การหาโมทีฟและดิสคอร์ดสำหรับอนุกรมเวลา โดยใช้เมทริกซ์โพรไฟล์แบบประมาณ

นายกรกฎ ปริวัฒนศักดิ์

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2560

TIME SERIES MOTIF AND DISCORD DISCOVERY USING APPROXIMATED MATRIX PROFILE

Mr. Korakot Pariwatthanasak

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science Program in Computer Science
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2017

หัวข้อวิทยานิพนธ์	การหาโมทีฟและดิสคอร์ดสำหรับอนุกรมเวลา โดยใช้เมทริกซ์โพรไฟล์แบบประมาณ
โดย	นายกรกฎ ปริวัฒนศักดิ์
สาขาวิชา	วิทยาศาสตร์คอมพิวเตอร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	รองศาสตราจารย์ ดร.โชติรัตน์ รัตนามหัทธนะ

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

.....คณบดีคณะวิศวกรรมศาสตร์

(รองศาสตราจารย์ ดร.สุพจน์ เตชวรสินสกุล)

คณะกรรมการสอบวิทยานิพนธ์

.....ประธานกรรมการ

(อาจารย์ ดร.ดวงดาว วิชาดากุล)

.....อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

(รองศาสตราจารย์ ดร.โชติรัตน์ รัตนามหัทธนะ)

.....กรรมการภายนอกมหาวิทยาลัย

(ดร.เหมวรรณ ศิวรักษ์)

กรกฎ ปริวัฒนศักดิ์ : การหาโมทีฟและดิสคอร์ดสำหรับอนุกรมเวลา โดยใช้เมทริกซ์โพรไฟล์แบบประมาณ (TIME SERIES MOTIF AND DISCORD DISCOVERY USING APPROXIMATED MATRIX PROFILE) อ.ที่ปรึกษาวิทยานิพนธ์หลัก: รศ. ดร.โชติรัตน์ รัตนามหัทธนะ, หน้า.

การค้นพบโมทีฟคือการค้นหารูปแบบซึ่งเป็นลำดับย่อยที่อยู่ในข้อมูลอนุกรมเวลา การค้นพบโมทีฟเป็นปัญหาที่สำคัญในการทำเหมืองข้อมูลอนุกรมเวลาเนื่องจากสามารถประยุกต์ใช้ได้ในหลาย ๆ ขอบเขตความรู้ ในขณะเดียวกันการค้นพบดิสคอร์ดซึ่งก็เป็นวิธีการที่นิยมในการค้นหาความผิดปกติในข้อมูลอนุกรมเวลาด้วยเช่นกัน วิธีการหนึ่งที่ให้ผลลัพธ์สำหรับปัญหาการค้นพบโมทีฟและดิสคอร์ดได้ดีคือเมทริกซ์โพรไฟล์ เนื่องจากสามารถแก้ทั้งสองปัญหาได้โดยง่ายเพียงแค่คำนวณเมทริกซ์โพรไฟล์เท่านั้น อย่างไรก็ตามเวลาที่ใช้ในการคำนวณมีค่าสูงเมื่อข้อมูลอนุกรมเวลาใหญ่ขึ้น นอกจากนั้นเมทริกซ์โพรไฟล์ยังต้องการการกำหนดค่าพารามิเตอร์ความยาวของโมทีฟและดิสคอร์ดซึ่งผู้ใช้ไม่สามารถทราบได้แน่นอน

งานวิจัยนี้จึงนำเสนอเมทริกซ์โพรไฟล์แบบประมาณสำหรับทั้งสองปัญหาซึ่งลดเวลาในการคำนวณและยังคงให้ผลลัพธ์ที่ใกล้เคียงเดิมและนำเสนออัลกอริทึมสำหรับการค้นพบโมทีฟที่ไม่ต้องกำหนดค่าพารามิเตอร์ความยาวของโมทีฟอีกด้วย จากผลการทดลองบนข้อมูลสังเคราะห์และข้อมูลจริงพบว่า เมทริกซ์โพรไฟล์แบบประมาณสามารถลดเวลาในการคำนวณได้เป็นจำนวนมากและยังคงได้โมทีฟและดิสคอร์ดผลลัพธ์ที่ใกล้เคียงกับเมทริกซ์โพรไฟล์ นอกจากนั้นอัลกอริทึมการค้นพบโมทีฟที่นำเสนอยังให้ผลลัพธ์ที่ถูกต้องบนความยาวที่เหมาะสมโดยไม่จำเป็นต้องกำหนดค่าพารามิเตอร์ความยาวของโมทีฟก่อน

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิทยาศาสตร์คอมพิวเตอร์ ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

ปีการศึกษา 2560

5970103721 : MAJOR COMPUTER SCIENCE
KEYWORDS: TIME SERIES DATA MINING / MOTIF DISCOVERY / DISCORD DISCOVERY / MATRIX PROFILE

KORAKOT PARIWATTHANASAK: TIME SERIES MOTIF AND DISCORD DISCOVERY USING APPROXIMATED MATRIX PROFILE. ADVISOR: ASSOC. PROF. CHOTIRAT RATANAMAHATANA, Ph.D., pp.

Time series motif discovery, a procedure of finding patterns in a long time series sequence, has become one of the most prevalent time series mining tasks as it could be applied in various domains. Meanwhile, time series discord discovery, a procedure of detecting abnormality in a time series data, has also become the most popular technique in anomaly detection problem. Recently, matrix profile has become the competitive method for motif and discord discovery because if matrix profile is given, both time series problems can easily solve. However, its computation takes too much time for large time series data. Moreover, the parameter of motif and discord length has to be defined but it can not be trivially done.

This dissertation proposes an approximated version of the matrix profile for both problems, which reduces time computation and still get impressively correct motif results and also proposes a parameter-free algorithm for motif discovery task that solves the pre-parameter-defined problem. The experiment results on both synthetic and real datasets reconfirm that our approximated matrix profile can speed up the computation by a large margin and still obtain the motif and discord similar to the motif from the full matrix profile. Also, our parameter-free algorithm can give the reasonable motif results with proper length.

Department:	Computer Engineering	Student's Signature	.....
Field of Study:	Computer Science	Advisor's Signature	.....
Academic Year:	2017		

กิตติกรรมประกาศ

วิทยานิพนธ์นี้สำเร็จลุล่วงได้ด้วยความกรุณาจาก รองศาสตราจารย์ ดร.โชติรัตน์ รัตนามหัทธนะ อาจารย์ที่ปรึกษา ผู้คอยให้คำปรึกษา ให้แง่คิด ทั้งในด้านวิชาการและด้านอื่น ๆ และเป็นผู้ตรวจทานแก้ไขทำให้วิทยานิพนธ์เล่มนี้สำเร็จลุล่วงไปได้ด้วยดี ขอขอบพระคุณเป็นอย่างสูงมา ณ ที่นี้

ขอขอบพระคุณ ดร.ดวงดาว วิชาดากุล และ ดร.เหมวรรณ ศิวรักษ์ ผู้ให้เกียรติเป็นประธานกรรมการและกรรมการสอบวิทยานิพนธ์ ที่ชี้แนะแนวทางในการปรับปรุงวิทยานิพนธ์ให้มีคุณภาพยิ่งขึ้น

ที่สำคัญที่สุด ขอขอบพระคุณ คุณพ่อ คุณแม่ และครอบครัว ที่คอยเป็นกำลังใจในการทำงานวิจัยนี้เสมอมา จนกระทั่งวิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี

สารบัญ

หน้า

บทคัดย่อภาษาไทย ..... ง
บทคัดย่อภาษาอังกฤษ ..... จ
กิตติกรรมประกาศ ..... ฉ
สารบัญ ..... ช
สารบัญตาราง ..... ญ
สารบัญภาพ ..... ฎ
บทที่ 1 บทนำ 1
1.1 ที่มาและความสำคัญของปัญหา 1
1.2 วัตถุประสงค์ของงานวิจัย 5
1.3 ขอบเขตของงานวิจัย 5
1.4 ประโยชน์ที่ได้รับจากงานวิจัย 5
1.5 วิธีดำเนินงานวิจัย 5
1.6 ผลงานวิจัยที่ได้ตีพิมพ์ 6
บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง 7
2.1 ทฤษฎีที่เกี่ยวข้อง 7
2.1.1 ข้อมูลอนุกรมเวลา (Time Series Data) 7
อนุกรมเวลา (Time Series) 8
ลำดับย่อย (Subsequence) 8
2.1.2 ระยะห่างยุคลิด (Euclidean Distance) 8
2.1.3 สัมประสิทธิ์สหสัมพันธ์ (Correlation Coefficient) 9
2.1.4 การทำให้เป็นบรรทัดฐาน (Normalization) 9
การทำให้เป็นบรรทัดฐานแบบซี (Z-Normalization) 9

หน้า

ระยะห่างยุคลิดที่รองรับการทำให้เป็นบรรทัดฐานแบบซี (Z-Normalized Euclidean Distance) 10
2.1.5 โมทีฟ (Motif) 10
2.1.6 ดิสคอร์ด (Discord) 10
2.1.7 ปัญหาวันเกิด (Birthday Paradox Problem) [25] 10
หลักการที่ 1 11
หลักการที่ 2 11
2.2 งานวิจัยที่เกี่ยวข้อง 12
2.2.1 การค้นพบโมทีฟ (Motif Discovery) 12
2.2.2 การค้นพบดิสคอร์ด (Discord Discovery) 13
2.2.3 เมทริกซ์โพรไฟล์ (Matrix Profile) 14
คำนิยามและสัญลักษณ์ (Definition and Notation) 14
STAMP Algorithm 15
2.2.4 อัลกอริทึมการค้นพบโมทีฟที่ความยาวเหมาะสม (Proper Length Motif Discovery) 18
บทที่ 3 อัลกอริทึมการค้นพบโมทีฟและดิสคอร์ดของอนุกรมเวลา โดยใช้เมทริกซ์โพรไฟล์แบบประมาณ 22
3.1 เมทริกซ์โพรไฟล์แบบประมาณ (Approximated Matrix Profile) 22
3.1.1 ลำดับของการคำนวณเมทริกซ์โพรไฟล์ (Order of Computation) 22
3.1.2 จำนวนของลำดับย่อยที่ใช้ในการคำนวณเมทริกซ์โพรไฟล์หรือจำนวนการวนซ้ำ (Number of Iterations) 22
3.1.3 อัลกอริทึมเอเอ็มพี (Approximated Matrix Profile) 24

หน้า

3.2 อัลกอริทึมสำหรับการค้นพบโมทีฟสำหรับความยาวที่เหมาะสมโดยใช้เมทริกซ์โพรไฟล์แบบประมาณ (Proper Length Motif Discovery Algorithm using Approximated Matrix Profile) 26
บทที่ 4 การทดลองและวิเคราะห์ผลการทดลอง 30
4.1 การทดลองสำหรับอัลกอริทึมเอเอ็มพี 30
4.1.1 ข้อมูลสังเคราะห์ (Synthetic data) 30
4.1.2 ข้อมูลจริง (Real World Data) 30
4.1.3 การวิเคราะห์เวลาและหน่วยความจำที่ต้องใช้ในการประมวลผล (Time and Space Complexity) 31
4.1.4 ความแม่นยำของโมทีฟและดิสคอร์ดผลลัพธ์ 33
4.1.5 กรณีศึกษาข้อมูลเพลง (Case Study : Music Data) 35
4.2 การทดลองสำหรับอัลกอริทึมพีแอลเอเอ็มพี 38
4.2.1 กรณีศึกษาข้อมูลแผ่นดินไหว (Case Study : Seismic Data) 38
บทที่ 5 สรุปผลการวิจัย และข้อเสนอแนะ 41
5.1 สรุปผลงานวิจัย 41
5.2 ข้อจำกัดและข้อเสนอแนะ 42
รายการอ้างอิง 43
ภาคผนวก 48
ภาคผนวก ก 49
ประวัติผู้เขียนวิทยานิพนธ์ 81

สารบัญตาราง

ตารางที่ 1.1 เวลาที่ใช้ในการคำนวณเมทริกซ์โพรไฟล์สำหรับความยาวของอนุกรมเวลาที่แตกต่างกัน เมื่อกำหนดให้ความยาวของลำดับย่อยเป็น 256 (ที่มา : [14]) 4

ตารางที่ 2.1 การคำนวณ Sliding Dot Products (ที่มา : [31]) 16

ตารางที่ 2.2 อัลกอริทึม MASS (ที่มา : [31]) 17

ตารางที่ 2.3 อัลกอริทึม STAMP (ที่มา : [8]) 17

ตารางที่ 2.4 อัลกอริทึมการค้นพบโมทีฟที่ความยาวเหมาะสม (ที่มา : [21]) 20

ตารางที่ 3.1 การหาจำนวนลำดับย่อยที่ใช้ในการคำนวณเมทริกซ์โพรไฟล์ 23

ตารางที่ 3.2 อัลกอริทึม AMP 24

ตารางที่ 3.3 อัลกอริทึม PLAMP 27

ตารางที่ 4.1 ตารางแสดงรายละเอียดข้อมูลสังเคราะห์ที่ใช้ในงานวิจัย 30

ตารางที่ 4.2 ตารางแสดงรายละเอียดข้อมูลจริงที่ใช้ในงานวิจัย 31

ตารางที่ 4.3 ตารางแสดงตัวอย่างความสัมพันธ์ระหว่างค่า k และ n 32

ตารางที่ 4.4 ตารางแสดงผลการทดลองอัลกอริทึมเอเอ็มพีบนข้อมูลสังเคราะห์ 34

ตารางที่ 4.5 ตารางแสดงผลการทดลองอัลกอริทึมเอเอ็มพีบนข้อมูลจริง 35

ตารางที่ 4.6 ตารางแสดงผลการทดลองการค้นพบโมทีฟและดิสคอร์ดบนข้อมูลเพลง 37

สารบัญภาพ

ภาพที่ 1.1 ตัวอย่างการค้นพบโมทีฟในข้อมูลอุตสาหกรรม 2
ภาพที่ 1.2 ตัวอย่างการค้นพบดิสคอร์ดในข้อมูลคลื่นไฟฟ้าหัวใจ (ECG) (ที่มา : [14]) 2
ภาพที่ 1.3 ตัวอย่างเมทริกซ์โพรไฟล์ที่คำนวณจากข้อมูลแผ่นดินไหว (ที่มา : [14]) 3
ภาพที่ 1.4 โมทีฟผลลัพธ์ได้จากการหาค่าที่น้อยที่สุดของเมทริกซ์โพรไฟล์ (ที่มา : [14]) 4
ภาพที่ 1.5 ดิสคอร์ดผลลัพธ์ที่ได้จากการหาค่าที่มากที่สุดของเมทริกซ์โพรไฟล์ (ที่มา : [14]) 4
ภาพที่ 2.1 ภาพแสดงข้อมูลราคาหุ้นบริษัท Google ตั้งแต่ปี 2010 ถึงปี 2018 7
ภาพที่ 2.2 ภาพแสดงตัวอย่างของลำดับย่อยของข้อมูลราคาหุ้นบริษัท Google เฉพาะในปี 2017 (เส้นประ) 8
ภาพที่ 2.3 การวัดระยะห่างยุคลิดระหว่าง 2 อนุกรมเวลา (ที่มา : [9]) 9
ภาพที่ 2.4 กราฟระหว่างจำนวนคนและโอกาสที่จะมีคนเกิดตรงกัน (สีแดง) และกราฟระหว่างจำนวนคนและโอกาสที่จะมีคนเกิดไม่ตรงกัน (สีน้ำเงิน) 12
ภาพที่ 2.5 เวลาที่ใช้ในการหาโมทีฟของอนุกรมเวลาความยาว 65,536 จุดข้อมูล เปรียบเทียบระหว่างอัลกอริทึม MK และ PBMD (ที่มา : [14]) 13
ภาพที่ 2.6 ภาพแสดงขั้นตอนการคำนวณผลคูณจุดตามอัลกอริทึม SlidingDotProduct 16
ภาพที่ 2.7 ตัวอย่างเมทริกซ์โพรไฟล์ในข้อมูลการใช้พลังงาน (Power Usage Data) โมทีฟผลลัพธ์สามารถหาได้จากค่าที่น้อยที่สุดในเมทริกซ์โพรไฟล์ (ที่มา : [14]) 18
ภาพที่ 2.8 ตัวอย่างเมทริกซ์โพรไฟล์ในข้อมูลการใช้พลังงาน (Power Usage Data) ดิสคอร์ดผลลัพธ์สามารถหาได้จากค่าที่มากที่สุดในเมทริกซ์โพรไฟล์ (ที่มา : [14]) 18
ภาพที่ 3.1 ภาพตัวอย่างแสดงเมทริกซ์โพรไฟล์ (บน) และเมทริกซ์โพรไฟล์แบบประมาณ (ล่าง) 26
ภาพที่ 4.1 ภาพแสดงเวลาที่ใช้ของอัลกอริทึมเอเอ็มพี (เส้นประ) และอัลกอริทึมแสตมป์ (เส้นทึบ) บนข้อมูลสังเคราะห์ที่มีความยาวแตกต่างกัน 32
ภาพที่ 4.2 ภาพแสดงเวลาที่ใช้ของอัลกอริทึมเอเอ็มพี (เส้นประ) และอัลกอริทึมแสตมป์ (เส้นทึบ) . 33
ภาพที่ 4.3 ข้อมูลสัมประสิทธิ์เซปสตรัลที่คำนวณบนแกนความถี่แบบเมลของเพลง Stayin' alive ที่มีความยาว 12,356 จุดข้อมูล 36

ภาพที่ 4.4 โมทีฟผลลัพธ์ที่ความยาว 432 จุดข้อมูลจากอัลกอริทึมเอเอ็มพี (ซ้าย) และอัลกอริทึมแสตมป์ (ขวา) 37
ภาพที่ 4.5 ดิสคอร์ดผลลัพธ์ที่ความยาว 432 จุดข้อมูลจากอัลกอริทึมเอเอ็มพี (ซ้าย) และอัลกอริทึมแสตมป์ (ขวา) 38
ภาพที่ 4.6 ข้อมูลแผ่นดินไหวมีขนาด 40,000 จุดข้อมูล 39
ภาพที่ 4.7 โมทีฟผลลัพธ์ที่ได้จากอัลกอริทึมพีแอลเอเอ็มพีที่ตำแหน่ง 15,000 และตำแหน่ง 30,000 40
ภาพที่ 4.8 โมทีฟผลลัพธ์จากอัลกอริทึมพีแอลเอเอ็มพีที่ความยาว 1,016 จุดข้อมูล 40
ภาพที่ ก.1 ข้อมูลสังเคราะห์ชุดที่ 1 (RW1) 51
ภาพที่ ก.2 ข้อมูลสังเคราะห์ชุดที่ 2 (RW2) 54
ภาพที่ ก.3 ข้อมูลสังเคราะห์ชุดที่ 3 (RW3) 56
ภาพที่ ก.4 ข้อมูลสังเคราะห์ชุดที่ 4 (RW4) 59
ภาพที่ ก.5 ข้อมูลสังเคราะห์ชุดที่ 5 (RW5) 61
ภาพที่ ก.6 ชุดข้อมูลจริง Astro 64
ภาพที่ ก.7 ชุดข้อมูลจริง ECG 66
ภาพที่ ก.8 ชุดข้อมูลจริง EEG 69
ภาพที่ ก.9 ชุดข้อมูลจริง EMG 71
ภาพที่ ก.10 ชุดข้อมูลจริง GAP 74
ภาพที่ ก.11 ชุดข้อมูลจริง Insect 76
ภาพที่ ก.12 ชุดข้อมูลจริง Seismic 79

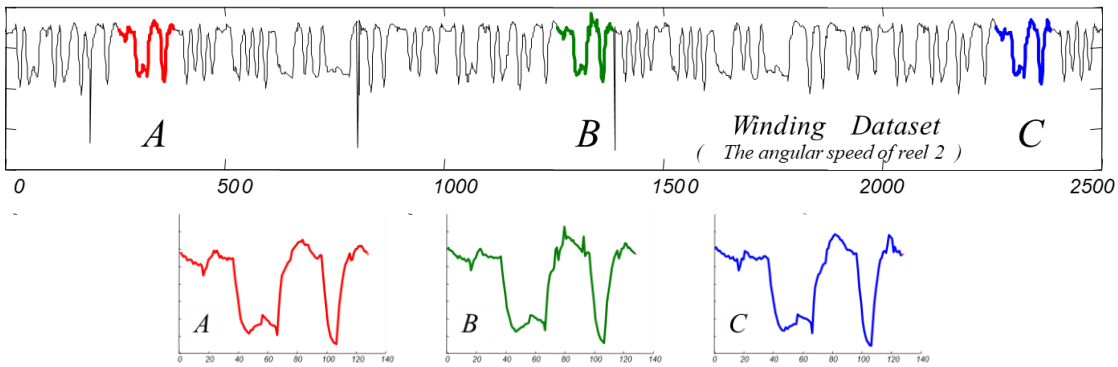
บทที่ 1 บทนำ

1.1 ที่มาและความสำคัญของปัญหา

ข้อมูลอนุกรมเวลา (Time Series Data) คือ ชุดของข้อมูลที่เก็บรวบรวมตามระยะเวลาเป็นช่วง ๆ อย่างต่อเนื่องกัน ซึ่งข้อมูลอนุกรมเวลานั้นสามารถพบได้อย่างแพร่หลายในชีวิตประจำวัน ตัวอย่างของข้อมูลประเภทนี้ได้แก่ ข้อมูลหุ้น ข้อมูลเสียง ข้อมูลอุณหภูมิ ข้อมูลคลื่นไฟฟ้า เป็นต้น ข้อมูลเหล่านี้สามารถนำมาวิเคราะห์เพื่อค้นหาความรู้ (Knowledge) ที่ซ่อนอยู่ในข้อมูลได้ ซึ่งการค้นหาและสกัดความรู้ออกมาจากข้อมูลนี้เรียกว่า การทำเหมืองข้อมูล (Data Mining) แต่ข้อมูลอนุกรมเวลามีคุณสมบัติและลักษณะที่แตกต่างไปจากข้อมูลทั่วไป ดังนั้นสำหรับข้อมูลอนุกรมเวลาจะถูกเรียกว่า การทำเหมืองข้อมูลอนุกรมเวลา (Time Series Data Mining) [1,2]

เนื่องจากลักษณะของข้อมูลอนุกรมเวลา เช่น ขึ้นอยู่กับเวลา มีขนาดใหญ่ มีหลายมิติ มีค่าเป็นจำนวนจริง และถูกปรับให้เป็นปัจจุบันตลอดเวลา ทำให้การทำเหมืองข้อมูลอนุกรมเวลานั้นแตกต่างออกไปจากการทำเหมืองข้อมูลทั่วไป แต่ยังคงมีวัตถุประสงค์คือการค้นหาและสกัดความรู้ออกมาจากข้อมูล ตัวอย่างปัญหาส่วนใหญ่ของการทำเหมืองข้อมูลอนุกรมเวลา เช่น การตรวจจับสิ่งผิดปกติ (Anomaly Detection) หรือการค้นพบดิสคอร์ด (Discord Discovery) [3] การค้นพบโมทีฟ (Motif Discovery) [4] การจัดกลุ่ม (Clustering) [5] การจำแนกประเภท (Classification) [6] เป็นต้น ซึ่งในงานวิจัยนี้สนใจปัญหาการทำเหมืองข้อมูลอนุกรมเวลา 2 ชนิด คือ การค้นพบโมทีฟและการค้นพบดิสคอร์ด

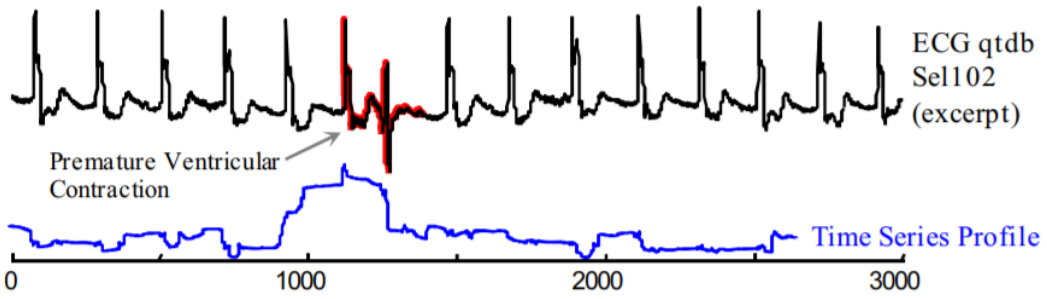
การค้นพบโมทีฟ (Motif Discovery) คือ การค้นหารูปแบบ (Pattern) ซึ่งเป็นลำดับย่อย (Subsequence) ที่อยู่ในข้อมูลอนุกรมเวลา นักวิจัยทางด้านการทำเหมืองข้อมูลอนุกรมเวลาสนใจการค้นพบโมทีฟเนื่องจาก โมทีฟนั้นมักจะประกอบด้วยสารสนเทศ (Information) ที่สำคัญเกี่ยวกับข้อมูลที่ศึกษาอยู่ ภาพที่ 1.1 แสดงตัวอย่างของการค้นพบโมทีฟในข้อมูลอุตสาหกรรม ในปัจจุบันการค้นพบโมทีฟได้ถูกนำไปศึกษาในข้อมูลหลาย ๆ ด้าน เช่น Animation [7], Entomology [8], Weather prediction [9], Seismology [10], Music [11] เป็นต้น คำนิยามของโมทีฟนั้นมีได้หลายแบบขึ้นอยู่กับปัญหาที่สนใจ ในปี 2014 Mueen A. [12] ได้เสนอว่ามี 2 คำนิยามของโมทีฟที่น่าสนใจ ได้แก่ โมทีฟที่ขึ้นอยู่กับความเหมือน (Similarity-based Motif) และโมทีฟที่ขึ้นอยู่กับความถี่ (Support-based or Frequency-based Motif) ซึ่งแต่ละความหมายจะถูกนำไปประยุกต์ใช้ในขอบเขต (Domain) และปัญหาที่แตกต่างกัน



ภาพที่ 1.1 ตัวอย่างการค้นพบโมทีฟในข้อมูลอุตสาหกรรม

(ที่มา : http://slideplayer.com/slide/5011127/)

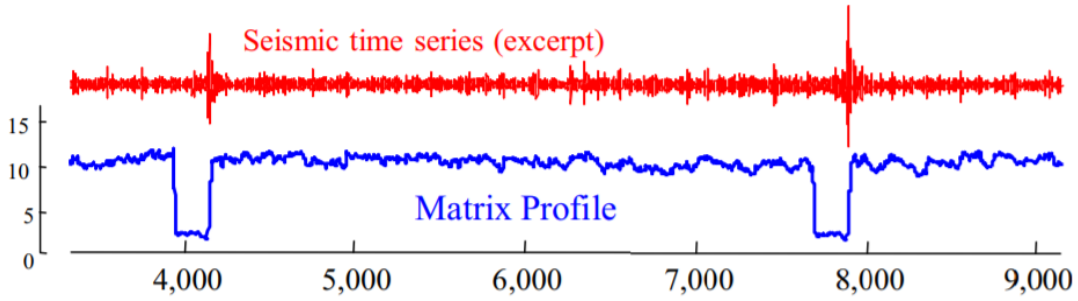
การค้นพบดิสคอร์ด (Discord Discovery) คือ การค้นหาลำดับย่อย (Subsequence) ที่มีรูปร่างแตกต่างจากลำดับย่อยอื่นมากที่สุดในอนุกรมเวลา ซึ่งลำดับย่อยนี้ถูกเรียกว่า ดิสคอร์ด (Discord) โดยส่วนใหญ่การค้นพบดิสคอร์ดมักถูกนำไปใช้ในปัญหาการตรวจจับสิ่งผิดปกติ (Anomaly Detection) เพราะลำดับย่อยที่มีรูปร่างแตกต่างมากมักจะเป็นสิ่งผิดปกติในข้อมูลอนุกรมเวลานั้น ๆ ตัวอย่างการประยุกต์ใช้ของการค้นพบดิสคอร์ด เช่น การตรวจจับสิ่งผิดปกติในข้อมูลคลื่นไฟฟ้าหัวใจ (Electrocardiogram ECG) [13] การตรวจจับการบุกรุกในระบบเครือข่าย (Network Intrusion Detection) [3] เป็นต้น ภาพที่ 1.2 แสดงตัวอย่างของการค้นพบดิสคอร์ดในข้อมูลคลื่นไฟฟ้าหัวใจ



ภาพที่ 1.2 ตัวอย่างการค้นพบดิสคอร์ดในข้อมูลคลื่นไฟฟ้าหัวใจ (ECG) (ที่มา : [14])

การวัดความคล้าย (Similarity measure) คือการวัดระยะห่าง (Distance) ของลำดับย่อยในอนุกรมเวลา ซึ่งเป็นสิ่งที่สำคัญสำหรับการค้นพบโมทีฟและดิสคอร์ด โดยทั่วไปการวัดความเหมือนที่นิยมใช้ในการค้นพบโมทีฟและดิสคอร์ดเช่น ระยะห่างยุคลิด (Euclidean Distance) [15,16] ไดนามิกไทม์วอร์ปปิง (Dynamic Time Warping) [17,18] สัมประสิทธิ์สหสัมพันธ์ (Correlation Coefficient) [19] สำหรับปัญหาการค้นพบโมทีฟและดิสคอร์ดงานวิจัย [12] ได้อ้างว่า วิธีวัดระยะห่าง (Distance Function) ที่แตกต่างกันมีผลน้อยมากกับผลลัพธ์ เพราะฉะนั้นวิธีวัดระยะห่างที่ง่ายและมีประสิทธิภาพที่สุดจึงเหมาะสมในการนำไปใช้ในปัญหาการค้นพบโมทีฟและดิสคอร์ด

ในปี 2016 Yeh C.-C. M. et al. [14] ได้นำเสนอ เมทริกซ์โพรไฟล์ (Matrix Profile) ซึ่งเป็นเวกเตอร์ (Vector) สำหรับเก็บข้อมูลระยะห่างระหว่างแต่ละลำดับย่อยในอนุกรมเวลา และลำดับย่อยเพื่อนบ้านใกล้ที่สุด (1 Nearest Neighbor Subsequence) ในอนุกรมเวลาเดียวกัน โดยใช้ระยะห่างยุคลิดเป็นวิธีวัดระยะห่าง (ดังแสดงในภาพที่ 1.3) สิ่งที่น่าสนใจสำหรับเมทริกซ์โพรไฟล์ คือ เมื่อคำนวณเมทริกซ์โพรไฟล์ได้แล้วนั้น ปัญหาทางด้านการทำเหมืองข้อมูลอนุกรมเวลาจะสามารถแก้ได้โดยง่ายเพราะระยะห่างที่ถูกเก็บในเมทริกซ์โพรไฟล์นั้นเป็นสารสนเทศที่สำคัญสำหรับปัญหาในการทำเหมืองข้อมูลอนุกรมเวลา ตัวอย่างปัญหาเช่น การค้นพบโมทีฟซึ่งสามารถดึงลำดับย่อยที่มีระยะห่างที่น้อยที่สุดในเมทริกซ์โพรไฟล์ได้ ในทำนองเดียวกัน การตรวจจับสิ่งผิดปกติก็สามารถดึงลำดับย่อยที่มีระยะห่างมากที่สุดในเมทริกซ์โพรไฟล์ได้เช่นเดียวกัน เป็นต้น นอกจากนั้นเมทริกซ์โพรไฟล์ยังรองรับการทำให้เป็นบรรทัดฐานของลำดับย่อย (Subsequence Normalization) ซึ่งเป็นอีกปัจจัยสำคัญสำหรับการค้นหาโมทีฟและดิสคอร์ดให้มีความแม่นยำมากขึ้น

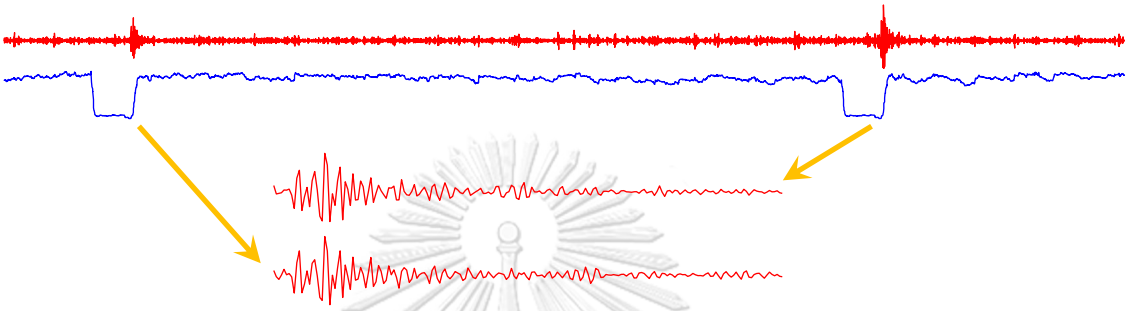


ภาพที่ 1.3 ตัวอย่างเมทริกซ์โพรไฟล์ที่คำนวณจากข้อมูลแผ่นดินไหว (ที่มา : [14])

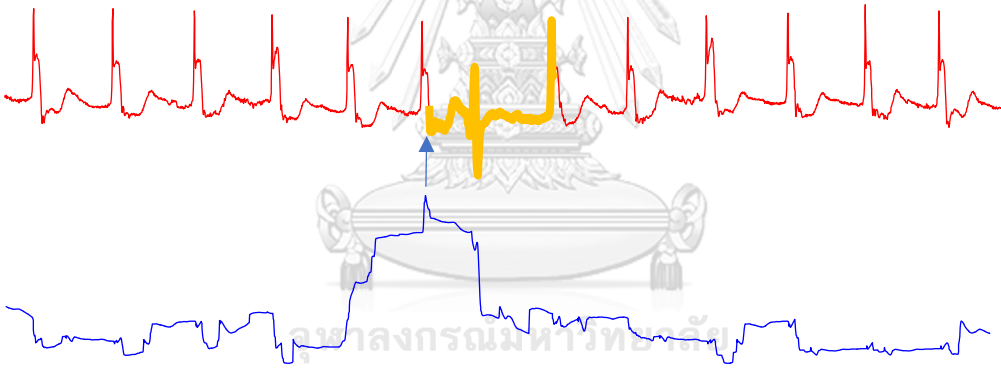
อย่างไรก็ตามการคำนวณเมทริกซ์โพรไฟล์นั้นยังใช้เวลานานพอสมควรสำหรับข้อมูลอนุกรมเวลาขนาดใหญ่ ตัวอย่างเช่นในตารางที่ 1.1 อนุกรมเวลาที่มีความยาว 2^{18} จุดข้อมูล ใช้เวลาในการคำนวณถึง 70.4 นาที สำหรับปัญหาการค้นพบโมทีฟและดิสคอร์ดนั้น ส่วนที่เป็นผลลัพธ์คือค่าที่น้อยที่สุดและค่าที่มากที่สุดในเมทริกซ์โพรไฟล์ เพราะฉะนั้นเพื่อลดระยะเวลาในการคำนวณเมทริกซ์โพรไฟล์อาจไม่จำเป็นต้องคำนวณเมทริกซ์โพรไฟล์ทั้งหมด กล่าวคือคำนวณเฉพาะบางส่วนของเมทริกซ์โพรไฟล์ เพื่อให้ได้ค่าที่น้อยที่สุดและค่าที่มากที่สุดก็เพียงพอ (ดังแสดงในภาพที่ 1.4 และ 1.5)

