การค้นพบโมทีฟและดิสคอร์ดสำหรับอนุกรมเวลา โดยใช้เมทริกซ์โพรไฟล์แบบประมาณที่มีการเพิ่มสมรรถนะ

นายชนะพล อ้นวงษา

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2562

MOTIF AND DISCORD DISCOVERY IN TIME SERIES USING ENHANCED APPROXIMATED MATRIX PROFILE

Mr. Chanapon Onwongsa

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science in Computer Science
Department of Computer Engineering
FACULTY OF ENGINEERING
Chulalongkorn University
Academic Year 2019

หัวข้อวิทยานิพนธ์	การค้นพบโมทีฟและดิสคอร์ดสำหรับอนุกรมเวลา โดยใช้เมทริกซ์โพรไฟล์แบบประมาณที่มีการเพิ่มสมรรถนะ
โดย	นายชนะพล อ้นวงษา
สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	รองศาสตราจารย์ ดร.โชติรัตน์ รัตนามหัทธนะ

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์
(ศาสตราจารย์ ดร.สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ
(อาจารย์ ดร.ดวงดาว วิชาดากุล)

..... อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก
(รองศาสตราจารย์ ดร.โชติรัตน์ รัตนามหัทธนะ)

..... กรรมการภายนอกมหาวิทยาลัย
(ดร.เหมวรรณ ศิวรักษ์)

ชนะพล อ้นวงษา : การค้นพบโมทีฟและดิสคอร์ดสำหรับอนุกรมเวลา โดยใช้เมทริกซ์โพรไฟล์แบบประมาณที่มีการเพิ่มสมรรถนะ. (MOTIF AND DISCORD DISCOVERY IN TIME SERIES USING ENHANCED APPROXIMATED MATRIX PROFILE) อ.ที่ปรึกษาหลัก : รศ. ดร.โชติรัตน์ รัตนามหัทธนะ

การค้นพบโมทีฟและดิสคอร์ดสำหรับอนุกรมเวลา เป็นสาขาหนึ่งของงานวิจัยการทำเหมืองข้อมูลอนุกรมเวลา ซึ่งการค้นหารูปแบบของลำดับย่อยที่เกิดขึ้นซ้ำโดยมีลักษณะคล้ายกันและรูปแบบของลำดับย่อยที่มีความผิดปกติ คือการค้นพบโมทีฟและการค้นพบดิสคอร์ดตามลำดับ วิธีการหนึ่งที่เป็นที่นิยมสำหรับงานด้านการค้นพบโมทีฟและดิสคอร์ด คือการคำนวณหาเมทริกซ์โพรไฟล์ เนื่องจากเป็นวิธีที่รวดเร็ว และได้คำตอบที่ถูกต้องแม่นยำ แต่ถ้าหากข้อมูลอนุกรมเวลามีขนาดที่ใหญ่มาก จะส่งผลให้เวลาที่ใช้ในการคำนวณนั้นนานมากตามไปด้วย และอีกหนึ่งปัญหาที่สำคัญ คือการกำหนดค่าพารามิเตอร์ความยาว ของลำดับย่อย ที่ใช้ในการค้นหาโมทีฟ ที่ผู้ใช้ไม่สามารถทราบได้แน่ชัดว่า ควรกำหนดความยาวของลำดับย่อยเป็นเท่าใด

เพื่อแก้ปัญหาที่กล่าวมาข้างต้น งานวิจัยนี้จึงได้นำเสนอเมทริกซ์โพรไฟล์แบบประมาณ ที่มีการเพิ่มสมรรถนะ สำหรับทั้งปัญหาการค้นพบโมทีฟและดิสคอร์ด ซึ่งลดเวลาในการคำนวณได้เป็นอย่างมาก และผลลัพธ์ที่ได้มีความใกล้เคียงเดิม อีกทั้งนำเสนออัลกอริทึมสำหรับการกำหนดพารามิเตอร์ค่าความยาวโมทีฟที่เหมาะสม จากผลการทดลอง อัลกอริทึมสามารถลดกระบวนการคำนวณลงได้ ส่งผลให้เวลาที่ใช้ในการคำนวณลดลงได้เป็นอย่างมาก อีกทั้งผลลัพธ์ที่ได้ มีค่าใกล้เคียงกับการใช้เมทริกซ์โพรไฟล์แบบปกติ และยังสามารถค้นพบโมทีฟได้ โดยไม่จำเป็นต้องกำหนดค่าความยาวของลำดับย่อย

สาขาวิชา วิทยาศาสตร์คอมพิวเตอร์ ลายมือชื่อนิสิต .....
ปีการศึกษา 2562 ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

6070148421 : MAJOR COMPUTER SCIENCE
KEYWORD: TIME SERIES DATA MINING, MOTIF DISCOVERY, DISCORD DISCOVERY, MATRIX PROFILE

Chanapon Onwongsa : MOTIF AND DISCORD DISCOVERY IN TIME SERIES USING ENHANCED APPROXIMATED MATRIX PROFILE. Advisor: Assoc. Prof. CHOTIRAT RATANAMAHATANA, Ph.D.

Time series motif and discord discovery are a branch of research in time series data mining. Finding the most similar repeated subsequences and the anomaly subsequences are the motif and the discord discovery, respectively. One of the most popular methods to discover motif and discord is by using a Matrix Profile structure because it is fast and accurate. However, if the time series data is very large, its computation time could be very long. Another important issue is the subsequence length parameter that is used to find the motif; it is difficult for the users to know exactly the proper length of the subsequence.

In order to solve these problems, this research proposes an Enhanced Approximated Matrix Profile for both the motif and the discord discovery, which highly reduces the computation time, and the result is similar to the original. An algorithm for determining the optimum motif length using the Proper Length Motif discovery algorithm is used in combination with the proposed Enhanced Approximated Matrix Profile. Based on experimental results, the proposed algorithm is able to reduce the computation process, resulting in a significant reduction in computational time. In addition, the results are accurate, and the motif can be discovered without having to determine the length of the subsequence.

Field of Study: Computer Science Student's Signature .....
Academic Year: 2019 Advisor's Signature .....

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จสมบูรณ์ได้ด้วยดี เนื่องจากได้รับความกรุณาเป็นอย่างสูงจาก รองศาสตราจารย์ ดร.โชติรัตน์ รัตนามหัทธนะ อาจารย์ที่ปรึกษา ผู้คอยให้คำแนะ คำปรึกษา รวมถึงแนวคิดทั้งในด้านงานวิจัยและด้านอื่น ๆ อีกทั้งเป็นผู้ตรวจทานแก้ไขให้วิทยานิพนธ์เล่มนี้ให้สำเร็จลุล่วงได้ด้วยดี ขอขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณ อ.ดร.ดวงดาว วิชาดากุล และ ดร.เหมวรรณ ศิวรักษ์ ผู้ให้เกียรติเป็นประธานกรรมการและกรรมการสอบวิทยานิพนธ์ ที่เป็นผู้คอยให้คำชี้แนะ รวมถึงให้แนวทางในการปรับปรุงวิทยานิพนธ์ให้มีคุณภาพที่ดียิ่งขึ้น

ความสำเร็จในการทำวิทยานิพนธ์เล่มนี้ สำคัญที่สุด ขอขอบพระคุณ คุณพ่อ คุณแม่ และครอบครัว ที่ส่งเสริม สนับสนุน และให้กำลังใจเป็นอย่างดีเสมอมา จนกระทั่งวิทยานิพนธ์เล่มนี้สำเร็จลุล่วงไปได้ด้วยดี

ชนะพล อ้นวงษา

สารบัญ

หน้า

สารบัญตาราง

สารบัญภาพ

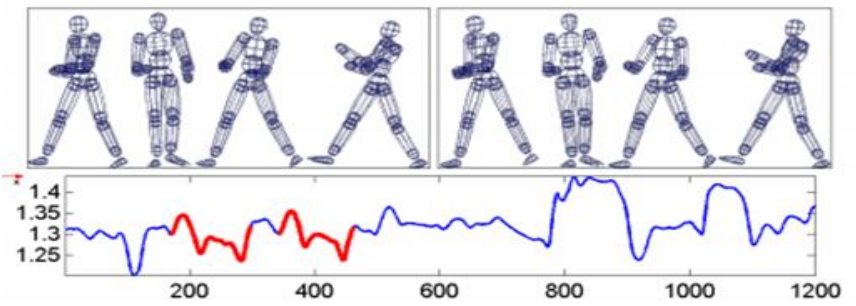
บทที่ 1 บทนำ

1.1 ที่มาและความสำคัญของปัญหา

ข้อมูลอนุกรมเวลา (Time Series Data) คือ ข้อมูลที่มีการเก็บรวบรวมค่าที่แปรตามระยะเวลา ตัวอย่างของข้อมูลอนุกรมเวลาที่สามารถพบได้ทั่วไป เช่น ข้อมูลอัตราการเต้นของหัวใจ ข้อมูลราคาหุ้น ข้อมูลระดับการสั่นสะเทือนของแผ่นดินไหว ซึ่งถ้านำข้อมูลอนุกรมเวลามาทำการค้นและสกัดข้อมูลแล้ว ทำให้ได้มาซึ่งสารสนเทศที่สำคัญของข้อมูล (Information) เมื่อมีการนำสารสนเทศ มาวิเคราะห์ก็จะได้มาซึ่งความรู้ (Knowledge) ที่เป็นประโยชน์ต่อผู้ที่จะนำไปศึกษาหรือใช้งานเพื่อนำความรู้ที่ได้มาไปพัฒนาให้ตรงตามวัตถุประสงค์ที่ได้ตั้งไว้

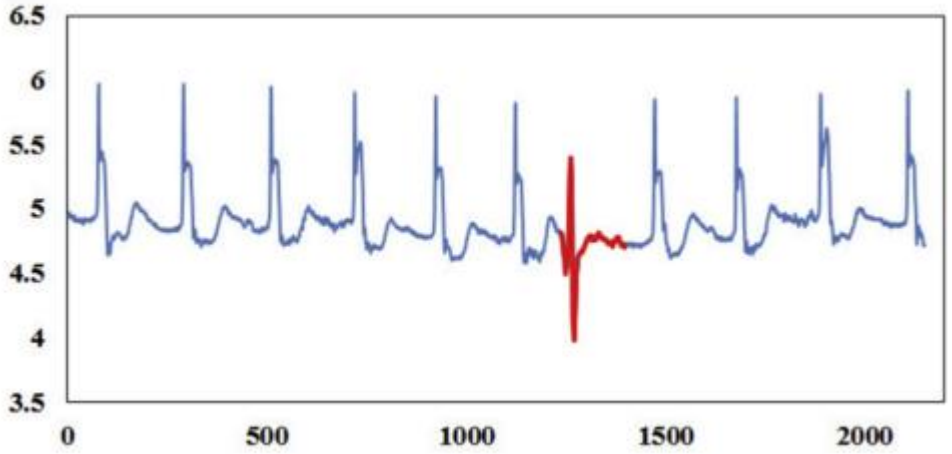
การค้นพบโมทีฟและดิสคอร์ด (Motif and Discord Discovery) ของอนุกรมเวลาได้มีการศึกษาเป็นระยะเวลาไม่น้อยกว่า 50 ปี [1] แต่ยังคงมีความสำคัญต่อการวิจัยในปัจจุบัน จากการค้นพบโมทีฟและดิสคอร์ด ทำให้ได้มาซึ่งสารสนเทศ ซึ่งจะนำไปใช้ประโยชน์ เพื่อสกัดหาความรู้จากข้อมูลได้อย่างมีประสิทธิภาพ

การค้นพบโมทีฟ (Motif Discovery) คือ การค้นหาลำดับย่อย (Subsequence) ที่ไม่ทับซ้อนกัน (Non-overlapping) ซึ่งมีรูปแบบ (Pattern) คล้ายกันมากที่สุดในอนุกรมเวลา นักวิจัยการทำเหมืองอนุกรมเวลา [2] มักให้ความสำคัญกับการค้นพบโมทีฟ เนื่องจากโมทีฟประกอบไปด้วยสารสนเทศ ที่มีความสำคัญต่อการค้นหาความรู้ (Knowledge) จากข้อมูลที่ศึกษา ภาพที่ 1 แสดงตัวอย่างของคู่โมทีฟที่ถูกค้นพบในข้อมูลการเคลื่อนที่ของตัวละครในเกมส์ ที่ถูกแปลงเป็นข้อมูลอนุกรมเวลา อีกทั้งในปัจจุบันนี้การค้นพบโมทีฟ ได้ถูกพัฒนาเพื่อนำไปใช้ในข้อมูลที่มีความหลากหลาย เช่น Seismology [3], Entomology [4], Stock [5] เป็นต้น



ภาพที่ 1 ตัวอย่างการค้นพบโมทีฟในข้อมูลการเคลื่อนที่ของตัวละครในเกมส์ที่มีท่าทางคล้ายกัน (ที่มา : [6])

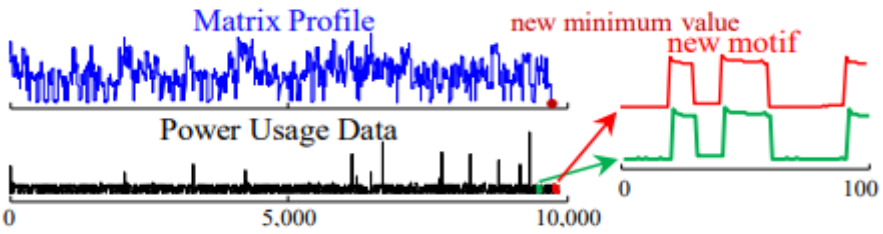
การค้นพบดิสคอร์ด (Discord Discovery) คือ การค้นหาลำดับย่อยใด ๆ ที่มีคุณลักษณะและรูปร่างแตกต่างกันมากที่สุดในอนุกรมเวลา ซึ่งปัญหาโดยส่วนใหญ่ที่มีความเกี่ยวข้องโดยตรงกับการค้นพบดิสคอร์ด คือ ปัญหาการตรวจจับความไม่ปกติที่เกิดขึ้น (Anomaly Detection) ตัวอย่างในภาพที่ 2 คือการค้นพบดิสคอร์ดในข้อมูลคลื่นไฟฟ้าหัวใจ เส้นสีแดงคือลำดับย่อยที่มีความแตกต่างกับลำดับย่อยอื่นมากที่สุด



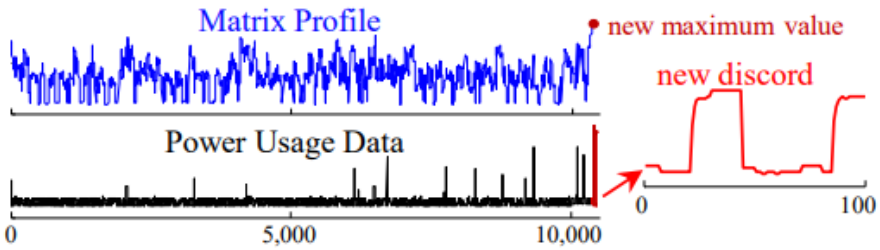
ภาพที่ 2 ตัวอย่างการค้นพบดิสคอร์ดในข้อมูลคลื่นไฟฟ้าหัวใจ (ECG)
(ที่มา : [7])

การวัดความคล้าย (Similarity Measure) คือการวัดระยะห่าง (Distance) ของลำดับย่อยในอนุกรมเวลา ซึ่งมีหลากหลายวิธีในการวัดให้เลือกใช้งาน ซึ่งการวัดที่ใช้กันอย่างแพร่หลายและเป็นที่ยอมรับโดยสากลสำหรับข้อมูลอนุกรมเวลาได้แก่ ระยะห่างยุคลิด (Euclidean Distance) [7] ไดนามิกไทม์วอร์ปปิง (Dynamic Time Warping) [8] สัมประสิทธิ์สหสัมพันธ์ (Correlation Coefficient) [9] เป็นต้น โดยทั่วไปแล้ว ฟังก์ชันระยะห่าง (Distance Function) ที่มีความซับซ้อนและใช้เวลาในการดำเนินการนานมาก อาจไม่ส่งผลหรือส่งผลน้อยมาก ต่อการคำนวณระยะห่าง ทำให้ผลลัพธ์จากการดำเนินการ ให้ผลลัพธ์ที่ไม่แตกต่างกันมาก สำหรับเกือบทุก ๆ ชุดข้อมูล จึงได้ข้อสรุปว่า ควรใช้วิธีวัดระยะห่างที่ง่ายไม่ซับซ้อน และยังคงประสิทธิภาพที่สุด [10] ดังนั้น ผู้วิจัยจึงเลือกใช้วิธีวัดระยะห่างแบบยุคลิดสำหรับงานวิจัยนี้ เนื่องจากการวัดระยะห่างแบบยุคลิดง่าย และใช้เวลาการดำเนินการน้อยกว่า เมื่อเทียบกับการวัดอื่นที่ได้กล่าวมาในข้างต้น

เมทริกซ์โพรไฟล์ (Matrix Profile) ถูกนำเสนอโดย Yeh C.-C. M. et al. [1] ในปี ค.ศ. 2016 เมทริกซ์โพรไฟล์เป็นเวกเตอร์ (Vector) สำหรับเก็บข้อมูลของระยะห่างของทุก ๆ ลำดับย่อยกับลำดับย่อยที่สนใจในอนุกรมเวลา โดยใช้ระยะห่างยุคลิด แสดงในภาพที่ 3 จะเห็นว่าเส้นสีน้ำเงินในแต่ละจุด คือระยะห่างแบบยุคลิดของลำดับย่อยของอนุกรมเวลาที่มีความยาว 100 จุด ซึ่งจุดสีแดงเล็ก ๆ บนเส้นสีน้ำเงินเป็นจุดที่บ่งบอกว่าตำแหน่งนั้นคือตำแหน่งที่เกิดโมทีฟเพราะเป็นจุดที่คำนวณได้ระยะห่างน้อยสุดระหว่างลำดับย่อยที่สนใจกับลำดับย่อยใด ๆ บนอนุกรมเวลา เส้นสีแดงและสีเขียวทางขวามือ คือคู่โมทีฟของอนุกรมเวลานี้ ในภาพที่ 4 แสดงการค้นพบดิสคอร์ด จะเห็นว่าสีน้ำเงินในแต่ละจุด คือระยะห่างแบบยุคลิดของลำดับย่อยของอนุกรมเวลาที่มีความยาว 100 จุด ซึ่งในภาพจะเห็นจุดสีแดงเล็ก ๆ บนเส้นสีน้ำเงิน ซึ่งจุดนั้นคือจุดสูงสุดเป็นจุดที่บ่งบอกว่าตำแหน่งนั้นคือตำแหน่งที่เกิดดิสคอร์ดเพราะเป็นจุดที่คำนวณได้ระยะห่างมากสุดระหว่างลำดับย่อยที่สนใจกับลำดับย่อยใด ๆ เส้นสีแดงทางขวามือคือดิสคอร์ดของอนุกรมเวลา เมทริกซ์โพรไฟล์นี้สามารถหาระยะห่างน้อยสุด และระยะห่างมากสุดได้พร้อมกัน ซึ่งสอดคล้องกับการหาโมทีฟและดิสคอร์ดตามลำดับ แต่เนื่องจากการคำนวณเมทริกซ์โพรไฟล์สำหรับข้อมูลอนุกรมเวลาที่มีขนาดใหญ่ จะใช้เวลามากขึ้นหลายเท่าตัว เนื่องจากอัตราการเติบโตของฟังก์ชันที่ใช้ในการดำเนินการ ไม่ได้มีอัตราการเติบโตในลักษณะของเชิงเส้น ดังแสดงในตารางที่ 1 เพื่อเป็นการลดเวลาในการดำเนินการ แต่ยังคงประสิทธิภาพสำหรับการค้นพบโมทีฟและดิสคอร์ด ดังนั้น การคำนวณทุกจุดข้อมูลบนอนุกรมเวลา เพื่อหาเมทริกซ์โพรไฟล์ทั้งหมด อาจไม่จำเป็นต้องคำนวณทุกคู่ลำดับย่อย ถ้ามีการคำนวณเฉพาะบางส่วนของเมทริกซ์โพรไฟล์ โดยอ้างอิงถึงหลักความน่าจะเป็นสำหรับจำนวนรอบที่เพียงพอต่อการค้น ซึ่งเป็นการเลือกเพียงแค่บางส่วนของจุดบนอนุกรมเวลาเพื่อให้ได้ค่าน้อยสุดและมากสุด ก็เป็นการเพียงพอในการค้นพบโมทีฟ และดิสคอร์ด



ภาพที่ 3 ตัวอย่างการค้นพบโมทีฟของข้อมูลการใช้พลังงานโดยการคำนวณเมทริกซ์โพรไฟล์ (ที่มา : [1])



ภาพที่ 4 ตัวอย่างการค้นพบดิสคอร์ดของข้อมูลการใช้พลังงานโดยการคำนวณเมทริกซ์โพรไฟล์ (ที่มา : [1])

ตารางที่ 1 เวลาในการดำเนินการสำหรับการคำนวณเมทริกซ์โพรไฟล์ของอนุกรมเวลาที่ความยาวต่างกัน เมื่อลำดับย่อยมีความยาว 256 จุดข้อมูล (ที่มา : [1])

ความยาวอนุกรมเวลา	2^{17}	2^{18}	2^{19}	2^{20}	2^{21}
เวลาในการคำนวณ	15.1 นาที	70.4 นาที	5.4 ชั่วโมง	24.4 ชั่วโมง	4.2 วัน

การค้นพบโมทีฟและดิสคอร์ด โดยการคำนวณเมทริกซ์โพรไฟล์นั้น ยังต้องมีการกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อย [11] ซึ่งในความเป็นจริงแล้ว ความยาวของลำดับย่อยเป็นค่าที่กำหนดขึ้นมาได้ยากมาก ทำให้บางครั้งจำเป็นต้องใช้ผู้เชี่ยวชาญเฉพาะด้าน ในการกำหนดช่วงของความยาวเพื่อให้ได้โมทีฟความยาวที่เหมาะสม [11] ทำให้เกิดงานวิจัยที่เกี่ยวข้องกับการค้นพบโมทีฟความยาวที่เหมาะสม (Proper Length Motif Discovery) [12] ซึ่งสามารถแก้ปัญหาการกำหนดความยาว ของลำดับย่อย

จากปัญหาที่กล่าวมาข้างต้น มี 2 ปัญหาหลักที่ผู้วิจัยต้องการแก้ คือ ลดเวลาการคำนวณของอนุกรมเวลาที่มีขนาดใหญ่ และกำจัดปัญหาที่ต้องกำหนดความยาวของลำดับย่อย โดยใช้การคำนวณเมทริกซ์โพรไฟล์โดยใช้การประมาณ ซึ่งใช้ทฤษฎีบททางคณิตศาสตร์เรื่องความน่าจะเป็นในการเลือกจำนวนรอบที่เพียงพอสำหรับการค้นพบโมทีฟและดิสคอร์ด เพื่อลดเวลาที่ใช้ในการดำเนินการสำหรับเมทริกซ์โพรไฟล์ โดยผลลัพธ์ของโมทีฟและดิสคอร์ดที่ได้ มีค่าใกล้เคียงกับค่าที่คำนวณได้จากเมทริกซ์โพรไฟล์ อีกหนึ่งปัญหาคือการกำหนดความยาวของลำดับย่อยสำหรับการค้น ในกรณีที่ไม่ทราบขนาดที่เฉพาะเจาะจงของลำดับย่อยสำหรับการค้นหาโมทีฟ ซึ่งปัญหานี้ถูกแก้ได้โดยอาศัยการพัฒนาและต่อยอดเพิ่มจากการใช้เมทริกซ์โพรไฟล์แบบประมาณ [13] และอัลกอริทึมจากงานวิจัย [12]

1.2 วัตถุประสงค์

1) เพื่อนำเสนออัลกอริทึมการค้นพบโมทีฟโดยแบ่งเป็นสองขั้นตอน ขั้นตอนแรก ทำการสร้างค่าเฉลี่ยของข้อมูลเพื่อสร้างตัวแทนอนุกรมเวลาใหม่ที่มีการลดมิติของข้อมูล ขั้นตอนที่สองประยุกต์ใช้ความน่าจะเป็น และทฤษฎีการประมาณค่า เพื่อสร้างเมทริกซ์โพรไฟล์แบบประมาณด้วยค่า $\tilde{k}$ ที่เหมาะสม (Approximated Matrix Profile with Proper $\tilde{k}$) สำหรับอนุกรมตัวแทน ซึ่งส่งผลให้ใช้เวลาในการประมวลผลน้อยกว่าเมทริกซ์โพรไฟล์แบบประมาณ (Approximated Matrix Profile) [13] ในขณะเดียวกัน ยังคงความถูกต้องของผลลัพธ์ ที่มีค่าใกล้เคียงกับการคำนวณจากเมทริกซ์โพรไฟล์ [1]
2) เพื่อนำเสนออัลกอริทึมสำหรับการค้นพบโมทีฟ โดยไม่จำเป็นต้องกำหนดพารามิเตอร์ความยาวของลำดับย่อย โดยการประยุกต์ใช้ตัวแทนอนุกรมเวลาและความน่าจะเป็น เพื่อสร้างเมทริกซ์โพรไฟล์แบบประมาณด้วยค่า $\tilde{k}$ ที่เหมาะสม ซึ่งสามารถลดเวลาการดำเนินการได้มากกว่าอัลกอริทึมสำหรับการค้นพบโมทีฟความยาวที่เหมาะสมโดยใช้เมทริกซ์โพรไฟล์แบบประมาณ (Proper Length Motif Discovery Algorithm using Approximated Matrix Profile) [13]

1.3 ขอบเขตของงานวิจัย

1) ในวิทยานิพนธ์นี้ทำการเปรียบเทียบความเร็วในการค้นพบโมทีฟกับงานวิจัย [13] และ [1]
2) ในวิทยานิพนธ์นี้ทำการเปรียบเทียบความถูกต้องของผลลัพธ์จากการค้นพบโมทีฟเทียบกับอัลกอริทึมแสตมป์ (**S**calable **T**ime series **A**nytime **M**atrix **P**rofile) [1]
3) ในวิทยานิพนธ์นี้จะไม่เปรียบเทียบผลในเมทริกซ์โพรไฟล์ แบบที่ใช้หน่วยประมวลผลกราฟิก (GPU-STAMP) [14] เนื่องจากหากปรับปรุงเมทริกซ์โพรไฟล์แบบปกติได้ ก็สามารถนำไปใช้กับเมทริกซ์โพรไฟล์แบบหน่วยประมวลผลกราฟฟิกได้เช่นกัน

1.4 ประโยชน์ที่ได้รับ

1) ได้อัลกอริทึมสำหรับค้นพบโมทีฟและดิสคอร์ด ซึ่งใช้เวลาในการคำนวณน้อยลงมาก ในขณะเดียวกัน ผลลัพธ์ที่ได้ยังคงเหมือนเดิม หรือใกล้เคียงกับผลลัพธ์ที่ถูกต้อง
2) ในกรณีที่ไม่มีการกำหนดค่าความยาวของลำดับย่อยสำหรับการค้นพบโมทีฟ อัลกอริทึมที่สร้าง สามารถหาโมทีฟความยาวที่เหมาะสมได้อย่างรวดเร็ว ในขณะเดียวกัน ผลลัพธ์ที่ได้ยังคงเหมือนเดิม หรือใกล้เคียงกับผลลัพธ์ที่ถูกต้อง

1.5 ขั้นตอนดำเนินงาน

1) ทบทวนงานวิจัยที่เกี่ยวข้องกับการทำเหมืองข้อมูลอนุกรมเวลา
2) ศึกษาอัลกอริทึมในหลากหลายรูปแบบ จากงานวิจัยที่เกี่ยวข้องกับปัญหาการค้นพบโมทีฟและดิสคอร์ด
3) ศึกษารูปแบบของอนุกรมเวลาที่มีการลดมิติข้อมูลในหลายรูปแบบ จากงานวิจัยที่เกี่ยวข้องเพื่อเปรียบเทียบว่าแบบใด เหมาะสมสำหรับปัญหาการค้นพบโมทีฟและดิสคอร์ด
4) ศึกษางานวิจัยเมทริกซ์โพรไฟล์ และเมทริกซ์โพรไฟล์แบบประมาณ เพื่อหาวิธีปรับปรุงในเรื่องของเวลาการดำเนินการ สำหรับการค้นหาโมทีฟและดิสคอร์ดให้น้อยลง ในขณะเดียวกันยังคงความถูกต้องใกล้เคียงเดิม
5) ออกแบบและพัฒนาอัลกอริทึม โดยผสานวิธีการลดมิติข้อมูลเข้ากับวิธีการของเมทริกซ์โพรไฟล์แบบประมาณ โดยเรียกว่าเมทริกซ์โพรไฟล์แบบประมาณที่มีการเพิ่มสมรรถนะ
6) ทดสอบประสิทธิภาพของอัลกอริทึมเมทริกซ์โพรไฟล์แบบประมาณ ที่มีการเพิ่มสมรรถนะและวัดผลโดยเทียบจากเวลาที่ใช้ และความถูกต้องที่ได้กับอัลกอริทึมการค้นพบโมทีฟอื่น ๆ
7) ออกแบบและพัฒนาอัลกอริทึม โดยที่อัลกอริทึมไม่จำเป็นต้องกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อย เพื่อประยุกต์ใช้กับเมทริกซ์โพรไฟล์แบบประมาณที่มีการเพิ่มสมรรถนะและทำการวัดผลโดยเทียบจากเวลาที่ใช้ในการดำเนินการ และวัดความถูกต้องที่ได้เทียบกับอัลกอริทึมการค้นพบโมทีฟที่เชื่อถือได้ (STAMP algorithm)
8) วิเคราะห์ สรุปผลการทดลอง และเรียบเรียงวิทยานิพนธ์

1.6 ผลงานวิจัยที่ได้ตีพิมพ์

Chanapon Onwongsa and Chotirat Ann Ratanamahatana. 2020. An Enhanced Time Series Motif Discovery Using Approximated Matrix Profile. In *2020 2nd International Conference on Image Processing and Machine Vision (IPMV 2020), August 05-07, 2020, Bangkok, Thailand*. ACM, New York, NY, USA.

บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 ข้อมูลอนุกรมเวลา (Time Series Data)

ข้อมูลอนุกรมเวลา คือ ลำดับของข้อมูลที่มีการแปรค่าตามเวลา โดยแต่ละจุดข้อมูลที่มีการบันทึกจะมีระยะห่างที่เท่ากันซึ่งถูกแบ่งเป็นช่วง ๆ ข้อมูลอนุกรมเวลาเป็นลำดับของข้อมูลที่ไม่ต่อเนื่องกันในเชิงทฤษฎีทางคณิตศาสตร์ (Discrete-time Data) ตัวอย่างของอนุกรมเวลา เช่น ข้อมูลราคาหุ้น [15] ข้อมูลเสียง [16] ข้อมูลคลื่นหัวใจ [17] ข้อมูลปริมาณการใช้ไฟฟ้า [18] ข้อมูลแผ่นดินไหว [3] ภาพที่ 5 แสดงตัวอย่างข้อมูลอนุกรมเวลาของราคาหุ้นที่มีการเปลี่ยนแปลงของมูลค่าตามเวลาของบริษัท Tesla

ภาพที่ 5 ภาพแสดงข้อมูลราคาหุ้นบริษัท Tesla ตั้งแต่กลางปี ค.ศ. 2010 ถึงปี ค.ศ. 2020 ซึ่งในกรอบสีเหลือง บนภาพแสดงลำดับย่อยของข้อมูลอนุกรมเวลา ปี ค.ศ. 2010 ถึงปี ค.ศ. 2020

คำนิยามของข้อมูลอนุกรมเวลาและลำดับย่อยที่ใช้ในวิทยานิพนธ์ฉบับนี้ มีดังนี้

อนุกรมเวลา (Time Series)

อนุกรมเวลา (Time Series) T คือลำดับของจำนวนจริง t_i โดยอนุกรมเวลา T นิยามโดย $T = t_1, t_2, \dots, t_n$ โดยที่ n คือความยาวของอนุกรมเวลา T

ลำดับย่อย (Subsequence)

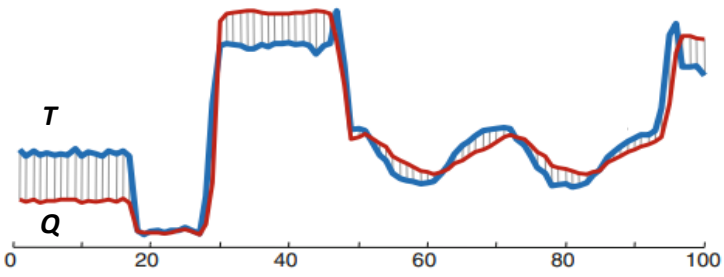
ลำดับย่อย (Subsequence) $T_{i,m}$ ของอนุกรมเวลา T คือสับเซตต่อเนื่อง (Contiguous Set) ของค่าใน T ที่เริ่มจากตำแหน่ง i และมีความยาว m ซึ่ง $T_{i,m}$ นิยามโดย

$T_{i,m} = t_i, t_{i+1}, \dots, t_{i+m-1}$ โดยที่ $1 \leq i \leq n - m + 1$

2.1.2 ระยะห่างยุคลิด (Euclidean Distance)

ระยะห่างยุคลิด (Euclidean Distance) คือ การวัดระยะห่างระหว่างจุดข้อมูล 2 จุด กำหนดให้อนุกรมเวลา $T = t_1, t_2, \dots, t_n$ และ $Q = q_1, q_2, \dots, q_n$ มีความยาว n ดังแสดงในภาพที่ 6 จะได้ระยะห่างยุคลิดของ 2 อนุกรมเวลาดังสมการ (1)

$$Euclidean(T, Q) = \sqrt{\sum_{i=1}^{n} (t_i - q_i)^2} \quad (1)$$



ภาพที่ 6 การวัดระยะห่างยุคลิดของ 2 อนุกรมเวลา T และ Q ที่มีความยาว 100 จุดข้อมูล (ที่มา : [19])

2.1.3 สัมประสิทธิ์สหสัมพันธ์ (Correlation Coefficient)

กำหนดให้อนุกรมเวลา $T = t_1, t_2, \dots, t_n$ และ $Q = q_1, q_2, \dots, q_n$ มีความยาว n จะได้สัมประสิทธิ์สหสัมพันธ์ของ 2 อนุกรมเวลาดังสมการ (2)

$$corr(T,Q) = \frac{\sum_{i=1}^{n} t_i q_i - n\mu_T \mu_Q}{n\sigma_{TQ}\sigma_{QT}} \quad (2)$$

โดยที่ $\mu_T = \frac{\sum_{i=1}^{n} t_i}{n}$, $\sigma_{TQ} = \sqrt{\frac{\sum_{i=1}^{n} t_i q_i}{n} - \mu_T^2}$, $\mu_Q = \frac{\sum_{i=1}^{n} q_i}{n}$ และ $\sigma_{QT} = \sqrt{\frac{\sum_{i=1}^{n} t_i q_i}{n} - \mu_Q^2}$

2.1.4 การทำให้เป็นบรรทัดฐานเดียวกัน (Normalization)

การวัดระยะห่างระหว่างข้อมูลอนุกรมเวลา มักต้องทำให้ข้อมูลเป็นบรรทัดฐานเดียวกันก่อนนำมาหาระยะห่าง ซึ่งโดยปกติแล้วข้อมูลอนุกรมเวลาที่ได้จะมีมาตราส่วน (Scale) ที่แตกต่างกัน ส่งผลให้ข้อมูลที่มีมาตราส่วนใหญ่กว่าจะมีความสำคัญมากกว่า ซึ่งอาจจะไม่เป็นจริงเสมอไป ขึ้นอยู่กับชุดข้อมูลที่นำมาทดลอง ทำให้ผลลัพธ์จากการคำนวณที่ไม่ได้ทำให้เป็นบรรทัดฐานเดียวกันก่อนอาจเกิดความผิดพลาดขึ้นได้ การทำให้เป็นบรรทัดฐานเดียวกัน คือการปรับมาตราส่วนและแอมพลิจูด (Amplitude) ของข้อมูลให้เป็นมาตราส่วนในระดับเดียวกัน และยังคงคุณสมบัติที่สำคัญของอนุกรมเวลา โดยในงานนี้ใช้การทำให้เป็นบรรทัดฐานแบบซี (Z-Normalization) เนื่องจากเป็นที่แพร่หลายในการใช้งาน อีกทั้งยังคงคุณลักษณะและคงคุณสมบัติเดิมของอนุกรมเวลาไว้ จะเห็นว่า ภาพที่ 7 คืออนุกรมเวลาที่ทั้งสองอนุกรมเวลา ยังไม่ถูกทำให้เป็นบรรทัดฐานแบบซี ซึ่งจะเห็นว่ามาตรส่วนมีความแตกต่างกันมาก ทำให้อาจตีความผิดพลาดไปว่าสองอนุกรมเวลานี้แตกต่างกันอย่างสิ้นเชิง แต่เมื่อทำให้เป็นบรรทัดฐานแบบซีแล้ว ดังแสดงในภาพที่ 8 จะเห็นว่าอนุกรมเวลาสองอนุกรมนี้มีความคล้ายคลึงกัน

การทำให้เป็นบรรทัดฐานแบบซี (Z-Normalization)

กำหนดให้อนุกรมเวลา $T = t_1, t_2, \dots, t_n$ มีค่า $\mu_T = \frac{\sum_{i=1}^{n} t_i}{n}$ และ $\sigma_T = \sqrt{\frac{\sum_{i=1}^{n} (t_i - \mu_T)^2}{n}}$ จะได้ $\hat{T} = \hat{t_1}, \hat{t_2}, \dots, \hat{t_n}$ เป็นอนุกรมเวลาที่ถูกทำให้เป็นบรรทัดฐานแบบซี โดยที่ $\hat{t_i} = \frac{t_i - \mu_T}{\sigma_T}$ สำหรับ $i = 1, 2, \dots, n$

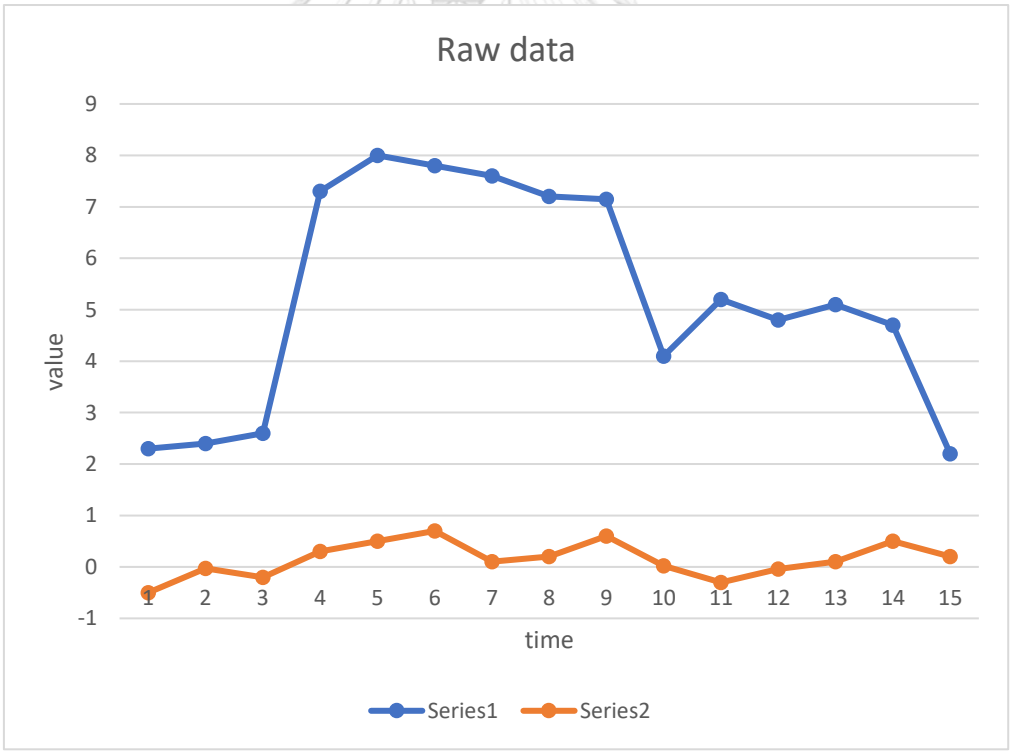
ระยะห่างยุคลิดที่ถูกทำให้เป็นบรรทัดฐานแบบซี (Z-Normalized Euclidean Distance)

กำหนดให้อนุกรมเวลา $T = t_1, t_2, \dots, t_n$ และ $Q = q_1, q_2, \dots, q_n$ มีความยาว n จะได้ระยะห่างยุคลิดที่ข้อมูลถูกทำให้เป็นบรรทัดฐานแบบซีของสองอนุกรมเวลาดังสมการ (3)

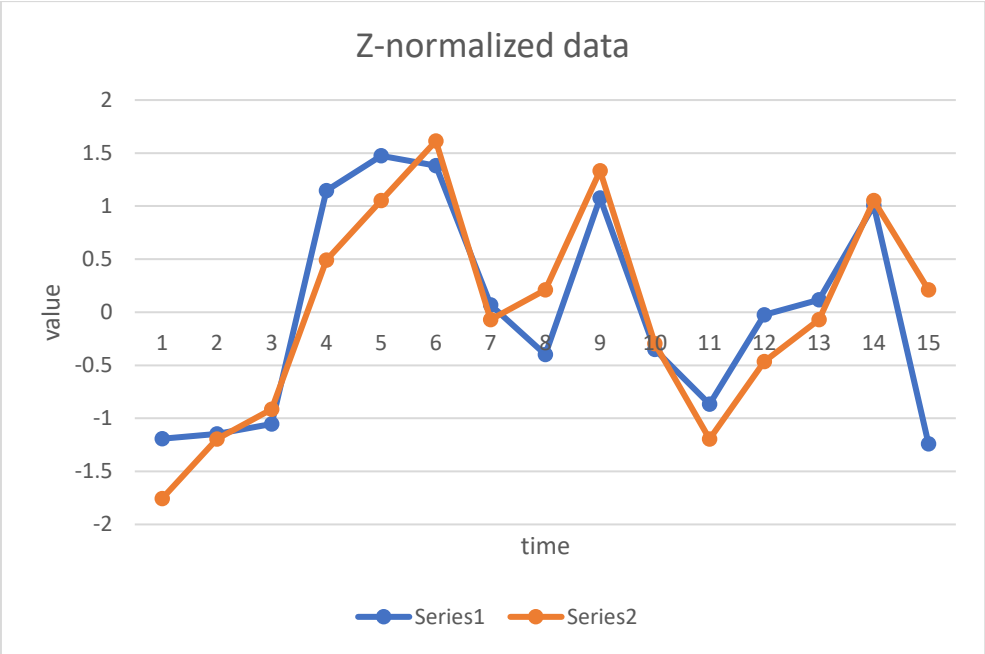
$$Dist(T, Q) = \sqrt{\sum_{i=1}^{n} (\hat{t}_i - \hat{q}_i)^2} \quad (3)$$

โดยที่ $\hat{t}_i = \frac{t_i - \mu_T}{\sigma_T}$ และ $\hat{q}_i = \frac{q_i - \mu_Q}{\sigma_Q}$ สำหรับ $i = 1, 2, \dots, n$

การดำเนินการต่อจากนี้สำหรับ การหาระยะห่างยุคลิดที่ถูกทำให้เป็นบรรทัดฐานแบบซี จะเรียกว่าระยะห่างยุคลิดแบบซี *(Z-Normalized Euclidean Distance)*



ภาพที่ 7 ข้อมูลอนุกรมเวลาที่ถูกสร้างขึ้น 2 อนุกรมเวลาที่ยังไม่ถูกทำให้เป็นบรรทัดฐานแบบซี



ภาพที่ 8 ข้อมูลอนุกรมเวลาจากภาพที่ 7 ที่ถูกทำให้เป็นบรรทัดฐานแบบซีแล้ว

2.1.5 โมทีฟ (Motif)

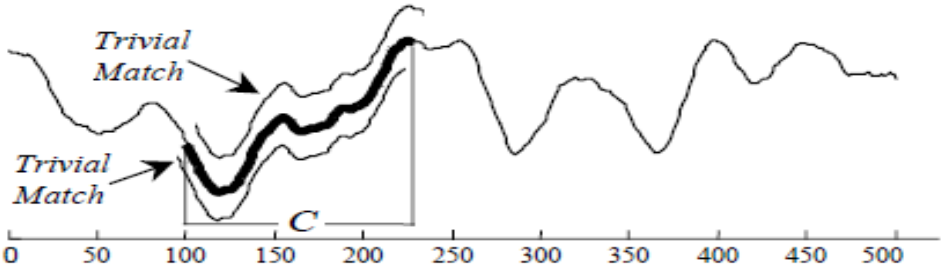
เนื่องจากนิยามของโมทีฟมีหลากหลายแบบ [20] สำหรับการนิยามโมทีฟที่ใช้ในวิทยานิพนธ์นี้ โมทีฟสำหรับอนุกรมเวลา (Time Series Motif) คือคู่ของลำดับย่อยที่มีความคล้ายคลึงกันมากที่สุด กล่าวคือ ลำดับย่อย $T_{x,m}$ และ $T_{y,m}$ จะเป็นคู่โมทีฟ ที่มีความยาว m ก็ต่อเมื่อ $Dist(T_{x,m}, T_{y,m}) \leq Dist(T_{i,m}, T_{j,m})$ โดยที่ $|i-j| \geq m$ และ $|x-y| \geq m$ สำหรับทุก $i, j \in [1, 2, \dots, n-m+1]$ โดยที่ $Dist$ คือฟังก์ชันที่คำนวณระยะห่างยุคลิดที่ข้อมูลถูกทำให้เป็นบรรทัดฐานแบบซี *(Z-Normalized Euclidean Distance)* ของสองลำดับย่อยใด ๆ บนอนุกรมเวลา

2.1.6 ดิสคอร์ด (Discord)

ดิสคอร์ดสำหรับอนุกรมเวลา (Time Series Discord) คือ คู่ของลำดับย่อยในอนุกรมเวลาที่มีความแตกต่างกัน จากลำดับย่อยอื่นมากที่สุด ซึ่งลำดับย่อย $T_{x,m}$ และ $T_{y,m}$ เป็นดิสคอร์ด ที่มีความยาว m ก็ต่อเมื่อ $Dist(T_{x,m}, T_{y,m}) \geq Dist(T_{i,m}, T_{j,m})$ โดยที่ $|i-j| \geq m$ และ $|x-y| \geq m$ สำหรับทุก $i, j \in [1, 2, \dots, n-m+1]$ โดยที่ $Dist$ คือฟังก์ชันที่คำนวณระยะห่างยุคลิดที่ข้อมูลถูกทำให้เป็นบรรทัดฐานแบบซี *(Z-Normalized Euclidean Distance)* ระหว่างสองลำดับย่อยใด ๆ บนอนุกรมเวลา

2.1.7 ลำดับย่อยแบบทับซ้อน (Trivial Match) [21]

กำหนดอนุกรมเวลา T ที่มีลำดับย่อย C ตำแหน่งเริ่มต้นของ C คือ p และมีลำดับย่อยแบบทับซ้อน M ตำแหน่งเริ่มต้นของ M คือ q กล่าวคือ M กับ C เป็นลำดับย่อยแบบทับซ้อน ถ้า $p = q$ หรือมีอนุกรมย่อย M' ที่เริ่มต้นที่จุด q' ที่ $Dist(C, M') >$ จำนวนจริงบวกใด ๆ ซึ่ง $q < q' < p$ หรือ $p < p' < q$



ภาพที่ 9 ลำดับย่อยแบบทับซ้อน (ที่มา : [21])

2.1.8 ค่าเฉลี่ยตัวแทนของอนุกรมเวลา

เนื่องจากข้อมูลอนุกรมเวลา มีค่าแปรตามเวลาที่เปลี่ยนไป โดยส่วนใหญ่แล้วแต่ละค่าที่เกิดขึ้นเป็นผลมาจากค่าก่อนหน้า อีกทั้งข้อมูลของอนุกรมเวลาโดยส่วนใหญ่จะมีสัญญาณรบกวน (Noises) อาจส่งผลให้มีบางค่าในอนุกรมเวลา มีค่าน้อยกว่า หรือมากกว่าค่าที่ควรจะเป็น ซึ่งวิธีการที่เป็นที่นิยมในการแก้ปัญหานี้ คือ การหาค่าเฉลี่ยเคลื่อนที่ n จุด (n – point running average) ซึ่งจำนวน n ที่เป็นไปได้นั้น ขึ้นอยู่กับว่า จะนำข้อมูลไปวิเคราะห์ในด้านใด และเป็นข้อมูลแบบใด ซึ่งการหาค่าเฉลี่ยเคลื่อนที่ มีนิยามดังนี้

$$\text{ค่าเฉลี่ยเคลื่อนที่} = \frac{\text{ข้อมูลล่าสุด } n \text{ จำนวน}}{n}$$

2.1.9 ปัญหาวันเกิด (Birthday Paradox Problem) [22]

จากทฤษฎีทางด้านความน่าจะเป็น ถ้ามีจำนวนคนที่สนใจ ทั้งหมด X คน โดยที่ X เป็นจำนวนเต็มบวกใด ๆ สำหรับการหาความน่าจะเป็นที่จะมี 2 คนใด ๆ เกิดในวันและเดือนเดียวกันจะใช้ 2 คุณสมบัติที่สำคัญของความน่าจะเป็นดังต่อไปนี้

คุณสมบัติที่ 1

กำหนดให้ $P(E)$ คือ ความน่าจะเป็นที่เหตุการณ์ E (Event) จะเกิดขึ้น และ $P(E')$ คือความน่าจะเป็นที่เหตุการณ์ E จะไม่เกิดขึ้น จะได้ว่า $P(E) = 1 - P(E')$

คุณสมบัติที่ 2

ความน่าจะเป็นที่เหตุการณ์ที่ 1 จะเกิด และความน่าจะเป็นที่เหตุการณ์ที่ 2 จะเกิด เป็นอิสระต่อกัน จะมีค่าเท่ากับผลคูณของความน่าจะเป็นที่เหตุการณ์ที่ 1 จะเกิดคูณกับความน่าจะเป็นที่เหตุการณ์ที่ 2 จะเกิด กล่าวคือ ถ้า $P(A)$ คือความน่าจะเป็นที่เหตุการณ์ A จะเกิด และ $P(B)$ คือความน่าจะเป็นที่เหตุการณ์ B จะเกิด โดยที่ความน่าจะเป็นที่เหตุการณ์ A และ B จะเกิดขึ้นพร้อมกัน คือ $P(A \cap B)$ เมื่อเหตุการณ์ A และ B เป็นอิสระต่อกันแล้ว จะนิยามโดย $P(A \cap B) = P(A)P(B)$ และมีการกำหนดให้ 1 ปี มี 365 วัน โดยจะพิจารณาความน่าจะเป็น ดังนี้

- คนที่ 1 มีความน่าจะเป็น ที่จะมีวันและเดือนเกิดไม่ตรงกับคนก่อนหน้าของเขา คือ $\frac{365}{365}$ เนื่องจากเป็นคนแรก จึงไม่มีคนใดก่อนหน้านี้ที่เกิดวันและเดือนเดียวกันกับเขา
- คนที่ 2 มีความน่าจะเป็นเท่ากับ $\frac{364}{365}$ เนื่องจากคนที่ 2 วันและเดือนเกิดจะต้องไม่ตรงกับคนก่อนหน้าของเขา นั่นคือ วันและเดือนเกิดจะต้องไม่ตรงกับคนที่ 1 ข้างต้น
- คนที่ 3 มีความน่าจะเป็นเท่ากับ $\frac{363}{365}$ เนื่องจากคนที่ 3 วันและเดือนเกิดจะต้องไม่ตรงกับคนก่อนหน้าของเขา นั่นคือ วันและเดือนเกิดจะต้องไม่ตรงกับคนที่ 1 และคนที่ 2 ข้างต้น

จากการอุปนัยทางคณิตศาสตร์ เมื่อทำการพิจารณาความน่าจะเป็นจนถึงคนที่ X ใด ๆ พบว่า ความน่าจะเป็นของคนที่ X ใด ๆ จะมีวันเกิดไม่ตรงกับคนก่อนหน้าของเขา คือ $\frac{365-X+1}{365}$ ดังนั้นความน่าจะเป็นที่ไม่มีใครมีวันและเดือนเกิดที่ตรงกันเลยคือ

$$\left(\frac{365}{365}\right)\left(\frac{364}{365}\right)\left(\frac{363}{365}\right)\dots\left(\frac{365-X+1}{365}\right) = \left(\frac{365!}{365^X(365-X)!}\right)$$

จะได้ว่าความน่าจะเป็นที่จะมีคนเกิดวันและเดือนตรงกันคือ

$$1 - \left(\frac{365!}{365^X(365-X)!}\right)$$

2.1.10 การประมาณค่า n! โดยที่ n มีขนาดใหญ่ด้วยวิธีการประมาณค่าของสเตอร์ลิง (Stirling's approximation)

การประมาณของสเตอร์ลิง คือการประมาณค่าของ $n!$ ซึ่งสามารถใช้ได้กับทั้งค่า n ไม่ว่าจะเป็นค่าที่เล็ก หรือค่าที่ใหญ่ก็ตาม การประมาณของสเตอร์ลิงก็ยังให้ผลที่แม่นยำ และใกล้เคียงกับค่าจริง ซึ่งถ้าเมื่อ n มีค่าที่ใหญ่มากเท่าไร ก็จะส่งผลให้ค่าความผิดพลาดน้อยลงตามไปด้วย โดยการประมาณค่า $n!$ ของสเตอร์ลิง นิยามดังนี้

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$

ซึ่งเมื่อมีการขยายขอบเขตการประมาณของ $n!$ จะได้ขอบเขตบนและขอบเขตล่างของ $n!$ ดังต่อไปนี้

$$\sqrt{2\pi n}\left(\frac{n}{e}\right)^n \leq n! \leq e\sqrt{n}\left(\frac{n}{e}\right)^n$$

โดย $n = 1,2,3, \ldots$ และ e เป็นค่าคงตัวทางคณิตศาสตร์ที่มีค่าประมาณ 2.71828.

จากหัวข้อที่ 2.1.9 กล่าวถึงความน่าจะเป็นที่คนสองคนใด ๆ มีวันและเดือนเกิดเดียวกัน เปรียบ เทียบได้กับความน่าจะเป็นที่จะเกิดคู่โมทีฟใด ๆ บนอนุกรมเวลา ซึ่งการกำหนดค่าความน่าจะเป็นสำหรับปัญหาวันเกิด เมื่อนำไปสร้างเป็นอัลกอริทึม โดยอัลกอริทึมนี้จะนำไปสู่การกำหนดค่าของจำนวนการวน รอบในการค้นลำดับย่อยเพื่อค้นหาโมทีฟและดิสคอร์ดบนอนุกรมเวลา ซึ่งอัลกอริทึมนั้นจะมีการคำนวณค่าแฟคทอเรียล ซึ่งถ้าอนุกรมเวลามีขนาดใหญ่ จะส่งผลให้ค่าของจำนวนที่นำมาแฟคทอเรียลจะมีขนาดใหญ่ตามไปด้วย ทำให้เวลาที่ใช้ในการคำนวณนานมาก ผู้วิจัยจึงนำวิธีการประมาณค่าของสเตอร์ลิงมาใช้ในหัวข้อที่ 2.1.10 เนื่องจากการประมาณด้วยวิธีนี้ให้ผลที่แม่นยำ และลดเวลาในการดำเนินการได้เป็นอย่างมาก

2.2 งานวิจัยที่เกี่ยวข้อง

2.2.1 การค้นพบโมทีฟ (Motif Discovery)

ปัญหาทั่วไปของการค้นพบโมทีฟ (Motif Discovery) เช่น ความยาวของลำดับย่อยที่ต่างกันสำหรับการค้นพบโมทีฟ (Detection of Motifs with Variable Lengths) [11] การจัดการข้อมูลที่มีการไหลเข้ามาตลอด (Data Stream Handling) [23] เป็นต้น ปัญหาและขอบเขต (Domain) สำหรับการค้นพบโมทีฟ ที่สนใจศึกษาจะส่งผลโดยตรงต่อการสร้างอัลกอริทึมเพื่อที่จัดการกับปัญหา และขอบเขตที่แตกต่างกันไป เช่น อัลกอริทึมสำหรับค้นพบโมทีฟที่ความยาวเดียวกัน (Fixed Length Motif) [1] อัลกอริทึมสำหรับค้นพบโมทีฟที่ความยาวต่างกัน (Variable Length Motif) [11] อัลกอริทึมสำหรับการค้นพบโมทีฟแบบแม่นตรง (Exact Motif) [4] อัลกอริทึมสำหรับโมทีฟแบบ

ประมาณ (Approximated Motif) [13] อีกทั้งในปี 2013 งานวิจัย [12] ได้มีการนำเสนออัลกอริทึมที่หาความยาวของลำดับย่อยที่เหมาะสมสำหรับการค้นพบโมทีฟ (Proper Length Motif Discovery) ส่งผลให้ ไม่จำเป็นต้องกำหนดความยาวของลำดับย่อยของโมทีฟ ซึ่งอาศัยขอบเขตความรู้ในเรื่องการบีบอัดข้อมูล (Compression) เกี่ยวข้องโดยตรงกับหลักการ MDL (Minimum Description Length) เพื่อวัดความถี่ ความคล้าย และคุณสมบัติที่เกิดขึ้น จากการเกิดโมทีฟ โดยอิงจากค่าการประหยัดบิต (Bitsave) กล่าวคือ โมทีฟที่มีการคำนวณค่าการประหยัดบิต แล้วมีค่าการประหยัดบิตสูงสุด จะเป็นโมทีฟที่สามารถบีบอัดข้อมูลตามหลักการของ MDL ได้ดีที่สุด ทำให้ได้กลุ่มของลำดับย่อยที่เกิดขึ้นถี่ที่สุด อีกทั้งมีรูปร่างและคุณสมบัติคล้ายกันมากที่สุด

2.2.2 การค้นพบดิสคอร์ด (Discord Discovery)

ปัญหาทั่วไปของการค้นพบดิสคอร์ด (Discord Discovery) จะเป็นปัญหาการค้นที่ใช้การวัดระยะห่างระหว่างคู่ของลำดับย่อยคล้ายคลึงกับการค้นพบโมทีฟ แต่จะใช้ค่าของระยะที่คำนวณได้ที่มีค่ามากที่สุดมาเป็นตัวกำหนดการเกิดดิสคอร์ด ความหมายของดิสคอร์ด คือลำดับย่อยที่มีคุณลักษณะและรูปร่างแตกต่างมากที่สุด หรือเป็นรูปร่างที่มีความถี่ในการเกิดขึ้นน้อยที่สุด

อัลกอริทึมสำหรับการค้นพบดิสคอร์ด คืออัลกอริทึมการค้นพบดิสคอร์ดแบบไล่ค้นทุกลำดับย่อยที่เป็นไปได้ (Brute Force Discord Discovery, BFDD) [23] ซึ่งใช้วิธีการวัดระยะห่างของแต่ละคู่ลำดับย่อย โดยทุกจุดข้อมูล จะถูกทำให้เป็นบรรทัดฐานแบบซี ก่อนที่จะนำมาหาระยะยุคลิดแบบซี โดยผลลัพธ์จะเป็นลำดับย่อยที่มีระยะห่างมากสุด ต่อมาในปี ค.ศ. 2016 ได้มีการนำเสนออัลกอริทึมคู่ขนานเพื่อใช้สำหรับแก้ปัญหาการค้นพบดิสคอร์ด (Parallel Discord Discovery) [24] โดยใช้หลักการแบ่งปัญหาใหญ่ออกเป็นปัญหาย่อย แล้วแก้แต่ละปัญหาย่อยที่แบ่งไว้สำหรับการค้นพบดิสคอร์ดคู่ขนานกันไป เพื่อเป็นการลดเวลาในการดำเนินการ

2.2.3 เมทริกซ์โพรไฟล์ (Matrix Profile) [1]

เมทริกซ์โพรไฟล์ (Matrix Profile) คือเมทริกซ์ที่สร้างขึ้นมาเพื่อเก็บค่าโพรไฟล์ระยะทาง โดยค่าของโพรไฟล์ระยะทางเป็นเวกเตอร์ที่มาจากการคำนวณระยะห่างยุคลิดแบบซี (Z-Normalized Euclidean Distance) ของแต่ละคู่ลำดับย่อยทั้งหมดบนอนุกรมเวลา โดยเมทริกซ์โพรไฟล์รองรับการคำนวณสำหรับการค้นพบทั้งโมทีฟและดิสคอร์ดพร้อม ๆ กัน แต่ถ้าหากอนุกรมเวลามีขนาดใหญ่มาก

จะส่งผลให้การคำนวณเมทริกซ์โพรไฟล์ใช้เวลาในการดำเนินการนานมาก เนื่องจากเมทริกซ์โพรไฟล์มีการคำนวณความซับซ้อนเชิงเวลาที่ไม่เป็นเชิงเส้น ส่งผลให้ระยะเวลาในการดำเนินการ จะแปรตามความยาวอนุกรมเวลา [1] อีกทั้งเมทริกซ์โพรไฟล์นี้จะเป็นส่วนสำคัญหนึ่งในวิธีหลักที่วิทยานิพนธ์นี้ใช้สำหรับอ้างอิงและมีการปรับปรุงแก้ไข สำหรับพัฒนาอัลกอริทึมเมทริกซ์โพรไฟล์ จึงต้องมีการอธิบายในรายละเอียดของขั้นตอนการดำเนินการ

คำนิยามและสัญลักษณ์ (Definition and Notation)

1) เซตของลำดับย่อยทั้งหมด (All-SubsequenceSet) ที่มีความยาว m ใดๆ เซตของลำดับย่อยทั้งหมด จะประกอบด้วยเซตของลำดับย่อยแต่ละตัว คือ $T_{i,m}$ ของอนุกรมเวลา T เป็นเซตของลำดับย่อย ที่มีความยาว m ทั้งหมดที่เป็นไปได้ ที่มีการเรียงลำดับแล้ว (Ordered Set) นิยามโดย

   $X = \{T_{i,m}, T_{i+1,m}, \dots, T_{n-m+1,m}\}$ โดยกำหนด $X[i] = T_{i,m}$

   เมื่อ m และ n เป็นจำนวนนับ และ i เป็นจำนวนนับโดยที่ $1 \leq i \leq n - m + 1$

2) โพรไฟล์ระยะทาง (Distance Profile)

   โพรไฟล์ระยะทาง $Dist$ คือเวกเตอร์ที่ได้จากการคำนวณ โดยใช้การวัดค่าระยะห่างแบบ ยุคลิดระหว่างลำดับย่อยที่ต้องการจะคำนวณ (Query) กับทุก ๆ ลำดับย่อยในเซตของลำดับย่อยทั้งหมด (All-Subsequence Set) ซึ่งการหาโพรไฟล์ระยะทางแบบยุคลิดจำเป็น ต้องมีการกำหนดความยาวของลำดับย่อยก่อน

3) ฟังก์ชัน 1 Nearest Neighbor-join Function (1NN-join Function)

   กำหนดเซตของลำดับย่อยทั้งหมด 2 เซต X, Y และเซตลำดับย่อย 2 ชุด คือ $X[i], Y[j]$ ฟังก์ชัน 1NN-join ใช้สัญลักษณ์ θ_{1nn} เป็นฟังก์ชันที่คืนค่า จริง (True) หรือ เท็จ (False) เมื่อ $Y[j]$ เป็นเพื่อนบ้านใกล้สุด (Nearest Neighbor) ของ $X[i]$ กล่าวคือ $Y[j]$ จะเป็นเพื่อนบ้านใกล้สุดของ $X[i]$ เมื่อคำนวณระยะห่างยุคลิดแบบซีระหว่าง $X[i]$ และ $Y[j]$ แล้ว ค่าที่ได้มีค่าห่างกันน้อยสุด (Minimum distance)

4) เซตของคู่ลำดับที่คล้ายกัน (Similarity Join Set)

   กำหนดเซตของลำดับย่อยทั้งหมด 2 เซต X, Y เซตของคู่ลำดับย่อยที่คล้ายกัน J_{XY} คือ ลำดับย่อยใน X และเพื่อนบ้านใกล้สุดใน Y ที่เป็นเซตที่เก็บสมาชิกเป็นคู่ ของทุก

J_{XY} เป็นผลมาจาก ข้อ 3) ซึ่งทำการเก็บค่าสมาชิกของคู่ลำดับย่อย เมื่อลำดับย่อยสองลำดับใด ๆ เป็นเพื่อนบ้านใกล้สุด

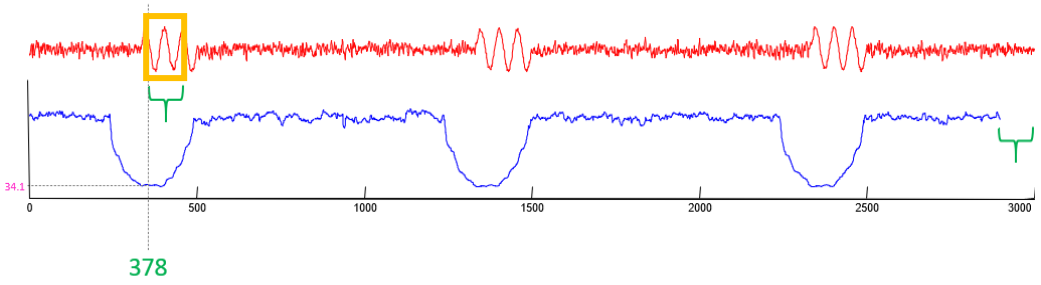
5) เมทริกซ์โพรไฟล์ (Matrix Profile)

เมทริกซ์โพรไฟล์ P_{XY} คือเมทริกซ์ที่มีการเก็บโพรไฟล์ระยะทาง ซึ่งโพรไฟล์ระยะทางเป็นเวกเตอร์ที่ได้มาจากการคำนวณระยะห่างยุคลิดแบบซี (Z-Normalized Euclidean Distance) ของแต่ละคู่ลำดับย่อยทั้งหมดบนอนุกรมเวลา จะมีการเก็บค่าเวกเตอร์ระหว่างแต่ละคู่ลำดับย่อยที่มีการคำนวณระยะห่างยุคลิดแบบซีใน J_{XY} โดยที่ $P_{XY}[i]$ เก็บค่าระยะห่างระหว่าง $X[i]$ และเพื่อนบ้านใกล้สุด $Y[i]$

6) เมทริกซ์โพรไฟล์อินเด็กซ์ (Matrix Profile Index)

เมทริกซ์โพรไฟล์อินเด็กซ์ I_{XY} คือเวกเตอร์ที่เก็บค่าตำแหน่ง ที่สัมพันธ์กับการคำนวณโพรไฟล์ระยะทางของลำดับย่อยใด ๆ บนอนุกรมเวลา T โดยค่าของตำแหน่งที่ได้จะเป็นเซตของจำนวนเต็ม โดยที่ $I_{XY}[i] = j$ ถ้า $(X[i], Y[j]) \in J_{XY}$

ภาพที่ 10 แสดงอนุกรมเวลา (สีแดง) และเมทริกซ์โพรไฟล์ที่ (สีน้ำเงิน) คือ ค่าระยะห่างที่เกิดจากลำดับย่อยที่สนใจกับลำดับย่อยใด ๆ โดยปีกกาสีเขียวด้านซ้ายมือของภาพ คือลำดับย่อยที่สนใจซึ่งจะเป็นลำดับย่อยที่มีลักษณะรูปร่างที่คล้ายกับลำดับย่อยในกรอบสีเหลือง โดยการหาค่าในเมทริกซ์โพรไฟล์ระหว่างลำดับย่อยที่สนใจกับลำดับย่อยที่มีตำแหน่งเริ่มต้นที่ 378 จะได้ค่าในเมทริกซ์โพรไฟล์ของตำแหน่งนี้คือ 34.1 กล่าวคือ เส้นสีน้ำเงินคือส่วนหนึ่งของเมทริกซ์โพรไฟล์ ซึ่งเมทริกซ์โพรไฟล์เป็นชื่อเรียกของการคำนวณโพรไฟล์ระยะทางทั้งหมด ซึ่งโพรไฟล์ระยะทางจะได้มาจากการคำนวณระยะระหว่างลำดับย่อยที่สนใจกับลำดับย่อยใด ๆ



ภาพที่ 10 เส้นสีน้ำเงินแทนเมทริกซ์โพรไฟล์ เส้นสีแดงแทนอนุกรมเวลา
(ที่มา : https://www.cs.ucr.edu/~eamonn/MatrixProfile.html)

