zs,zt),f
0xxxxxx(zp,z7,zo,zq,zr,zs,zt),f0xxxxxx(zq,z7,zo,zp,zr,zs,zt), f0xxxxxx(zr,z7,zo,zp,zq,zs,zt),
f0xxxxxx(zs,z7,zo,zp,zq,zr,zt), f0xxxxxx(zt,z7,zo,zp,zq,zr,zs))) then
zu=zs
zv=z7
zw=zo
zx=zp
zy=zq
zz=zr
za1=zt
else
if(f0xxxxxx(zt,z7,zo,zp,zq,zr,zs)==min(f0xxxxxx(z7,zo,zp,zq,zr,zs,zt),f0xxxxxx(zo,z7,zp,zq,zr,zs,zt),f
0xxxxxx(zp,z7,zo,zq,zr,zs,zt),f0xxxxxx(zq,z7,zo,zp,zr,zs,zt), f0xxxxxx(zr,z7,zo,zp,zq,zs,zt),
f0xxxxxx(zs,z7,zo,zp,zq,zr,zt), f0xxxxxx(zt,z7,zo,zp,zq,zr,zs))) then
zu=zt
zv=z7
zw=zo
zx=zp
zy=zq
zz=zr
za1=zs
```

```
                                          end if
                                          fzu_zvzwzxzyzzza1=f0xxxxxx(zu,zv,zw,zx,zy,zz,za1)
                                          if(fzu_zvzwzxzyzzza1<=finv(1,n-8,0.90D0)) then
                                            !
                                            ! Remove zu
                                            !
                                            !write(25,*) 'Next y = f(_,_,_,_,_,_):', zv, zw, zx, zy, zz, za1
                                            zj=zv
                                            zk=zw
                                            zl=zx
                                            zm=zy
                                            zn=zz
                                            z6=za1
                                            !
                                            ! Back to STEP 12
                                            !
                                            if(.true.) cycle loop6
                                           else
                                            !
                                            ! Retain zu (and zv,zw,zx,zy,zz,za1)
                                            !
                                            !write(25,*) 'Next y = (_,_,_,_,_,_,_):', zu,zv,zw,zx,zy,zz,za1
                                            if(p<8) exit outer2
                                          end if
                                        end do loop7
                                      end if
                                    end if
                                  end do loop6
                                end if
                              end if
                            end do loop5
                          end if
                        end if
                      end do loop4
                    end if
                  end if
                end do loop3
              end if
            end if
          end do loop2
        end if
      end if
    end do loop1
  end do outer2
  !
  ! Regression with all X's
  !
  call ols(y,x,b,n,p,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
  write(25,*) 'Coef. b =', b
  if(.true.) exit outer1
end do outer1
return
end subroutine stepwise
!
!
!
function corr(u,w,n)
implicit none
integer:: n,i
double precision:: corr,ubar,wbar,a,b,c,suma,sumb,sumc
```

```
double precision:: u(n), w(n)
ubar = sum(u)/dble(n)
wbar = sum(w)/dble(n)
suma = 0.0D0
do i=1,n
   a = (u(i)-ubar)*(w(i)-wbar)
   suma = suma + a
end do
!
sumb = 0.0D0
do i=1,n
   b = (u(i)-ubar)**2.0D0
   sumb = sumb + b
end do
!
sumc = 0.0D0
do i=1,n
   c = (w(i)-wbar)**2.0D0
   sumc = sumc + c
end do
!
corr = suma/(sqrt(sumb)*sqrt(sumc))
return
end function corr
!
!
!
function finv(v1,v2,prob)
implicit none
double precision:: finv,prob
double precision:: array1(100,15)
integer:: i,j,v1,v2
if(prob==0.90D0) then
   open(17,file='C:/Documents and Settings/tee/Desktop/small_tee_thesis/finv90.txt')
   read(17,*) ((array1(i,j),j=1,15),i=1,100)
   finv=array1(v2,v1)
   close(17)
  else if(prob==0.95D0) then
   open(18,file='C:/Documents and Settings/tee/Desktop/small_tee_thesis/finv95.txt')
   read(18,*) ((array1(i,j),j=1,15),i=1,100)
   finv=array1(v2,v1)
   close(18)
  else if(prob==0.99D0) then
   open(19,file='C:/Documents and Settings/tee/Desktop/small_tee_thesis/finv99.txt')
   read(19,*) ((array1(i,j),j=1,15),i=1,100)
   finv=array1(v2,v1)
   close(19)
  else
   write(25,*) 'finv(',v1,',',v2,') not available'
end if
return
end function finv
!
!
!
subroutine partial_f(y,x,n,p,f0,f0x,f0xx,f0xxx,f0xxxx,f0xxxxx,f0xxxxxx)
implicit none
integer:: n,p,i,j,k,m,q,r,s
double precision:: y(n,1), x(n,p+1), yhat(n,1), e(n,1), z0(n,2), b0(2,1), ss0(p), f0(p), z0x(n,3), b0x(3,1),
ss0x(p,p), f0x(p,p), z0xx(n,4), b0xx(4,1), ss0xx(p,p,p), f0xx(p,p,p), z0xxx(n,5), b0xxx(5,1),
```