ตารางที่ 1.1 เวลาที่ใช้ในการคำนวณเมทริกซ์โพรไฟล์สำหรับความยาวของอนุกรมเวลาที่แตกต่างกัน เมื่อกำหนดให้ความยาวของลำดับย่อยเป็น 256 (ที่มา : [14])

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



ภาพที่ 1.4 โมทีฟผลลัพธ์ได้จากการหาค่าที่น้อยที่สุดของเมทริกซ์โพรไฟล์ (ที่มา : [14])



ภาพที่ 1.5 ดิสคอร์ดผลลัพธ์ที่ได้จากการหาค่าที่มากที่สุดของเมทริกซ์โพรไฟล์ (ที่มา : [14])

นอกจากนั้นเมทริกซ์โพรไฟล์ยังต้องการการกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อย ซึ่งสามารถกำหนดได้ยากและยังเป็นปัญหาในการค้นพบโมทีฟอยู่ [20] ในบางครั้งจำเป็นต้องใช้ความรู้หรือผู้เชี่ยวชาญเฉพาะด้านในการกำหนด ในปี 2013 งานวิจัย [21] ได้นำเสนออัลกอริทึมการค้นพบโมทีฟที่ความยาวเหมาะสม (Proper Length Motif Discovery) ซึ่งช่วยแก้ปัญหาในการกำหนดความยาวของลำดับย่อยได้ เพราะฉะนั้นเพื่อลดปัญหาดังกล่าว การประยุกต์ใช้อัลกอริทึมของงานวิจัย [21] จึงเป็นประโยชน์ต่อเมทริกซ์โพรไฟล์

จากปัญหาหลัก 2 ปัญหาคือ เมทริกซ์โพรไฟล์ใช้เวลาในการคำนวณสำหรับข้อมูลอนุกรมเวลาขนาดใหญ่ค่อนข้างนาน และผู้ใช้จำเป็นต้องกำหนดความยาวของลำดับย่อยก่อน ดังนั้นงานวิจัยนี้จึงนำเสนอวิธีการคำนวณเมทริกซ์โพรไฟล์แบบประมาณ (Approximated Matrix profile) เพื่อลด

เวลาในการคำนวณเมทริกซ์โพรไฟล์ แต่ยังให้ผลลัพธ์ของโมทีฟและดิสคอร์ดใกล้เคียงกับเมทริกซ์โพรไฟล์ และอัลกอริทึมสำหรับการค้นพบโมทีฟที่ไม่จำเป็นต้องกำหนดพารามิเตอร์ความยาวของลำดับย่อย โดยใช้ประโยชน์จากเมทริกซ์โพรไฟล์แบบประมาณและอัลกอริทึมการค้นพบโมทีฟของงานวิจัย [21]

1.2 วัตถุประสงค์ของงานวิจัย

1) เพื่อนำเสนอเมทริกซ์โพรไฟล์แบบประมาณ (Approximated Matrix Profile) ในปัญหาการค้นพบโมทีฟและดิสคอร์ด ซึ่งใช้เวลาในการคำนวณน้อยกว่าเมทริกซ์โพรไฟล์ (Matrix Profile)

1.3 ขอบเขตของงานวิจัย

1) ในงานวิจัยนี้จะทำการเปรียบเทียบผลลัพธ์กับงานวิจัย [14] เพื่อการเปรียบเทียบผลลัพธ์ของโมทีฟและดิสคอร์ด
2) ในงานวิจัยนี้จะไม่เปรียบเทียบผลในเมทริกซ์โพรไฟล์รุ่นหน่วยประมวลผลกราฟิก (GPU-STAMP) [22] เนื่องจากถ้าปรับปรุงเมทริกซ์โพรไฟล์รุ่นปกติได้ ก็สามารถนำไปประยุกต์ใช้กับเมทริกซ์โพรไฟล์รุ่นหน่วยประมวลผลกราฟิกได้

1.4 ประโยชน์ที่ได้รับจากงานวิจัย

1) ได้อัลกอริทึมสำหรับปัญหาการค้นพบโมทีฟและดิสคอร์ด ที่ใช้เวลาในการคำนวณน้อยลง แต่ได้ผลลัพธ์ที่เหมือนหรือใกล้เคียงกับผลลัพธ์ที่ถูกต้อง

1.5 วิธีดำเนินงานวิจัย

1) ศึกษาเกี่ยวกับการทำเหมืองข้อมูลอนุกรมเวลา
2) ศึกษาเกี่ยวกับปัญหาการค้นพบโมทีฟและดิสคอร์ดด้วยอัลกอริทึมแบบต่าง ๆ จากงานวิจัยที่เกี่ยวข้อง
3) ทดลองสร้างโปรแกรมสำหรับปัญหาการค้นพบโมทีฟและดิสคอร์ดเพื่อให้ได้ผลลัพธ์เท่ากับผลเฉลยของอัลกอริทึมที่ศึกษา
4) สรุปข้อดีข้อเสียของแต่ละงานวิจัย และกำหนดงานวิจัยที่จะใช้เป็นงานวิจัยหลักในการประยุกต์ต่อยอดและเปรียบเทียบผลลัพธ์
5) ศึกษางานวิจัยเมทริกซ์โพรไฟล์โดยละเอียด และหาจุดที่จะสามารถปรับปรุงต่อยอดให้ดีขึ้นได้
6) ออกแบบและสร้างโปรแกรมเมทริกซ์โพรไฟล์แบบประมาณ
7) ทดสอบโปรแกรมที่สร้างขึ้น โดยเก็บผลลัพธ์เป็นเวลาที่ใช้ในการคำนวณและคำตอบของโมทีฟและดิสคอร์ดที่ได้จากเมทริกซ์โพรไฟล์แบบประมาณ

8) เปรียบเทียบผลลัพธ์ระหว่างเมทริกซ์โพรไฟล์และเมทริกซ์โพรไฟล์แบบประมาณ
9) ออกแบบและสร้างอัลกอริทึมการค้นพบโมทีฟต่อยอดจากเมทริกซ์โพรไฟล์แบบประมาณที่ไม่จำเป็นต้องกำหนดค่าพารามิเตอร์
10) เปรียบเทียบผลลัพธ์กับอัลกอริทึมการค้นพบโมทีฟอื่น ๆ
11) วิเคราะห์และสรุปผลการทดลอง
12) สรุปผล เรียบเรียง และจัดทำวิทยานิพนธ์

1.6 ผลงานวิจัยที่ได้ตีพิมพ์

K. Pariwatthanasak, C. A. Ratanamahatana, “Time series motif discovery using approximated matrix profile”, Third International Congress on Information and Communication Technology (ICICT), Brunel University, London, 27-28 February 2018.

บทที่ 2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 ข้อมูลอนุกรมเวลา (Time Series Data)

ข้อมูลอนุกรมเวลา คือลำดับของข้อมูลที่ถูกบันทึกไว้ตามจุดเวลาต่าง ๆ กล่าวคือข้อมูลอนุกรมเวลามีการบันทึกแต่ละจุดข้อมูลด้วยระยะห่างของเวลาที่เท่ากัน เช่น วินาที ชั่วโมง ปี เป็นต้น ดังนั้นข้อมูลอนุกรมเวลาเป็นลำดับของข้อมูลที่ไม่ต่อเนื่อง (Discrete-time Data) เนื่องจากข้อมูลต่าง ๆ ในชีวิตประจำวันมีการเปลี่ยนแปลงไปตามกาลเวลา ทำให้สามารถพบข้อมูลอนุกรมเวลาได้ทั่วไป ตัวอย่างเช่น ข้อมูลคลื่นไฟฟ้าหัวใจ (Electrocardiogram หรือ ECG) [1323] ข้อมูลเสียง (Audio Data) [11] ข้อมูลราคาหุ้น (Stock Price) [24] ข้อมูลเกี่ยวกับแผ่นดินไหว (Seismic Data) [10] เป็นต้น ภาพที่ 2.1 คือภาพแสดงตัวอย่างของข้อมูลราคาหุ้นบริษัท Google สังเกตได้ว่าราคาของหุ้นมีการเปลี่ยนแปลงตลอดเวลา

ภาพที่ 2.1 ภาพแสดงข้อมูลราคาหุ้นบริษัท Google ตั้งแต่ปี 2010 ถึงปี 2018

นอกจากนั้นเราอาจสนใจเฉพาะบางส่วนในข้อมูลอนุกรมเวลา ซึ่งสามารถดึงออกมาจากข้อมูลอนุกรมเวลาได้ โดยจะเรียกข้อมูลบางส่วนนี้ว่า ลำดับย่อย (Subsequence) ภาพที่ 2.2 แสดงตัวอย่างของลำดับย่อยของข้อมูลราคาหุ้นบริษัท Google เฉพาะในปี 2017

ภาพที่ 2.2 ภาพแสดงตัวอย่างของลำดับย่อยของข้อมูลราคาหุ้นบริษัท Google เฉพาะในปี 2017 (เส้นประ)

คำนิยามของข้อมูลอนุกรมเวลาและลำดับย่อยที่ใช้ในวิทยานิพนธ์นี้แสดงดังต่อไปนี้

อนุกรมเวลา (Time Series)

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

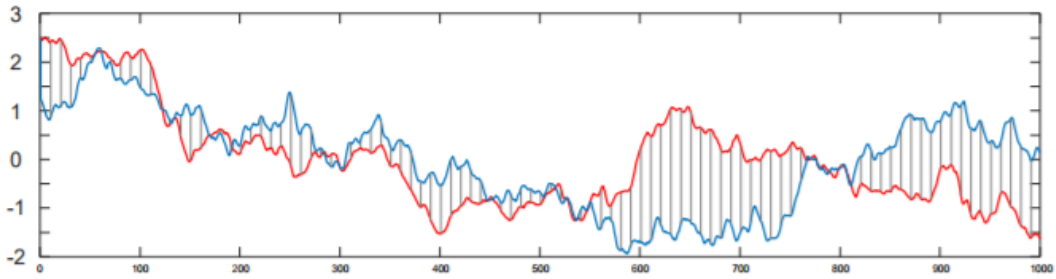
ลำดับย่อย (Subsequence)

ลำดับย่อย (Subsequence) $T_{i,m}$ ของอนุกรมเวลา T คือ สับเซตต่อเนื่อง (Continuous Subset) ของค่าใน T ที่เริ่มต้นจากตำแหน่ง i และมีความยาว m

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

2.1.2 ระยะห่างยุคลิด (Euclidean Distance)

ระยะห่างยุคลิด (Euclidean Distance) คือวิธีการวัดระยะห่างระหว่างจุดข้อมูล 2 จุด (ดังแสดงในภาพที่ 2.3)



ภาพที่ 2.3 การวัดระยะห่างยุคลิดระหว่าง 2 อนุกรมเวลา (ที่มา : [9])

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

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

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 จะได้ว่าสัมประสิทธิ์สหสัมพันธ์ของ 2 อนุกรมเวลานี้คำนวณได้จากสมการ

$$corr(T, Q) = \frac{\sum_{i=1}^{n} t_i q_i - n\mu_T \mu_Q}{n\sigma_T \sigma_Q}$$

โดยที่ $\mu_T = \frac{\sum_{i=1}^{n} t_i}{n}$ และ $\sigma_T^2 = \frac{\sum_{i=1}^{n} t_i q_i}{n} - \mu_T^2$

2.1.4 การทำให้เป็นบรรทัดฐาน (Normalization)

การวัดระยะห่างระหว่างข้อมูลอนุกรมเวลานั้นจำเป็นต้องมีการทำให้เป็นบรรทัดฐานก่อน เพราะว่าข้อมูลอนุกรมเวลานั้นอาจมีมาตราส่วน (Scale) ที่แตกต่างกัน ทำให้ผลลัพธ์ที่ได้มีความผิดพลาดตามไปด้วย ซึ่งวิธีการทำให้เป็นบรรทัดฐานคือการปรับมาตราส่วนและแอมพลิจูด (Amplitude) ของข้อมูลให้อยู่ในระดับเดียวกัน โดยในวิทยานิพนธ์นี้ใช้การทำให้เป็นบรรทัดฐานแบบซี (Z-Normalization)

การทำให้เป็นบรรทัดฐานแบบซี (Z-Normalization)

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

ระยะห่างยุคลิดที่รองรับการทำให้เป็นบรรทัดฐานแบบซี (Z-Normalized Euclidean Distance)

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

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

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

หลังจากนี้จะเรียกว่า ระยะห่างยุคลิดแบบซี (Z-Normalized Euclidean Distance)

2.1.5 โมทีฟ (Motif)

จากที่กล่าวไว้ในบทที่ 1 ว่าคำนิยามของโมทีฟมีได้หลายแบบ ดังนั้นคำนิยามของโมทีฟที่จะใช้ในวิทยานิพนธ์นี้คือ

โมทีฟสำหรับอนุกรมเวลา (Time Series Motif) คือคู่ของลำดับย่อยที่เหมือนกันมากที่สุด กล่าวคือ $T_{a,m}$ และ $T_{b,m}$ เป็นโมทีฟก็ต่อเมื่อ $dist(T_{a,m}, T_{b,m}) \leq dist(T_{i,m}, T_{j,m})$ สำหรับทุก $i, j \in [1,2, \dots, n - m + 1]$ โดยที่ $a \neq b$ และ $i \neq j$ และ $dist$ คือฟังก์ชันที่คำนวณระยะห่างยุคลิดที่รองรับการทำให้เป็นบรรทัดฐานแบบซี (Z-Normalized Euclidean Distance) ระหว่างลำดับย่อยนำเข้า

2.1.6 ดิสคอร์ด (Discord)

ดิสคอร์ดสำหรับอนุกรมเวลา (Time Series Discord) คือลำดับย่อยที่แตกต่างกันจากลำดับย่อยอื่นมากที่สุด กล่าวคือ $T_{a,m}$ เป็นดิสคอร์ดก็ต่อเมื่อ $dist(T_{a,m}, T_{b,m}) \geq dist(T_{i,m}, T_{j,m})$ สำหรับทุก $b, i, j \in [1,2, \dots, n - m + 1]$ โดยที่ $a \neq b$ และ $i \neq j$ และ $dist$ คือฟังก์ชันที่คำนวณระยะห่างยุคลิดที่รองรับการทำให้เป็นบรรทัดฐานแบบซี (Z-Normalized Euclidean Distance) ระหว่างลำดับย่อยนำเข้า

2.1.7 ปัญหาวันเกิด (Birthday Paradox Problem) [25]

ในปัญหาด้านทฤษฎีความน่าจะเป็น หากมีคน N คน เมื่อ N เป็นจำนวนนับหรือจำนวนเต็มบวกจะมีโอกาสมากน้อยเพียงใดที่จะมีคนเกิดวันเดียวกันบ้าง กล่าวคือจะมีความน่าจะเป็นเท่าไรที่จะมี 2 คนใด ๆ เกิดในวันที่และเดือนเดียวกัน ปัญหานี้จะอาศัยหลักการ 2 หลักการดังต่อไปนี้

หลักการที่ 1

กำหนดให้โอกาสที่จะเกิดเหตุการณ์ E คือ $P(E)$ และโอกาสที่จะไม่เกิดเหตุการณ์ E คือ $P(E^c)$ จะได้ว่า $P(E) = 1 - P(E^c)$

หลักการที่ 2

โอกาสที่จะเกิดเหตุการณ์ที่ 1 และ 2 ซึ่งเป็นอิสระต่อกัน เท่ากับผลคูณของโอกาสที่จะเกิดเหตุการณ์ที่ 1 และโอกาสที่จะเกิดเหตุการณ์ที่ 2

กำหนดให้ใน 1 ปีมี 365 วันเพื่อความสะดวกในการคำนวณ โดยพิจารณาโอกาสที่คนที่ i ใด ๆ จะมีวันเกิดไม่ตรงกับคนก่อนหน้า กล่าวคือคนที่ $1, 2, \dots, i-1$ เริ่มจากคนที่ 1 ไปเรื่อย ๆ จะพบว่า

คนที่ 1 มีโอกาสที่จะมีวันเกิดไม่ตรงกับคนก่อนหน้าคือ $\frac{365}{365}$ เพราะไม่มีคนใดอยู่ก่อนหน้าเลย

คนที่ 2 มีโอกาสที่จะมีวันเกิดไม่ตรงกับคนก่อนหน้าคือ $\frac{364}{365}$ เพราะต้องไม่ตรงกับคนที่ 1

คนที่ 3 มีโอกาสที่จะมีวันเกิดไม่ตรงกับคนก่อนหน้าคือ $\frac{363}{365}$ เพราะต้องไม่ตรงกับคนที่ 1 และ 2

พิจารณาจนถึงคนที่ N พบว่า คนที่ N มีโอกาสที่จะมีวันเกิดไม่ตรงกับคนก่อนหน้าคือ $\frac{365-N+1}{365}$

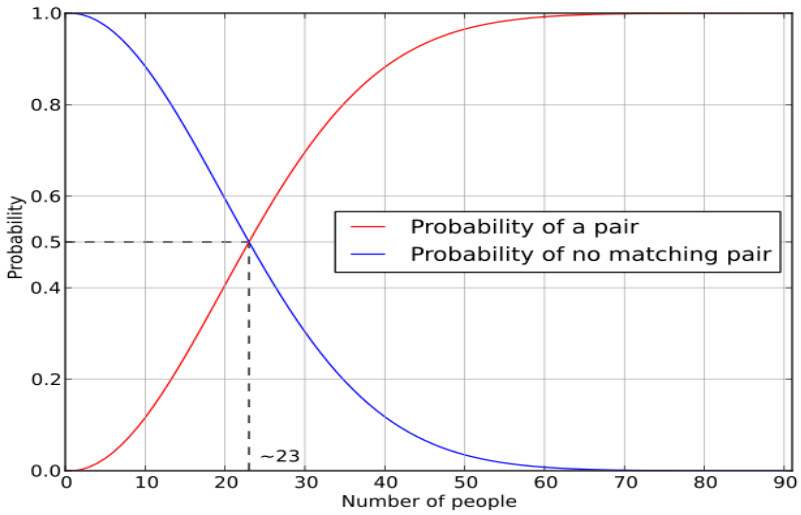
เพราะฉะนั้นโอกาสที่จะไม่มีใครเกิดตรงกันเลยคือ

$$\frac{365}{365} \cdot \frac{364}{365} \cdot \frac{363}{365} \cdot \dots \cdot \frac{365-N+1}{365} = \frac{365!}{365^N (365-N)!}$$

ซึ่งโอกาสที่จะมีคนเกิดตรงกันคือ

$$1 - \frac{365!}{365^N (365-N)!}$$

ภาพที่ 2.4 แสดงกราฟระหว่างจำนวนคนและโอกาสที่จะมีคนเกิดตรงกัน สังเกตได้ว่าเมื่อมีคน 60 คน หรือ $N = 60$ ก็มีโอกาสเกือบ 100% แล้ว



ภาพที่ 2.4 กราฟระหว่างจำนวนคนและโอกาสที่จะมีคนเกิดตรงกัน (สีแดง) และกราฟระหว่างจำนวนคนและโอกาสที่จะมีคนเกิดไม่ตรงกัน (สีน้ำเงิน)
(ที่มา : https://en.wikipedia.org/wiki/Birthday_problem)

2.2 งานวิจัยที่เกี่ยวข้อง

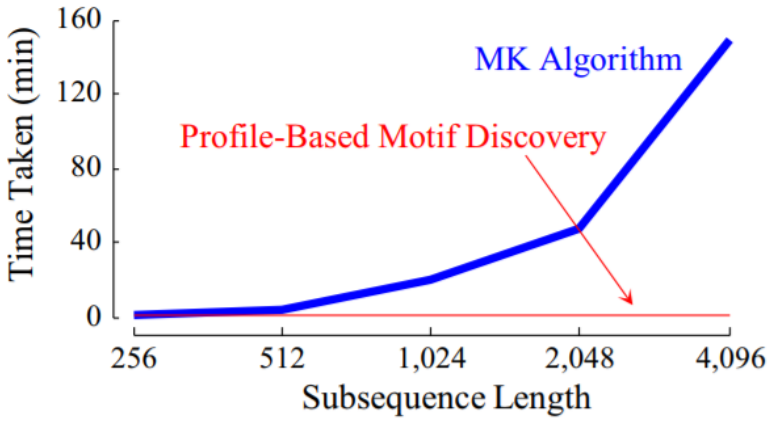
ในส่วนนี้จะแบ่งเป็น 4 ส่วนย่อย คือ งานวิจัยที่เกี่ยวกับการค้นพบโมทีฟ งานวิจัยที่เกี่ยวกับการค้นพบดิสคอร์ด วิธีการคำนวณหาเมทริกซ์โพรไฟล์ [14] และอัลกอริทึมการค้นพบโมทีฟที่มีความยาวเหมาะสม [21] ซึ่งเป็นวิธีหลักที่งานวิทยานิพนธ์นี้อ้างอิง

2.2.1 การค้นพบโมทีฟ (Motif Discovery)

การค้นพบโมทีฟ (Motif Discovery) ถูกนำไปประยุกต์ใช้ในหลาย ๆ ด้านดังที่กล่าวไว้ในบทที่ 1 ปัญหาทั่วไปของการค้นพบโมทีฟ เช่น การจัดการกับกระแสข้อมูล (Data Stream Handling) [26] การค้นพบโมทีฟในความยาวที่แตกต่างกัน (Detection of Motifs with Variable Lengths) [20] เป็นต้น อัลกอริทึมสำหรับการค้นพบโมทีฟ (Motif Discovery Algorithm) จะแตกต่างกันไปขึ้นอยู่กับปัญหาและขอบเขต (Domain) ที่กำลังศึกษาอยู่ เช่น อัลกอริทึมสำหรับโมทีฟแบบแม่นยำ (Exact Motif) [8] อัลกอริทึมสำหรับโมทีฟแบบประมาณ (Approximate Motif) [27] อัลกอริทึมสำหรับหาโมทีฟแบบความยาวแตกต่างกัน (Variable Length Motif) [20] และอัลกอริทึมสำหรับโมทีฟแบบความยาวเท่ากัน (Fixed Length Motif) [14] เป็นต้น

ในปี 2009 งานวิจัย [8] ได้อ้างว่า หากสามารถหาคู่ของโมทีฟที่เหมือนกันมากที่สุดแล้ว (ซึ่งเป็นไปตามนิยามที่จะศึกษาในงานวิจัยนี้) จะสามารถหาโมทีฟในตามนิยามอื่น ๆ ได้ เพียงแค่เพิ่มการคำนวณเล็กน้อยเท่านั้น นอกจากนั้นยังได้นำเสนออัลกอริทึมสำหรับการค้นพบโมทีฟ ชื่อว่า เอ็มเค

(MK Algorithm) ผลลัพธ์จากวิธีนี้เป็นโมทีฟแบบแม่นยำ (Exact Motif) ซึ่งเป็นวิธีที่เร็วและแพร่หลายที่สุด จนกระทั่งปี 2016 งานวิจัย [14] ได้อ้างว่า อัลกอริทึมเอ็มเค ใช้เวลาในการคำนวณหาโมทีฟขึ้นอยู่กับความยาวของลำดับย่อย ในขณะที่อัลกอริทึมพีบีเอ็มดี (Profile-Based Motif Discovery) [14] ที่ได้จากการคำนวณหาเมทริกซ์โพรไฟล์ใช้เวลาในการคำนวณหาโมทีฟไม่ขึ้นอยู่กับความยาวของลำดับย่อย (ดังแสดงในภาพที่ 2.5) และยังได้ผลลัพธ์เป็นโมทีฟแบบแม่นยำอีกด้วย



ภาพที่ 2.5 เวลาที่ใช้ในการหาโมทีฟของอนุกรมเวลาความยาว 65,536 จุดข้อมูล เปรียบเทียบระหว่างอัลกอริทึม MK และ PBMD (ที่มา : [14])

อย่างไรก็ตามอัลกอริทึมดังกล่าวจำเป็นต้องกำหนดพารามิเตอร์ความยาวของลำดับย่อยและเป็นความยาวของโมทีฟด้วย ซึ่งเป็นเรื่องยากที่จะทราบค่าที่เหมาะสม ในบางครั้งจำเป็นต้องใช้ความรู้หรือผู้เชี่ยวชาญเฉพาะด้านในการกำหนด ดังนั้นในปี 2013 งานวิจัย [21] ได้นำเสนออัลกอริทึมการค้นพบโมทีฟที่ความยาวเหมาะสม (Proper Length Motif Discovery) ซึ่งช่วยแก้ปัญหาในการกำหนดความยาวของลำดับย่อยได้ โดยงานวิจัยนี้ใช้เทคนิคการบีบอัดข้อมูล (Compression) ตามหลักการ MDL (Minimum Description Length) เพื่อวัดความคล้ายและจำนวนในการเกิดของโมทีฟด้วยค่าประหยัดบิต (Bitsave) โมทีฟที่มีค่าประหยัดบิตมากที่สุดคือโมทีฟที่มีความสามารถในการบีบอัดข้อมูลได้ดีที่สุด ซึ่งเป็นกลุ่มของลำดับย่อยที่มีรูปร่างคล้ายกันและเกิดขึ้นบ่อยที่สุด

2.2.2 การค้นพบดิสคอร์ด (Discord Discovery)

การค้นพบดิสคอร์ด (Discord discovery) คือการค้นหาลำดับย่อยที่มีระยะห่างจากลำดับย่อยอื่น ๆ ในข้อมูลอนุกรมเวลามากที่สุด ซึ่งเรียกลำดับย่อยนี้ว่าดิสคอร์ด (Discord) กล่าวคือดิสคอร์ดนั้นเป็นลำดับย่อยที่มีรูปร่างแตกต่างมากที่สุดหรือเป็นรูปร่างที่เกิดขึ้นน้อยที่สุด โดยทั่วไปการค้นพบมักจะถูกนำไปใช้ในปัญหาการตรวจจับสิ่งผิดปกติ (Anomaly Detection)

อัลกอริทึมสำหรับการค้นพบดิสคอร์ดได้ถูกคิดค้นและพัฒนามาอย่างต่อเนื่อง ตัวอย่างอัลกอริทึมที่ง่ายที่สุดคือ อัลกอริทึมการค้นพบดิสคอร์ดแบบบรูทฟอร์ซ (Brute Force Discord

Discovery, BFDD) [28] วิธีนี้เป็นการวัดระยะห่างของแต่ละคู่ลำดับย่อยด้วยระยะห่างยุคลิดและมีการทำให้เป็นบรรทัดฐานแบบซีเพื่อทำการปรับข้อมูลก่อน โดยดิสคอร์ดผลลัพธ์จะเป็นลำดับย่อยที่มีระยะห่างมากที่สุด หลังจากนั้นในปี 2005 งานวิจัย [29] ได้นำเสนออัลกอริทึมฮอทแซคซ์ (HOT SAX) เป็นวิธีหาดิสคอร์ดที่ใช้ระยะห่างยุคลิดเช่นเดียวกัน แต่มีการแปลงข้อมูลให้เป็นตัวอักษรด้วยวิธีการประมาณแบบรวมกลุ่มตัวอักษร (Symbolic Aggregate approXimation, SAX) ซึ่งเป็นการลดมิติของข้อมูล ในปี 2016 [30] ได้นำเสนออัลกอริทึมการค้นพบดิสคอร์ดแบบคู่ขนาน (Parallel Discord Discovery) ซึ่งแบ่งปัญหาหาการค้นพบดิสคอร์ดออกเป็นปัญหาย่อย แล้วแก้แต่ละปัญหาย่อยคู่ขนานกันไป

บางอัลกอริทึมจะเป็นอัลกอริทึมที่น่าสนใจเฉพาะกรณีที่ดีที่สุด (Best Case) แต่ในกรณีที่แย่ที่สุดอัลกอริทึมเหล่านี้ก็ใช้เวลาคำนวณเหมือนกับอัลกอริทึมแบบบรูทฟอร์ซ (Brute Force Algorithm) งานวิจัย [14] เป็นวิธีการค้นพบดิสคอร์ดโดยการค้นหาค่าที่มากที่สุดในเมทริกซ์โพรไฟล์ ซึ่งเป็นเวกเตอร์ที่เก็บระยะห่างแบบยุคลิดและรองรับการทำให้เป็นบรรทัดฐานแบบซี เพราะฉะนั้นถ้าคำนวณเมทริกโพรไฟล์ในปัญหาการค้นพบโมทีฟมาแล้ว ก็จะได้ดิสคอร์ดโดยการค้นหาค่าที่มากที่สุดอีกเพียงเล็กน้อยเท่านั้น อีกทั้งยังสามารถหาได้ทั้งแบบทุกเวลา (Anytime Algorithm) และแบบต่อยอดเพิ่มขึ้น (Incremental Algorithm) อีกด้วย

2.2.3 เมทริกซ์โพรไฟล์ (Matrix Profile)

จากที่กล่าวไว้ในหัวข้อ 2.2.1 และ 2.2.2 การคำนวณเพียงแค่เมทริกซ์โพรไฟล์ จะสามารถรองรับได้ทั้งการค้นพบโมทีฟและดิสคอร์ด ดังนั้นในหัวข้อนี้ จะอธิบายถึงการคำนวณเมทริกซ์โพรไฟล์ [14] ซึ่งเป็นหนึ่งในวิธีหลักที่งานวิจัยนี้อ้างอิง

คำนิยามและสัญลักษณ์ (Definition and Notation)

1) โพรไฟล์ระยะทาง (Distance Profile)

   โพรไฟล์ระยะทาง D คือเวกเตอร์ของค่าระยะห่างยุคลิดระหว่างลำดับย่อยที่กำหนด (Query) และแต่ละลำดับย่อยในเซตของลำดับย่อยทุกตัว (All-Subsequences Set)

2) เซตของลำดับย่อยทุกตัว (All-Subsequences Set)

   เซตของลำดับย่อยทุกตัว A สำหรับอนุกรมเวลา T คือเซตอันดับ (Ordered Set) ของลำดับย่อยที่มีความยาว m ทั้งหมดที่เป็นไปได้

   $$A = \{T_{1,m}, T_{2,m}, \dots, T_{n-m+1,m}\}$$ โดยกำหนด $A[i] = T_{i,m}$

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

กำหนดเซตของลำดับย่อยทุกตัว 2 เซต A, B และลำดับย่อย 2 ลำดับ $A[i], B[j]$ ฟังก์ชัน 1NN-join $\theta_{1nn}(A[i], B[j])$ คือ ฟังก์ชันที่คืนเฉพาะค่า จริง (True) เมื่อ $B[j]$ เป็นเพื่อนบ้านใกล้ที่สุด (Nearest Neighbor) ของ $A[i]$

4) เซตคู่ของลำดับย่อยที่คล้ายกัน (Similarity Join Set)

กำหนดเซตของลำดับย่อยทุกตัว 2 เซต A, B เซตคู่ของลำดับย่อยที่คล้ายกัน J_{AB} คือ เซตที่มีสมาชิกเป็นคู่ของทุก ๆ ลำดับย่อยใน A และเพื่อนบ้านใกล้ที่สุดใน B

$$J_{AB} = \{(A[i], B[j]) | \theta_{1nn}(A[i], B[j])\}$$

5) เมทริกซ์โพรไฟล์ (Matrix Profile)

เมทริกซ์โพรไฟล์ P_{AB} คือ เวกเตอร์ที่เก็บระยะห่างยุคลิดแบบซี (Z-Normalized Euclidean Distance) ระหว่างแต่ละคู่ลำดับย่อยใน J_{AB} โดยที่ $P_{AB}[i]$ เก็บค่าระยะห่างระหว่าง $A[i]$ และเพื่อนบ้านใกล้ที่สุดใน B

6) เมทริกซ์โพรไฟล์อินเด็กซ์ (Matrix Profile Index)

เมทริกซ์โพรไฟล์อินเด็กซ์ I_{AB} สำหรับ J_{AB} คือ เวกเตอร์ที่เก็บค่าจำนวนเต็ม โดยที่

$$I_{AB}[i] = j \text{ ถ้า } (A[i], B[j]) \in J_{AB}$$

STAMP Algorithm

อัลกอริทึมในการคำนวณเมทริกซ์โพรไฟล์ที่จะใช้ในวิทยานิพนธ์นี้ คือ อัลกอริทึมแสตมป์ (STAMP Algorithm) ซึ่งเป็นอัลกอริทึมแบบทุกเวลา (Anytime Algorithm) กล่าวคือ อัลกอริทึมที่สามารถเริ่มคำนวณที่ลำดับย่อยตำแหน่งใดก่อนก็ได้ในข้อมูลอนุกรมเวลา