STAMP Algorithm [1]

อัลกอริทึมแสตมป์ (STAMP Algorithm) เป็นอัลกอริทึมที่ใช้ในการคำนวณหาค่าต่ำสุด และค่าสูงสุดของเมทริกซ์โพรไฟล์ เพื่อบ่งบอกค่าและตำแหน่งของการเกิดโมทีฟและดิสคอร์ดตามลำดับ ซึ่งสามารถคำนวณได้ ณ เวลาใด ๆ (Anytime Algorithm) กล่าวคือ เป็นอัลกอริทึมที่สามารถเริ่มคำนวณ ณ ตำแหน่งของลำดับย่อยใดก่อนก็ได้ในอนุกรมเวลาที่สนใจ ซึ่งลำดับการเรียงตัวของตำแหน่งของลำดับย่อยจะมาจากการสุ่มแบบเรียงสับเปลี่ยน (Random Permutation) และอัลกอริทึมนี้ถ้าดำเนินการจนจบกระบวนการ จะได้โมทีฟและดิสคอร์ดแบบแม่นตรง แต่ถ้ามีการเลือกหยุดก่อนจบการดำเนินการ จะได้โมทีฟและดิสคอร์ดแบบประมาณ กล่าวคือ ถ้าต้องการโมทีฟและดิสคอร์ดแบบประมาณ ไม่จำเป็นต้องดำเนินการจนจบ ก็สามารถคืนค่าโมทีฟและดิสคอร์ดแบบประมาณได้

งานวิจัย [1] ในปี 2015 ได้มีการนำเสนออัลกอริทึมแมส (**M**ueen's **A**lgorithm for **S**imilarity **S**earch) สำหรับคำนวณระยะห่างของแต่ละคู่ลำดับย่อยซึ่งใช้การวัดระยะยุคลิดแบบซีเพื่อใช้เป็นมาตรวัดความคล้ายคลึงกันของแต่ละคู่ลำดับย่อย โดยมีการคำนวณในแต่ละจุดข้อมูลแบบคอนโวลูชัน จากนั้นนำค่าที่ได้ไปแปลงเป็นข้อมูลโดยกระบวนการ Fast Fourier Transform (FFT) จากนั้นก็แปลงค่ากลับมาผ่านกระบวนการ Inverse Fast Fourier Transform (IFFT) จากนั้นนำค่าที่ได้ไปหาความสัมพันธ์ระหว่างระยะห่างยุคลิดแบบซีกับค่าสัมประสิทธิ์สหสัมพันธ์ สุดท้ายแล้วจะได้ผลลัพธ์ของอัลกอริทึมแมส คือโพรไฟล์ระยะทาง (Distance Profile) ซึ่งเป็นค่าที่เก็บระยะห่างยุคลิดแบบซีของลำดับย่อยที่สนใจกับ ทุก ๆ ลำดับย่อยที่สัมพันธ์กับลำดับนั้นในอนุกรมเวลา ซึ่งมีความซับซ้อนเชิงเวลาเป็น $O(nlog(n))$ อธิบายได้ดังขั้นตอนตามตารางที่ 2-4 ดังต่อไปนี้

ตารางที่ 2 การคำนวณ InnerProductsSlidingSequence (ที่มา: [1])

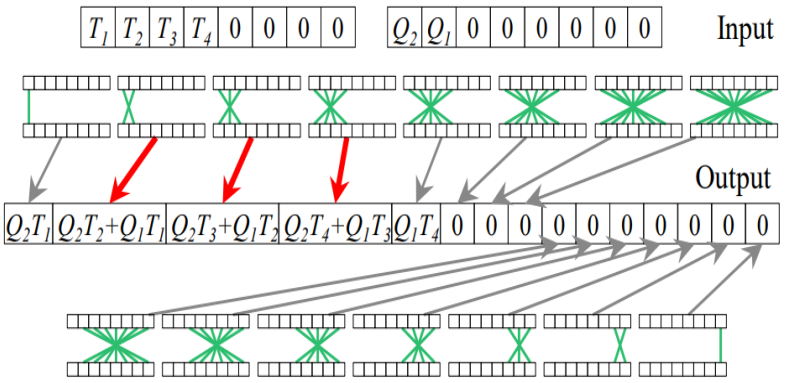
Procedure InnerProductsSlidingSequence(T, Q)

Input:
T: a user-defined time series sequence
Q: a query of time series T

Output:
QT: the inner product between subsequences T and Q

1: $n \leftarrow$ Len(T) // A time series T of length n
2: $m \leftarrow$ Len(Q) // A subsequences Q of length m
3: $T_{zeros} \leftarrow$ Append T with n zeros
4: $Q_{reverse} \leftarrow$ ReverseOfSubsequences(Q)
5: $Q_{reversezeros} \leftarrow$ Append $Q_{reverse}$ with $2n - m$ zeros
6: $Q_{reversezerosf} \leftarrow$ FFT($Q_{reversezerosa}$)
7: $T_{zerosf} \leftarrow$ FFT(T_{zeros})
8: $QT \leftarrow$ InverseFFT(MultiplyElementwise ($Q_{reversezerosf}, T_{zerosf}$))
9: **return** QT

ในตารางที่ 2 เป็นการคำนวณผลคูณภายในของลำดับย่อย (Inner Product) โดยอาศัยหลักการคอนโวลูชัน (Convolution) [25] แล้วทำการคูณจุดข้อมูลแต่ละจุดแบบผลคูณภายในมีการใช้ซีโร่แพดดิง (Zero Padding) เพื่อรองรับการคำนวณแบบคอนโวลูชัน แสดงดังภาพที่ 11 จากนั้นทำการแปลงข้อมูลที่ได้โดยใช้ ฟาสฟูเรียร์ทรานสฟอร์ม (Fast Fourier transform) และทำการแปลงกลับมาโดย Inverse Fast Fourier Transform (IFFT) ซึ่งค่าที่ได้จะถูกนำไปใช้ในกระบวนการถัดไปโดยผ่านอัลกอริทึมแมส



ภาพที่ 11 แสดงการคำนวณแบบคอนโวลูชันบนลำดับย่อยที่สนใจกับลำดับย่อยใด ๆ โดยมีการใช้ซีโร่แพดดิง (ที่มา: [1, 25])

ตารางที่ 3 อัลกอริทึม MASS (ที่มา: [24])

Procedure MASS(T, Q)

Input:
T: a user-defined time series sequence
Q: a query of time series T

Output:
$Dist$: a distance profile of the query Q

1: $QT \longleftarrow$ InnerProductsSlidingSequences(T, Q)
2: $M_T, \mu_Q \longleftarrow$ CalculateMean(T, Q)
3: $\Sigma_T, \sigma_Q \longleftarrow$ CalculateStandardDiviation(T, Q)
4 $Dist \longleftarrow$CalculateDistanceProfile($T, Q, QT, \mu_Q, \sigma_Q, M_T, \Sigma_T$)
5: **return** $Dist$

อัลกอริทึมแมสจะคืนผลลัพธ์เป็นค่าจริง ซึ่งได้จากการคำนวณระยะห่างระหว่างลำดับย่อยที่สนใจ Q กับ ทุก ๆ ลำดับย่อยใน T ซึ่งในบรรทัดที่ 2 และ 3 ของตารางที่ 3 เป็นการคำนวณค่าเฉลี่ยเลขคณิตและส่วนเบี่ยงเบนมาตรฐานของลำดับย่อย ตามลำดับ ในบรรทัดที่ 4 เป็นการหาระยะห่างของแต่ละลำดับย่อย โดยอาศัยค่าสัมประสิทธิ์สหสัมพันธ์กับการคำนวณระยะห่างยุคลิดแบบซี ซึ่งแต่ละค่าในเวกเตอร์ $Dist_i$ คำนวณได้จากสมการ (4)

$$Dist_i = \sqrt{2m\left(1 - \frac{QT_i - m\mu_Q M_{Ti}}{m\sigma_Q \Sigma_{Ti}}\right)} \quad (4)$$

เมื่อ m คือความยาวของลำดับย่อย, QT_i คือ อินเวอร์สฟาสฟูเรียร์ทรานสฟอร์มของผลคูณจุดระหว่าง $Q_{reversezerosf}$ และ T_{zerosf} จากตารางที่ 2 ค่า μ_Q คือค่าเฉลี่ยเลขคณิตของ Q, σ_Q คือ ค่าเบี่ยงเบนมาตรฐานของ Q, M_{Ti} คือ ค่าเฉลี่ยเลขคณิตที่แปรตามลำดับ i ของ $T_{i,m}$ และ Σ_{Ti} คือ ค่าเบี่ยงเบนมาตรฐานแปรตามลำดับ i ของ $T_{i,m}$

อัลกอริทึม ณ เวลาใด ๆ สำหรับการหาเมทริกซ์โพรไฟล์สำหรับข้อมูลอนุกรมเวลาขนาดใหญ่ คือ อัลกอริทึมแสตมป์ (Scalable Time series Anytime Matrix Profile) แสดงในตารางที่ 4 ซึ่งผลลัพธ์ที่ได้จากการดำเนินการ คือเมทริกซ์โพรไฟล์ P ซึ่งบรรจุโพรไฟล์ระยะห่างยุคลิตแบบซีของ

แต่ละคู่ลำดับย่อย และเมทริกซ์โพรไฟล์อินเด็กซ์ I ที่เก็บค่าตำแหน่งของแต่ละคู่ลำดับย่อยที่สัมพันธ์กับเมทริกซ์โพรไฟล์

ตารางที่ 4 อัลกอริทึม STAMP (ที่มา: [1])

Procedure STAMP(T, m)

Input:
T: a user-defined time series sequence
m: length of the given subsequence

Output:
P: an updated matrix profile
I: an associated updated matrix profile index

1: $n \leftarrow$ Len(T)
2: $P \leftarrow$ infinity
3: $I \leftarrow$ zero
4: $idxes \leftarrow$ range(1, $n - m + 1$)
5: **for** idx **in** $idxes$ // In any order idx in $idxes$
6: $Dist \leftarrow$ MASS(Q, T) // Q is subseq in T
7: $P, I \leftarrow$ UpdateMinElementwise($P, I, Dist, idx$)
8: **end for**
9: **return** P, I

ในตารางที่ 4 แสดงการคำนวณหาเมทริกซ์โพรไฟล์ (Matrix Profile) จากอัลกอริทึม STAMP ผลลัพธ์ที่ได้ คือเมทริกซ์โพรไฟล์ และตำแหน่งของเมทริกซ์โพรไฟล์ (Matrix Profile Index) โดยอาศัยอัลกอริทึมแมสจากตารางที่ 3 ซึ่งการคำนวณจะไม่เรียงลำดับ กล่าวคือ อัลกอริทึม STAMP จะทำการสุ่มลำดับการคำนวณ และอัลกอรึทึมดังกล่าวเป็นอัลกอริทึม ณ เวลาใด ๆ (Anytime Algorithm)

2.2.4 อัลกอริทึมการค้นพบโมทีฟและดิสคอร์ดของอนุกรมเวลา โดยใช้เมทริกซ์โพรไฟล์แบบประมาณ (Approximated Matrix Profile) [13]

อัลกอริทึมการค้นพบโมทีฟและดิสคอร์ดของอนุกรมเวลา โดยใช้เมทริกซ์โพรไฟล์แบบประมาณจะมีวิธีการคล้ายกับ STAMP algorithm เกือบทุกประการดังที่ได้กล่าวใน 2.2.3 แตกต่างกันที่จำนวนของการวนรอบของอัลกอริทึม อัลกอริทึมนี้จะอ้างอิงกับปัญหาวันเกิดโดยคำนวณหาการวนรอบ k อธิบายดังนี้ จากความน่าจะเป็นที่มีคนเกิดวันและเดือนตรงกันสามารถนำมาเขียนในรูปการประมาณ

ของสมการความน่าจะเป็นได้ดังสมการ (5)

$$p(x) \approx 1 - e^{\frac{-x(x-1)}{(2\times365)}} \quad (5)$$

โดย $p(x)$ คือ ความน่าจะเป็นที่จะมีคน x คน เกิดวันและเดือนตรงกัน ค่าคงตัว $e \approx 2.718281828$ ต่อมาได้มีการดัดแปลงปัญหาวันเกิดกับการค้นพบโมทีฟ [13] ได้สมการความน่าจะเป็นดังสมการ (6)

$$Prob(k,n,m) = 1 - e^{\frac{-k^2}{2(n-m+1)}} \quad (6)$$

โดย $Prob(k,n,m)$ คือ ความน่าจะเป็นของการค้นพบโมทีฟ k คือ จำนวนครั้งในการวนรอบเพื่อหาโมทีฟ n และ m คือ คือความยาวของอนุกรมเวลา และความยาวของลำดับย่อยที่ต้องการค้นหาโมทีฟ ตามลำดับ

2.2.5 อัลกอริทึมค้นพบโมทีฟที่ความยาวเหมาะสม (Proper Length Motif Discovery) [12]

จุดอ่อนหลักจุดหนึ่งของการค้นพบโมทีฟ คือการต้องกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อยที่เหมาะสม เพื่อแก้ไขปัญหาที่กล่าวมาข้างต้น จึงได้มีการสร้างอัลกอริทึมการค้นพบโมทีฟที่ความยาวเหมาะสม [12] แสดงในตารางที่ 5 ซึ่งสามารถแก้ปัญหาการกำหนดค่าความยาวของลำดับย่อยได้ ดังนั้นในส่วนนี้จะอธิบายรายละเอียดและขั้นตอนต่าง ๆ ในอัลกอริทึมนี้ ซึ่งจะเป็นอีกส่วนสำคัญในการนำมาประยุกต์ใช้กับอัลกอริทึมหลักของวิทยานิพนธ์นี้

คำนิยามต่าง ๆ สำหรับการคำนวณค่าประหยัดบิตมีดังนี้ [12, 26] การทำให้แต่ละจุดข้อมูลไม่ต่อเนื่อง เป็นบรรทัดฐานเดียวกัน (Discrete Normalization)

การทำให้ในแต่ละข้อมูลไม่ต่อเนื่อง คือการแปลงข้อมูลในแต่ละจุดของอนุกรมเวลา T ที่เป็นจำนวนจริง (Real-valued) ให้เป็นข้อมูลไม่ต่อเนื่อง จะได้ค่าที่ดำเนินการแล้วอยู่ในช่วง $[1, 2^s]$ ซึ่งคำนวณได้จากสมการ (7)

$$Discrete(T) = round\left[\left(\frac{T-max}{max-min}+1\right)*(2^s-1)\right]+1 \quad (7)$$

เมื่อกำหนดค่า $s = 64$ อ้างอิงจากงานวิจัย [26]

1) ค่าคุณลักษณะความยาวของข้อมูลอนุกรมเวลา (Description Length)
ค่าคุณลักษณะความยาวของข้อมูลอนุกรมเวลา T คือ ค่าที่เป็นผลลัพธ์จำนวนบิตที่เพียงพอต่อการอธิบายคุณลักษณะของตัวมันเอง สำหรับการคำนวณได้จากสมการ (8) และ (9)

$$DL(T) = n * Entropy(T) \tag{8}$$

$$Entropy(T) = -\sum_{i} P_i \log_2(P_i) \tag{9}$$

เมื่อ n คือพารามิเตอร์ความยาวของข้อมูลอนุกรมเวลา T และ P_i คือค่าความน่าจะเป็นที่เหตุการณ์ i จะปรากฏขึ้นบนอนุกรมเวลา T

2) ค่าคุณลักษณะความยาวสำหรับการเก็บข้อมูลของลำดับย่อย R ใด ๆ โดยที่ R เป็นสมาชิกของกลุ่ม G โดยใช้ H ซึ่ง H คือสมมติฐานที่เป็นศูนย์กลางของกลุ่ม ซึ่ง H จะถูกบรรจุด้วยค่าเฉลี่ยของทุก ๆ สมาชิกภายในกลุ่ม กล่าวคือ ใช้สมมติฐาน H เพื่อการบีบอัดข้อมูล ซึ่งคำนวณได้จากสมการ (10)

$$DLC(G) = DL(H) + \sum_{R \in G} DL(R|H) \tag{10}$$

เมื่อสมมติฐาน H ที่ใช้ในงานวิจัยนี้คือ ค่าคุณลักษณะความยาวของลำดับย่อยเฉลี่ยสำหรับคู่ของลำดับย่อยเริ่มต้นที่พิจารณาในรอบนั้น และนิยามค่าคุณลักษณะความยาวของลำดับย่อย R ใด ๆ โดยใช้สมมติฐาน H ในการบีบอัดข้อมูล คือ $DL(R|H)$ โดยที่ $DL(R|H) = DL(R - H)$

3) ค่าการประหยัดบิต (Bitsave) คือ การวัดค่าที่เปลี่ยนแปลงไปของค่าคุณลักษณะความยาวของก่อนและหลังการใช้สมมติฐาน H กล่าวคือ เป็นการวัดค่าความแตกต่างระหว่างค่าคุณลักษณะความยาวในการเก็บข้อมูล (Description Length) ของลำดับย่อยก่อนบีบอัดและหลังบีบอัด โดยใช้สมมติฐาน H ดังสมการ (11)

$$Bitsave = DL(Before) - DL(After) \tag{11}$$

4) ค่าการประหยัดบิตสำหรับการสร้างกลุ่ม G โดยเริ่มจากลำดับย่อยที่มีความคล้ายคลึงกันมากที่สุด (A, B) คำนวณได้จากสมการ (12)

$$Bitsave = DL(A) + DL(B) - DLC(G) \tag{12}$$

5) ค่าการประหยัดบิตสำหรับการเพิ่มลำดับย่อย A' ซึ่งเป็นเพื่อนบ้านของ A เข้าไปในกลุ่ม G ซึ่งจะได้กลุ่มใหม่ G' คำนวณได้จากสมการ (13)

$$Bitsave = DL(A') + DLC(G) - DLC(G') \tag{13}$$

6) ผลรวมค่าการประหยัดบิตของลำดับย่อยที่เป็นโมทีฟ คือ ค่าที่วัดได้จากผลต่างระหว่างผลรวมทั้งหมดสำหรับค่าการประหยัดบิตที่หาจากโมทีฟแต่ละตัว และค่าคุณลักษณะความยาวสำหรับการเก็บข้อมูลของสมมติฐาน H ในสมการ (14)

$$totalbitsave = \sum_{j} \left[DL\left(T_{o(j),v}\right) - DL(T_{o(j),v}|H) \right] - DL(H) \qquad (14)$$

เมื่อ $o(j)$ คือ ตำแหน่งเริ่มต้นของลำดับย่อยที่ j ที่เป็นสมาชิกในชุดโมทีฟ $T_{o(j),v}$ คือ ลำดับย่อยของอนุกรมเวลา T ซึ่งเริ่มต้นที่ตำแหน่ง $o(j)$ และมีความยาว v

ตารางที่ 5 อัลกอริทึมการค้นพบโมทีฟที่ความยาวเหมาะสม (ที่มา : [12])

Procedure ProperLengthMotifDiscovery(T)

Input:
T: Time series T

Output:
$Motif$: Proper length of time series motif discovery

1: **for** l = 2 **to** Len($\lfloor T/2 \rfloor$)
2: $\{A, B\} \leftarrow$ MotifCandidateDiscovery(T, l)
3: **for each** k **in** set(A, B)
4: $G \leftarrow$ CreateGroup(A, B)
5: **If** Bitsave$(G) < 0$ **then break**
6: $G \leftarrow$ AddAllNeighobor(G, T, l)
7: $Motif \leftarrow$ UpdateResult$(Motif, G)$
8: **end for**
9: **return** $Motif$

อัลกอริทึมนี้เป็นอัลกอริทึมสำหรับการค้นหาโมทีฟความยาวที่เหมาะสม โดยเริ่มจากการค้นความยาวของลำดับย่อยตั้งแต่ 2 จนถึงครึ่งหนึ่งของความยาวทั้งหมดของอนุกรมเวลาที่สนใจ โดยผลลัพธ์จากอัลกอริทึมนี้จะคืนค่าเป็น โมทีฟที่ความยาวเหมาะสม จะเห็นว่าข้อมูลนำเข้าสำหรับอัลกอริทึมนี้ จะมีเฉพาะข้อมูลอนุกรมเวลาเท่านั้น โดยไม่จำเป็นต้องกำหนดพารามิเตอร์ความยาวของลำดับย่อย

บรรทัดที่ 2 เป็นการหาลำดับย่อยซึ่งมีความคล้ายคลึงกันมากที่สุด k อันดับ ที่มีความยาวของลำดับย่อยขนาด l

บรรทัดที่ 4 เป็นการสร้างกลุ่มของลำดับย่อยใด ๆ สองลำดับย่อย ที่ได้จากบรรทัดที่ 2 โดยเริ่มจากคู่ลำดับย่อยที่คล้ายกันมากที่สุด

บรรทัดที่ 5 เมื่อมีการสร้างกลุ่มของลำดับย่อยจากบรรทัดที่ 4 จากนั้นจะนำมาหาค่าการประหยัดบิตสำหรับกลุ่มที่กำลังค้นหา สำหรับค่าความยาวของลำดับย่อยในรอบนั้น ๆ ถ้าคำนวณแล้วค่าประหยัดบิตที่ได้ติดลบ จะหยุดค้นหาโมทีฟทันที โดยค่าประหยัดบิตที่ได้อ้างอิงจากสมการ (12)

บรรทัดที่ 6 สำหรับกรณีที่ค่าประหยัดบิตยังคงเป็นค่าบวกอยู่ การค้นหาก็จะยังคงดำเนินต่อไปตราบเท่าที่ความยาวเป็น l และค่าการประหยัดบิตยังไม่ติดลบ โดยฟังก์ชัน AddAllNeighbor จะทำการค้นหาลำดับย่อยเพื่อนบ้าน (Neighbor) ที่คล้ายกับลำดับย่อยเฉลี่ยระหว่างคู่ของลำดับย่อยเริ่มต้นที่พิจารณาในรอบนั้น ๆ อ้างอิงตามสมการ (13)

บรรทัดที่ 7 เพื่อที่จะหาคู่โมทีฟที่มีความคล้ายคลึงกันมากที่สุด ฟังก์ชัน UpdateResult จะทำการรวมกลุ่มโมทีฟเพื่อให้ได้ชุดโมทีฟล่าสุด โดยหาได้จากการคำนวณค่าการประหยัดบิตของกลุ่มลำดับย่อย อ้างอิงตามสมการ (14) ในกรณีที่ลำดับย่อยซ้อนทับกัน (Overlapping) จะทำการละลำดับย่อยที่ให้ค่าประหยัดบิตต่ำกว่าทิ้งไป

บรรทัดที่ 8 และ 9 เป็นเงื่อนไขหยุดการดำเนินการ และคืนค่าโมทีฟที่ได้

บทที่ 3 แนวคิดและวิธีดำเนินการ

แนวคิดของงานวิจัยนี้ จะนำเสนออัลกอริทึมสำหรับการค้นพบโมทีฟและดิสคอร์ดเพื่อเพิ่มสมรรถนะของอัลกอริทึมเดิม โดยมี 2 ส่วนหลัก คือ

1.) สร้างเมทริกซ์โพรไฟล์แบบประมาณที่มีการเพิ่มสมรรถนะ โดยเริ่มจากการสร้างอัลกอริทึมที่สามารถลดมิติข้อมูลของอนุกรมเวลา เพื่อให้ข้อมูลมีขนาดที่เล็กลง จากนั้นใช้แนวคิดของปัญหาวันเกิด โดยประยุกต์ใช้ความรู้ในเรื่องความน่าจะเป็น และการประมาณค่าแฟคทอเรียลที่มีขนาดใหญ่โดยใช้การประมาณของสเตอร์ลิง เพื่อให้ได้จำนวนรอบสำหรับการสร้างโพรไฟล์ระยะทาง เพื่อใช้ในการค้นพบโมทีฟและดิสคอร์ดได้อย่างเหมาะสม เป็นการลดเวลาในการดำเนินการ โดยยังคงความถูกต้องของโมทีฟและดิสคอร์ดได้ใกล้เคียงกับวิธีเมทริกซ์โพรไฟล์

2.) เพื่อแก้ปัญหาการกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อย จึงได้วิธีการค้นพบโมทีฟและดิสคอร์ดที่มีการเพิ่มสมรรถนะ ผสานกับวิธีการการค้นพบโมทีฟความยาวที่เหมาะสม เพื่อให้ได้เอัลกอริทึมที่มีทั้งความเร็ว และความแม่นยำ อีกทั้งยังไม่จำเป็นต้องกำหนดความยาวของลำดับย่อย

3.1 การใช้ค่าเฉลี่ยตัวแทนอนุกรมเวลาเพื่อลดมิติของข้อมูล

สำหรับการค้นพบโมทีฟจะใช้ค่า $n = 2$ เนื่องจากการที่ค่า n เป็น 2 นั้นจะส่งผลให้ ข้อมูลอนุกรมเวลายังคงคุณสมบัติและคุณลักษณะเดิมได้มากที่สุดในขณะเดียวกันยังสามารถลดความสำคัญของข้อมูลที่มีสัญญาณรบกวนได้ แต่ถ้าหากใช้การลดมิติข้อมูลที่เยอะมากเกินไปโดยที่ $n > 2$ จะส่งผลให้ความเร็วในการดำเนินการมากขึ้น แต่ในทางกลับกันจะทำให้ข้อมูลอนุกรมเวลานั้นมีรูปร่างที่แตกต่างจากเดิมมาก ส่งผลต่อการค้นพบโมทีฟและดิสคอร์ดที่ไม่แม่นยำ ดังนั้นในงานนี้จึงเลือกใช้ค่า $n = 2$ ซึ่งเป็นการหาค่าเฉลี่ยเคลื่อนที่ 2 จุด มีนิยามดังนี้

$$\bar{T}_{\left\lfloor \frac{i+1}{2} \right\rfloor} = \frac{T_i + T_{i+1}}{2}$$

เมื่อ T_i คือ ค่าของอนุกรม T ที่จุด i และ $\bar{T}_{\left\lfloor \frac{i+1}{2} \right\rfloor}$ คือ ค่าเฉลี่ยระหว่างค่าของอนุกรม T ที่จุด i และค่าเฉลี่ยระหว่างค่าของอนุกรม T ที่จุด $i + 1$ โดยที่ i เป็นเซตของจำนวนเต็มบวกที่เป็นเลขคี่ โดยค่าเฉลี่ยตัวแทนอนุกรมเวลาแสดงดังตารางที่ 6

ตารางที่ 6 การคำนวณค่าเฉลี่ยเคลื่อนที่ 2 จุด

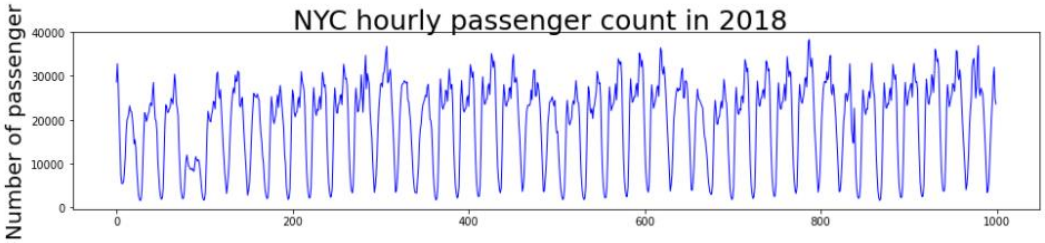
Procedure Downsampling(T, m)

Input:
T: a time series sequence
M: a motif length

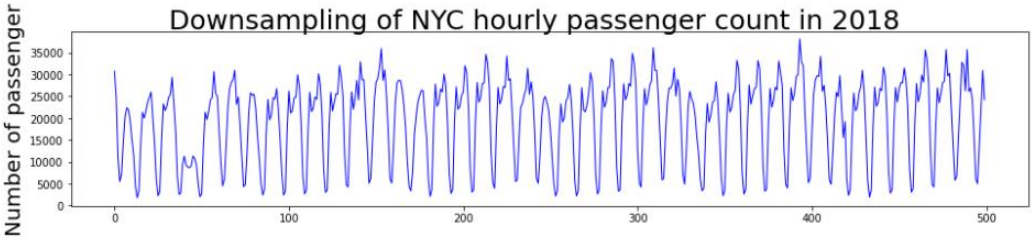
Output:
$\bar{T}$: a downsampled time series $\bar{T}$
n: a downsampled time series length
m: a downsampled subsequence length

1: $n \longleftarrow \text{Len}(\bar{T})$
2: $idxes \longleftarrow \text{range}(1, N)$
3: for i in range(N) :
4: $\quad \bar{T} \longleftarrow (T_i + T_{i+1})/2$
5: $\quad n \longleftarrow \text{Len}(\bar{T})$
6: $\quad m \longleftarrow \lfloor M/2 \rfloor$
7: return $\bar{T}, n, m$

ตัวอย่างการลดมิติข้อมูล ของข้อมูลจำนวนผู้โดยสารแท็กซี่ในเมืองนิวยอร์ก ในปี 2018 โดยข้อมูลตั้งต้นทั้งหมด 1,000 จุดข้อมูล จากนั้นข้อมูลตั้งต้นเดิมในภาพที่ 12 ได้ถูกลดมิติของข้อมูล ด้วยวิธีการหาค่าเฉลี่ยเคลื่อนที่ 2 จุด เป็น 500 จุดข้อมูล แสดงดังภาพที่ 13



ภาพที่ 12 แสดงข้อมูลจำนวนผู้โดยสารแท็กซี่ในเมืองนิวยอร์ก ในปี 2018 โดยมีจุดข้อมูลทั้งหมด 1,000 จุดข้อมูล



ภาพที่ 13 แสดงข้อมูลจำนวนผู้โดยสารแท็กซี่ในเมืองนิวยอร์ก ในปี 2018 ได้ถูกลดมิติของข้อมูล ด้วยวิธีการหาค่าเฉลี่ยเคลื่อนที่ 2 จุด เป็น 500 จุดข้อมูล

3.2 เมทริกซ์โพรไฟล์แบบประมาณด้วยวิธีการของสเตอร์ลิง (Approximated Matrix Profile using Stirling's Approximation)

เพื่อเป็นการลดเวลาสำหรับการดำเนินการแต่ยังคงความถูกต้องใกล้เคียงกับคำตอบแบบแม่นตรง เมทริกซ์โพรไฟล์แบบประมาณที่มีการเพิ่มสมรรถนะ จะทำการหาจำนวนรอบที่เหมาะสมในการสร้างโพรไฟล์ระยะทาง จากนั้นอัลกอริทึมจะค้นหาค่าโพรไฟล์ระยะทางที่มีค่าน้อยที่สุดสำหรับโมทีฟและมีค่าที่มากที่สุดสำหรับดิสคอร์ด โดยจะอธิบายแนวคิดหลักซึ่งเป็นส่วนสำคัญของสมมติฐานนี้ ซึ่งถูกแบ่งออกเป็น 2 ส่วนย่อย ตามสมมติฐานที่ได้ตั้งไว้ ดังนี้ ส่วนแรกจะกล่าวถึงลำดับสำหรับการคำนวณเมทริกซ์โพรไฟล์ (Order of Computation) และส่วนที่สองกล่าวถึงจำนวนลำดับย่อยหรือก็คือจำนวนรอบของการวนซ้ำที่เหมาะสมสำหรับการคำนวณเมทริกซ์โพรไฟล์ (Number of Iterations)

3.2.1 ลำดับสำหรับการคำนวณเมทริกซ์โพรไฟล์ (Order of Computation) [13]

สำหรับการเลือกลำดับย่อยมาคำนวณ เพื่อวัดความคล้ายคลึงกันของแต่ละคู่ลำดับย่อยนั้นได้มาจากการสุ่ม เป็นผลให้อัลกอริทึมแสตมป์ (ตารางที่ 4) เป็นอัลกอริทึมแบบ ณ เวลาใด ๆ (Anytime algorithm) กล่าวคือ เมื่ออัลกอริทึมแสตมป์สามารถดำเนินการจนเสร็จสิ้นกระบวนการแล้วจะคืนค่าโมทีฟและดิสคอร์ดแบบแม่นตรงได้ หรือสามารถหยุดก่อนที่การดำเนินการจะเสร็จสิ้นได้โดยจะคืนค่าโมทีฟและดิสคอร์ดแบบประมาณได้ แต่ไม่สามารถระบุได้ว่าจะต้องใช้การวนรอบเป็นจำนวนกี่รอบจึงจะเพียงพอสำหรับการค้นพบโมทีฟ จึงเป็นที่มาในการนำเสนอจำนวนรอบในการวนซ้ำที่เพียงพอโดยไม่จำเป็นต้องใช้การวนรอบทั้งหมดในการดำเนินการ ดังนั้นการคำนวณหาจำนวนรอบของการวนซ้ำ

ที่เหมาะสม จะทำให้ค่าที่หาได้เข้าใกล้ผลลัพธ์ได้เร็วขึ้นและความถูกต้องยังคงใกล้เคียงเดิม แนวคิดที่นำเสนอคือ

1) ทำการสุ่มเลือกสำหรับลำดับย่อยแรก
2) สำหรับลำดับย่อยถัดไปที่จะถูกนำมาใช้คำนวณ จะได้จากการคำนวณโพรไฟล์ระยะทาง (Distance Profile) ซึ่งเป็นลำดับย่อยที่เป็นเพื่อนบ้านใกล้ที่สุดของลำดับย่อยแรก
3) ทำการสุ่มลำดับย่อยตัวใหม่ถ้าลำดับย่อยเพื่อนบ้านใกล้สุดถูกนำมาใช้ในการคำนวณแล้ว

3.2.2 จำนวนลำดับย่อยสำหรับการคำนวณเมทริกซ์โพรไฟล์ หรือจำนวนรอบของการวนซ้ำ (Number of Iterations)