```
ss0xxx(p,p,p,p), f0xxx(p,p,p,p), z0xxxx(n,6), b0xxxx(6,1), ss0xxxx(p,p,p,p,p), f0xxxx(p,p,p,p,p),
z0xxxxx(n,7), b0xxxxx(7,1), ss0xxxxx(p,p,p,p,p,p), f0xxxxx(p,p,p,p,p,p), z0xxxxxx(n,8),
b0xxxxxx(8,1), ss0xxxxxx(p,p,p,p,p,p,p), f0xxxxxx(p,p,p,p,p,p,p)
double precision:: ss_reg,ms_reg,ss_res,ms_res,f_cal
outermost: do
  !
  ! Calculate F( )|-
  !
  z0(:,1) = x(:,1)
  do i=1,p
    z0(:,2) = x(:,i+1)
    call ols(y,z0,b0,n,1,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
    ss0(i) = ss_reg
    f0(i) = f_cal
  end do
  if(p<2) exit outermost
  !
  ! Calculate F()|()
  !
  z0x(:,1)=x(:,1)
  do j=1,p
    z0x(:,2)=x(:,j+1)
    ss0x(j,j) = -9.0D0
    f0x(j,j) = -9.0D0
    do i=1,p
      if(i==j) cycle
      z0x(:,3) = x(:, i+1)
      call ols(y,z0x,b0x,n,2,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
      ss0x(i,j) = ss_reg - ss0(j)
      f0x(i,j) = ss0x(i,j)/(ss_res/dble(n-3))
    end do
  end do
  if(p<3) exit outermost
  !
  ! Calculate F()|()()
  !
  z0xx(:,1) = x(:,1)
  do j=1,p
    z0xx(:,2) = x(:,j+1)
    ss0xx(:,j,j) = -9.0D0
    f0xx(:,j,j) = -9.0D0
    do k=1,p
      if(k==j) cycle
      z0xx(:,3) = x(:,k+1)
      ss0xx(j,j,k) = -9.0D0
      ss0xx(k,j,k) = -9.0D0
      f0xx(j,j,k) = -9.0D0
      f0xx(k,j,k) = -9.0D0
      do i=1,p
        if(i==j.or.i==k) cycle
          z0xx(:,4) = x(:,i+1)
          call ols(y,z0xx,b0xx,n,3,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
          ss0xx(i,j,k) = ss_reg - ss0(j) - ss0x(k,j)
          f0xx(i,j,k) = ss0xx(i,j,k)/(ss_res/dble(n-4))
      end do
    end do
  end do
  if(p<4) exit outermost
  !
  ! Calculate F()|()()()
```