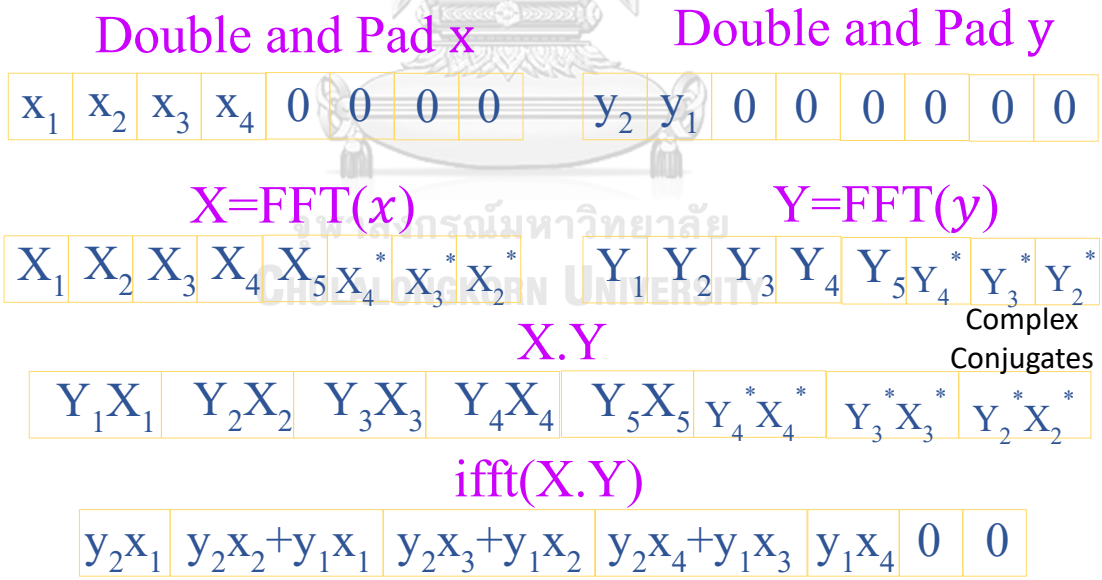
ในปี 2015 งานวิจัย [31] ได้นำเสนออัลกอริทึมสำหรับคำนวณความคล้ายกันของแต่ละคู่ลำดับย่อยโดยใช้ระยะห่างยุคลิด ชื่อว่า อัลกอริทึมแมส (**M**ueen's **A**lgorithm for **S**imilarity **S**earch) วิธีนี้ใช้เวลาเพียงแค่ $O(nlogn)$ โดยอาศัยการแปลงข้อมูลแบบ Fast Fourier Transform (FFT) และความสัมพันธ์ระหว่างระยะห่างยุคลิดและค่าสัมประสิทธิ์สหสัมพันธ์ (Correlation Coefficient) ผลลัพธ์ของอัลกอริทึมแมสคือโพรไฟล์ระยะทาง (Distance Profile) ที่เก็บระยะห่างยุคลิดระหว่างลำดับย่อยที่กำหนดและทุก ๆ ลำดับย่อยในอนุกรมเวลา

เนื่องจากอัลกอริทึมแมสมีการทำงานในแต่ละขั้นตอนที่มีรายละเอียดมาก จึงได้ทำการสรุปขั้นตอนต่าง ๆ ดังแสดงในตารางที่ 2.1 และ 2.2

ตารางที่ 2.1 การคำนวณ Sliding Dot Products (ที่มา : [31])

Procedure SlidingDotProduct(Q, T) Input : A query Q, and a user provided time series T Output : The dot product between Q and all subsequences in T	
1	$n \leftarrow Length(T), m \leftarrow Length(Q)$
2	$T_a \leftarrow Append\ T\ with\ n\ zeros$
3	$Q_r \leftarrow Reverse(Q)$
4	$Q_{ra} \leftarrow Append\ Q_r\ with\ 2n - m\ zeros$
5	$Q_{raf} \leftarrow FFT(Q_{ra}), T_{af} \leftarrow FFT(T_a)$
6	$QT \leftarrow InverseFFT(ElementwiseMulti(Q_{raf}, T_{af}))$
7	**return** QT

ในตารางที่ 2.1 เป็นการคำนวณผลคูณจุด (Dot Product) โดยอาศัยหลักการคอนโวลูชัน (Convolution) และซีโร่แพดดิง (Zero Padding) ซึ่งจะถูกนำไปใช้ต่อในอัลกอริทึมแมส ภาพที่ 2.6 แสดงหลักการคำนวณผลคูณจุด



ภาพที่ 2.6 ภาพแสดงขั้นตอนการคำนวณผลคูณจุดตามอัลกอริทึม SlidingDotProduct ในตารางที่ 2.1 (ที่มา : [31])

ตารางที่ 2.2 อัลกอริทึม MASS (ที่มา : [31])

Procedure MASS(Q, T) Input : A query Q, and a user provided time series T Output : A distance profile D of the query Q	
1	$QT \leftarrow SlidingDotProduct(Q, T)$
2	$\mu_Q, \sigma_Q, M_T, \Sigma_T \leftarrow ComputeMeanStd(Q, T)$
3	$D \leftarrow CalculateD(QT, \mu_Q, \sigma_Q, M_T, \Sigma_T)$
4	**return** D

อัลกอริทึมแมสเป็นการคำนวณระยะห่างระหว่างลำดับย่อยที่กำหนด Q และทุก ๆ ลำดับย่อยในอนุกรมเวลา T ซึ่งในบรรทัดที่ 3 ของตารางที่ 2.2 วิธีการคำนวณระยะห่างนั้นอาศัยความสัมพันธ์ระหว่างระยะห่างยุคลิดแบบซีและค่าสัมประสิทธิ์สหสัมพันธ์ โดยแต่ละค่าในเวกเตอร์ D คำนวณได้จากสมการ

$$D[i] = \sqrt{2m(1 - \frac{QT[i] - m\mu_Q M_T[i]}{m\sigma_Q \Sigma_T[i]})}$$

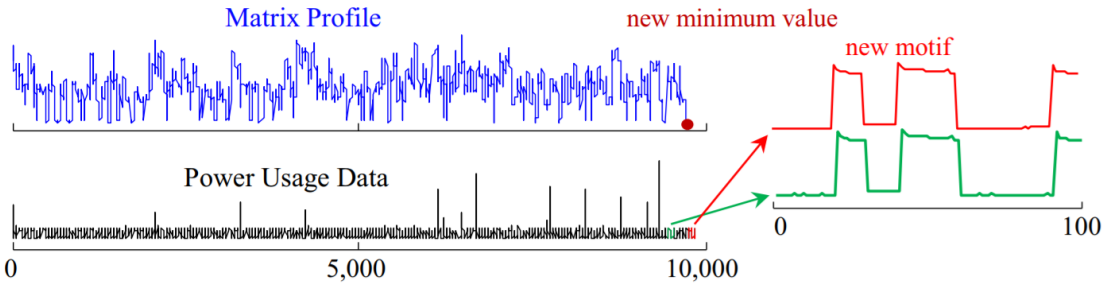
เมื่อ m คือความยาวของลำดับย่อย, μ_Q คือค่าเฉลี่ยเลขคณิตของ Q, $M_T[i]$ คือ ค่าเฉลี่ยเลขคณิตของ $T_{i,m}$, σ_Q คือ ค่าเบี่ยงเบนมาตรฐานของ Q และ $\Sigma_T[i]$ คือ ค่าเบี่ยงเบนมาตรฐานของ $T_{i,m}$

อัลกอริทึมแสตมป์ (**S**calable **T**ime **S**eries **A**nytime **M**atrix **P**rofile) แสดงในตารางที่ 2.3 ซึ่งผลลัพธ์ที่ได้จะเป็นเมทริกซ์โพรไฟล์ P_{AB} และเมทริกซ์โพรไฟล์อินเด็กซ์ I_{AB}

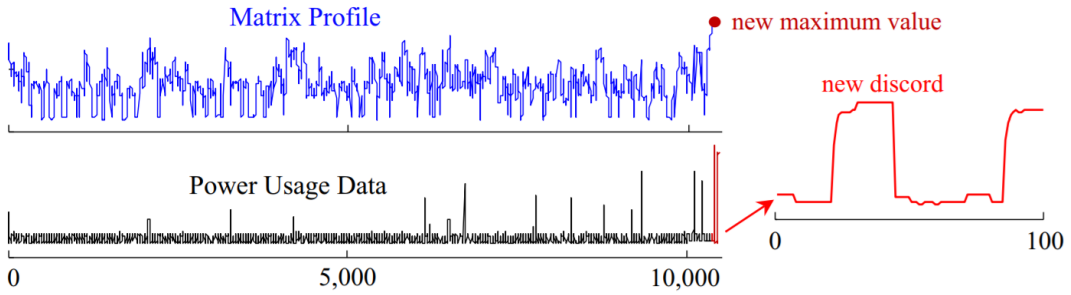
ตารางที่ 2.3 อัลกอริทึม STAMP (ที่มา : [8])

Procedure STAMP(T_a, T_b, m) Input : Two user provided time series, T_a, T_b, subsequence length m Output : A matrix profile P_{AB} and associated matrix profile index I_{AB}	
1	$n_B \leftarrow Length(T_B)$
2	$P_{AB} \leftarrow infs, I_{AB} \leftarrow zeros, idxes \leftarrow 1{:}\, n_B - m + 1$
3	**For each** idx **in** $idxes$ // random for anytime algorithm
4	$D \leftarrow MASS(B[idx], T_A)$
5	$P_{AB}, I_{AB} \leftarrow ElementWiseMin(P_{AB}, I_{AB}, D, idx)$
6	**end for**
7	**return** P_{AB}, I_{AB}

ภาพที่ 2.7 และ 2.8 แสดงตัวอย่างของเมทริกซ์โพรไฟล์สำหรับข้อมูลการใช้พลังงาน จะเห็นได้ว่าค่าที่มากที่สุดของเมทริกซ์โพรไฟล์ให้ผลเป็นดิสคอร์ดของอนุกรมเวลา และค่าที่น้อยที่สุดของเมทริกซ์โพรไฟล์ให้ผลเป็นโมทีฟของอนุกรมเวลา นอกจากนั้นยังสามารถหาคู่ผลลัพธ์ได้จากเมทริกซ์โพรไฟล์อินเด็กซ์



ภาพที่ 2.7 ตัวอย่างเมทริกซ์โพรไฟล์ในข้อมูลการใช้พลังงาน (Power Usage Data) โมทีฟผลลัพธ์สามารถหาได้จากค่าที่น้อยที่สุดในเมทริกซ์โพรไฟล์ (ที่มา : [14])



ภาพที่ 2.8 ตัวอย่างเมทริกซ์โพรไฟล์ในข้อมูลการใช้พลังงาน (Power Usage Data) ดิสคอร์ดผลลัพธ์สามารถหาได้จากค่าที่มากที่สุดในเมทริกซ์โพรไฟล์ (ที่มา : [14])

2.2.4 อัลกอริทึมการค้นพบโมทีฟที่ความยาวเหมาะสม (Proper Length Motif Discovery)

จากที่กล่าวในส่วนที่ 2.2.1 แล้วว่าอัลกอริทึมการค้นพบโมทีฟที่ความยาวเหมาะสม [21] สามารถแก้ปัญหาการกำหนดค่าความยาวของลำดับย่อยได้ ซึ่งเป็นอีกหนึ่งจุดอ่อนของเมทริกซ์โพรไฟล์ ดังนั้นในส่วนนี้จะอธิบายถึงขั้นตอนต่าง ๆ ในอัลกอริทึมนี้

คำนิยามต่าง ๆ สำหรับการคำนวณค่าประหยัดบิตมีดังนี้ [21,32]

1) การทำให้เป็นข้อมูลไม่ต่อเนื่อง (Discrete Normalization)

การทำให้เป็นข้อมูลไม่ต่อเนื่องคือการแปลงข้อมูลอนุกรมเวลา T ที่เป็นจำนวนจริง (Real-valued) ให้เป็นข้อมูลไม่ต่อเนื่อง ซึ่งคำนวณได้จากสมการ (2.1)

$$Discrete(T) = round\left(\left(\frac{T - min + 1}{max - min + 1}\right) * (2^a - 1)\right) + 1 \quad (2.1)$$

เมื่อกำหนดค่า $a = 64$ ซึ่งมีที่มาจากงานวิจัย [32]

2) ค่าความยาวในการเก็บข้อมูลอนุกรมเวลา (Description Length)

ค่าความยาวในการเก็บข้อมูลอนุกรมเวลา T คำนวณได้จากสมการ (2.2) และสมการ (2.3)

$$DL(T) = m * Entropy(T) \quad (2.2)$$

$$Entropy(T) = -\sum_i p_i \log_2\left(\frac{1}{p_i}\right) \quad (2.3)$$

เมื่อ m คือความยาวของข้อมูลอนุกรมเวลา T และ p_i คือ ค่าความน่าจะเป็นที่ค่า i จะปรากฏขึ้นในอนุกรมเวลา T

3) ค่าความยาวในการเก็บข้อมูลลำดับย่อย E ใด ๆ ที่เป็นสมาชิกของกลุ่ม $group$ เมื่อใช้สมมติฐาน H บีบอัดข้อมูล คำนวณได้จากสมการ (2.4)

$$DLC(group) = DL(H) + \sum_{E \in group} DL(E|H) \quad (2.4)$$

เมื่อสมมติฐาน H ที่ใช้ในงานวิจัยนี้คือ ลำดับย่อยเฉลี่ยระหว่างคู่ของลำดับย่อยเริ่มต้นที่พิจารณาในรอบนั้นและ $DL(E|H) = DL(E - H)$

4) ค่าประหยัดบิต (Bitsave) คือ ค่าผลต่างระหว่างค่าความยาวในการเก็บข้อมูล (Description Length) ของข้อมูลก่อนบีบอัดและค่าความยาวในการเก็บข้อมูลหลังบีบอัดโดยการใช้สมมติฐาน H

$$bitsave = DL(before) - DL(after) \quad (2.5)$$

5) ค่าประหยัดบิตของการสร้างกลุ่ม $group$ โดยเริ่มจากลำดับย่อยที่คล้ายกันมากที่สุด (A, B) คำนวณได้จากสมการ (2.6)

$$bitsave = DL(A) + DL(B) - DLC(group) \quad (2.6)$$

6) ค่าประหยัดบิตของการเพิ่มลำดับย่อยเพื่อนบ้าน A' เข้าในกลุ่ม $group$ ซึ่งจะทำให้ได้กลุ่มใหม่ $group'$ คำนวณได้จากสมการ (2.7)

$$bitsave = DL(A') + DLC(group) - DLC(group') \quad (2.7)$$

7) ค่าประหยัดบิตรวมของชุดของโมทีฟ คือ ผลต่างระหว่างผลรวมของค่าประหยัดบิตของโมทีฟแต่ละตัว และค่าความยาวในการเก็บข้อมูลของสมมติฐาน H ในสมการ (2.8)

$$totalbitsave = \sum_i \left(DL\left(T_{o(i),w}\right) - DL(T_{o(i),w}|H)\right) - DL(H) \quad (2.8)$$

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

ตารางที่ 2.4 อัลกอริทึมการค้นพบโมทีฟที่ความยาวเหมาะสม (ที่มา : [21])

Procedure ProperLengthMotifDiscovery(T) Input : Time series T Output : Motif	
1	**For** $l = 2$ **to** $T.length/2$
2	$\{A, B\} \leftarrow$**MotifCandidateDiscovery**(T, l)
3	**For each** k-th Motif Pair **in** $\{A, B\}$
4	$group \leftarrow CreateGroup(A, B)$
5	**If** $group.bitsave < 0$ **then break**
6	$group \leftarrow$ **AddAllNeighbor**$(group, T, l)$
7	$Motif \leftarrow$**UpdateAnswer**$(Motif, group)$
8	**end for**
9	**If** k=1 and $group.bitsave < 0$ **then break**
10	**end for**
11	**return** $Motif$

อัลกอริทึมนี้เป็นการค้นหาโมทีฟที่มีความยาวตั้งแต่ 2 จนถึงความยาวครึ่งหนึ่งของความยาวอนุกรมเวลา โดยให้ผลลัพธ์คือโมทีฟที่ความยาวเหมาะสม สังเกตได้ว่าข้อมูลนำเข้าของอัลกอริทึมนี้มีเฉพาะข้อมูลอนุกรมเวลาซึ่งไม่จำเป็นต้องกำหนดความยาวของลำดับย่อย

บรรทัดที่ 2 เป็นการหาลำดับย่อยที่ความยาว l ที่คล้ายกันมากที่สุด k อันดับ

บรรทัดที่ 4 เป็นการสร้างกลุ่มของลำดับย่อย โดยเริ่มจากคู่ลำดับย่อยที่คล้ายกันมากที่สุดที่ได้จากบรรทัดที่ 2 ก่อน

บรรทัดที่ 5 เป็นเงื่อนไขการหยุดค้นหาโมทีฟสำหรับค่าความยาวของลำดับย่อยในรอบนั้น ๆ เมื่อพบว่าค่าประหยัดบิตมีค่าติดลบ โดยค่าประหยัดบิตคำนวณได้จากสมการ (2.6)

บรรทัดที่ 6 ในกรณีที่ค่าประหยัดบิตยังไม่ติดลบ ฟังก์ชัน AddAllNeighbor จะทำการค้นหาลำดับย่อยเพื่อนบ้าน (Neighbor) ที่คล้ายกับลำดับย่อยเฉลี่ยระหว่างคู่ของลำดับย่อยเริ่มต้นที่พิจารณาในรอบนั้น ๆ โดยมีเงื่อนไขคือจะต้องมีความยาว l และเมื่อเพิ่มเข้าไปในกลุ่มแล้วค่าประหยัดบิตยังคงไม่ติดลบ ซึ่งคำนวณได้จากสมการ (2.7)

บรรทัดที่ 7 คำนวณค่าประหยัดบิตรวมของกลุ่มจากสมการ (2.8) และรวมกลุ่มเข้าเป็นชุดโมทีฟล่าสุด ในกรณีที่ลำดับย่อยมีการซ้อนทับ (Overlap) จะทิ้งลำดับย่อยที่ให้ค่าประหยัดบิตต่ำกว่า

บรรทัดที่ 9 เป็นเงื่อนไขหยุดประมวลผลเมื่อค่าประหยัดบิตติดลบ

จากที่ได้กล่าวถึงงานวิจัย [14,21] ซึ่งเป็นงานวิจัยหลักที่วิทยานิพนธ์นี้อ้างอิง จะเห็นได้ว่าเมทริกซ์โพรไฟล์นั้นสามารถนำมาประยุกต์ใช้ได้ทั้งปัญหาการค้นพบโมทีฟและดิสคอร์ด แต่เนื่องจากปัญหาระยะเวลาในการคำนวณที่นาน รวมถึงการกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อยที่สามารถใช้ประโยชน์จากอัลกอริทึมการค้นพบโมทีฟที่ความยาวเหมาะสมได้ ดังนั้นจึงนำเสนอการปรับปรุงเมทริกซ์โพรไฟล์ โดยจะกล่าวรายละเอียดในบทถัดไป

บทที่ 3 อัลกอริทึมการค้นพบโมทีฟและดิสคอร์ดของอนุกรมเวลา โดยใช้เมทริกซ์โพรไฟล์แบบประมาณ

แนวคิดที่วิทยานิพนธ์นี้นำเสนอเพื่อปรับปรุงอัลกอริทึมสำหรับการค้นพบโมทีฟและดิสคอร์ดนั้นมี 2 ส่วนหลักคือ เมทริกซ์โพรไฟล์แบบประมาณเพื่อลดเวลาในการคำนวณเมทริกซ์โพรไฟล์โดยยังคงได้โมทีฟและดิสคอร์ดผลลัพธ์ที่ใกล้เคียงวิธีเดิม และอัลกอริทึมการค้นพบโมทีฟสำหรับความยาวที่เหมาะสมเพื่อแก้ปัญหาการกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อยในเมทริกซ์โพรไฟล์

3.1 เมทริกซ์โพรไฟล์แบบประมาณ (Approximated Matrix Profile)

จากที่ได้กล่าวไปในบทที่ 1 ว่าอาจไม่จำเป็นต้องคำนวณเมทริกซ์โพรไฟล์ทั้งหมด การคำนวณเฉพาะบางส่วนของเมทริกซ์โพรไฟล์ เพื่อให้ได้ค่าที่น้อยที่สุดและค่าที่มากที่สุดก็เพียงพอ ดังนั้นในส่วนนี้จะนำเสนอและอธิบายแนวคิดของสมมติฐานนี้ ซึ่งแบ่งเป็น 2 ส่วนย่อยคือ ลำดับของการคำนวณเมทริกซ์โพรไฟล์ (Order of Computation) และจำนวนลำดับย่อยที่ใช้ในการคำนวณเมทริกซ์โพรไฟล์ (Number of Iterations)

3.1.1 ลำดับของการคำนวณเมทริกซ์โพรไฟล์ (Order of Computation)

เนื่องจากอัลกอริทึมแสตมป์ (ตารางที่ 2.3) เป็นอัลกอริทึมแบบทุกวลา (Anytime Algorithm) ซึ่งลำดับในการเลือกลำดับย่อยมาคำนวณนั้นได้จากการสุ่ม ดังนั้นการปรับลำดับในการเลือกลำดับย่อยมาคำนวณน่าจะทำให้เข้าใกล้ผลลัพธ์ได้เร็วขึ้น แนวคิดที่นำเสนอคือ

1) ลำดับย่อยแรกที่ถูกเลือกมาคำนวณนั้นได้จากการสุ่ม
2) ลำดับย่อยถัดไปที่จะนำมาใช้คำนวณคือ ลำดับย่อยที่เป็นเพื่อนบ้านใกล้ที่สุดของลำดับย่อยแรกซึ่งได้มาโดยอัตโนมัติจากการคำนวณโพรไฟล์ระยะทาง (Distance Profile)
3) ในกรณีที่ลำดับย่อยเพื่อนบ้านใกล้สุดถูกนำมาคำนวณแล้ว จะใช้ลำดับย่อยที่เหลือที่ยังไม่ได้นำมาคำนวณจากการสุ่ม

3.1.2 จำนวนของลำดับย่อยที่ใช้ในการคำนวณเมทริกซ์โพรไฟล์หรือจำนวนการวนซ้ำ (Number of Iterations)

การคำนวณเมทริกซ์โพรไฟล์ในอัลกอริทึมแสตมป์จะมีการวนซ้ำทั้งหมด $n-m+1$ รอบ เมื่อ n คือความยาวของอนุกรมเวลาและ m คือความยาวของลำดับย่อย ซึ่งจำนวนการวนซ้ำนั้นหมายถึงจำนวนลำดับย่อยทั้งหมดในอนุกรมเวลา ในส่วนนี้จึงนำเสนอวิธีในการหาจำนวนของลำดับย่อย k โดยที่ $k < n-m+1$ ทำให้การคำนวณเมทริกซ์โพรไฟล์นั้นไม่จำเป็นต้องมีการวนซ้ำทั้งหมด $n-m+1$ รอบ แต่มีการวนซ้ำเพียงแค่ k รอบ

เนื่องจากการค้นพบโมทีฟตามนิยามที่ได้กล่าวไปแล้วนั้น เป็นการเลือกคู่ของลำดับย่อยที่มีระยะห่างน้อยที่สุดจากลำดับย่อยที่เป็นไปได้ทั้งหมด ดังนั้นปัญหานี้อาจมองในมุมของปัญหาวันเกิด (Birthday Paradox) กล่าวคือ

1) วันทั้งหมด 365 วันที่เป็นไปได้หมายถึง จำนวนระยะห่างระหว่างลำดับย่อยและลำดับย่อยเพื่อนบ้านทั้งหมดที่เป็นไปได้ ซึ่งมีค่าเท่ากับ $n - m + 1$
2) ความน่าจะเป็นที่คน 2 คนจะเกิดวันเดียวกันหมายถึง ความน่าจะเป็นที่ลำดับย่อย 2 ลำดับย่อยใด ๆ จะเป็นโมทีฟ
3) จำนวนลำดับย่อยอย่างน้อยที่จะนำมาคิดเพื่อที่จะได้ผลลัพธ์โมทีฟด้วยความน่าจะเป็น p ก็คือ จำนวนคนอย่างน้อยที่ต้องมีเพื่อที่จะยืนยันได้ว่ามีคนเกิดวันเดียวกันด้วยความน่าจะเป็น p

เพราะฉะนั้นงานวิจัยนี้ได้นำเสนออัลกอริทึมการหาจำนวนลำดับย่อย (FindK Algorithm) ซึ่งเป็นอัลกอริทึมสำหรับคำนวณหาจำนวนของลำดับย่อยหรือจำนวนการวนซ้ำ k ดังแสดงในตารางที่ 3.1 โดยมีข้อมูลนำเข้าคือ ข้อมูลอนุกรมเวลา T ที่มีความยาว n ความยาวของลำดับย่อย m และค่าความน่าจะเป็น p ในงานวิจัยนี้จะกำหนดค่า $p = 0.999$ และมีข้อมูลนำออกคือจำนวนการวนซ้ำ k

ตารางที่ 3.1 การหาจำนวนลำดับย่อยที่ใช้ในการคำนวณเมทริกซ์โพรไฟล์

Procedure FindK(n, m, p) Input : Time series length n, subsequence length m, user provided probability p Output : A number of subsequences k using in Matrix Profile calculation	
1	$k \leftarrow 1$
2	$prob \leftarrow 0$
3	**while** $prob < p$
4	$prob \leftarrow CalculateProb(k, n, m)$
5	$k++$
6	**end while**
7	**return** k

วิธีการคำนวณค่าความน่าจะเป็นในบรรทัดที่ 4 ในตารางที่ 3.4 คือ

$$CalculateProb(k, n, m) = 1 - e^{-k^2/(n-m)(n-m+1)}$$

ซึ่งมีการดัดแปลงมาจากการคำนวณค่าความน่าจะเป็นแบบประมาณของปัญหาวันเกิด [25]

3.1.3 อัลกอริทึมเอเอ็มพี (Approximated Matrix Profile)

จากแนวคิดทั้งสองส่วน (3.1.1 และ 3.1.2) ทำให้มีการปรับปรุงอัลกอริทึมแสตมป์ ให้เป็นอัลกอริทึมเอเอ็มพี (Approximated Matrix Profile) (ดังแสดงในตารางที่ 3.2)

ตารางที่ 3.2 อัลกอริทึม AMP

Procedure AMP(T, m, p) Input : Time series, T, subsequence length m, user provided probability p Output : A matrix profile P_A and associated matrix profile index I_A	
1	$n \leftarrow Length(T)$
2	$P_A \leftarrow infs, I_A \leftarrow zeros$
3	$idxes \leftarrow 1{:}\, n - m + 1$ // random for anytime algorithm
4	$idx \leftarrow idxes[1], idxes.remove(idxes[1])$
5	$k \leftarrow FindK(n, m, p)$ // $p = 0.999$
6	**For each** $(i = 1; i \leq k; i++)$
7	$D \leftarrow MASS(T_{idx,m}, T)$
8	$P_A, I_A \leftarrow ElementWiseMin(P_A, I_A, D, idx)$
9	$idx \leftarrow I_A[idx]$
10	If idx is not in $idxes$
11	$idx \leftarrow idxes[1]$
12	$idxes.remove(idxes[j]\ where\ idxes[j] == idx)$
13	**end for**
14	**return** P_A, I_A

อัลกอรีทึมนี้เป็นการคำนวณเมทริกซ์โพรไฟล์แบบประมาณ โดยให้ผลลัพธ์คือ เมทริกซ์โพรไฟล์แบบประมาณ P_A และเมทริกซ์โพรไฟล์อินเด็กซ์ I_A ข้อมูลนำเข้าของอัลกอริทึมนี้คือข้อมูลอนุกรมเวลา T ความยาวของลำดับย่อย m และค่าความน่าจะเป็น p โดยในงานวิจัยนี้กำหนดให้ $p = 0.999$

บรรทัดที่ 1-2 เป็นการกำหนดค่าเริ่มต้นของอัลกอริทึม โดยกำหนดเมทริกซ์โพรไฟล์แบบประมาณ P_A เป็นเวกเตอร์ที่มีความยาว $n - m + 1$ ซึ่งแต่ละค่าคือค่าอนันต์ (Infinity) และเมทริกซ์โพรไฟล์อินเด็กซ์ I_A เป็นเวกเตอร์ที่มีความยาว $n - m + 1$ ซึ่งแต่ละค่าคือศูนย์

บรรทัดที่ 3 เป็นการกำหนดตำแหน่งเริ่มต้นของลำดับย่อยที่จะนำมาใช้ในแต่ละรอบ และทำการสุ่มทุกค่าในเวกเตอร์ $idxes$

บรรทัดที่ 4 หลังจากทำการสุ่มในบรรทัดที่ 3 ลำดับย่อยแรกที่ใช้ในการคำนวณคือลำดับย่อยที่มีตำแหน่งเริ่มต้นเป็นค่าแรกในเวกเตอร์ $idxes$ โดยกำหนดให้เป็นตัวแปร idx หลังจากนั้นทำการลบค่าแรกในเวกเตอร์ $idxes$

บรรทัดที่ 5 อัลกอริทึม FindK คำนวณหาจำนวนการวนซ้ำ k ตามรายละเอียดในส่วนที่ 3.1.2 โดยกำหนดค่าความน่าจะเป็น $p = 0.999$

บรรทัดที่ 6-13 ทำการคำนวณเมทริกซ์โพรไฟล์แบบประมาณ ซึ่งมีการวนซ้ำทั้งหมด k รอบ โดยในแต่ละรอบจะใช้ลำดับย่อยที่กำหนด (Query) เป็นลำดับย่อยในอนุกรมเวลา T ความยาว m และมีตำแหน่งเริ่มต้นที่ idx

บรรทัดที่ 7 คำนวณโพรไฟล์ระยะทางจากอัลกอริทึมแมส เมื่อ $T_{idx,m}$ คือลำดับย่อยในอนุกรมเวลา T ความยาว m และมีตำแหน่งเริ่มต้นที่ idx โดยมีเงื่อนไขว่า $D[i] = inf$ เมื่อ $i \in [idx - \frac{m}{2}, idx + \frac{m}{2}]$ เพื่อป้องกันปัญหาการวัดระยะทางระหว่างลำดับย่อยที่กำหนดและลำดับย่อยที่มีการซ้อนทับกัน (Trivial Match) ซึ่งปัญหานี้ก่อให้เกิดการค้นพบโมทีฟที่ไม่มีความหมาย

บรรทัดที่ 8 ฟังก์ชัน ElementWiseMin ทำการเปรียบเทียบค่าที่น้อยกว่าเป็นคู่ ๆ ในแต่ละตำแหน่งระหว่างเวกเตอร์ P_A และเวกเตอร์ D กล่าวคือ $P_A[i] = \min(P_A[i], D[i])$ สำหรับ $i = 1$ ถึง $D.length$ หลังจากนั้นทำการปรับเวกเตอร์ I_A ด้วยค่า idx กล่าวคือ $I_A[i] = idx$ เมื่อ $D[i] \leq P_A[i]$ สำหรับทุก i ที่ทำการปรับในเวกเตอร์ P_A ก่อนหน้า

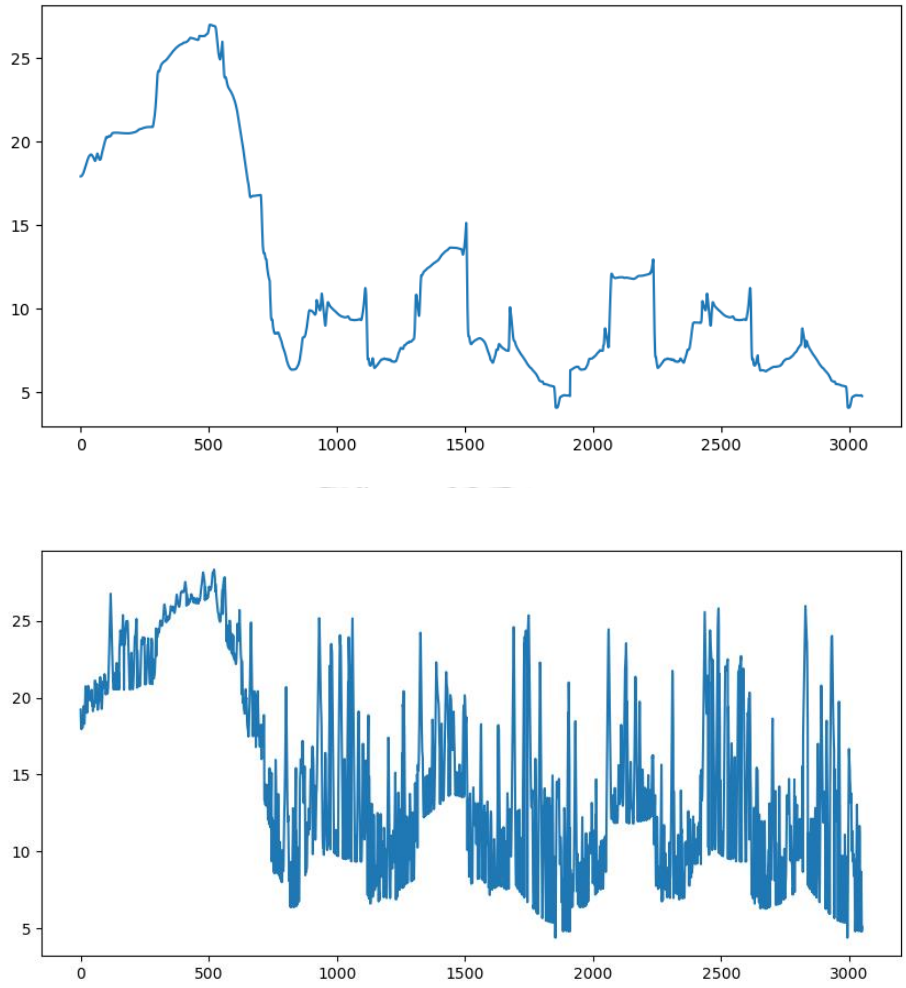
บรรทัดที่ 9 ทำการค้นหาตำแหน่งของลำดับย่อยเพื่อนบ้านของลำดับย่อยที่พิจารณาในรอบนั้น ๆ โดยหาได้จากเวกเตอร์ I_A เพื่อใช้เป็นลำดับย่อยในการคำนวณรอบถัดไปดังแนวคิดในส่วนที่ 3.1.1

บรรทัดที่ 10-11 เป็นเงื่อนไขที่ตรวจสอบการถูกใช้แล้วของตำแหน่งของลำดับย่อยเพื่อนบ้านที่ได้จากบรรทัดที่ 9 ในกรณีที่ถูกใช้ไปแล้วในรอบถัดไปจะใช้ตำแหน่งของลำดับย่อยเป็นค่าแรกในเวกเตอร์ $idxes$ และปรับค่า idx ใหม่สำหรับรอบถัดไป

บรรทัดที่ 12 ทำการลบค่า idx ในเวกเตอร์ $idxes$ เพราะจะถูกใช้ในรอบถัดไป

บรรทัดที่ 14 คืนค่าเมทริกซ์โพรไฟล์แบบประมาณ P_A และเมทริกซ์โพรไฟล์อินเด็กซ์ I_A

ภาพที่ 3.1 แสดงตัวอย่างเมทริกซ์โพรไฟล์แบบประมาณเปรียบเทียบกับเมทริกซ์โพรไฟล์ในงานวิจัย [14] ในข้อมูลคลื่นไฟฟ้าหัวใจ



ภาพที่ 3.1 ภาพตัวอย่างแสดงเมทริกซ์โพรไฟล์ (บน) และเมทริกซ์โพรไฟล์แบบประมาณ (ล่าง)