สำหรับอัลกอริทึมแสตมป์จะมีการคำนวณรอบการวนซ้ำในเมทริกซ์โพรไฟล์ทั้งหมด มีจำนวนรอบเป็น $n-m+1$ รอบ โดยที่ความยาวของอนุกรมเวลา คือ n และความยาวของลำดับย่อย คือ m สำหรับอัลกอริทึมแสตมป์จำเป็นที่จะต้องใช้ลำดับย่อยทั้งหมดในการคำนวณ เพื่อให้ได้โมทีฟแบบแม่นตรง ซึ่งถ้าอนุกรมเวลามีขนาดใหญ่ จะใช้เวลานานมาก ผู้วิจัยจึงนำเสนอวิธีในการหาจำนวนของลำดับย่อย $\tilde{k}$ ที่เหมาะสม โดยที่ $\tilde{k} < n-m+1$ ส่งผลให้จำนวนการสร้างโพรไฟล์ระยะทางซึ่งจะแปรตามจำนวนรอบการค้นนั้นไม่จำเป็นต้องมีการสร้างทุก ๆ ลำดับย่อยในอนุกรมเวลา หรือก็คือวนซ้ำแค่เพียง $\tilde{k}$ รอบ ไม่จำเป็นต้องวนซ้ำถึง $n-m+1$ รอบ

มุมมองสำหรับการค้นพบโมทีฟเปรียบเสมือนการเลือกคู่ของลำดับย่อยที่มีความคล้ายคลึงกันมากที่สุด กล่าวคือ สามารถวัดได้จากการหาระยะห่างน้อยที่สุดจากลำดับย่อยที่เป็นไปได้ทั้งหมด ดังนั้นปัญหาการค้นพบโมทีฟสามารถมองแบบปัญหาวันเกิดได้ [13]_ (Birthday Paradox) กล่าวคือ

1) วันทั้งหมด 365 วันที่เป็นไปได้หมายถึง จำนวนระยะห่างระหว่างลำดับย่อยและลำดับย่อยเพื่อนบ้านทั้งหมดที่เป็นไปได้ ซึ่งมีค่าเท่ากับ $n-m+1$
2) ความน่าจะเป็นที่ลำดับย่อย 2 ลำดับย่อยใด ๆ จะเป็นโมทีฟ เทียบเท่ากับ ความน่าจะเป็นที่คน 2 คนจะเกิดวันและเดือนเดียวกัน
3) จำนวนลำดับย่อยที่เพียงพอจะนำมาคิด สำหรับการค้นพบโมทีฟด้วยความน่าจะเป็น p คือจำนวนคนที่เพียงพอด้วยความน่าจะเป็น p เพื่อยืนยันได้ว่ามีคนเกิดวันและเดือนเดียวกัน

เพราะฉะนั้นงานวิจัยนี้ได้นำเสนอการหาจำนวนลำดับย่อยที่เพียงพอ สำหรับการค้นพบโมทีฟและดิสคอร์ด โดยมีการใช้มุมมองปัญหาวันเกิด ผนวกเข้ากับทฤษฎีการประมาณค่าแฟคทอเรียลขนาดใหญ่ของสเตอร์ลิง ดังแสดงในตารางที่ 7 เพื่อสร้างอัลกอริทึมการหาจำนวนลำดับย่อย $\tilde{k}$ ที่เหมาะสม (FindProperK Algorithm) แทน (FindK Algorithm) ในงานวิจัย [13] สำหรับอัลกอริทึม FindProperK ที่แสดงด้านล่างนี้ มีข้อมูลนำเข้าเป็น ข้อมูลอนุกรมเวลา T ที่มีขนาด n ลำดับย่อยขนาด m และค่าความน่าจะเป็น p ในงานวิจัยนี้จะกำหนดค่า $p = 0.999$ โดยค่านำออก คือ จำนวนรอบของการวนซ้ำ $\tilde{k}$

ตารางที่ 7 จำนวนลำดับย่อยที่เหมาะสม $\tilde{k}$ สำหรับการวนรอบของการคำนวณเมทริกซ์โพรไฟล์

Procedure FindProperK(n, m, p)

Input:
n: a length of time series T
m: a length of a given subsequence
p: a user-defined probability

Output:
$\tilde{k}$: a maximum number of iterations

1: $\tilde{k} \leftarrow 1$
2: $probability \leftarrow 0$ // Initial probability is 0
3: **while** $probability < p$:
4: $\quad probability \leftarrow CalOptProb(\tilde{k}, n, m)$
5: $\quad \tilde{k} = \tilde{k} + 1$
6: **end while** // End while loop when probability is greater than p
7: **return** $\tilde{k} - 1$

อธิบายวิธีการคำนวณสำหรับการหาจำนวนการวนรอบที่เหมาะสม $\tilde{k}$ รอบ ด้วยค่าความน่าจะเป็นที่กำหนดไว้แล้วด้วยค่า p แสดงดังบรรทัดที่ 4 ของตารางที่ 7

$$CalProperProb(\tilde{k}, n, m) = 1 - \left(\frac{\sqrt{2\pi}}{e^{\tilde{k}+1}}\right)\left(\frac{n-m}{n-m-\tilde{k}}\right)^{n-m-\tilde{k}+0.5}$$

สำหรับสมการ $CalProperProb(\tilde{k}, n, m) = 1 - \left(\frac{\sqrt{2\pi}}{e^{\tilde{k}+1}}\right)\left(\frac{n-m}{n-m-\tilde{k}}\right)^{n-m-\tilde{k}+0.5}$ มีที่มาจากการคำนวณหาความน่าจะเป็นโดยใช้ทฤษฎีปัญหาวันเกิดซึ่งอยู่ในรูปของการคำนวณของแฟคทอเรียล และประยุกต์ใช้การประมาณของสเตอร์ลิงมาประมาณค่าของแฟคทอเรียล โดยเลือกใช้ขอบเขตบนและขอบเขตล่างจากการประมาณของสเตอร์ลิงตามทฤษฎีในบทที่ 2

3.2.3 อัลกอริทึมเมทริกซ์โพรไฟล์แบบประมาณด้วยค่า $\tilde{k}$ ที่เหมาะสม (Approximated Matrix Profile with Proper $\tilde{k}$)

จากแนวคิดของเรื่องลำดับการคำนวณ และจำนวนรอบของการวนซ้ำ ผู้วิจัยได้มีการปรับปรุงและพัฒนาเพิ่มเติมจากอัลกอริทึมเอเอ็มพี (Approximated Matrix Profile) [13] และอัลกอริทึมแสตมป์เพื่อสร้างอัลกอริทึมที่มีการเพิ่มสมรรถนะ สำหรับการหาจำนวนรอบการค้นโพรไฟล์ระยะทางในเมทริกซ์โพรไฟล์ (Enhanced Approximated Matrix Profile) ดังแสดงในตารางที่ 8 โดยสามารถหาค่า $\tilde{k}$ ที่เหมาะสมด้วยแนวคิดการประมาณค่าแฟคทอเรียลขนาดใหญ่ผ่านวิธีการประมาณค่าของสเตอร์ลิง ซึ่งการประมาณค่านี้ มีความใกล้เคียงกับค่าจำนวนการวนรอบที่มาจากการประมาณโดยใช้อนุกรมเทเลอร์อันดับที่หนึ่งของอัลกอริทึมเอเอ็มพี แต่แตกต่างตรงที่จำนวนการวนรอบเป็นค่า $\tilde{k}$ ค่าใหม่ที่เหมาะสมมากกว่า ในแง่ของเวลาที่เร็วขึ้น อีกทั้งความแม่นยำไม่ต่างจากอัลกอรึทึมเอเอ็มพี อัลกอริทึมดังกล่าว คือ อัลกอริทึมเอเอ็มพีเอสเอ (Approximated Matrix Profile using Stirling's Approximation-AMPSA) ซึ่งมีความซับซ้อนเชิงเวลา $O(\tilde{k}nlog(n))$

ตารางที่ 8 อัลกอริทึม AMPSA

Procedure AMPSA($\bar{T}$, n, m, p)

Input:
$\bar{T}$: a time series $\bar{T}$ of length n
m: a length of a given subsequence
p: a user-defined probability

Output:
P: an updated matrix profile
I: an associated updated matrix profile index

1: $P \leftarrow infinity$ // initialize all values in P to infinity
2: $I \leftarrow zero$ // initialize all values in I to zero
3: $idxes \leftarrow range(1, n-m+1)$
4: $new\_idx \leftarrow RandomShuffle(idxes)$
5: $\tilde{k} \leftarrow$ **$FindOptimal(n, m, p)$**
6: **for** i **in** $range(\tilde{k})$:
7: $idx \leftarrow new\_idx(1)$ // uses 1st index
8: $\bar{D} \leftarrow MASS(\bar{T}_{idx}, \bar{T})$ // $\bar{T}_{idx}$ is a query subseq of $\bar{T}$ index idx
9: $new\_idx.remove(new\_idx(1))$ // remove used index
10: $\bar{P}, \bar{I} \leftarrow UpdateMinElementwise(\bar{P}, \bar{I}, \bar{D}, idx)$
11: $P \leftarrow \bar{P}_{idx} * 2, I \leftarrow \bar{I}_{idx} * 2$
12: **return** P, I

อัลกอรึทึม AMPSA เป็นการคำนวณจำนวนรอบของการวนซ้ำสำหรับการสร้างโพรไฟล์ระยะทางโดยบางส่วนของอัลกอริทึมมีการดัดแปลงจากงานวิจัย [1] และ [13] เพื่อนำไปสร้างเมทริกซ์โพรไฟล์แบบประมาณ โดยข้อมูลนำเข้าของอัลกอริทึมนี้คือข้อมูลอนุกรมเวลา $\bar{T}$ ซึ่งเป็นข้อมูลอนุกรมเวลา T ที่ถูกนำมาหาค่าเฉลี่ยเคลื่อนที่แล้ว ที่ความยาวของลำดับย่อย m สำหรับ $\bar{T}$ กล่าวคือ ถ้าต้องการทราบความยาวของลำดับย่อยใน T ค่าความยาวของลำดับย่อยนั้นจะมีค่า $m * 2$ และค่าความน่าจะเป็น p โดยในงานวิจัยนี้กำหนดให้ $p = 0.999$ โดยอัลกอริทึมจะคืนค่า P และ I ซึ่งคือเมทริกซ์โพรไฟล์แบบประมาณ และเมทริกซ์โพรไฟล์อินเด็กซ์ ตามลำดับ สำหรับ T ถ้าต้องการทราบค่า P และ I ของ T แล้ว ค่า P และ I ของ T คือ $\bar{P} * 2$ และ $\bar{I} * 2$ ตามลำดับ

บรรทัดที่ 1-2 เป็นการกำหนดค่าเริ่มต้นของเวกเตอร์ที่มีความยาว $n - m + 1$ สำหรับเมทริกซ์โพรไฟล์แบบประมาณ P ทุกตำแหน่งเป็นอนันต์ (Infinity) และเมทริกซ์โพรไฟล์อินเด็กซ์ I เป็นเวกเตอร์ที่มีความยาวเท่ากับ P โดยกำหนดให้ค่าเริ่มต้นแต่ละค่าใน I คือศูนย์

บรรทัดที่ 3 เป็นการสร้างช่วงความยาวสำหรับใช้ในการคำนวณขั้นถัดไป

บรรทัดที่ 4 ทำการสุ่มเลขที่ได้จากบรรทัดที่ 3 แล้วนำมาทำการเรียงสับเปลี่ยน โดยจะเลือกลำดับย่อยแรกแล้วเลือกเรียงไปตามลำดับย่อยที่สุ่มได้จนครบจำนวนการวนซ้ำ ซึ่งรอบของการวนซ้ำถูกกำหนดจากอัลกอริทึม $FindOptimal$ ในบรรทัดที่ 5

บรรทัดที่ 5 อัลกอริทึม $FindOptimal$ เป็นการคำนวณหาจำนวนรอบการวนซ้ำ $\tilde{k}$ โดยกำหนดค่าความน่าจะเป็น $p = 0.999$

บรรทัดที่ 6-10 ทำการคำนวณสร้างโพรไฟล์ระยะทางตามจำนวนรอบที่เหมาะสม เพื่อทำการสร้างเมทริกซ์โพรไฟล์แบบประมาณที่มีการเพิ่มสมรรถนะ และเมทริกซ์โพรไฟล์อินเด็กซ์ ซึ่งมีจำนวนรอบของการวนซ้ำทั้งหมด $\tilde{k}$ รอบ โดยในแต่ละรอบจะใช้ลำดับย่อยที่มาจากการเรียงสับเปลี่ยนกำหนดโดยบรรทัดที่ 4 และจำนวนรอบในการคำนวณกำหนดโดยบรรทัดที่ 5

บรรทัดที่ 8 เป็นการคำนวณโพรไฟล์ระยะทางโดยใช้อัลกอริทึมแมส เมื่อ $T_{idx,m}$ คือลำดับย่อยในอนุกรมเวลา T ที่มีตำแหน่งเริ่มต้นที่ idx และมีความยาว m โดยมีเงื่อนไขว่า $Dist_i = \infty$ เมื่อ i มีค่าอยู่ในช่วงระหว่าง $idx - \frac{m}{2}$ และ $idx + \frac{m}{2}$ เพื่อป้องกันการเกิดลำดับย่อยที่มีการซ้อนทับกัน (Trivial Match)

บรรทัดที่ 9 ลำดับย่อยที่ถูกเลือกไปใช้แล้วจะมีการนำออกไป และจะนำค่าถัดไปที่ได้มาใช้ในงานต่อ ทำเช่นนี้ไปเรื่อย ๆ จนกว่าจะครบจำนวนรอบการวนซ้ำที่ถูกกำหนดโดยบรรทัดที่ 5

บรรทัดที่ 10 ฟังก์ชัน $UpdateMinElementwise$ จะทำการเปรียบเทียบค่าโพรไฟล์ระยะทางเป็นคู่ ๆ เพื่อหาค่าที่น้อยกว่าระหว่างเวกเตอร์ $\overline{D}$ และเวกเตอร์ $\bar{P}$ กล่าวคือ $\bar{P}_i = min(\bar{P}_i,\ \overline{D}_i)$ ตั้งแต่ $i = 1$ ถึง ความยาวของเวกเตอร์ $\overline{D}$ จากนั้นทำการปรับเวกเตอร์ $\bar{I}$ ที่สัมพันธ์กับ $\overline{D}$ ด้วยค่า idx กล่าวคือ $\bar{I}_i = idx$ เมื่อ $\overline{D}_i \leq \bar{P}_i$ ทุก ๆ ค่า i ที่ทำการปรับในเวกเตอร์ $\bar{P}$ ในรอบก่อนหน้านี้

บรรทัดที่ 11 เพื่อที่จะหาค่า P และ I โดยประมาณสำหรับ T แล้ว จะได้ว่า ค่า P และ I ของ T คือ $\bar{P} * 2$ และ $\bar{I} * 2$ ตามลำดับ

บรรทัดที่ 14 คืนค่า P และ I ซึ่งเป็นเมทริกซ์โพรไฟล์แบบประมาณ และเมทริกซ์โพรไฟล์อินเด็กซ์ ตามลำดับ

3.3 อัลกอริทึมสำหรับการค้นพบโมทีฟสำหรับความยาวที่เหมาะสม โดยใช้เมทริกซ์โพรไฟล์แบบประมาณที่มีโดยวิธีการประมาณแบบสเตอร์ลิง (Proper Length Motif Discovery using Approximated Matrix Profile by Stirling's Approximation-PLAMPSA)

ปัญหาสำหรับการกำหนดพารามิเตอร์ความยาวของลำดับย่อย โดยปกติแล้วนั้น จะใช้ผู้เชี่ยวชาญเฉพาะด้าน มาเป็นผู้กำหนดความยาวของลำดับย่อย เพราะความยาวของลำดับย่อยแต่ละค่า จะแสดงถึง คุณลักษณะเฉพาะของลำดับย่อยนั้น ๆ ที่แตกต่างกันไป แต่ถ้าหากจำเป็นต้องหาความยาวของลำดับย่อยที่เหมาะสมโดยไม่ต้องกำหนดพารามิเตอร์ความยาวของลำดับย่อย สามารถทำได้ด้วยวิธีการจากงานวิจัย [12] แต่เนื่องจากว่าอัลกอริทึมในการหาโมทีฟในปัจจุบันใช้เวลานานมาก อีกทั้งอัลกอริทึม [12] ยังใช้เวลานาน เนื่องจากยังไม่มีการลดมิติข้อมูลอนุกรมเวลา อีกทั้งจำนวนการวนรอบการค้นเพื่อหาโพรไฟล์ระยะทางยังมากเกินไปสำหรับการค้นพบโมทีฟและดิสคอร์ด ดังนั้นผู้วิจัยจึงได้นำเสนออัลกอริทึมสำหรับการค้นพบโมทีฟที่ความยาวเหมาะสม โดยใช้เมทริกซ์โพรไฟล์แบบประมาณที่มีการเพิ่มสมรรถนะมาผสานกับอัลกอริทึมการหาโมทีฟความยาวที่เหมาะสม หรืออัลกอริทึมพีแอลเอเอ็มพีเอสเอ (**P**roper **L**ength Motif Discovery using **A**pproximated **M**atrix **P**rofile by **S**tirling's **A**pproximation-**PLAMPSA**)

อัลกอริทึมพีแอลเอเอ็มพีเอสเอจะทำการลดมิติข้อมูล ก่อนจะนำไปคำนวณด้วยจำนวนการวนรอบที่เหมาะสมแสดงในบรรทัดที่ 4 และบรรทัดที่ 5 ของอัลกอริทึม PLAMPSA และผสานกับวิธีการค้นพบโมทีฟความยาวเหมาะสม [12] ดังแสดงในตารางที่ 9

ตารางที่ 9 อัลกอริทึม PLAMPSA (ที่มา: [12])

Procedure PLAMPSA(T)

Input:
T: a time series T of length n

Output:
$Motif$: a proper length of motif discovery

1: $totalbitsave = -inf,\ numofmotif = 0,\ properlength = 0$
2: $Motif \leftarrow emptyset$
3: **for** $l = 1\% * Len(T)$ **to** $40\% * Len(T)$
4: $\bar{T}, n, l \leftarrow \boldsymbol{Downsampling}(T, l)$
5: $P, I \leftarrow \boldsymbol{AMPSA}(\bar{T}, l, p)$ // *provide probability* $p = 0.999$
6: $(A, B) \leftarrow FindClosestSimilarityPair(P, I)$
7: $G \leftarrow CreateGroup(A, B)$
8: **if** $Bitsave(G) < 0$ **then break**
9: **while** $Bitsave(G) > 0$
10: $G \leftarrow AddNeighbor(G, P, I)$
11: **end while**
12: $cost \leftarrow TotalBitsave(G)$
13: **if** $Len(G) > numofmotif$
14: $numofmotif \leftarrow Len(G),\ properlength \leftarrow l,$
$totalbitsave \leftarrow cost,\ Motif \leftarrow (A, B)$
15: **elseif** $Len(G) = numofmotif$ **and** $cost > totalbitsave$
16: $numofmotif \leftarrow Len(G),\ properength \leftarrow l,$
$totalbitsave \leftarrow cost,\ Motif \leftarrow (A, B)$
17: **end for**
18: **return** $Motif,\ properlength$

อัลกอรึทึม PLAMPSA ใช้สำหรับการค้นหาโมทีฟที่คล้ายคลึงกันมากที่สุดโดยได้ความยาวที่เหมาะสม ในอัลกอริทึมนี้จะทำการค้นหาโมทีฟที่มีความยาวตั้งแต่ 1% จนถึง 40% ของความยาวอนุกรมเวลา T ที่สนใจ (สามารถเปลี่ยนช่วงความยาวที่ค้นหาได้ตามความเหมาะสม) ซึ่งเห็นได้ว่าไม่จำเป็นที่จะต้องกำหนดพารามิเตอร์ความยาวของลำดับย่อยในการค้นหาสำหรับข้อมูลอนุกรมเวลา T

ซึ่งสามารถแก้ปัญหาการกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อยได้ เพราะโดยปกติแล้วในการค้นพบโมทีฟ จำเป็นจะต้องมีการกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อย

บรรทัดที่ 1 กำหนดค่าเริ่มต้นให้ตัวแปร $totalbitsave$ ซึ่งเป็นตัวแปรที่บ่งบอกค่าประหยัดบิตโดยจะทำการเลือกความยาวของโมทีฟที่เหมาะสมจากการเปรียบเทียบค่าประหยัดบิต ตัวแปรถัดมา คือ $numbermotif$ และ $properlength$ เป็นจำนวนโมทีฟ และ ความยาวโมทีฟที่เหมาะสมตามลำดับ กำหนดตัวแปรทั้งสองมีค่าเริ่มต้นเป็นศูนย์

บรรทัดที่ 2 กำหนดให้ตัวแปร $Motif$ มีค่าเริ่มต้นเป็นเซตว่าง

บรรทัดที่ 3-17 ทำการค้นหาโมทีฟความยาวที่เหมาะสม โดยผ่านค่าพารามิเตอร์ความยาวตั้งแต่ 1% จนถึง 40% ของความยาวของอนุกรมเวลา โดยใช้อัลกอริทึม AMPSA ผสานกับอัลกอริทึม Proper Length Motif Discovery จากงานวิจัย [12]

บรรทัดที่ 4 เป็นการลดมิติข้อมูลโดยการหาค่าเฉลี่ยเคลื่อนที่ ซึ่งผลลัพธ์ที่ได้คืออนุกรมเวลาและลำดับย่อยสำหรับค้นหาโมทีฟที่ถูกลดมิติข้อมูลลงไปครึ่งหนึ่ง

บรรทัดที่ 5 อัลกอริทึม AMPSA ทำการคำนวณเมทริกซ์โพรไฟล์แบบประมาณ โดยมีการใช้จำนวนการวนรอบ $\tilde{k}$ เมื่อผ่านการคำนวณแล้วจะคืนค่าเมทริกซ์โพรไฟล์และเมทริกซ์โพรไฟล์อินเด็กซ์

บรรทัดที่ 6 ฟังก์ชัน $FindClosestSimilarityPair$ จะทำการค้นหาคู่ของลำดับย่อยจากค่าระยะห่างน้อยที่สุด 2 อันดับ กล่าวคือ คู่ลำดับย่อยที่มีความคล้ายคลึงกันมากที่สุด 2 อันดับ โดยที่ลำดับย่อยคู่นั้นจะต้องไม่มีส่วนที่ซ้อนทับกัน

บรรทัดที่ 7 ทำการสร้างกลุ่มของลำดับย่อย จากบรรทัดที่ 6 ที่เป็นคู่ลำดับย่อยเริ่มต้น

บรรทัดที่ 8 เงื่อนไขการหยุดในรอบนั้น ถ้าค่าประหยัดบิตของกลุ่มเริ่มต้นน้อยกว่าศูนย์

บรรทัดที่ 9-11 เป็นเงื่อนไขการเลือกลำดับย่อยเพื่อนบ้าน (Neighbor) เพื่อเพิ่มเข้าไปในกลุ่มตามค่าความยาวและค่าการประหยัดบิตของลำดับย่อยในรอบนั้น ๆ โดยจะพิจารณาจากค่าประหยัดบิตที่มีค่ามากกว่าศูนย์

บรรทัดที่ 10 ฟังก์ชัน $AddNeighbor$ ทำหน้าที่เพิ่มลำดับย่อยเพื่อนบ้านเข้าไปในกลุ่ม G ครั้งละหนึ่งตัว โดยการเพิ่มลำดับย่อยนั้นจะต้องเป็นลำดับย่อยที่ไม่มีการซ้อนทับกัน ซึ่งถ้าลำดับย่อยเพื่อนบ้านถูกเพิ่มเข้ามาแล้ว ให้ทำการพิจารณาว่า ค่าประหยัดบิตที่ได้ยังคงเป็นบวกอยู่หรือไม่ ถ้า

ยังคงเป็นบวกอยู่ก็เพิ่มลำดับย่อยเพื่อนบ้านที่มีค่าน้อยสุดในอันดับถัด ๆ มา คือ 3, 4, 5, ... ไปเรื่อย ๆ เพราะว่าตอนเริ่มต้นค่าที่น้อยที่สุด 2 อันดับ ในเมทริกซ์โพรไฟล์แบบประมาณถูกใช้ในการสร้างกลุ่มตอนเริ่มต้นแล้ว จะเห็นว่าไม่จำเป็นต้องคำนวณระยะห่างระหว่างคู่ลำดับย่อยใหม่ สำหรับการเพิ่มลำดับย่อยเพื่อนบ้านในส่วนนี้

บรรทัดที่ 12 เป็นการคำนวณค่าประหยัดบิตรวมของกลุ่ม G เมื่อไม่สามารถเพิ่มลำดับย่อยเพื่อนบ้านได้แล้ว และทำการเก็บค่าประหยัดบิตรวมไว้ในตัวแปร $cost$

บรรทัดที่ 13 และ 15 เป็นการตรวจสอบค่าความยาวที่เหมาะสมของลำดับย่อยในรอบการพิจารณานั้น โดยเมื่อกลุ่มของโมทีฟ G ใด ๆ มีจำนวนสมาชิกมากกว่าแสดงถึงความสามารถในการเป็นตัวแทนได้ดีกว่า สำหรับกรณีจำนวนสมาชิกในกลุ่มเท่ากันจะเลือกโมทีฟที่มีค่าประหยัดบิตรวมสูงสุด

บรรทัดที่ 14 และ 16 ทำการปรับเปลี่ยนค่าตัวแปรให้เป็นปัจจุบัน เมื่อเป็นไปตามเงื่อนไขการตรวจสอบในบรรทัดที่ 13 และ 15

ในบทที่ 3 กล่าวถึงแนวคิดและวิธีดำเนินการ เพื่อสร้างอัลกอริทึมเอเอ็มพีเอสเอ ซึ่งมีคุณสมบัติในการลดมิติของข้อมูลอนุกรมเวลา และลดจำนวนการวนรอบ สำหรับการค้นพบโมทีฟและดิสคอร์ด โดยใช้ทฤษฎีทางคณิตศาสตร์เรื่องกฎการนับและความน่าจะเป็น มาประยุกต์ใช้กับการสร้างจำนวนการวนรอบที่เหมาะสมเพื่อค้นหาโมทีฟและดิสคอร์ด อีกทั้งสร้างอัลกอริทึม พีแอลเอเอ็มพีเอสเอจากการประยุกต์ใช้อัลกอริทึมเอเอ็มพีเอสเอ และคุณสมบัติของอัลกอริทึมค้นหาโมทีฟที่ความยาวเหมาะสม [12] เพื่อค้นหาโมทีฟได้ที่ความยาวเหมาะสม โดยไม่จำเป็นต้องกำหนดพารามิเตอร์ความยาวของลำดับย่อย อีกทั้งยังสามารถค้นหาโมทีฟที่ความยาวเหมาะสมได้โดยใช้เวลาน้อยลงเป็นอย่างมาก ถัดไปจะเป็นบทที่ 4 จะกล่าว ถึงการทดลองและวิเคราะห์ผลการทดลองของอัลกอริทึมที่สร้างขึ้นกับข้อมูลจริง และข้อมูลที่ถูกสร้างขึ้น เพื่อประเมินในเรื่องความถูกต้อง และเวลาที่ใช้ในการดำเนินการ

บทที่ 4 การทดลองและวิเคราะห์ผลการทดลอง

สำหรับบทนี้จะเกี่ยวข้องกับการวัดผลการทดลองเพื่อประเมินคุณภาพของงานที่นำเสนอ โดยแบ่งส่วนพิจารณาออกเป็นสองส่วนหลัก คือ การทดลองอัลกอริทึมเอเอ็มพีเอสเอ (AMPSA) เพื่อเปรียบเทียบผลการทดลองเชิงเวลาในการดำเนินการกับอัลกอริทึมเอเอ็มพี (AMP) [13] ซึ่งทั้งสองอัลกอริทึมนี้ใช้สำหรับการคำนวณเมทริกซ์โพรไฟล์แบบประมาณและเปรียบเทียบความถูกต้องสำหรับการค้นพบโมทีฟและดิสคอร์ดกับอัลกอริทึมแสตมป์ (STAMP) [1] ซึ่งเป็นอัลกอริทึมสำหรับการคำนวณเมทริกซ์โพรไฟล์แบบแม่นตรง ส่วนถัดมาคือการทดลองอัลกอริทึมการค้นพบโมทีฟสำหรับความยาวที่เหมาะสม ซึ่งเรียกว่าอัลกอริทึม พีแอลเอเอ็มพีเอสเอ (PLAMPSA) ซึ่งเป็นการทำงานโดยประยุกต์ทั้งสองอัลกอริทึมหลัก คืออัลกอริทึมการค้นพบโมทีฟสำหรับความยาวที่เหมาะสม [12] ผสานเข้ากับอัลกอริทึมเอเอ็มพีเอสเอที่สร้างขึ้น โดยในงานวิจัยนี้ส่วนใหญ่จะเลือกใช้ข้อมูลที่มีขนาดใหญ่ เพื่อต้องการให้เห็นโดยชัดว่าเมื่อข้อมูลอนุกรมเวลายิ่งมีขนาดใหญ่ เวลาในการดำเนินการยิ่งใช้เวลามากสำหรับอัลกอริทึมเอเอ็มพีและอัลกอริทึมแสตมป์ แต่ถ้ามีการลดมิติของข้อมูลลงแล้วและใช้อัลกอริทึมเอเอ็มพีเอสเอก็จะใช้เวลาในการดำเนินการลดลงไปได้เป็นอย่างมาก

4.1 การทดลองสำหรับอัลกอริทึมเอเอ็มพีเอสเอ

สำหรับการทดลองในส่วนนี้จะเป็นส่วนของการค้นพบโมทีฟ และดิสคอร์ด ของอัลกอริทึมเอเอ็มพีเอสเอ เปรียบเทียบกับอัลกอริทึมเอเอ็มพี และอัลกอริทึมแสตมป์ ซึ่งเป็นการทดลองเพื่อเปรียบเทียบด้านเวลาและความถูกต้องตามลำดับ โดยจะทดสอบกับทั้งชุดข้อมูลจริง (Real World Data) และชุดข้อมูลสังเคราะห์ (Synthetic Data)

4.1.1 ข้อมูลจริง (Real World Data)

เพื่อความหลากหลาย และครอบคลุมของชุดข้อมูล ข้อมูลจริงที่ใช้สำหรับงานวิจัยนี้จะประกอบไปด้วยข้อมูลจากหลากหลายขอบเขต (Domain) โดยข้อมูลที่นำมาใช้ทดสอบจะมาจากข้อมูลจริงที่ถูกรวบรวมไว้โดยนักวิเคราะห์ข้อมูล และทำการอัพโหลดขึ้นบนฐานข้อมูลของแคกเกิล (Kaggle) และผู้วิจัยได้ทำการอ้างอิงที่มาของข้อมูลในส่วนท้ายของชื่อชุดข้อมูล ดังตารางที่ 10 เพื่อที่จะเปรียบเทียบความถูกต้องในการค้นพบโมทีฟ โดยเปรียบเทียบกับอัลกอริทึมแสตมป์ [1] สำหรับข้อมูลที่มีขนาดใหญ่โดยมีมากกว่า 200,000 จุดข้อมูล จะใช้เพียง 200,000 จุดข้อมูลแรก เนื่องจากอัลกอริทึมแสตมป์ใช้ระยะเวลาการดำเนินการเป็นเวลานาน ถ้าขนาดของอนุกรมเวลามีขนาดใหญ่ โดยมีรายละเอียดตามตารางที่ 10

ตารางที่ 10 รายละเอียดข้อมูลจริงที่ใช้ในงานวิจัย

ชื่อชุดข้อมูล	ความยาว	ความยาวที่ถูกลดมิติโดยอัลกอริทึมเอเอ็มพีเอสเอ	รายละเอียด
EEG	200,000	100,000	ข้อมูลคลื่นไฟฟ้าสมอง [27]
ECG	200,000	100,000	ข้อมูลคลื่นไฟฟ้าหัวใจ [28]
AEP	121,273	60,636	ข้อมูลการใช้ไฟฟ้า [18]
Seoul_Minute_PM2.5	200,000	100,000	ข้อมูล PM 2.5 [29]
Earthquake	200,000	100,000	ข้อมูลแผ่นดินไหว [30]
Hessi_solar_flare	113,942	56,971	ข้อมูลจำนวนครั้งการประทุบนดวงอาทิตย์ [31]
Weather_Szeged	96,452	48,226	ข้อมูลความชื้นสัมพัทธ์ [32]

4.1.2 ข้อมูลสังเคราะห์ (Synthetic Data)

เพื่อทดสอบข้อมูลที่มีความยาวที่หลากหลายของอนุกรมเวลา สำหรับข้อมูลสังเคราะห์ที่ใช้ในงานวิจัยนี้ เป็นข้อมูลการเดินแบบสุ่ม (Random Walk) ที่มีขนาดแตกต่างกันทั้งหมด 5 ชุด ซึ่งในอนุกรมเวลานี้จะมีบางส่วนที่ถูกแทนที่ด้วยลำดับย่อย (โมทีฟ) ในตำแหน่งที่ทราบแน่ชัด เพื่อใช้ทดสอบความถูกต้องแม่นยำในการค้นพบโมทีฟ โดยกำหนดพารามิเตอร์ความยาวของลำดับย่อย (โมทีฟ) คิดเป็น 5% ของความยาวข้อมูลอนุกรมเวลา แสดงดังตารางที่ 11

สำหรับการทดลองนี้ ใช้การเพิ่มข้อมูลด้วยการเดินแบบสุ่ม เพื่อลดโอกาสในการค้นพบโมทีฟหรือดิสคอร์ดที่เกิดขึ้นภายในข้อมูลสุ่ม ซึ่งไม่ใช่โมทีฟหรือดิสคอร์ดที่แท้จริง

ตารางที่ 11 รายละเอียดข้อมูลสังเคราะห์ที่ใช้ในงานวิจัย

ชื่อชุดข้อมูล	RW1	RW2	SyntheticData1	SyntheticData2	SyntheticData3
ความยาวข้อมูล	10,000	50,000	100,000	200,000	500,000

4.1.3 การวิเคราะห์ความซับซ้อนเชิงเวลาและหน่วยความจำที่ใช้ในการประมวณผล (Time and Space Complexity)

สำหรับอัลกอริทึมเอเอ็มพีเอสเอ มีความซับซ้อนเชิงเวลา (Time Complexity) คือ $O(\tilde{k}n\log n)$ โดยที่ $\tilde{k} < k \ll n$ เมื่อ $\tilde{k}$ มีค่าน้อยกว่า k และมีค่าน้อยกว่า n มาก ๆ โดยตารางที่ 12 แสดงความสัมพันธ์ระหว่างจำนวนการวนซ้ำของอัลกอริทึมโดยมี ค่า $\tilde{k}$ ค่า k และค่า n ส่งผลให้ความซับซ้อนเชิงเวลาของอัลกอริทึมเอเอ็มพี และอัลกอริทึมแสตมป์คือ $O(kn\log n)$ และ $O(n^2\log n)$ ตามลำดับ สำหรับหน่วยความจำของทั้งสามอัลกอริทึมคือ $O(n)$