```
!
z0xxx(:,1) = x(:,1)
do j=1,p
  z0xxx(:,2) = x(:,j+1)
  ss0xxx(:,j,j,:) = -9.0D0
  f0xxx(:,j,j,:) = -9.0D0
  do k=1,p
    if (k==j) cycle
    z0xxx(:,3) = x(:,k+1)
    ss0xxx(:,j,k,j) = -9.0D0
    ss0xxx(:,j,k,k) = -9.0D0
    f0xxx(:,j,k,j) = -9.0D0
    f0xxx(:,j,k,k) = -9.0D0
    do m=1,p
      if(m==j.or.m==k) cycle
      z0xxx(:,4)=x(:,m+1)
      ss0xxx(j,j,k,m) = -9.0D0
      ss0xxx(k,j,k,m) = -9.0D0
      ss0xxx(m,j,k,m) = -9.0D0
      f0xxx(j,j,k,m) = -9.0D0
      f0xxx(k,j,k,m) = -9.0D0
      f0xxx(m,j,k,m) = -9.0D0
      do i=1,p
        if(i==j.or.i==k.or.i==m) cycle
        z0xxx(:,5) = x(:,i+1)
        call ols(y,z0xxx,b0xxx,n,4,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
        ss0xxx(i,j,k,m) = ss_reg - ss0(j) - ss0x(k,j) - ss0xx(m,j,k)
        f0xxx(i,j,k,m) = ss0xxx(i,j,k,m)/(ss_res/dble(n-5))
      end do
    end do
  end do
end do
if(p<5) exit outermost
!
! Calculate F()|()()()()
!
z0xxxx(:,1) = x(:,1)
do j=1,p
  z0xxxx(:,2) = x(:,j+1)
  ss0xxxx(:,j,j,:,:) = -9.0D0
  f0xxxx(:,j,j,:,:) = -9.0D0
  do k=1,p
    if (k==j) cycle
    z0xxxx(:,3) = x(:,k+1)
    ss0xxxx(:,j,k,j,:) = -9.0D0
    ss0xxxx(:,j,k,k,:) = -9.0D0
    f0xxxx(:,j,k,j,:) = -9.0D0
    f0xxxx(:,j,k,k,:) = -9.0D0
    do m=1,p
      if(m==j.or.m==k) cycle
      z0xxxx(:,4)=x(:,m+1)
      ss0xxxx(:,j,k,m,j) = -9.0D0
      ss0xxxx(:,j,k,m,k) = -9.0D0
      ss0xxxx(:,j,k,m,m) = -9.0D0
      f0xxxx(:,j,k,m,j) = -9.0D0
      f0xxxx(:,j,k,m,k) = -9.0D0
      f0xxxx(:,j,k,m,m) = -9.0D0
      do q=1,p
        if(q==j.or.q==k.or.q==m) cycle
        z0xxxx(:,5)=x(:,q+1)
```

```
            ss0xxxx(j,j,k,m,q)=-9.0D0
            ss0xxxx(k,j,k,m,q)=-9.0D0
            ss0xxxx(m,j,k,m,q)=-9.0D0
            ss0xxxx(q,j,k,m,q)=-9.0D0
            f0xxxx(j,j,k,m,q)=-9.0D0
            f0xxxx(k,j,k,m,q)=-9.0D0
            f0xxxx(m,j,k,m,q)=-9.0D0
            f0xxxx(q,j,k,m,q)=-9.0D0
            do i=1,p
              if(i==j.or.i==k.or.i==m.or.i==q) cycle
              z0xxxx(:,6) = x(:,i+1)
              call ols(y,z0xxxx,b0xxxx,n,5,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
              ss0xxxx(i,j,k,m,q) = ss_reg - ss0(j) - ss0x(k,j) - ss0xx(m,j,k) - ss0xxx(q,j,k,m)
              f0xxxx(i,j,k,m,q) = ss0xxxx(i,j,k,m,q)/(ss_res/dble(n-6))
            end do
          end do
        end do
      end do
    end do
    if(p<6) exit outermost
    !
    ! Calculate F()|()()()()()
    !
    z0xxxxx(:,1) = x(:,1)
    do j=1,p
      z0xxxxx(:,2) = x(:,j+1)
      ss0xxxxx(:,j,j,:,:,:) = -9.0D0
      f0xxxxx(:,j,j,:,:,:) = -9.0D0
      do k=1,p
        if (k==j) cycle
        z0xxxxx(:,3) = x(:,k+1)
        ss0xxxxx(:,j,k,j,:,:) = -9.0D0
        ss0xxxxx(:,j,k,k,:,:) = -9.0D0
        f0xxxxx(:,j,k,j,:,:) = -9.0D0
        f0xxxxx(:,j,k,k,:,:) = -9.0D0
        do m=1,p
          if(m==j.or.m==k) cycle
          z0xxxxx(:,4)=x(:,m+1)
          ss0xxxxx(:,j,k,m,j,:) = -9.0D0
          ss0xxxxx(:,j,k,m,k,:) = -9.0D0
          ss0xxxxx(:,j,k,m,m,:) = -9.0D0
          f0xxxxx(:,j,k,m,j,:) = -9.0D0
          f0xxxxx(:,j,k,m,k,:) = -9.0D0
          f0xxxxx(:,j,k,m,m,:) = -9.0D0
          do q=1,p
            if(q==j.or.q==k.or.q==m) cycle
            z0xxxxx(:,5)=x(:,q+1)
            ss0xxxxx(:,j,k,m,q,j)=-9.0D0
            ss0xxxxx(:,j,k,m,q,k)=-9.0D0
            ss0xxxxx(:,j,k,m,q,m)=-9.0D0
            ss0xxxxx(:,j,k,m,q,q)=-9.0D0
            f0xxxxx(:,j,k,m,q,j)=-9.0D0
            f0xxxxx(:,j,k,m,q,k)=-9.0D0
            f0xxxxx(:,j,k,m,q,m)=-9.0D0
            f0xxxxx(:,j,k,m,q,q)=-9.0D0
            do r=1,p
              if(r==j.or.r==k.or.r==m.or.r==q) cycle
              z0xxxxx(:,6)=x(:,r+1)
              ss0xxxxx(j,j,k,m,q,r)=-9.0D0
              ss0xxxxx(k,j,k,m,q,r)=-9.0D0
```