3.2 อัลกอริทึมสำหรับการค้นพบโมทีฟสำหรับความยาวที่เหมาะสมโดยใช้เมทริกซ์โพรไฟล์แบบประมาณ (Proper Length Motif Discovery Algorithm using Approximated Matrix Profile)

เนื่องจากเมทริกซ์โพรไฟล์แบบประมาณนั้นสามารถลดเวลาในการคำนวณได้อย่างมากเมื่อเทียบกับเมทริกซ์โพรไฟล์ และเมทริกซ์โพรไฟล์ยังจำเป็นต้องกำหนดพารามิเตอร์ความยาวของลำดับย่อยซึ่งอาจต้องใช้ความรู้หรือผู้เชี่ยวชาญเฉพาะด้าน อีกทั้งโมทีฟหรือดิสคอร์ดผลลัพธ์ที่มีประโยชน์อาจไม่ใช่ลำดับย่อยที่มีความยาวที่กำหนดไว้ในตอนแรกเสมอไป ดังนั้นในส่วนนี้จะเสนออัลกอริทึมสำหรับการค้นพบโมทีฟสำหรับความยาวที่เหมาะสมโดยใช้เมทริกซ์โพรไฟล์แบบประมาณหรือ

อัลกอริทึมพีแอลเอเอ็มพี (**P**roper **L**ength Motif Discovery Algorithm using **A**pproximated **M**atrix **P**rofile, PLAMP)

ในอัลกอริทึมนี้จะประยุกต์ใช้เมทริกซ์โพรไฟล์แบบประมาณและวิธีการค้นพบโมทีฟความยาวเหมาะสม [21] รวมเข้าด้วยกัน (ดังแสดงในตารางที่ 3.3) โดยคำจำกัดความของโมทีฟ คือ คู่ของลำดับย่อยที่เหมือนกันมากที่สุด

ตารางที่ 3.3 อัลกอริทึม PLAMP

Procedure PLAMP(T) Input : Time series T Output : Motif	
1	$totalbitsave = -inf, numbermotif = 0, bestlength = 0$
2	$Motif \leftarrow \emptyset$
3	**For** $l = T.length * (\frac{10}{100})$ **to** $T.length * (\frac{40}{100})$
4	$P_A, I_A \leftarrow$**AMP**(T, l, p) // $p = 0.999$
5	$(A, B) \leftarrow$**FindClosetPair**(P_A, I_A)
6	$group \leftarrow CreateGroup(A, B)$
7	**If** $group.bitsave < 0$ **then break**
8	**While** $group.bitsave > 0$
9	$group \leftarrow$ **AddNeighbor**$(group, P_A, I_A)$
10	**end while**
11	$cost \leftarrow group.totalbitsave$
12	**If** $group.length > numbermotif$
13	$numbermotif \leftarrow group.length, bestlength \leftarrow l$ $totalbitsave \leftarrow cost, Motif \leftarrow (A, B)$
14	**elseIf** $group.length = numbermotif$ **and** $cost > totalbitsave$
15	$numbermotif \leftarrow group.length, bestlength \leftarrow l$ $totalbitsave \leftarrow cost, Motif \leftarrow (A, B)$
16	**end for**
17	**return** $Motif, bestlength$

อัลกอรึทึมนี้เป็นการค้นหาโมทีฟที่มีความยาวตั้งแต่ 10% จนถึง 40% ของความยาวของอนุกรมเวลา โดยให้ผลลัพธ์คือคู่ของโมทีฟที่คล้ายกันมากที่สุดตามคำนิยามในงานวิจัยนี้และความยาว

ที่เหมาะสม สังเกตได้ว่าข้อมูลนำเข้าของอัลกอริทึมนี้มีเฉพาะข้อมูลอนุกรมเวลา T ไม่จำเป็นต้องกำหนดความยาวของลำดับย่อยซึ่งเป็นจุดด้อยของเมทริกซ์โพรไฟล์ที่ต้องการปรับปรุง

บรรทัดที่ 1-2 เป็นกำหนดตัวแปรเริ่มต้น โดยกำหนดให้ตัวแปร $totalbitsave$ เป็นค่าอนันต์ที่ติดลบ ซึ่งเป็นตัวแปรที่ใช้ในการเปรียบเทียบค่าประหยัดบิตสำหรับการเลือกความยาวของโมทีฟที่เหมาะสม ตัวแปร $numbermotif$ และ $bestlength$ เป็นศูนย์ และกำหนดให้ตัวแปร $motif$ ซึ่งเป็นตัวแปรที่เก็บโมทีฟผลลัพธ์มีค่าเริ่มเป็นเป็นเซตว่าง

บรรทัดที่ 3-16 ทำการค้นหาโมทีฟในแต่ละความยาว โดยมีความยาวตั้งแต่ 10% จนถึง 40% ของความยาวของอนุกรมเวลา

บรรทัดที่ 4 อัลกอริทึมเอเอ็มพีทำการคำนวณเมทริกซ์โพรไฟล์แบบประมาณของความยาวลำดับย่อยที่พิจารณาในรอบนั้น รวมถึงคืนค่าเมทริกซ์โพรไฟล์อินเด็กซ์

บรรทัดที่ 5 ทำการค้นหาคู่ของลำดับย่อยที่คล้ายกันมากที่สุดโดยเลือกจากตำแหน่งของเมทริกซ์โพรไฟล์แบบประมาณที่มีค่าระยะห่างน้อยที่สุด 2 อันดับ นอกจากนั้นยังมีเงื่อนไขในการค้นหาคือคู่ของลำดับย่อยนั้นต้องไม่มีการซ้อนทับกัน

บรรทัดที่ 6 เป็นการสร้างกลุ่มของลำดับย่อย โดยเริ่มจากคู่ลำดับย่อยที่ได้จากฟังก์ชันในบรรทัดที่ 5

บรรทัดที่ 7 เงื่อนไขการสิ้นสุดการทำงานในรอบนั้นเมื่อพบว่าค่าประหยัดบิตของกลุ่มเริ่มต้นมีค่าติดลบ โดยค่าประหยัดบิตคำนวณได้จากสมการ (2.6)

บรรทัดที่ 8-10 เป็นเงื่อนไขการเพิ่มลำดับย่อยเพื่อนบ้าน (Neighbor) สำหรับค่าความยาวของลำดับย่อยในรอบนั้น ๆ เมื่อพบว่าค่าประหยัดบิตยังมีค่าเป็นบวก โดยค่าประหยัดบิตของกลุ่มใหม่คำนวณได้จากสมการ (2.7)

บรรทัดที่ 9 เป็นการเพิ่มลำดับย่อยเพื่อนบ้านเข้าไปในกลุ่ม $group$ ทีละตัว แล้วตรวจสอบว่าค่าประหยัดบิตยังเป็นบวกอยู่หรือไม่ เนื่องจากค่าที่น้อยที่สุด 2 อันดับในเมทริกซ์โพรไฟล์แบบประมาณถูกใช้ในการสร้างกลุ่มเริ่มต้นแล้ว ดังนั้นในการเพิ่มลำดับย่อยเพื่อนบ้านจึงใช้ใช้ลำดับย่อยที่มีค่าน้อยที่สุดเป็นอันดับ 3, 4, 5, ... ในเมทริกซ์โพรไฟล์แบบประมาณ และยังคงมีเงื่อนไขว่าลำดับย่อยที่เพิ่มเข้าไปต้องไม่มีการซ้อนทับกัน จะเห็นได้ว่าการเพิ่มลำดับย่อยเพื่อนบ้านในส่วนนี้ไม่จำเป็นต้องคำนวณระยะห่างระหว่างคู่ลำดับย่อยใหม่ ซึ่งเป็นข้อดีของการคำนวณเมทริกซ์โพรไฟล์ไว้ก่อนหน้าแล้ว

บรรทัดที่ 11 เมื่อไม่สามารถเพิ่มลำดับย่อยเพื่อนบ้านได้แล้ว จึงทำการคำนวณค่าประหยัดบิตรวมของกลุ่ม $group$ ในสมการ (2.8) แลทำการเก็บไว้ในตัวแปร $cost$

บรรทัดที่ 12, 14 เป็นการตรวจสอบว่าค่าความยาวของลำดับย่อยในรอบการพิจารณานั้นเหมาะสมหรือไม่ โดยเปรียบเทียบจากจำนวนสมาชิกในกลุ่ม $group$ ถ้ามีมากกว่าหมายความว่าโม

ทีฟที่ความยาวในรอบนั้นสามารถเป็นตัวแทนได้ดีกว่า ในกรณีที่จำนวนสมาชิกเท่ากันจะเลือกโมทีฟที่มีค่าประหยัดบิตรวมสูงที่สุด

บรรทัดที่ 13, 15 ทำการปรับค่าตัวแปรต่าง ๆ เมื่อเงื่อนไขการตรวจสอบในบรรทัดที่ 12, 14 เป็นจริง

อัลกอริทึมที่นำเสนอ 2 อัลกอริทึมคือ อัลกอริทึมเอเอ็มพี (AMP) สำหรับคำนวณเมทริกซ์โพรไฟล์แบบประมาณซึ่งอาศัยหลักการ 2 หลักการคือ การจัดลำดับของลำดับย่อยที่นำมาคำนวณ และการลดจำนวนของลำดับย่อยที่ใช้ในการคำนวณหรือจำนวนการวนซ้ำ และอัลกอริทึมพีแอลเอเอ็มพี (PLAMP) สำหรับการค้นพบโมทีฟที่ความยาวเหมาะสมโดยใช้เมทริกซ์โพรไฟล์แบบประมาณ ซึ่งทั้งสองอัลกอริทึมนี้จะถูกนำไปใช้ในการทดลองเพื่อเปรียบเทียบความเร็วและความแม่นยำในบทถัดไป

บทที่ 4 การทดลองและวิเคราะห์ผลการทดลอง

ในบทนี้จะกล่าวถึงการทดลองเพื่อประเมินคุณภาพของงานที่นำเสนอโดยแบ่งเป็น 2 ส่วนหลักคือ การทดลองอัลกอริทึมเอเอ็มพี (AMP) สำหรับการคำนวณเมทริกซ์โพรไฟล์แบบประมาณเทียบกับอัลกอริทึมแสตมป์ (STAMP) สำหรับการคำนวณเมทริกซ์โพรไฟล์ [14] อีกส่วนคือการทดลองอัลกอริทึมการค้นพบโมทีฟสำหรับความยาวที่เหมาะสมโดยใช้เมทริกซ์โพรไฟล์แบบประมาณ (PLAMP)

4.1 การทดลองสำหรับอัลกอริทึมเอเอ็มพี

ในส่วนนี้จะใช้อัลกอริทึมเอเอ็มพีในปัญหาการค้นพบโมทีฟและดิสคอร์ดเปรียบเทียบกับอัลกอริทึมแสตมป์ โดยจะทำการทดลองกับทั้งข้อมูลสังเคราะห์ (Synthetic Data) และข้อมูลจริง (Real World Data)

4.1.1 ข้อมูลสังเคราะห์ (Synthetic data)

ข้อมูลสังเคราะห์ที่ใช้ในงานวิจัยนี้คือ ข้อมูลการเดินแบบสุ่ม (Random Walk) ทั้งหมด 5 ชุดข้อมูลที่มีขนาดแตกต่างกัน ดังแสดงในตารางที่ 4.1 โดยกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อยเท่ากับ 1% ของความยาวของข้อมูลอนุกรมเวลา

ตารางที่ 4.1 ตารางแสดงรายละเอียดข้อมูลสังเคราะห์ที่ใช้ในงานวิจัย

ชื่อข้อมูล	RW1	RW2	RW3	RW4	RW5
ความยาว	10,000	50,000	100,000	200,000	500,000

4.1.2 ข้อมูลจริง (Real World Data)

ข้อมูลจริงที่ใช้ในงานวิจัยนี้ประกอบด้วยข้อมูลจากหลาย ๆ ขอบเขต (Domain) เพื่อความหลากหลายของชุดข้อมูล โดยมีรายละเอียดตามตารางที่ 4.2 เนื่องจากเมื่อข้อมูลมีขนาดใหญ่การคำนวณเมทริกซ์โพรไฟล์จากอัลกอริทึมแสตมป์ใช้เวลานานมาก ดังนั้นถ้าข้อมูลมีขนาดใหญ่กว่า 200,00 จุดข้อมูล วิทยานิพนธ์นี้จะใช้ข้อมูล 200,000 จุดข้อมูลแรกในการทดลองยกเว้นข้อมูลการให้อาหารแมลง (Insect) ที่จะใช้ข้อมูลทั้งหมด

ตารางที่ 4.2 ตารางแสดงรายละเอียดข้อมูลจริงที่ใช้ในงานวิจัย

ชื่อข้อมูล	ความยาว	ความยาวที่ใช้	รายละเอียด
Astro	1,151,350	200,000	ข้อมูลตัวแทนวัตถุบนท้องฟ้า (Celestial objects) [33]
ECG	7,824,879	200,000	ข้อมูลคลื่นไฟฟ้าหัวใจของคนขับยานพาหนะ [34]
EEG	539,922	200,000	ข้อมูลคลื่นไฟฟ้าสมองเกิดขึ้นในการนอนหลับช่วง NREM [35]
EMG	2,887,347	200,000	ข้อมูลคลื่นไฟฟ้ากล้ามเนื้อของคนขับยานพาหนะ [34]
GAP	2,049,280	200,000	ข้อมูลการใช้พลังงานไฟฟ้า (Global active electric power) ในประเทศฝรั่งเศสช่วงปี 2006-2008 [36]
Insect	205,000	205,000	ข้อมูลการให้อาหารของเพลี้ยไก่แจ้ส้ม (Asian citrus psyllid) [37]
Seismic	40,000	40,000	ข้อมูลแผ่นดินไหวที่ Sonoma County [22]
Music	12,356	12,356	ข้อมูลเพลง Stayin' alive ของศิลปิน BeeGees

4.1.3 การวิเคราะห์เวลาและหน่วยความจำที่ต้องใช้ในการประมวลผล (Time and Space Complexity)

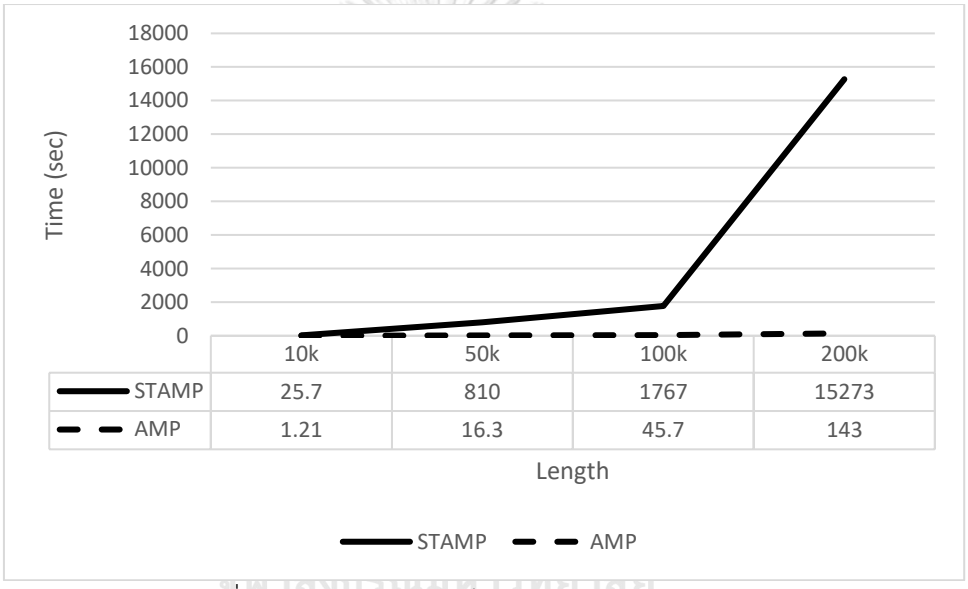
เวลาที่ต้องใช้ในการประมวลผล (Time Complexity) ของอัลกอริทึมเอเอ็มพีคือ $O(kn\log n)$ โดยที่ $k \ll n$ เมื่อ n มีค่ามาก ตารางที่ 4.3 แสดงตัวอย่างความสัมพันธ์ระหว่างค่า k และ n เพื่อแสดงให้เห็นว่าเมื่อ n มีค่ามาก k จะมีค่าน้อยกว่ามาก ๆ ในขณะที่เวลาที่ต้องใช้ในการประมวลผลของอัลกอริทึมแสตมป์คือ $O(n^2\log n)$ สำหรับหน่วยความจำที่ต้องใช้ในการประมวลผล (Space Complexity) ของทั้งสองอัลกอริทึมคือ $O(n)$

เพื่อแสดงว่าอัลกอริทึมเอเอ็มพีใช้เวลาน้อยกว่าอัลกอริทึมแสตมป์อย่างมีนัยสำคัญเมื่อข้อมูลอนุกรมเวลามีขนาดใหญ่มาก ๆ จึงได้ทำการทดลองการคำนวณเมทริกซ์โพรไฟล์ทั้ง 2 แบบโดยใช้ทั้งข้อมูลสังเคราะห์ดังแสดงในภาพที่ 4.1 และข้อมูลจริงดังแสดงในภาพที่ 4.2 สังเกตได้ว่าเมื่อข้อมูล

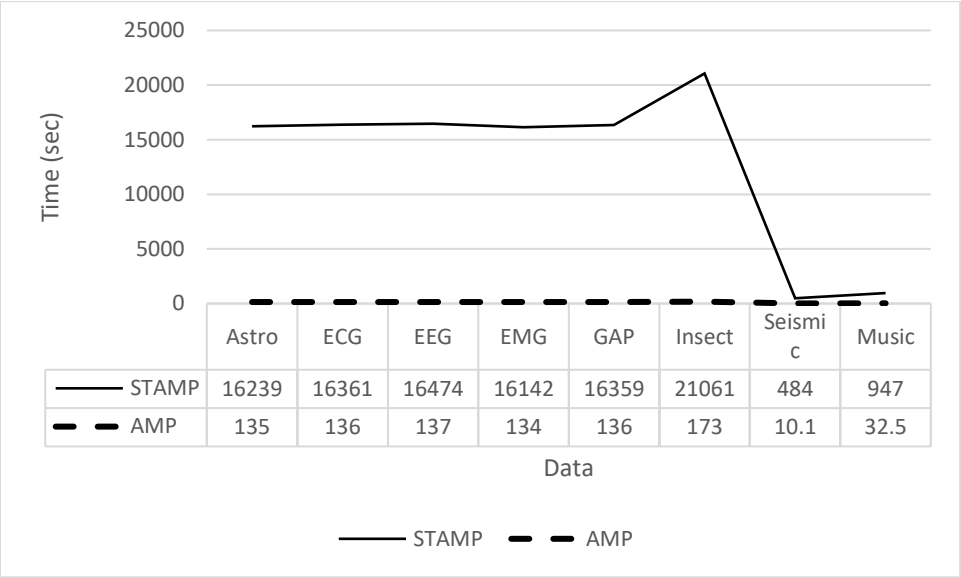
อนุกรมเวลามีขนาดใหญ่ขึ้น อัลกอริทึมเอเอ็มพีสามารถลดเวลาที่ใช้ในการคำนวณได้อย่างมาก ตัวอย่างเช่น กรณีที่ข้อมูลมีขนาด 200,000 จุดข้อมูล อัลกอริทึมเอเอ็มพีใช้เวลา 143 วินาทีหรือ 2 นาที 13 วินาที ในขณะที่อัลกอริทึมแสตมป์ใช้เวลาไปถึง 15,273 วินาทีหรือ 4 ชั่วโมง 14 นาที 33 วินาที

ตารางที่ 4.3 ตารางแสดงตัวอย่างความสัมพันธ์ระหว่างค่า k และ n

n	2^{15}	2^{16}	2^{17}	2^{18}	2^{19}	2^{20}	2^{21}	2^{22}	2^{23}
k	639	903	1277	1806	2554	3611	5107	7222	10213



ภาพที่ 4.1 ภาพแสดงเวลาที่ใช้ของอัลกอริทึมเอเอ็มพี (เส้นประ) และอัลกอริทึมแสตมป์ (เส้นทึบ) บนข้อมูลสังเคราะห์ที่มีความยาวแตกต่างกัน



ภาพที่ 4.2 ภาพแสดงเวลาที่ใช้ของอัลกอริทึมเอเอ็มพี (เส้นประ) และอัลกอริทึมแสตมป์ (เส้นทึบ) บนข้อมูลจริง

4.1.4 ความแม่นยำของโมทีฟและดิสคอร์ดผลลัพธ์

เนื่องจากอัลกอริทึมแสตมป์เป็นอัลกอริทึมแบบแม่นยำ (Exact Algorithm) และในงานวิจัยนี้ทำการเปรียบเทียบโมทีฟและดิสคอร์ดผลลัพธ์กับอัลกอริทึมแสตมป์ ดังนั้นจะถือว่าผลลัพธ์ที่ได้จากอัลกอริทึมแสตมป์เป็นผลเฉลยที่ถูกต้อง

1) อัตราส่วนซ้อนทับ (Overlapping Ratio, OR)

อัตราส่วนซ้อนทับเป็นค่าที่บ่งบอกว่าผลลัพธ์ที่ได้จากอัลกอริทึมมีการซ้อนทับ (Overlap) กับผลเฉลยมากน้อยเพียงใด ดังนั้นถ้าค่าอัตราส่วนซ้อนทับมีค่ามากหมายความว่าผลลัพธ์ที่ได้จากอัลกอริทึมนั้นมีประสิทธิภาพ ซึ่งค่าอัตราส่วนซ้อนทับคำนวณเป็นค่าเปอร์เซ็นต์ระหว่างความยาวของส่วนที่ซ้อนทับกันและความยาวของผลเฉลยดังสมการที่ (4.1)

$$OR(A,B) = \frac{|A \cap B|}{|A|} * 100 \quad (4.1)$$

เมื่อ A คือผลเฉลยของโมทีฟหรือดิสคอร์ด และ B คือผลลัพธ์ที่ได้จากอัลกอริทึม

2) ค่าเอโอดี [13]

เพื่อเป็นการวัดประสิทธิภาพของอัลกอริทึมโดยรวม งานวิจัยนี้จึงใช้ค่าเอโอดี (AoD) หรือ Accuracy-on-Detection ซึ่งบ่งบอกว่าโดยเฉลี่ยแต่ละโมทีฟและดิสคอร์ดผลลัพธ์ตรงกับผลเฉลยมากน้อยเพียงใด ค่าเอโอดีคำนวณเป็นค่าเปอร์เซ็นต์ของอัตราส่วนซ้อนทับเฉลี่ยระหว่างผลเฉลยและผลลัพธ์ที่ได้จากอัลกอริทึมดังสมการที่ (4.2)

$$AoD = \frac{\sum_{i=1}^{mt} OR(A_i, B_i) + \sum_{j=1}^{dc} OR(C_j, D_j)}{mt + dc} * 100 \quad (4.2)$$

เมื่อ mt คือจำนวนโมทีฟผลลัพธ์, dc คือจำนวนดิสคอร์ดผลลัพธ์

A_i คือผลเฉลยโมทีฟตัวที่ i

B_i คือโมทีฟผลลัพธ์จากอัลกอริทึมตัวที่ i

C_i คือผลเฉลยดิสคอร์ดตัวที่ j

และ D_i คือดิสคอร์ดผลลัพธ์จากอัลกอริทึมตัวที่ j

จากนิยามของโมทีฟและดิสคอร์ดที่ใช้ในงานวิจัยนี้ จะได้ว่า $mt = 2$ และ $dc = 1$ และในงานวิจัยนี้จะถือว่าโมทีฟและดิสคอร์ดผลลัพธ์เป็นผลลัพธ์ที่ใกล้เคียงเมื่อเทียบกับผลเฉลยก็ต่อเมื่อค่าเอโอดีมีค่ามากกว่า 90%

ผลการทดลองการค้นพบโมทีฟและดิสคอร์ดด้วยอัลกอริทึมเอเอ็มพีบนข้อมูลสังเคราะห์และข้อมูลจริงแสดงในตารางที่ 4.4 และ 4.5 ตามลำดับ

ตารางที่ 4.4 ตารางแสดงผลการทดลองอัลกอริทึมเอเอ็มพีบนข้อมูลสังเคราะห์

ชื่อข้อมูล	AMP			
	OR_motif1 (%)	OR_motif2 (%)	OR_discord (%)	AoD (%)
RW1	99	98	98	98.33
RW2	99.2	99.2	98.2	98.87
RW3	93.6	99.5	99.6	97.57
RW4	99.4	99.4	99.4	99.4
RW5	99.46	99.42	99.28	99.39

ตารางที่ 4.5 ตารางแสดงผลการทดลองอัลกอริทึมเอเอ็มพีบนข้อมูลจริง

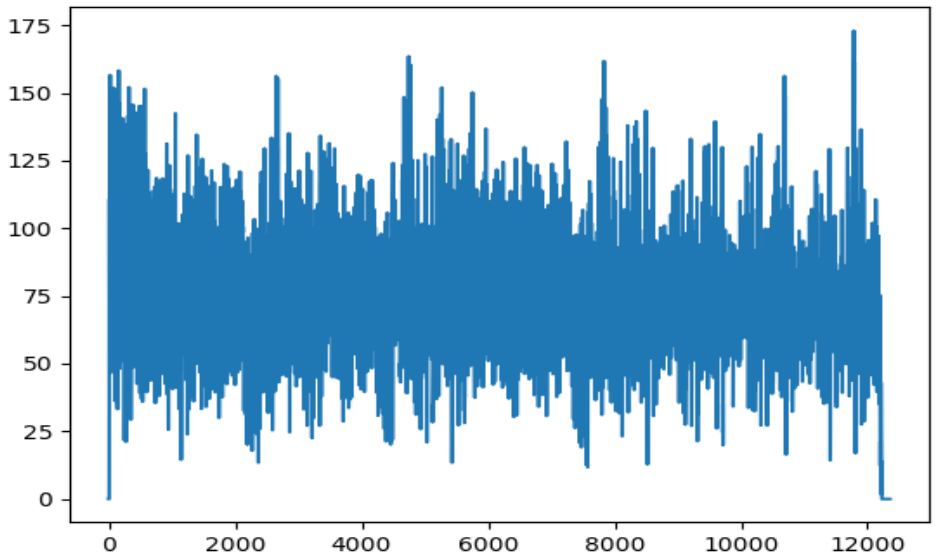
ชื่อข้อมูล	AMP			
	OR_motif1 (%)	OR_motif2 (%)	OR_discord (%)	AoD (%)
Astro	98	97.9	98.7	98.2
ECG	99	99	98.8	98.93
EEG	99.82	99.82	96	98.55
EMG	99.18	98.85	98.95	98.99
GAP	99.4	99.4	98.45	99.08
Insect	100	100	97.14	99.05
Seismic	99.15	99.15	81.29	93.19
Music	93.98	93.98	95.14	94.37

จากผลการทดลองพบว่าค่าเอโอดีที่ได้จากอัลกอริทึมเอเอ็มพีนั้นมีค่ามากกว่า 90% ในทุกชุดข้อมูลทั้งข้อมูลสังเคราะห์และข้อมูลจริงตามที่กำหนดไว้ ดังนั้นนั้นหมายความว่าโมทีฟและดิสคอร์ดผลลัพธ์ที่ได้จากอัลกอริทึมเอเอ็มพีมีความใกล้เคียงกับผลลัพธ์จากอัลกอริทึมแสตมป์ ซึ่งตรงกับวัตถุประสงค์ของหลักของงานวิจัยนี้คือต้องการลดเวลาในการคำนวณเมทริกซ์โพรไฟล์ แต่ยังได้ผลลัพธ์ที่ใกล้เคียงเดิม

4.1.5 กรณีศึกษาข้อมูลเพลง (Case Study : Music Data)

เพื่อเป็นการแสดงว่าอัลกอริทึมเอเอ็มพีสามารถนำไปประยุกต์ใช้ได้จริง ในส่วนนี้จึงนำเสนอกรณีศึกษาสำหรับข้อมูลเพลง โดยเพลงที่ใช้คือ เพลง Stayin’ alive ของศิลปิน BeeGees ซึ่งเป็นเพลงที่มีชื่อเสียงและแพร่หลายไปทั่วโลก และเป็นหนึ่งในเพลงที่ใช้ในการค้นพบโมทีฟและดิสคอร์ดในงานวิจัย [11] สำหรับกรณีศึกษานี้ทำการทดลองทั้งการค้นพบโมทีฟและดิสคอร์ด

ก่อนที่จะทำการประมวลผล ข้อมูลจะถูกแปลงจากข้อมูลดิบ (Raw Data) ให้เป็นข้อมูลสัมประสิทธิ์เคปสตรัลที่คำนวณบนแกนความถี่แบบเมล (Mel Frequency Cepstral Coefficient Data, MFCC) ซึ่งเป็นชนิดของข้อมูลที่ถูกใช้อย่างแพร่หลายในปัญหาการค้นหาข้อมูลหรือสารสนเทศสำหรับเพลง (Music Information Retrieval) [38] ภาพที่ 4.3 แสดงข้อมูลสัมประสิทธิ์เคปสตรัลที่คำนวณบนแกนความถี่แบบเมลของเพลง Stayin' alive ที่มีความยาว 12,356 จุดข้อมูล (4 นาที 46 วินาที) สำหรับค่าพารามิเตอร์ที่กำหนดในอัลกอริทึมเอเอ็มพีคือ ความยาวของลำดับย่อย m เป็น 432 หมายถึงโมทีฟและดิสคอร์ดที่ต้องการในเพลงมีความยาว 10 วินาที และค่าความน่าจะเป็น p เป็น 0.999



ภาพที่ 4.3 ข้อมูลสัมประสิทธิ์เคปสตรัลที่คำนวณบนแกนความถี่แบบเมลของเพลง Stayin' alive ที่มีความยาว 12,356 จุดข้อมูล

จำนวนการวนซ้ำในการคำนวณลดลงจาก 11,925 เป็น 399 และอัลกอริทึมเอเอ็มพีใช้เวลาในการคำนวณ 32.5 วินาที ในขณะที่อัลกอริทึมแสตมป์ใช้เวลาในการคำนวณ 947 วินาที ซึ่งเร็วกว่าประมาณ 30 เท่า

โมทีฟผลลัพธ์ที่ได้จากอัลกอริทึมเอเอ็มพีคือ ลำดับย่อยที่ตำแหน่ง 4109 และ 7199 ในขณะที่โมทีฟผลลัพธ์จากอัลกอริทึมแสตมป์คือ ลำดับย่อยที่ตำแหน่ง 4083 และ 7173 ดังแสดงในภาพที่ 4.4 ซึ่งมีค่าใกล้เคียงกัน โดยมีค่าอัตราส่วนซ้อนทับของโมทีฟตัวแรกและตัวที่สองเป็น 93.98% เท่ากัน เมื่อแปลงโมทีฟผลลัพธ์กลับไปในเพลง โมทีฟทั้งคู่คือส่วนของเพลงที่มีความยาว 10 วินาที โดยเริ่มต้นที่

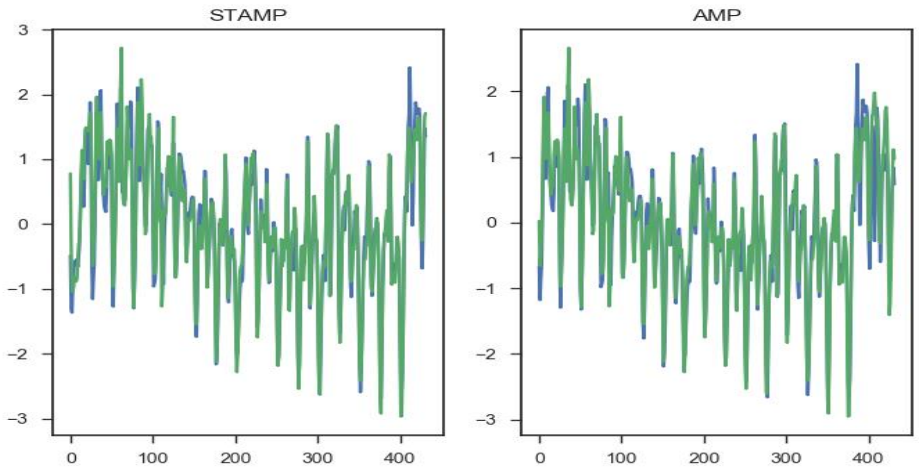
นาทีที่ 1:35 และ 2:47 ซึ่งเป็นท่อนที่เกิดขึ้นบ่อย มีการร้องประสานเสียง หรือเป็นท่อนฮุกของเพลง (Chorus or Refrain Part)

นอกจากนั้นดิสคอร์ดผลลัพธ์ที่ได้จากอัลกอริทึมเอเอ็มพีคือ ลำดับย่อยที่ตำแหน่ง 11,820 และดิสคอร์ดผลลัพธ์จากอัลกอริทึมแสตมป์คือ ลำดับย่อยที่ตำแหน่ง 11,841 ดังแสดงในภาพที่ 4.5 โดยมีค่าอัตราส่วนซ้อนทับเป็น 95.14% ผลลัพธ์ของดิสคอร์ดที่ได้เป็นส่วนของเพลงที่มีความยาว 10 วินาที โดยเริ่มต้นที่นาทีที่ 4:33 ซึ่งเป็นท่อนที่เกิดขึ้นน้อยในเพลง

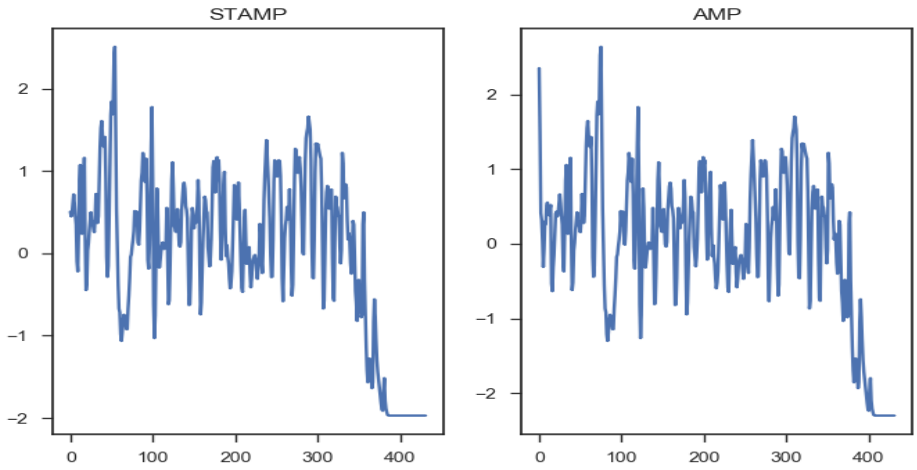
เพราะฉะนั้นจากผลลัพธ์ที่ได้ทั้งโมทีฟและดิสคอร์ดทำให้สรุปได้ว่าอัลกอริทึมเอเอ็มพีให้ผลลัพธ์ที่ดีและใช้ได้จริง