เพื่อแสดงให้เห็นว่าถ้าข้อมูลอนุกรมเวลายิ่งมีขนาดใหญ่มากเท่าไร เวลาที่ใช้ในการดำเนินการสำหรับอัลกอริทึมเอเอ็มพีเอสเอ จะใช้เวลาน้อยกว่าอย่างมีนัยสำคัญ เมื่อเทียบกับอัลกอริทึมเอเอ็มพีและอัลกอริทึมแสตมป์ ดังแสดงในภาพที่ 14 15 และ 16

ตารางที่ 12 ความสัมพันธ์ของ ค่า $\tilde{k}$ ค่า k และค่า n

n	2^{15}	2^{16}	2^{17}	2^{18}	2^{19}	2^{20}
k	639	903	1277	1806	2554	3611
$\tilde{k}$	**440**	**622**	**879**	**1243**	**1758**	**2486**

ตารางที่ 13 เวลาเฉลี่ย (วินาที) ในการคำนวณสำหรับอัลกอริทึม STAMP AMP และ AMPSA บนข้อมูลสังเคราะห์ โดยกำหนดความยาวของลำดับย่อยในการค้น คิดเป็น 5% ของความยาวที่กำหนด

	2^{15}	2^{16}	2^{17}	2^{18}	2^{19}	2^{20}
AMPSA	**4**	**11**	**32**	**90**	**258**	**766**
AMP	12	33	96	273	796	2,343
STAMP	539	2,275	8,944	3,6672	144,856	605,498

ตารางที่ 14 เวลาเฉลี่ย (วินาที) ในการคำนวณสำหรับอัลกอริทึม STAMP AMP และ AMPSA บนข้อมูลจริง โดยกำหนดความยาวของลำดับย่อยในการค้น คิดเป็น 5% ของความยาวที่กำหนด

ชื่อชุดข้อมูล	AMPSA	AMP	STAMP
EEG	**109**	367	21,468
ECG	**106**	384	21,457
AEP	**128**	412	22,184
Seoul_Minute_PM2.5	**119**	371	21,329
Earthquake	**102**	368	21,288
Hessi_solar_flare	**123**	396	21,821
Weather_Szeged	**75**	284	18,031

4.1.4 ความถูกต้องของการค้นพบโมทีฟและดิสคอร์ดผลลัพธ์

เนื่องจากการวัดความถูกต้องของโมทีฟและดิสคอร์ดผลลัพธ์ จะเทียบโดยให้ผลเฉลยที่ถูกต้องมาจากอัลกอริทึมแสตมป์ เหตุผลที่เลือกใช้ผลเฉลยจากอัลกอริทึมแสตมป์ เนื่องจากอัลกอริทึมแสตมป์เป็นงานวิจัยสำหรับการค้นพบโมทีฟและดิสคอร์ด ที่มีการอ้างอิงและพัฒนาต่อยอดอย่างกว้างขวาง อีกทั้งแสตมป์เป็นอัลกอริทึมแบบแม่นตรง (Exact Algorithm) โดยจะทำการเปรียบเทียบโมทีฟและดิสคอร์ด ด้วยวิธีการหาอัตราส่วนซ้อนทับ (Overlapping Ratio, OR) และค่าเอโอดี (AoD) [33]

1) อัตราส่วนซ้อนทับ (Overlapping Ratio, OR)

อัตราส่วนซ้อนทับ คือ การวัดเฉพาะส่วนของลำดับย่อยที่เป็นโมทีฟหรือดิสคอร์ดที่หาได้จากอัลกอริทึมใด ๆ ซ้อนทับกับโมทีฟหรือดิสคอร์ดที่เป็นผลเฉลย จากนั้นทำการเทียบอัตราส่วนซ้อนทับต่อผลเฉลยแล้วคำนวณในรูปของเปอร์เซ็นต์ ดังสมการที่ (15) ซึ่งถ้าอัตราส่วนซ้อนทับที่คำนวณได้มีค่ามากแสดงว่าผลลัพธ์ที่ได้มีประสิทธิผล

$$OR(X,Y) = \frac{|X \cap Y|}{|Y|} * 100 \quad (15)$$

เมื่อ X คือผลลัพธ์ที่ได้จากอัลกอริทึม และ Y คือ ผลเฉลยของโมทีฟหรือดิสคอร์ด

2) ค่าเอโอดี

ค่าเอโอดี (AoD) หรือ Accuracy-on-Detection เป็นผลรวมของจำนวนอัตราการซ้อนทับของโมทีฟและดิสคอร์ดเทียบกับผลรวมของจำนวนโมทีฟและดิสคอร์ด ดังสมการที่ (16) เป็นการเปรียบเทียบว่าโดยเฉลี่ยแล้ว แต่ละโมทีฟและดิสคอร์ดที่หาได้จากอัลกอริทึมนั้น ตรงกับผลลัพธ์ที่ถูกต้องมากหรือน้อยเพียงใด

$$AoD = \frac{\sum_{i=1}^{m} OR(X_i, Y_i) + \sum_{j=1}^{d} OR(U_j, V_j)}{m + d} \quad (16)$$

เมื่อ m คือ จำนวนโมทีฟผลลัพธ์, d คือจำนวนดิสคอร์ดผลลัพธ์

X_i คือ โมทีฟผลลัพธ์จากอัลกอริทึมตัวที่ i

Y_i คือ ผลเฉลยของโมทีฟผลลัพธ์ตัวที่ i

U_j คือ ดิสคอร์ดผลลัพธ์จากอัลกอริทึมตัวที่ j

และ V_j คือ ผลเฉลยของดิสคอร์ดผลลัพธ์ตัวที่ j

สำหรับโมทีฟและดิสคอร์ดที่ใช้ในงานวิจัยนี้ จะใช้ค่า $m = 2$ และ $d = 1$ โดยถ้าค่าเอโอดีที่ได้มีค่ามากกว่า 95% จะถือว่าใกล้เคียงกับผลเฉลย

ตารางที่ 15 และ 16 เป็นการทดลองอัลกอริทึมเอเอ็มพีเอสเอบนข้อมูลจริงและข้อมูลสังเคราะห์ตามลำดับ โดยผลการทดลอง แสดงดังนี้

ตารางที่ 15 ผลการทดลองอัลกอริทึมเอเอ็มพีเอสเอบนข้อมูลจริง

ชื่อชุดข้อมูล	AMPSA			
	OR 1^{st} motif (%)	OR 2^{nd} motif (%)	OR discord (%)	AoD (%)
EEG	99.9995	99.9990	99.9050	99.9678
ECG	99.9585	99.9585	99.8660	99.9276
AEP	**99.9934**	**99.9934**	**99.9917**	**99.9928**
Seoul_Minute_PM2.5	99.9915	99.9920	99.2670	99.7502
Earthquake	99.9885	99.9890	99.9635	99.9803
Hessi_solar_flare	99.9982	99.9982	99.3066	99.7677
Weather_Szeged	99.9958	99.9958	99.5536	99.8485

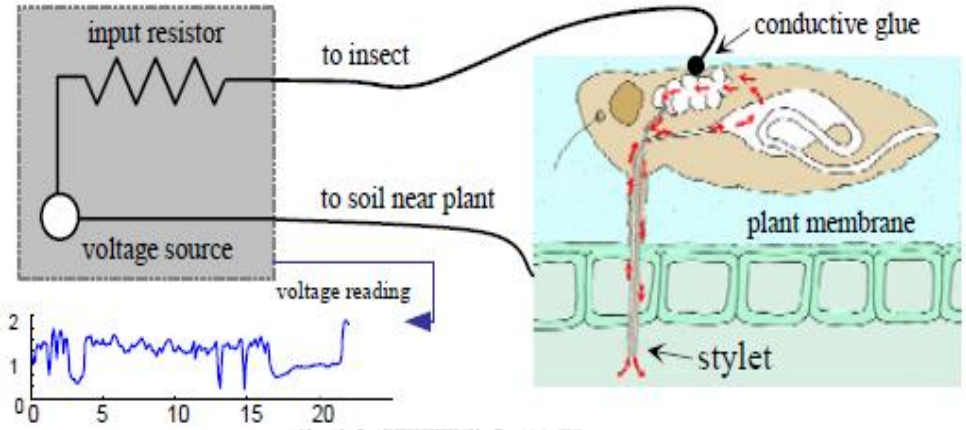
ตารางที่ 16 ผลการทดลองอัลกอริทึมเอเอ็มพีเอสเอบนข้อมูลสังเคราะห์

ชื่อชุดข้อมูล	AMPSA			
	OR 1^{st} motif (%)	OR 2^{nd}motif (%)	OR discord (%)	AoD (%)
RW1	99.9800	99.9800	99.9000	99.9533
RW2	99.8540	99.8540	99.9920	99.9000
SyntheticData1	99.9870	99.9860	99.4560	99.8096
SyntheticData2	**99.9900**	**99.9910**	**99.9820**	**99.9876**
SyntheticData3	99.2846	99.9824	99.8360	99.7009

จากตารางที่ 15 และ 16 ผลการทดลองยืนยันได้ว่า ความถูกต้องของการค้นพบโมทีฟและดิสคอร์ดของอัลกอริทึมเอเอ็มพีเอสเอ สำหรับทุกชุดข้อมูล ที่แสดงทั้งข้อมูลจริงและข้อมูลสังเคราะห์ มีค่าใกล้เคียงกับผลลัพธ์ที่ถูกต้อง วัดได้จากค่าเฉลี่ยรวมอัตราส่วนการซ้อนทับของทั้งโมทีฟและดิสคอร์ดหรือค่าเอโอดีที่มีค่ามากกว่า 95% สำหรับทุกชุดข้อมูลทั้งข้อมูลจริงและข้อมูลสังเคราะห์ ดังนั้น จึงสามารถสรุปได้ว่าโมทีฟและดิสคอร์ดผลลัพธ์ของอัลกอริทึมเอเอ็มพีเอสเอ ใกล้เคียงกับผลลัพธ์ของอัลกอริทึมแสตมป์ซึ่งเป็นอัลกอริทึมแบบแม่นตรง โดยผลลัพธ์ที่ได้นั้นเป็นไปตามวัตถุประสงค์หลักของงานวิจัยนี้ นั่นคือ ต้องการสร้างเมทริกซ์โพรไฟล์แบบประมาณเพื่อลดเวลาในการดำเนินการ แต่ยังคงได้ผลลัพธ์ใกล้เคียงกับผลลัพธ์ที่ถูกต้อง

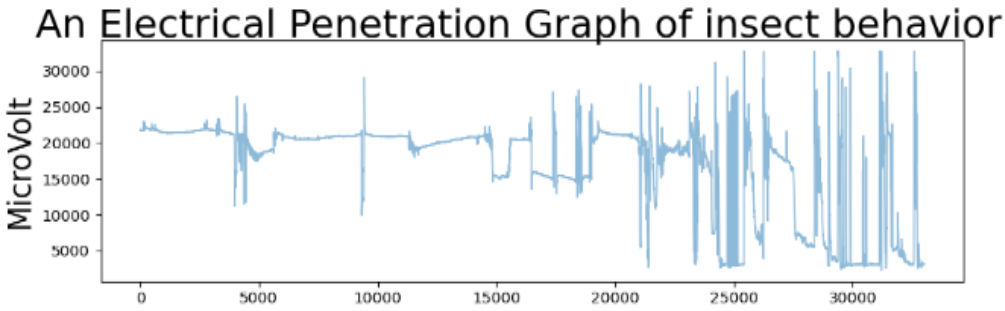
4.1.5 กรณีศึกษาข้อมูลพฤติกรรมแมลง (Case Study : Insect Behavior Data)

เพื่อที่จะแสดงประสิทธิภาพการทำงานของอัลกอริทึมเอเอ็มพีเอสเอ ซึ่งสามารถนำไปใช้ได้กับข้อมูลที่มีความหลากหลาย ผู้วิจัยจึงเสนอกรณีศึกษาซึ่งเป็นข้อมูลอนุกรมเวลาที่ศึกษาเกี่ยวกับพฤติกรรมแมลง [4] โดยต่อสายไฟขนาดเล็ก ทำการเชื่อมสายไฟกับตัวแมลงและดินใกล้กับต้นไม้ ซึ่งภายในวงจรจะมีตัววัดความต่างศักย์ไฟฟ้า แสดงดังภาพที่ 17 สำหรับกรณีศึกษาข้อมูลพฤติกรรมแมลงนี้ทำการทดลองเฉพาะการค้นพบโมทีฟ



ภาพที่ 14 วิธีดำเนินการเพื่อสร้างชุดข้อมูลอนุกรมเวลาของข้อมูลพฤติกรรมแมลง
(ที่มา: [4])

ภาพที่ 15 ข้อมูลอนุกรมเวลาที่ศึกษาเกี่ยวกับพฤติกรรมแมลง ที่มีความยาว 33,021 จุดข้อมูล โดยกำหนดค่าพารามิเตอร์สำหรับอัลกอริทึมเอเอ็มพีเอสเอ ดังนี้ ค่าแรก คือ ความยาวของลำดับย่อย 480 จุดข้อมูล และค่าความน่าจะเป็นสำหรับการค้นพบโมทีฟ คือ 0.999



ภาพที่ 15 ข้อมูลอนุกรมเวลาที่ศึกษาเกี่ยวกับพฤติกรรมแมลง มีขนาด 33,021 จุดข้อมูล
(ที่มา: [4])

ผลการทดลองของอัลกอริทึมเอเอ็มพีเอสเอเปรียบเทียบกับอัลกอริทึมเอเอ็มพีและอัลกอริทึมแสตมป์ ดังแสดงในตารางที่ 17 เพื่อเปรียบเทียบจำนวนรอบการวนซ้ำของอัลกอริทึมเอเอ็มพีเอสเอเทียบกับอัลกอริทึมเอเอ็มพี จะเห็นได้ว่าจำนวนรอบในการคำนวณลดลงจาก 666 เป็น 462 ซึ่งเวลาที่คำนวณได้จะลดลงอย่างไม่เป็นเชิงเส้น เนื่องมาจากในทุก ๆ รอบการคำนวณด้วยอัลกอริทึมเอเอ็มพีเอสเอ อัลกอริทึมจะลดปริมาณข้อมูลที่ใช้ในการคำนวณต่อรอบลดลงเป็นครึ่งหนึ่งเสมอ อีกทั้งจำนวนการวนรอบที่ใช้จะแปรตามปริมาณข้อมูลที่ลดลงเป็นครึ่งหนึ่ง ส่งผลให้เวลาที่ใช้ในการคำนวณลดลงเกือบ 5 เท่า และเมื่อเปรียบเทียบจำนวนรอบการวนซ้ำของอัลกอริทึมเอเอ็มพีเอสเอ เทียบกับอัลกอริทึมแสตมป์ จะเห็นได้ว่า จำนวนรอบในการคำนวณลดลงจาก 32,542 เป็น 666 โดยเวลาที่ใช้ในการคำนวณลดลงประมาณ 48 เท่า

ผลลัพธ์ของโมทีฟที่ได้จากอัลกอริทึมเอเอ็มพีเอสเอ อัลกอริทึมเอเอ็มพี และอัลกอริทึมแสตมป์ คือ คู่ลำดับย่อยที่ตำแหน่งเริ่มต้น (3556, 8924) (3548, 8916) และ (3553, 8921) ตามลำดับ ผลลัพธ์โมทีฟที่ได้แสดงถึงช่วงที่แมลงกำลังดูดอาหารจากพืช โดยจะเห็นว่าตำแหน่งของโมทีฟ จากแต่ละอัลกอริทึมให้ค่าใกล้เคียงกัน ซึ่งในงานนี้จะถือว่าอัลกอริทึมแสตมป์ให้ค่าของตำแหน่งโมทีฟได้ถูกต้อง ซึ่งเมื่อเทียบความถูกต้องโดยใช้ค่าอัตราส่วนซ้อนทับของโมทีฟ ทำให้ได้โมทีฟตัวแรกและตัวที่สองจากอัลกอริทึม เอเอ็มพีเอสเอ และ อัลกอริทึมเอเอ็มพี คิดเป็น 99.37 % และ 98.96% ตามลำดับ

จากผลลัพธ์ที่กล่าวมาทั้งหมดข้างต้น สามารถสรุปได้ว่าอัลกอริทึมเอเอ็มพีเอสเอให้ผลลัพธ์ที่มีประสิทธิภาพและสามารถนำไปใช้ได้จริง

ตารางที่ 17 ผลการทดลองการค้นพบโมทีฟบนข้อมูลพฤติกรรมแมลง

Algorithm	Number of iterations	Motif pair	Time	OR 1st Motif	OR 2nd Motif
AMPSA	**462**	**3,556 | 8,924**	**6 sec**	**99.37 %**	**99.37%**
AMP	666	3,548 | 8,916	27 sec	98.96%	98.96%
STAMP	32,542	3,553 | 8,921	1,300 sec	100%	100%

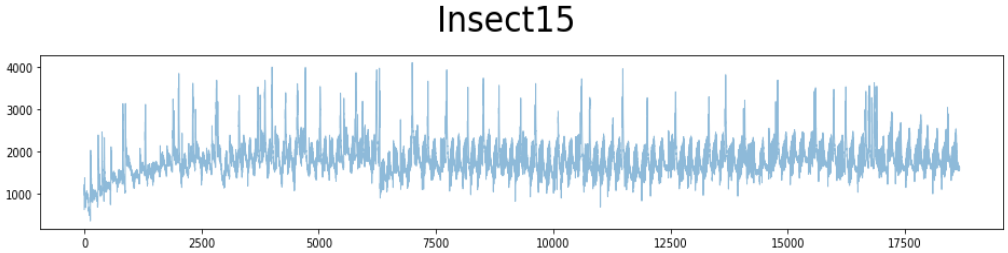
4.2 การทดลองสำหรับอัลกอริทึมพีแอลเอเอ็มพีเอสเอ

เพื่อที่จะประยุกต์การใช้งานอัลกอริทึมเอเอ็มพีเอสเอ โดยสามารถใช้งานได้กับข้อมูลอนุกรมเวลาซึ่งที่ไม่จำเป็นต้องกำหนดความยาวของลำดับย่อย เพื่อแก้ปัญหาดังกล่าวนี้ ผู้วิจัยได้นำอัลกอริทึมการค้นพบโมทีฟที่ความยาวเหมาะสม (Proper Length Motif Discovery) [12] มาประยุกต์ใช้งานกับอัลกอริทึมเอเอ็มพีเอสเอ อธิบายโดยยกกรณีศึกษา ดังต่อไปนี้

4.2.1 กรณีศึกษาข้อมูลพฤติกรรมแมลง (Case Study : Insect Behavior Data)

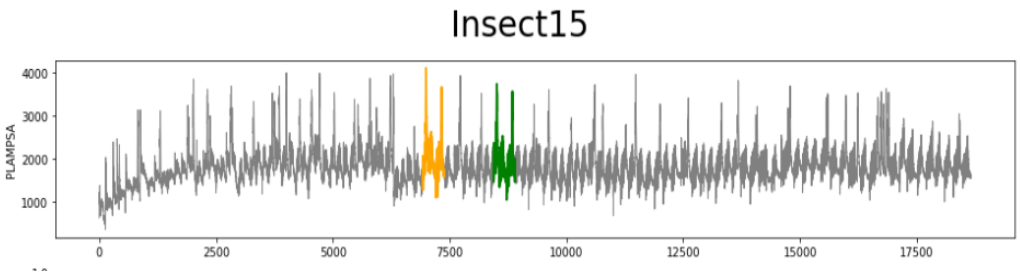
สำหรับกรณีศึกษาที่เป็นข้อมูลของพฤติกรรมแมลง [4] ซึ่งข้อมูลดังกล่าวเป็นคนละชุดข้อมูลกับหัวข้อ 4.1.5 โดยข้อมูลพฤติกรรมของแมลงชุดนี้จะมีความแตกต่างออกไปจากชุดข้อมูลอย่างชัดเจน เนื่องจากข้อมูลพฤติกรรมแมลงชุดนี้ เมื่อนำมาสกัดจะพบคู่โมทีฟที่มีความคล้ายคลึงกันมากหลายคู่ แต่มีความยาวที่เกิดขึ้นในแต่ละคู่โมทีฟระบุได้ไม่ชัดเจนว่าควรมีความยาวเท่าใด

กรณีศึกษาที่จะแสดงต่อไปนี้ เป็นการหาโมทีฟความยาวที่เหมาะสมสำหรับข้อมูลพฤติกรรมแมลง insect15 แสดงดังภาพที่ 16 เป็นข้อมูลอนุกรมเวลา ขนาด 18,667 จุดข้อมูล เป็นข้อมูลที่แสดงความต่างศักย์ไฟฟ้าที่วัดได้จากตัวแมลงกับต้นไม้ต่อเข้ากันจนครบวงจร

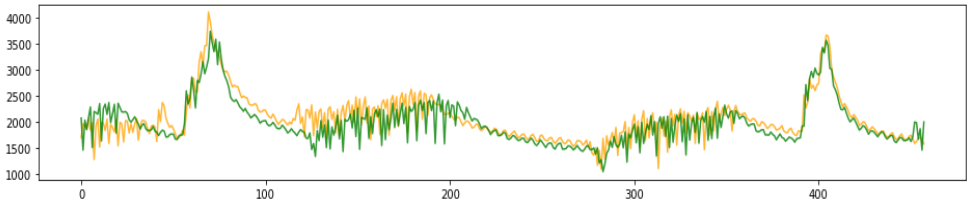


ภาพที่ 16 ข้อมูลอนุกรมเวลาที่ศึกษาเกี่ยวกับพฤติกรรมแมลง มีขนาด 18,667 จุดข้อมูล (ที่มา: [4])

ผลการทดลองที่ได้จากอัลกอริทึมพีเอลเอเอ็มพีเอสเอ ได้ผลลัพธ์โมทีฟตำแหน่งเริ่มต้นอยู่ที่ตำแหน่ง 6,998 และ ตำแหน่ง 8,496 ที่ความยาวของลำดับย่อยเป็น 458 ซึ่งใกล้เคียงกับตำแหน่งของข้อมูลพฤติกรรมของแมลง ในขณะที่แมลงกำลังกำจัดน้ำหวานที่ติดเป็นคราบเหนียวทิ้ง [4] โดยคู่โมทีฟที่แสดงพฤติกรรมดังกล่าวของแมลงแสดงในภาพที่ 17 และภาพที่ 18 ตามลำดับ โดยภาพที่ 17 แสดงโมทีฟผลลัพธ์บนข้อมูลอนุกรมเวลา และภาพที่ 18 แสดงเฉพาะส่วนโมทีฟ จากผลการทดลองสรุปได้ว่าอัลกอริทึมพีเอลเอเอ็มพีเอสเอ สามารถค้นหาโมทีฟ ได้อย่างมีประสิทธิภาพ อีกทั้งยังไม่จำเป็นต้องกำหนดค่าความยาวของลำดับย่อย



ภาพที่ 17 ผลลัพธ์โมทีฟที่ตำแหน่ง 6,998 (เหลือง) และตำแหน่ง 8,496 (เขียว) ที่ได้จากอัลกอริทึมพีแอลเอเอ็มพีเอสเอ บนข้อมูลพฤติกรรมแมลง



ภาพที่ 18 ผลลัพธ์โมทีฟมีความยาว 458 จุดข้อมูล ที่ได้จากอัลกอริทึมพีแอลเอเอ็มพีเอสเอ ที่ตำแหน่ง 6,998 (เหลือง) และตำแหน่ง 8,496 (เขียว

บทที่ 5 สรุปผลการวิจัย และข้อเสนอแนะ

5.1 สรุปผลงานวิจัย

อัลกอริทึมหลักสำหรับการค้นพบโมทีฟและดิสคอร์ดที่นิยมในปัจจุบัน คือเมทริกซ์โพรไฟล์ จะให้ผลลัพธ์การค้นพบโมทีฟและดิสคอร์ดเป็นแบบแม่นตรง ซึ่งใช้การคำนวณระยะทางของค่าแต่ละคู่ลำดับย่อยผ่านระยะยุคลิดซึ่งไม่ซับซ้อน และตรงไปตรงมา การได้มาซึ่งโมทีฟและดิสคอร์ดนั้นสามารถหาได้จากค่าสูงสุดและค่าต่ำสุดของเมทริกซ์โพรไฟล์ โดยรวมแล้วอัลกอริทึมสำหรับการหาเมทริกซ์โพรไฟล์มีความซับซ้อนเชิงเวลาเป็น $O(n^2 \log n)$ ปัญหาที่พบคือ เมื่ออนุกรมเวลามีขนาดใหญ่มาก การค้นพบโมทีฟและดิสคอร์ดจะยิ่งใช้เวลามากขึ้นซึ่งเพิ่มขึ้นแบบไม่เชิงเส้น กล่าวคือ อัลกอริทึมจะใช้เวลานานมาก เมื่ออนุกรมเวลามีขนาดใหญ่ และอีกหนึ่งปัญหา คือ การกำหนดความยาวของลำดับย่อย ซึ่งโดยปกติแล้วจะไม่ทราบค่าที่แน่ชัด จำเป็นที่จะต้องมีผู้เชี่ยวชาญหรือความรู้เฉพาะทางในด้านนั้น ๆ เพื่อกำหนดความยาวของลำดับย่อย

เพื่อที่จะแก้ปัญหาที่ได้กล่าวมาข้างต้น งานวิจัยนี้จึงได้นำเสนอเมทริกซ์โพรไฟล์แบบประมาณที่มีการเพิ่มสมรรถนะ (อัลกอริทึมเอเอ็มพีเอสเอ) โดยปรับปรุงจากเมทริกซ์โพรไฟล์และเมทริกซ์โพรไฟล์แบบประมาณ (อัลกอริทึมเอเอ็มพี) หลักการทำงานที่สำคัญของเมทริกซ์โพรไฟล์แบบประมาณที่มีการเพิ่มสมรรถนะนั้น คือ การลดมิติของข้อมูลอนุกรมเวลา โดยยังคงคุณสมบัติเดิมของอนุกรมเวลาตั้งต้นเอาไว้ให้ได้มากที่สุด ซึ่งการลดมิติของข้อมูลอนุกรมเวลา จะส่งผลอย่างมากในการลดเวลาที่ใช้ในการดำเนินการ ก่อนที่จะนำมาคำนวณโดยใช้เมทริกซ์โพรไฟล์แบบประมาณที่มีการปรับปรุงจำนวนรอบในการค้นที่เหมาะสม โดยจำนวนรอบที่เหมาะสมหาได้จากทฤษฎีวันเกิด และทำการประมาณค่าที่ได้จากทฤษฎีวันเกิดด้วยการประมาณแบบสเตอร์ลิง กล่าวคือ สำหรับการค้นพบโมทีฟและดิสคอร์ดใด ๆ ไม่จำเป็นต้องใช้ครบทุกค่าภายในเมทริกซ์โพรไฟล์ เพียงแต่ใช้ค่าสูงสุดและต่ำสุดก็เป็นการเพียงพอ

เพื่อที่จะวัดประสิทธิผลความถูกต้อง รวมถึงเวลาที่ใช้ในการค้นพบโมทีฟและดิสคอร์ดผลลัพธ์จึงเลือกใช้วิธีการวัดผลที่มีชื่อเรียกว่าเอโอดี คือ ผลรวมของจำนวนอัตราการซ้อนทับของโมทีฟและดิสคอร์ดเทียบกับจำนวนโมทีฟและดิสคอร์ด ผลที่วัดได้ในบทที่ 4 ยืนยันได้ว่า การทดลองที่ได้มีค่าเอโอดีสูง กล่าวคือ โดยเฉลี่ยแล้วโมทีฟและดิสคอร์ดที่ได้มีความใกล้เคียงกับผลเฉลยที่ถูกต้องสูงมาก ซึ่งเวลาที่ใช้ในการค้นพบโมทีฟและดิสคอร์ดลดลงเป็นอย่างมาก เมื่อเทียบกับวิธีการเมทริกซ์โพรไฟล์แบบแม่นตรงและเมทริกซ์โพรไฟล์แบบประมาณ และอีกหนึ่งปัญหาหลักสำหรับการกำหนดความยาว

ของลำดับย่อย ผู้วิจัยจึงได้นำวิธีการค้นพบโมทีฟที่ความยาวเหมาะสม (Proper Length Motif Discovery) มาประยุกต์ใช้กับเมทริกซ์โพรไฟล์แบบประมาณที่มีการเพิ่มสมรรถนะ (อัลกอริทึมพีแอลเอเอ็มพีเอสเอ) เพื่อลดเวลาในการดำเนินการ โดยแสดงในส่วนของกรณีศึกษาข้อมูลพฤติกรรมของแมลง (insect15) ซึ่งผลลัพธ์โมทีฟที่ได้แสดงให้เห็นว่า อัลกอริทึมพีแอลเอเอ็มพีเอสเอ สามารถให้ค่าที่ถูกต้องอีกทั้งสามารถนำไปใช้ได้บนข้อมูลที่ต้องการศึกษา

5.2 ข้อจำกัดและข้อเสนอแนะ

ข้อจำกัดและข้อเสนอแนะต่อไปนี้เป็นแนวทางในการวิจัย เพื่อแก้ไขข้อจำกัดบางส่วน และทำการปรับปรุงอัลกอริทึมการค้นพบโมทีฟและดิสคอร์ด โดยใช้เมทริกซ์โพรไฟล์แบบประมาณโดยเพิ่มสมรรถนะให้ดียิ่งขึ้น

ข้อจำกัดของอัลกอริทึมการค้นพบโมทีฟและดิสคอร์ด โดยใช้เมทริกซ์โพรไฟล์แบบประมาณโดยเพิ่มสมรรถนะ นั่นคือ สามารถค้นพบโมทีฟและดิสคอร์ดได้ที่ความยาวของคู่ลำดับย่อยใดๆ ที่มีความยาวเท่ากันเท่านั้นไม่สามารถค้นคู่ลำดับย่อยที่มีความยาวที่แตกต่างกันได้ ซึ่งโดยปกติแล้วชุดข้อมูลส่วนใหญ่ที่สามารถค้นพบโมทีฟและดิสคอร์ดได้นั้น ลำดับย่อยที่ค้นเจอจะเป็นคู่โมทีฟและดิสคอร์ดที่มีความยาวเท่ากัน แต่อาจมีบางชุดข้อมูลที่ความยาวของลำดับย่อยสำหรับการค้นพบโมทีฟและดิสคอร์ดอาจไม่เท่ากันก็ได้ ซึ่งการแก้ปัญหาการวัดความยาวของลำดับย่อยที่ต่างกัน สามารถใช้การวัดระยะแบบไดนามิกไทม์วอร์ปปิง (Dynamic Time Warping) แต่จะมีความยุ่งยากซับซ้อนมาก ถ้านำวิธีการวัดระยะแบบไดนามิกไทม์วอร์ปปิงมาใช้กับอัลกอริทึมเอเอ็มพีเอสเอ อีกทั้งค่า $\tilde{k}$ เป็นค่าที่บ่งบอกจำนวนการวนรอบที่เหมาะสมกับการหาคู่โมทีฟที่คล้ายกันมากที่สุดคู่เดียวเท่านั้น ถ้าต้องการมากกว่าหนึ่งคู่จะต้องมีการคำนวณ $\tilde{k}$ ตัวใหม่เพื่อหาจำนวนรอบที่เหมาะสม

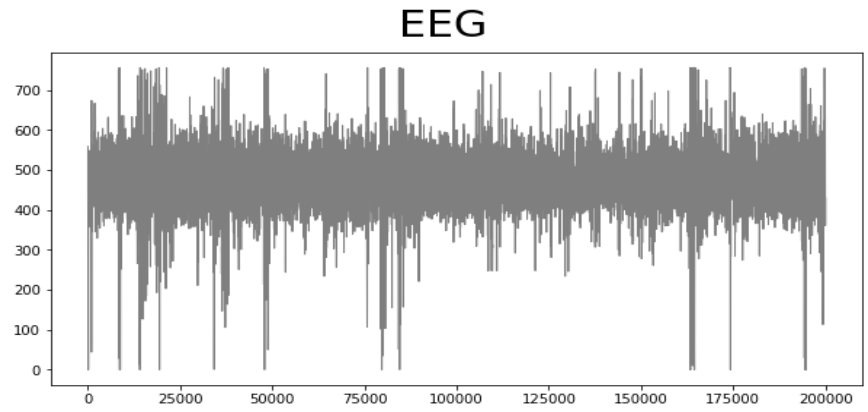
สำหรับข้อเสนอแนะ เนื่องจากเมทริกซ์โพรไฟล์สามารถนำไปประยุกต์ใช้กับวิธีการอื่น ๆ ในการทำเหมืองข้อมูลอนุกรมเวลา ยกตัวอย่างเช่น การจำแนกประเภทของอนุกรมเวลาหรือจำแนกประเภทของลำดับย่อย (Classification) การจัดกลุ่ม (Clustering) สำหรับอนุกรมเวลาหรือลำดับย่อย การหาตัวแทนกลุ่ม (Shapelet Discovery) การแบ่งส่วน (Segmentation) ฯลฯ อีกทั้งยังสามารถทำการคำนวณแบบต่อยอดเพิ่มเติม (Incremental) ในส่วนของการคำนวณเมทริกซ์โพรไฟล์ ซึ่งจะลดเวลาลงไปได้มาก แต่จะมีความซับซ้อนมากเนื่องจากลำดับในการค้นเป็นแบบสุ่ม ทำให้การจัดเรียงของการคำนวณแบบต่อยอดเพิ่มเติมต้องมีการออกแบบให้สอดคล้อง เพื่อรองรับลำดับการค้นแบบสุ่ม