```
                ss0xxxxx(m,j,k,m,q,r)=-9.0D0
                ss0xxxxx(q,j,k,m,q,r)=-9.0D0
                ss0xxxxx(r,j,k,m,q,r)=-9.0D0
                f0xxxxx(j,j,k,m,q,r)=-9.0D0
                f0xxxxx(k,j,k,m,q,r)=-9.0D0
                f0xxxxx(m,j,k,m,q,r)=-9.0D0
                f0xxxxx(q,j,k,m,q,r)=-9.0D0
                f0xxxxx(r,j,k,m,q,r)=-9.0D0
                do i=1,p
                  if(i==j.or.i==k.or.i==m.or.i==q.or.i==r) cycle
                  z0xxxxx(:,7) = x(:,i+1)
                  call ols(y,z0xxxxx,b0xxxxx,n,6,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
                  ss0xxxxx(i,j,k,m,q,r) = ss_reg - ss0(j) - ss0x(k,j) - ss0xx(m,j,k) - ss0xxx(q,j,k,m) -
ss0xxxx(r,j,k,m,q)
                  f0xxxxx(i,j,k,m,q,r) = ss0xxxxx(i,j,k,m,q,r)/(ss_res/dble(n-7))
                end do
              end do
            end do
          end do
        end do
      end do
      if(p<7) exit outermost
      !
      ! Calculate F()|()()()()()()
      !
      z0xxxxxx(:,1) = x(:,1)
      do j=1,p
        z0xxxxxx(:,2) = x(:,j+1)
        ss0xxxxxx(:,j,j,:,:,:,:) = -9.0D0
        f0xxxxxx(:,j,j,:,:,:,:) = -9.0D0
        do k=1,p
          if (k==j) cycle
          z0xxxxxx(:,3) = x(:,k+1)
          ss0xxxxxx(:,j,k,j,:,:,:) = -9.0D0
          ss0xxxxxx(:,j,k,k,:,:,:) = -9.0D0
          f0xxxxxx(:,j,k,j,:,:,:) = -9.0D0
          f0xxxxxx(:,j,k,k,:,:,:) = -9.0D0
          do m=1,p
            if(m==j.or.m==k) cycle
            z0xxxxxx(:,4)=x(:,m+1)
            ss0xxxxxx(:,j,k,m,j,:,:) = -9.0D0
            ss0xxxxxx(:,j,k,m,k,:,:) = -9.0D0
            ss0xxxxxx(:,j,k,m,m,:,:) = -9.0D0
            f0xxxxxx(:,j,k,m,j,:,:) = -9.0D0
            f0xxxxxx(:,j,k,m,k,:,:) = -9.0D0
            f0xxxxxx(:,j,k,m,m,:,:) = -9.0D0
            do q=1,p
              if(q==j.or.q==k.or.q==m) cycle
              z0xxxxxx(:,5)=x(:,q+1)
              ss0xxxxxx(:,j,k,m,q,j,:)=-9.0D0
              ss0xxxxxx(:,j,k,m,q,k,:)=-9.0D0
              ss0xxxxxx(:,j,k,m,q,m,:)=-9.0D0
              ss0xxxxxx(:,j,k,m,q,q,:)=-9.0D0
              f0xxxxxx(:,j,k,m,q,j,:)=-9.0D0
              f0xxxxxx(:,j,k,m,q,k,:)=-9.0D0
              f0xxxxxx(:,j,k,m,q,m,:)=-9.0D0
              f0xxxxxx(:,j,k,m,q,q,:)=-9.0D0
              do r=1,p
                if(r==j.or.r==k.or.r==m.or.r==q) cycle
                z0xxxxxx(:,6)=x(:,r+1)
```