ตารางที่ 4.6 ตารางแสดงผลการทดลองการค้นพบโมทีฟและดิสคอร์ดบนข้อมูลเพลง

	Number of iterations	time	Motif pair	Motif time	Discord	Discord time
AMP	399	32.5 sec	4,109, 7,199	1:35, 2:47	11,820	4:33
STAMP	11,925	15 min 47 sec	4,083, 7,173	1:35, 2:47	11,841	4:33



ภาพที่ 4.4 โมทีฟผลลัพธ์ที่ความยาว 432 จุดข้อมูลจากอัลกอริทึมเอเอ็มพี (ซ้าย) และอัลกอริทึมแสตมป์ (ขวา)



ภาพที่ 4.5 ดิสคอร์ดผลลัพธ์ที่ความยาว 432 จุดข้อมูลจากอัลกอริทึมเอเอ็มพี (ซ้าย) และอัลกอริทึมแสตมป์ (ขวา)

4.2 การทดลองสำหรับอัลกอริทึมพีแอลเอเอ็มพี

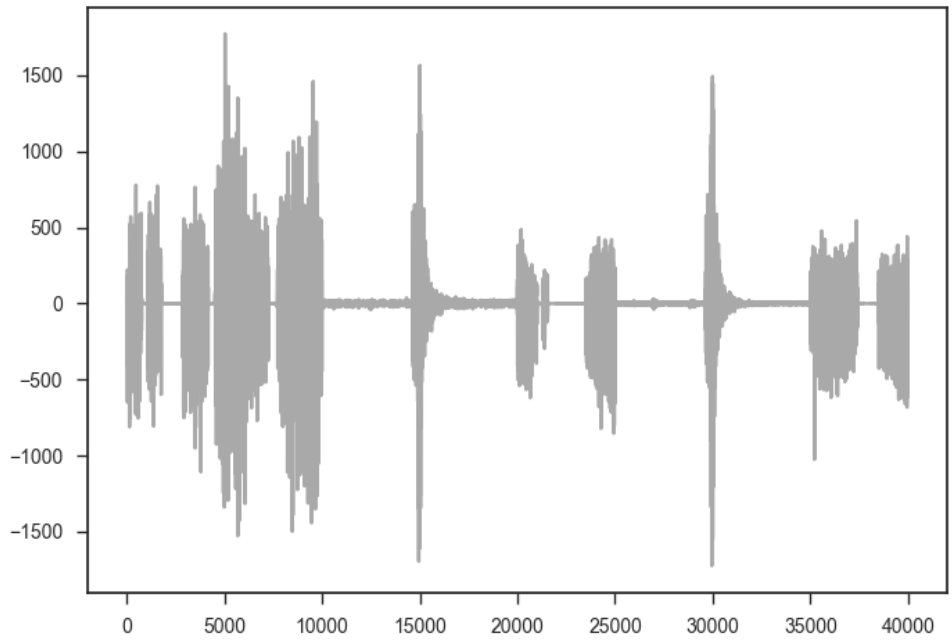
เนื่องจากอัลกอริทึมพีแอลเอเอ็มพีนั้นเป็นส่วนเพิ่มเติมที่ปรับปรุงมาจากอัลกอริทึมเอเอ็มพี โดยใช้ประโยชน์จากอัลกอริทึมการค้นพบโมทีฟที่ความยาวเหมาะสม (Proper Length Motif Discovery) [21] มีวัตถุประสงค์เพื่อแก้ปัญหาที่ผู้ใช้จำเป็นต้องกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อยในอัลกอริทึมเอเอ็มพี ดังนั้นในส่วนนี้จะนำเสนอเฉพาะกรณีศึกษาที่อัลกอริทึมพีแอลเอเอ็มพีถูกนำไปใช้

4.2.1 กรณีศึกษาข้อมูลแผ่นดินไหว (Case Study : Seismic Data)

การตรวจหารูปแบบของการเกิดแผ่นดินไหว (Earthquake) ที่เกิดขึ้นบ่อยครั้งนั้นเป็นเรื่องสำคัญ เพราะว่าจะนำไปสู่การค้นหาการเกิดแผ่นดินไหวนำ (Foreshock) การเกิดแผ่นดินไหวตาม (Aftershock) และภูเขาไฟระเบิด (Volcanic Activity) เป็นต้น งานวิจัย [22] แสดงว่าการค้นพบโมทีฟในข้อมูลแผ่นดินไหวจะเป็นประโยชน์ต่อการตรวจหารูปแบบของแผ่นดินไหวที่เกิดขึ้นบ่อยครั้ง แต่อย่างไรก็ตามข้อมูลแผ่นดินไหวเป็นข้อมูลที่มีข้อมูลรบกวน (Noise) ค่อนข้างมาก และผลลัพธ์ที่ได้จากการค้นพบโมทีฟที่ตรึงค่าความยาวของลำดับย่อยนั้นอาจจะเป็นคำตอบที่ผิดหรือไม่ใช่รูปแบบของแผ่นดินไหวที่ต้องการ ดังนั้นการค้นหาโมทีฟที่มีความยาวของลำดับย่อยหลายค่า แล้วเลือกโมทีฟผลลัพธ์ที่มีความยาวที่เหมาะสมจึงเป็นการแก้ปัญหานี้ได้

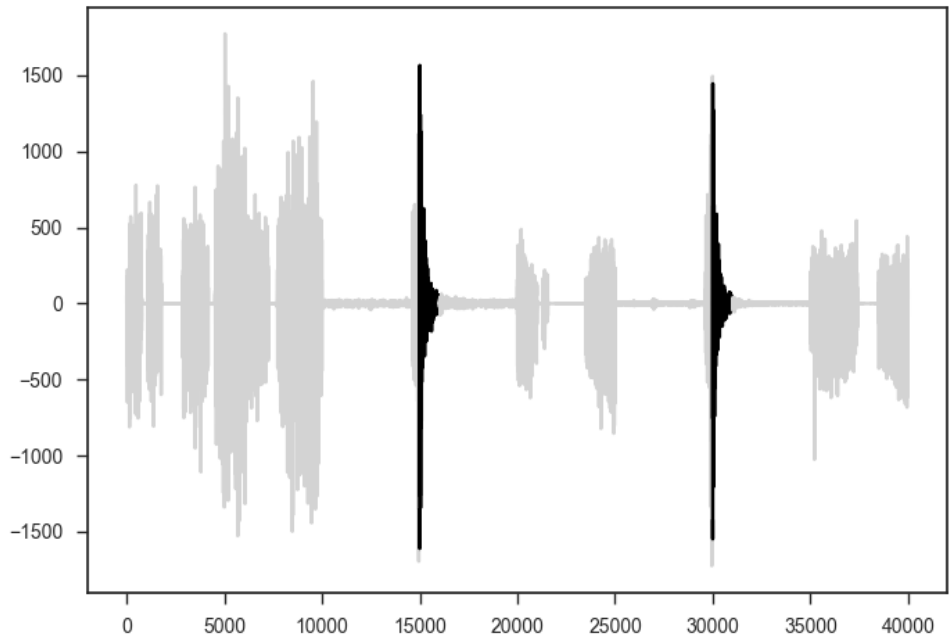
ข้อมูลที่ใช้เป็นข้อมูลแผ่นดินไหวที่ Sonoma County, California มีขนาด 40,00 จุดข้อมูล ความถี่ 50Hz. โดยรูปแบบของแผ่นดินไหวเกิดขึ้นในข้อมูลนี้ 2 ครั้งซึ่งแต่ละครั้งมีความยาว 20 วินาที

ดังแสดงในภาพที่ 4.6 อัลกอริทึมพีแอลเอเอ็มพีจะถูกใช้ในการค้นหาโมทีฟที่ความยาวเหมาะสมสำหรับข้อมูลนี้

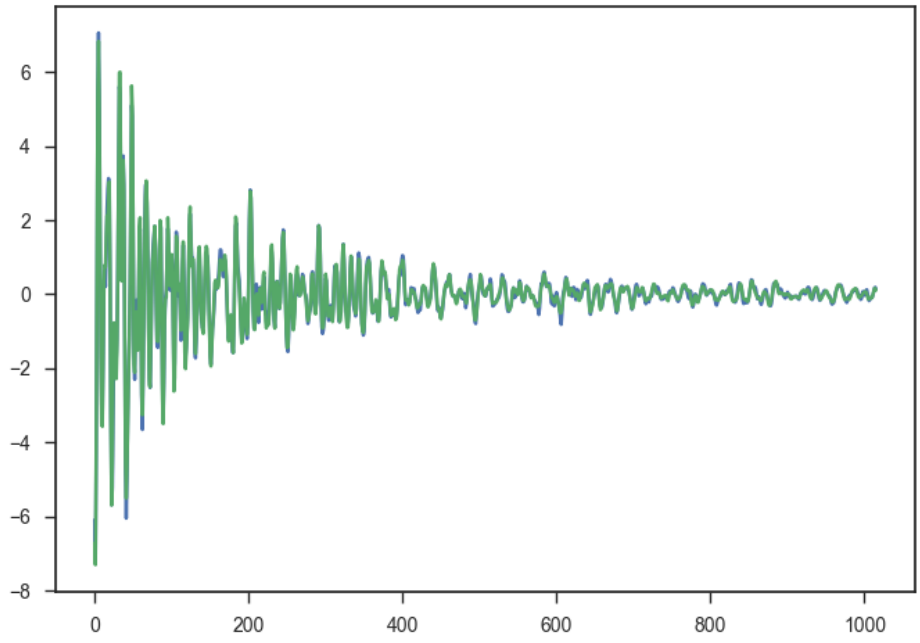


ภาพที่ 4.6 ข้อมูลแผ่นดินไหวมีขนาด 40,000 จุดข้อมูล

ผลการทดลองพบว่าโมทีฟผลลัพธ์ที่ได้อยู่ในตำแหน่งที่ 15,000 และ 30,000 ที่ความยาว 1,016 จุดข้อมูลดังแสดงในภาพที่ 4.7 ซึ่งตรงกับรูปแบบของแผ่นดินไหว 2 ครั้งที่เกิดขึ้นในข้อมูลนี้ดังที่กล่าวไว้ก่อนหน้า เพราะฉะนั้นกรณีศึกษานี้แสดงให้เห็นว่าอัลกอริทึมเอเอ็มพีสามารถค้นพบโมทีฟผลลัพธ์ได้อย่างมีประสิทธิภาพที่ความยาวเหมาะสม เมื่อผู้ใช้ไม่ทราบค่าความยาวของโมทีฟหรือลำดับย่อยที่แน่นอน ซึ่งเป็นการแก้ปัญหาการกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อยของทั้งอัลกอริทึมแสตมป์และเอเอ็มพีตามวัตถุประสงค์ของงานวิจัยนี้



ภาพที่ 4.7 โมทีฟผลลัพธ์ที่ได้จากอัลกอริทึมพีแอลเอเอ็มพีที่ตำแหน่ง 15,000 และตำแหน่ง 30,000



ภาพที่ 4.8 โมทีฟผลลัพธ์จากอัลกอริทึมพีแอลเอเอ็มพีที่ความยาว 1,016 จุดข้อมูล

บทที่ 5 สรุปผลการวิจัย และข้อเสนอแนะ

5.1 สรุปผลงานวิจัย

การค้นพบโมทีฟและดิสคอร์ดโดยใช้เมทริกซ์โพรไฟล์นั้นให้ผลลัพธ์ที่มีประสิทธิภาพ เนื่องจากเมทริกซ์โพรไฟล์เป็นการคำนวณค่าระยะห่างยุคลิดแบบแม่นยำ (Exact) แต่อย่างไรก็ตามเวลาที่ใช้ในการคำนวณเมทริกซ์โพรไฟล์นั้นสูง โดยมีขีดจำกัดเชิงสัญกรณ์คือ $O(n^2 \log n)$ ซึ่งเมื่อนำไปใช้กับข้อมูลอนุกรมเวลาที่มีขนาดใหญ่แล้วนั้น ทำให้เวลาในการคำนวณยิ่งสูงตามไปด้วย ในขณะที่ผลลัพธ์ของโมทีฟและดิสคอร์ดที่ต้องการจากเมทริกซ์โพรไฟล์มีเพียงแค่ค่าต่ำสุดและค่าสูงสุดเท่านั้น นอกจากนั้นอีกหนึ่งปัญหาของเมทริกซ์โพรไฟล์คือ ผู้ใช้จำเป็นต้องกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อย ซึ่งในบางครั้งผู้ใช้ไม่ทราบค่าที่แน่นอนหรือต้องใช้ความรู้เฉพาะด้านในการกำหนด

งานวิจัยนี้จึงได้นำเสนอเมทริกซ์โพรไฟล์แบบประมาณ โดยปรับปรุงจากเมทริกซ์โพรไฟล์ซึ่งใช้เวลาในการคำนวณสูง เมทริกซ์โพรไฟล์แบบประมาณสามารถลดเวลาที่ใช้ในการคำนวณและคงไว้ซึ่งค่าสูงสุดและต่ำสุดที่ใกล้เคียงกับเมทริกซ์โพรไฟล์สำหรับนำไปใช้ในปัญหาการค้นพบโมทีฟและดิสคอร์ด โดยมีแนวคิดของงานวิจัยที่ไม่จำเป็นต้องใช้ลำดับย่อยทั้งหมดที่เป็นไปได้ในการคำนวณ แต่ทำการหาจำนวนของลำดับย่อยบางส่วนที่จะใช้เท่านั้น และจัดลำดับของลำดับย่อยที่ใช้ใหม่เพื่อให้เข้าใกล้คำตอบที่ต้องการเร็วขึ้น ผลลัพธ์ของเวลาที่ใช้ในการคำนวณเมทริกซ์โพรไฟล์แบบประมาณได้ทำการนำเสนอไว้ในบทที่ 4 ซึ่งจะเห็นได้ว่าวิธีการที่นำเสนอนั้นใช้เวลาน้อยกว่าเมทริกซ์โพร์ไฟล์มาก โดยเฉพาะเมื่อข้อมูลอนุกรมเวลามีขนาดใหญ่ขึ้น

ในส่วนของความแม่นยำของโมทีฟและดิสคอร์ดผลลัพธ์นั้น จากผลการทดลองในบทที่ 4 จะเห็นได้ว่าวิธีการที่นำเสนอมีค่าเอโอดีที่สูงในทุกชุดข้อมูล หมายความว่าวิธีการที่นำเสนอยังคงไว้ซึ่งผลลัพธ์ที่ใกล้เคียงกับเมทริกซ์โพรไฟล์ แม้ว่าโมทีฟและดิสคอร์ดผลลัพธ์ที่ได้จะไม่ได้ถูกต้องหรือเหมือนกับผลลัพธ์จากเมทริกซ์โพรไฟล์ทั้งหมด เนื่องด้วยข้อจำกัดของวิธีการที่นำเสนอนั้นให้ผลลัพธ์เป็นแบบประมาณ ไม่ใช่แบบแม่นยำ แต่มีความใกล้เคียงค่อนข้างมาก เมื่อแลกเปลี่ยนกับเวลาในการคำนวณที่ลดลงไปอย่างมากแล้วนั้น นับเป็นการแลกเปลี่ยนที่คุ้มค่า

อีกหนึ่งปัญหาของเมทริกซ์โพรไฟล์ที่กล่าวไว้คือการกำหนดค่าพารามิเตอร์ความยาวของลำดับย่อย ในงานวิจัยนี้จึงนำเสนออัลกอริทึมพีแอลเอเอ็มพีซึ่งเป็นอัลกอริทึมสำหรับการค้นพบโมทีฟโดยได้ประยุกต์ใช้วิธีการค้นพบโมทีฟที่ความยาวเหมาะสม (Proper Length Motif Discovery) รวมกับเมทริกซ์โพร์ไฟล์แบบประมาณ เพื่อให้ผลลัพธ์โมทีฟที่ความยาวเหมาะสม กรณีศึกษาข้อมูลแผ่นดินไหวในบทที่ 4 แสดงให้เห็นแล้วว่าอัลกอริทึมพีแอลเอเอ็มพีสามารถนำไปใช้ได้จริงและยังให้ผลลัพธ์ที่ถูกต้อง

5.2 ข้อจำกัดและข้อเสนอแนะ

ข้อเสนอแนะต่อไปนี้เป็นแนวทางในการวิจัยที่จะนำเสนอเพื่อปรับปรุงอัลกอริทึมการค้นพบโมทีฟและดิสคอร์ดโดยใช้เมทริกซ์โพรไฟล์แบบประมาณให้ดียิ่งขึ้น รวมถึงแก้ปัญหาข้อจำกัดต่าง ๆ ในงานนี้

เนื่องจากในงานวิจัย [14] ได้มีการนำเมทริกซ์โพรไฟล์ไปประยุกต์ใช้ในปัญหาของการทำเหมืองข้อมูลอนุกรมเวลาอื่น ๆ ด้วย ตัวอย่างเช่น การจัดกลุ่ม (Clustering) การแบ่งส่วน (Segmentation) การหาตัวแทนกลุ่ม (Shapelet Discovery) เป็นต้น ดังนั้นเมทริกซ์โพรไฟล์แบบประมาณน่าจะสามารถประยุกต์ใช้ในปัญหาอื่น ๆ ได้เช่นเดียวกัน

นอกจากนั้นอัลกอริทึมพีแอลเอเอ็มพีในได้นำเสนอในบทที่ 3 เป็นอัลกอริทึมสำหรับการค้นพบโมทีฟเท่านั้น อาจจะมีการปรับปรุงและพัฒนาวิธีการนี้สำหรับการค้นพบดิสคอร์ดที่ความยาวเหมาะสมได้ อีกทั้งวิธีการนี้ยังใช้เวลานาน เนื่องจากในแต่ละรอบการทำงานต้องทำการคำนวณเมทริกซ์โพรไฟล์ที่ความยาวของลำดับย่อยที่พิจารณาในรอบนั้น ๆ ใหม่ ดังนั้นอาจทำการปรับปรุงอัลกอริทึมจากแนวคิดการคำนวณแบบต่อยอดเพิ่มเติม (Incremental) กล่าวคือเมื่อทำการคำนวณเมทริกซ์โพรไฟล์ที่ความยาวของลำดับย่อย m แล้ว เมทริกซ์โพรไฟล์ที่ความยาวของลำดับย่อย $m + 1$ สามารถหาได้โดยมีการคำนวณเพิ่มเติมแค่เล็กน้อยเท่านั้น ก็จะช่วยลดเวลาในการคำนวณลงได้

อย่างไรก็ตามอีกหนึ่งข้อจำกัดของทั้งเมทริกซ์โพรไฟล์และเมทริกซ์โพรไฟล์แบบประมาณคือทั้งสองวิธีคำนวณระยะห่างของคู่ของลำดับย่อยที่มีความยาวเท่ากันเท่านั้น ไม่สามารถคำนวณระยะห่างของคู่ของลำดับย่อยที่มีความยาวแตกต่างกันได้ ทำให้โมทีฟหรือดิสคอร์ดผลลัพธ์ที่ได้จากสองวิธีนี้จำเป็นต้องมีความยาวเท่ากัน ซึ่งในบางข้อมูลโมทีฟและดิสคอร์ดที่ถูกต้องอาจไม่จำเป็นต้องมีความยาวเท่ากัน เพราะฉะนั้นในอนาคตอาจมีการปรับปรุงโดยใช้ใช้การวัดระยะห่างแบบไดนามิกไทม์วอร์ปปิง (Dynamic Time Warping) หรือลำดับย่อยร่วมยาวสุด (Longest Common Subsequence) ซึ่งรองรับการวัดระยะห่างของข้อมูลอนุกรมเวลาที่มีความยาวต่างกัน

รายการอ้างอิง

[1] Esling, P., & Agon, C. Time-series data mining. *ACM Computing Surveys,* 45(1), 1-34. (2012)

[2] Fu, T.-c. A review on time series data mining. *Engineering Applications of Artificial Intelligence,* 24(1), 164-181. (2011)

[3] ZHONG, S., KHOSHGOFTAAR, T. M., & SELIYA, N. CLUSTERING-BASED NETWORK INTRUSION DETECTION. *International Journal of Reliability, Quality and Safety Engineering,* 14(02), 169-187. (2007)

[4] Patel, P., Keogh, E., Lin, J., & Lonardi, S. (2002). *Mining motifs in massive time series databases.* Paper presented at the Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on.

[5] Keogh, E., & Lin, J. Clustering of time-series subsequences is meaningless: implications for previous and future research. *Knowledge and information systems,* 8(2), 154-177. (2005)

[6] Bakshi, B., & Stephanopoulos, G. Representation of process trends—IV. Induction of real-time patterns from operating data for diagnosis and supervisory control. *Computers & Chemical Engineering,* 18(4), 303-332. (1994)

[7] Yankov, D., Keogh, E., Medina, J., Chiu, B., & Zordan, V. (2007). *Detecting time series motifs under uniform scaling*. Paper presented at the Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, San Jose, California, USA.

[8] Mueen, A., Keogh, E., Zhu, Q., Cash, S., & Westover, B. Exact Discovery of Time Series Motifs *Proceedings of the* 2009 *SIAM International Conference on Data Mining* (pp. 473-484).

[9] McGovern, A., Rosendahl, D. H., Brown, R. A., & Droegemeier, K. K. Identifying predictive multi-dimensional time series motifs: an application to severe weather prediction. *Data Mining and Knowledge Discovery,* 22(1), 232-258. (2011)

[10] Cassisi, C., Aliotta, M., Cannata, A., Montalto, P., Patanè, D., Pulvirenti, A., & Spampinato, L. Motif Discovery on Seismic Amplitude Time Series: The Case Study of Mt Etna 2011 Eruptive Activity. *Pure and Applied Geophysics,* 170(4), 529-545. (2013)

[11] Silva, D. F., Yeh, C.-C. M., Batista, G. E. d. A. P. A., & Keogh, E. (2016). *SiMPle: assessing music similarity using subsequences joins.* Paper presented at the International Society for Music Information Retrieval Conference, XVII.

[12] Mueen, A. Time series motif discovery: dimensions and applications. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery,* 4(2), 152-159. (2014)

[13] Sivaraks, H., & Ratanamahatana, C. A. Robust and accurate anomaly detection in ECG artifacts using time series motif discovery. *Computational and mathematical methods in medicine,* 2015. (2015)

[14] Yeh, C.-C. M., Zhu, Y., Ulanova, L., Begum, N., Ding, Y., Dau, H. A., Silva, D. F., Mueen, A., & Keogh, E. (2016). *Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View that Includes Motifs, Discords and Shapelets.* Paper presented at the Data Mining (ICDM), 2016 IEEE 16th International Conference on.

[15] Lotufo, R. A., & Zampirolli, F. A. (2001). *Fast multidimensional parallel Euclidean distance transform based on mathematical morphology.* Paper presented at the Computer Graphics and Image Processing, 2001 Proceedings of XIV Brazilian Symposium on.

[16] Meijster, A., Roerdink, J. B., & Hesselink, W. H. (2002). A general algorithm for computing distance transforms in linear time *Mathematical Morphology and its applications to image and signal processing* (pp. 331-340): Springer.

[17] Berndt, D. J., & Clifford, J. (1994). *Using dynamic time warping to find patterns in time series.* Paper presented at the KDD workshop.

[18] Keogh, E. J., & Pazzani, M. J. (2000). *Scaling up dynamic time warping for datamining applications.* Paper presented at the Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining.

[19] Lawrence, I., & Lin, K. A concordance correlation coefficient to evaluate reproducibility. *Biometrics*, 255-268. (1989)

[20] Nunthanid, P., Niennattrakul, V., & Ratanamahatana, C. A. (2011, 17-19 May 2011). *Discovery of variable length time series motif.* Paper presented at the The 8th Electrical Engineering/ Electronics, Computer, Telecommunications and Information Technology (ECTI) Association of Thailand - Conference 2011.

[21] Yingchareonthawornchai, S., Sivaraks, H., Rakthanmanon, T., & Ratanamahatana, C. A. (2013). *Efficient proper length time series motif discovery.* Paper presented at the Data Mining (ICDM), 2013 IEEE 13th International Conference on.

[22] Zhu, Y., Zimmerman, Z., Senobari, N. S., Yeh, C.-C. M., Funning, G., Mueen, A., Brisk, P., & Keogh, E. (2016). *Matrix Profile II: Exploiting a Novel Algorithm and GPUs to Break the One Hundred Million Barrier for Time Series Motifs and Joins.* Paper presented at the Data Mining (ICDM), 2016 IEEE 16th International Conference on.

[23] Shorten, G., & Burke, M. Use of dynamic time warping for accurate ECG signal timing characterization. *Journal of medical engineering & technology,* 38(4), 188-201. (2014)

[24] Bollerslev, T., & Mikkelsen, H. O. Modeling and pricing long memory in stock market volatility. *Journal of econometrics,* 73(1), 151-184. (1996)

[25] Ball, W. R. Other questions on probability. *Mathematical Recreations and Essays,* 45. (1960)

[26] Mueen, A., & Keogh, E. (2010). *Online discovery and maintenance of time series motifs*. Paper presented at the Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, Washington, DC, USA.

[27] Chiu, B., Keogh, E., & Lonardi, S. (2003). *Probabilistic discovery of time series motifs*. Paper presented at the Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, Washington, D.C.

[28] Keogh, E., Lin, J., Lee, S.-H., & Van Herle, H. Finding the most unusual time series subsequence: algorithms and applications. *Knowledge and information systems,* 11(1), 1-27. (2007)

[29] Keogh, E., Lin, J., & Fu, A. (2005). *Hot sax: Efficiently finding the most unusual time series subsequence.* Paper presented at the Data mining, fifth IEEE international conference on.

[30] Huang, T., Zhu, Y., Mao, Y., Li, X., Liu, M., Wu, Y., Ha, Y., & Dobbie, G. (2016). *Parallel Discord Discovery.* Paper presented at the Pacific-Asia Conference on Knowledge Discovery and Data Mining.

[31] Mueen, A., Viswanathan, K., Gupta, C., & Keogh, E. The fastest similarity search algorithm for time series subsequences under Euclidean distance. *url: www. cs. unm. edu/~ mueen/FastestSimilaritySearch. html (accessed* 24 *May,* 2016). (2015)

[32] Rakthanmanon, T., Keogh, E. J., Lonardi, S., & Evans, S. (2011). *Time series epenthesis: Clustering time series streams requires ignoring some data.* Paper presented at the Data Mining (ICDM), 2011 IEEE 11th International Conference on.

[33] Soldi, S., Beckmann, V., Baumgartner, W., Ponti, G., Shrader, C. R., Lubiński, P., Krimm, H., Mattana, F., & Tueller, J. Long-term variability of AGN at hard X-rays. *Astronomy & Astrophysics,* 563, A57. (2014)

[34] Healey, J. A., & Picard, R. W. Detecting stress during real-world driving tasks using physiological sensors. *IEEE Transactions on intelligent transportation systems,* 6(2), 156-166. (2005)

[35] Terzano, M. G., Parrino, L., Smerieri, A., Chervin, R., Chokroverty, S., Guilleminault, C., Hirshkowitz, M., Mahowald, M., Moldofsky, H., & Rosa, A. Atlas, rules, and recording techniques for the scoring of cyclic alternating pattern (CAP) in human sleep. *Sleep medicine,* 3(2), 187-199. (2002)

[36] Lichman, M. (2013). UCI machine learning repository.

[37] Neupane, D., Moss, C. B., & van Bruggen, A. (2016). *Estimating citrus production loss due to citrus huanglongbing in Florida.* Paper presented at the 2016 Annual Meeting, Southern Agricultural Economics Association.

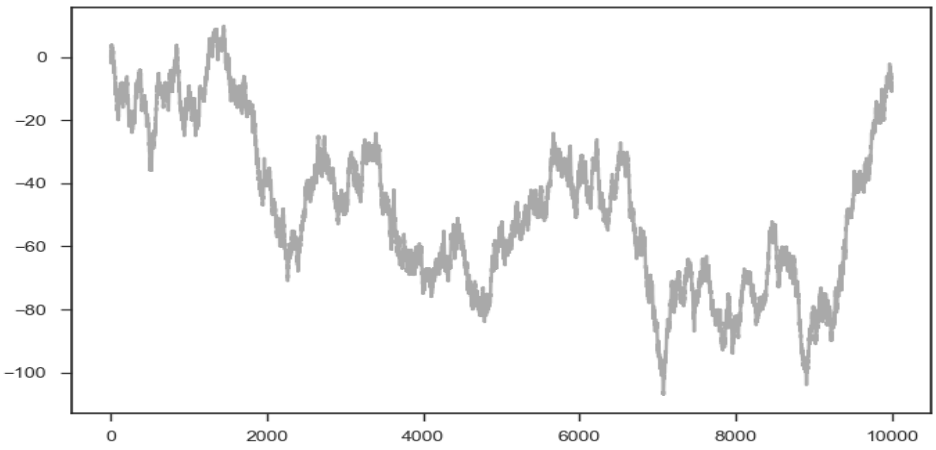
[38] Müller, M. (2007). *Information retrieval for music and motion* (Vol. 2): Springer.

ภาคผนวก

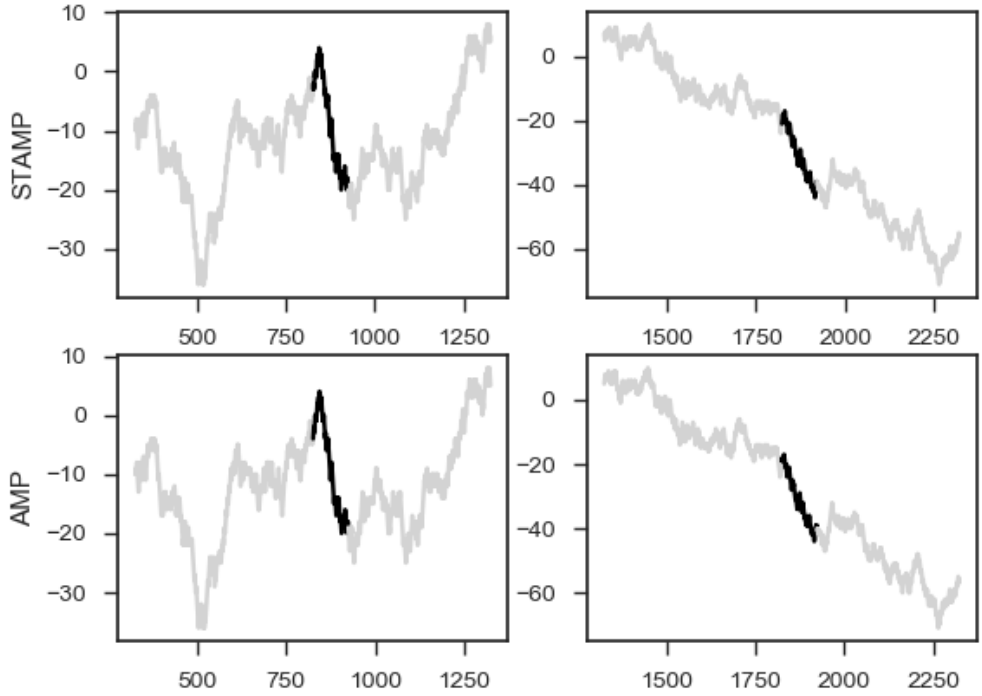
ภาคผนวก ก

ข้อมูลทั้งหมด 12 ชุดยกเว้นข้อมูลเพลง Music (แสดงไว้ในส่วนกรณีศึกษาข้อมูลเพลง) ซึ่งแบ่งเป็นข้อมูลสังเคราะห์ 5 ชุด และข้อมูลจริง 7 ชุด ที่กล่าวไว้ในบทที่ 4 รวมถึงโมทีฟและดิสคอร์ดผลลัพธ์ที่ได้จากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพีของข้อมูลแต่ละชุด จะถูกแสดงในส่วนนี้ โดยในแต่ละภาพจะประกอบด้วย ข้อมูล ข้อมูลที่มีการแสดงตำแหน่งของโมทีฟผลลัพธ์จากทั้งสองอัลกอริทึม ข้อมูลที่มีการแสดงตำแหน่งของดิสคอร์ดผลลัพธ์จากทั้งสองอัลกอริทึม โมทีฟผลลัพธ์จากทั้งสองอัลกอริทึม และดิสคอร์ดผลลัพธ์จากทั้งสองอัลกอริทึม

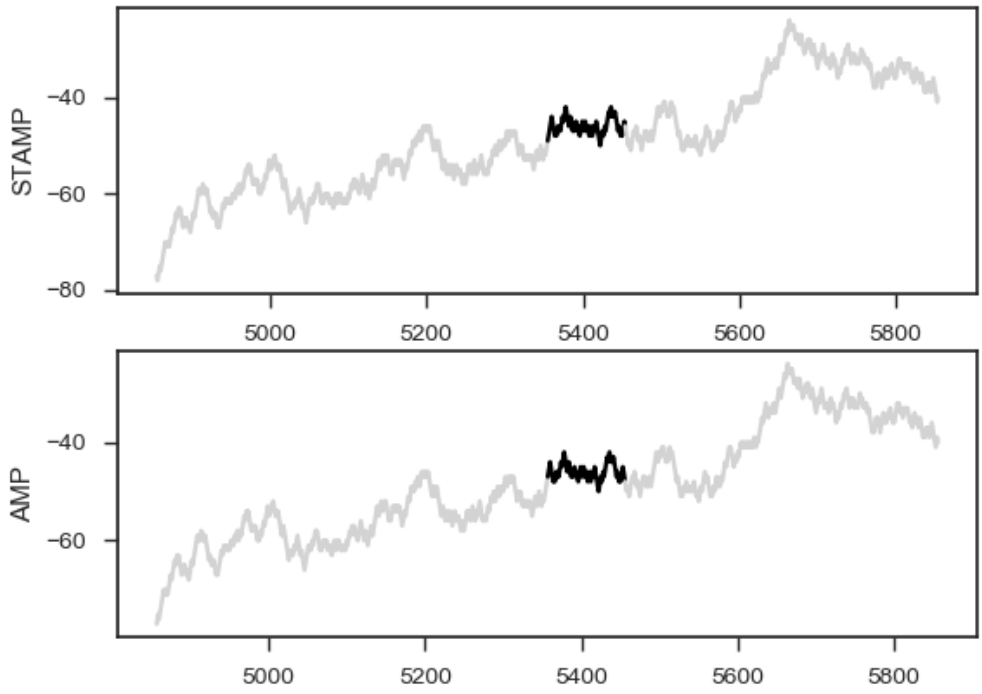
1. ข้อมูลสังเคราะห์ชุดที่ 1 มีความยาว 10,000 จุดข้อมูล (RW1)