ภาคผนวก

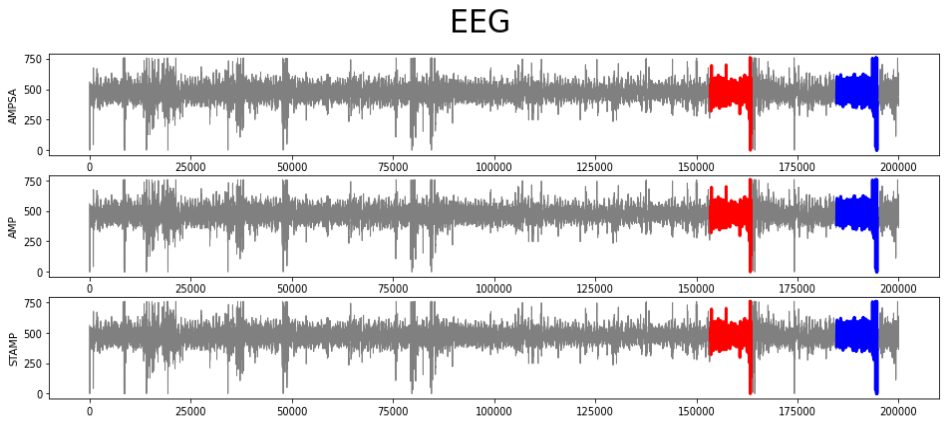
ภาคผนวก ก

ข้อมูลที่ใช้ในการทดลองที่แสดงในภาคผนวก ก มีจำนวน 12 ชุดข้อมูล โดยไม่รวมชุดข้อมูลพฤติกรรมของแมลง (Insect Behavior) ซึ่งได้ถูกแสดงไว้ในส่วนกรณีศึกษาข้อมูลแล้ว ข้อมูลที่ถูกนำมาแสดงนี้แบ่งเป็นชุดข้อมูลจริง 7 ชุด และชุดข้อมูลสังเคราะห์ 5 ชุด ซึ่งเป็นชุดข้อมูลเดียวกันกับข้อมูลที่กล่าวไว้ในบทที่ 4 ซึ่งจะถือว่าผลเฉลยของโมทีฟและดิสคอร์ดที่ได้จากอัลกอริทึมแสตมป์เป็นผลเฉลยที่ถูกต้อง เหตุผลที่เลือกใช้ผลเฉลยจากอัลกอริทึมแสตมป์ เนื่องจากอัลกอริทึมแสตมป์เป็นงานวิจัยสำหรับการค้นพบโมทีฟและดิสคอร์ดที่ได้รับการยอมรับ ซึ่งมีการอ้างอิงและพัฒนาต่อยอดในวงกว้างเป็นอันดับต้น ๆ โดยในแต่ละภาพที่ถูกแสดงจะมีคำอธิบายใต้ภาพเพื่อแสดงรายละเอียดที่เกี่ยวข้องกับชุดข้อมูล

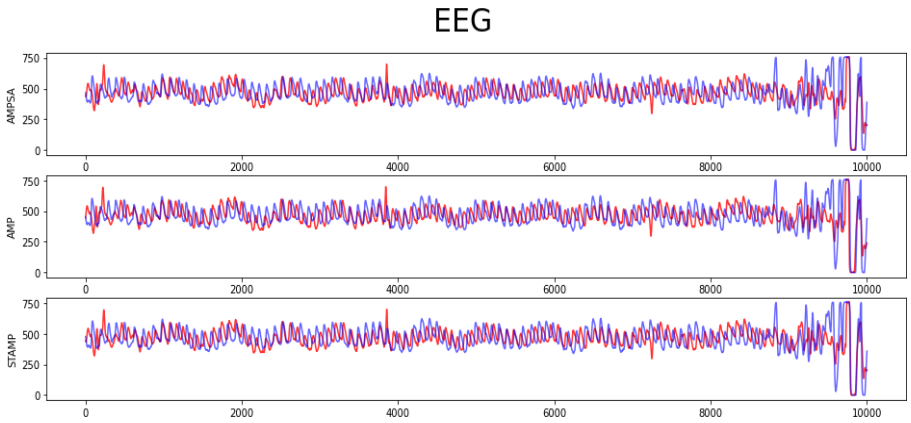
1. ชุดข้อมูลจริง EEG



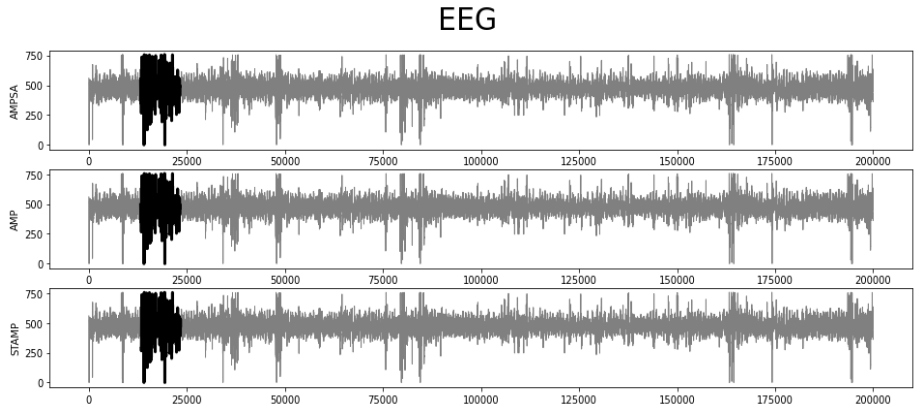
ภาพที่ 1.1 ชุดข้อมูลจริง EEG มีความยาว 200,000 จุดข้อมูล



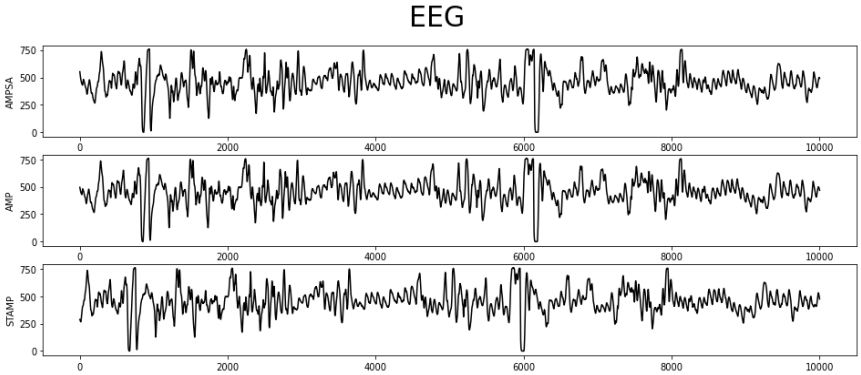
ภาพที่ 1.2 สีแดงและสีน้ำเงินแทนโมทีฟของข้อมูล EEG ของภาพที่ 1.1 มีความยาว 10,000 จุดข้อมูล ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 1.3 สีแดงและสีน้ำเงินเป็นภาพขยายโมทีฟจากภาพที่ 1.2 ของข้อมูล EEG มีความยาว 10,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

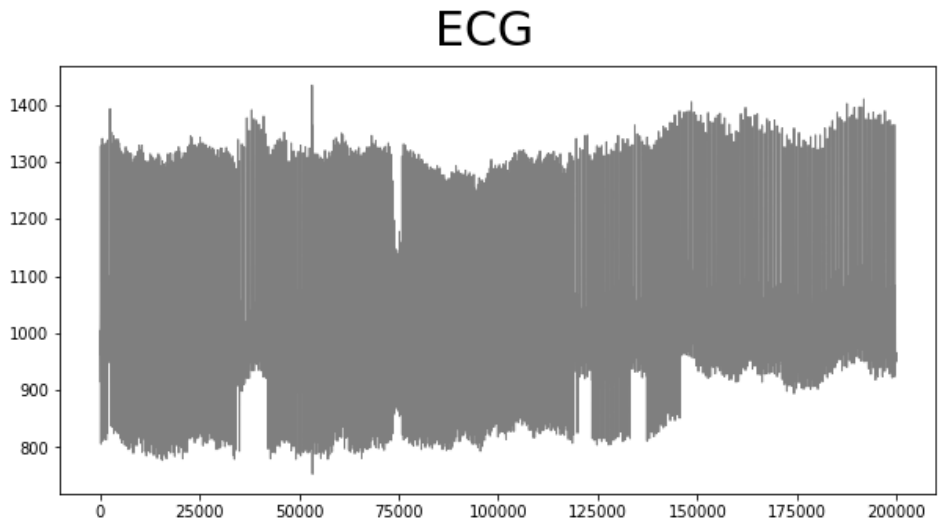


ภาพที่ 1.4 สีดำแทนดิสคอร์ดของข้อมูล EEG มีความยาว 10,000 จุดข้อมูลของภาพที่ 1.1 ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

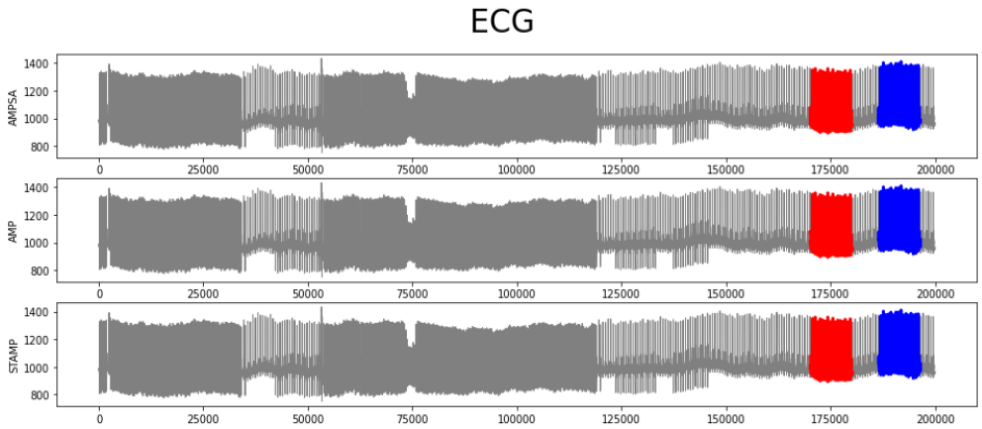


ภาพที่ 1.5 สีดำแทนดิสคอร์ดของข้อมูล EEG เป็นภาพขยายดิสคอร์ดจากภาพที่ 1.4 มีความยาว 10,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

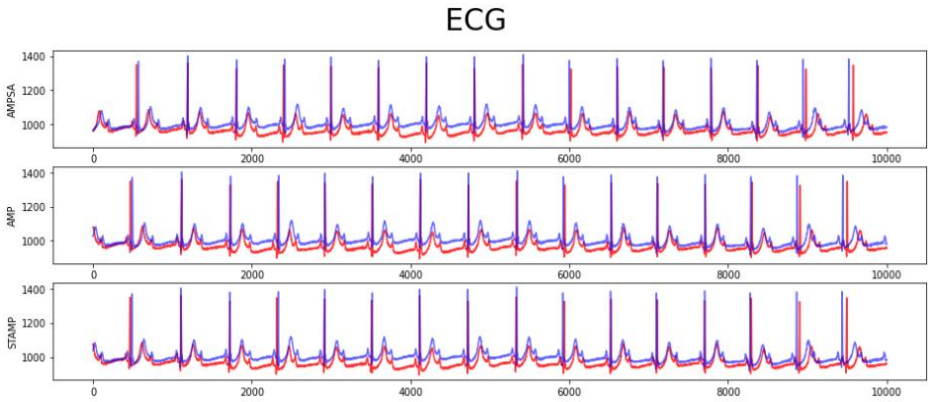
2. ชุดข้อมูลจริง ECG



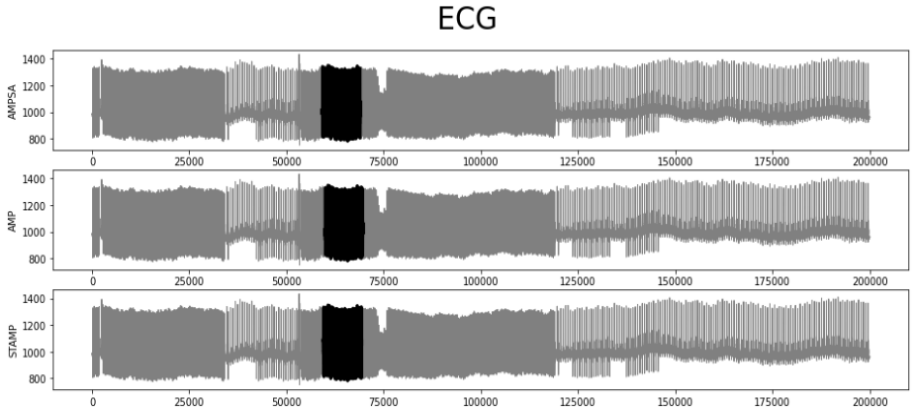
ภาพที่ 2.1 ชุดข้อมูลจริง ECG มีความยาว 200,000 จุดข้อมูล



ภาพที่ 2.2 สีแดงและสีน้ำเงินแทนโมทีฟของข้อมูล ECG ของภาพที่ 2.1 มีความยาว 10,000 จุดข้อมูล ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



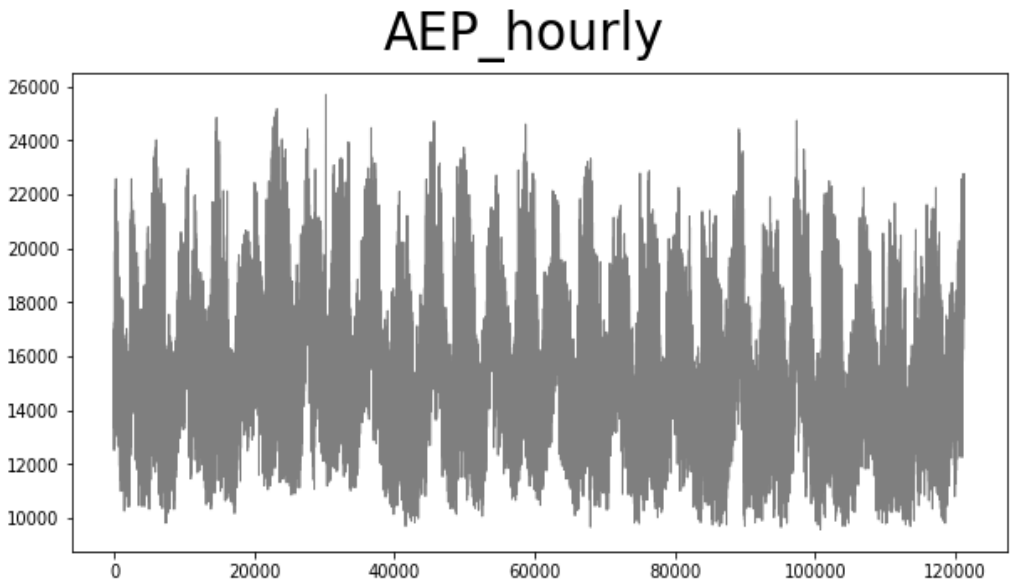
ภาพที่ 2.3 สีแดงและสีน้ำเงินเป็นภาพขยายโมทีฟจากภาพที่ 2.2 ของข้อมูล ECG มีความยาว 10,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



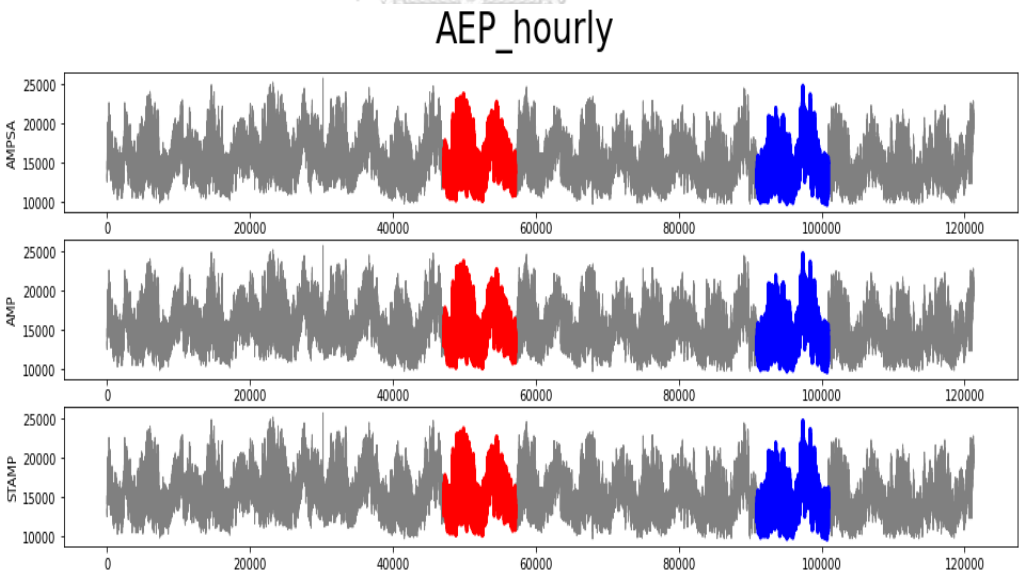
ภาพที่ 2.4 สีดำแทนดิสคอร์ดของข้อมูล ECG มีความยาว 10,000 จุดข้อมูลของภาพที่ 2.1 ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

ภาพที่ 2.5 สีดำแทนดิสคอร์ดของข้อมูล ECG เป็นภาพขยายดิสคอร์ดจากภาพที่ 2.4 มีความยาว 10,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

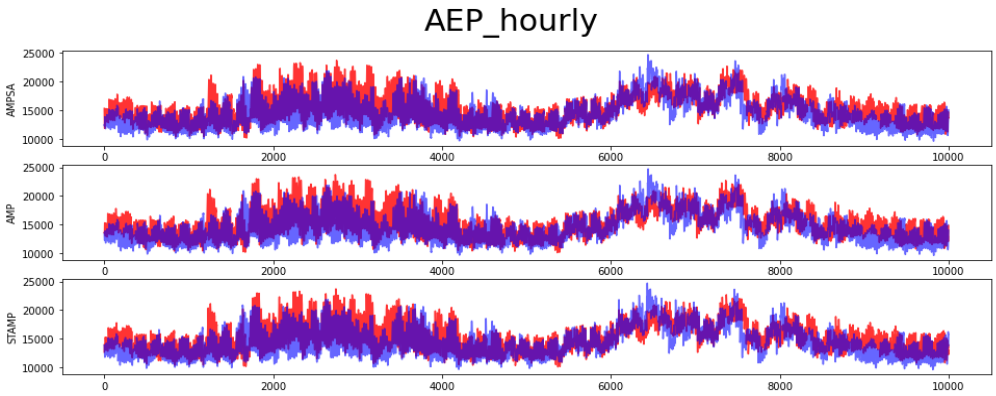
3. ชุดข้อมูลจริง AEP_hourly



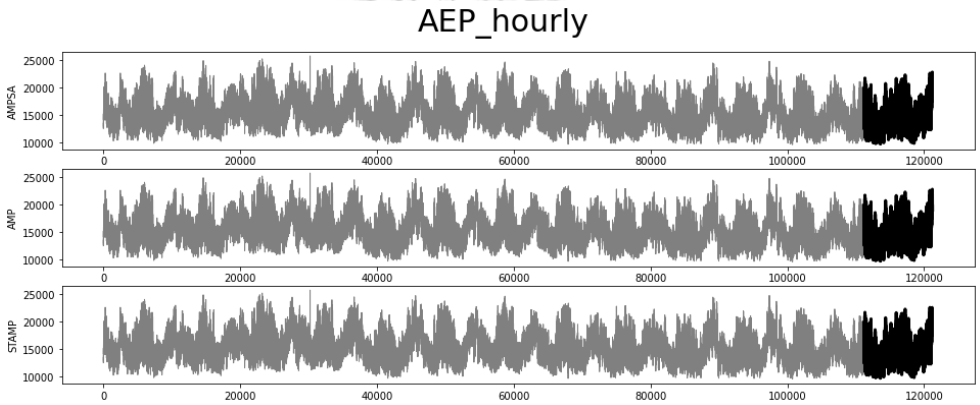
ภาพที่ 3.1 ชุดข้อมูลจริง AEP_hourly มีความยาว 121,273 จุดข้อมูล



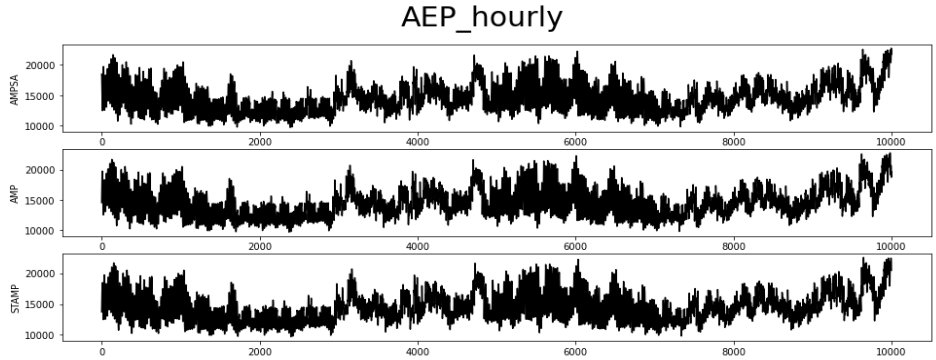
ภาพที่ 3.2 สีแดงและสีน้ำเงินแทนโมทีฟของข้อมูล AEP_hourly ของภาพที่ 3.1 มีความยาว 6,064 จุดข้อมูล ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 3.3 สีแดงและสีน้ำเงินเป็นภาพขยายโมทีฟจากภาพที่ 3.2 ของข้อมูล AEP_hourly มีความยาว 6,064 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

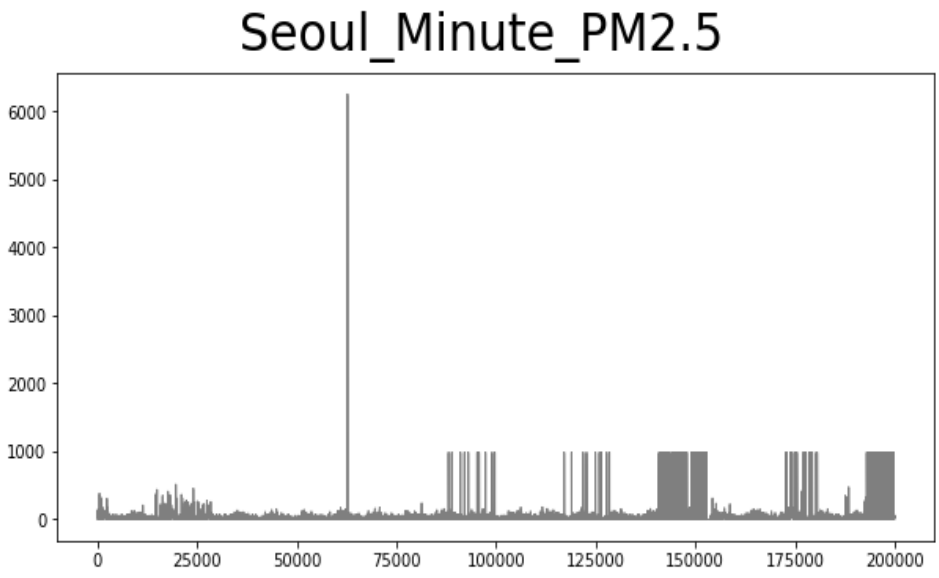


ภาพที่ 3.4 สีดำแทนดิสคอร์ดของข้อมูล AEP_hourly มีความยาว 6,064 จุดข้อมูลของภาพที่ 3.1 ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

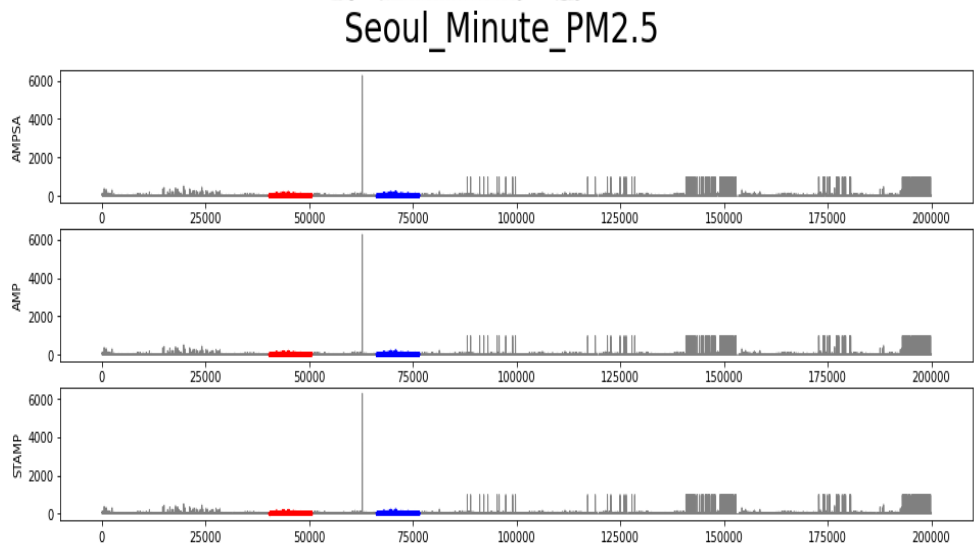


ภาพที่ 3.5 สีดำแทนดิสคอร์ดของข้อมูล AEP_hourly เป็นภาพขยายดิสคอร์ดจากภาพที่ 3.4 มีความยาว 6,064 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

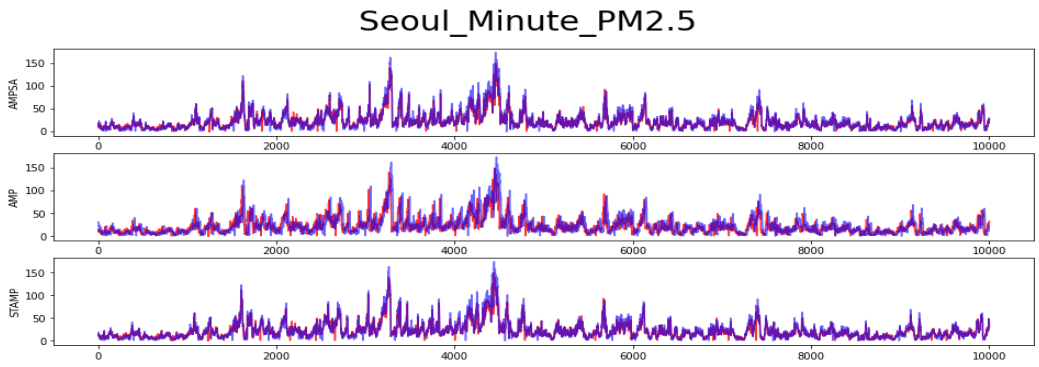
4. ชุดข้อมูลจริง Seoul_Minute_PM2.5



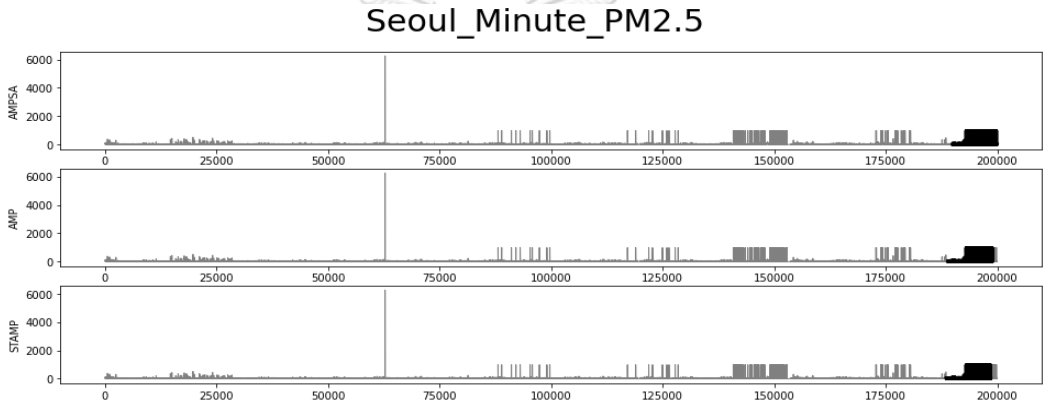
ภาพที่ 4.1 ชุดข้อมูลจริง Seoul_Minute_PM2.5 มีความยาว 200,000 จุดข้อมูล



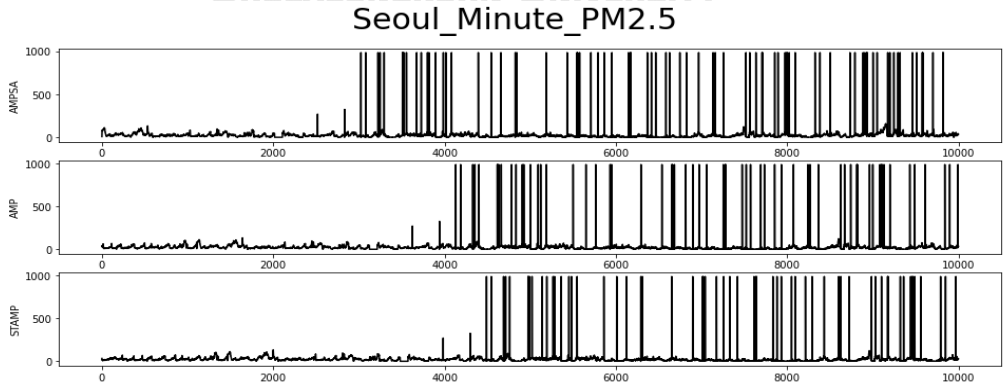
ภาพที่ 4.2 สีแดงและสีน้ำเงินแทนโมทีฟของข้อมูล Seoul_Minute_PM2.5 ของภาพที่ 4.1 มีความยาว 10,000 จุดข้อมูล ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 4.3 สีแดงและสีน้ำเงินเป็นภาพขยายโมทีฟจากภาพที่ 4.2 ของข้อมูล Seoul_Minute_PM2.5 มีความยาว 10,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

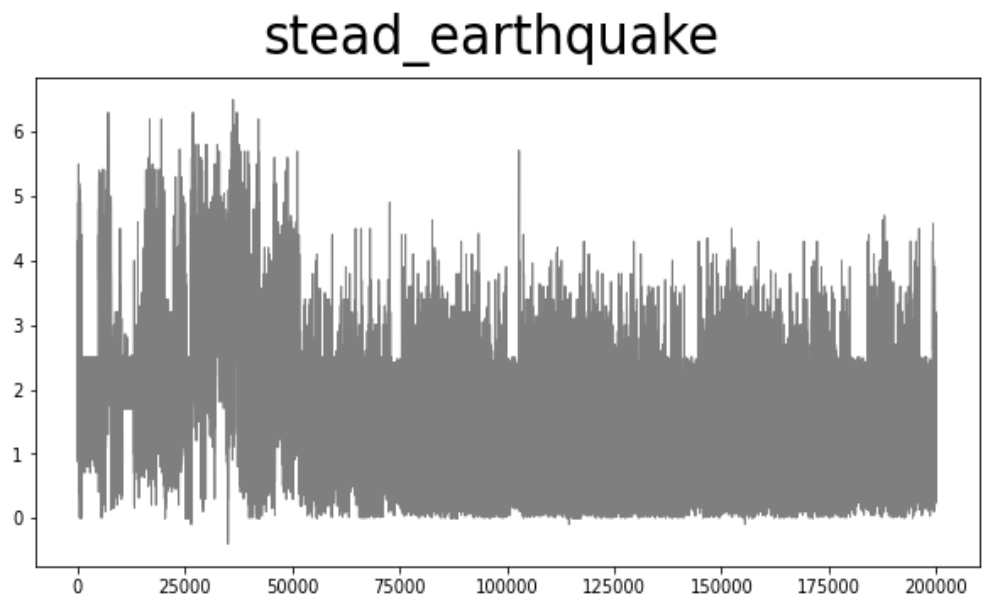


ภาพที่ 4.4 สีดำแทนดิสคอร์ดของข้อมูล Seoul_Minute_PM2.5 มีความยาว 10,000 จุดข้อมูลของภาพที่ 4.1 ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

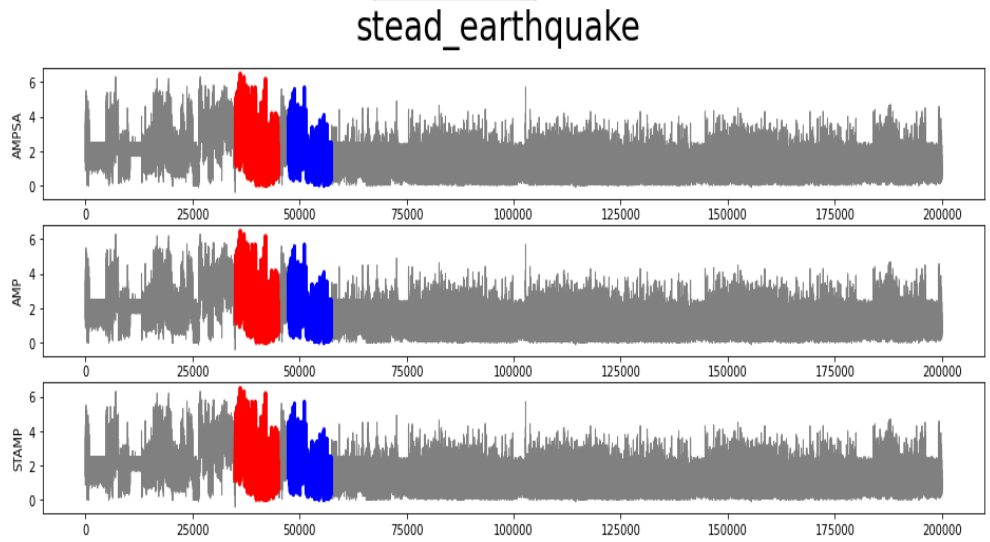


ภาพที่ 4.5 สีดำแทนดิสคอร์ดของข้อมูล Seoul_Minute_PM2.5 เป็นภาพขยายดิสคอร์ดจากภาพที่ 4.4 มีความยาว 10,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