```
          ss0xxxxxx(:,j,k,m,q,r,j)=-9.0D0
          ss0xxxxxx(:,j,k,m,q,r,k)=-9.0D0
          ss0xxxxxx(:,j,k,m,q,r,m)=-9.0D0
          ss0xxxxxx(:,j,k,m,q,r,q)=-9.0D0
          ss0xxxxxx(:,j,k,m,q,r,r)=-9.0D0
          f0xxxxxx(:,j,k,m,q,r,j)=-9.0D0
          f0xxxxxx(:,j,k,m,q,r,k)=-9.0D0
          f0xxxxxx(:,j,k,m,q,r,m)=-9.0D0
          f0xxxxxx(:,j,k,m,q,r,q)=-9.0D0
          f0xxxxxx(:,j,k,m,q,r,r)=-9.0D0
          do s=1,p
            if(s==j.or.s==k.or.s==m.or.s==q.or.s==r) cycle
            z0xxxxxx(:,7)=x(:,s+1)
            ss0xxxxxx(j,j,k,m,q,r,s)=-9.0D0
            ss0xxxxxx(k,j,k,m,q,r,s)=-9.0D0
            ss0xxxxxx(m,j,k,m,q,r,s)=-9.0D0
            ss0xxxxxx(q,j,k,m,q,r,s)=-9.0D0
            ss0xxxxxx(r,j,k,m,q,r,s)=-9.0D0
            ss0xxxxxx(s,j,k,m,q,r,s)=-9.0D0
            f0xxxxxx(j,j,k,m,q,r,s)=-9.0D0
            f0xxxxxx(k,j,k,m,q,r,s)=-9.0D0
            f0xxxxxx(m,j,k,m,q,r,s)=-9.0D0
            f0xxxxxx(q,j,k,m,q,r,s)=-9.0D0
            f0xxxxxx(r,j,k,m,q,r,s)=-9.0D0
            f0xxxxxx(s,j,k,m,q,r,s)=-9.0D0
            do i=1,p
              if(i==j.or.i==k.or.i==m.or.i==q.or.i==r.or.i==s) cycle
              z0xxxxxx(:,8) = x(:,i+1)
              call ols(y,z0xxxxxx,b0xxxxxx,n,7,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
              ss0xxxxxx(i,j,k,m,q,r,s) = ss_reg - ss0(j) - ss0x(k,j) - ss0xx(m,j,k) - ss0xxx(q,j,k,m) -
ss0xxxx(r,j,k,m,q) - ss0xxxxx(s,j,k,m,q,r)
              f0xxxxxx(i,j,k,m,q,r,s) = ss0xxxxxx(i,j,k,m,q,r,s)/(ss_res/dble(n-8))
            end do
          end do
        end do
      end do
    end do
  end do
  end do
  if(p<8) exit outermost
end do outermost
return
end subroutine partial_f
!
!
!
subroutine ols(y,x,b,n,p,yhat,e,ss_reg,ms_reg,ss_res,ms_res,f_cal)
implicit none
integer:: n,p
double precision:: y(n,1), x(n,p+1), b(p+1,1), d1(p+1,p+1), d2(p+1,p+1), d3(p+1,1), yhat(n,1), e(n,1),
aa(1,1), ab(1,1)
double precision:: ss_reg,ms_reg,ss_res,ms_res,f_cal
d1 = matmul(transpose(x),x)
call invm(p+1,d1,d2)
d3 = matmul(transpose(x),y)
b = matmul(d2,d3)
yhat=matmul(x,b)
e = y-yhat
!
!ANOVA
```

```
!
aa = matmul(matmul(transpose(b),transpose(x)),y)-(sum(y)*sum(y)/n)
ss_reg = aa(1,1)
if(p==0) ms_reg = 9999999
if(p/=0) ms_reg = ss_reg/dble(p)
ab = matmul(transpose(y),y)-matmul(matmul(transpose(b),transpose(x)),y)
ss_res = ab(1,1)
ms_res = ss_res/dble(n-p-1)
if(ms_res==0.0D0) f_cal = 9999999
if(ms_res/=0.0D0) f_cal = ms_reg/ms_res
return
end subroutine ols
!
!
!
!
!
     subroutine invm(p,mat,inv)
     implicit none
     integer:: p,i,j,n,n1,m1,m2,k,i1
     double precision:: c,d,e,f
     double precision:: mat(p,p), matv(2*p,2*p), inv(p,p)
     do 5 i=1,p
       do 5 j=1,p
         matv(i,j)=mat(i,j)
 5   continue
     n = 2*p
     n1=p+1
     m1=p-1
     do 20 i=1,p
       m1=m1+1
       do 20 j=n1,n
         m2=j-m1
         if(m2==1) matv(i,j) = 1.0D0
         if(m2/=1) matv(i,j) = 0.0D0
20   continue
     do 60 i=1,p
       do 25 k=i,p
         if(matv(k,i)==0.0D0) go to 25
         i1=k
         go to 30
25      continue
     write(*,27)
27   format(/,'Sorry. The matrix is singular.')
     stop
30   if(i1==i) go to 40
     do 35 j=1,n
       e=matv(i1,j)
       f=matv(i,j)
       matv(i,j)=e
       matv(i1,j)=f
35   continue
40   d=matv(i,i)
     do 45 j=i,n
       matv(i,j)=matv(i,j)/d
45   continue
     do 55 k=1,p
       if(k==i) go to 55
                     if(matv(k,i)==0.0D0) go to 55
       c=matv(k,i)
```

```
      do 50 j=1,n
         matv(k,j)=matv(k,j)-(c*matv(i,j))
50      continue
55   continue
60   continue
     do 65 i=1,p
      do 65 j=n1,n
         k=j-p
         inv(i,k)=matv(i,j)
65   continue
     return
     end subroutine invm
!
!
!
!
!
!
!
subroutine covariance(corr_matrix, stdev_vec, cov_matrix, n)
implicit none
integer:: i,j,n
double precision:: corr_matrix(n,n), stdev_vec(n), cov_matrix(n,n)
do i=1,n
   do j=1,n
      cov_matrix(i,j) = corr_matrix(i,j)*stdev_vec(i)*stdev_vec(j)
   end do
end do
return
end subroutine covariance
!
!
!
!
!
subroutine mnorm(ix,mu,sigma,n,x)
implicit none
integer:: ix,n,i,k,i1,j,j1
double precision:: sigma(n,n),c(n,n),z(n),x(n),mu(n)
double precision:: sum, diff,a,znorm
a = sqrt(sigma(1,1))
do i=1,n
   c(i,1)=sigma(i,1)/a
end do
i=2
do
   i1 = i-1
   sum = 0.0D0
   do k=1,i1
      sum=c(i,k)*c(i,k) + sum
   end do
   diff = sigma(i,i) - sum
   c(i,i) = sqrt(diff)
   if(i==n) exit
   i = i+1
   i1 = i-1
   do j=2, i1
      j1=j-1
      sum=0.0D0
      do k=1,j1
```

```
      sum=c(i,k)*c(j,k)+sum
    end do
    c(i,j)=(sigma(i,j)-sum)/c(j,j)
  end do
end do
do i=1,n
  z(i)=znorm(ix)
end do
do i=1,n
  sum=0.0D0
  do j=1,i
    sum=c(i,j)*z(j)+sum
  end do
  x(i)=mu(i)+sum
end do
return
end

function znorm(ix)
implicit none
double precision:: r1,r2,v1,v2,s,znorm,urand
integer::ix
do
  r1 = urand(ix)
  r2 = urand(ix)
  v1 = 2.0D0*r1-1.0D0
  v2 = 2.0D0*r2-1.0D0
  s = (v1*v1)+(v2*v2)
  if(s>1.0D0) cycle
  znorm = v1*sqrt(-2.0D0*log(s)/s)
  if(.true.) exit
end do
return
end function znorm

function urand(ix)
implicit none
double precision:: urand
integer:: ix
ix= 16807*ix
if (ix<0) ix=(ix+2147483647)+1
urand = dble(ix)/2147483647.0D0
return
end function urand
```

ประวัติผู้เขียนวิทยานิพนธ์

นายธนาพันธ์ อัครเดชากร เกิดเมื่อวันที่ 8 กันยายน พ.ศ. 2524 ที่จังหวัดลำปาง จบการศึกษาชั้นมัธยมศึกษาตอนปลายจากโรงเรียนบุญวาทย์วิทยาลัย จังหวัดลำปาง สำเร็จการศึกษาปริญญาศิลปศาสตรบัณฑิต สาขาเศรษฐศาสตร์ จากคณะเศรษฐศาสตร์ มหาวิทยาลัยธรรมศาสตร์ เมื่อปีการศึกษา 2545 และเข้าศึกษาต่อในระดับปริญญาสถิติศาสตรมหาบัณฑิต สาขาวิชาสถิติ คณะพาณิชยศาสตร์และการบัญชี จุฬาลงกรณ์มหาวิทยาลัย เมื่อปีการศึกษา 2547