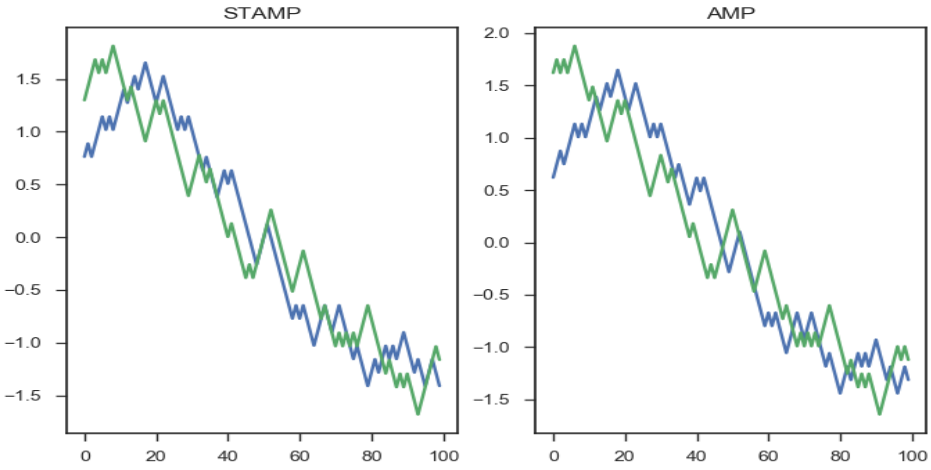
ข้อมูลสังเคราะห์ชุดที่ 1 มีความยาว 10,000 จุดข้อมูล (RW1)



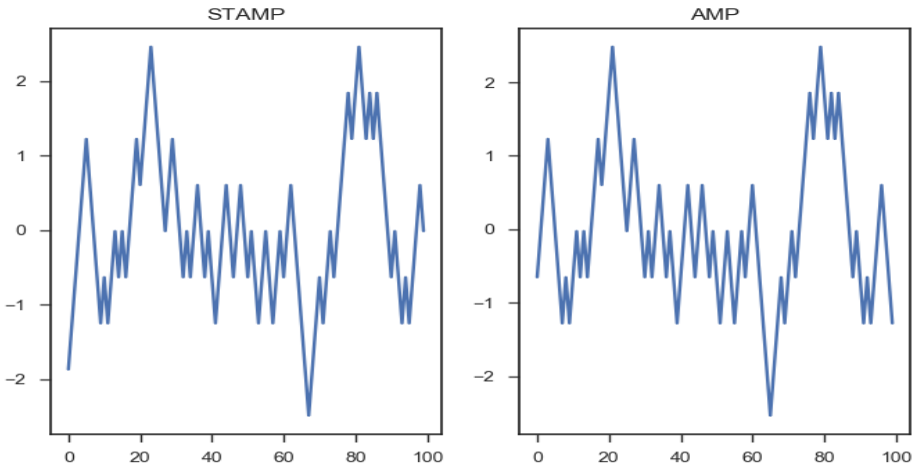
โมทีฟผลลัพธ์ที่ความยาว 100 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 827, 1823) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 826, 1825)



ดิสคอร์ดผลลัพธ์ที่ความยาว 100 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 5355) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 5357)



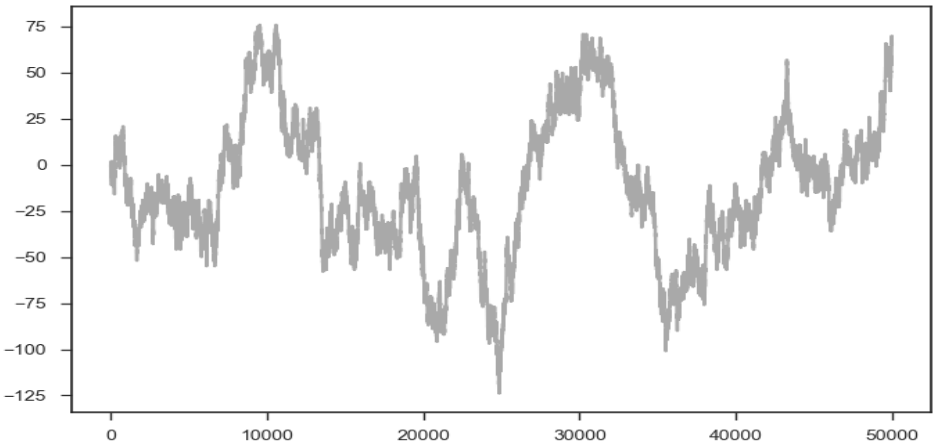
โมทีฟผลลัพธ์ที่ความยาว 100 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี



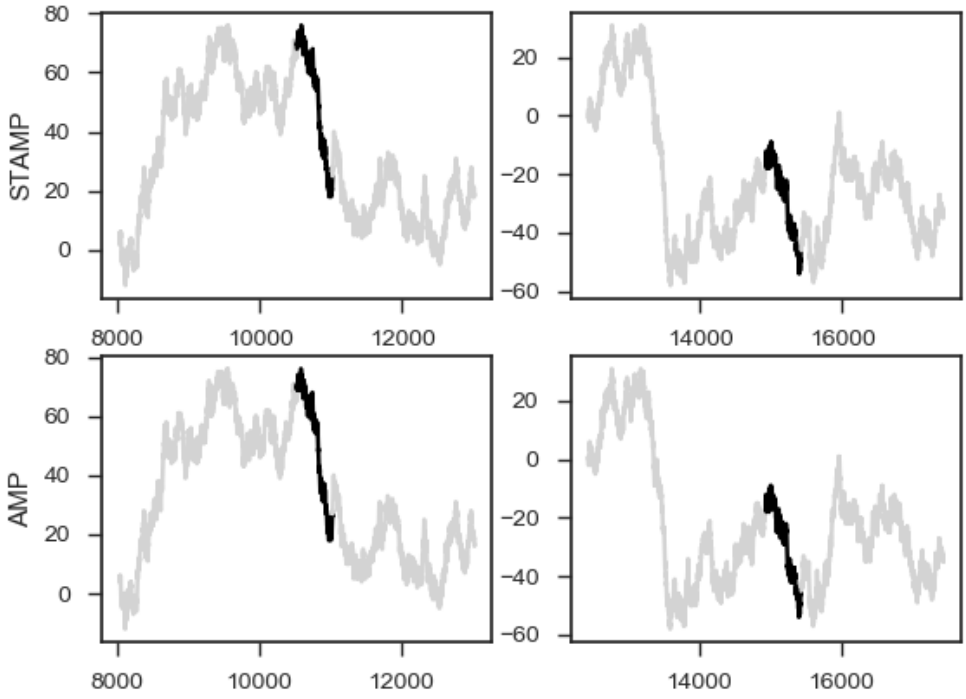
ดิสคอร์ดผลลัพธ์ที่ความยาว 100 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี

ภาพที่ ก.1 ข้อมูลสังเคราะห์ชุดที่ 1 (RW1)

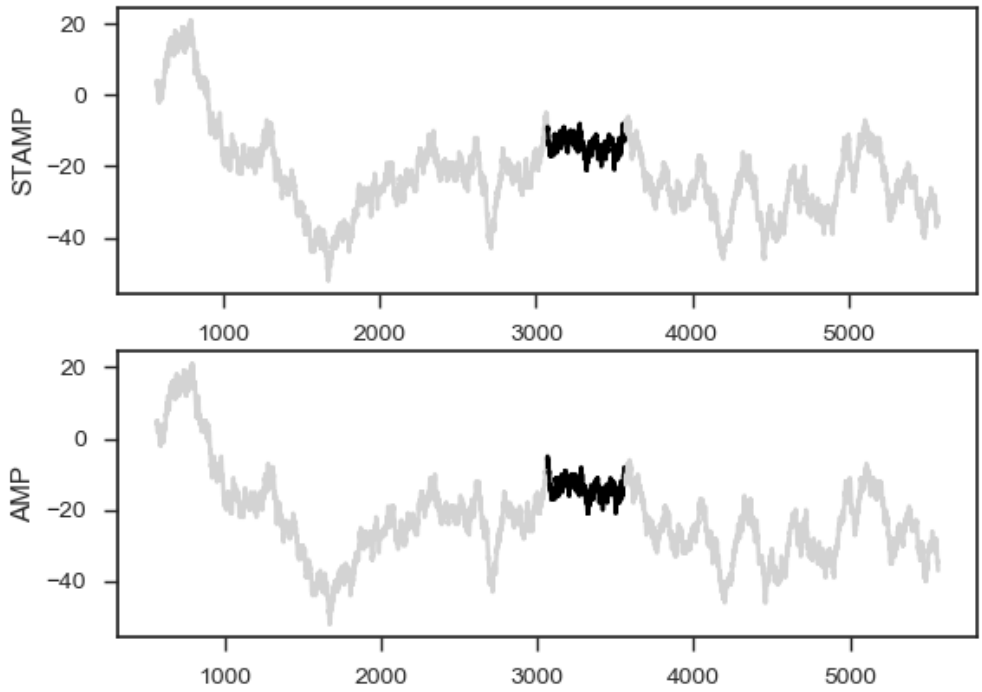
2. ข้อมูลสังเคราะห์ชุดที่ 2 มีความยาว 50,000 จุดข้อมูล (RW2)



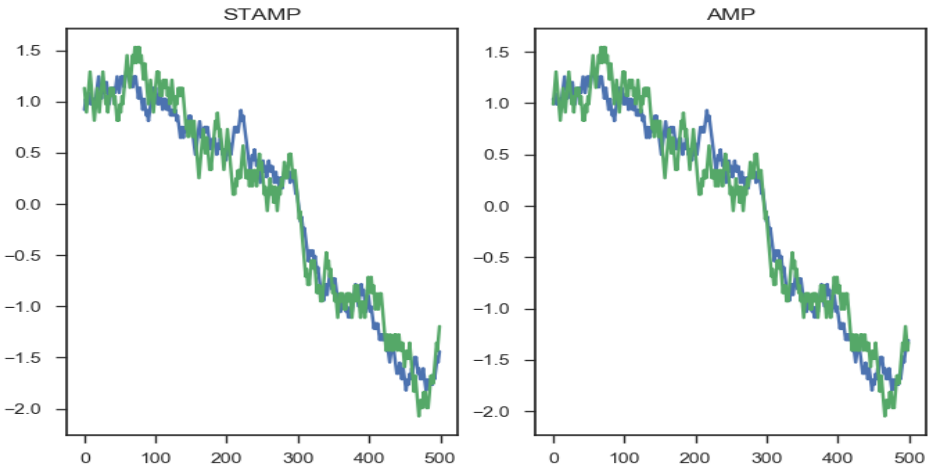
ข้อมูลสังเคราะห์ชุดที่ 2 มีความยาว 50,000 จุดข้อมูล (RW2)



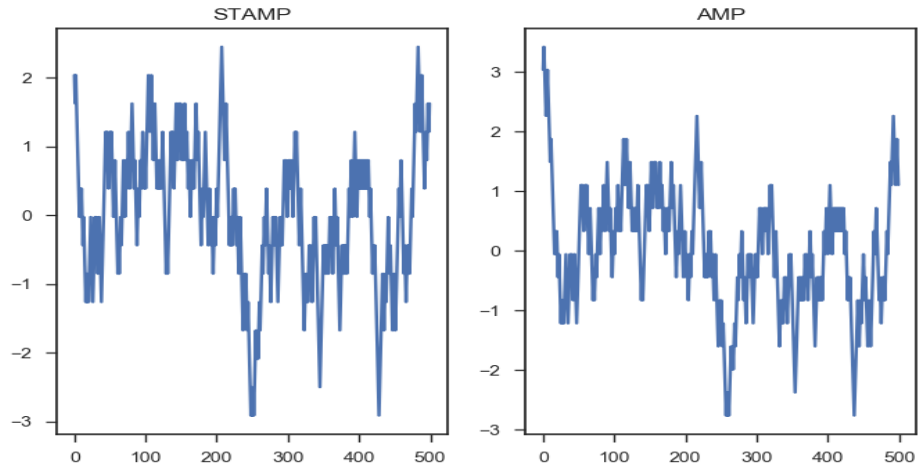
โมทีฟผลลัพธ์ที่ความยาว 500 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 10522, 14926) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 10526, 14930)



ดิสคอร์ดผลลัพธ์ที่ความยาว 500 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 3073) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 3064)



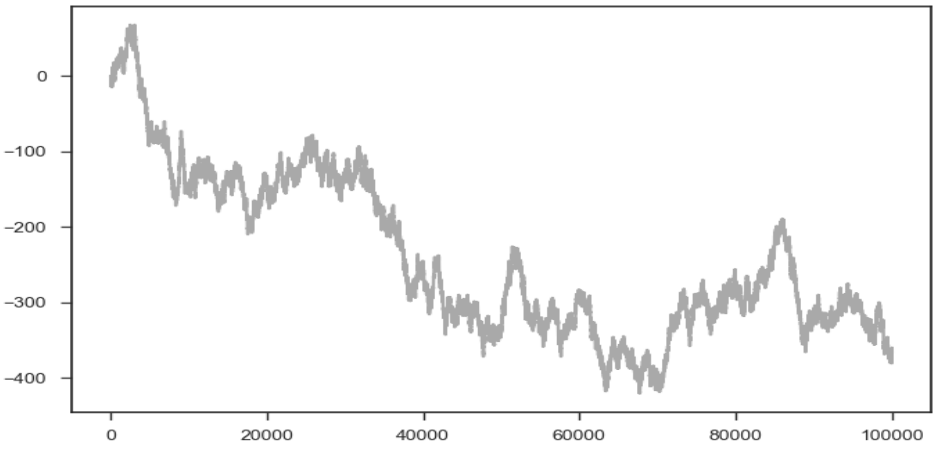
โมทีฟผลลัพธ์ที่ความยาว 500 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี



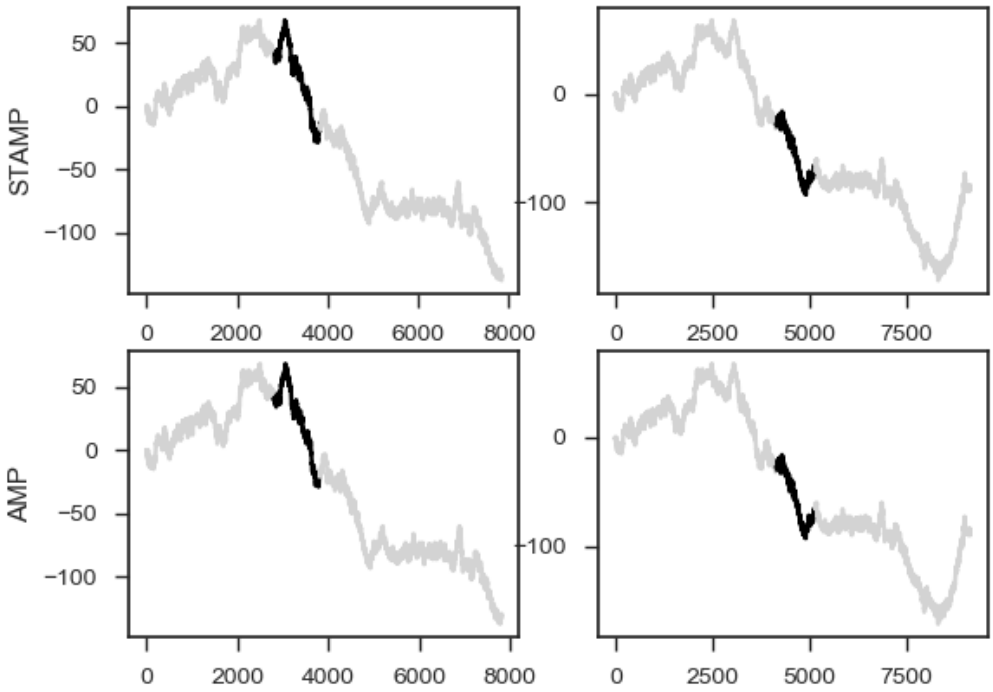
ดิสคอร์ดผลลัพธ์ที่ความยาว 500 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี

ภาพที่ ก.2 ข้อมูลสังเคราะห์ชุดที่ 2 (RW2)

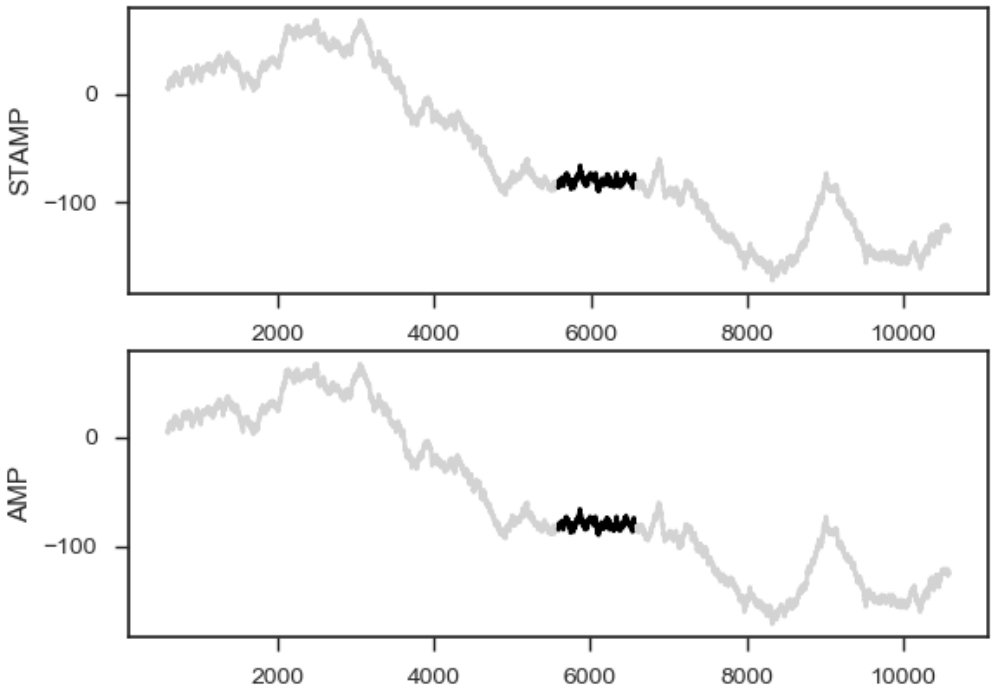
3. ชุดข้อมูลสังเคราะห์ชุดที่ 3 มีความยาว 100,000 จุดข้อมูล (RW3)



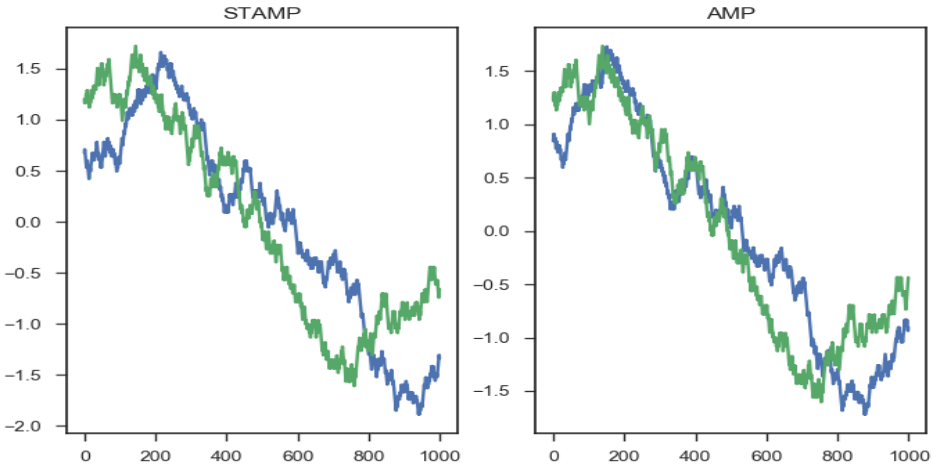
ชุดข้อมูลสังเคราะห์ชุดที่ 3 มีความยาว 100,000 จุดข้อมูล (RW3)



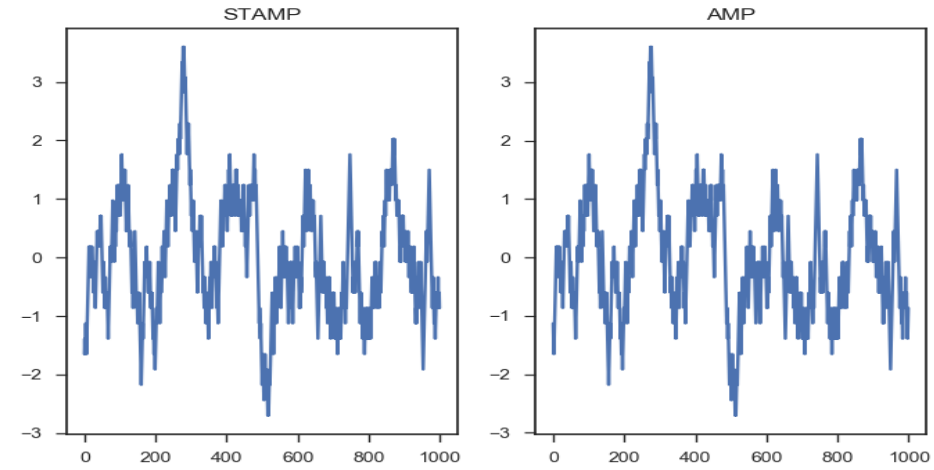
โมทีฟผลลัพธ์ที่ความยาว 1000 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 2839, 4160) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 2903, 4165)



ดิสคอร์ดผลลัพธ์ที่ความยาว 1000 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 5590) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 5594)



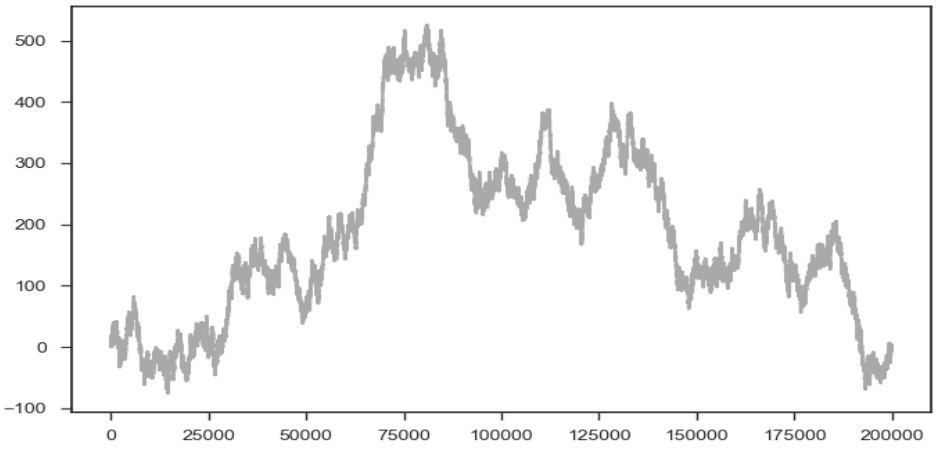
โมทีฟผลลัพธ์ที่ความยาว 1000 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี



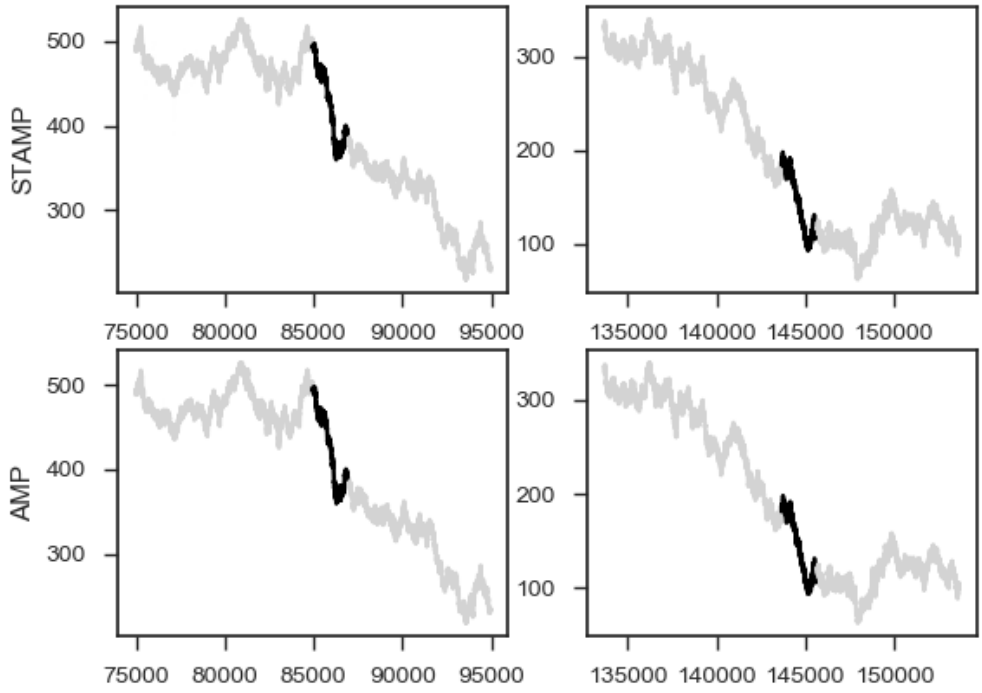
ดิสคอร์ดผลลัพธ์ที่ความยาว 1000 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี

ภาพที่ ก.3 ข้อมูลสังเคราะห์ชุดที่ 3 (RW3)

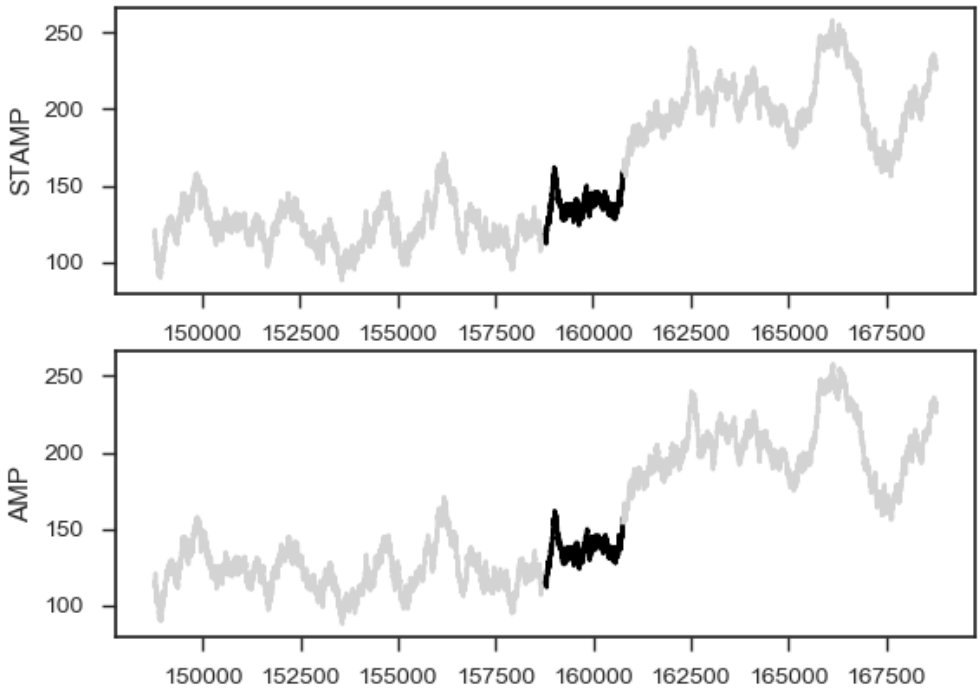
4. ชุดข้อมูลสังเคราะห์ชุดที่ 4 มีความยาว 200,000 จุดข้อมูล (RW4)



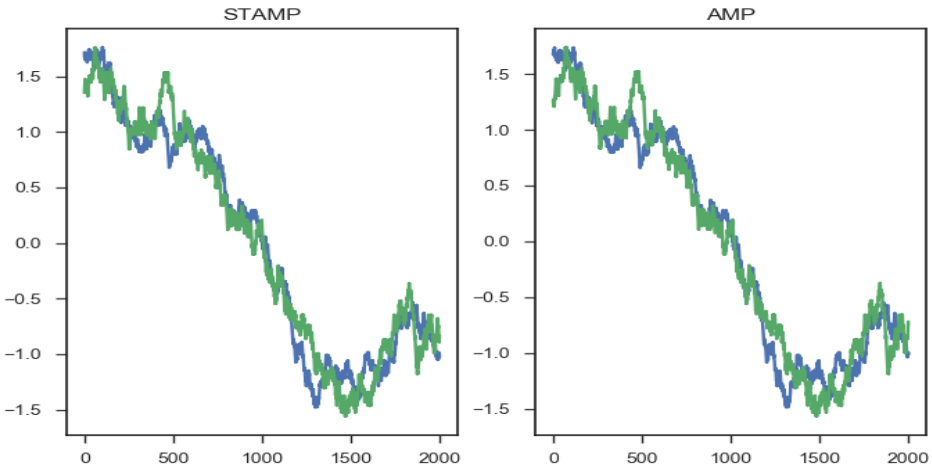
ชุดข้อมูลสังเคราะห์ชุดที่ 4 มีความยาว 200,000 จุดข้อมูล (RW4)



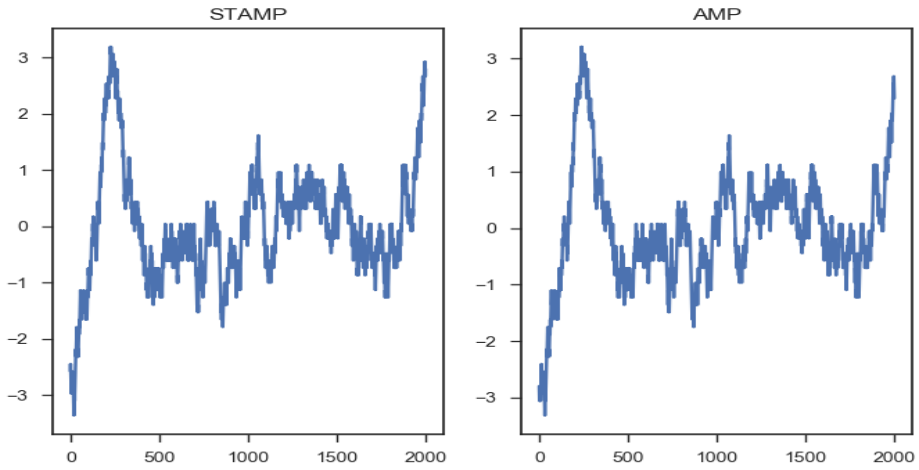
โมทีฟผลลัพธ์ที่ความยาว 2000 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 84963, 143681) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 84951, 143669)



ดิสคอร์ดผลลัพธ์ที่ความยาว 2000 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 158784) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 158772)



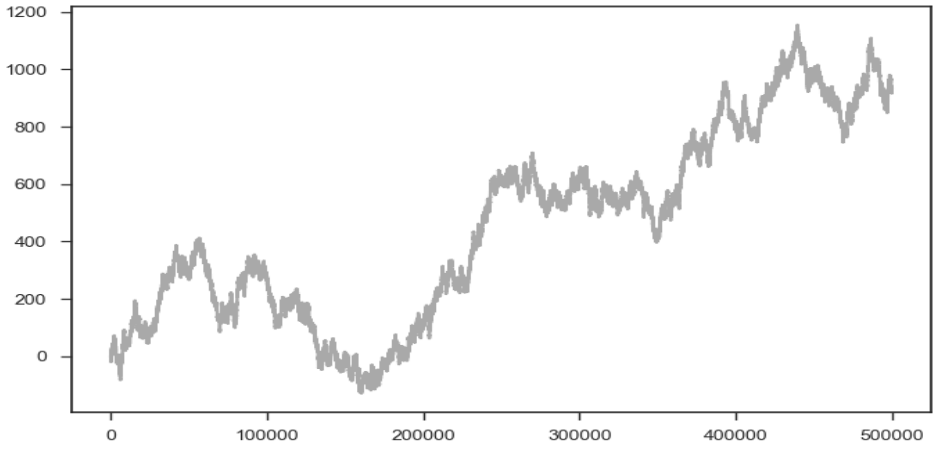
โมทีฟผลลัพธ์ที่ความยาว 2000 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี



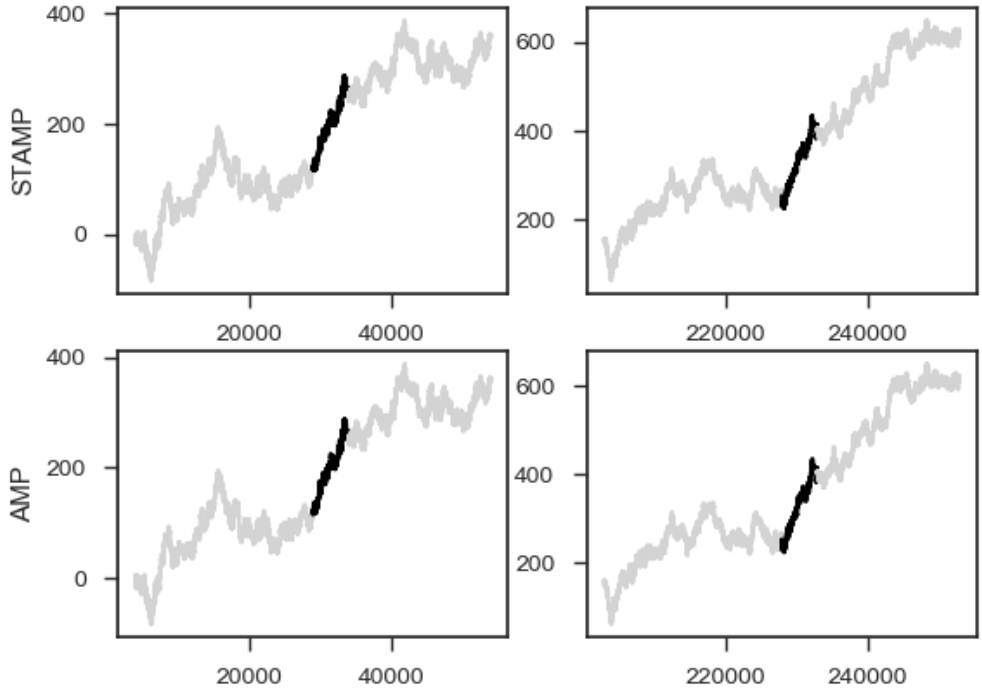
ดิสคอร์ดผลลัพธ์ที่ความยาว 2000 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี

ภาพที่ ก.4 ข้อมูลสังเคราะห์ชุดที่ 4 (RW4)

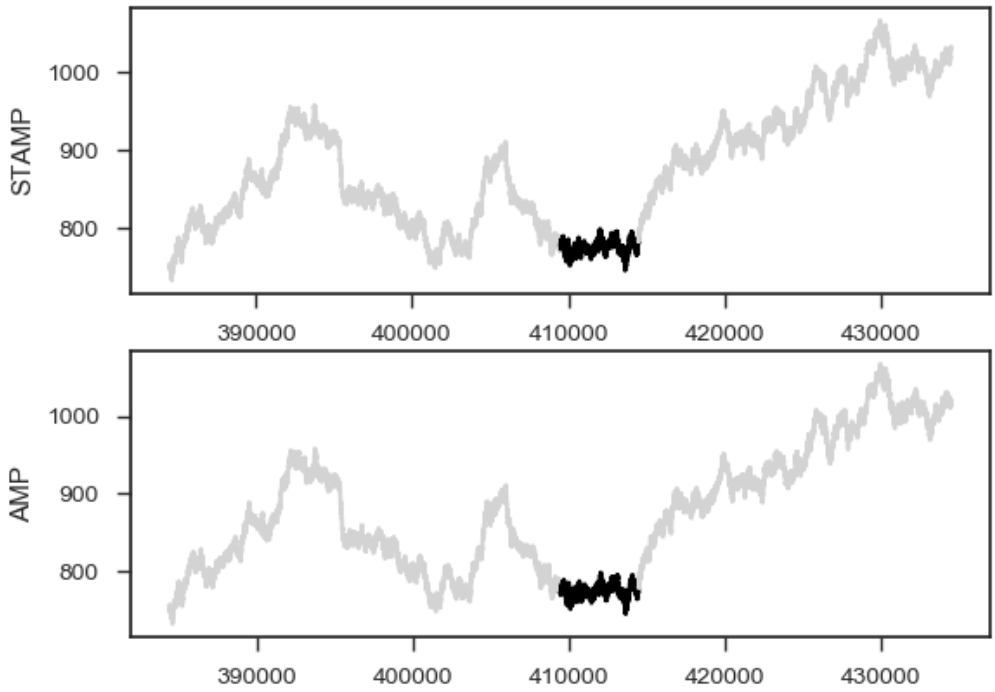
5. ชุดข้อมูลสังเคราะห์ชุดที่ 5 มีความยาว 500,000 จุดข้อมูล (RW5)



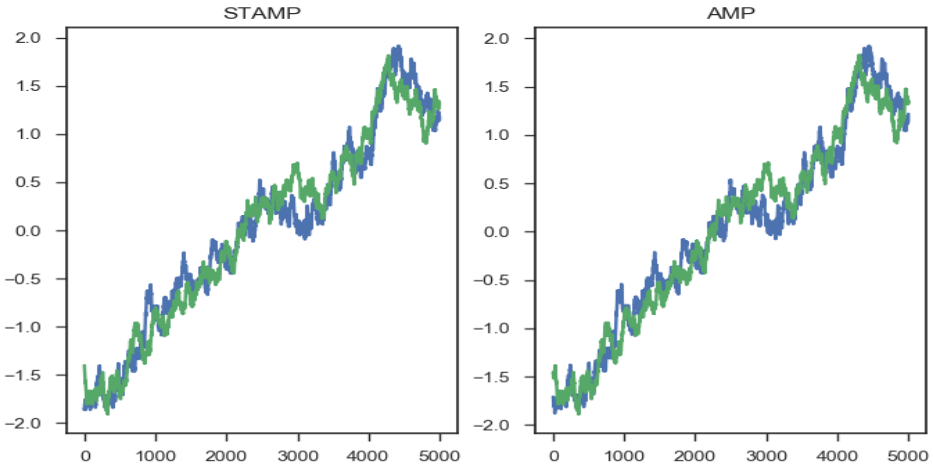
ชุดข้อมูลสังเคราะห์ชุดที่ 5 มีความยาว 500,000 จุดข้อมูล (RW5)



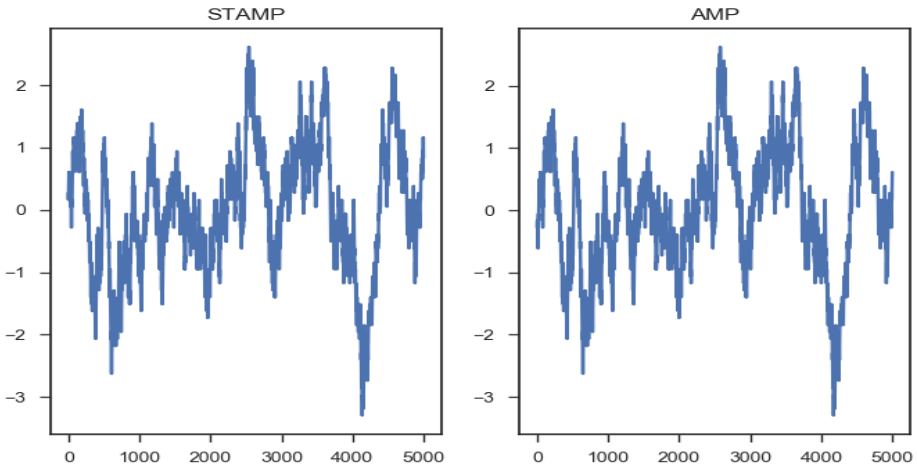
โมทีฟผลลัพธ์ที่ความยาว 5000 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 28984, 227945) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 28957, 227916)



ดิสคอร์ดผลลัพธ์ที่ความยาว 5000 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 409492) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 409456)



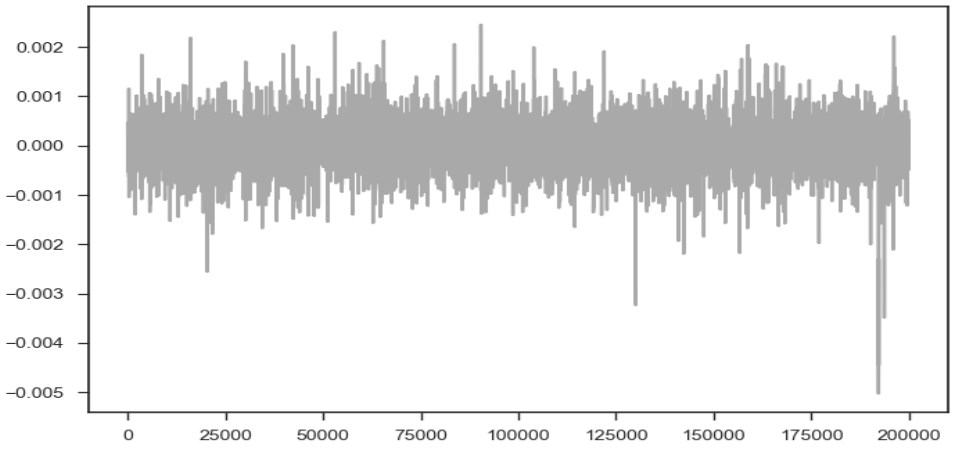
โมทีฟผลลัพธ์ที่ความยาว 5000 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี



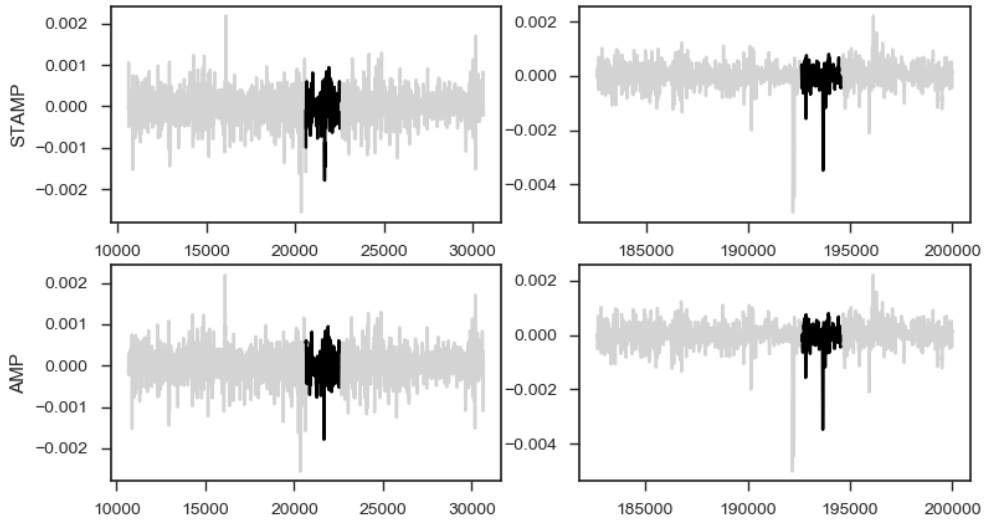
ดิสคอร์ดผลลัพธ์ที่ความยาว 5000 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี

ภาพที่ ก.5 ข้อมูลสังเคราะห์ชุดที่ 5 (RW5)

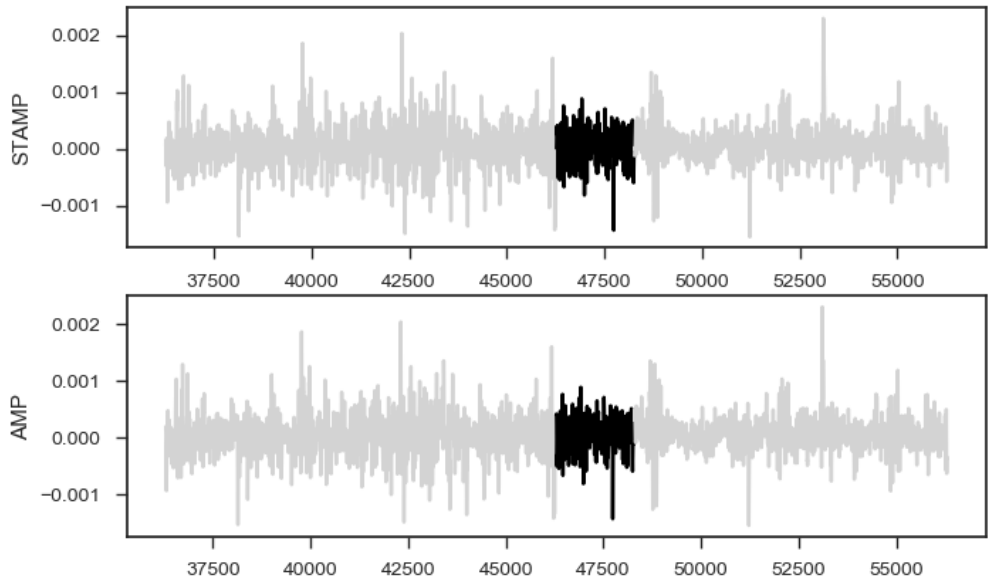
6. ชุดข้อมูลจริง Astro มีความยาว 200,000 จุดข้อมูล



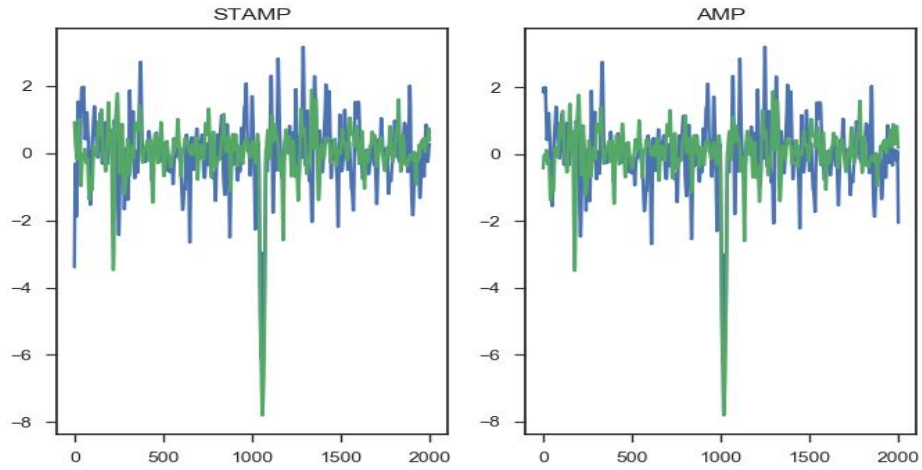
ชุดข้อมูลจริง Astro มีความยาว 200,000 จุดข้อมูล



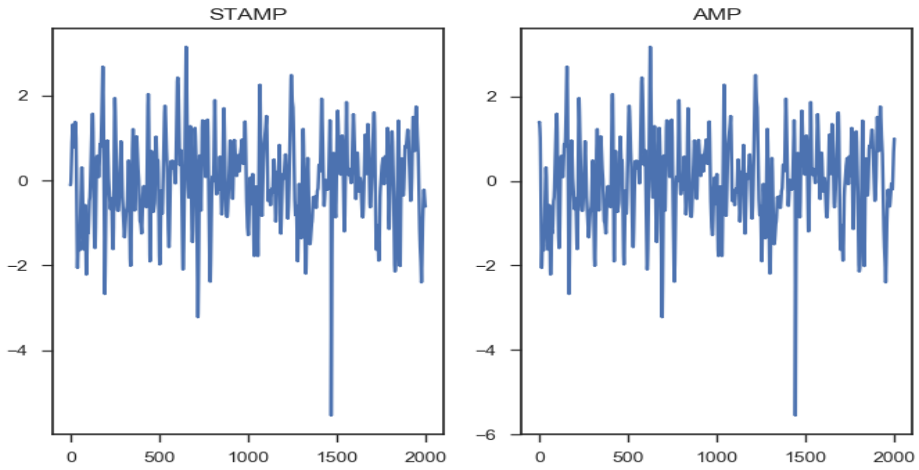
โมทีฟผลลัพธ์ที่ความยาว 2000 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 20623, 192615) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 20663, 192657)



ดิสคอร์ดผลลัพธ์ที่ความยาว 2000 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 46269) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 46295)



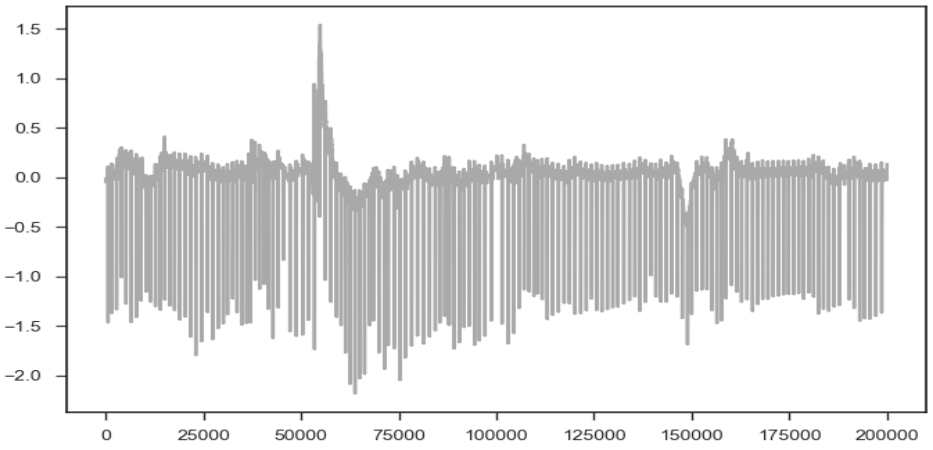
โมทีฟผลลัพธ์ที่ความยาว 2000 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี



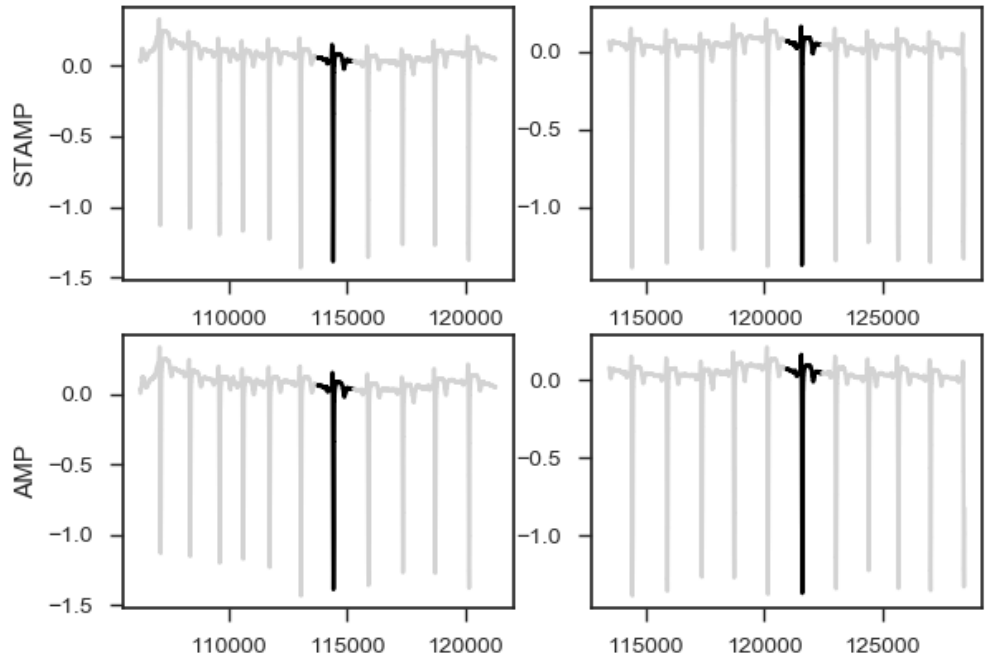
ดิสคอร์ดผลลัพธ์ที่ความยาว 2000 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี

ภาพที่ ก.6 ชุดข้อมูลจริง Astro

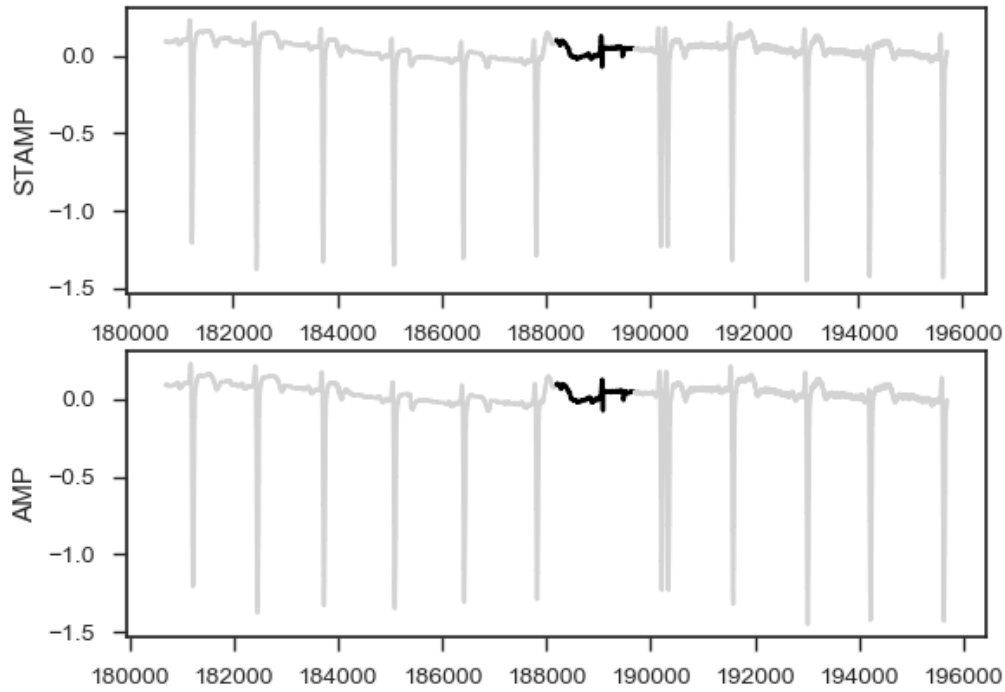
7. ชุดข้อมูลจริง ECG มีความยาว 200,000 จุดข้อมูล



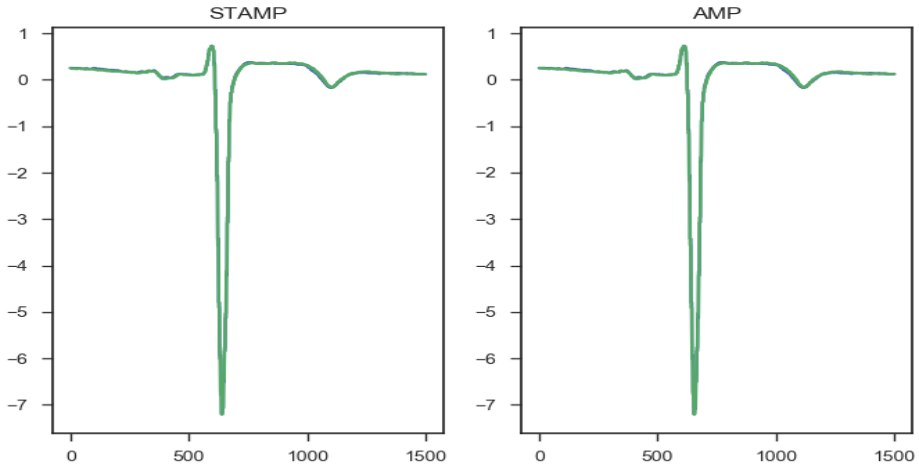
ชุดข้อมูลจริง ECG มีความยาว 200,000 จุดข้อมูล



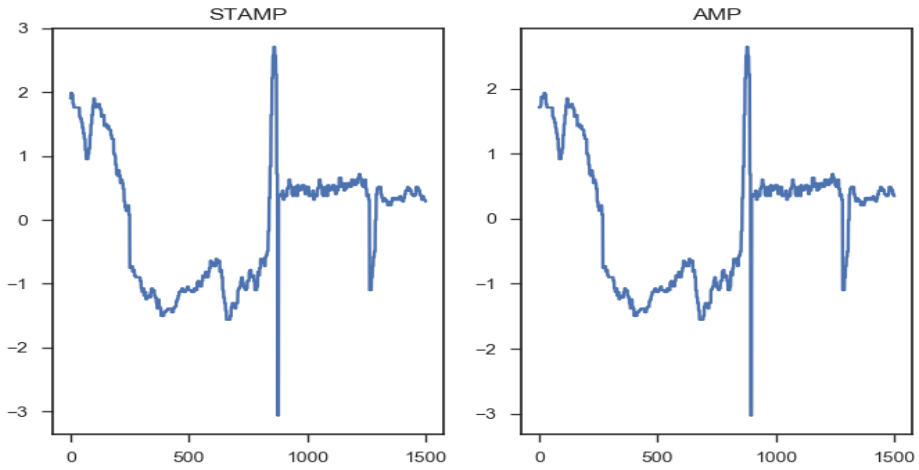
โมทีฟผลลัพธ์ที่ความยาว 1500 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 113785, 120961) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 113770, 120946)



ดิสคอร์ดผลลัพธ์ที่ความยาว 1500 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 188208) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 188190)



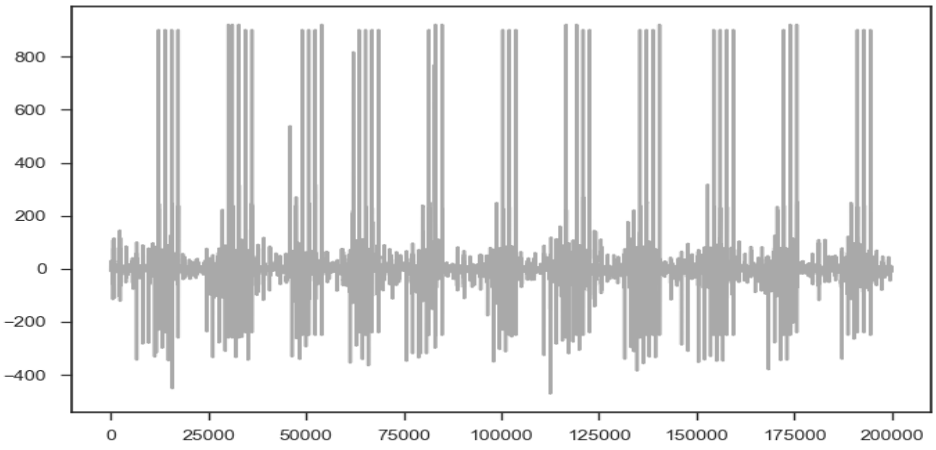
โมทีฟผลลัพธ์ที่ความยาว 1500 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี



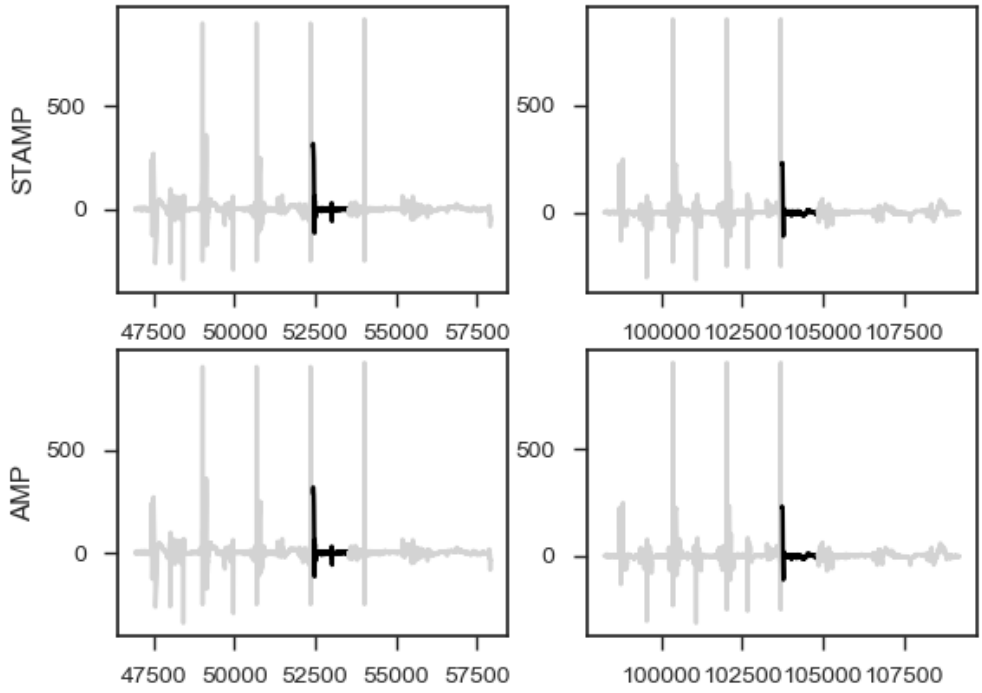
ดิสคอร์ดผลลัพธ์ที่ความยาว 1500 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี

ภาพที่ ก.7 ชุดข้อมูลจริง ECG

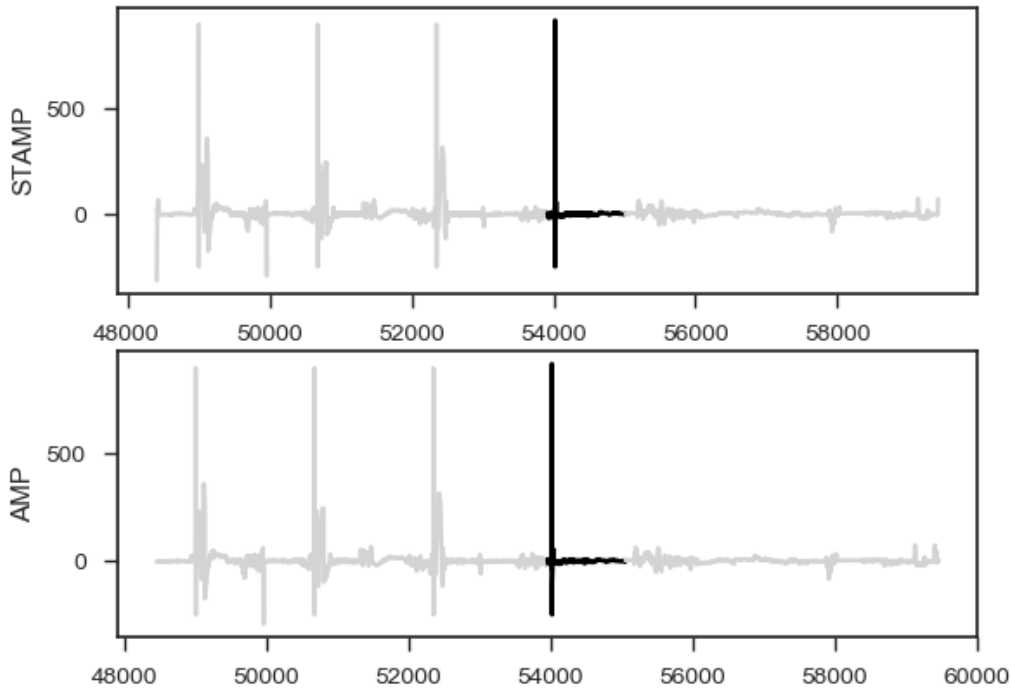
8. ชุดข้อมูลจริง EEG มีความยาว 200,000 จุดข้อมูล



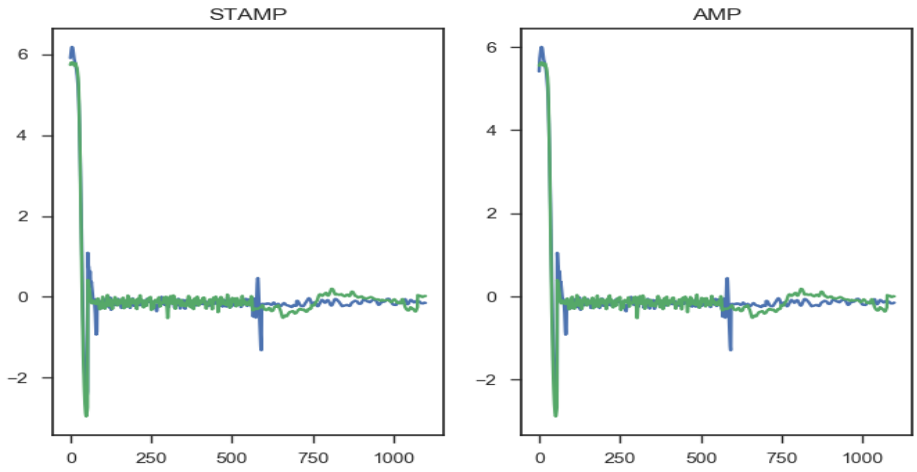
ชุดข้อมูลจริง EEG มีความยาว 200,000 จุดข้อมูล



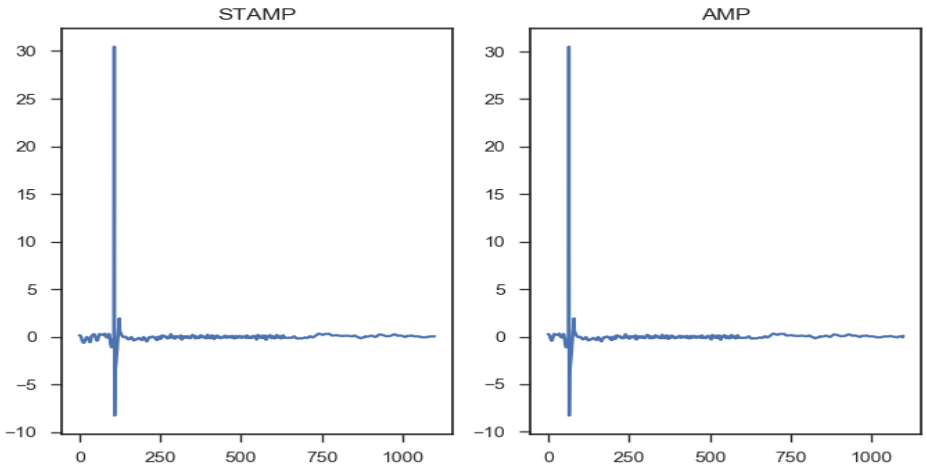
โมทีฟผลลัพธ์ที่ความยาว 1100 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 52430, 103726) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 52428, 103724)



ดิสคอร์ดผลลัพธ์ที่ความยาว 1100 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 53915) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 53959)



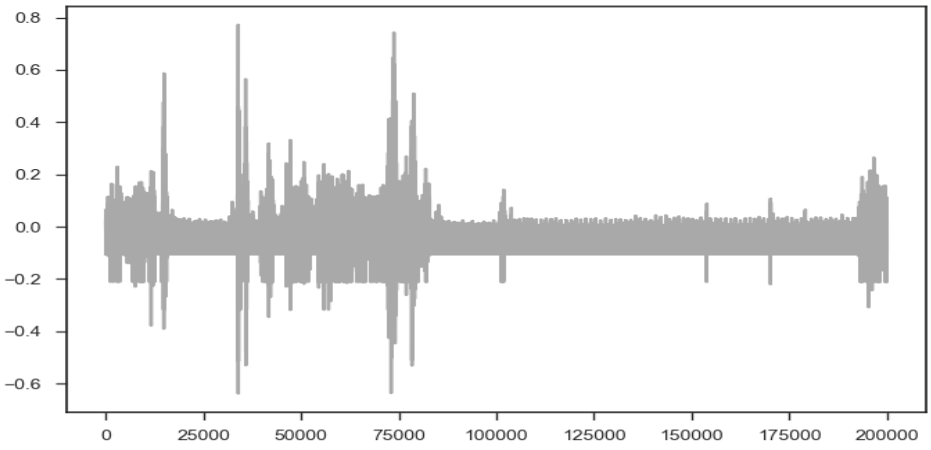
โมทีฟผลลัพธ์ที่ความยาว 1100 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี



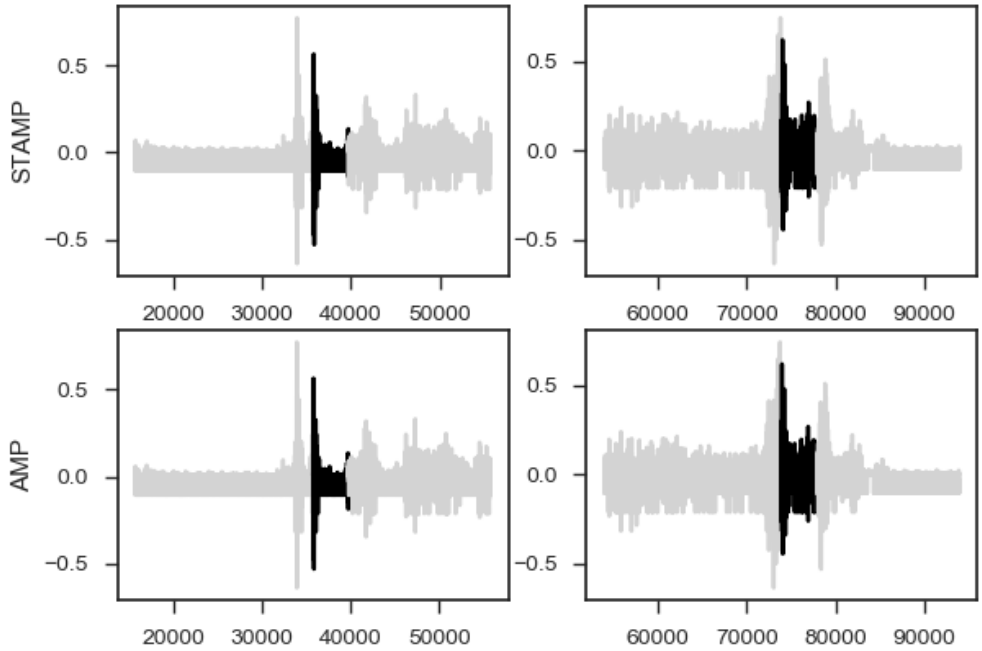
ดิสคอร์ดผลลัพธ์ที่ความยาว 1100 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี

ภาพที่ ก.8 ชุดข้อมูลจริง EEG

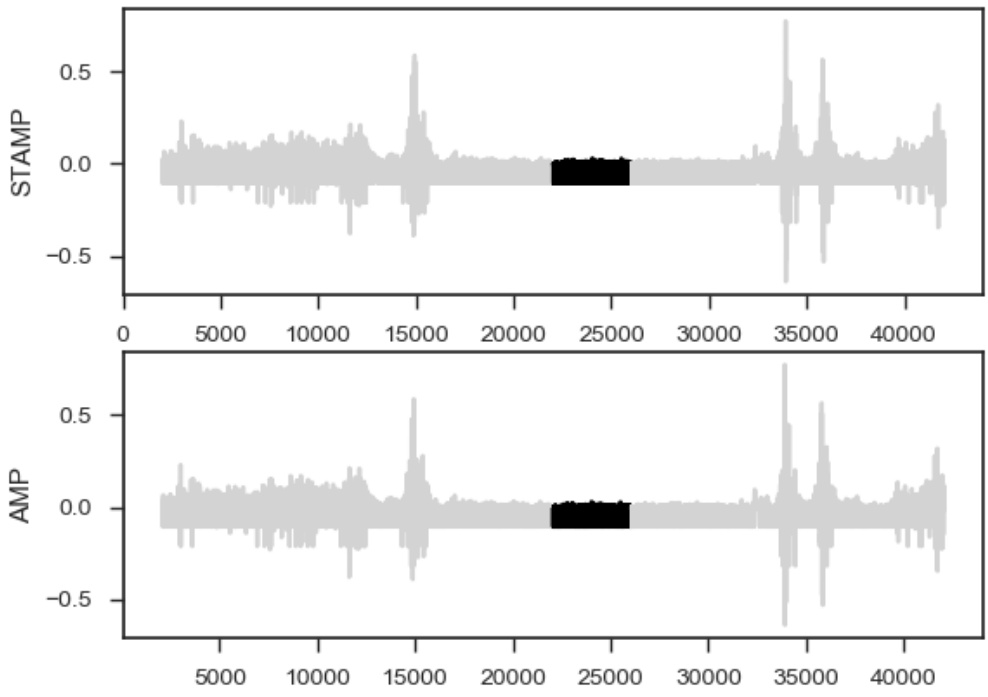
9. ชุดข้อมูลจริง EMG มีความยาว 200,000 จุดข้อมูล



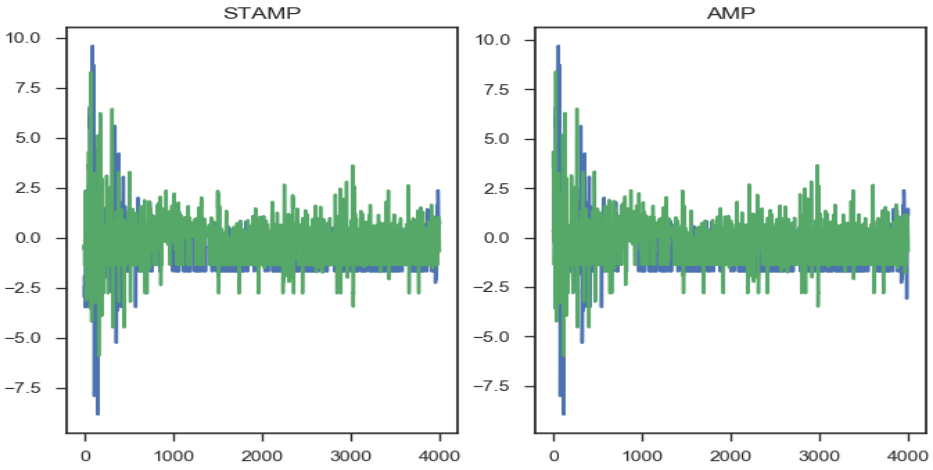
ชุดข้อมูลจริง EMG มีความยาว 200,000 จุดข้อมูล



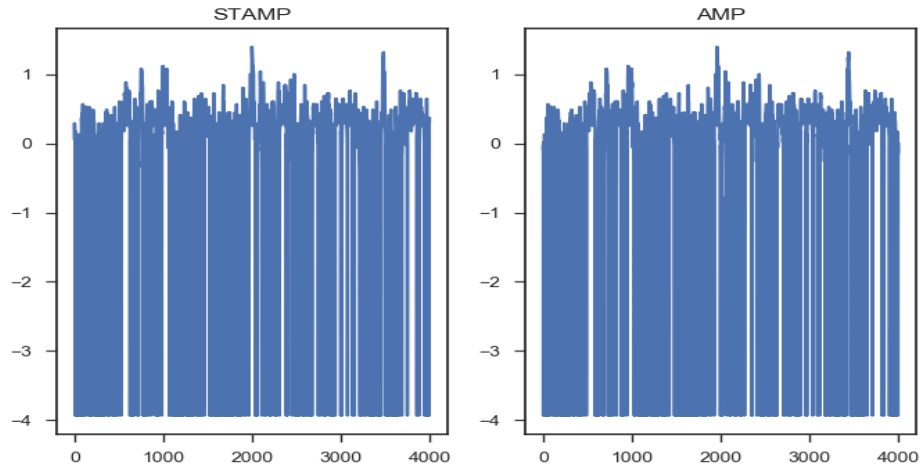
โมทีฟผลลัพธ์ที่ความยาว 4000 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 35723, 73921) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 35756, 73967)



ดิสคอร์ดผลลัพธ์ที่ความยาว 4000 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 53915) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 53959)



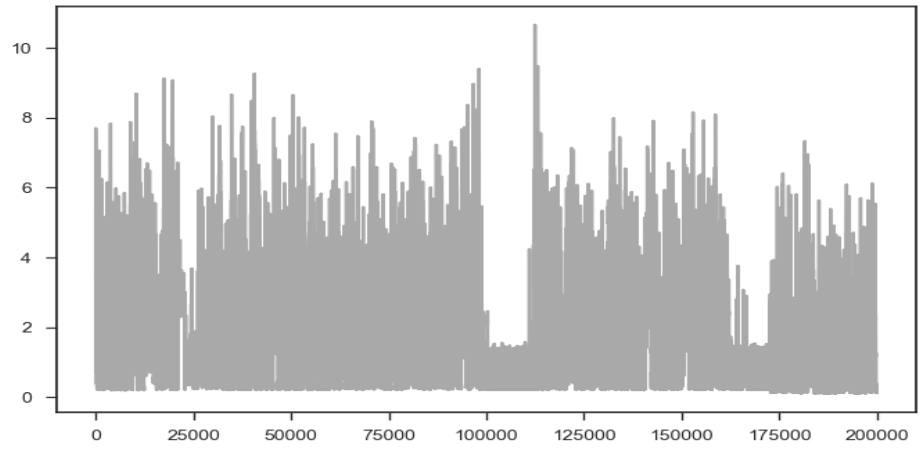
โมทีฟผลลัพธ์ที่ความยาว 4000 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี



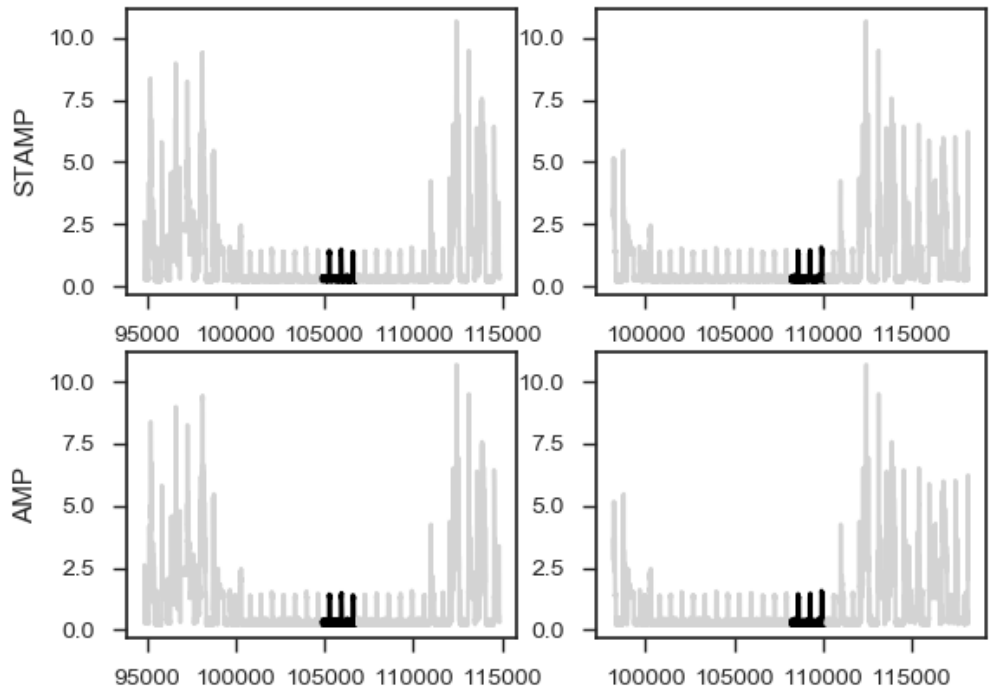
ดิสคอร์ดผลลัพธ์ที่ความยาว 4000 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี

ภาพที่ ก.9 ชุดข้อมูลจริง EMG

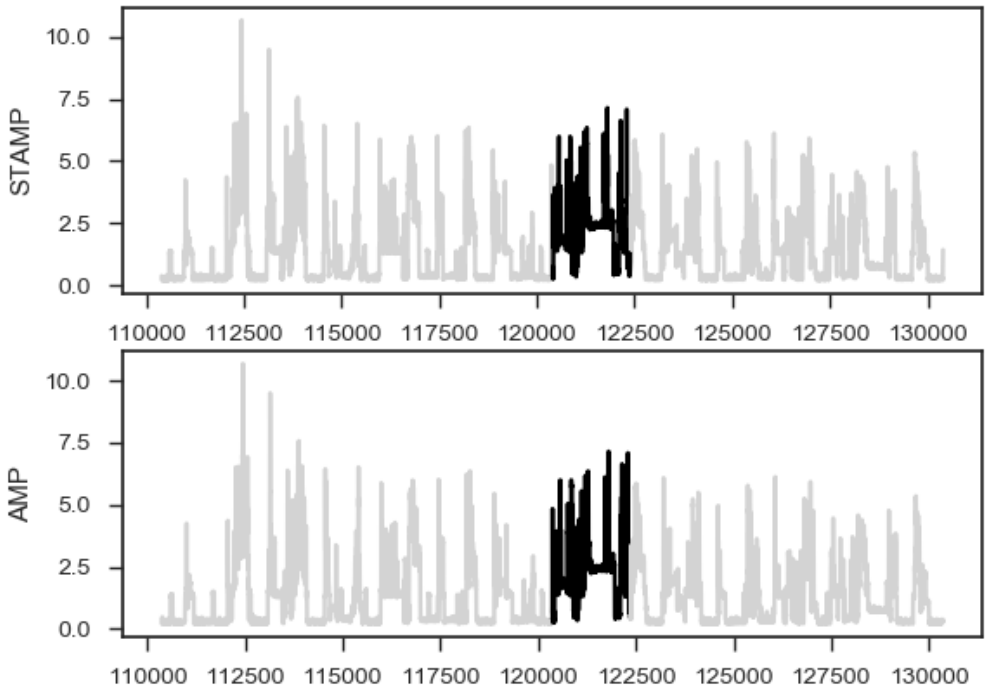
10. ชุดข้อมูลจริง GAP มีความยาว 200,000 จุดข้อมูล



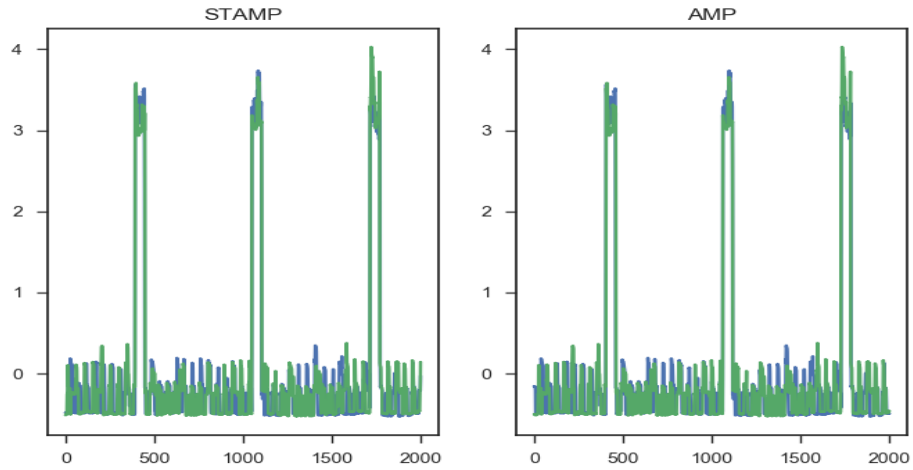
ชุดข้อมูลจริง GAP มีความยาว 200,000 จุดข้อมูล



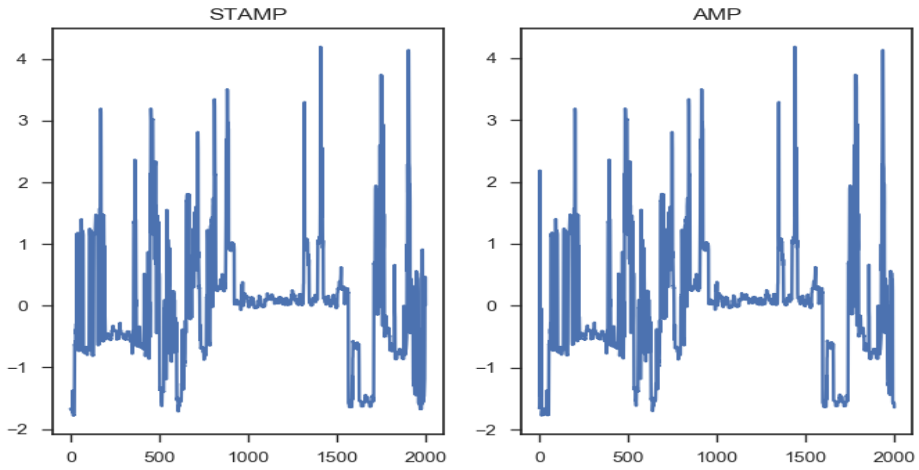
โมทีฟผลลัพธ์ที่ความยาว 2000 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 104873, 108200) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 104861, 108188)



ดิสคอร์ดผลลัพธ์ที่ความยาว 2000 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 120395) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 120364)



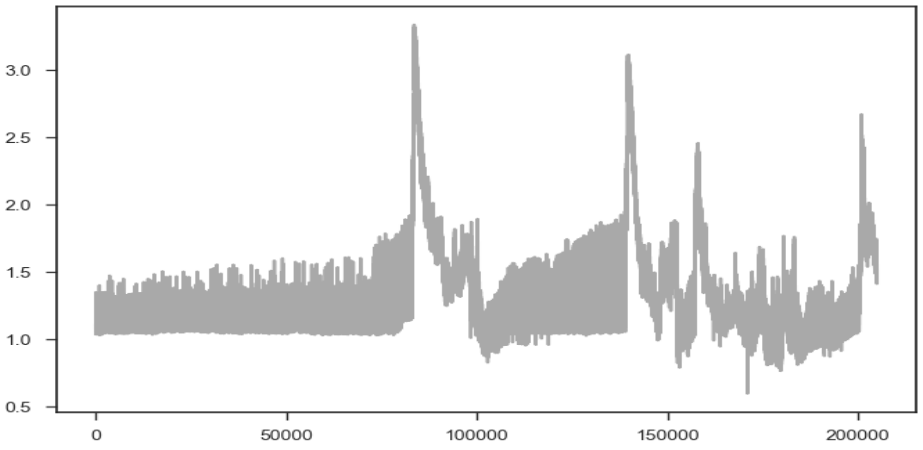
โมทีฟผลลัพธ์ที่ความยาว 2000 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี



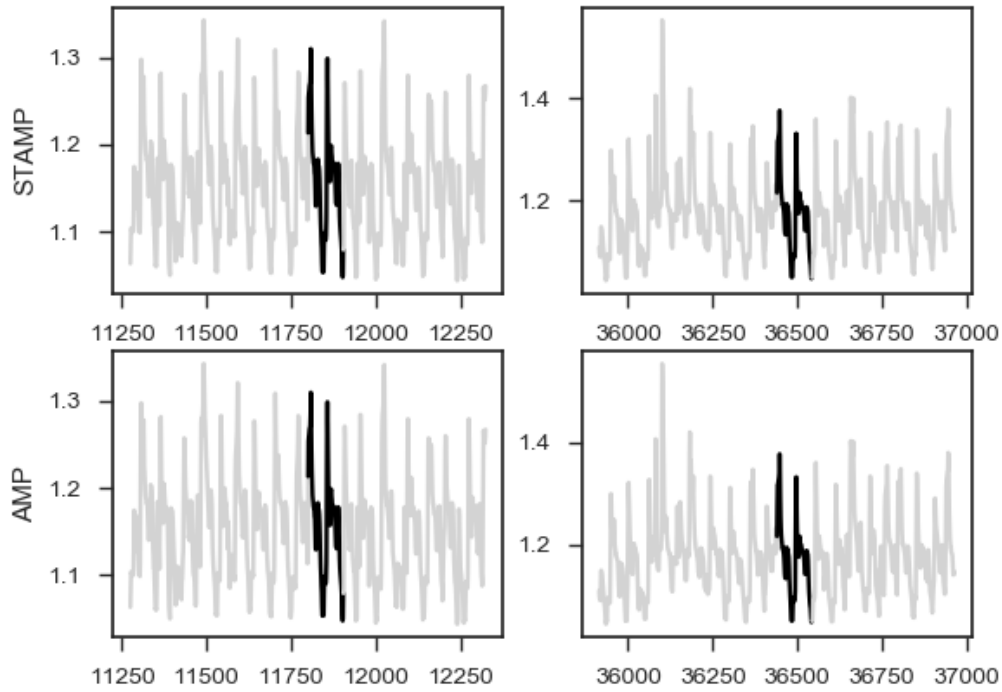
ดิสคอร์ดผลลัพธ์ที่ความยาว 2000 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี

ภาพที่ ก.10 ชุดข้อมูลจริง GAP

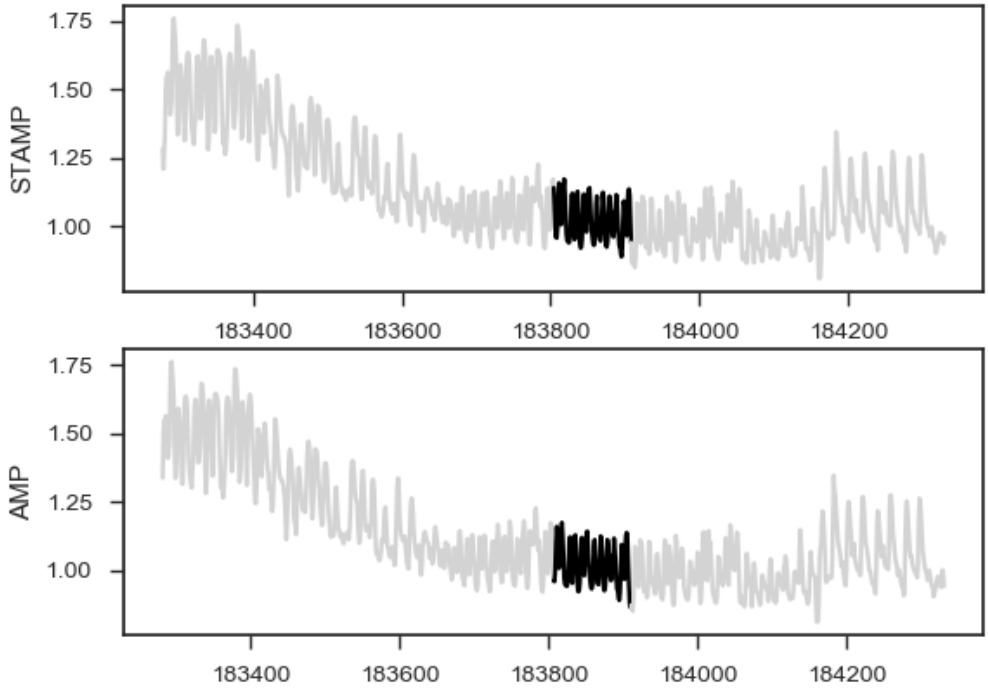
11. ชุดข้อมูลจริง Insect มีความยาว 205,000 จุดข้อมูล



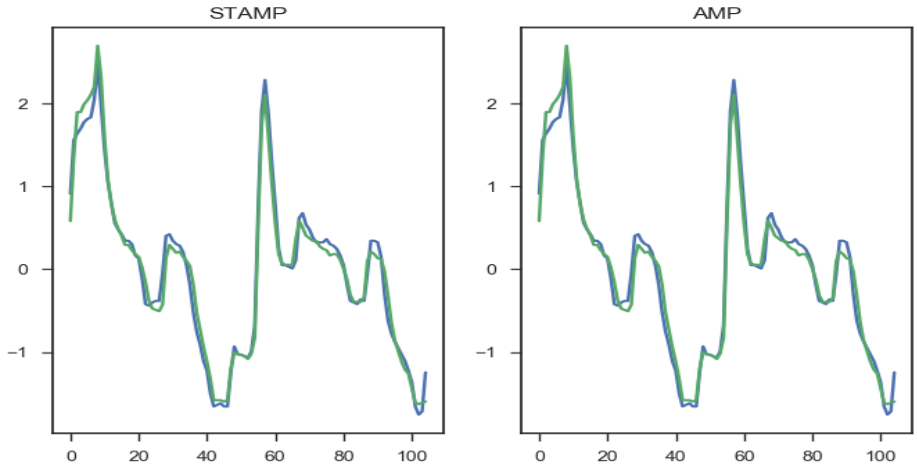
ชุดข้อมูลจริง Insect มีความยาว 205,000 จุดข้อมูล



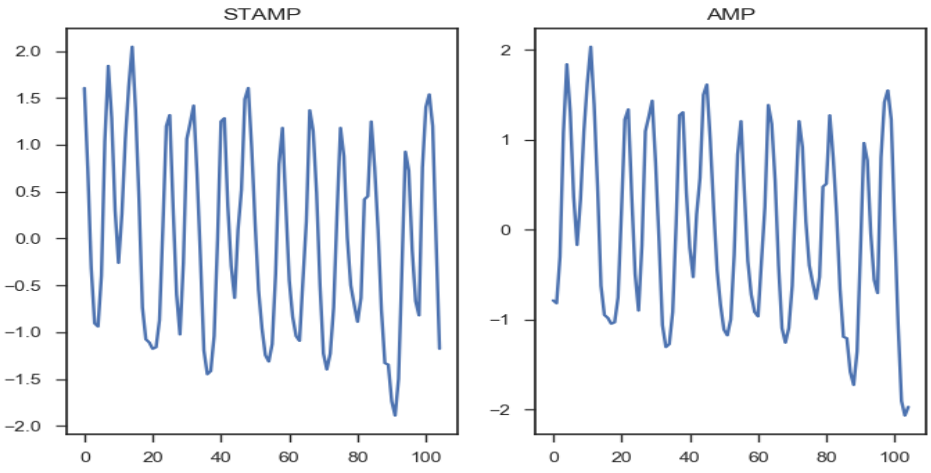
โมทีฟผลลัพธ์ที่ความยาว 105 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 11800, 36440) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 11800, 36440)



ดิสคอร์ดผลลัพธ์ที่ความยาว 105 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 183804) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 183807)



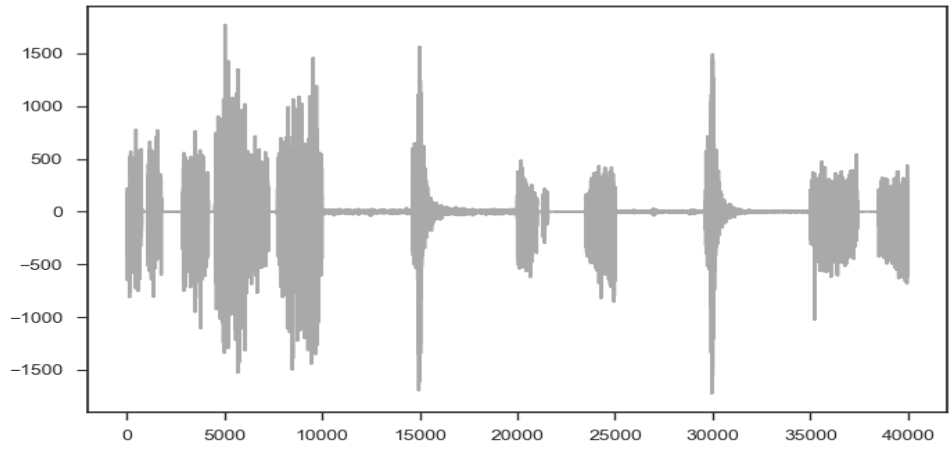
โมทีฟผลลัพธ์ที่ความยาว 105 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี



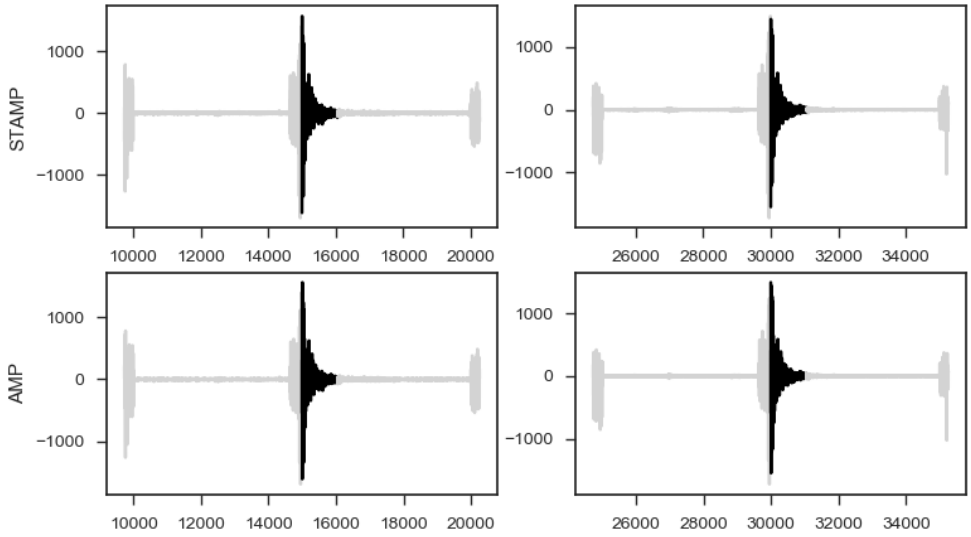
ดิสคอร์ดผลลัพธ์ที่ความยาว 105 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี

ภาพที่ ก.11 ชุดข้อมูลจริง Insect

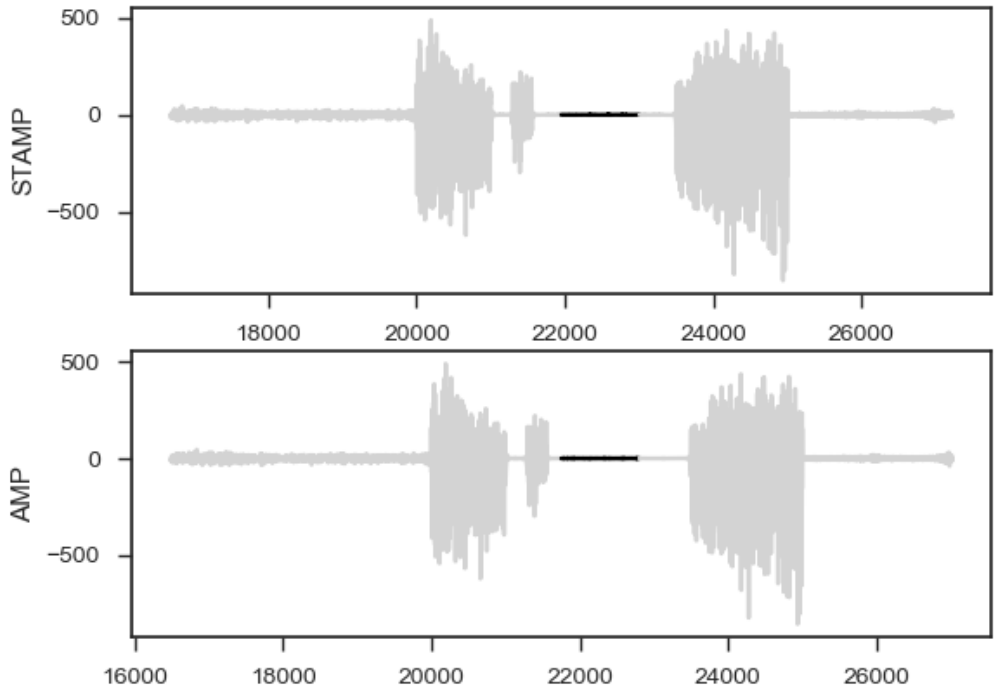
12. ชุดข้อมูลจริง Seismic มีความยาว 40,000 จุดข้อมูล



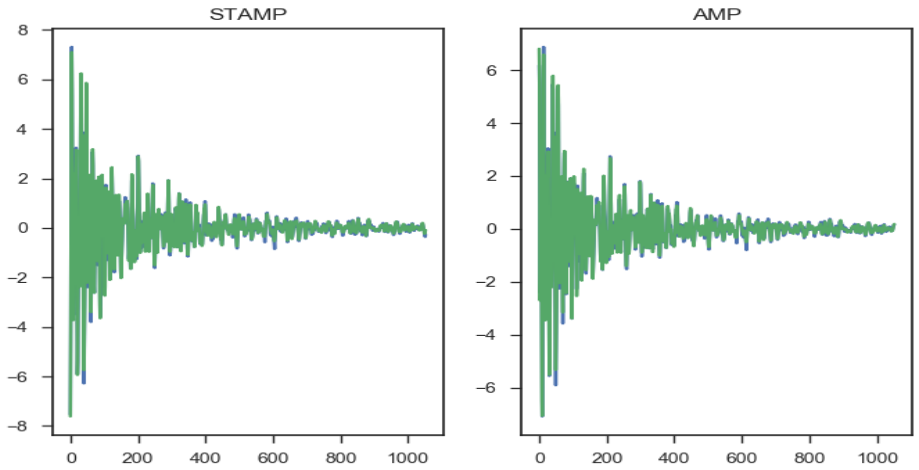
ชุดข้อมูลจริง Seismic มีความยาว 40,000 จุดข้อมูล



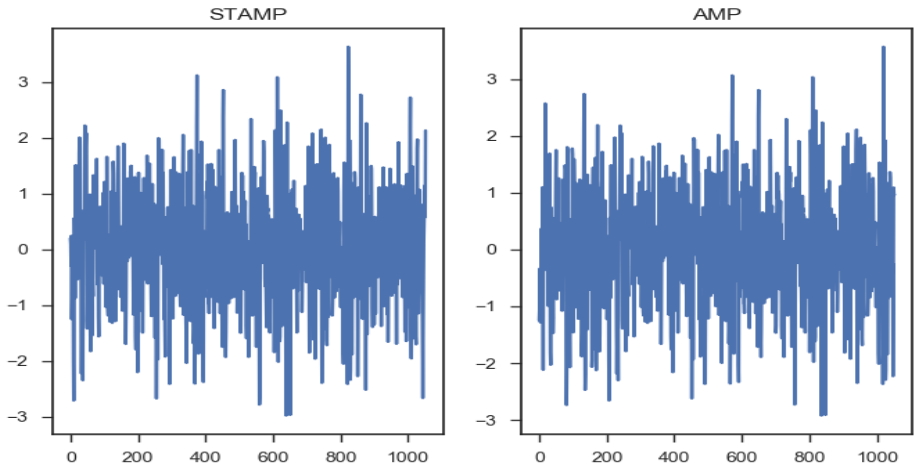
โมทีฟผลลัพธ์ที่ความยาว 1053 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 15001, 30001) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 14992, 29992)



ดิสคอร์ดผลลัพธ์ที่ความยาว 1053 จุดข้อมูลจากอัลกอริทึมแสตมป์ (ตำแหน่ง 21951) และอัลกอริทึมเอเอ็มพี (ตำแหน่ง 21754)



โมทีฟผลลัพธ์ที่ความยาว 1053 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี



ดิสคอร์ดผลลัพธ์ที่ความยาว 1053 จุดข้อมูลจากอัลกอริทึมแสตมป์และอัลกอริทึมเอเอ็มพี

ภาพที่ ก.12 ชุดข้อมูลจริง Seismic

ประวัติผู้เขียนวิทยานิพนธ์

นาย กรกฎ ปริวัฒนศักดิ์ เกิดวันที่ 31 พฤษภาคม 2536 สถานที่เกิดจังหวัดนครราชสีมา สำเร็จการศึกษาปริญญาวิทยาศาสตรบัณฑิต สาขาวิชาคณิตศาสตร์ จากภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2557 เข้าศึกษาในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2559