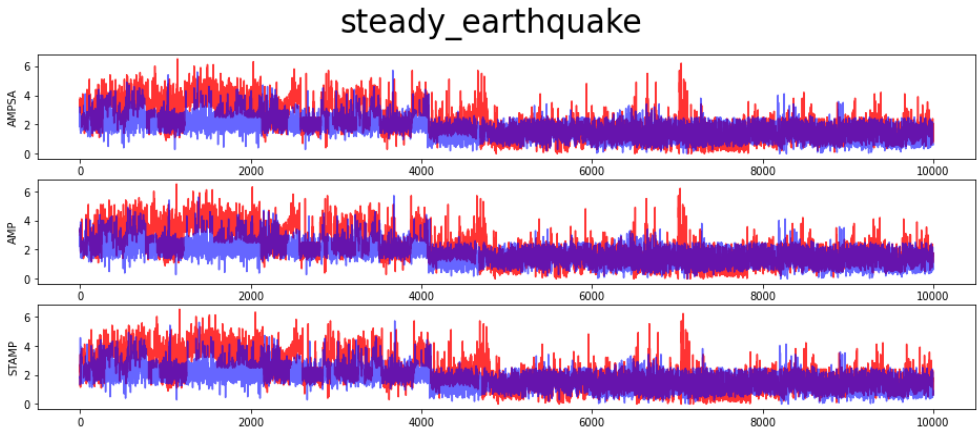
5. ชุดข้อมูลจริง Earthquake



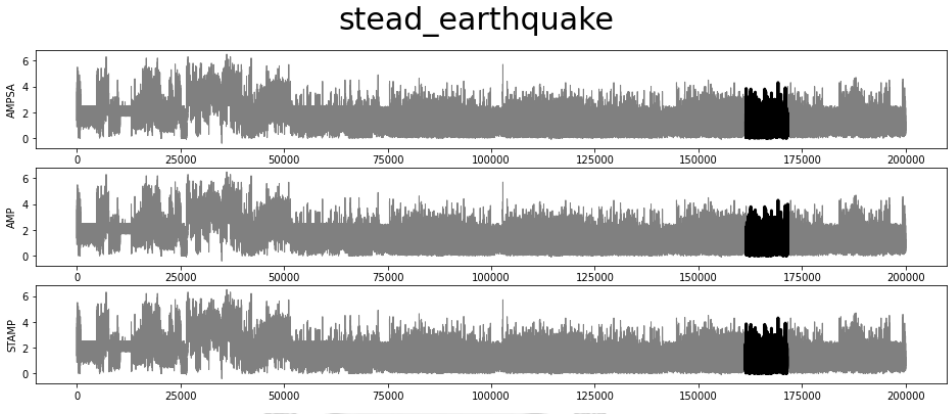
ภาพที่ 5.1 ชุดข้อมูลจริง Earthquake มีความยาว 200,000 จุดข้อมูล



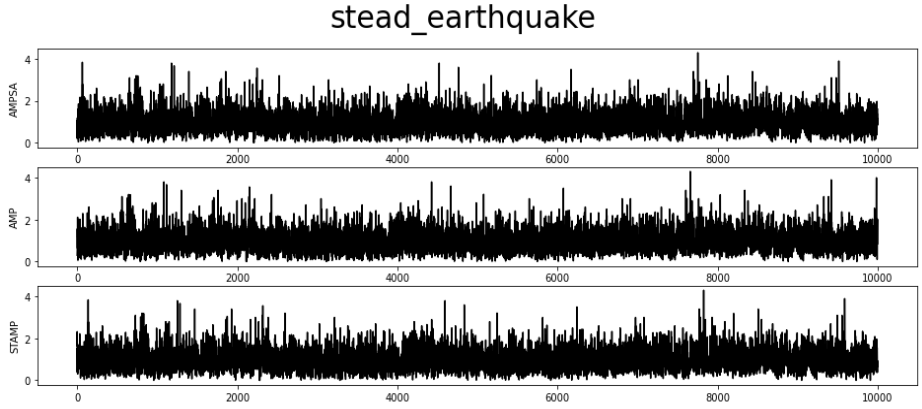
ภาพที่ 5.2 สีแดงและสีน้ำเงินแทนโมทีฟของข้อมูล Earthquake ของภาพที่ 5.1 มีความยาว 10,000 จุดข้อมูล ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 5.3 สีแดงและสีน้ำเงินเป็นภาพขยายโมทีฟจากภาพที่ 5.2 ของข้อมูล Earthquake มีความยาว 10,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

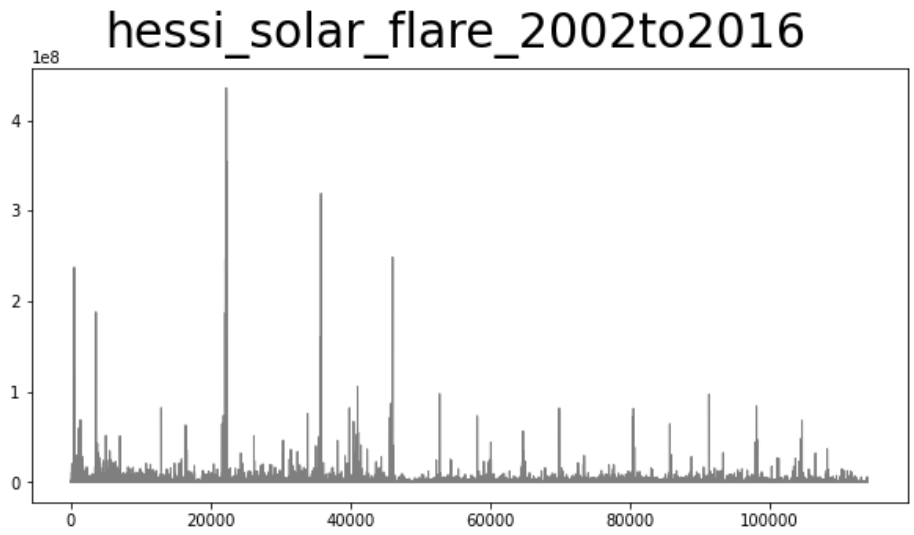


ภาพที่ 5.4 สีดำแทนดิสคอร์ดของข้อมูล Earthquake มีความยาว 10,000 จุดข้อมูลของภาพที่ 5.1 ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

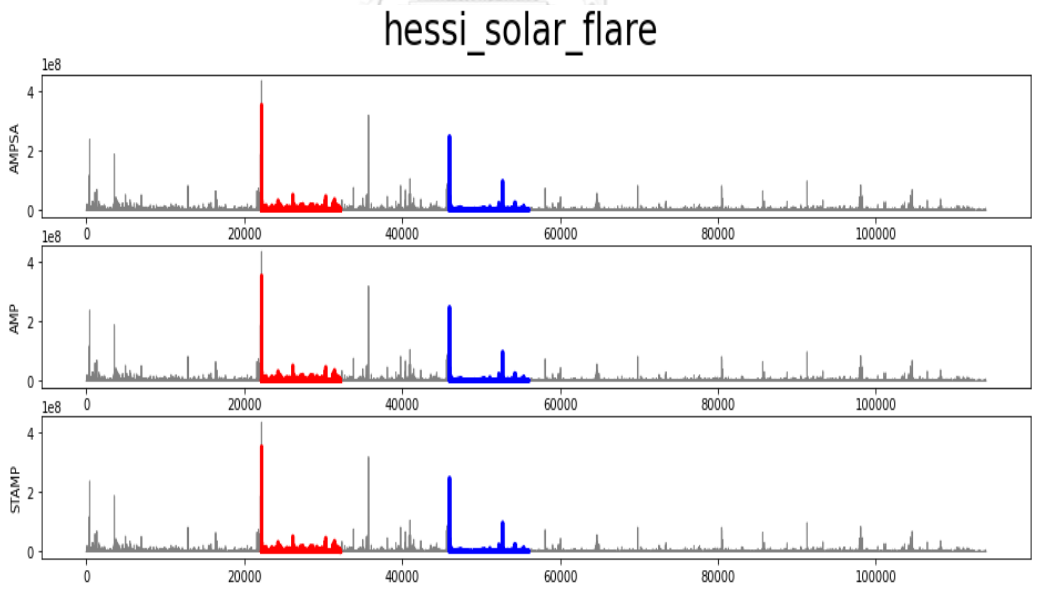


ภาพที่ 5.5 สีดำแทนดิสคอร์ดของข้อมูล Earthquake เป็นภาพขยายดิสคอร์ดจากภาพที่ 5.4 มีความยาว 10,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

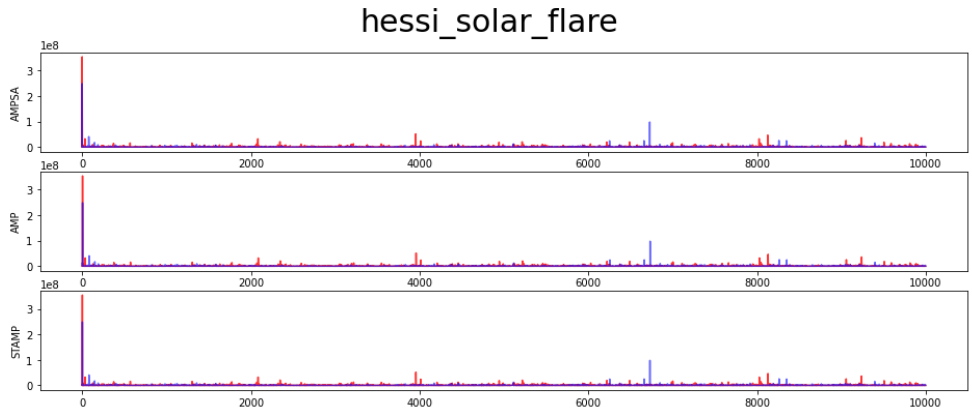
6. ชุดข้อมูลจริง Hessi_solar_flare



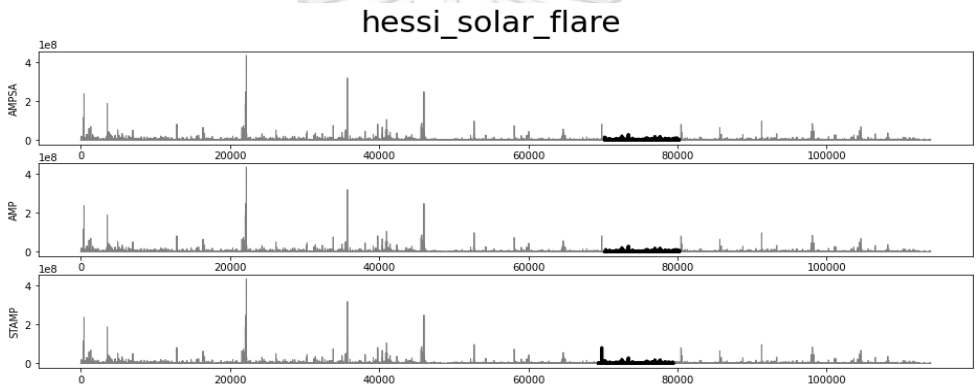
ภาพที่ 6.1 ชุดข้อมูลจริง Hessi_solar_flare มีความยาว 113,942 จุดข้อมูล



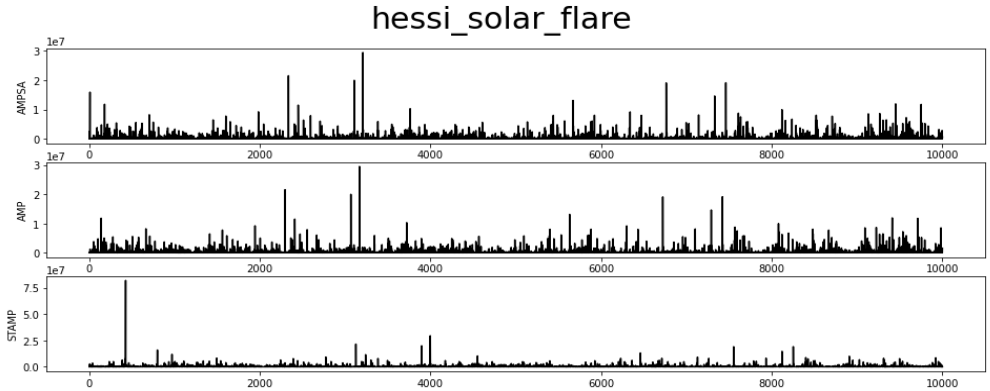
ภาพที่ 6.2 สีแดงและสีน้ำเงินแทนโมทีฟของข้อมูล Hessi_solar_flare ของภาพที่ 6.1 มีความยาว 5,698 จุดข้อมูล ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 6.3 สีแดงและสีน้ำเงินเป็นภาพขยายโมทีฟจากภาพที่ 6.2 ของข้อมูล Hessi_solar_flare มีความยาว 5,698 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

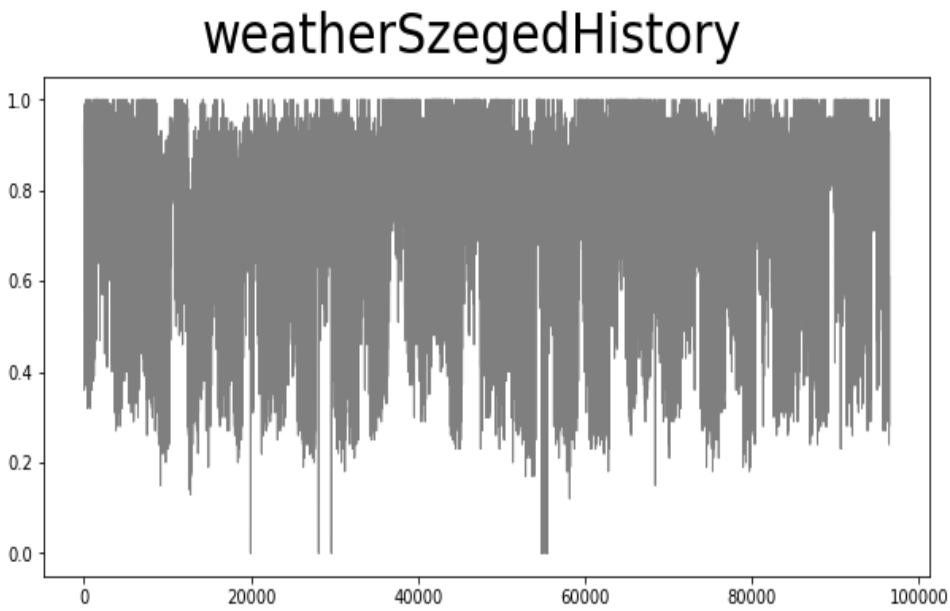


ภาพที่ 6.4 สีดำแทนดิสคอร์ดของข้อมูล Hessi_solar_flare มีความยาว 5,698 จุดข้อมูลของภาพที่ 6.1 ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

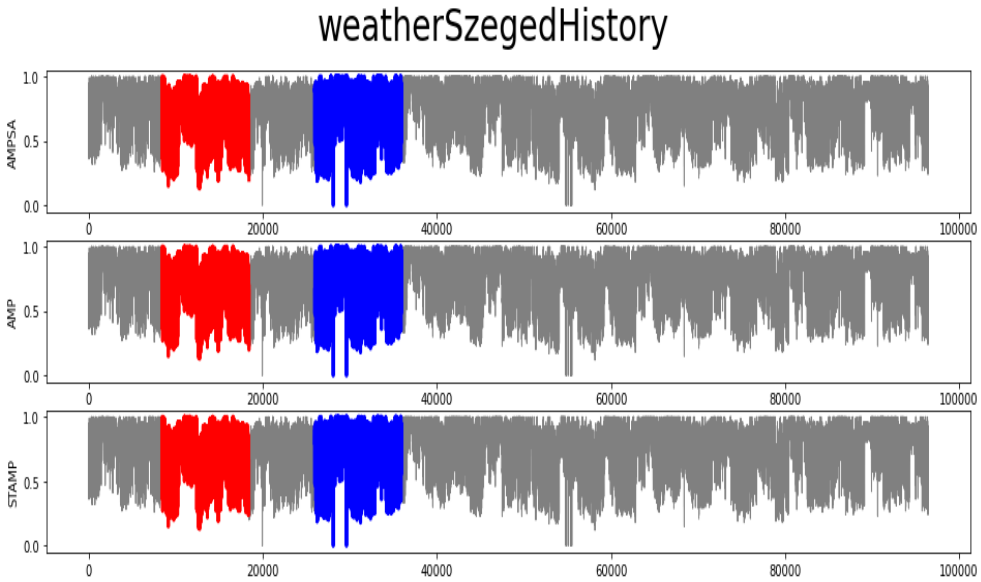


ภาพที่ 6.5 สีดำแทนดิสคอร์ดของข้อมูล Hessi_solar_flare เป็นภาพขยายดิสคอร์ดจากภาพที่ 6.4 มีความยาว 5,698 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

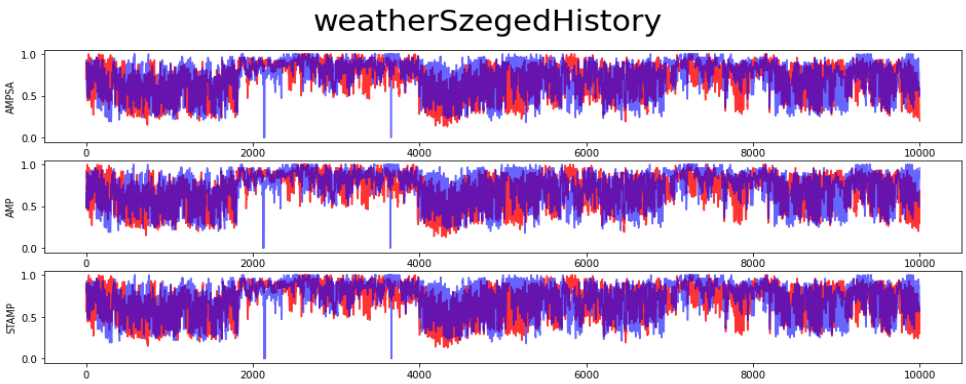
7. ชุดข้อมูลจริง Weather_Szeged



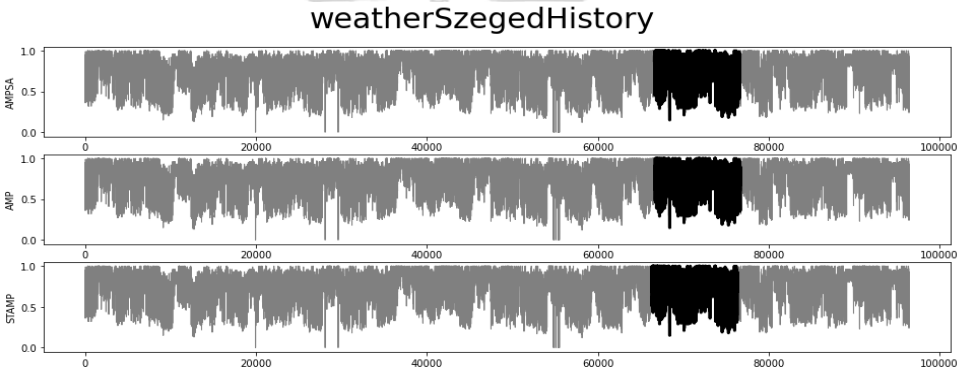
ภาพที่ 7.1 ชุดข้อมูลจริง Weather_Szeged มีความยาว 96,452 จุดข้อมูล



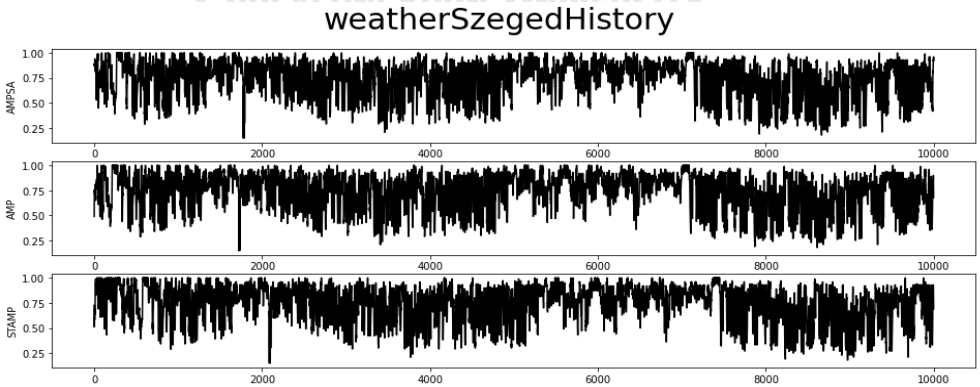
ภาพที่ 7.2 สีแดงและสีน้ำเงินแทนโมทีฟของข้อมูล Weather_Szeged ของภาพที่ 7.1 มีความยาว 4,823 จุดข้อมูล ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 7.3 สีแดงและสีน้ำเงินเป็นภาพขยายโมทีฟจากภาพที่ 7.2 ของข้อมูล Weather_Szeged มีความยาว 4,823 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

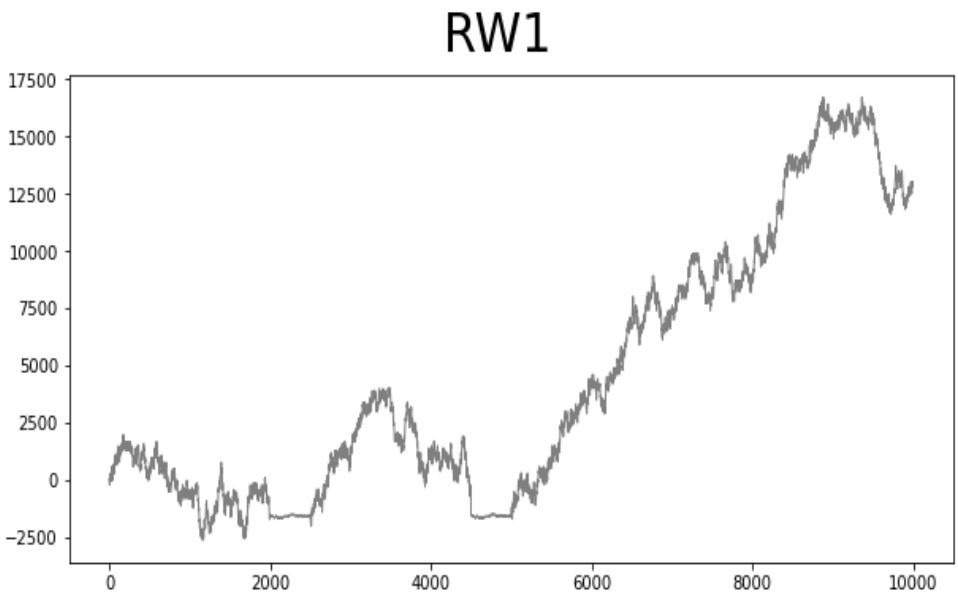


ภาพที่ 7.4 สีดำแทนดิสคอร์ดของข้อมูล Weather_Szeged มีความยาว 4,823 จุดข้อมูล ของภาพที่ 7.1 ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

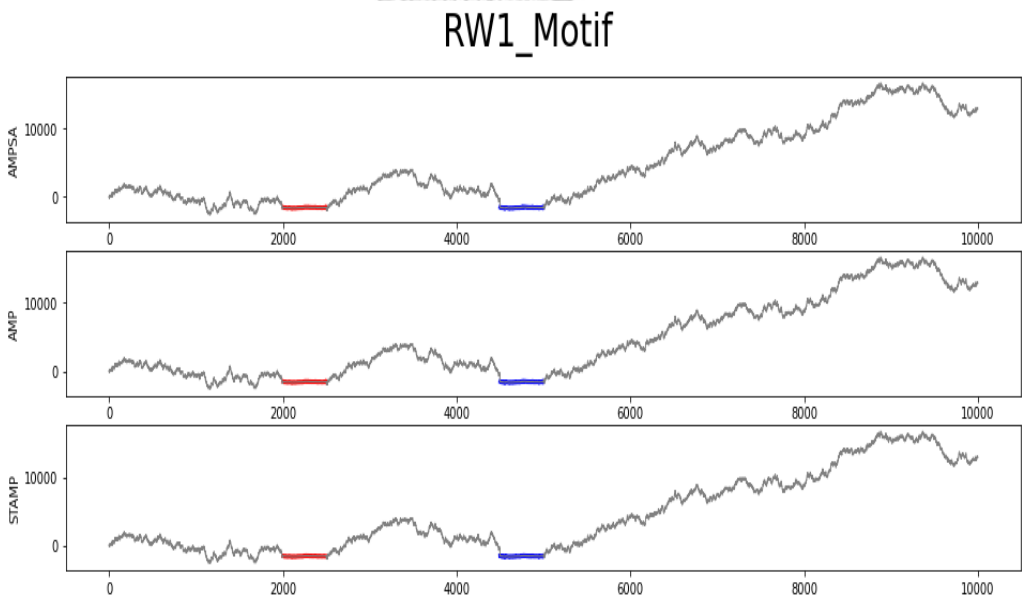


ภาพที่ 7.5 สีดำแทนดิสคอร์ดของข้อมูล Weather_Szeged เป็นภาพขยายดิสคอร์ดจากภาพที่ 7.4 มีความยาว 4,823 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

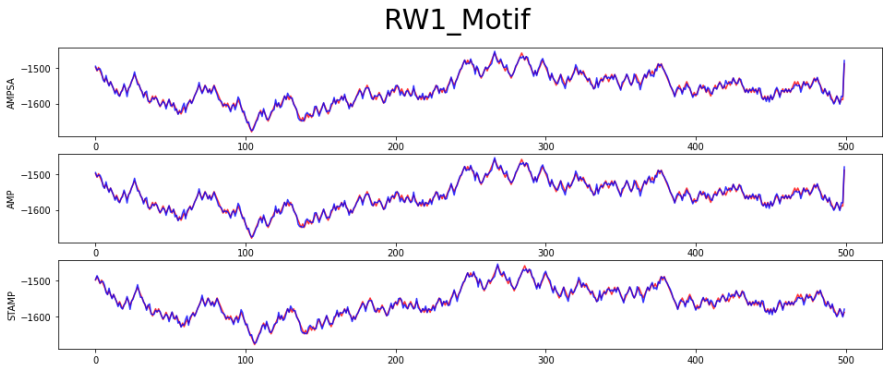
8. ชุดข้อมูลสังเคราะห์ RW1



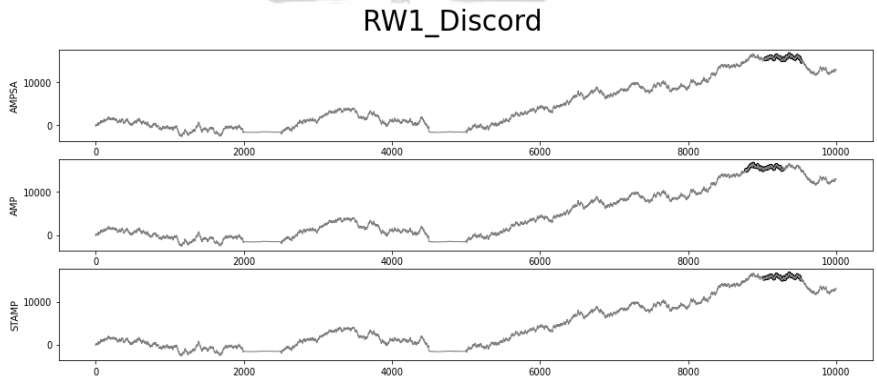
ภาพที่ 8.1 ชุดข้อมูลสังเคราะห์ RW1 มีความยาว 10,000 จุดข้อมูล



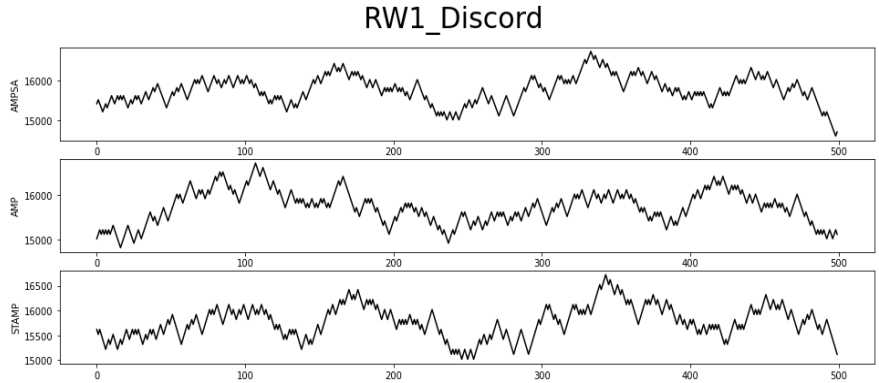
ภาพที่ 8.2 สีแดงและสีน้ำเงินแทนโมทีฟของข้อมูล RW1 ของภาพที่ 8.1 มีความยาว 500 จุดข้อมูล ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 8.3 สีแดงและสีน้ำเงินเป็นภาพขยายโมทีฟจากภาพที่ 8.2 ของข้อมูล RW1 มีความยาว 500 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

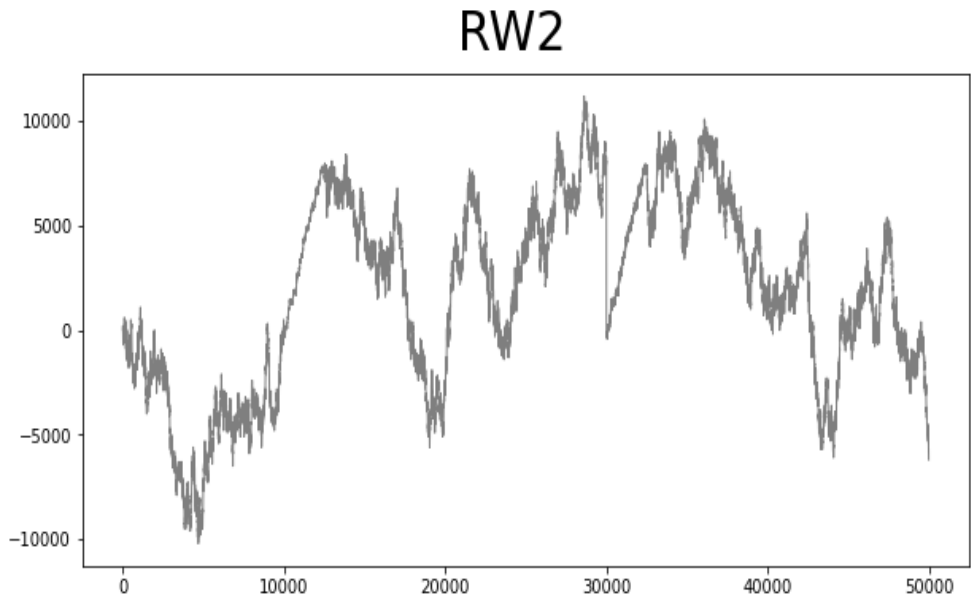


ภาพที่ 8.4 สีดำแทนดิสคอร์ดของข้อมูล RW1 มีความยาว 500 จุดข้อมูลของภาพที่ 8.1 ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

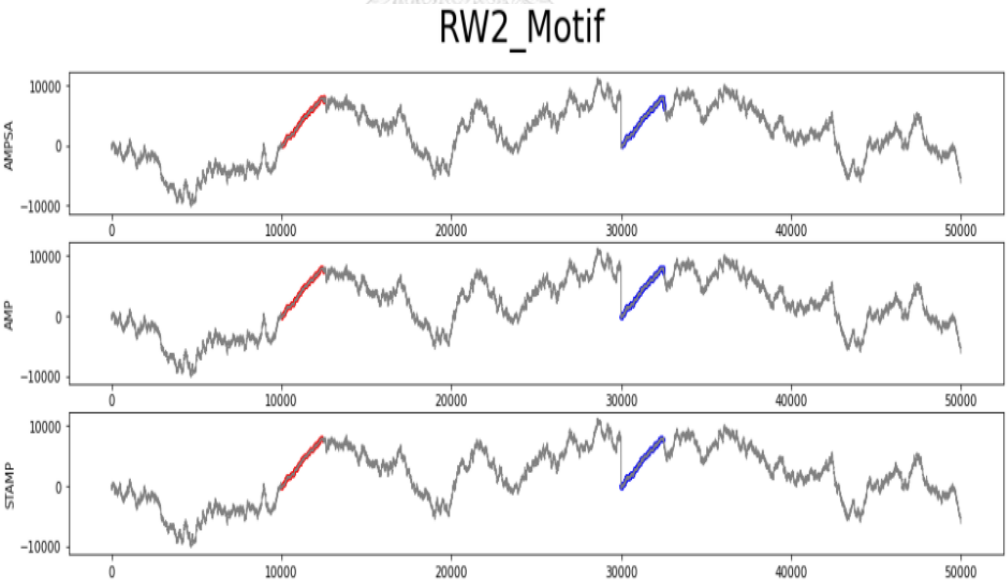


ภาพที่ 8.5 สีดำแทนดิสคอร์ดของข้อมูล RW1 เป็นภาพขยายดิสคอร์ดจากภาพที่ 8.4 มีความยาว 500 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

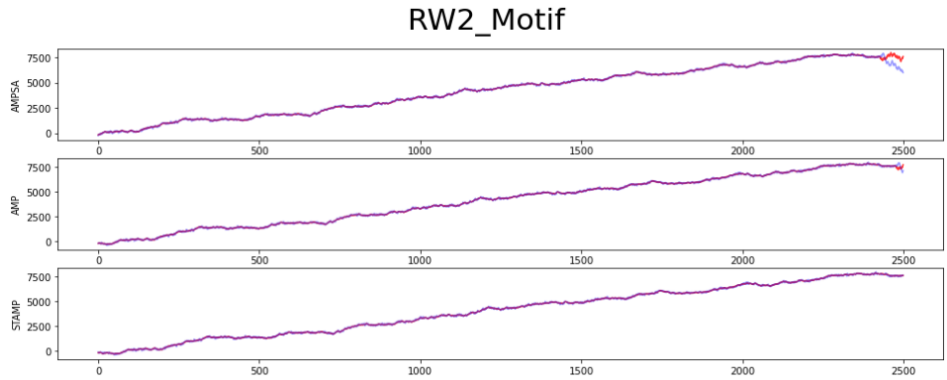
9. ชุดข้อมูลสังเคราะห์ RW2



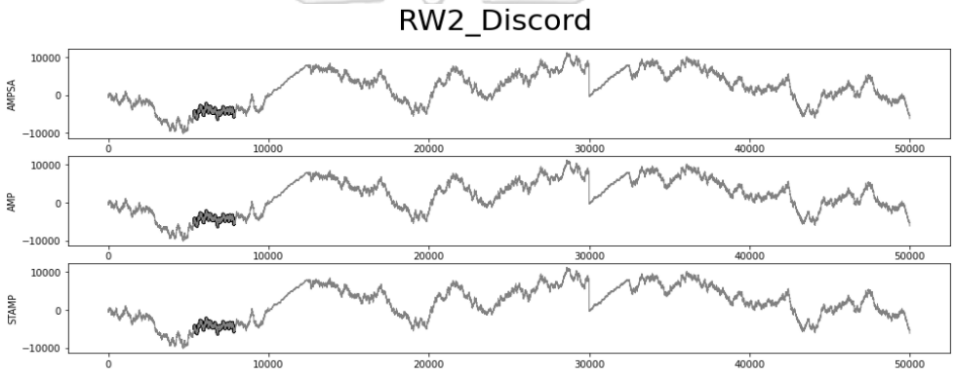
ภาพที่ 9.1 ชุดข้อมูลสังเคราะห์ RW2 มีความยาว 50,000 จุดข้อมูล



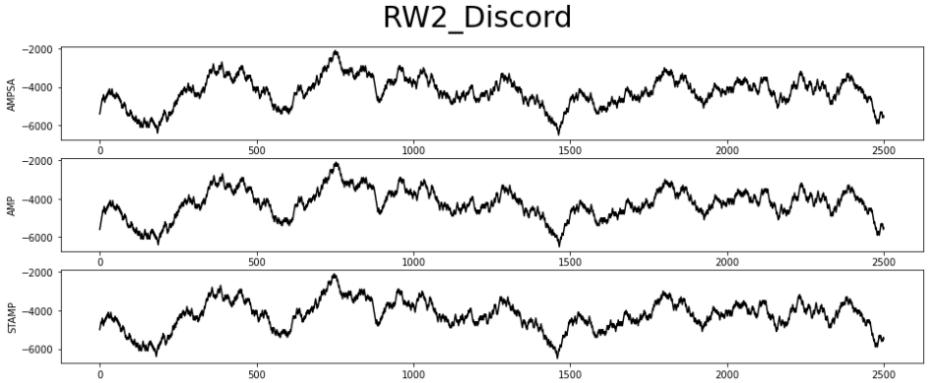
ภาพที่ 9.2 สีแดงและสีน้ำเงินแทนโมทีฟของข้อมูล RW2 ของภาพที่ 9.1 มีความยาว 2,500 จุดข้อมูล ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 9.3 สีแดงและสีน้ำเงินเป็นภาพขยายโมทีฟจากภาพที่ 9.2 ของข้อมูล RW2 มีความยาว 2,500 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

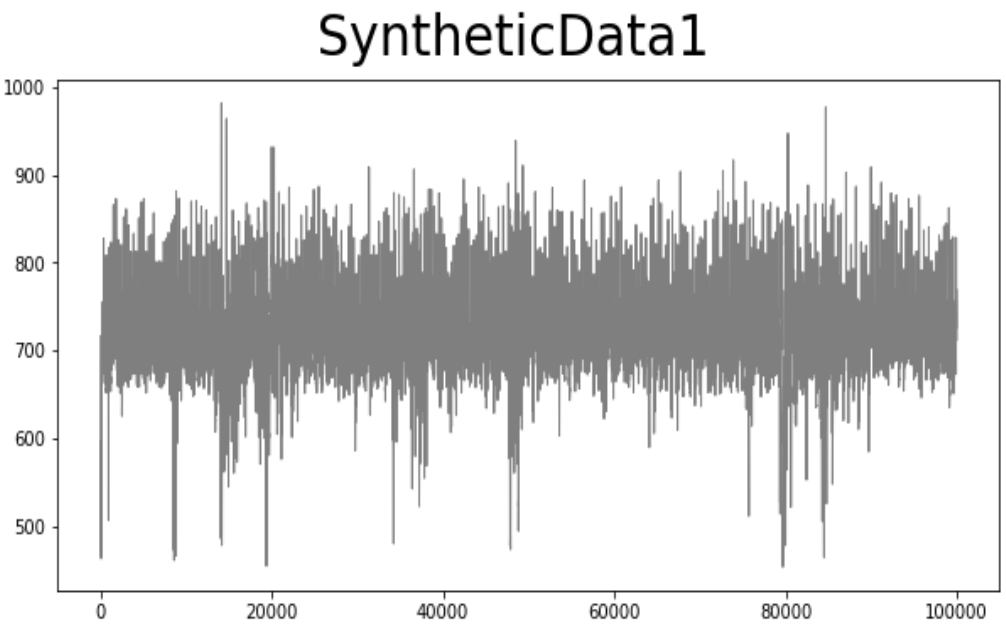


ภาพที่ 9.4 สีดำแทนดิสคอร์ดของข้อมูล RW1 มีความยาว 2,500 จุดข้อมูลของภาพที่ 9.1 ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

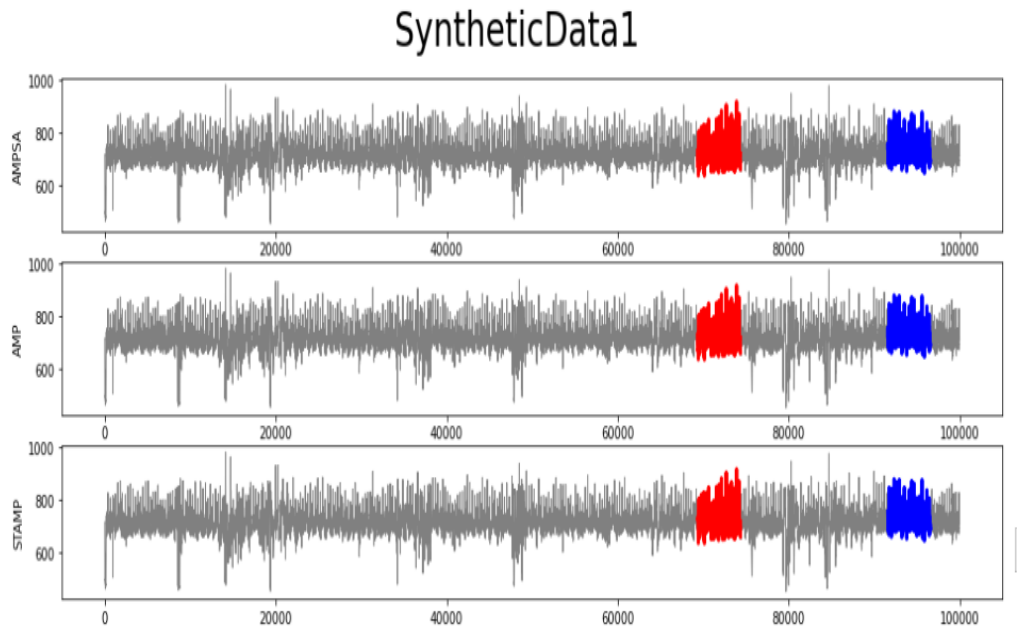


ภาพที่ 9.5 สีดำแทนดิสคอร์ดของข้อมูล RW2 เป็นภาพขยายดิสคอร์ดจากภาพที่ 9.4 มีความยาว 2,500 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

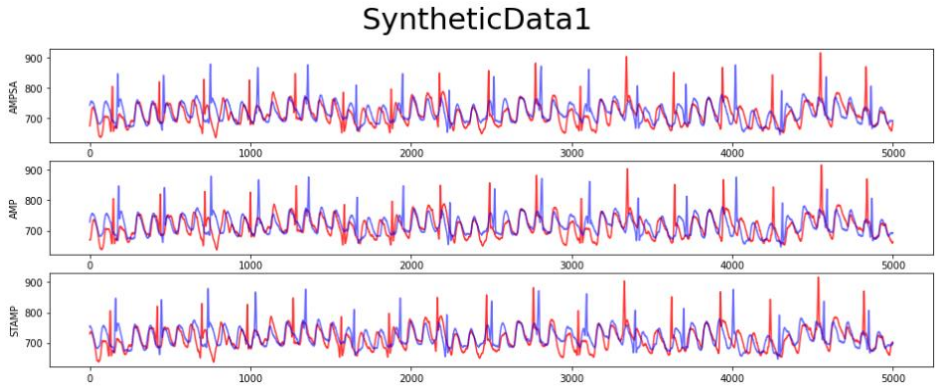
10. ชุดข้อมูลสังเคราะห์ SyntheticData1



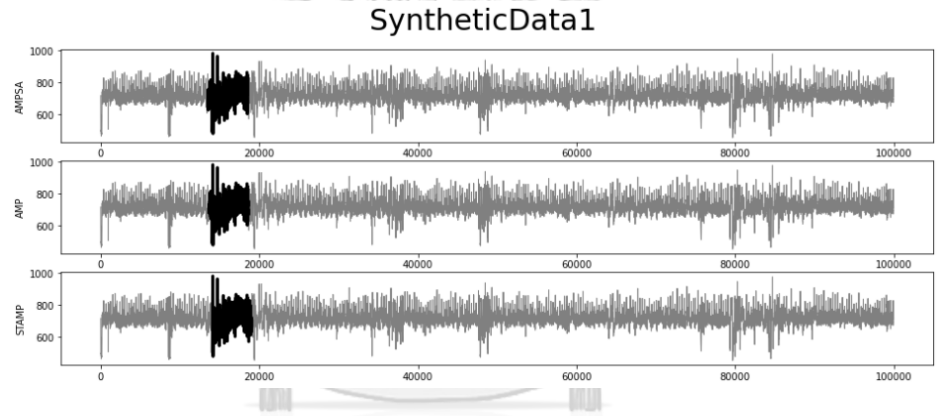
ภาพที่ 10.1 ชุดข้อมูลสังเคราะห์ SyntheticData1 มีความยาว 100,000 จุดข้อมูล



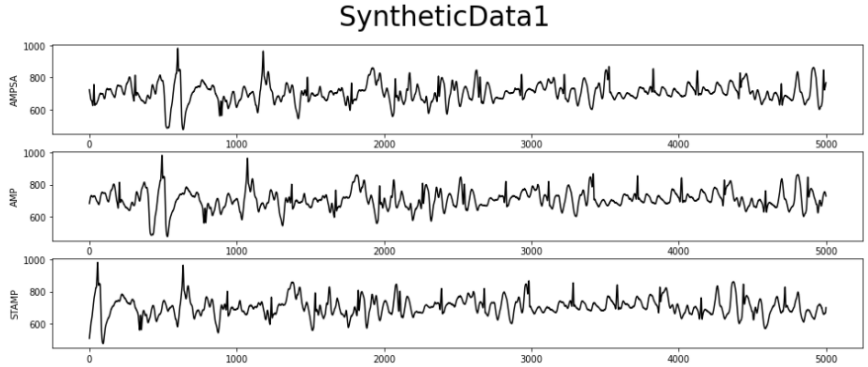
ภาพที่ 10.2 สีแดงและสีน้ำเงินแทนโมทีฟของข้อมูล SyntheticData1 ของภาพที่ 10.1 มีความยาว 5,000 จุดข้อมูล ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 10.3 สีแดงและสีน้ำเงินเป็นภาพขยายโมทีฟจากภาพที่ 10.2 ของข้อมูล SyntheticData1 มีความยาว 5,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

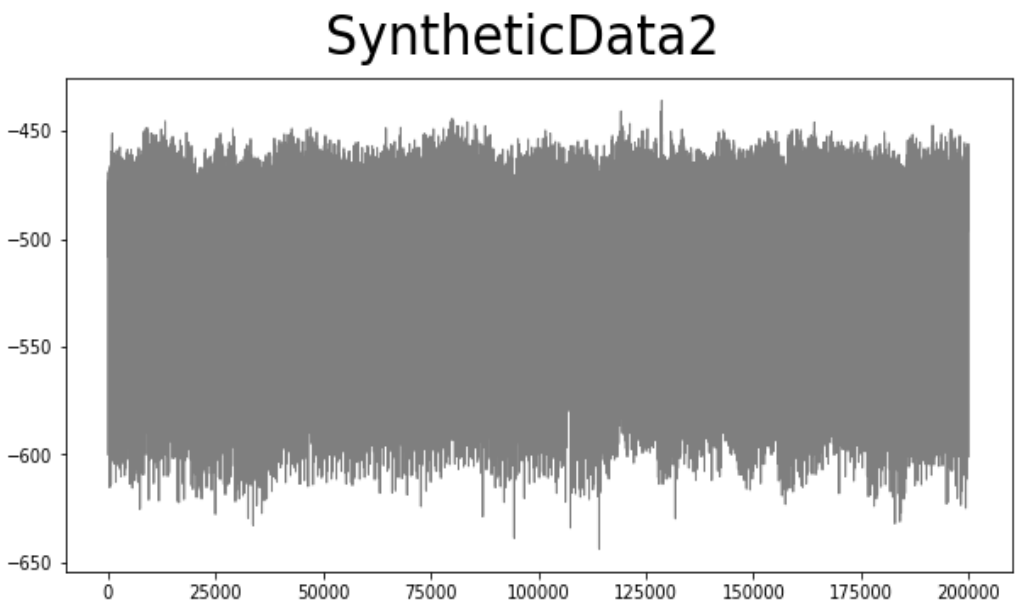


ภาพที่ 10.4 สีดำแทนดิสคอร์ดของข้อมูล SyntheticData1 มีความยาว 5,000 จุดข้อมูลของภาพที่ 10.1 ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

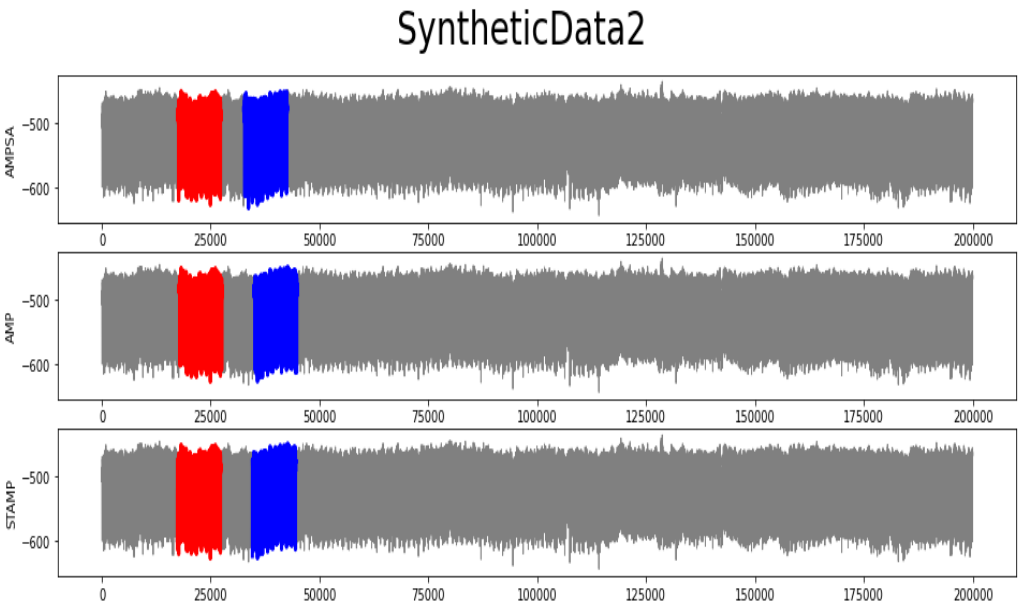


ภาพที่ 10.5 สีดำแทนดิสคอร์ดของข้อมูล SyntheticData1 เป็นภาพขยายดิสคอร์ดจากภาพที่ 10.4 มีความยาว 5,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

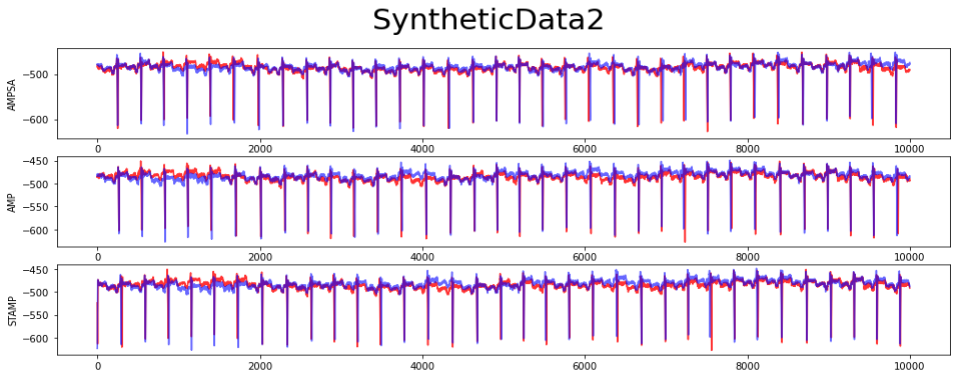
11. ชุดข้อมูลสังเคราะห์ SyntheticData2



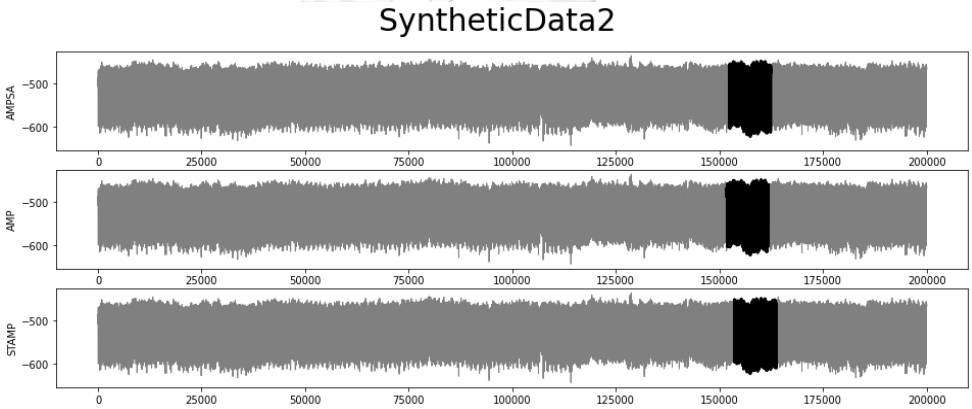
ภาพที่ 11.1 ชุดข้อมูลสังเคราะห์ SyntheticData2 มีความยาว 200,000 จุดข้อมูล



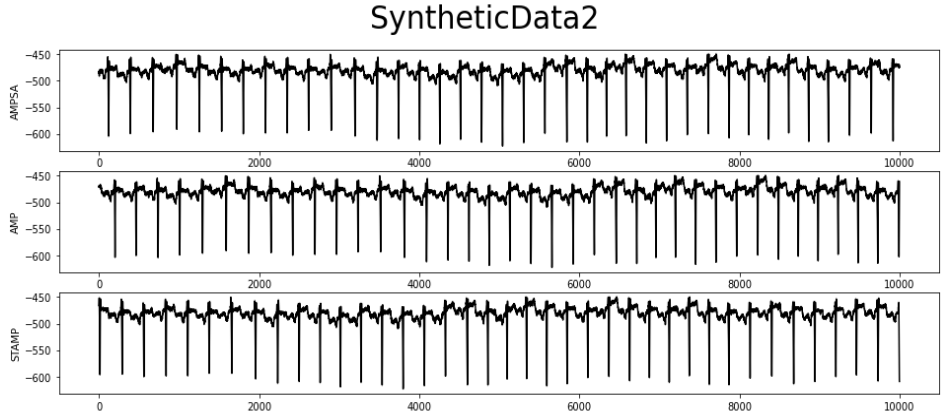
ภาพที่ 11.2 สีแดงและสีน้ำเงินแทนโมทีฟของข้อมูล SyntheticData2 ของภาพที่ 11.1 มีความยาว 10,000 จุดข้อมูล ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 11.3 สีแดงและสีน้ำเงินเป็นภาพขยายโมทีฟจากภาพที่ 11.2 ของข้อมูล SyntheticData2 มีความยาว 10,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

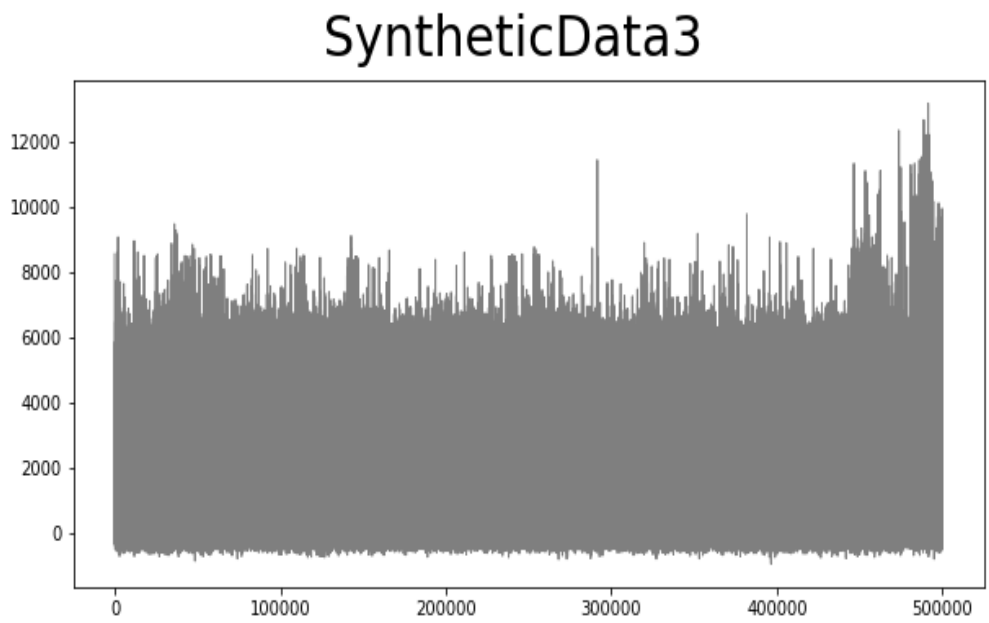


ภาพที่ 11.4 สีดำแทนดิสคอร์ดของข้อมูล SyntheticData2 มีความยาว 10,000 จุดข้อมูล ของภาพที่ 11.1 ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

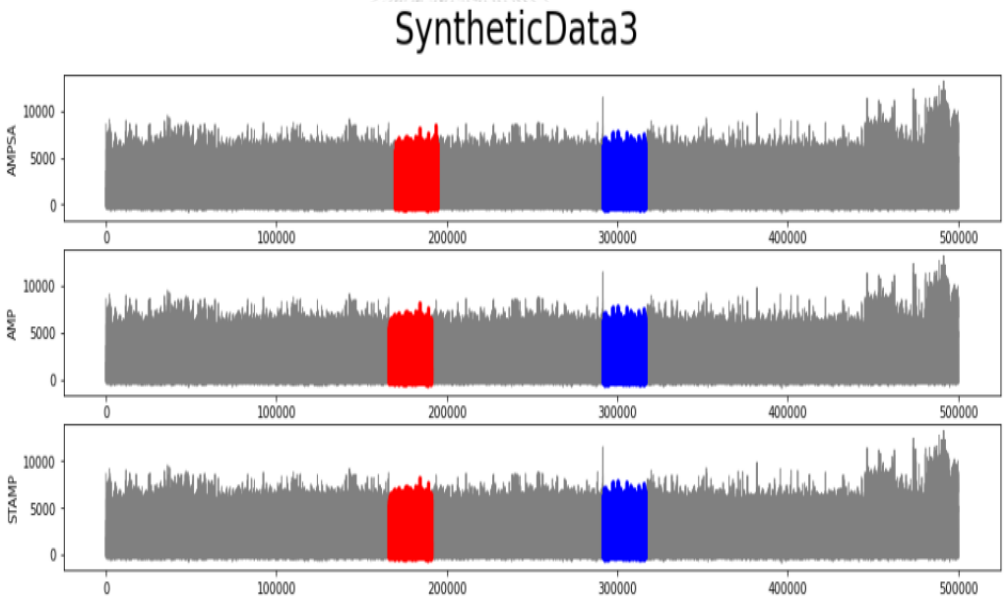


ภาพที่ 11.5 สีดำแทนดิสคอร์ดของข้อมูล SyntheticData2 เป็นภาพขยายดิสคอร์ดจากภาพที่ 11.4 มีความยาว 10,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

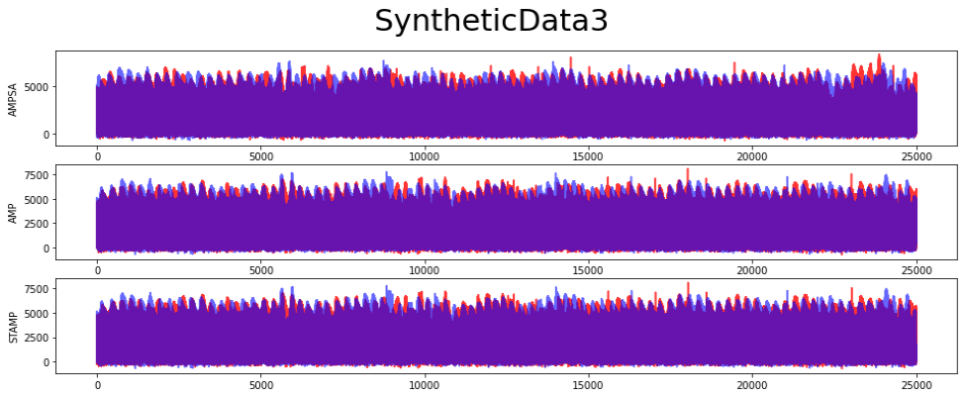
12. ชุดข้อมูลสังเคราะห์ SyntheticData3



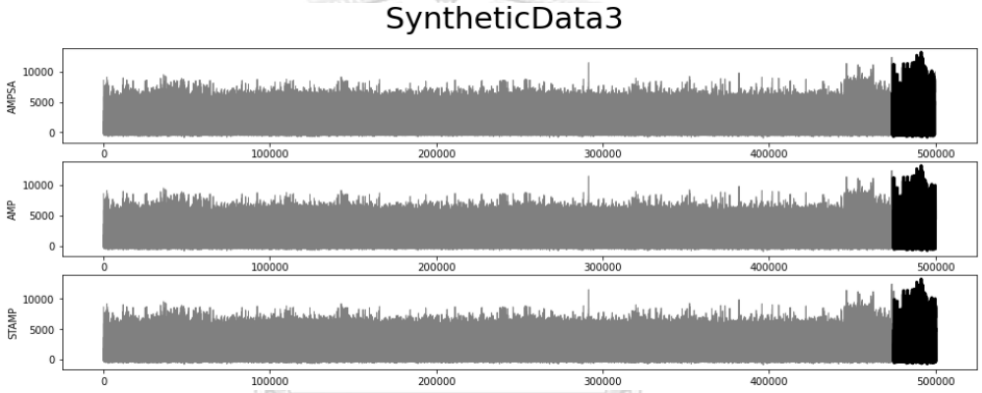
ภาพที่ 12.1 ชุดข้อมูลสังเคราะห์ SyntheticData3 มีความยาว 500,000 จุดข้อมูล



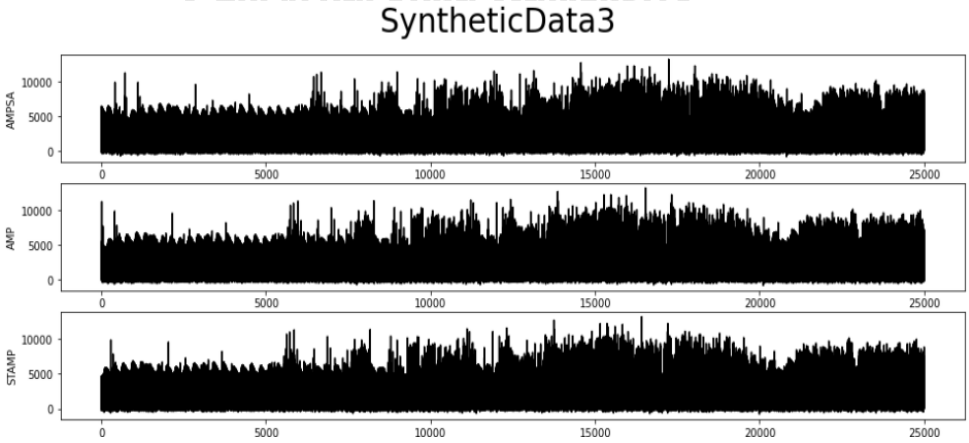
ภาพที่ 12.2 สีแดงและสีน้ำเงินแทนโมทีฟของข้อมูล SyntheticData3 ของภาพที่ 12.1 มีความยาว 25,000 จุดข้อมูล ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 12.3 สีแดงและสีน้ำเงินเป็นภาพขยายโมทีฟจากภาพที่ 12.2 ของข้อมูล SyntheticData3 มีความยาว 25,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 12.4 สีดำแทนดิสคอร์ดของข้อมูล SyntheticData3 มีความยาว 25,000 จุดข้อมูล ของภาพที่ 12.1 ที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ



ภาพที่ 12.5 สีดำแทนดิสคอร์ดของข้อมูล SyntheticData3 เป็นภาพขยายดิสคอร์ดจากภาพที่ 12.4 มีความยาว 25,000 จุดข้อมูลที่ได้จากอัลกอริทึม AMPSA AMP และ STAMP ตามลำดับ

บรรณานุกรม

1. Yeh, C.-C.M., et al. *Matrix profile I: all pairs similarity joins for time series: a unifying view that includes motifs, discords and shapelets*. in 2016 *IEEE 16th international conference on data mining (ICDM)*. 2016. Ieee.
2. Fu, T.-c., *A review on time series data mining.* Engineering Applications of Artificial Intelligence, 2011. 24(1): p. 164-181.
3. Cassisi, C., et al., *Motif discovery on seismic amplitude time series: The case study of mt etna 2011 eruptive activity.* Pure and Applied Geophysics, 2013. 170(4): p. 529-545.
4. Mueen, A., et al. *Exact discovery of time series motifs*. in *Proceedings of the 2009 SIAM international conference on data mining*. 2009. SIAM.
5. Ferreira, P.G., et al. *Mining approximate motifs in time series*. in *International Conference on Discovery Science*. 2006. Springer.
6. Yankov, D., et al. *Detecting time series motifs under uniform scaling*. in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2007.
7. Zheng, D., F. Li, and T. Zhao, *Self-adaptive statistical process control for anomaly detection in time series.* Expert Systems with Applications, 2016. 57: p. 324-336.
8. Meijster, A., J.B. Roerdink, and W.H. Hesselink, *A general algorithm for computing distance transforms in linear time*, in *Mathematical Morphology and its applications to image and signal processing*. 2002, Springer. p. 331-340.
9. Keogh, E.J. and M.J. Pazzani. *Scaling up dynamic time warping for datamining applications*. in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2000.
10. Lawrence, I. and K. Lin, *A concordance correlation coefficient to evaluate reproducibility.* Biometrics, 1989: p. 255-268.
11. Nunthanid, P., V. Niennattrakul, and C.A. Ratanamahatana. *Discovery of variable length time series motif*. in *The 8th Electrical Engineering/Electronics, Computer,*

Telecommunications and Information Technology (ECTI) Association of Thailand-Conference 2011. 2011. IEEE.

12. Yingchareonthawornchai, S., et al. *Efficient proper length time series motif discovery*. in 2013 *IEEE 13th International Conference on Data Mining*. 2013. IEEE.
13. Pariwatthanasak, K. and C.A. Ratanamahatana. *Time Series Motif Discovery Using Approximated Matrix Profile*. in *Third International Congress on Information and Communication Technology*. 2019. Springer.
14. Zhu, Y., et al. *Matrix profile ii: Exploiting a novel algorithm and gpus to break the one hundred million barrier for time series motifs and joins*. in 2016 *IEEE 16th international conference on data mining (ICDM)*. 2016. IEEE.
15. Bollerslev, T. and H.O. Mikkelsen, *Modeling and pricing long memory in stock market volatility.* Journal of econometrics, 1996. 73(1): p. 151-184.
16. Silva, D.F., et al. *SiMPle: Assessing Music Similarity Using Subsequences Joins*. in *ISMIR*. 2016.
17. Shorten, G. and M. Burke, *Use of dynamic time warping for accurate ECG signal timing characterization.* Journal of medical engineering & technology, 2014. 38(4): p. 188-201.
18. Mulla, R., *Hourly Energy Consumption*. 2018. [cited 2018 Aug 21]. Available from: https://www.kaggle.com/robikscube/hourly-energy-consumption
19. Batista, G.E., et al., *CID: an efficient complexity-invariant distance for time series.* Data Mining and Knowledge Discovery, 2014. 28(3): p. 634-669.
20. Mueen, A., *Time series motif discovery: dimensions and applications.* Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2014. 4(2): p. 152-159.
21. Chiu, B., E. Keogh, and S. Lonardi. *Probabilistic discovery of time series motifs*. in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2003.
22. Mises, R.v., *Probability, statistics and truth.* 1939.
23. Mueen, A. and E. Keogh. *Online discovery and maintenance of time series motifs*. in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2010.

24. Mueen, A., et al., *The fastest similarity search algorithm for time series subsequences under Euclidean distance.* url: www. cs. unm. edu/~ mueen/FastestSimilaritySearch. html (accessed 21 August 2018), 2017.
25. Wikipedia, *Convolution*. 2004. [cited 2018 Aug 21]. Available from: https://en.wikipedia.org/wiki/Convolution
26. Rakthanmanon, T., et al. *Time series epenthesis: Clustering time series streams requires ignoring some data*. in 2011 *IEEE 11th International Conference on Data Mining*. 2011. IEEE.
27. Haohan, W., *Confused student EEG brainwave data*. 2018. [cited 2018 Aug 21]. Available from: https://www.kaggle.com/wanghaohan/confused-eeg
28. Saeed, M., et al., *Multiparameter Intelligent Monitoring in Intensive Care II (MIMIC-II): a public-access intensive care unit database.* Critical care medicine, 2011. 39(5): p. 952.
29. Bappe., *Air Pollution in Seoul*. 2020. [cited 2020 Mar 3]. Available from: https://www.kaggle.com/bappekim/air-pollution-in-seoul
30. Mousavi, S.M., et al., *A Global Data Set of Seismic Signals for AI.* AGUFM, 2019. 2019: p. S52A-01.
31. Samaha, K., *Solar Flares from RHESSI Mission*. 2016. [cited 2018 Aug 21]. Available from: https://www.kaggle.com/khsamaha/solar-flares-rhessi
32. Budincsevity, N., *Weather in Szeged* 2006-2016. 2016. [cited 2018 Aug 21]. Available from: https://www.kaggle.com/budincsevity/szeged-weather
33. Niennattrakul, V., D. Wanichsan, and C.A. Ratanamahatana. *Accurate subsequence matching on data stream under time warping distance*. in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. 2009. Springer.

ประวัติผู้เขียน

ชื่อ-สกุล	นายชนะพล อ้นวงษา
วัน เดือน ปี เกิด	8 สิงหาคม 2536
สถานที่เกิด	ลพบุรี
วุฒิการศึกษา	ปริญญาวิทยาศาสตรบัณฑิต สาขาวิชาวิทยาศาสตร์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2558 ปัจจุบันเข้าศึกษาในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมศาสตร์คอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย