

การพัฒนาโปรแกรมในการหาค่าเริ่มต้นในการแก้ปัญหาค่าเหมาะสมที่สุด
ของระบบพลศาสตร์ ด้วยระเบียบวิธีเชิงตัวเลข

ปริญญานิพนธ์
ของ
ธนาคม สวาสดิพันธ์

เสนอต่อบัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ เพื่อเป็นส่วนหนึ่งของการศึกษา
ตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมเครื่องกล
พฤษภาคม 2552

การพัฒนาโปรแกรมในการหาค่าเริ่มต้นในการแก้ปัญหาค่าเหมาะสมที่สุด
ของระบบพลศาสตร์ ด้วยระเบียบวิธีเชิงตัวเลข

ปริญญานิพนธ์
ของ
ธนาคม สวาสดิพันธ์

เสนอต่อบัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ เพื่อเป็นส่วนหนึ่งของการศึกษา
ตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมเครื่องกล
พฤษภาคม 2552
ลิขสิทธิ์เป็นของมหาวิทยาลัยศรีนครินทรวิโรฒ

การพัฒนาโปรแกรมในการหาค่าเริ่มต้นในการแก้ปัญหาค่าเหมาะสมที่สุด
ของระบบพลศาสตร์ ด้วยระเบียบวิธีเชิงตัวเลข

บทคัดย่อ
ของ
ธนาคม สวาสดิพันธ์

เสนอต่อบัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ เพื่อเป็นส่วนหนึ่งของการศึกษา
ตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมเครื่องกล
พฤษภาคม 2552

ธนาคม สวาสดิพันธ์. (2552). การพัฒนาโปรแกรมในการหาค่าเริ่มต้นในการแก้ปัญหาความ

เหมาะสมสูงสุดของระบบพลศาสตร์ ด้วยระเบียบวิธีเชิงตัวเลข. ปรินูญานิพนธ์ วศ.ม.

(เครื่องกล). กรุงเทพฯ:บัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ.

คณะกรรมการควบคุม: ผู้ช่วยศาสตราจารย์ ดร.พิชัย อัมฤมงคล,

พันเอก ผู้ช่วยศาสตราจารย์ ดร.ทวิวัชร วีระเกล้า.

ปรินูญานิพนธ์นี้มีวัตถุประสงค์ที่มุ่งเน้นวิจัยเพื่อศึกษาปัญหาความเหมาะสมสูงสุดทางด้านพลศาสตร์ (Dynamic Optimization Problem) ซึ่งเป็นปัญหาความเหมาะสมสูงสุดทางด้านวิศวกรรมแขนงหนึ่ง โดยงานวิจัยนี้เป็นการศึกษาและพัฒนาเพิ่มเติมส่วนของโปรแกรมเพื่อใช้ในการแก้ปัญหาความเหมาะสมสูงสุดทางด้านพลศาสตร์ (Programming for Dynamic Optimization Problem) ซึ่งได้มีผู้ทำวิจัยไว้ก่อนหน้านี้แล้ว โดยในงานวิจัยนี้ได้มีการปรับปรุงและพัฒนาในส่วน of ค่าเริ่มต้น (Initial Guess) ให้มีความเหมาะสมก่อนที่จะนำไปใช้ในกระบวนการอื่นๆ ของโปรแกรม

ผลของงานวิจัยนี้คือ โปรแกรมดังกล่าวใช้เวลาในการหาค่าตอบจากค่าเริ่มต้นที่ถูกปรับปรุงและพัฒนาแล้ว ทั้งนี้ในกรณีที่ใช้โปรแกรมโดยวิธีตรงนั้นเวลาที่ใช้จะลดลงโดยเฉลี่ยประมาณ 66.20% และโดยวิธีอ้อมจะลดลงโดยเฉลี่ยประมาณ 15% นั้นหมายความว่า การปรับปรุงและพัฒนาค่าเริ่มต้นให้เหมาะสมก่อนที่จะนำไปใช้ในกระบวนการอื่นๆ ของโปรแกรม จะทำให้โปรแกรมห่วงการมีประสิทธิภาพสูงขึ้นกว่าที่เป็นอยู่เดิมได้

THE DEVELOPMENT OF FINDING INITIAL GUESS FOR DYNAMICS OPTIMIZATIKON
PROBLEM WITH NUMERICAL ANALYSIS

AN ABSTRACT
BY
TANAKOM SAWASDIPHAN

Presented in Partial Fulfillment of the Requirements for the
Master of Engineering Degree in Mechanical Engineering
at Srinakharinwirot University

May 2009

Tanakom Sawasdiphan. (2009). *The Development of Finding Initial Guess for Dynamics Optimization Problem with Numerical Analysis*. Master thesis, M.Eng. (Mechanical Engineering). Bangkok: Graduate School, Srinakharinwirot University. Advisor Committee: Assist. Prof. Dr. Pichai Asadamongkon, Col. Assist. Prof. Dr.Tawiwat Veeraklaew.

The purpose of this research is to study about the dynamic optimization problem that is the one part of engineering optimization problems. This research has developed for the initial guess before to be used in existing program (Programming for Dynamic Optimization Problem). It is quite obvious that the CPU-runtime used to get the solution is less than the existing program as about 15% for Direct method solving, and about 66.2% for Indirect method solving. This make the program become more effectiveness than the existing program.

ปริญญานิพนธ์

เรื่อง

การพัฒนาโปรแกรมในการหาค่าเริ่มต้นในการแก้ปัญหาค่าความเหมาะสมสูงสุด
ของระบบพลศาสตร์ ด้วยระเบียบวิธีเชิงตัวเลข

ของ

นายธนาคม สวาสดิพันธ์

ได้รับอนุมัติจากบัณฑิตวิทยาลัยให้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมเครื่องกล
ของมหาวิทยาลัยศรีนครินทรวิโรฒ

.....คณบดีบัณฑิตวิทยาลัย

(รองศาสตราจารย์ ดร.สมชาย สันติวัฒนกุล)

วันที่ เดือน.....พ.ศ. 2552

คณะกรรมการควบคุมปริญญานิพนธ์

คณะกรรมการสอบปากเปล่า

.....ประธาน

.....ประธาน

(ผู้ช่วยศาสตราจารย์ ดร.พิชัย อัมมมงคล)

(พันเอก ผู้ช่วยศาสตราจารย์ ดร.อโณทัย สุขแสงพนมรุ้ง)

.....กรรมการ

.....กรรมการ

(พันเอก ผู้ช่วยศาสตราจารย์ ดร.ทวิวัชร วีระแก้ว)

(ผู้ช่วยศาสตราจารย์ ดร.จุฑารัตน์ คุรุเจริญ)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.พิชัย อัมมมงคล)

.....กรรมการ

(พันเอก ผู้ช่วยศาสตราจารย์ ดร.ทวิวัชร วีระแก้ว)

ประกาศคุณูปการ

ปริญญานิพนธ์ฉบับนี้ สำเร็จได้ด้วยความกรุณาของ ผู้ช่วยศาสตราจารย์ ดร.พิชัย อัมฤมมงคล ประธานกรรมการควบคุมการทำปริญญานิพนธ์ และพันเอก ผู้ช่วยศาสตราจารย์ ดร.ทวิวัชร วีระแก้วกรรมการควบคุมการทำปริญญานิพนธ์ ที่กรุณาให้คำปรึกษา แนะนำ ให้ความช่วยเหลือ และแก้ไขความบกพร่อง อีกทั้งให้กำลังใจขณะดำเนินการทำปริญญานิพนธ์ ผู้วิจัยขอขอบพระคุณเป็นอย่างสูงตลอดไป

ขอขอบพระคุณ คณาจารย์จากทางคณะวิศวกรรมศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ และคณาจารย์จากทางส่วนการศึกษา โรงเรียนนายร้อยพระจุลจอมเกล้า รวมทั้งคณาจารย์ที่ได้รับเชิญให้มาประสาทวิชาต่างๆ ซึ่งเป็นประโยชน์ต่องานวิจัยเป็นอย่างยิ่ง

ขอขอบคุณบิดา มารดา น้องสาว น้องชายและภรรยา ตลอดจนเพื่อนร่วมงานทุกท่าน ในความสนับสนุน ช่วยเหลือและให้กำลังใจ เป็นอย่างดีตลอดเวลา

ขอขอบคุณเพื่อนร่วมชั้น โครงการความร่วมมือหลักสูตรปริญญาโท มศว. และ รร. จปร. สาขาวิศวกรรมเครื่องกล รุ่น 3 ทุกท่านที่ให้ความช่วยเหลือและเป็นกำลังใจ

ธนาคม สวาสดิพันธ์

สารบัญ

บทที่	หน้า
1 บทนำ	1
ความเป็นมาและความสำคัญของปัญหา	1
วัตถุประสงค์ของโครงการวิจัย	2
ขอบเขตของโครงการวิจัย	2
วิธีดำเนินการวิจัยโดยย่อ	2
ประโยชน์ที่คาดว่าจะได้รับ	3
2 ทฤษฎีและงานวิจัยที่เกี่ยวข้อง	4
บทปริทัศน์วรรณกรรม (Literature Review)	4
ทฤษฎีความเหมาะสมสูงสุด (Optimization)	5
ตัวแปรออกแบบ	5
พารามิเตอร์ออกแบบ	5
ฟังก์ชันออกแบบ	6
ทฤษฎีความเหมาะสมสูงสุดกับปัญหาทางวิศวกรรม	6
ทฤษฎีความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization)	8
แคลคูลัสฮอพอาริเอชันส์	8
การแก้ปัญหาค่าเหมาะสมสูงสุดพลศาสตร์ด้วยแคลคูลัสฮอพอาริเอชันส์	22
แบบเงื่อนไขแบบไม่เท่ากัน	37
อัลกอริทึมเชิงตัวเลข (Numerical Algorithms)	39
บทนำ	39
การโปรแกรมแบบไม่เชิงเส้น	40
แนวทางวิธีโดยตรง	41
แนวทางวิธีโดยอ้อม	45

สารบัญ (ต่อ)

บทที่	หน้า
3 วิธีการดำเนินการวิจัย	53
การศึกษาโปรแกรม (Study of The Program)	53
แนวทางวิธีโดยตรง	53
แนวทางวิธีโดยอ้อม	54
การออกแบบโปรแกรมคอมพิวเตอร์ (Computer Programming)	55
ขั้นตอนในการแก้ปัญหาโปรแกรม	56
4 ผลการทดสอบและวิเคราะห์ผลการทดสอบ	57
ผลการทดสอบ	57
ตัวอย่างปัญหาการเคลื่อนที่แบบมวลเดียว	57
ตัวอย่างปัญหาหุ่นยนต์แขนกลแบบโพลาไร	59
ตัวอย่างปัญหาหุ่นยนต์แบบ 2 ขึ้นต่อโยง	62
วิเคราะห์ผลการทดสอบ	65
5 สรุปและวิจารณ์ผลการทดสอบ	66
สรุปและวิจารณ์ผลการทดสอบ	66
ข้อเสนอแนะ	68
บรรณานุกรม	69
ภาคผนวก	71
ประวัติย่อผู้วิจัย	90

บัญชีตาราง

ตาราง	หน้า
1 ผลการรันโปรแกรมเปรียบเทียบ CPU Run-time (One mass system)	59
2 ผลการรันโปรแกรมเปรียบเทียบ CPU Run-time (The Polar Manipulator)	62
3 ผลการรันโปรแกรมเปรียบเทียบ CPU Run-time (Planar Two-link Manipulator)	65
4 สรุปผลการรันโปรแกรมเปรียบเทียบ CPU Run-time ของแต่ละปัญหา	67

บัญชีภาพประกอบ

ภาพประกอบ	หน้า
1 แสดงรูปแบบของยกกำลังแบบเชิงท่อนไม่ต่อเนื่อง (Piecewise Polynomial) ของสถานะ (States)	42
2 แสดงรูปแบบของค่าคงที่แบบเชิงท่อนไม่ต่อเนื่อง (Piecewise Constant) ของตัวนำเข้า (Inputs)	43
3 แสดงรูปแบบของขั้นตอน (Flowchart) ของการพัฒนาโปรแกรม	51
4 แสดงรูปแบบของขั้นตอน (Flowchart) ของการพัฒนาโปรแกรม โดยมีการเพิ่มขั้นตอนของการหาค่าเริ่มต้น (Initial Guess) เข้ามาในโปรแกรมเดิม	56
5 แสดงรูปแบบการเคลื่อนที่แบบมวลเดี่ยว (One Mass System)	57
6 แสดงรูปเส้น Curve ของ $x^{(1)}$ และ	58
7 แสดงรูปเส้น Curve ของ u	58
8 แสดงรูปหุ่นยนต์แขนกลแบบโพลาร์ (Polar Manipulator)	59
9 แสดงรูปเส้น Curve ของ θ_1 , $\theta_1^{(1)}$, c_2 และ $c_2^{(1)}$	61
10 แสดงรูปเส้น Curve ของ u_1 และ u_2	61
11 แสดงรูปแขนกล 2 ชั้นต่อโยง (Planar Two-link Manipulator)	62
12 แสดงรูปเส้น Curve ของ q_1 , $q_1^{(1)}$, q_2 และ $q_2^{(1)}$	64
13 แสดงรูปเส้น Curve ของ u_1 และ u_2	64
14 แสดงรูปแบบของขั้นตอนออฟไลน์ (Offline) และออนไลน์ (Online) ของโปรแกรม	66

บทที่ 1

บทนำ

ความเป็นมาและความสำคัญของปัญหา

ในการดำเนินชีวิตประจำวันของมนุษย์เรานั้น ล้วนแล้วแต่จะเกี่ยวข้องหรือประสบพบเจอกับปัญหาทางด้านความเหมาะสมสูงสุดกันทั้งนั้น สืบเนื่องมาจากมนุษย์เรานั้น พยายามที่จะทำอะไรก็ตามแต่เพื่อจะให้ไปถึงจุดมุ่งหมายหรือเป้าหมายที่ได้ตั้งเอาไว้ให้ได้ดีที่สุด เร็วที่สุด สะดวกที่สุด หรือให้ได้ตามเงื่อนไขต่างๆ ที่มีอยู่ ซึ่งนั่นก็หมายความว่าทำให้มีความเหมาะสมที่สุดนั่นเอง ปัญหาที่มีอยู่ว่า ในการตัดสินใจที่แก้ปัญหาต่างๆ นั้น แต่ละคนก็จะมีวิธีการที่แตกต่างกันไปตามความรู้ความสามารถหรือข้อมูลของแต่ละคนๆ นั้นมีอยู่ ทั้งนี้ ในทางวิศวกรรมแล้ว ถือว่าปัญหาทางด้านความเหมาะสมสูงสุดนั้น มีความสำคัญต่อกระบวนการและวิธีการทางด้านวิศวกรรมเป็นอย่างมาก เช่น ในการออกแบบหาขนาดของเพลาล้อของรถยนต์ ซึ่งขนาดของเพลาล้อที่เหมาะสมนั้น ไม่ใช่มีเพียงน้ำหนักของรถหรือน้ำหนักรวมของสิ่งของที่บรรทุกในรถเท่านั้น ที่จะเป็นเงื่อนไขในการหาขนาดของเพลาล้อ หากแต่ยังมีความเร็วสูงสุดที่กำหนดไว้สำหรับเพลาล้อนั้น รวมทั้งชนิดของน้ำมันหรือสารหล่อลื่นที่ใช้ และอาจจะมีเงื่อนไขอื่นๆ อีกมากมายในการที่จะหาแค่ขนาดของเพลาล้อของรถยนต์ให้มีความเหมาะสมสูงสุด

ด้วยเหตุผลดังกล่าว จึงมีผู้ทำการวิจัยจำนวนมากได้เลือกที่จะทำการวิจัยในส่วนที่เกี่ยวข้องกับปัญหาทางด้านความเหมาะสมสูงสุด (Optimization Problem) ไม่ว่าจะเป็นทางด้านวิศวกรรม ด้านการแพทย์ ด้านการกีฬา ด้านธุรกิจ หรือด้านอื่นๆ

ในส่วนของงานวิจัยนี้ เป็นงานวิจัยทางด้านปัญหาความเหมาะสมสูงสุดทางด้านพลศาสตร์ (Dynamic Optimization Problem) ซึ่งก็เป็นปัญหาความเหมาะสมสูงสุดทางด้านวิศวกรรมแขนงหนึ่งนั้น และงานวิจัยนี้จะเป็นการศึกษาและพัฒนาเพิ่มเติมโปรแกรมเพื่อใช้ในการแก้ปัญหาความเหมาะสมสูงสุดทางด้านพลศาสตร์ (Dynamic Optimization Problem) ซึ่งได้เคยมีผู้ทำวิจัยไว้ก่อนหน้านี้ โดยโปรแกรมดังกล่าวสามารถที่จะแก้ไขปัญหาลักษณะโดยวิธีตรง (Direct Method) และโดยวิธีอ้อม (In-direct Method) ได้ ซึ่งหลักการในการทำงานของโปรแกรมหาดังกล่าวจะต้องมีกระบวนการในการหาค่าเริ่มต้น (Initial Guess) เพื่อนำไปใช้ในกระบวนการหาค่าคำตอบสุดท้าย (Final Solution) ทั้งนี้โปรแกรมหาดังกล่าวยังไม่ได้ให้ความสำคัญกับการหาค่าเริ่มต้นให้มีความเหมาะสมมากนัก ซึ่งหากมีการปรับปรุงและพัฒนาในส่วนของคุณค่าเริ่มต้นให้มี

ความเหมาะสมก่อนที่จะนำไปใช้ในกระบวนการอื่นๆ ของโปรแกรม ก็จะทำให้โปรแกรมหาค่าที่เหมาะสมมีประสิทธิภาพสูงขึ้นกว่าที่เป็นอยู่ได้

วัตถุประสงค์ของโครงการวิจัย

เพื่อเพิ่มความสามารถในการหาค่าเริ่มต้น (Initial Guess) ในการแก้ปัญหาความเหมาะสมสูงสุดของระบบพลศาสตร์ ให้มีความเหมาะสมมากกว่าที่เป็นอยู่

ขอบเขตของโครงการวิจัย

พัฒนาโปรแกรมในส่วนของการหาค่าเริ่มต้น (Initial Guess) ก่อนที่จะนำค่าดังกล่าวไปใช้ในกระบวนการต่างๆ ในการแก้ปัญหาความเหมาะสมสูงสุดของโปรแกรมเดิมเพื่อหาคำตอบสุดท้าย โดยที่โปรแกรมหาค่าเริ่มต้นจะใช้ระยะเวลาในการรันโปรแกรมน้อยลงกว่าที่เป็นอยู่

1. การศึกษาวิจัยครั้งนี้ มุ่งศึกษาค้นคว้าศึกษาในส่วนของ Dynamic Optimization Software
2. ใช้ Equation of Motion เป็นตัวหลักในการหาค่าเริ่มต้น (Initial Guess)
3. เปรียบเทียบการใช้ค่าเริ่มต้น (Initial Guess) ในการแก้ปัญหาโดยวิธีตรง (Direct Method) และโดยวิธีอ้อม (In-direct Method)
4. การพัฒนาโปรแกรมจะใช้โปรแกรม Matlab ในการพัฒนาโปรแกรม

วิธีดำเนินการวิจัยโดยย่อ

1. ศึกษาโปรแกรมที่ใช้ในการแก้ปัญหาความเหมาะสมสูงสุดของระบบพลศาสตร์ ของผู้ที่ได้ทำวิจัยไว้ก่อนหน้านี้แล้ว
2. ศึกษาวิธีการแก้ปัญหาโดยวิธีตรง (Direct Method) และโดยวิธีอ้อม (In-direct Method) ในการแก้ไขปัญหาทางด้านความเหมาะสมสูงสุด
3. พัฒนาโปรแกรมในส่วนของการหาค่าเริ่มต้น (Initial Guess) ก่อนที่จะนำค่าดังกล่าวไปใช้ในกระบวนการต่างๆ ในการแก้ปัญหาความเหมาะสมสูงสุดของโปรแกรมเดิมเพื่อหาคำตอบสุดท้าย
4. ทดสอบรันโปรแกรมเปรียบเทียบเวลาที่ใช้ในการรันโปรแกรม (CPU Runtime) กับโปรแกรมเดิมที่ยังไม่ได้มีการปรับปรุงในส่วนของการหาค่าเริ่มต้น (Initial Guess)
5. สรุปและรายงานผล

ประโยชน์ที่คาดว่าจะได้รับ

1. ผลการวิจัยครั้งนี้จะทำให้ได้โปรแกรมคอมพิวเตอร์ที่สามารถใช้คำนวณหาคำตอบของปัญหาในระยะเวลาที่เร็วขึ้นได้
2. ผลการวิจัยครั้งนี้สามารถนำไปใช้เป็นแนวทางในการพัฒนาโปรแกรมในลักษณะเดียวให้สามารถใช้กับงานได้อย่างมีประสิทธิภาพมากขึ้น
3. ผลการวิจัยครั้งนี้จะให้ข้อมูลสำหรับการประยุกต์เอาหลักการออปติไมซ์เซชัน (Optimization) ไปใช้งานกับปัญหาด้านพลศาสตร์และเป็นฐานข้อมูลเพื่อเปรียบ เทียบกับวิธีการแบบอื่นๆ
4. ผลการวิจัยครั้งนี้จะทำให้ได้โปรแกรมคอมพิวเตอร์ที่สามารถนำไปใช้ช่วยในการพัฒนางานวิจัยทางด้านอื่นๆ ต่อไป

บทที่ 2

ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในการวิจัยครั้งนี้ ได้ศึกษาเอกสารและงานวิจัยที่เกี่ยวข้อง และนำเสนอตามหัวข้อต่อไปนี้

1. บทปริทัศน์วรรณกรรม (Literature Review)
2. ทฤษฎีความเหมาะสมสูงสุด (Optimization)
3. ทฤษฎีความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization)
4. อัลกอริทึมเชิงตัวเลข (Numerical Algorithms)

1. บทปริทัศน์วรรณกรรม (Literature Review)

จากการศึกษาปัญหาความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization Problem) ในการแก้ปัญหาดังกล่าวจะประกอบไปด้วยสองวิธีการหลักๆ คือการแก้ปัญหโดยวิธีตรง (Direct Method) และโดยวิธีอ้อม (Indirect Method) ในส่วน Indirect Method จะถูกแก้ปัญหโดยผ่านกระบวนการทางด้าน Calculus of Variation เพื่อให้ได้ออกมาเป็น Necessary condition ซึ่งจะถูกละเปลี่ยนออกมาในรูปของสมการอนุพันธ์ (ODE) รวมทั้งพีชคณิต (Algebra) ด้วย

จากปัญหาดังกล่าว ได้มีผู้ทำการวิจัยได้ทำการเขียนโปรแกรมเพื่อใช้ในการแก้ปัญหาดังกล่าวมาแล้ว ดังนี้

ทวิวัชร วีระเกล้า (Tawiwat Veeraklaew: Dissertation) ได้เขียนโปรแกรมเพื่อใช้ในการแก้ปัญหาด้านความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization Programming) โดยโปรแกรมหาดังกล่าวสามารถที่จะแก้ปัญหในรูปแบบโดยตรง (Direct Method) และโดยอ้อม (Indirect Method) ได้ ซึ่งหลักการในการทำงานของโปรแกรมหาดังกล่าวจะต้องมีกระบวนการในการหาค่าเริ่มต้น (Initial Guess) เพื่อนำไปใช้ในกระบวนการหาค่าคำตอบสุดท้าย (Final Solution) ทั้งนี้โปรแกรมดังกล่าวผู้วิจัยมีข้อเสนอแนะว่า โปรแกรมดังกล่าวยังไม่ได้ให้ความสำคัญกับการหาค่าเริ่มต้นให้มีความเหมาะสมมากนัก ซึ่งหากมีการปรับปรุงในส่วนหาค่าเริ่มต้นให้มีความเหมาะสม ก่อนที่จะนำไปใช้ในกระบวนการอื่นๆ ของโปรแกรม ก็จะทำให้โปรแกรมหาดังกล่าวมีประสิทธิภาพสูงขึ้นกว่าที่เป็นอยู่ได้

2. ทฤษฎีความเหมาะสมสูงสุด (Optimization)

ทฤษฎีความเหมาะสมสูงสุด (Optimization) เป็นวิธีการหรือเครื่องมือทางคณิตศาสตร์ที่นำมาใช้โดยมีจุดประสงค์เพื่อหาค่าที่เหมาะสมที่สุด ซึ่งค่าเหมาะสมที่สุดนี้เป็นได้ทั้งค่าที่ต่ำสุดหรือค่าสูงสุด องค์ประกอบโครงสร้างของปัญหาทางด้านความเหมาะสมสูงสุดในเทอมทางคณิตศาสตร์ที่สำคัญและจำเป็นต้องทำความเข้าใจสามเทอมด้วยกัน คือ ดีไซน์วาร์เอเบิล (Design Variables), ดีไซน์พารามิเตอร์ (Design Parameters) และดีไซน์ฟังก์ชัน (Design Functions) ซึ่งมีรายละเอียดดังนี้

2.1 ตัวแปรออกแบบ (Design Variables)

คือตัวแปรที่ใช้ในการออกแบบในทางคณิตศาสตร์ หรือก็คือตัวแปรของการออกแบบที่ไม่ทราบค่าที่จะต้องถูกค้นหาค่า ยกตัวอย่างเช่น ปัญหาการหาขนาดที่เหมาะสม คือ ตัวแปรออกแบบก็คือความยาวและความสูง เป็นต้น

ในการกำหนดตัวแปรออกแบบ (Design Variables) มีหลักการพื้นฐานคือตัวแปรออกแบบ (Design Variables) จะต้องเป็นเชิงเส้นอิสระ (Linear Independent) หมายความว่าตัวแปรออกแบบ (Design Variables) จะไม่สามารถถูกกำหนดขึ้นมาจากการใช้ความสัมพันธ์ทางเลขคณิตได้ ตัวอย่างเช่น การออกแบบหน้าตัดของสี่เหลี่ยมผืนผ้าที่เหมาะสม เราจะไม่สามารถกำหนดตัวแปรออกแบบ (Design Variables) เป็นความยาว, ความกว้างและพื้นที่ได้ เพราะถ้าหากเรากำหนดตัวแปรออกแบบ (Design Variables) เป็นความยาวและความกว้างแล้ว ตัวแปรที่สามคือพื้นที่จะเกิดขึ้นมาเองโดยอัตโนมัติ ตัวแปรออกแบบ (Design Variables) สามารถมีได้หลายตัวเพื่อที่จะให้การออกแบบนั้นมีความสมบูรณ์ และกลุ่มตัวแปรออกแบบ (Design Variables) เหล่านี้จะถูกเรียกว่า ดีไซน์เวกเตอร์ (Design Vector: Matrix) และใช้อักษร n แทนจำนวนใดๆ ของตัวแปรออกแบบ (Design variables) และในเทอมทางคณิตศาสตร์นั้นมักจะใช้อักษร x และมีตัวเลขน้อยเพื่อระบุลำดับที่ของตัวแปรของการออกแบบนั้นๆ มักมีการเขียนสัญลักษณ์ของดีไซน์เวกเตอร์ (Design Vector) หลายลักษณะดังนี้ คือ $[X]$, x หรือ x , $[x_1, x_2, \dots, x_n]^T$ และ $x_i, i = 1, 2, \dots, n$

2.2 พารามิเตอร์ออกแบบ (Design Parameters)

คือสิ่งที่คงที่ไม่มีเปลี่ยนแปลงในขณะที่กำลังออกแบบอยู่ถึงแม้ว่าจะมีการออกแบบในลักษณะที่แตกต่างกัน เช่น ภาระที่กระทำ, คุณสมบัติของวัสดุและลักษณะรูปทรงเป็นต้น พารามิเตอร์ออกแบบ (Design Parameters) สามารถมีได้หลายตัวเช่นเดียวกับตัวแปรออกแบบ

(Design Variables) และเป็นเวกเตอร์ (Vector: Matrix) เช่นกัน สามารถเขียนสัญลักษณ์ได้คล้ายๆ กันดังนี้ คือ $[P]$, P หรือ p และ $[p_1, p_2, \dots, p_q]^T$

2.3 ฟังก์ชันออกแบบ (Design Functions)

คือข้อมูลสำคัญเกี่ยวกับการออกแบบโดยจะใช้ตัวแปรออกแบบ (Design Variables) และพารามิเตอร์ออกแบบ (Design Parameters) มาประเมินค่า และเป็นตัวพิสูจน์แบบจำลองทางคณิตศาสตร์ของปัญหาการออกแบบด้วย ตัวแปรออกแบบ (Design Functions) นี้สามารถแสดงออกมาในรูปแบบของฟังก์ชันวัตถุประสงค์ (Objective Function) และ/หรือเงื่อนไขบังคับ (Constraints) ซึ่งมีรายละเอียดดังต่อไปนี้

2.3.1 ฟังก์ชันวัตถุประสงค์ (Objective Function)

เป็นฟังก์ชันออกแบบ (Design Function) ตัวหนึ่งของทฤษฎีความเหมาะสมสูงสุด (Optimization) โดยทั่วไปจะเป็นแบบฟังก์ชันเดียว รูปแบบของฟังก์ชันวัตถุประสงค์ (Objective Function) จะเป็นการหาปริมาณที่น้อยที่สุดหรือปริมาณที่มากที่สุด และปริมาณที่ว่านี้ก็เป็นปริมาณ-สเกลาร์ ในทอมนทางคณิตศาสตร์นั้นจะเขียนสัญลักษณ์ได้หลายแบบดังนี้ คือ $f(X)$ และ $f(x_1, x_2, \dots, x_q)$

2.3.2 เงื่อนไขบังคับ (Constraint)

เป็นฟังก์ชันออกแบบ (Design Function) ตัวหนึ่งที่มีอิทธิพลต่อการออกแบบเป็นอย่างมาก และสามารถมีได้มากกว่าหนึ่งฟังก์ชันได้ (เป็นได้ทั้งสมการและอสมการ) ดังนั้นจึงเป็นเวกเตอร์ (Vector: Matrix) และจะใช้เครื่องหมายดำเนินการเปรียบเทียบสามตัวคือ $\leq$, $\geq$ และ $=$ ซึ่งเงื่อนไขบังคับ (Constraints) นี้จะมีได้สามรูปแบบด้วยกันคือ แบบไม่มีเงื่อนไข (Unconstraint) แบบเงื่อนไขแบบเท่ากัน (Equality Constraints) และแบบเงื่อนไขแบบไม่เท่ากัน (Inequality Constraints)

2.4 ทฤษฎีความเหมาะสมสูงสุดกับปัญหาทางวิศวกรรม (Engineering Optimization)

ปัญหาทางวิศวกรรมที่ตัวแปรออกแบบ (Design Variables) ไม่ขึ้นกับเวลาและมีการนำเอาทฤษฎีความเหมาะสมสูงสุด (Optimization) เข้ามาประยุกต์ใช้งานจะเรียกกันว่า ความ

เหมาะสมสูงสุดสถิตยศาสตร์ (Static Optimization) ส่วนปัญหาที่ขึ้นกับเวลาจะเรียกว่า ความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization) ดังที่ได้กล่าวมาแล้วในหัวข้อที่ผ่านมาพบว่า เป้าหมายของทฤษฎีความเหมาะสมสูงสุด (Goal of Optimization) คือการค้นหาค่าที่เหมาะสมของตัวแปรออกแบบ (Design Variables) ที่มีฟังก์ชันวัตถุประสงค์ (Objective Function) เป็นค่า น้อยที่สุดหรือมากที่สุด (Minimum or Maximum)

2.4.1 ความเหมาะสมสูงสุดสถิตยศาสตร์ (Static Optimization)

ปัญหาทางวิศวกรรมที่ตัวแปรออกแบบ (Design Variables) ไม่ขึ้นกับเวลา ถือว่าเป็น ปัญหาพื้นฐาน แต่ก็มีผลสำคัญต่อการเข้าใจความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization) เพราะต้องอาศัยหลักการแก้ปัญหาของความเหมาะสมสูงสุดสถิตยศาสตร์ (Static Optimization) เช่นกัน

2.4.2 ความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization)

ปัญหาทางวิศวกรรมที่ตัวแปรต่างๆ (Variables) ขึ้นกับเวลา (Time-Dependent) จะถูก เรียกว่าพลศาสตร์ (Dynamics) และมักจะใช้สมการเชิงอนุพันธ์ (Differential Equations) มาเป็น ตัวแสดงระบบพลศาสตร์ (Dynamic Systems) ดังนี้

$$\dot{x}_i = f_i(x_1, \dots, x_n, u_1, \dots, u_m, t), i = 1, \dots, n \quad (2.1)$$

เมื่อ t คือเวลา (Time), x_i คือตัวแปรสถานะ (State Variables) ตัวอย่างเช่น พิกัดทั่วไปหรืออนุพันธ์เวลา (Generalized Coordinates or Time Derivatives), u_i คือ ตัวควบคุมนำเข้า (Control Inputs) และ f_i คือฟังก์ชันไม่เชิงเส้น (Nonlinear Functions) ของตัวแปรสถานะ (State Variables) และตัวควบคุมนำเข้า (Control Inputs) หากทำการกำหนดค่าให้กับตัวควบคุมนำเข้า (Control Inputs) $u_1(t), \dots, u_m(t)$ เป็นเงื่อนไขเริ่มต้น (Initial Condition) เราก็จะสามารถ ทำการคำนวณวิถี (Trajectory) ของตัวแปรสถานะ (State Variables) $x_i(t)$ ได้ด้วยระเบียบวิธีวิเคราะห์หรือระเบียบวิธีเชิงตัวเลข (Analytical or Numerical Method) ได้ สำหรับระบบทาง วิศวกรรมที่แสดงได้ด้วยสมการ (2.1) นั้น โดยปกติแล้วจะต้องมีการกำหนดสภาวะเริ่มต้น (Initial State) เสมอ จากนั้นก็ทำการเลือกช่วงเวลาให้กับตัวควบคุมนำเข้า (Control Inputs)

$u_1(t), \dots, u_m(t)$ เพื่อให้ฟังก์ชันวัตถุประสงค์ (Objective Function) เป็นค่าน้อยที่สุดหรือมากที่สุด ปัญหาทางด้านพลศาสตร์จะเรียกฟังก์ชันวัตถุประสงค์ (Objective Function) ว่าฟังก์ชันต้นทุน (Cost Function) และเขียนเป็นสมการดังนี้

$$J = \Phi(t, x_1, \dots, x_n, u_1, \dots, u_m)_{t_f} + \int_{t_0}^{t_f} L(t, x_1, \dots, x_n, u_1, \dots, u_m) dt \quad (2.2)$$

เมื่อ t_0 คือเวลาเริ่มต้น, t_f คือเวลาสุดท้าย, ส่วนฟังก์ชันต้นทุน (Cost Function) ประกอบไปด้วยสองส่วน ส่วนแรกคือ $\Phi(t, x_1, \dots, x_n, u_1, \dots, u_m)$ ซึ่งจะขึ้นอยู่กับเวลาสุดท้าย (Final Time) และสภาวะสุดท้าย (Final State) ของระบบ ส่วนที่สองคือ $\int_{t_0}^{t_f} L(t, x_1, \dots, x_n, u_1, \dots, u_m) dt$ เป็นส่วนที่ขึ้นอยู่กับเวลาที่ผ่านไป (Time History) ของตัวแปรสถานะ (State Variables) และตัวแปรควบคุม (Control Variables) โดยที่ Φ และ L คือฟังก์ชันไม่เชิงเส้น (Nonlinear Functions) ของตัวแปรสถานะ (State Variables) และตัวแปรควบคุม (Control Variables)

3. ทฤษฎีความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization)

การแก้ปัญหาคความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization) นั้นแบ่งได้เป็นสองลักษณะใหญ่ๆ ด้วยกัน คือ วิธีโดยตรง (Direct Methods) และวิธีโดยอ้อม (Indirect Methods) ซึ่งวิธีโดยตรง (Direct Methods) จะให้ผลคำตอบที่หยาบมากกว่าแต่ก็มีความซับซ้อนน้อยกว่าวิธีโดยอ้อม (Indirect Methods)

3.1 แคลคูลัสของฟาริเอชันส์ (Calculus of Variations)

โดยจะเริ่มจากระบบขั้นความเสรีเดี่ยว (Single-stage Systems) และระบบหลายระดับขั้นความเสรี (Multi-stage Systems) ตามลำดับ สัญลักษณ์ที่ใช้แทนแคลคูลัสของฟาริเอชันส์ (Calculus of Variations) ก็คือฟังก์ชันนอล (Functional) เขียนเป็นสมการได้เป็น

$$J[x] = \int_{t_0}^{t_f} F(t, x, \dot{x}) dt \quad (2.3)$$

ซึ่งเป็นการเปลี่ยนฟังก์ชันต่อเนื่อง (Continuous Functions) $x(t)$ ให้เป็นจำนวนจริง นั่นคือ ถ้ากำหนด $x(t)$ ก็จะสามารถหาค่าของฟังก์ชันนอล (Functional) ได้โดยการใช้ระเบียบวิธีเชิงวิเคราะห์หรือระเบียบวิธีเชิงตัวเลข (Analytical or Numerical Method) โดยที่ฟังก์ชันนอล (Functional) อาจขึ้นอยู่กับหลายๆ ฟังก์ชันต่อเนื่อง (Continuous Functions) ก็ได้ดังเช่น

$$J[x_1, x_2, \dots, x_n] = \int_{t_0}^{t_f} F(t, x_1, x_2, \dots, x_n, \dot{x}_1, \dot{x}_2, \dots, \dot{x}_n) dt \quad (2.4)$$

ฟังก์ชันนอล (Functional) สมการ (2.4) เป็นการเปลี่ยนฟังก์ชันต่อเนื่อง (Continuous Functions) หลายๆ ฟังก์ชัน ก็คือ $x_1(t)$, $x_2(t)$, ..., $x_n(t)$ ให้เป็นจำนวนจริงนั่นเอง ในการนำเอาแคลคูลัสสอพอวาไรเอเบิล (Calculus of Variations) มาใช้นั้นมีเป้าหมายเดียวกันกับความเหมาะสมสูงสุดสถิตยศาสตร์ (Static Optimization) นั่นก็คือ ค้นหาเงื่อนไขจำเป็น (Necessary Conditions) สำหรับเอ็กทรีมัม (Extremum) สำหรับฟังก์ชันนอล (Functional) และการตรวจสอบว่าเอ็กทรีมัม (Extremum) ที่หาได้เป็นค่าน้อยที่สุดหรือมากที่สุด (Sufficient Conditions) ในการนำเอาแคลคูลัสสอพอวาไรเอชันส์ (Calculus of Variations) มาใช้กับทฤษฎีความเหมาะสมพลศาสตร์ (Dynamic Optimization) จะเกี่ยวข้องกับจุดเริ่มต้นและจุดสุดท้าย ซึ่งมีได้หลายกรณี ดังรายละเอียดต่อไปนี้

3.1.1 ฟังก์ชันนอลของฟังก์ชันเดียว (Functionals of a Single Function)

เป็นฟังก์ชันนอล (Functional) ของระบบขั้นความเสรีเดียว (Single-stage Systems) โดยมีรายละเอียดของจุดสุดท้ายและเวลาสุดท้ายหลายลักษณะดังนี้ คือ

1. กำหนดจุดสุดท้ายและเวลาสุดท้ายไว้ตายตัว (Fixed End Time and End Points)

ถ้าฟังก์ชัน $F(t, x, \dot{x})$ สามารถทำอนุพันธ์อันดับหนึ่งและสองได้อย่างต่อเนื่องโดยอยู่ระหว่างช่วงเวลา $t_0 \leq t \leq t_f$ และสอดคล้องกับเงื่อนไขขอบเขต (Boundary Conditions) $x(t_0) = x_0$ และ $x(t_f) = x_f$ ก็จะเป็นการกำหนดการค้นหาค่าของฟังก์ชันเป้าหมาย (Desire Function) $x(t)$ ที่เป็นเอ็กทรีมัม (Extremum) ของฟังก์ชันนอล (Functionals) $J[x] = \int_{t_0}^{t_f} F(t, x, \dot{x}) dt$ นั่นเอง ซึ่งมีวิธีการทำได้ดังนี้ คือ เรากำหนดให้ $x(t)$ เป็นฟังก์ชันเป้าหมาย (Desire Function) ที่เป็นเอ็กทรีมัม (Extremum) ของฟังก์ชันนอล (Functionals) $J[x]$ ถ้า $x(t)$ ถูกเพิ่มค่าด้วย $h(t)$ ดังนั้น

เพื่อให้ยังคงสอดคล้องกับเงื่อนไขขอบเขต (Boundary Conditions) อยู่ ดังนั้น $h(t_0) = h(t_f) = 0$ ก็จะทำให้พบการเปลี่ยนแปลงของฟังก์ชันนอล (Functional) ΔJ เป็น

$$\Delta J = J[x+h] - J[x] = \int_{t_0}^{t_f} [F(t, x+h, \dot{x}+\dot{h}) - F(t, x, \dot{x})] dt \quad (2.5)$$

เมื่อใช้อนุกรมเทย์เลอร์ (Taylor Series) และตัดเทอมที่มีเลขชี้กำลังสูงทิ้งไป ได้การเปลี่ยนแปลงเป็น

$$\delta J = \int_{t_0}^{t_f} \left(\frac{\partial F}{\partial x} h + \frac{\partial F}{\partial \dot{x}} \dot{h} \right) dt \quad (2.6)$$

เมื่อ δJ เป็นค่าโดยประมาณของ ΔJ เนื่องจากได้ตัดเทอมที่มีเลขชี้กำลังสูงทิ้งไป ดังนั้นเมื่อทำการอินทิเกรตบายพาส (Integrating by Parts) สมการ (2.6) ก็จะได้พบว่า

$$\delta J = \int_{t_0}^{t_f} \left(\frac{\partial F}{\partial x} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}} \right) h dt + \left(\frac{\partial F}{\partial \dot{x}} h \right) \Big|_{t_f} - \left(\frac{\partial F}{\partial \dot{x}} h \right) \Big|_{t_0} \quad (2.7)$$

เมื่อเงื่อนไขจำเป็น (Necessary Conditions) ก็คือ $\delta J = 0$ ที่ $h(t_0) = h(t_f) = 0$ จึงจะสอดคล้องกับเงื่อนไขขอบเขต (Boundary Conditions) และกำหนดไว้อีกว่า F จะต้องทำอนุพันธ์ได้อย่างต่อเนื่องสองครั้ง นั่นก็หมายความว่า $\frac{\partial F}{\partial x} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}}$ เป็นฟังก์ชันที่ต่อเนื่อง และเมื่อ $\delta J = 0$ ซึ่งพบว่า

$$\frac{\partial F}{\partial x} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}} = 0 \quad (2.8)$$

ซึ่งสมการ (2.8) นี้เป็นที่รู้จักกันในนาม สมการของออยเลอร์ (Euler's Equation) ซึ่งจะทำให้ผลเฉลยของ $x(t)$ สอดคล้องกับเงื่อนไขขอบเขต (Boundary Conditions) $x(t_0) = x_0$

และ $x(t_f) = x_f$ ซึ่งเราสามารถทำการหาค่าโดยใช้ความรู้ทางแคลคูลัส (Calculus) หาค่าได้โดยวิธีออยเลอร์-ควอซี (Euler-Cauchy) ได้

2. กำหนดเวลาสุดท้ายไว้ตายตัวแต่จุดสุดท้ายแปรผันได้ (Fixed End Time, Variable End Points) แตกต่างกับกรณี (2.3.1.1.1) เพียงแค่จุดสุดท้ายเท่านั้น ดังนั้นจึงใช้สมการ (2.5-2.7) ได้ เพียงแต่เงื่อนไขขอบเขต (Boundary Conditions) $x(t_0)$ และ $x(t_f)$ ไม่ได้ถูกกำหนดไว้ตายตัว และ $h(t_0)$, $h(t_f)$ ก็สามารถกำหนดได้ตามใจชอบ ทำให้ได้เงื่อนไขจำเป็น (Necessary Conditions) เป็น

$$\frac{\partial F}{\partial x} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}} = 0 \quad (2.9)$$

$$\frac{\partial F}{\partial \dot{x}} \Big|_{t_0} = 0 \quad (2.10)$$

$$\frac{\partial F}{\partial \dot{x}} \Big|_{t_f} = 0 \quad (2.11)$$

ซึ่งสมการ (2.9-2.11) นี้เป็นที่รู้จักกันในนามสมการของออยเลอร์ (Euler's Equation) ซึ่งจะทำให้ผลเฉลยของ $x(t)$ สอดคล้องกับเงื่อนไขขอบเขต (Boundary Conditions) $x(t_0)$ และ $x(t_f)$ ซึ่งสามารถหาค่าโดยใช้ความรู้ทางแคลคูลัส (Calculus) โดยวิธีออยเลอร์-ควอซี (Euler-Cauchy) ได้

3. เวลาสุดท้ายและจุดสุดท้ายแปรผันได้ (Variable End Time and End Points) กรณีนี้ถือได้ว่าเป็นรูปแบบทั่วไปของปัญหาทางวิศวกรรมเลยก็ว่าได้ แต่ก็มีวิธีการหาคำตอบคล้ายๆ กับกรณี (2.3.1.1.1) ถ้า $x(t)$ เป็นฟังก์ชันเป้าหมาย (Desire Function) ที่เป็นเอ็กทรีมัม (Extremum) ของฟังก์ชันนอล (Functionals) $J[x]$ ถ้า $x(t)$ ถูกเพิ่มค่าด้วย $h(t)$ ดังนั้นก็จะพบการเปลี่ยนแปลงของฟังก์ชันนอล (Functional) ΔJ เป็น

$$\begin{aligned}
\Delta J &= J[x+h] - J[x] \\
&= \int_{t_0+\delta t_0}^{t_f+\delta t_f} F(t, x+h, \dot{x}+\dot{h}) dt - \int_{t_0}^{t_f} F(t, x, \dot{x}) dt \\
&= \int_{t_0}^{t_f} [F(t, x+h, \dot{x}+\dot{h}) - F(t, x, \dot{x})] dt \\
&\quad + \int_{t_f}^{t_f+\delta t_f} F(t, x+h, \dot{x}+\dot{h}) dt - \int_{t_0+\delta t_0}^{t_0+\delta t_0+\delta t_f} F(t, x+h, \dot{x}+\dot{h}) dt
\end{aligned} \tag{2.12}$$

เมื่อใช้อนุกรมเทย์เลอร์ (Taylor Series) และตัดเทอมที่มีเลขชี้กำลังสูงทิ้งไป จะได้การเปลี่ยนแปลงเป็น

$$\begin{aligned}
\delta J &= \int_{t_0}^{t_f} \left(\frac{\partial F}{\partial x} h + \frac{\partial F}{\partial \dot{x}} \dot{h} \right) dt + F(t, x+h, \dot{x}+\dot{h}) \Big|_{t_f} \delta t_f \\
&\quad - F(t, x+h, \dot{x}+\dot{h}) \Big|_{t_0} \delta t_0
\end{aligned} \tag{2.13}$$

เมื่อ δJ เป็นค่าโดยประมาณของ ΔJ เนื่องจากได้ตัดเทอมที่มีเลขชี้กำลังสูงทิ้งไป ดังนั้นเมื่อทำการอินทิเกรตบายพาส (Integrating by Parts) สมการ (2.13) ก็จะได้พบว่

$$\begin{aligned}
\delta J &= \int_{t_0}^{t_f} \left(\frac{\partial F}{\partial x} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}} \right) h dt + \left(\frac{\partial F}{\partial \dot{x}} h \right) \Big|_{t_f} - \left(\frac{\partial F}{\partial \dot{x}} h \right) \Big|_{t_0} \\
&\quad + F(t, x+h, \dot{x}+\dot{h}) \Big|_{t_f} \delta t_f - F(t, x+h, \dot{x}+\dot{h}) \Big|_{t_0} \delta t_0
\end{aligned} \tag{2.14}$$

เมื่อ $h(t_0) = \delta x \Big|_{t_0} - \dot{x} \Big|_{t_0} \delta t_0$ และ $h(t_f) = \delta x \Big|_{t_f} - \dot{x} \Big|_{t_f} \delta t_f$ ดังนั้นสมการ (2.14) เขียน

ได้เป็น

$$\begin{aligned}
\delta J &= \int_{t_0}^{t_f} \left(\frac{\partial F}{\partial x} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}} \right) h dt + \left(\frac{\partial F}{\partial \dot{x}} \delta x \right) \Big|_{t_f} - \left(\frac{\partial F}{\partial \dot{x}} \delta x \right) \Big|_{t_0} \\
&\quad + \left[F - \frac{\partial F}{\partial \dot{x}} \dot{x} \right] \Big|_{t_f} \delta t_f - \left[F - \frac{\partial F}{\partial \dot{x}} \dot{x} \right] \Big|_{t_0} \delta t_0
\end{aligned} \tag{2.15}$$

เมื่อเงื่อนไขจำเป็น (Necessary Conditions) ก็คือ $\delta J = 0$ ที่ $h(t)$, $\delta x|_{t_f}$, $\delta x|_{t_0}$, δt_f และ δt_0 จึงจะสอดคล้องกับเงื่อนไขขอบเขต (Boundary Conditions) ทำให้ได้เงื่อนไขจำเป็น (Necessary Conditions) และมีเงื่อนไขขอบเขต (Boundary Conditions) คือ

$$\frac{\partial F}{\partial x} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}} = 0 \quad (2.16)$$

$$\frac{\partial F}{\partial \dot{x}}|_{t_0} = 0, \frac{\partial F}{\partial \dot{x}}|_{t_f} = 0, \left[F - \frac{\partial F}{\partial \dot{x}} \dot{x} \right]|_{t_f} = 0, \left[F - \frac{\partial F}{\partial \dot{x}} \dot{x} \right]|_{t_0} = 0 \quad (2.17)$$

ซึ่งจุดสุดท้ายที่ t_0, t_f และค่าของฟังก์ชันที่จุดสุดท้าย $x(t_0), x(t_f)$ สามารถคำนวณได้จากเงื่อนไขขอบเขต (Boundary Conditions) ทั้ง 4 นั้นนั่นเอง

3.1.2 ฟังก์ชันนอลของหลายฟังก์ชัน (Functionals of N Functions)

เป็นฟังก์ชันนอล (Functional) ของระบบหลายระดับขั้นความเสรี (Multistage systems) โดยมีรายละเอียดของจุดสุดท้ายและเวลาสุดท้ายหลายลักษณะดังนี้ คือ

1. กำหนดจุดสุดท้ายและเวลาสุดท้ายไว้ตายตัว (Fixed End Times and End Points)

ถ้าฟังก์ชัน $F(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n)$ สามารถทำอนุพันธ์อันดับหนึ่งและสองได้อย่างต่อเนื่องโดยอยู่ระหว่างช่วงเวลา $t_0 \leq t \leq t_f$ และสอดคล้องกับเงื่อนไขขอบเขต (Boundary Conditions)

$x_i(t_0) = x_{i1}$ และ $x_i(t_f) = x_{i2}$ ก็จะเป็นการกำหนดการค้นหาค่าของฟังก์ชันเป้าหมาย (Desire Function) $x_i(t), i = 1, \dots, n$ ที่เป็นเอ็กทรีมัม (Extremum) ของฟังก์ชันนอล (Functionals)

$J[x] = \int_{t_0}^{t_f} F(t, x, \dot{x}) dt$ นั้นเอง ซึ่งมีวิธีการทำได้ดังนี้ คือ เรากำหนดให้ $x(t)$ เป็นฟังก์ชันเป้าหมาย (Desire Function) ที่เป็นเอ็กทรีมัม (Extremum) ของฟังก์ชันนอล (Functionals) คือ

$$J[x_1, \dots, x_n] = \int_{t_0}^{t_f} F(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n) dt \quad (2.18)$$

ถ้า $x_i(t), i=1, \dots, n$ ถูกเพิ่มค่าด้วย $h_i(t), i=1, \dots, n$ และเพื่อให้ยังคงสอดคล้องกับเงื่อนไขขอบเขต (Boundary Conditions) ดังนั้น $h_i(t_0) = h_i(t_f) = 0$ ก็จะทำให้พบการเปลี่ยนแปลงของฟังก์ชันนอล (Functional) ΔJ เป็น

$$\begin{aligned}\Delta J &= J[x_1 + h_1, \dots, x_n + h_n] - J[x_1, \dots, x_n] \\ &= \int_{t_0}^{t_f} \left[F(t, x_1 + h_1, \dots, x_n + h_n, \dot{x}_1 + \dot{h}_1, \dots, \dot{x}_n + \dot{h}_n) \right. \\ &\quad \left. - F(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n) \right] dt\end{aligned}\quad (2.19)$$

ใช้อนุกรมเทย์เลอร์ (Taylor Series) และตัดเทอมที่มีเลขชี้กำลังสูงทิ้งไปก็จะได้การเปลี่ยนแปลงเป็น

$$\delta J = \int_{t_0}^{t_f} \sum_{i=1}^n \left(\frac{\partial F}{\partial x_i} h_i + \frac{\partial F}{\partial \dot{x}_i} \dot{h}_i \right) dt \quad (2.20)$$

เมื่อ δJ เป็นค่าโดยประมาณของ ΔJ เนื่องจากได้ตัดเทอมที่มีเลขชี้กำลังสูงทิ้งไป ดังนั้นเมื่อทำการอินทิเกรตบายพาส (Integrating by parts) สมการ (2.20) ก็จะได้พบว่า

$$\delta J = \int_{t_0}^{t_f} \sum_{i=1}^n \left(\frac{\partial F}{\partial x_i} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}_i} \right) h_i dt + \sum_{i=1}^n \left[\frac{\partial F}{\partial \dot{x}_i} h_i \Big|_{t_f} - \frac{\partial F}{\partial \dot{x}_i} h_i \Big|_{t_0} \right] \quad (2.21)$$

เมื่อเงื่อนไขจำเป็น (Necessary Conditions) ก็คือ $\delta J = 0$ ที่ $h_i(t_0) = h_i(t_f) = 0$ จึงจะสอดคล้องกับเงื่อนไขขอบเขต (Boundary Conditions) และกำหนดไว้อีกว่า F จะต้องทำอนุพันธ์ได้อย่างต่อเนื่องสองครั้งนั้นก็หมายความว่า $\frac{\partial F}{\partial x_i} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}_i}$ เป็นฟังก์ชันที่ต่อเนื่องและเมื่อ $\delta J = 0$ ดังนั้นพบว่า

$$\frac{\partial F}{\partial x_i} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}_i} = 0, i=1, \dots, n \quad (2.22)$$

ซึ่งสมการ (2.22) นี้เป็นที่รู้จักกันในนามสมการของออยเลอร์ (Euler's Equation) ซึ่งจะทำให้ผลเฉลยของ $x_i(t)$ สอดคล้องกับเงื่อนไขขอบเขต (Boundary Conditions) $x_i(t_0) = x_{i1}$ และ $x_i(t_f) = x_{i2}$ ซึ่งเราสามารถทำการหาค่าโดยใช้ความรู้ทางแคลคูลัส (Calculus) หาค่าได้โดยวิธีออยเลอร์-ควอซี (Euler-Cauchy) ได้

2. กำหนดเวลาสุดท้ายไว้ตายตัวแต่จุดสุดท้ายแปรผันได้ (Fixed End Time and Variable End Points)

แตกต่างกับกรณี (1.) เพียงแค่จุดสุดท้ายเท่านั้น ดังนั้นจึงใช้สมการ (2.19-2.21) ได้ เพียงแต่เงื่อนไขขอบเขต (Boundary Conditions) $x_i(t_0)$ และ $x_i(t_f)$ ไม่ได้ถูกกำหนดไว้ตายตัว และ $h_i(t_0), h_i(t_f)$ ก็สามารถกำหนดได้ตามใจชอบ ทำให้ได้เงื่อนไขจำเป็น (Necessary Condition) เป็น

$$\frac{\partial F}{\partial x_i} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}_i} = 0, i = 1, \dots, n \quad (2.23)$$

$$\frac{\partial F}{\partial \dot{x}_i} \Big|_{t_0} = 0, i = 1, \dots, n \quad (2.24)$$

$$\frac{\partial F}{\partial \dot{x}_i} \Big|_{t_f} = 0, i = 1, \dots, n \quad (2.25)$$

ซึ่งสมการ (2.23-2.25) นี้เป็นที่รู้จักกันในนามสมการของออยเลอร์ (Euler's Equation) ซึ่งจะทำให้ผลเฉลยของ $x_i(t)$ สอดคล้องกับเงื่อนไขขอบเขต (Boundary Conditions) $x_i(t_0)$ และ $x_i(t_f)$ ซึ่งเราสามารถทำการหาค่าโดยใช้ความรู้ทางแคลคูลัส (Calculus) หาค่าได้โดยวิธีออยเลอร์-ควอซี (Euler-Cauchy) ได้เช่นกัน

3. เวลาสุดท้ายและจุดสุดท้ายแปรผันได้ (Variable End Time and End Points)

กรณีนี้ถือได้ว่าเป็นรูปแบบทั่วไปของปัญหาทางวิศวกรรมเลยก็ว่าได้ แต่ก็มีวิธีการหาคำตอบคล้ายๆ กับกรณี (1.) ถ้า $x_i(t), i = 1, \dots, n$ เป็นฟังก์ชันเป้าหมาย (Desire function) ที่เป็นเอ็กท

รีมั (Extremum) ของฟังก์ชันนอล (Functionals) $J[x_1, \dots, x_n]$ ถ้า $x_i(t)$ ถูกเพิ่มค่าด้วย $h_i(t)$ ดังนั้นก็จะพบการเปลี่ยนแปลงของฟังก์ชันนอล (Functional) ΔJ เป็น

$$\begin{aligned}
 \Delta J &= J[x_1 + h_1, \dots, x_n + h_n] - J[x_1, \dots, x_n] \\
 &= \int_{t_0 + \delta t_0}^{t_f + \delta t_f} F(t, x_1 + h_1, \dots, x_n + h_n, \dot{x}_1 + \dot{h}_1, \dots, \dot{x}_n + \dot{h}_n) dt \\
 &\quad - \int_{t_0}^{t_f} F(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n) dt \\
 \Delta J &= \int_{t_0}^{t_f} \left[F(t, x_1 + h_1, \dots, x_n + h_n, \dot{x}_1 + \dot{h}_1, \dots, \dot{x}_n + \dot{h}_n) \right. \\
 &\quad \left. - F(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n) \right] dt \\
 &\quad + \int_{t_f}^{t_f + \delta t_f} F(t, x_1 + h_1, \dots, x_n + h_n, \dot{x}_1 + \dot{h}_1, \dots, \dot{x}_n + \dot{h}_n) dt \\
 &\quad - \int_{t_0}^{t_0 + \delta t_0} F(t, x_1 + h_1, \dots, x_n + h_n, \dot{x}_1 + \dot{h}_1, \dots, \dot{x}_n + \dot{h}_n) dt
 \end{aligned} \tag{2.26}$$

เมื่อใช้อนุกรมเทย์เลอร์ (Taylor series) และตัดเทอมที่เป็นอนุพันธ์อันดับสูง (Higher-order Derivative) ทิ้งไป จะได้การเปลี่ยนแปลงเป็น

$$\begin{aligned}
 \delta J &= \int_{t_0}^{t_f} \sum_{i=1}^n \left(\frac{\partial F}{\partial x_i} h_i + \frac{\partial F}{\partial \dot{x}_i} \dot{h}_i \right) dt \\
 &\quad + F(t, x_1 + h_1, \dots, x_n + h_n, \dot{x}_1 + \dot{h}_1, \dots, \dot{x}_n + \dot{h}_n) \Big|_{t_f} \delta t_f \\
 &\quad - F(t, x_1 + h_1, \dots, x_n + h_n, \dot{x}_1 + \dot{h}_1, \dots, \dot{x}_n + \dot{h}_n) \Big|_{t_0} \delta t_0
 \end{aligned} \tag{2.27}$$

เมื่อ δJ เป็นค่าโดยประมาณของ ΔJ เนื่องจากได้ตัดเทอมที่เป็นอนุพันธ์อันดับสูง (Higher-order Derivative) ทิ้งไป ดังนั้นเมื่อทำการอินทิเกรตบายพาส (Integrating by Parts) สมการ (2.27) ก็จะพบว่า

$$\begin{aligned}
 \delta J &= \int_{t_0}^{t_f} \sum_{i=1}^n \left(\frac{\partial F}{\partial x_i} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}_i} \right) h_i dt + \sum_{i=1}^n \left(\frac{\partial F}{\partial \dot{x}_i} h_i \right) \Big|_{t_f} - \sum_{i=1}^n \left(\frac{\partial F}{\partial \dot{x}_i} h_i \right) \Big|_{t_0} \\
 &\quad + \left[F(t, x_1 + h_1, \dots, x_n + h_n, \dot{x}_1 + \dot{h}_1, \dots, \dot{x}_n + \dot{h}_n) \right]_{t_f} \delta t_f \\
 &\quad - \left[F(t, x_1 + h_1, \dots, x_n + h_n, \dot{x}_1 + \dot{h}_1, \dots, \dot{x}_n + \dot{h}_n) \right]_{t_0} \delta t_0
 \end{aligned} \tag{2.28}$$

$$\text{เมื่อ } h_i(t_0) = \delta x_i|_{t_0} - \dot{x}_i|_{t_0} \delta t_0 \quad \text{และ} \quad h_i(t_f) = \delta x_i|_{t_f} - \dot{x}_i|_{t_f} \delta t_f \quad \text{ดังนั้นสมการ (2.28)}$$

จะเป็น

$$\begin{aligned} \delta J = & \int_{t_0}^{t_f} \sum_{i=1}^n \left(\frac{\partial F}{\partial x_i} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}_i} \right) h_i dt + \sum_{i=1}^n \left(\frac{\partial F}{\partial \dot{x}_i} \delta x_i \right) \Big|_{t_f} \\ & - \sum_{i=1}^n \left(\frac{\partial F}{\partial \dot{x}_i} \delta x_i \right) \Big|_{t_0} + \left[F - \sum_{i=1}^n \frac{\partial F}{\partial \dot{x}_i} \dot{x}_i \right] \Big|_{t_f} \delta t_f \\ & - \left[F - \sum_{i=1}^n \frac{\partial F}{\partial \dot{x}_i} \dot{x}_i \right] \Big|_{t_0} \delta t_0 \end{aligned} \quad (2.29)$$

$$\text{เมื่อ } \delta J = 0 \quad \text{ที่ } h_i(t), \quad \delta x_i|_{t_f}, \quad \delta x_i|_{t_0}, \quad \delta t_f \quad \text{และ} \quad \delta t_0 \quad \text{จึงจะสอดคล้องกับเงื่อนไข}$$

ขอบเขต (Boundary Conditions) ทำให้ได้เงื่อนไขจำเป็น (Necessary Conditions) เป็น

$$\frac{\partial F}{\partial x_i} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}_i} = 0, i = 1, \dots, n \quad (2.30)$$

และมีเงื่อนไขขอบเขต (Boundary Conditions) เป็น

$$\begin{aligned} \frac{\partial F}{\partial \dot{x}_i} \Big|_{t_0} = 0, \quad \frac{\partial F}{\partial \dot{x}_i} \Big|_{t_f} = 0, i = 1, \dots, n \\ \left[F - \sum_{i=1}^n \frac{\partial F}{\partial \dot{x}_i} \dot{x}_i \right] \Big|_{t_f} = 0, \quad \left[F - \sum_{i=1}^n \frac{\partial F}{\partial \dot{x}_i} \dot{x}_i \right] \Big|_{t_0} = 0, i = 1, \dots, n \end{aligned} \quad (2.31)$$

ซึ่งจุดสุดท้ายที่ t_0, t_f และค่าของฟังก์ชันที่จุดสุดท้าย $x_i(t_0), x_i(t_f)$ สามารถคำนวณได้จากเงื่อนไขขอบเขต (Boundary Conditions) ทั้ง 4 นั้นนั่นเอง

3.1.3 ฟังก์ชันนอลมีเงื่อนไขบังคับ (Functionals with Constraints)

เป็นฟังก์ชันนอล (Functional) ที่มีเงื่อนไขบังคับเข้ามาเกี่ยวข้องด้วยซึ่งอาจจะเป็นได้ทั้งแบบเงื่อนไขแบบเท่ากัน (Equality Constraints) และแบบเงื่อนไขแบบไม่เท่ากัน (Inequality Constraints) ในที่นี้จะกล่าวถึงเฉพาะวิธีตัวคูณลากรางจ์ (Lagrange Multiplier)

1. ฟังก์ชันคอนสเทรนต์ (Function Constraints)

ทำการพิจารณาฟังก์ชันนอล (Functional) $J[x_1, \dots, x_n]$ ซึ่งจะต้องเกี่ยวข้องหรือมีความสอดคล้องกับฟังก์ชันแบบเงื่อนไขแบบเท่ากัน (Equality Constraints) $g_j(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n) = 0, j = 1, \dots, m$ และ $m < n$ เสมอ เมื่อเราใช้เทคนิควิธีการแบบตัวคูณลากรางจ์ (Lagrange Multiplier) ก็จะได้ฟังก์ชันนอลใหม่เป็น

$$J'[x_1, \dots, x_n] = \int_{t_0}^{t_f} \left[F(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n) + \sum_{j=1}^m \lambda_j(t) g_j(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n) \right] dt \quad (2.32)$$

เมื่อ $\lambda_j(t)$ คือตัวคูณลากรางจ์ (Lagrange Multiplier) และ $F' = F + \sum_{j=1}^m \lambda_j g_j$ ซึ่งจะเป็น F' ตัวใหม่ของเงื่อนไขจำเป็น (Necessary Conditions) เพื่อใช้สำหรับเอ็กทรีมัม (Extremum) ของฟังก์ชันนอล (Functional) นั้นเอง โดยจะนำไปใช้กับลักษณะของเวลาและจุดสุดท้ายทั้งสามแบบดังที่กล่าวมาแล้ว นั่นก็คือ กำหนดจุดสุดท้ายและเวลาสุดท้ายไว้ตายตัว (Fixed End Time and End Points), กำหนดเวลาสุดท้ายไว้ตายตัวแต่จุดสุดท้ายแปรผันได้ (Fixed End Time, Variable End Points) และเวลาสุดท้ายและจุดสุดท้ายแปรผันได้ (Variable End Time and End Points) ซึ่งกระทำได้เพียงแค่เปลี่ยนจาก F เป็น F' ก็สามารถทำการหาเงื่อนไขจำเป็น (Necessary Conditions) ซึ่ง $\delta J' = 0$ ได้เช่นกัน โดยที่ $g_j = 0$ ก็ทำให้สามารถเอ็กทรีมัม (Extremum) ได้

2. ฟังก์ชันเงื่อนไขบังคับที่จุดสุดท้าย (End Point Function Value Constraints)

เอ็กทรีมัมของฟังก์ชันนอล (Functional) $J[x_1, \dots, x_n]$ ซึ่งฟังก์ชัน $x_i(t_f)$ จะต้องสอดคล้องกับเงื่อนไขบังคับ (Constraints) $s_k(t, x_1, \dots, x_n)_{t_f} = 0, k = 1, \dots, p$ ณ เวลาสุดท้าย การหาผลเฉลยของปัญหานี้จะสามารถทำการเขียนฟังก์ชันนอล (Functional) ใหม่ได้เป็น

$$J'[x_1, \dots, x_n] = \int_{t_0}^{t_f} F(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n) dt + \sum_{k=1}^p \nu_k s_k \quad (2.33)$$

เมื่อ ν_k คือค่าคงตัวคูณลากรางจ์ (Constant Lagrange multiplier) ทำให้สามารถดัดแปลงสมการ (2.33) ได้เป็น

$$\begin{aligned} \delta J' = & \int_{t_0}^{t_f} \sum_{i=1}^n \left(\frac{\partial F}{\partial x_i} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}_i} \right) h_i dt + \sum_{i=1}^n \left[\left\{ \frac{\partial F}{\partial \dot{x}_i} + \sum_{k=1}^p \nu_k \frac{\partial s_k}{\partial \dot{x}_i} \right\} \delta x_i \right]_{t_f} \\ & - \sum_{i=1}^n \left(\frac{\partial F}{\partial \dot{x}_i} \delta x_i \right) \Big|_{t_0} + \left[F - \sum_{i=1}^n \frac{\partial F}{\partial \dot{x}_i} \dot{x}_i + \sum_{k=1}^p \nu_k \frac{\partial s_k}{\partial t} \right]_{t_f} \delta t_f \\ & - \left[F - \sum_{i=1}^n \frac{\partial F}{\partial \dot{x}_i} \dot{x}_i \right]_{t_0} \delta t_0 \end{aligned} \quad (2.32)$$

เมื่อเงื่อนไขจำเป็น (Necessary condition) ก็คือ $\delta J' = 0$ ที่ $h_i(t)$, $\delta x_i|_{t_f}$, $\delta x_i|_{t_0}$, δt_f และ δt_0 จึงจะสอดคล้องกับเงื่อนไขขอบเขต (Boundary Conditions) จะได้เงื่อนไขจำเป็น (Necessary Conditions) เป็น

$$\frac{\partial F}{\partial x_i} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}_i} = 0, i = 1, \dots, n \quad (2.33)$$

โดยที่ $s_k = 0$ จึงจะทำให้ได้คำตอบที่เป็นที่ยอมรับได้ (Feasible) และมีเงื่อนไขขอบเขตเป็น

$$\begin{aligned} \left[F - \sum_{k=1}^p \nu_k \frac{\partial s_k}{\partial x_i} \right]_{t_f} &= 0; \quad \frac{\partial F}{\partial \dot{x}_i} \Big|_{t_0} = 0 \quad i = 1, \dots, n \\ \left[F - \sum_{i=1}^n \frac{\partial F}{\partial \dot{x}_i} \dot{x}_i + \sum_{k=1}^p \nu_k \frac{\partial s_k}{\partial t} \right]_{t_f} &= 0; \quad \left[F - \sum_{i=1}^n \frac{\partial F}{\partial \dot{x}_i} \dot{x}_i \right]_{t_0} = 0 \end{aligned} \quad (2.34)$$

3. ฟังก์ชันเงื่อนไขบังคับทั่วไป (General Constraints)

จากหัวข้อ (1. และ 2.) เราสามารถเอ็กซ์ตรีมัม (Extremum) ของฟังก์ชันนอล

(Functional) $J[x_1, \dots, x_n] = \int_{t_0}^{t_f} F(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n) dt$ ได้โดยที่ฟังก์ชัน $x_i(t)$ สอดคล้องกับ

ฟังก์ชันแบบเงื่อนไขแบบเท่ากัน (Equality Constraints) $g_j(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n) = 0$

เมื่อ $j = 1, \dots, m$ โดย $m < n$ เสมอ และสอดคล้องกับเงื่อนไขบังคับที่จุดสุดท้าย (End point

constraints) $s_k(t, x_1, \dots, x_n)_{t_f} = 0, k = 1, \dots, p$ โดย $p < m$ เสมอ ดังนั้นเมื่อใช้เทคนิควิธีตัว

คูณลากรางจ์ (Lagrange Multiplier) ก็จะได้ฟังก์ชันนอล (Functional) ใหม่เป็น

$$J'[x_1, \dots, x_n] = \int_{t_0}^{t_f} \left[F(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n) + \sum_{j=1}^m \lambda_j(t) g_j(t, x_1, \dots, x_n, \dot{x}_1, \dots, \dot{x}_n) \right] dt + \sum_{k=1}^p \nu_k s_k \quad (2.35)$$

เมื่อ $\lambda_j(t)$ และ ν_k คือตัวคูณลากรางจ์ (Lagrange Multiplier) ดังนั้นหากกรณี

ปัญหาเป็นแบบเวลาสุดท้ายและจุดสุดท้ายแปรผันได้ (Variable End Time and End Points) ก็จะสามารถดัดแปลงสมการ (2.29) ได้เป็น

$$\begin{aligned} \delta J' = & \int_{t_0}^{t_f} \sum_{i=1}^n \left(\frac{\partial F'}{\partial x_i} - \frac{d}{dt} \frac{\partial F'}{\partial \dot{x}_i} \right) \delta x_i dt \\ & + \sum_{i=1}^n \left[\left\{ \frac{\partial F'}{\partial \dot{x}_i} + \sum_{k=1}^p \nu_k \frac{\partial s_k}{\partial \dot{x}_i} \right\} \delta x_i \right]_{t_f} - \sum_{i=1}^n \left(\frac{\partial F'}{\partial \dot{x}_i} \delta x_i \right) \Big|_{t_0} \\ & + \left[F' - \sum_{i=1}^n \frac{\partial F'}{\partial \dot{x}_i} \dot{x}_i + \sum_{k=1}^p \nu_k \frac{\partial s_k}{\partial t} \right]_{t_f} \delta t_f \\ & - \left[F' - \sum_{i=1}^n \frac{\partial F'}{\partial \dot{x}_i} \dot{x}_i \right]_{t_0} \delta t_0 \end{aligned} \quad (2.36)$$

เมื่อ $F' = F + \sum_{j=1}^m \lambda_j g_j$ ทำให้ได้เงื่อนไขจำเป็น (Necessary Conditions)

สำหรับเอ็กซ์ตรีมัม (Extremum) ดังนี้

$$\begin{aligned}
\frac{\partial F'}{\partial \dot{x}_i} - \frac{d}{dt} \frac{\partial F'}{\partial \dot{x}_i} &= 0 \\
\left[\frac{\partial F'}{\partial \dot{x}_i} + \sum_{k=1}^p \nu_k \frac{\partial s_k}{\partial \dot{x}_i} \right]_{t_f} &= 0 \\
\frac{\partial F'}{\partial \dot{x}_i} \Big|_{t_0} &= 0, i = 1, \dots, n \\
\left[F' - \sum_{i=1}^n \frac{\partial F'}{\partial \dot{x}_i} \dot{x}_i + \sum_{k=1}^p \nu_k \frac{\partial s_k}{\partial t} \right]_{t_f} &= 0 \\
\left[F' - \sum_{i=1}^n \frac{\partial F'}{\partial \dot{x}_i} \dot{x}_i \right]_{t_0} &= 0
\end{aligned} \tag{2.37}$$

โดยที่ $g_j = 0, j = 1, \dots, m$ และ $s_k = 0, k = 1, \dots, p$ จึงจะทำให้ได้คำตอบที่เป็นที่ยอมรับได้

3.1.4 การตรวจสอบเอ็กทรีมัมว่าเป็นค่าน้อยที่สุดหรือมากที่สุด (Sufficient Conditions for Minimum and Maximum)

จากหัวข้อที่ผ่านมาเป็นการหาเงื่อนไขจำเป็น (Necessary Conditions) สำหรับใช้เอ็กทรีมัม (Extremum) เท่านั้น ซึ่งยังไม่บอกไม่ได้ว่าเป็นค่าน้อยที่สุดหรือมากที่สุด ในหัวข้อนี้จะเป็นการตรวจสอบเพื่อบ่งชี้ว่าเอ็กทรีมัม (Extremum) เป็นค่าใดกันแน่ ในที่นี้จะกล่าวเฉพาะการตรวจหาค่าน้อยที่สุดเท่านั้น เพราะการตรวจหาค่ามากที่สุดก็ใช้หลักการเดียวกัน เรามาพิจารณาปัญหาการหาค่า-สเกลาร์ $x(t)$ ที่ทำให้ฟังก์ชันนอล (Functional)

$J[x(t)] = \int_{t_0}^{t_f} F(x, \dot{x}, t) dt$ นี้มีค่าน้อยที่สุด

เมื่อ t_0 และ t_f ถูกกำหนดไว้ตายตัว และ F สามารถทำอนุพันธ์อย่างต่อเนื่องได้อย่างน้อยสองครั้งเทียบกับ $x, \dot{x}$ และ t ต่อไปหากเรากำหนดให้ $x(t)$ เป็นค่าที่ทำให้ฟังก์ชันมีค่าน้อยที่สุด ต่อไปลองทำการเพิ่มค่า $x(t)$ ด้วย $h(t_0) = h(t_f) = 0$ เราจะพบการเปลี่ยนแปลงของฟังก์ชันต้นทุน (Cost Function) ที่สอดคล้องกับการเพิ่มค่า $h(t)$ เป็น $\Delta J = J[x+h] - J[x]$ แล้วใช้อนุกรมเทย์เลอร์ (Taylor Series) กระจายเทอมก็จะได้

$$\Delta J = \delta J + \delta^2 J + \text{high-order-terms} \tag{2.38}$$

เมื่อ

$$\delta J = \int_{t_0}^{t_f} \left(\frac{\partial F}{\partial x} - \frac{d}{dt} \frac{\partial F}{\partial \dot{x}} \right) h dt + \frac{\partial F}{\partial \dot{x}} h \Big|_{t_f} - \frac{\partial F}{\partial \dot{x}} h \Big|_{t_0}$$

$$\delta^2 J = \frac{1}{2} \int_{t_0}^{t_f} \left(h^2 \frac{\partial^2 F}{\partial x^2} + 2h\dot{h} \frac{\partial^2 F}{\partial x \partial \dot{x}} + \dot{h}^2 \frac{\partial^2 F}{\partial \dot{x}^2} \right) dt$$

ซึ่งจากในหัวข้อที่ผ่านมา มา เราพบว่า $x(t)$ จะเป็นค่าน้อยที่สุดก็ต่อเมื่อ $\Delta J = \delta^2 J > 0$ (ค่ามากที่สุดก็คือ $\Delta J = \delta^2 J < 0$) และเราสามารถเขียนให้อยู่ในรูปแบบอย่างง่ายได้เป็น

$$\delta^2 J = \frac{1}{2} \int_{t_0}^{t_f} \begin{bmatrix} h(t) & \dot{h}(t) \end{bmatrix} \tilde{F} \begin{bmatrix} h(t) \\ \dot{h}(t) \end{bmatrix} dt \quad (2.39)$$

เมื่อ

$$\tilde{F} = \begin{bmatrix} \frac{\partial^2 F}{\partial x^2} & \frac{\partial^2 F}{\partial x \partial \dot{x}} \\ \frac{\partial^2 F}{\partial x \partial \dot{x}} & \frac{\partial^2 F}{\partial \dot{x}^2} \end{bmatrix}$$

ซึ่งถ้าค่าของ $\tilde{F}$ เป็นบวก (Positive definite) ก็จะทำให้ $x(t)$ เป็นค่าน้อยที่สุด แต่หากค่า $\tilde{F}$ เป็นลบ (Negative definite) ก็จะทำให้ $x(t)$ เป็นค่ามากที่สุด

3.2 การแก้ปัญหาความเหมาะสมสูงสุดพลศาสตร์ด้วยแคลคูลัสอพอวาไรชันส์ (Calculus of Variation with Dynamic Optimization)

จากที่กล่าวมาแล้วในส่วนของแคลคูลัสอพอวาไรชันส์ (Calculus of Variations) นั้นพบว่าประโยชน์อย่างมากกับการนำมาแก้ปัญหาทางด้านความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization) ดังนั้นต่อไปนี้จะมาดูวิธีการนำเอาแคลคูลัสอพอวาไรชันส์ (Calculus of Variations) มาใช้แก้ปัญหาความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization) ดังนี้

3.2.1 เวลาสุดท้ายถูกกำหนดไว้ตายตัว (Fixed Final Time)

ลักษณะเช่นนี้ถือว่าเป็นลักษณะที่ถือว่าธรรมดาทั่วไปแต่ต้องเป็นระบบที่เข้าข่ายดังนี้คือ มีรูปแบบการระบุปัญหาโดยสมการ (2.1), มีลักษณะของฟังก์ชันต้นทุน (Cost Function) ดังสมการ (2.2), เวลาเริ่มต้นและเวลาสุดท้ายคือ t_0 และ t_f ถูกกำหนดไว้ตายตัว (Fixed End Time) และระบุสถานะเริ่มต้นไว้แล้ว $x_1(t_0), \dots, x_n(t_f)$ ตามลำดับ ตัวอย่างเช่น รถไฟที่วิ่งระหว่างเมืองสองเมืองที่มีกำหนดเวลาออกและเวลาถึงสถานีไว้ตายตัวแล้ว ซึ่งฟังก์ชันวัตถุประสงค์ (Objective Function) อาจจะเป็นการสิ้นเปลืองพลังงานเชื้อเพลิงที่น้อยที่สุดก็ได้ ซึ่งเราจะกล่าวถึงลักษณะของปัญหาที่พบในงานด้านความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization) 3 ลักษณะด้วยกัน คือ สภาวะสุดท้าย $x_1(t_f), \dots, x_n(t_f)$ ถูกกำหนดไว้ตายตัว, สภาวะสุดท้าย $x_1(t_f), \dots, x_n(t_f)$ ถูกกำหนดไว้ตายตัวแต่มีเงื่อนไขบังคับ (Constraints) และสุดท้ายคือคอนโทรลวาริเอเบิลและสเตตวาริเอเบิล (Control Variables and State Variables) ต้องสอดคล้องกับเงื่อนไขบังคับตลอดเวลา ดังรายละเอียดต่อไปนี้

1. สภาวะสุดท้ายถูกกำหนดไว้ตายตัว (Final States are Prescribed)

ปัญหานี้มีการกำหนดเวลาและจุดสุดท้ายไว้ตายตัว แล้วทำให้เทอม $\Phi(t, x_1, \dots, x_n)$ ในสมการ (2.2) ไม่มีความเกี่ยวข้องกับทฤษฎีความเหมาะสมสูงสุด (Optimization) ลักษณะของปัญหานี้จะคล้ายกับหัวข้อ (2.3.1.3.1) ทำให้เราได้ฟังก์ชันนอล (Functional) เป็น

$$J[x_1, \dots, x_n, u_1, \dots, u_m] = \int_{t_0}^{t_f} L'(t, x_1, \dots, x_n, u_1, \dots, u_m) dt \quad (2.40)$$

เมื่อ

$$L'(t, x_1, \dots, x_n, u_1, \dots, u_m) = L(t, x_1, \dots, x_n, u_1, \dots, u_m) + \sum_{i=1}^n \lambda_i(t) [f_i(t, x_1, \dots, x_n, u_1, \dots, u_m) - \dot{x}_i]$$

เมื่อ $\lambda_i(t)$ คือตัวคูณลากรางจ์ (Lagrange Multipliers) ดังนั้นมีการเปลี่ยนแปลงของฟังก์ชันดังนี้

$$\begin{aligned}
\delta J' = & \int_{t_0}^{t_f} \sum_{j=1}^n \left(\frac{\partial L'}{\partial x_j} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{x}_j} \right) h_{x_j} dt \\
& + \int_{t_0}^{t_f} \sum_{k=1}^m \left(\frac{\partial L'}{\partial u_k} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{u}_k} \right) h_{u_k} dt \\
& + \sum_{k=1}^m \left(\frac{\partial L'}{\partial \dot{u}_k} h_{u_k} \Big|_{t_f} - \frac{\partial L'}{\partial \dot{u}_k} h_{u_k} \Big|_{t_0} \right) \\
& + \sum_{j=1}^n \left(\frac{\partial L'}{\partial \dot{x}_j} h_{x_j} \Big|_{t_f} - \frac{\partial L'}{\partial \dot{x}_j} h_{x_j} \Big|_{t_0} \right)
\end{aligned} \tag{2.41}$$

แต่ถ้ามีการกำหนดสภาวะสุดท้ายไว้ตายตัวแล้วที่ t_0 และ t_f , $h_{x_j} \Big|_{t_f} = h_{x_j} \Big|_{t_0} = 0$
 ทำให้เราเขียนเทอมต่างๆ ของสมการ (2.41) แยกส่วนย่อยได้เป็น

$$\begin{aligned}
\frac{\partial L'}{\partial x_j} &= \frac{\partial L}{\partial x_j} + \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial x_j}, \quad \frac{\partial L'}{\partial \dot{x}_j} = -\lambda_j, \\
\frac{\partial L'}{\partial u_k} &= \frac{\partial L}{\partial u_k} + \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial u_k}, \quad \frac{\partial L'}{\partial \dot{u}_k} = 0
\end{aligned} \tag{2.42}$$

เมื่อทำการเปลี่ยนแปลงค่าของ h_{x_j} และ h_{u_k} , $\delta J' = 0$ ถ้า $\frac{\partial L'}{\partial x_j} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{x}_j} = 0$ และ $\frac{\partial L'}{\partial u_k} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{u}_k} = 0$ สำหรับ $j=1, \dots, n$ และ $k=1, \dots, m$ ด้วยการแทนสมการ (2.42) กลับเข้าไปในสมการดังกล่าวทำให้เราได้เงื่อนไขจำเป็น (Necessary Conditions) เป็น

$$\dot{\lambda}_j = -\frac{\partial L}{\partial x_j} - \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial x_j}, \quad j=1, \dots, n \tag{2.43}$$

$$\frac{\partial L}{\partial u_k} + \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial u_k} = 0, \quad k=1, \dots, m \tag{2.44}$$

ผลเฉลยของสภาวะ (States) $x_j(t)$, ตัวคูณลากรางจ์ (Lagrange Multipliers) $\lambda_j(t)$ (บางครั้งเรียกโคสเตต (Costates)) และตัวควบคุมนำเข้า (Control Inputs) $u_k(t)$ สามารถ

หาค่าได้โดยใช้สมการสถานะ (State Equations) คือสมการ (2.1) จำนวน n สมการ, สมการสถานะร่วม (Costate Equations) คือสมการ (2.43) จำนวน n สมการ และสมการเพิ่มเติม (Additional Optimality Equations) คือสมการ (2.44) จำนวน m สมการ ทำให้มีสมการเชิงอนุพันธ์อันดับหนึ่ง (First Order Different Equations) อยู่จำนวน $2n$ สมการ และมีสมการสามัญ (Ordinary Equations) จำนวน m สมการ ในการหาผลเฉลยนี้มีความจำเป็นจะต้องมีการกำหนดเงื่อนไขขอบเขต (Boundary Conditions) จำนวน $2n$ เงื่อนไข และต้องกำหนดสถานะที่เวลา t_0 และ t_f ด้วย ซึ่งสมการเชิงอนุพันธ์ (Different Equations) ที่มีการระบุเงื่อนไขขอบเขต (Boundary Conditions) ที่เวลาต่างกันสองเวลาแบบนี้จะมีการเรียกชื่อว่าปัญหาเงื่อนไขขอบเขตสองจุด

(TPBVP: Two Point Boundary Value Problems) ส่วน
$$H = L + \sum_{i=1}^n \lambda_i f_i$$
 เรียกว่าฮามิลโตเนียนของระบบ (Hamiltonian of System)

2. สถานะสุดท้ายต้องเป็นไปตามเงื่อนไขบังคับ (Final States Lie On a Constraint Surface) ปัญหานี้เขียนในเทอมทางคณิตศาสตร์ได้เป็น $s_l(t, x_1, \dots, x_n)_{t_f} = 0$ เมื่อ $l = 1, \dots, p$ ปัญหานี้มีรูปแบบคล้ายกับหัวข้อ (2.3.1.3.2) ทำให้เราได้ฟังก์ชันนอล (Functional) เป็น

$$\begin{aligned} J[x_1, \dots, x_n, u_1, \dots, u_m] = & \Phi(t, x_1, \dots, x_n)_{t_f} \\ & + \sum_{l=1}^p v_l s_l(t, x_1, \dots, x_n)_{t_f} \\ & + \int_{t_0}^{t_f} L'(t, x_1, \dots, x_n, u_1, \dots, u_m) dt \end{aligned} \quad (2.45)$$

เมื่อ v_l ก็คือค่าคงตัวตัวคูณลากรางจ์ (Constant Lagrange Multipliers) และ L' เขียนเป็น

$$\begin{aligned} L'(t, x_1, \dots, x_n, u_1, \dots, u_m) = & L(t, x_1, \dots, x_n, u_1, \dots, u_m) \\ & + \sum_{i=1}^n \lambda_i(t) [f_i(t, x_1, \dots, x_n, u_1, \dots, u_m) - \dot{x}_i] \end{aligned} \quad (2.46)$$

เมื่อ $\lambda_i(t)$ คือตัวคูณลากรางจ์ (Lagrange Multipliers) ซึ่งเป็นฟังก์ชันของเวลา เมื่อทำการกำหนดเวลาสุดท้ายไว้ตายตัวดังนั้นจะพบการเปลี่ยนแปลงของฟังก์ชันดังนี้

$$\begin{aligned} \delta J' = & \int_{t_0}^{t_f} \sum_{j=1}^n \left(\frac{\partial L'}{\partial x_j} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{x}_j} \right) h_{x_j} dt \\ & + \int_{t_0}^{t_f} \sum_{k=1}^m \left(\frac{\partial L'}{\partial u_k} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{u}_k} \right) h_{u_k} dt \\ & + \sum_{k=1}^m \left(\frac{\partial L'}{\partial \dot{u}_k} h_{u_k} \Big|_{t_f} - \frac{\partial L'}{\partial \dot{u}_k} h_{u_k} \Big|_{t_0} \right) \\ & + \sum_{j=1}^n \left(\frac{\partial L'}{\partial \dot{x}_j} \delta x_j \right) \Big|_{t_f} + \sum_{j=1}^n \left(\frac{\partial \Phi}{\partial x_j} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial x_j} \right) \Big|_{t_f} h_{x_j} \Big|_{t_f} \end{aligned} \quad (2.47)$$

เมื่อทำการเปลี่ยนแปลงค่าของ h_{x_j} , h_{u_k} และค่าขอบ (Boundary Values) $h_{x_j} \Big|_{t_f}$ จะทำให้ $\frac{\partial L'}{\partial \dot{u}_k} = 0$, ค่าการเปลี่ยนแปลง $\delta J' = 0$ ถ้า $\frac{\partial L'}{\partial x_j} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{x}_j} = 0$, $\frac{\partial L'}{\partial u_k} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{u}_k} = 0$ และ $\left[\frac{\partial L'}{\partial \dot{x}_j} + \frac{\partial \Phi}{\partial x_j} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial x_j} \right]_{t_f} = 0$ ทำให้เราได้เงื่อนไขจำเป็น (Necessary Conditions) เป็น

$$\dot{\lambda}_j = -\frac{\partial L}{\partial x_j} - \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial x_j}, \quad j = 1, \dots, n \quad (2.48)$$

$$\frac{\partial L}{\partial u_k} + \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial u_k} = 0, \quad k = 1, \dots, m \quad (2.49)$$

$$\lambda_j \Big|_{t_f} = \left[\frac{\partial \Phi}{\partial x_j} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial x_j} \right]_{t_f}, \quad j = 1, \dots, n \quad (2.50)$$

ผลเฉลยของสภาวะ (States) $x_j(t)$, ตัวคูณลากรางจ์ (Lagrange Multipliers) $\lambda_j(t)$ และตัวควบคุมนำเข้า (Control Inputs) $u_k(t)$ สามารถหาค่าได้โดยใช้สมการสภาวะ (State Equations) คือสมการ (2.50) จำนวน n สมการ, สมการสภาวะร่วม (Costate Equations)

คือสมการ (2.48) จำนวน n สมการ และสมการเพิ่มเติม (Additional Optimality Equations) คือสมการ (2.49) จำนวน m สมการ ทำให้มีสมการเชิงอนุพันธ์อันดับหนึ่ง (First Order Different Equations) อยู่จำนวน $2n$ สมการ และมีสมการสามัญ (Ordinary Equations) จำนวน m สมการ ในการหาผลเฉลยนี้มีความจำเป็นจะต้องมีการกำหนดเงื่อนไขขอบเขต (Boundary Conditions) จำนวน $2n$ เงื่อนไข และต้องกำหนดสภาวะสุดท้าย $x_j(t_f)$ ด้วย ส่วนสภาวะร่วม (Costate) $\lambda_j(t_f)$ สามารถคำนวณได้จากสมการ (2.50) ส่วนค่า v_l สามารถหาได้โดยการแทนค่าในผลเฉลย ณ t_f ในสมการ $s_l|_{t_f} = 0$

3. เงื่อนไขบังคับสนับสนุน (Auxiliary Constraints)

สำหรับกรณีปัญหาที่ตัวแปรสถานะ (State Variables) และตัวแปรควบคุม (Control Variables) จำเป็นจะต้องสอดคล้องกับเงื่อนไขบังคับ (Constraints) ซึ่งเขียนในเทอมทางคณิตศาสตร์เป็น $g_r(t, x_1, \dots, x_n, u_1, \dots, u_m)|_{t_f} = 0$ และ $r = 1, \dots, p'$ โดยอยู่ในช่วงระหว่าง t_0 และ t_f ซึ่งที่สภาวะสุดท้าย จะต้องเป็นไปตามเงื่อนไขที่เขียนในเทอมทางคณิตศาสตร์ เป็น $s_l(t, x_1, \dots, x_n)|_{t_f} = 0$ และ $l = 1, \dots, p$ ทำให้เราได้ฟังก์ชันนอล (Functional) เป็น

$$\begin{aligned} J'[x_1, \dots, x_n, u_1, \dots, u_m] = & \Phi(t, x_1, \dots, x_n)|_{t_f} \\ & + \sum_{l=1}^p v_l s_l(t, x_1, \dots, x_n)|_{t_f} \\ & + \int_{t_0}^{t_f} L'(t, x_1, \dots, x_n, u_1, \dots, u_m) dt \end{aligned} \quad (2.51)$$

เมื่อ v_l ก็คือค่าคงตัวคูณลากรางจ์ (Constant Lagrange Multipliers) และ L' เขียนเป็น

$$\begin{aligned} L'(t, x_1, \dots, x_n, u_1, \dots, u_m) = & L(t, x_1, \dots, x_n, u_1, \dots, u_m) \\ & + \sum_{i=1}^n \lambda_i(t) [f_i(t, x_1, \dots, x_n, u_1, \dots, u_m) - \dot{x}_i] \\ & + \sum_{r=1}^{p'} \mu_r(t) g_r(t, x_1, \dots, x_n, u_1, \dots, u_m) \end{aligned} \quad (2.52)$$

เมื่อ $\lambda_i(t)$ และ $\mu_r(t)$ คือตัวคูณลากรางจ์ (Lagrange Multipliers) ซึ่งเป็นฟังก์ชันของเวลา เมื่อทำการกำหนดเวลาสุดท้ายไว้ตายตัวดังนั้นจะพบการเปลี่ยนแปลงของฟังก์ชันดังนี้

$$\begin{aligned} \delta J' = & \int_{t_0}^{t_f} \sum_{j=1}^n \left(\frac{\partial L'}{\partial x_j} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{x}_j} \right) h_{x_j} dt \\ & + \int_{t_0}^{t_f} \sum_{k=1}^m \left(\frac{\partial L'}{\partial u_k} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{u}_k} \right) h_{u_k} dt \\ & + \sum_{k=1}^m \left(\frac{\partial L'}{\partial \dot{u}_k} h_{u_k} \Big|_{t_f} - \frac{\partial L'}{\partial \dot{u}_k} h_{u_k} \Big|_{t_0} \right) \\ & + \sum_{j=1}^n \left(\frac{\partial L'}{\partial \dot{x}_j} h_{x_j} \right) \Big|_{t_f} + \sum_{j=1}^n \left(\frac{\partial \Phi}{\partial x_j} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial x_j} \right) \Big|_{t_f} h_{x_j} \Big|_{t_f} \end{aligned} \quad (2.53)$$

เมื่อทำการเปลี่ยนแปลงค่าของ h_{x_j} , h_{u_k} และค่าขอบ (Boundary Values) $h_{x_j} \Big|_{t_f}$ จะทำให้เกิดค่าการเปลี่ยนแปลงของ $\delta J' = 0$ ถ้า $\frac{\partial L'}{\partial x_j} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{x}_j} = 0$, $\frac{\partial L'}{\partial u_k} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{u}_k} = 0$ และ

$$\left[\frac{\partial L'}{\partial \dot{x}_j} + \frac{\partial \Phi}{\partial x_j} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial x_j} \right]_{t_f} = 0$$

ทำให้เราได้เงื่อนไขจำเป็น (Necessary Conditions) เป็น

$$\dot{\lambda}_j = -\frac{\partial L}{\partial x_j} - \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial x_j} - \sum_{l=1}^{p'} \mu_r \frac{\partial g_r}{\partial x_j}, \quad j = 1, \dots, n \quad (2.54)$$

$$\frac{\partial L}{\partial u_k} + \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial u_k} + \sum_{l=1}^{p'} \mu_r \frac{\partial g_r}{\partial u_k} = 0, \quad k = 1, \dots, m \quad (2.55)$$

$$\lambda_j \Big|_{t_f} = \left[\frac{\partial \Phi}{\partial x_j} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial x_j} \right]_{t_f}, \quad j = 1, \dots, n \quad (2.56)$$

ผลเฉลยของสถานะ (States) $x_j(t)$, สถานะร่วม (Costate) $\lambda_j(t)$, ตัวควบคุมนำเข้า (Control Inputs) $u_k(t)$, ตัวคูณลากรางจ์ (Lagrange Multipliers) $\mu_l(t)$ และ ν_l สามารถหาค่า

ได้โดยการใช้สมการสถานะ (State Equations) คือสมการ (2.50) จำนวน n สมการ, สมการสถานะร่วม (Costate Equations) คือสมการ (2.54) จำนวน n สมการ และสมการเพิ่มเติม (Additional Optimality Equation) คือสมการ (2.55) จำนวน m สมการ, สมการเงื่อนไขบังคับ (Constraint Equations) $g_r(t, x_1, \dots, x_n, u_1, \dots, u_m)_{t_f} = 0$ จำนวน p' สมการ และสมการเงื่อนไขบังคับ (Constraint Equations) ที่จุดสุดท้าย (End-Point Constraint Equations) จำนวน P สมการ ทำให้มีสมการเชิงอนุพันธ์ (Different Equations) อยู่จำนวน $2n$ สมการ และมีสมการสามัญ (Ordinary Equations) จำนวน $m + p' + p$ สมการ มีเงื่อนไขขอบเขต (Boundary Conditions) จำนวน $2n$ เงื่อนไข และต้องกำหนดสถานะที่ t_0 และสถานะร่วม (Costate) $\lambda_j(t_f)$ สามารถคำนวณได้จากสมการ (2.56)

3.2.2 เวลาสุดท้ายแปรผัน (Variable Final Time)

ลักษณะเช่นนี้มีลักษณะคล้ายกับหัวข้อ (2.3.2.1) เพียงแต่ที่เวลาสุดท้าย t_f แปรผันได้ แต่ต้องมีรูปแบบการระบุปัญหาโดยสมการ (2.50), มีลักษณะของฟังก์ชันต้นทุน (Cost Functional) ดังสมการ (2.51), เวลาเริ่มต้นคือ t_0 ถูกกำหนดไว้ตายตัว (Fixed Start Time) แต่เวลาสุดท้าย t_f มีการแปรผัน (Variable End Time) และมีการระบุสถานะเริ่มต้นไว้แล้ว $x_1(t_0), \dots, x_n(t_0)$ ตามลำดับ เช่น การแข่งรถเป็นต้น จะกล่าวถึงลักษณะของปัญหาที่พบในงานด้านความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization) 3 ลักษณะด้วยกัน คือ สถานะสุดท้าย $x_1(t_f), \dots, x_n(t_f)$ ถูกกำหนดไว้ตายตัว, สถานะสุดท้าย $x_1(t_f), \dots, x_n(t_f)$ ถูกกำหนดไว้ให้เป็นไปตามเงื่อนไขบังคับ (Constraints) และสุดท้ายคือตัวแปรสถานะ (State Variables) และตัวแปรควบคุม (Control Variables) ต้องสอดคล้องกับเงื่อนไขบังคับตลอดเวลาการเคลื่อนที่ ดังรายละเอียดต่อไปนี้

1. สถานะสุดท้ายถูกกำหนดไว้ตายตัว (Final States are Prescribed) ปัญหานี้มีฟังก์ชันนอล (Functional) เป็น

$$J[x_1, \dots, x_n, u_1, \dots, u_m] = \Phi(t, x_1, \dots, x_n)_{t_f} + \int_{t_0}^{t_f} L(t, x_1, \dots, x_n, u_1, \dots, u_m) dt \quad (2.57)$$

เมื่อ

$$L'(t, x_1, \dots, x_n, u_1, \dots, u_m) = L(t, x_1, \dots, x_n, u_1, \dots, u_m) + \sum_{i=1}^n \lambda_i(t) [f_i(t, x_1, \dots, x_n, u_1, \dots, u_m) - \dot{x}_i]$$

เมื่อ $\lambda_i(t)$ คือตัวคูณลากรางจ์ (Lagrange Multipliers) ด้วยการกำหนดเวลาและสถานะเริ่มต้น รวมทั้งระบุสถานะสุดท้ายไว้ ดังนั้นจะพบการเปลี่ยนแปลงของฟังก์ชันดังนี้

$$\begin{aligned} \delta J' = & \int_{t_0}^{t_f} \sum_{j=1}^n \left(\frac{\partial L'}{\partial x_j} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{x}_j} \right) h_{x_j} dt \\ & + \int_{t_0}^{t_f} \sum_{k=1}^m \left(\frac{\partial L'}{\partial u_k} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{u}_k} \right) h_{u_k} dt \\ & + \sum_{k=1}^m \left(\frac{\partial L'}{\partial \dot{u}_k} \delta u_k \Big|_{t_f} - \frac{\partial L'}{\partial \dot{u}_k} \delta u_k \Big|_{t_0} \right) \\ & + \left[\frac{\partial \Phi}{\partial t} + L' - \sum_{j=1}^n \dot{x}_j \frac{\partial L'}{\partial \dot{x}_j} - \sum_{k=1}^m \dot{u}_k \frac{\partial L'}{\partial \dot{u}_k} \right]_{t_f} \delta t_f \end{aligned} \quad (2.58)$$

เมื่อเราทำการประเมินค่าแต่ละส่วนในสมการ (2.58) ก็จะทำให้เราสามารถหาเงื่อนไขจำเป็น (Necessary Conditions) ได้ดังนี้

$$\dot{\lambda}_j = -\frac{\partial L}{\partial x_j} - \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial x_j}, \quad j = 1, \dots, n \quad (2.59)$$

$$\frac{\partial L}{\partial u_k} + \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial u_k} = 0, \quad k = 1, \dots, m \quad (2.60)$$

$$\left[\frac{\partial \Phi}{\partial t} + L + \sum_{i=1}^n \lambda_i f_i \right]_{t_f} = 0 \quad (2.61)$$

ผลเฉลยของสถานะ (States) $x_j(t)$, ตัวคูณลากรางจ์ (Lagrange Multipliers) $\lambda_j(t)$ (บางครั้งเรียกสถานะร่วม (Costates)), ตัวควบคุมนำเข้า (Control Inputs) $u_k(t)$ และเวลาสุดท้าย t_f สามารถหาค่าได้โดยใช้สมการสถานะ (State Equations) คือสมการ (2.50)

จำนวน n สมการ, สมการสภาวะร่วม (Costate equations) คือสมการ (2.59) จำนวน n สมการ และสมการเพิ่มเติม (Additional Optimality Equations) คือสมการ (2.60) จำนวน m สมการและสมการเงื่อนไขขอบเขต (Boundary Conditions) คือสมการ (2.61) ทำให้มีสมการเชิงอนุพันธ์อันดับหนึ่ง (First Order Different Equations) อยู่จำนวน $2n$ สมการ และมีสมการสามัญ (Ordinary Equations) จำนวน $m+1$ สมการ ในการหาผลเฉลยนี้มีความจำเป็นจะต้องมีการกำหนดเงื่อนไขขอบเขต (Boundary Conditions) จำนวน $2n$ เงื่อนไข และต้องกำหนดสภาวะที่เวลา t_0 และ t_f ด้วย

2. สภาวะสุดท้ายต้องเป็นไปตามเงื่อนไขบังคับ (Final States Lie On a Constraint Surface) ปัญหานี้ให้อยู่ในเทอมทางคณิตศาสตร์ได้เป็น $s_l(t, x_1, \dots, x_n)_{t_f} = 0$ เมื่อ $l = 1, \dots, p$ ปัญหานี้มีรูปแบบคล้ายกับ (2.3.2.1.2) ทำให้เราได้ฟังก์ชันนอล (Functional) เป็น

$$J[x_1, \dots, x_n, u_1, \dots, u_m] = \Phi(t, x_1, \dots, x_n)_{t_f} + \sum_{l=1}^p \nu_l s_l(t, x_1, \dots, x_n)_{t_f} + \int_{t_0}^{t_f} L'(t, x_1, \dots, x_n, u_1, \dots, u_m) dt \quad (2.62)$$

เมื่อ ν_l ก็คือค่าคงตัวตัวคูณลากรางจ์ (Constant Lagrange Multipliers) และ L' เขียนเป็น

$$L'(t, x_1, \dots, x_n, u_1, \dots, u_m) = L(t, x_1, \dots, x_n, u_1, \dots, u_m) + \sum_{i=1}^n \lambda_i(t) [f_i(t, x_1, \dots, x_n, u_1, \dots, u_m) - \dot{x}_i] \quad (2.63)$$

เมื่อ $\lambda_i(t)$ คือตัวคูณลากรางจ์ (Lagrange Multipliers) ซึ่งเป็นฟังก์ชันของเวลา เมื่อเวลาสุดท้ายแปรผัน (Variable End Time) ดังนั้นจะพบการเปลี่ยนแปลงของฟังก์ชัน และได้เงื่อนไขจำเป็น (Necessary Conditions) ดังนี้

$$\begin{aligned}
\delta J' = & \int_{t_0}^{t_f} \sum_{j=1}^n \left(\frac{\partial L'}{\partial x_j} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{x}_j} \right) h_{x_j} dt \\
& + \int_{t_0}^{t_f} \sum_{k=1}^m \left(\frac{\partial L'}{\partial u_k} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{u}_k} \right) h_{u_k} dt \\
& + \sum_{k=1}^m \left(\frac{\partial L'}{\partial \dot{u}_k} \delta u_k \Big|_{t_f} - \frac{\partial L'}{\partial \dot{u}_k} \delta u_k \Big|_{t_0} \right) + \sum_{j=1}^n \left(\frac{\partial L'}{\partial \dot{x}_j} \delta x_j \right) \Big|_{t_f} \\
& + \sum_{j=1}^n \left(\frac{\partial \Phi}{\partial x_j} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial x_j} \right) \Big|_{t_f} \delta x_j \Big|_{t_f} \\
& + \left[\frac{\partial \Phi}{\partial t} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial t} + L' - \sum_{j=1}^n \dot{x}_j \frac{\partial L'}{\partial \dot{x}_j} - \sum_{k=1}^m \dot{u}_k \frac{\partial L'}{\partial \dot{u}_k} \right]_{t_f} \delta t_f
\end{aligned} \tag{2.64}$$

$$\dot{\lambda}_j = -\frac{\partial L}{\partial x_j} - \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial x_j}, \quad j = 1, \dots, n \tag{2.65}$$

$$\frac{\partial L}{\partial u_k} + \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial u_k} = 0, \quad k = 1, \dots, m \tag{2.66}$$

$$\lambda_j \Big|_{t_f} = \left[\frac{\partial \Phi}{\partial x_j} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial x_j} \right]_{t_f}, \quad j = 1, \dots, n \tag{2.67}$$

$$\left[\frac{\partial \Phi}{\partial t} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial t} + L + \sum_{i=1}^n \lambda_i f_i \right]_{t_f} = 0 \tag{2.68}$$

ผลเฉลยของสภาวะ (States) $x_j(t)$, สภาวะร่วม (Costates) $\lambda_j(t)$, ตัวควบคุมนำเข้า (Control Inputs) $u_k(t)$ และเวลาสุดท้าย t_f สามารถหาค่าได้โดยใช้สมการสภาวะ (State Equations) คือสมการ (2.50) จำนวน n สมการ, สมการสภาวะร่วม (Costate Equations) คือสมการ (2.65) จำนวน n สมการ และสมการเพิ่มเติม (Additional Optimality Equations) คือสมการ (2.66) จำนวน m สมการ และสมการเงื่อนไขขอบเขต (Boundary Conditions) คือสมการ (2.68) มีสมการเชิงอนุพันธ์ (Different Equations) อยู่จำนวน $2n$ สมการ และมีสมการสามัญ (Ordinary Equations) จำนวน $m+1$ สมการ การหาผลเฉลยนี้มีความจำเป็นจะต้องมีการกำหนดเงื่อนไขขอบเขต (Boundary Conditions) จำนวน $2n$ เงื่อนไข และต้องกำหนดสภาวะเริ่มต้นที่เวลา t_0 ด้วย ส่วน สภาวะร่วม (Costate) $\lambda_j(t_f)$ สามารถคำนวณได้จากสมการ (2.67)

2.3.2.2.3 เงื่อนไขบังคับสนับสนุน (Auxiliary Constraints)

สำหรับกรณีปัญหาที่ตัวแปรสถานะ (State Variables) และตัวแปรควบคุม (Control Variables) จำเป็นจะต้องสอดคล้องกับเงื่อนไขบังคับ (Constraints) ซึ่งเขียนในเทอมทางคณิตศาสตร์เป็น $g_r(t, x_1, \dots, x_n, u_1, \dots, u_m)_{t_f} = 0$ และ $r = 1, \dots, p'$ โดยอยู่ในช่วงระหว่าง t_0 และ t_f ซึ่งที่สภาวะสุดท้าย จะต้องเป็นไปตามเงื่อนไขที่เขียนในเทอมทางคณิตศาสตร์ เป็น $s_l(t, x_1, \dots, x_n)_{t_f} = 0$ และ $l = 1, \dots, p$ ทำให้เราได้ฟังก์ชันนอล (Functional) เป็น

$$J'[x_1, \dots, x_n, u_1, \dots, u_m] = \Phi(t, x_1, \dots, x_n)_{t_f} + \sum_{l=1}^p \nu_l s_l(t, x_1, \dots, x_n)_{t_f} + \int_{t_0}^{t_f} L'(t, x_1, \dots, x_n, u_1, \dots, u_m) dt \quad (2.69)$$

เมื่อ ν_l ก็คือค่าคงตัวตัวคูณลากรางจ์ (Constant Lagrange Multipliers) และ L' เขียนเป็น

$$L'(t, x_1, \dots, x_n, u_1, \dots, u_m) = L(t, x_1, \dots, x_n, u_1, \dots, u_m) + \sum_{i=1}^n \lambda_i(t) [f_i(t, x_1, \dots, x_n, u_1, \dots, u_m) - \dot{x}_i] + \sum_{r=1}^{p'} \mu_r(t) g_r(t, x_1, \dots, x_n, u_1, \dots, u_m) \quad (2.70)$$

เมื่อ $\lambda_i(t)$ และ $\mu_r(t)$ คือตัวคูณลากรางจ์ (Lagrange Multipliers) ซึ่งเป็นฟังก์ชันของเวลา เมื่อเวลาสุดท้ายมีการแปรผัน (Variable End Time) ดังนั้นจะพบการเปลี่ยนแปลงของฟังก์ชันนอล (Functional) ดังนี้

$$\begin{aligned} \delta J' = & \int_{t_0}^{t_f} \sum_{j=1}^n \left(\frac{\partial L'}{\partial x_j} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{x}_j} \right) h_{x_j} dt \\ & + \int_{t_0}^{t_f} \sum_{k=1}^m \left(\frac{\partial L'}{\partial u_k} - \frac{d}{dt} \frac{\partial L'}{\partial \dot{u}_k} \right) h_{u_k} dt \\ & + \sum_{k=1}^m \left(\frac{\partial L'}{\partial \dot{u}_k} \delta u_k \Big|_{t_f} - \frac{\partial L'}{\partial \dot{u}_k} \delta u_k \Big|_{t_0} \right) + \sum_{j=1}^n \left(\frac{\partial L'}{\partial \dot{x}_j} \delta x_j \Big|_{t_f} \right. \\ & + \sum_{j=1}^n \left(\frac{\partial \Phi}{\partial x_j} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial x_j} \right) \Big|_{t_f} \delta x_j \Big|_{t_f} \\ & + \left[\frac{\partial \Phi}{\partial t} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial t} + L' - \sum_{j=1}^n \dot{x}_j \frac{\partial L'}{\partial x_j} - \sum_{k=1}^m \dot{u}_k \frac{\partial L'}{\partial u_k} \right]_{t_f} \delta t_f \end{aligned} \quad (2.71)$$

เมื่อเราทำการประเมินค่าแต่ละส่วนในสมการ (2.71) ก็จะทำให้เราสามารถหาเงื่อนไขจำเป็น (Necessary Conditions) ได้ดังนี้

$$\dot{\lambda}_j = -\frac{\partial L}{\partial x_j} - \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial x_j} - \sum_{r=1}^{p'} \mu_r \frac{\partial g_r}{\partial x_j}, j = 1, \dots, n \quad (2.72)$$

$$\frac{\partial L}{\partial u_k} + \sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial u_k} + \sum_{r=1}^{p'} \mu_r \frac{\partial g_r}{\partial u_k} = 0, k = 1, \dots, m \quad (2.73)$$

$$\lambda_j|_{t_f} = \left[\frac{\partial \Phi}{\partial x_j} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial x_j} \right]_{t_f}, j = 1, \dots, n \quad (2.74)$$

$$\left[\frac{\partial \Phi}{\partial t} + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial t} + L + \sum_{r=1}^{p'} \mu_r g_r + \sum_{i=1}^n \lambda_i f_i \right]_{t_f} = 0 \quad (2.75)$$

ผลเฉลยของตัวแปรสถานะ (State Variables) $x_j(t)$, สภาวะร่วม (Costate) $\lambda_j(t)$, ตัวควบคุมนำเข้า (Control Inputs) $u_k(t)$, ตัวคูณลากรางจ์ (Lagrange Multipliers) $\mu_i(t)$ และเวลาสุดท้าย t_f สามารถหาค่าได้โดยการใช้สมการสถานะ (State Equations) คือสมการ (2.50) จำนวน n สมการ, สมการสภาวะร่วม (Costate Equations) คือสมการ (2.72) จำนวน n สมการ และสมการเพิ่มเติม (Additional Optimality Equations) คือสมการ (2.73) จำนวน m สมการ, สมการเงื่อนไขบังคับ (Constraint Equations) $g_r(t, x_1, \dots, x_n, u_1, \dots, u_m)_{t_f} = 0$ จำนวน p' สมการ และสมการเงื่อนไขขอบเขต (Boundary Conditions) คือสมการ (2.75) ทำให้มีสมการเชิงอนุพันธ์ (Different Equations) อยู่จำนวน $2n$ สมการ และมีสมการสามัญ (Ordinary Equations) จำนวน $m + p' + 1$ สมการ มีเงื่อนไขขอบเขต (Boundary Conditions) จำนวน $2n$ เงื่อนไข และต้องกำหนดสภาวะที่เวลา t_0 และสภาวะร่วม (Costate) $\lambda_j(t_f)$ สามารถคำนวณได้จากสมการ (2.74)

4. การเคลื่อนที่ด้วยเวลาน้อยที่สุด (Minimum Time Motion)

ลักษณะของปัญหาแบบนี้ในทางวิศวกรรมถือว่าเป็นประโยชน์อย่างมากต่อการนำไปใช้ในการหาเวลาน้อยที่สุดในขบวนการทำงานต่างๆ ส่วนมากแล้วการเคลื่อนที่แบบนี้จะทำ

ให้ค่า $L=1$ และ $\Phi=0$ ดังนั้นจากหัวข้อที่ (2.3.2.2.3) จะทำให้ได้สมการเงื่อนไขจำเป็น (Necessary Conditions) ดังนี้

$$\dot{\lambda}_j = -\sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial x_j} - \sum_{r=1}^{p'} \mu_r \frac{\partial g_r}{\partial x_j}, j=1, \dots, n \quad (2.76)$$

$$\sum_{i=1}^n \lambda_i \frac{\partial f_i}{\partial u_k} + \sum_{r=1}^{p'} \mu_r \frac{\partial g_r}{\partial u_k} = 0, k=1, \dots, m \quad (2.77)$$

$$\lambda_j|_{t_f} = \left[\sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial x_j} \right]_{t_f}, j=1, \dots, n \quad (2.78)$$

$$\left[1 + \sum_{l=1}^p \nu_l \frac{\partial s_l}{\partial t} + \sum_{r=1}^{p'} \mu_r g_r + \sum_{i=1}^n \lambda_i f_i \right]_{t_f} = 0 \quad (2.79)$$

ผลเฉลยของตัวแปรสถานะ (State Variables) $x_j(t)$, สภาวะร่วม (Costate) $\lambda_j(t)$, ตัวควบคุมนำเข้า (Control Inputs) $u_k(t)$, ตัวคูณลากรางจ์ (Lagrange multipliers) $\mu_i(t)$ และ เวลาสุดท้าย t_f สามารถหาค่าได้โดยใช้สมการสถานะ (State Equations) คือสมการ (2.50) จำนวน n สมการ, สมการสภาวะร่วม (Costate Equations) คือสมการ (2.76) จำนวน n สมการ, สมการเพิ่มเติม (Additional Optimality Equations) คือสมการ (2.77) จำนวน m สมการ, สมการเงื่อนไขบังคับ (Constraint Equations) $g_r(t, x_1, \dots, x_n, u_1, \dots, u_m)_{t_f} = 0$ จำนวน p' สมการ และ สมการเงื่อนไขขอบเขต (Boundary Conditions) คือสมการ (2.79) ทำให้มีสมการเชิงอนุพันธ์ (Different Equations) อยู่จำนวน $2n$ สมการ และมีสมการสามัญ (Ordinary Equations) จำนวน $m + p' + 1$ สมการ มีเงื่อนไขขอบเขต (Boundary Conditions) จำนวน $2n$ เงื่อนไข และ ต้องกำหนดสภาวะที่เวลา t_0 และสภาวะร่วม (Costate) $\lambda_j(t_f)$ สามารถคำนวณได้จากสมการ (2.78)

3.2.3 การตรวจสอบเอ็กทรีมัมว่าเป็นค่าน้อยที่สุดหรือมากที่สุด

(Sufficient Conditions for Minimum and Maximum)

จากหัวข้อที่ผ่านมาเป็นการหาเงื่อนไขจำเป็น (Necessary Conditions) สำหรับใช้เอ็กทรีมัม (Extremum) เท่านั้น ซึ่งยังไม่บอกได้ว่าเป็นค่าน้อยที่สุดหรือมากที่สุด ในหัวข้อนี้จะเป็นการ

ตรวจสอบเพื่อบ่งชี้ว่าเอ็กซ์ตรีมัม (Extremum) เป็นค่าใดกันแน่ ในที่นี้จะกล่าวเฉพาะการตรวจหาค่า
น้อยที่สุดเท่านั้น เพราะการตรวจหาค่ามากที่สุดก็ใช้หลักการเดียวกัน เรามาพิจารณาปัญหาการหา
ค่า $\bar{u}(t) \in R^m$ ที่ทำให้ฟังก์ชันนอล (Functional) นี้มีค่าน้อยที่สุด

$$J[\bar{x}, \bar{u}] = \Phi(t_f, \bar{x}) + \int_{t_0}^{t_f} L(t, \bar{x}, \bar{u}) dt \quad (2.80)$$

โดยสอดคล้องกับ $\dot{\bar{x}} = \bar{f}(t, \bar{x}, \bar{u})$ และ $\bar{x}(t_0) = \bar{x}_0$ เมื่อ $\bar{x}(t) \in R^n$, t_0 และ t_f ถูก
กำหนดไว้ตายตัว ต่อไปหากเรากำหนดให้ $\bar{h}_x(t) \in R^n$ และ $\bar{h}_u(t) \in R^m$ เป็นค่าที่เพิ่มเข้าไปใน
ฟังก์ชัน จะพบการเปลี่ยนแปลงของฟังก์ชันต้นทุน (Cost Function) ดังนี้

$$\Delta J = J[\bar{x} + \bar{h}_x, \bar{u} + \bar{h}_u] - J[\bar{x}, \bar{u}] \quad (2.81)$$

แล้วใช้อนุกรมเทย์เลอร์ (Taylor Series) กระจายเทอมก็จะได้

$$\Delta J = \delta J + \delta^2 J + \text{higher-order-terms} \quad (2.82)$$

เมื่อ

$$\begin{aligned} \delta J &= \bar{h}_x^T \left[\frac{\partial \Phi}{\partial \bar{x}} - \bar{\lambda} \right] \Big|_{t_f} + \int_{t_0}^{t_f} \left[\bar{h}_x^T \left(\frac{\partial H}{\partial \bar{x}} + \dot{\bar{\lambda}} \right) + \bar{h}_u^T \frac{\partial H}{\partial \bar{u}} \right] dt \\ \delta^2 J &= \frac{1}{2} \bar{h}_x^T \frac{\partial^2 \Phi}{\partial \bar{x}^2} \bar{h}_x \Big|_{t_f} + \frac{1}{2} \int_{t_0}^{t_f} \begin{bmatrix} \bar{h}_x^T & \bar{h}_u^T \end{bmatrix} \tilde{H} \begin{bmatrix} \bar{h}_x \\ \bar{h}_u \end{bmatrix} dt \\ \tilde{H} &= \begin{bmatrix} \frac{\partial^2 H}{\partial \bar{x}^2} & \frac{\partial^2 H}{\partial \bar{u} \partial \bar{x}} \\ \left(\frac{\partial^2 H}{\partial \bar{u} \partial \bar{x}} \right)^T & \frac{\partial^2 H}{\partial \bar{u}^2} \end{bmatrix} \end{aligned} \quad (2.83)$$

ซึ่งเวกเตอร์สภาวะร่วม (Costate Vector) แสดงอยู่ในรูปของสเกลาร์ฮามิลโตเนียน
ฟังก์ชัน (Scalar Hamiltonian Function) $H = L + \bar{\lambda}^T \bar{f}$ และ $\bar{\lambda}(t) \in R^n$ เรียบร้อยแล้ว

เมื่อ $\bar{x}(t)$ และ $\bar{u}(t)$ เป็นเอ็กทรีมัม (Extremum) นั่นคือ $\delta J = 0$ ทำให้สมการ (2.82) เขียนได้เป็น $\Delta J = \delta^2 J + \text{higher-order-terms}$ ซึ่งเราจะพบว่าค่าของ $\bar{x}(t)$ และ $\bar{u}(t)$ จะเป็นค่าน้อยที่สุด เมื่อ $\delta^2 J$ มีค่าเป็นบวกทั้งหมด (Positive Definite) นั่นก็คือ

$$\frac{\partial^2 \Phi}{\partial \bar{x}^2} > 0; \tilde{H} > 0 \quad (2.84)$$

3.3 แบบเงื่อนไขแบบไม่เท่ากัน (Inequality Constraints)

ในหัวข้อนี้จะกล่าวถึงปัญหาความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization) ที่มีแบบเงื่อนไขแบบไม่เท่ากัน (Inequality Constraints) ของการค้นหาค่า $\bar{u}(t) \in R^m$ ที่ทำให้ฟังก์ชันนอล (Functional) มีค่าน้อยที่สุด (Minimum) ดังสมการนี้คือ

$$J[\bar{x}, \bar{u}] = \Phi(t, \bar{x}) \Big|_{t_f} + \int_{t_0}^{t_f} L(t, \bar{x}, \bar{u}) dt \quad (2.85)$$

โดยสอดคล้องกับ

$$\dot{\bar{x}} = \bar{f}(t, \bar{x}, \bar{u}) \quad (2.86)$$

$$c_i(\bar{x}, \bar{u}, t) \leq 0, i = 1, 2, \dots, p \quad (2.87)$$

$$\bar{x}(t_0) = \bar{x}_0 \quad (2.88)$$

เมื่อ $\bar{x}(t) \in R^n$, เวลาเริ่มต้นและเวลาสุดท้าย (t_0 และ t_f) ต่างก็ถูกกำหนดไว้ตายตัว แล้ว สมการแบบเงื่อนไขแบบไม่เท่ากัน (Inequality Constraints) คือ $c_i(\bar{x}, \bar{u}, t), i = 1, \dots, p$ จะต้องประกอบไปด้วยตัวแปรเงื่อนไขแบบไม่เท่ากัน (Variable Inequality Constraints) และตัวแปรสถานะเงื่อนไขแบบไม่เท่ากัน (State Variable Inequality Constraints) และเมื่อใช้วิธีการเติมสแลกวาริเอเบิล (Slack Variables) คือ $s_i(t)$ เข้าไปในสมการเงื่อนไขบังคับ (Constraint Equation) จะได้สมการเป็น

$$c_i(\bar{x}, \bar{u}, t) + s_i^2(t) = 0, i = 1, \dots, p \quad (2.89)$$

ซึ่งสแลกวาริเอเบิล (Slack Variables) ในสมการ (2.89) จะเป็นตัวแปรควบคุม (Control Variables) ตัวใหม่ในปัญหา เมื่อทำการเพิ่มสมการ (2.86) และสมการ (2.89) เข้าไปในสมการฟังก์ชันนอล (Functional) สมการ (2.85) ด้วยการใช้ตัวคูณลากรางจ์ (Lagrange Multipliers) $\bar{\lambda} \in R^n$ และ $\bar{\mu} \in R^p$ ทำให้ได้สมการใหม่เป็น

$$J' = \Phi + \int_{t_0}^{t_f} \left(L + \sum_{i=1}^n \lambda_i (f_i - \dot{x}_i) + \sum_{i=1}^p \mu_i (c_i + s_i^2) \right) dt \quad (2.90)$$

เมื่อ $\delta J' = 0$ ทำให้เราได้เงื่อนไขจำเป็น (Necessary Conditions) ดังต่อไปนี้ คือ

$$\dot{x}_i = f_i, i = 1, 2, \dots, n \quad (2.91)$$

$$\dot{\lambda}_i = -\frac{\partial L}{\partial x_i} - \sum_{j=1}^n \frac{\partial f_j}{\partial x_i} \lambda_j - \sum_{j=1}^p \frac{\partial c_j}{\partial x_i} \mu_j, i = 1, 2, \dots, n \quad (2.92)$$

$$0 = \frac{\partial L}{\partial u_i} + \sum_{j=1}^n \frac{\partial f_j}{\partial u_i} \lambda_j + \sum_{j=1}^p \frac{\partial c_j}{\partial u_i} \mu_j, i = 1, 2, \dots, m \quad (2.93)$$

$$0 = 2\mu_i s_i, i = 1, 2, \dots, p \quad (2.94)$$

$$0 = c_i + s_i^2, i = 1, 2, \dots, p \quad (2.95)$$

$$x_i(t_0) = x_{0i}, i = 1, 2, \dots, n \quad (2.96)$$

$$\lambda_i(t_f) = \frac{\partial \Phi}{\partial x_i} \Big|_{t_f}, i = 1, 2, \dots, n \quad (2.97)$$

ซึ่งสมการ (2.91-2.97) คือลักษณะของบีวีพี-ดีเออี (BVP-DAE: Two Point Boundary Value Problem Involving Differential and Algebraic Equations) เมื่อสมการ (2.91 และ 2.92) คือสมการอนุพันธ์ (Differential Equations) ส่วนสมการ (2.93-2.95) คือสมการพีชคณิต (Algebraic Equations) และสมการ (2.96 และ 2.97) คือสมการเงื่อนไขขอบเขต (Boundary Conditions) สำหรับกรณีที่เป็นการหาค่าน้อยที่สุด (Minimum) เราสามารถหาเงื่อนไขจำเป็น (Necessary Conditions) ได้ เมื่อ $\delta J'' = 0$ นั่นก็คือ

$$J'' = \Phi + \int_{t_0}^{t_f} \left(L + \sum_{i=1}^n \lambda_i (f_i - \dot{x}_i) + \sum_{i=1}^p \mu_i c_i \right) dt \quad (2.98)$$

เมื่อ $\delta J'' = 0$ ทำให้เราได้เงื่อนไขจำเป็น (Necessary Conditions) ดังต่อไปนี้ คือ

$$\dot{x}_i = f_i, i = 1, 2, \dots, n \quad (2.99)$$

$$\dot{\lambda}_i = -\frac{\partial L}{\partial x_i} - \sum_{j=1}^n \frac{\partial f_j}{\partial x_i} \lambda_j - \sum_{j=1}^p \frac{\partial c_j}{\partial x_i} \mu_j, i = 1, 2, \dots, n \quad (2.100)$$

$$0 = \frac{\partial L}{\partial u_i} + \sum_{j=1}^n \frac{\partial f_j}{\partial u_i} \lambda_j + \sum_{j=1}^p \frac{\partial c_j}{\partial u_i} \mu_j, i = 1, 2, \dots, m \quad (2.101)$$

$$0 = \mu_i c_i, i = 1, 2, \dots, p \quad (2.102)$$

$$0 \leq \mu_i, i = 1, 2, \dots, p \quad (2.103)$$

$$0 \geq c_i, i = 1, 2, \dots, p \quad (2.104)$$

$$x_i(t_0) = x_{0i}, i = 1, 2, \dots, n \quad (2.105)$$

$$\lambda_i(t_f) = \frac{\partial \Phi}{\partial x_i} \Big|_{t_f}, i = 1, 2, \dots, n \quad (2.106)$$

4. อัลกอริทึมเชิงตัวเลข (Numerical Algorithms)

4.1 บทนำ (Introduction)

ในหัวข้อนี้ จะแสดงถึงอัลกอริทึมเชิงตัวเลข (Numerical Algorithms) วิธีหนึ่งที่ได้รับการพัฒนาขึ้นมาเพื่อใช้แก้ปัญหาการหาความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization) ซึ่งอัลกอริทึมนี้จะใช้ทั้งวิธีโดยตรง (Direct Method) และวิธีโดยอ้อม (Indirect Method) โดยจะเลือกใช้แนวทางใดทางหนึ่งในการพิจารณาปัญหาที่สามารถระบุลักษณะสมบัติได้ โดยใช้สมการดิฟเฟอเรนเชียล (Differential Equation) ที่มีลำดับใดๆ ทั้งนี้ มีเทคนิคเชิงตัวเลข (Numerical Techniques) มากมายหลายวิธีที่สามารถเลือกใช้ได้แต่อย่างไรก็ตามจุดประสงค์ในที่นี้ คือ ต้องการที่จะพัฒนาวิธีเชิงตัวเลขแบบทั่วไปขึ้นมาวิธีหนึ่ง ซึ่งสามารถนำไปประยุกต์ใช้ได้กับประเภทของปัญหาที่กล่าวไว้แล้วข้างต้น หลังจากที่ได้พิจารณาระเบียบวิธีเชิงตัวเลข (Numerical Method) หลายวิธี ตัวอย่างเช่น การเลือกค่าแบบหลายครั้ง (Multiple Shooting) หรือวิธีส่วนเหลือแบบถ่วง

น้ำหนัก (Weighted Residual) และวิธีการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) เป็นต้น สามารถสรุปได้ว่า วิธีการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) เป็นเทคนิคที่เป็นสามัญทั่วไปมากที่สุด สำหรับใช้จัดการกับปัญหาการหาผลลัพธ์ที่ดีที่สุดที่มีเงื่อนไขที่ซับซ้อนมากยิ่งขึ้น ด้วยเหตุนี้จึงมีการเลือกใช้เทคนิคดังกล่าวเป็นแนวทางเบื้องต้นในวิทยานิพนธ์ฉบับนี้ เพื่อให้ได้แนวทางการแก้ปัญหาที่ดีที่สุดและสามารถนำผลลัพธ์มาเปรียบเทียบกันได้

4.2 การโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming)

วิธีการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) ซึ่งเป็นวิธีที่จะนำมาใช้ในวิทยานิพนธ์นี้มีโครงสร้างที่จำเป็นจะต้องพิจารณาเวกเตอร์ขนาดมิติ n' -vector $y = (y_1, \dots, y_{n'})$ ซึ่งทำให้ได้ค่าปริมาณสเกลาร์ของฟังก์ชันวัตถุประสงค์ (Objective Function) ต่ำที่สุด ดังนี้

$$J(y) \quad (2.107)$$

ทั้งนี้ขึ้นอยู่กับเงื่อนไข (Constraints) ที่มีเงื่อนไขแบบเท่ากัน (Equality Constraints) และเงื่อนไขแบบไม่เท่ากัน (Inequality Constraints) ซึ่งอยู่ในรูปแบบ ดังนี้

$$b_l \leq \left\{ \begin{matrix} y \\ C(y) \end{matrix} \right\} \leq b_u \quad (2.108)$$

โดย ค่า b_l และ b_u คือ ค่าขอบจำกัดด้านล่างและด้านบนที่เหมาะสม $C(y)$ คือ เวกเตอร์ที่มีขนาด $(k \times 1)$ ของเงื่อนไขอื่นๆ ที่เป็นทั้งแบบเชิงเส้นและไม่เชิงเส้น และ $C(y)$ คือ เซตของเงื่อนไขความเท่ากัน ซึ่ง $k \leq n'$

ในส่วนนี้จะได้นำเสนอวิธีการแบบทั่วไปสำหรับการแก้ปัญหาการควบคุมเพื่อให้ได้ผลลัพธ์ที่ดีที่สุดโดยใช้วิธีการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) วิธีการแก้ปัญหามีพื้นฐานมาจากทฤษฎีค่าใกล้เคียง (Collocation Theory) และประกอบด้วยขั้นตอนต่างๆ ได้แก่ (1) การเลือกจุด (โหนด) จำนวน $N-1$ จุด ซึ่งอยู่ระหว่างค่า t_0 และ t_f แล้วแบ่งเวลาทั้งหมดออกเป็นช่วง จำนวน n ช่วง (2) เลือกรูปแบบที่เป็นไปได้ของ $x(t)$, $\lambda(t)$, $u(t)$ เป็นต้น ในแต่ละช่วงเวลาข้างต้น คือ เป็นการจัดตั้งผลเฉลยพื้นฐานแล้วจึงเพิ่มโหมดฟังก์ชันเพื่อคัดกรองผลเฉลยที่ได้ในกรณีที่เป็น (3) คัดค่าความต่อเนื่องจนถึงอนุพันธ์ลำดับที่ $(p-1)$ ของ ฟังก์ชัน $x(t)$ และ $\lambda(t)$ ที่จุดโหนดที่ $N-1$ แต่คงให้มีความไม่ต่อเนื่องได้สำหรับฟังก์ชัน $u(t)$ (4) แก้ปัญหา

เพื่อหาค่าผลเฉลยพื้นฐานโดยใช้เงื่อนไขขอบเขตที่กำหนดให้ของฟังก์ชัน $x(t)$ และ $\lambda(t)$ ที่เวลา t_0 และ t_f รวมทั้งเงื่อนไขของความต่อเนื่องที่จุดโหนดดังกล่าว เพื่อให้ได้ค่าตั้งต้นสำหรับการแก้ปัญหา (5) ทำการทดสอบหาค่าที่ได้จากสมการดิฟเฟอเรนเชียล (Differential Equation) จนได้ค่าที่เหมาะสม โดยการเลือกจุดข้างเคียงในแต่ละช่วงค่าที่กำหนด คือ จุดข้างเคียงแต่ละจุด จะให้สมการออกมาจำนวน n สมการซึ่งสอดคล้องกับสมการพลศาสตร์ (Dynamic Equation) ของระบบนั้นๆ (6) ทำปัญหาดังกล่าวให้เป็นปัญหาโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) ที่มีฟังก์ชันวัตถุประสงค์ (Objective Function) เพื่อหาค่าสัมประสิทธิ์ของโหมด (Mode Coefficients) ที่ดีที่สุด เพื่อให้ฟังก์ชัน J มีค่าน้อยที่สุด

4.3 แนวทางวิธีโดยตรง (Direct Scheme)

ในหัวข้อนี้จะได้อธิบายถึงวิธีการหาผลลัพธ์ที่ดีที่สุดแบบอาศัยเส้นทางโดยตรงซึ่งใช้การโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) วิธีนี้จะแสดงถึงตัวแปรสถานะ (State Variables) และตัวแปรควบคุม (Control Variables) ต่างๆ โดยอาศัยสมการยกกำลัง (Piecewise Polynomials) และค่าคงที่ (Piecewise Constants) แบบเชิงท่อนไม่ต่อเนื่องตามลำดับ ส่วนวิธีการหาค่าใกล้เคียงจะใช้สำหรับการแปลงปัญหาการควบคุมเพื่อให้ได้ผลลัพธ์ที่ดีที่สุดไปเป็นปัญหาโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) ข้อได้เปรียบขั้นต้นของวิธีดังกล่าว คือ การขยายปัญหาทั่วไปซึ่งรวมเงื่อนไขความไม่เท่ากัน (Inequality Constraints) ของตัวแปรควบคุม (Control Variables) และ/หรือตัวแปรสถานะ (State Variables) ต่างๆ จะสามารถจัดการได้ง่ายขึ้นเนื่องจากผลเฉลยที่ดีที่สุดสำหรับตัวแปรควบคุม $u(t)$ จะสามารถเป็นค่าไม่ต่อเนื่องได้ ความคิดเบื้องต้นของวิธีดังกล่าวนี้ได้เสนอโดย Johnson และ Hahm and Johnson นอกจากนี้ยังมีวิธีอื่นที่คล้ายคลึงกันซึ่งพิจารณาตัวแปรควบคุม (Control Variables) เป็นสมการยกกำลังแบบเชิงท่อนไม่ต่อเนื่อง (Piecewise Polynomials) ซึ่งอธิบายไว้โดย Hargraves

การแจกแจงปัญหาในวิธีโดยตรง (Direct Method) นี้ จะทำเพื่อการหาค่าผลเฉลยต่ำที่สุดของฟังก์ชันต้นทุน (Cost Function) ดังนี้

$$J[x, u] = \phi(x(t_f), x^{(1)}(t_f), \dots, x^{(p-1)}(t_f), t_f) + \int_{t_0}^{t_f} L(x(\tau), x^{(1)}(\tau), \dots, x^{(p-1)}(\tau), u(\tau), \tau) d\tau \quad (2.109)$$

โดยมีเงื่อนไข คือ

$$x^{(p)}(t) = f(x, x^{(1)}, \dots, x^{(p-1)}, u, t) \quad (2.110)$$

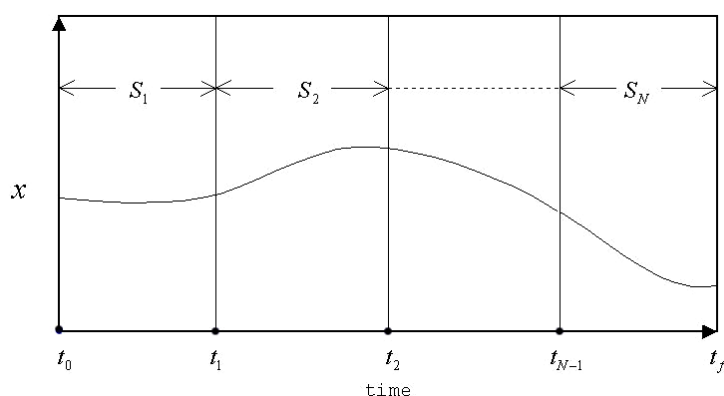
$$c_j(t, x, x^{(1)}, \dots, x^{(p-1)}, u) \leq 0, \quad j = 1, \dots, r \quad (2.111)$$

รวมทั้งกลุ่มเงื่อนไขขอบเขตที่กำหนดให้

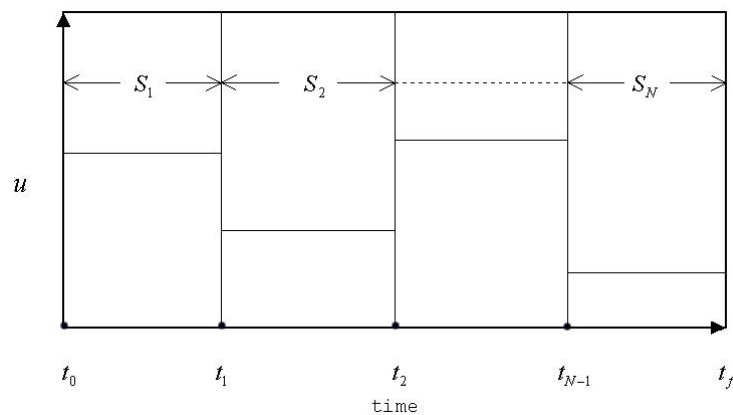
4.3.1 วิธีการแก้ปัญหา (Method of Solution)

เป้าหมายของหัวข้อนี้คือการได้อธิบายถึงวิธีโดยตรง (Direct Method) ซึ่งใช้เพื่อลดขนาดปัญหาการควบคุมเพื่อให้ได้ผลลัพธ์ที่ดีที่สุด ที่อธิบายไว้ในสมการที่ (2.109) – (2.111) ให้เป็นปัญหาการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) ทั้งนี้เพื่อให้ได้ผลสำเร็จเป็นที่น่าพอใจ จะให้ตัวแปรสถานะ (State Variables) และตัวแปรควบคุม (Control Variables) แสดงอยู่ในรูปของสมการยกกำลัง (Piecewise Polynomials) และค่าคงที่ (Piecewise Constants) แบบเชิงท่อนไม่ต่อเนื่องตามลำดับ นอกจากนี้จะทำการหาค่าจุดใกล้เคียงเพื่อให้ได้ผลลัพธ์ตามเงื่อนไขของปัญหา ตัวอย่างเช่น เงื่อนไขความเท่ากัน/ไม่เท่ากัน ซึ่งเป็นตัวกำหนดตัวแปรสถานะ (State Variables) และตัวแปรควบคุม (Control Variables) ต่างๆ ในขั้นต้น ความยาวของแต่ละช่วงค่าจะถูกกำหนดให้เป็นตามสมการ $S_i = t_i - t_{i-1}$ ดังแสดงในภาพประกอบ 1

ต่อจากนั้นให้ตัวแปรสถานะ (State Variables) แสดงอยู่ในรูปสมการยกกำลัง (Polynomials) ในแต่ละช่วงค่าเหล่านั้น นอกจากนี้จะมีการใช้ค่าคงที่ (Constants) เพื่อแสดงถึงตัวควบคุมนำเข้า (Control Inputs) ในลักษณะเดียวกันกับที่แสดงในภาพประกอบ 2



ภาพประกอบ 1 แสดงรูปแบบของยกกำลังแบบเชิงท่อนไม่ต่อเนื่อง (Piecewise Polynomial) ของสถานะ (States)



ภาพประกอบ 2 แสดงรูปแบบของค่าคงที่แบบเชิงท่อนไม่ต่อเนื่อง (Piecewise Constant) ของตัวนำเข้า (Inputs)

ค่าสัมประสิทธิ์ (Coefficients) ที่จะใช้เพื่อกำหนดองค์ประกอบของผลเฉลยสถานะ (State) และควบคุม (Control) จะคำนวณได้โดยใช้การโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) เพื่อให้ได้ตามสมการ (2.110) และ (2.111) และเงื่อนไขความต่อเนื่องที่จุดโหนด (Node Point) ต่างๆ

เพื่อให้เข้าใจได้ดีขึ้น จะกำหนดให้สถานะ $x(t)$ เป็นตัวแสดงค่าสำหรับแต่ละช่วงโดยใช้สมการยกกำลัง (Polynomial) ที่มีกำลังอย่างน้อยที่สุด $p-1$ เพื่อให้มั่นใจได้ว่าจะมีความต่อเนื่องในบริเวณที่ข้ามผ่านโหนด (Node) ต่างๆ

$$x = y_0 + y_1 t + y_2 t^2 + y_3 t^3, \dots, + y_{p-1} t^{p-1} \quad (2.112)$$

โดย $y_0, y_1, y_2, \dots, y_{p-1}$ คือ เวกเตอร์ที่มีมิติเท่ากับ n และฟังก์ชันควบคุม $u(t)$ จะแสดงอยู่ในรูป

$$u = y_u \quad (2.113)$$

โดยที่ y_u คือ เวกเตอร์ที่มีมิติเท่ากับ m ซึ่งจะสอดคล้องกับสมการดิฟเฟอเรนเชียล (Differential Equations) (2.110) และเงื่อนไขแบบไม่เท่ากัน (Inequality Constraints) (2.111) ที่จุดค่าข้างเคียงจำนวนจำกัดระหว่างค่า t_0 และ t_f ทั้งนี้วิธีการที่คล้ายคลึงกันนี้ได้ถูกนำมาใช้ในการหาค่าผลลัพธ์ที่ดีที่สุดแบบโดยตรง (Direct Optimization) ของปัญหาที่อยู่ในรูปอนุพันธ์ลำดับที่หนึ่ง (First-order Derivative Form)

ถ้ามีจำนวนช่วง N ค่า รวมทั้งองค์ประกอบของฟังก์ชัน $x(t)$ และ $u(t)$ เท่ากับ n จำนวนตัวแปรตัดสินใจ (Decision Variables) ทั้งหมด y จะมีค่า $\geq (np + m)N$ ตัวแปร x จะขึ้นอยู่กับเงื่อนไขขอบเขตที่ t_0 และ/หรือ t_f และความต่อเนื่องที่ข้ามผ่านรอยต่อของแต่ละช่วงทั้งหมดนี้จะทำให้ได้เงื่อนไขความเท่ากัน (Equality Constraints) อย่างมากที่สุดเท่ากับ $np(N+1)$ เงื่อนไข ถ้า N_c คือจำนวนของจุดข้างเคียง (Collocation Points) แล้วสมการที่ (2.110) จะให้เงื่อนไขแบบเท่ากัน (Equality Constraints) ออกมาเท่ากับ nN_c เงื่อนไข และสมการที่ (2.111) จะให้เงื่อนไขแบบไม่เท่ากัน (Inequality Constraints) ออกมาเท่ากับ rN_c เงื่อนไข ถึงจุดนี้ถือว่าได้มีการตั้งปัญหาการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) อย่างเหมาะสมแล้วและสามารถแก้ปัญหาดังกล่าวต่อไปได้โดยใช้เครื่องมือแบบมาตรฐานทั่วไปได้

4.3.2 ประเด็นต่างๆ เกี่ยวกับการคำนวณ (Computation Issues)

ในหัวข้อนี้จะเปรียบเทียบขนาดมิติ (Dimension) ของปัญหาการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) ซึ่งอยู่ในรูปอนุพันธ์ลำดับสูง (Higher-order Derivative Form) กับปัญหาอีกแบบหนึ่งที่อยู่ในรูปอนุพันธ์ลำดับที่หนึ่ง (First-order Derivative Form) สมมุติว่าได้เลือกจำนวนช่วงค่าและจำนวนจุดใกล้เคียงสำหรับปัญหาทั้งสองแบบอย่างเท่าๆ กัน ระบบที่อธิบายได้ในรูปของสมการดิฟเฟอเรนเชียล (Differential Equation) ลำดับที่ np จะสามารถแสดงได้ในรูปของสมการดิฟเฟอเรนเชียลลำดับที่หนึ่ง (First-order Differential Equation) จำนวน np สมการ ดังนั้นการแสดงองค์ประกอบของค่าสถานะในรูปของเชิงท่อนไม่ต่อเนื่อง (Piecewise States) อย่างน้อยจะต้องอยู่ในรูปของสมการยกกำลังลำดับที่หนึ่ง (First-degree polynomials) ทั้งนี้จะกำหนดให้ค่านำเข้าเป็นค่าคงที่แบบเชิงท่อนไม่ต่อเนื่อง (Piecewise Constants) ด้วยการกำหนดเช่นนี้จะทำให้ได้จำนวนของตัวแปรตัดสินใจ (Decision Variables) รวมทั้งหมดมากกว่า หรือเท่ากับ $(np + m)N$ จำนวน คล้ายคลึงกับกรณีของลำดับอนุพันธ์ที่สูงกว่า และจำนวนเงื่อนไขขอบเขตรวมกับความต่อเนื่องข้ามผ่านจุดไหนดจะได้จำนวนเงื่อนไขแบบเท่ากัน (Equality

Constraints) มากที่สุดเท่ากับ $np(N+1)$ เงื่อนไข ถ้า n_c คือจำนวนของจุดข้างเคียง (Collocation Points) แล้วสมการพลศาสตร์ (Dynamic Equations) ในรูปของอนุพันธ์ลำดับหนึ่ง (First-order Derivative Form) จะให้เงื่อนไขแบบเท่ากัน (Equality Constraints) ออกมาเท่ากับ npN_c เงื่อนไข และจะให้เงื่อนไขแบบไม่เท่ากัน (Inequality Constraints) ออกมาเท่ากับ rN_c เงื่อนไข

จากการเปรียบเทียบดังกล่าวนี้ จะสามารถตั้งข้อสังเกตได้ว่า ณ จุดข้างเคียง (Collocation Grid) จุดเดียวกันในเวลาหนึ่ง สมการพลศาสตร์ (Dynamic Equations) ในรูปของอนุพันธ์ลำดับสูง (Higher-order Derivative Form) จะให้เงื่อนไขแบบเท่ากัน (Equality Constraints) ออกมา มีจำนวนน้อยกว่า เมื่อเทียบกับสมการที่อยู่ในรูปของอนุพันธ์ลำดับหนึ่ง (First-order Derivative Form) เป็นสัดส่วนเท่ากับ P เท่า

4.4 แนวทางวิธีโดยอ้อม (Indirect Scheme)

การใช้วิธีโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) เพื่อแก้ปัญหาการควบคุมเพื่อให้ได้ผลลัพธ์ที่ดีที่สุด ที่ได้กล่าวมาแล้วในหัวข้อที่ผ่านมาที่ว่า วิธีที่เหมาะสม (Optimal Trajectory) สำหรับอนุพันธ์ลำดับที่ P^{th} -order (P^{th} -order Derivative) ด้วยเงื่อนไขบังคับทั่วไป (General Constraints) หาได้โดยเงื่อนไขที่เหมาะสมดังนี้

$$x^{(p)}(t) = f(x(t), x^{(1)}(t), \dots, x^{(p-1)}(t), u, t) \quad (2.114)$$

$$\frac{\partial H'}{\partial x} - \left(\frac{\partial H'}{\partial x^{(1)}} \right)^{(1)} + \dots + (-1)^{p-1} \left(\frac{\partial H'}{\partial x^{(p-1)}} \right)^{(p-1)} = (-1)^{(p-1)} \lambda^{(p)} \quad (2.115)$$

$$\begin{aligned} & H'(x(t), x^{(1)}(t), \dots, x^{(p-1)}(t), u^*(t), \lambda(t), t) \\ & = \min H'(x(t), x^{(1)}(t), \dots, x^{(p-1)}(t), u^*(t), \lambda(t), t) \end{aligned} \quad (2.116)$$

และ

$$c_j(t, x(t), x^{(1)}(t), \dots, x^{(p-1)}(t), u) \leq 0, \quad j = 1, \dots, r \quad (2.117)$$

จากการปรับปรุง Hamiltonian ซึ่งกำหนดโดย $H' = H + \sum_{j=1}^r \mu_j c_j(t, x, u^*)$ แล้ว เงื่อนไขที่เพียงพอ (Sufficient Condition) ที่เงื่อนไขที่ j มีผลแล้ว $\mu_j \geq 0$ เงื่อนไขแบบเท่ากัน (Equality Constraints) g_i ให้พิจารณาจากสมการ (2.117) รูปแบบการสลับโครงสร้างหามาจากสมการ (2.116) หรือหาจากทฤษฎีตัวแปร (Variational Theory) ดังนี้

$$\frac{\partial H'}{\partial u_j} = 0, \quad j = 1, \dots, m \quad (2.118)$$

ในส่วนงานวิจัยนี้ เลือกสมการ (2.118) มาใช้สำหรับกระบวนการสร้างการคำนวณแทนสมการ (2.116)

4.4.1 วิธีการแก้ปัญหา (Method of Solution)

ในหัวข้อนี้จะอธิบายถึงวิธีโดยอ้อม (Indirect Method) ซึ่งใช้เพื่อลดขนาดปัญหาการควบคุมเพื่อให้ได้ค่าผลลัพธ์ที่ดีที่สุด ที่อธิบายไว้ในสมการที่ (2.114) – (2.118) ให้เป็นปัญหาการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) เงื่อนไขเหล่านี้ มีสมการสถานะ (State Equations) จำนวน n สมการ (2.114) สมการสถานะรวม จำนวน n สมการ (2.115) เงื่อนไขแบบไม่เท่ากัน (Inequality Constraints) จำนวน r เงื่อนไข (2.117) และสมการที่เหมาะสม (Optimality Equations) จำนวน m สมการ (2.118) รวมตัวแปรที่ไม่รู้ค่าทั้งหมด $(2n + m + r)$ ดังนี้คือ n สถานะ (States) $x(t)$, n สถานะรวม (Costates) $\lambda(t)$, m ตัวควบคุม (Controls) $u(t)$, r ตัวแปรลากรางจ์ (Lagrange Variables) $\mu(t)$ เทคนิคค่าข้างเคียง (Collocation Technique) ถูกนำมาใช้ในการแปลงเงื่อนไขให้เป็นโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) ซึ่งจะคล้ายคลึงกันกับวิธีโดยตรง (Direct Method) วิธีการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) จะลดค่าสเกลของฟังก์ชันให้ต่ำที่สุด และใช้ฟังก์ชันต้นทุน (Cost Function) จากสมการ (2.109) เป็นฟังก์ชันวัตถุประสงค์ (Objective Function) และความต่อเนื่องของตัวแปรสถานะ (State Variables) ที่เข้าใกล้จุดและเพียงพอในการทำให้ตัวแปรลากรางจ์มีความต่อเนื่อง

การคงสถานะที่มีความสัมพันธ์กับความต่อเนื่องของวิธีที่เหมาะสม (Optimal Trajectory) ความต่อเนื่องไม่ใช่เพียงตัวแปรสถานะ (State Variables) เท่านั้น อนุพันธ์ลำดับที่

$(p-1)$ ยังเป็นตัวช่วยในการคำนวณความแปรปรวนซึ่งเรียกว่า เงื่อนไขขอบเขตธรรมชาติ (Natural Boundary Conditions) เมื่อขอบเขตของเวลาถูกแบ่งออกเป็น n ช่วงด้วยจุดระหว่างกลางจำนวน $(n-1)$ จุด โดยมีเงื่อนไขที่จำเป็นในการหาค่าระหว่างจุดด้วยการใช้การวิเคราะห์ตัวแปร δJ_i ดังนี้

$$\begin{aligned} \delta J_i = & \int_{t_{i-1}^+}^{t_i^-} h_i^T \left[\frac{\partial H'_i}{\partial x_i} - \frac{d}{dt} \frac{\partial H'_i}{\partial x_i^{(1)}} + \dots + (-1)^p \frac{d^p}{dt^p} \frac{\partial H'_i}{\partial x_i^{(p)}} \right] dt \\ & + h_i^T \left[\frac{\partial H'_i}{\partial x_i^{(1)}} - \frac{d}{dt} \frac{\partial H'_i}{\partial x_i^{(2)}} + \dots + (-1)^{p-1} \frac{d^{p-1}}{dt^{p-1}} \frac{\partial H'_i}{\partial x_i^{(p)}} \right]_{t_{i-1}^+}^{t_i^-} \\ & + h_i^{(1)T} \left[\frac{\partial H'_i}{\partial x_i^{(2)}} - \frac{d}{dt} \frac{\partial H'_i}{\partial x_i^{(3)}} + \dots + (-1)^{p-2} \frac{d^{p-2}}{dt^{p-2}} \frac{\partial H'_i}{\partial x_i^{(p)}} \right]_{t_{i-1}^+}^{t_i^-} \\ & + \dots + h_i^{(p)T} \left[\frac{\partial H'_i}{\partial x_i^{(p)}} \right]_{t_{i-1}^+}^{t_i^-} \end{aligned} \quad (2.119)$$

เมื่อ $h_i = \delta x_i$ และ $J = \sum_{i=1}^I J_i$ จำนวนทุกๆ ช่วงถูกตั้งค่าให้เป็น 0 ทั้งนี้ จากสมการ (2.115) จะได้ผลลัพธ์ความต่อเนื่องของสถานะ (Continuity of States Results) ดังนี้

$$\begin{aligned} x(t_i^-) &= x(t_i^+), x^{(1)}(t_i^-) = x^{(1)}(t_i^+), \dots, \\ x^{(p-1)}(t_i^-) &= x^{(p-1)}(t_i^+), \quad i = 1, \dots, N-1 \end{aligned} \quad (2.120)$$

จากผลลัพธ์ที่ขอบเขตของช่วง ตัวแปรหาได้จาก

$$\begin{aligned} h(t_i^-) &= h(t_i^+), h^{(1)}(t_i^-) = h^{(1)}(t_i^+), \dots, \\ h^{(p-1)}(t_i^-) &= h^{(p-1)}(t_i^+), \quad i = 1, \dots, N-1 \end{aligned} \quad (2.121)$$

นำไปแทนค่าในสมการ (2.119) ขอบเขตในส่วนของ δJ สามารถรวมในส่วนของ $h(t_i^-), h^{(1)}(t_i^-), \dots, h^{(p-1)}(t_i^-), i = 1, \dots, N$ ที่ขอบเขตของช่วงนี้ ตัวแปร h และอนุพันธ์ของมัน

เพิ่มเป็น $p-1$ และควรระวังว่าค่าสัมประสิทธิ์ตัวคูณในส่วนนี้ต้องเป็น 0 ผลในเงื่อนไขจำนวน np เงื่อนไขของแต่ละขอบเขตช่วงที่เพิ่มเติมในเงื่อนไขจำนวน np เงื่อนไขของสมการ (2.120) จากผลลัพธ์แต่ละช่วงขอบเขตภายใน t_0 ถึง t_f หาค่าได้จำนวน $2np$ เงื่อนไขและนำมาจัดรูปใหม่ได้ดังนี้

$$\begin{aligned}
& \left[\frac{\partial H'}{\partial x^{(1)}} - \left(\frac{\partial H'}{\partial x^{(2)}} \right)^{(1)} + \dots + (-1)^{p-2} \left(\frac{\partial H'}{\partial x^{(p-1)}} \right)^{(p-2)} + (-1)^p \lambda^{(p-1)} \right]_{t_i^-} = \\
& \left[\frac{\partial H'}{\partial x^{(1)}} - \left(\frac{\partial H'}{\partial x^{(2)}} \right)^{(1)} + \dots + (-1)^{p-2} \left(\frac{\partial H'}{\partial x^{(p-1)}} \right)^{(p-2)} + (-1)^p \lambda^{(p-1)} \right]_{t_i^+} \\
& \left[\frac{\partial H'}{\partial x^{(2)}} - \left(\frac{\partial H'}{\partial x^{(3)}} \right)^{(1)} + \dots + (-1)^{p-3} \left(\frac{\partial H'}{\partial x^{(p-1)}} \right)^{(p-3)} + (-1)^{p-1} \lambda^{(p-2)} \right]_{t_i^-} = \\
& \left[\frac{\partial H'}{\partial x^{(2)}} - \left(\frac{\partial H'}{\partial x^{(3)}} \right)^{(1)} + \dots + (-1)^{p-3} \left(\frac{\partial H'}{\partial x^{(p-1)}} \right)^{(p-3)} + (-1)^{p-1} \lambda^{(p-2)} \right]_{t_i^+} \\
& \vdots \\
& \left[\frac{\partial H'}{\partial x^{(p-1)}} - \lambda^{(1)} \right]_{t_i^-} = \left[\frac{\partial H'}{\partial x^{(p-1)}} - \lambda^{(1)} \right]_{t_i^+} \\
& \left[\frac{\partial H'}{\partial x^{(p)}} \right]_{t_i^-} = \left[\frac{\partial H'}{\partial x^{(p)}} \right]_{t_i^+}
\end{aligned} \tag{2.122}$$

$$\begin{aligned}
x(t_i^-) &= x(t_i^+), x^{(1)}(t_i^-) = x^{(1)}(t_i^+), \dots, \\
x^{(p-1)}(t_i^-) &= x^{(p-1)}(t_i^+), \quad i = 1, \dots, N-1
\end{aligned}$$

หากในกรณีที่ $p=1$ เงื่อนไขนี้จะใช้ความต่อเนื่องของตัวคูณลากรางจ์ (Lagrange Multiplier) โดยที่มีเงื่อนไขขอบเขตจำนวน np เงื่อนไข ในระหว่าง t_0 ถึง t_f

สมการ (2.110), (2.111), (2.115), (2.116) และ (2.118) สามารถนำไปหาค่าใกล้เคียง (Collocation Grid) จุดสิ้นสุดในระหว่าง t_0 ถึง t_f ได้ วิธีการที่คล้ายคลึงกันนี้จะถูกนำไปใช้ในการแก้ปัญหาความเหมาะสมสูงสุดโดยวิธีโดยอ้อม (Indirect Optimization) ในรูปของอนุพันธ์อันดับที่ 1 (First-order Derivative Form)

ถ้ามีจำนวนช่วง N ค่า รวมทั้งองค์ประกอบของฟังก์ชัน x , λ , u และ μ เท่ากับ จำนวนตัวแปรตัดสินใจ (Decision Variables) ทั้งหมด y จะมีค่า $\geq (3np + m)N$ ตัวแปร x จะ

ขึ้นอยู่กับเงื่อนไขขอบเขตที่ t_0 และ/หรือ t_f และความต่อเนื่องที่ข้ามผ่านรอยต่อของแต่ละช่วง ทั้งหมดนี้จะทำให้ได้เงื่อนไขความเท่ากัน (Equality Constraints) อย่างมากที่สุดเท่ากับ $2np(N+1)$ เงื่อนไข ถ้า N_c คือจำนวนของจุดข้างเคียง (Collocation Points) แล้วสมการที่ (2.110) และ (2.115) จะให้เงื่อนไขแบบเท่ากัน (Equality Constraints) ออกมาเท่ากับ nN_c เงื่อนไข ส่วนสมการที่เหมาะสมเพิ่มเติม (Control Optimality Equations) สมการที่ (2.118) จะให้เงื่อนไขแบบเท่ากัน (Equality Constraints) ออกมาเท่ากับ mN_c เงื่อนไข และสมการที่ (2.111) จะให้เงื่อนไขแบบไม่เท่ากัน (Inequality Constraints) ออกมาเท่ากับ rN_c เงื่อนไข ถึงจุดนี้ถือว่าได้มีการตั้งปัญหาการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) อย่างเหมาะสมแล้ว และสามารถแก้ปัญหาดังกล่าวต่อไปได้โดยใช้เครื่องมือแบบมาตรฐานทั่วไปได้

ในขณะที่สมการสถานะ (State Equations) และสมการสถานะร่วม (Costate Equations) ก็เป็นอีกทางเลือกหนึ่งในรูปของอนุพันธ์อันดับหนึ่ง (First-order Derivative Form) ซึ่งสามารถหาคำตอบได้จากแต่ละจุดที่ใกล้เคียง (Collocation Points) โดยที่จำนวนของเงื่อนไขแบบไม่เท่ากัน (Inequality Constraints) ของอนุพันธ์อันดับหนึ่ง (First-order Derivative Form) และอนุพันธ์อันดับสูงกว่า (Higher-order Derivative Form) ก็เป็นลักษณะเดียวกัน ในส่วนที่แตกต่างกันของทั้งสองทางเลือกก็คือสมการพลศาสตร์ (Dynamic Equations) และสมการสถานะร่วม (Costate Equations) ลักษณะคล้ายคลึงกับกรณีวิธีตรง (Direct Case) ซึ่งมีข้อสังเกตดังนี้คือ ณ จุดข้างเคียงจุดเดียวกันในเวลาหนึ่ง สมการพลศาสตร์ (Dynamic Equations) และสมการสถานะร่วม (Costate Equations) ในรูปของอนุพันธ์อันดับสูงกว่า (Higher-order Derivative Form) จะให้เงื่อนไขแบบเท่ากัน (Equality Constraints) เมื่อเทียบกับรูปของอนุพันธ์อันดับหนึ่ง (First-order Derivative Form) จะเป็นสัดส่วนน้อยกว่า P เท่า

4.4.2 ข้อเปรียบเทียบในการคำนวณ (Computational Comparisons)

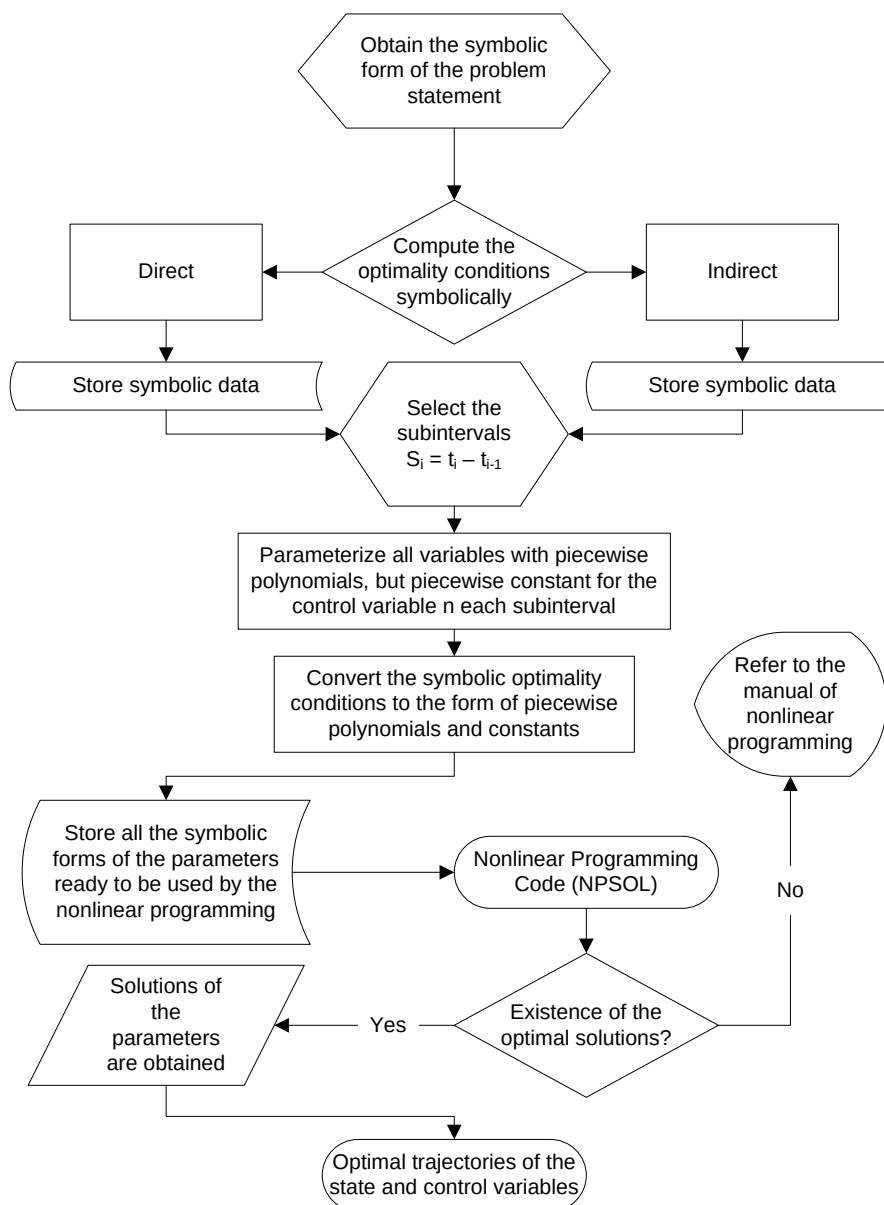
ขั้นตอนของวิธีการค่าใกล้เคียง (Collocation Procedure) ได้ถูกนำมาใช้ในระบบสมการ (2.114) สมการตัวคูณลากรางจ์ (Lagrange Multiplier) (2.115) สมการที่เหมาะสม (Optimality Equations) (2.118) และสมการเงื่อนไขแบบไม่เท่ากัน (Inequality Constraint) จะให้สมการไม่เชิงเส้น (Nonlinear Equations) จำนวน $2npN + mN$ และเงื่อนไขแบบไม่เท่ากัน (Inequality Constraint) จำนวน rN ในส่วนทางเลือกอื่น สมการ (2.114) และสมการ (2.115) ก็สามารถที่จะแปลงเป็นรูปอนุพันธ์อันดับหนึ่ง (First-order Derivative Form) จำนวน $2np$ สมการ และถ้า

$P=1$ การคำนวณด้วยรูปอนุพันธ์อันดับหนึ่ง (First-order Derivative Form) และอนุพันธ์อันดับสูงกว่า (Higher-order Derivative Form) จะเหมือนกัน แต่ถ้า P มากขึ้น อนุพันธ์อันดับสูงกว่า (Higher-order Derivative Form) จะมีสมการที่พิจารณาบ่อยลงรวมทั้งเงื่อนไขแบบเท่ากัน (Equality Constraints) ลดลงด้วย ดังนั้นในที่นี้จึงเลือกใช้วิธีอนุพันธ์อันดับสูงกว่า (Higher-order Derivative Form) เพื่อประสิทธิภาพในการคำนวณที่ดีกว่า

ข้อแตกต่างระหว่างแนวทางโดยตรง (Direct Schemes) และแนวทางโดยอ้อม (Indirect Schemes) มีอยู่ว่าเทคนิคแนวทางโดยอ้อม (Indirect Schemes) จะมีหลายเงื่อนไขที่เหมาะสมกว่าขั้นตอนของแนวทางโดยตรง (Direct Schemes) ภายใต้ตัวแปรลากรางจ์ (Lagrange Variables) และตัวแปรที่คิดขึ้นมานี้นำไปทำเป็นอัลกอริทึมทางโดยอ้อม (Indirect Schemes) จะมียุ่งยากมากกว่า เนื่องมาจากการคาดเดาค่าตัวแปรลากรางจ์ (Lagrange Variables) เริ่มต้นที่ต้องหาค่าที่เหมาะสมที่สุดในการแก้ปัญหา อย่างไรก็ตาม ข้อได้เปรียบของแนวทางโดยอ้อม (Indirect Schemes) จะมีความแม่นยำในการหาผลลัพธ์มากกว่า ซึ่งผลของความแม่นยำของเงื่อนไขที่เหมาะสมที่ของอนุพันธ์อันดับหนึ่งไม่ได้อยู่รวมอยู่ในวิธีแก้ปัญหานี้

4.4.3 การนำเสนอโปรแกรม (A General-purpose Program)

ในหัวข้อนี้จะอธิบายถึงขั้นตอนการทำงานโดยทั่วไปของโปรแกรม ซึ่งได้แสดงเป็นขั้นตอนตามรูปประกอบ 3 โดยการกำหนดปัญหาและจำนวนช่วงเป็น N ช่วง



ภาพประกอบ 3 แสดงรูปแบบของขั้นตอน (Flowchart) ของการพัฒนาโปรแกรม

ขั้นตอนตามภาพประกอบ 3 มีหัวข้อดังต่อไปนี้

1. การเลือกวิธีโดยตรง (Direct Method) หรือวิธีโดยอ้อม (Indirect Method)
2. การคำนวณเงื่อนไขที่เหมาะสมที่สุด
 - 2.1 วิธีโดยตรง จะแทนค่าต่างๆ ในสมการ (2.109), (2.110), (2.111) และ (2.120)
 - 2.2 วิธีโดยอ้อม จะแทนค่าต่างๆ ในสมการ (2.114), (2.115), (2.117) และ (2.118)

3. ลดเงื่อนไขจากขั้นตอนที่ 2.4.4.3.2 โดยใช้เทคนิคค่าใกล้เคียง (Collocation Technique)

4. หาตัวประกอบของตัวแปรทั้งหมดเช่น $x(t)$ และ $\lambda(t)$ ซึ่งแสดงด้วยสมการยกกำลัง และค่าคงที่ u

5. แปลงเงื่อนไขสัญลักษณ์ให้อยู่ในรูปแบบของ $C(y) \leq 0$ โดยที่ y เป็นเวกเตอร์ของทุกตัวสัมประสิทธิ์ที่ไม่รู้ค่า

6. สร้างการเชื่อมต่อกับตัวโปรแกรมแบบไม่เชิงเส้น

7. หาค่าผลลัพธ์ที่เหมาะสมที่สุด

บทที่ 3

วิธีการดำเนินงานวิจัย

ในการวิจัยครั้งนี้ ผู้วิจัยได้ดำเนินการตามขั้นตอนดังนี้

1. การศึกษาโปรแกรม (Study of The Program)
2. การออกแบบโปรแกรมคอมพิวเตอร์ (Computer Programming)

1. การศึกษาโปรแกรม (Study of The Program)

จากการศึกษาโปรแกรมโดยอ้างอิงจากบทที่ 2 สำหรับปัญหาความเหมาะสมสูงสุดพลศาสตร์ (Dynamic Optimization Problem) โดยจากขั้นตอนของโปรแกรม (Flowchart) เดิม (อ้างอิงจากภาพประกอบ 3) นั้นจะไม่มีกำหนดค่าเริ่มต้น (Initial Guess) ที่เหมาะสมหรือกำหนดค่าเริ่มต้นให้เป็นศูนย์ใดๆ ในส่วนงานวิจัยนี้จะเพิ่มเติมขั้นตอนของการหาค่าเริ่มต้นก่อนนำไปใช้ในกระบวนการอื่นๆ ของโปรแกรม ดังแสดงในภาพประกอบ 4 ทั้งนี้ แนวทางวิธีแก้ปัญหจะเป็นดังนี้

1.1 แนวทางวิธีโดยตรง (Direct Scheme)

วิธีการหาค่าผลลัพธ์ที่ดีที่สุดแบบอาศัยเส้นทางโดยตรงซึ่งใช้การโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) การแจกแจงปัญหาในวิธีโดยตรง (Direct Method) นี้ จะทำการหาค่าผลเฉลยต่ำที่สุดของฟังก์ชันต้นทุน (Cost Function) ดังสมการ (2.109-2.111) รวมทั้งกลุ่มเงื่อนไขขอบเขตที่กำหนดให้

1.1.1 วิธีการแก้ปัญหา (Method of Solution)

เป้าหมายของหัวข้อนี้คือการได้อธิบายถึงวิธีโดยตรง (Direct Method) ซึ่งใช้เพื่อลดขนาดปัญหาการควบคุมเพื่อให้ได้ค่าผลลัพธ์ที่ดีที่สุด ที่อธิบายไว้ในสมการที่ (2.109) – (2.111) ให้เป็นปัญหาการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming)

พิจารณาสมการ (2.112) ซึ่งเป็นสมการการเคลื่อนที่ดังต่อไปนี้

$$x = y_0 + y_1 t + y_2 t^2 + y_3 t^3, \dots, + y_{p-1} t^{p-1} \quad (3.1)$$

โดย $y_0, y_1, y_2, \dots, y_{p-1}$ คือ จริงๆ แล้วก็คือค่าเริ่มต้น ซึ่งเดิมจะกำหนดให้เป็น 0 ทั้งหมด ดังที่กล่าวไปแล้วข้างต้น แต่ในงานวิจัยนี้ จะทำการกำหนดค่าเริ่มต้นที่ใกล้เคียงก่อนนำไป

คำนวณ โดยการหาอนุพันธ์อันดับที่ 1, 2, ... ถึงอนุพันธ์สูงสุดที่มีในสมการการเคลื่อนที่ของระบบพลศาสตร์ โดยมีเงื่อนไขขอบเขตตามที่ระบุไว้แล้ว และให้สมการการเคลื่อนที่ของระบบเท่ากับ 0 และคำนวณไปเรื่อยๆ ตามจำนวนช่วงที่แบ่งไว้ โดยมีเงื่อนไขว่า อนุพันธ์อันดับต่างๆ ของช่วงที่ 1 จะต้องเท่ากับอนุพันธ์อันดับนั้นๆ ของช่วงที่ 2 และเช่นกัน อนุพันธ์อันดับต่างๆ ของช่วงที่ 2 จะต้องเท่ากับอนุพันธ์อันดับนั้นๆ ของช่วงถัดไปเรื่อยๆ จนถึงช่วงสุดท้าย หรือช่วงที่ N

จากเงื่อนไขและวิธีการที่ได้กล่าวไว้ข้างต้น สุดท้ายแล้วจะได้ชุดข้อมูลของค่า $y_0, y_1, y_2, \dots, y_{p-1}$ ทั้งหมด ซึ่งก็เป็นชุดข้อมูลดังกล่าวจะนำไปใช้เป็นค่าเริ่มต้น (Initial Guess) ที่ไม่ใช่ 0 (Zero Guess) นั่นเอง

1.2 แนวทางวิธีโดยอ้อม (Indirect Scheme)

การใช้วิธีโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) เพื่อแก้ปัญหาการควบคุมเพื่อให้ได้ผลลัพธ์ที่ดีที่สุด ที่ได้กล่าวมาแล้วในหัวข้อที่ผ่านมาที่ว่า วิธีที่เหมาะสม (Optimal Trajectory) สำหรับอนุพันธ์ลำดับที่ p^{th} -order (p^{th} -order Derivative) ด้วยเงื่อนไขบังคับทั่วไป (General Constraints) หาได้โดยเงื่อนไขที่เหมาะสมดังสมการ (2.114-2.117) จากการปรับปรุง Hamiltonian ซึ่งกำหนดโดย $H' = H + \sum_{j=1}^r \mu_j c_j(t, x, u^*)$ แล้ว เงื่อนไขที่เพียงพอ (Sufficient Condition) ที่เงื่อนไขที่ j มีผลแล้ว $\mu_j \geq 0$ เงื่อนไขแบบเท่ากัน (Equality Constraints) g_i ให้พิจารณาจากสมการ (2.117) รูปแบบการสลับโครงสร้างหามาจากสมการ (2.116) หรือหาจากทฤษฎีตัวแปร (Variational Theory) และ เลือกสมการ (2.118) มาใช้สำหรับกระบวนการสร้างการคำนวณแทนสมการ (2.116)

1.2.1 วิธีการแก้ปัญหา (Method of Solution)

วิธีโดยอ้อม (Indirect Method) จะใช้เพื่อลดขนาดปัญหาการควบคุมเพื่อให้ได้ค่าผลลัพธ์ที่ดีที่สุด ที่อธิบายไว้ในสมการที่ (2.114) – (2.118) ให้เป็นปัญหาการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) เงื่อนไขเหล่านี้ มีสมการสถานะ (State Equations) จำนวน n สมการ (2.114) สมการสถานะร่วม จำนวน n สมการ (2.115) เงื่อนไขแบบไม่เท่ากัน (Inequality Constraints) จำนวน r เงื่อนไข (2.117) และสมการที่เหมาะสม (Optimality Equations) จำนวน m สมการ (2.118) รวมตัวแปรที่ไม่รู้ค่าทั้งหมด $(2n + m + r)$ ดังนั้นคือ n สถานะ (States) $x(t)$, n สถานะร่วม (Costates) $\lambda(t)$, m ตัวควบคุม (Controls) $u(t)$, r ตัวแปรลากรางจ์ (Lagrange Variables) $\mu(t)$ เทคนิคค่าข้างเคียง (Collocation Technique) ถูกนำมาใช้ในการ

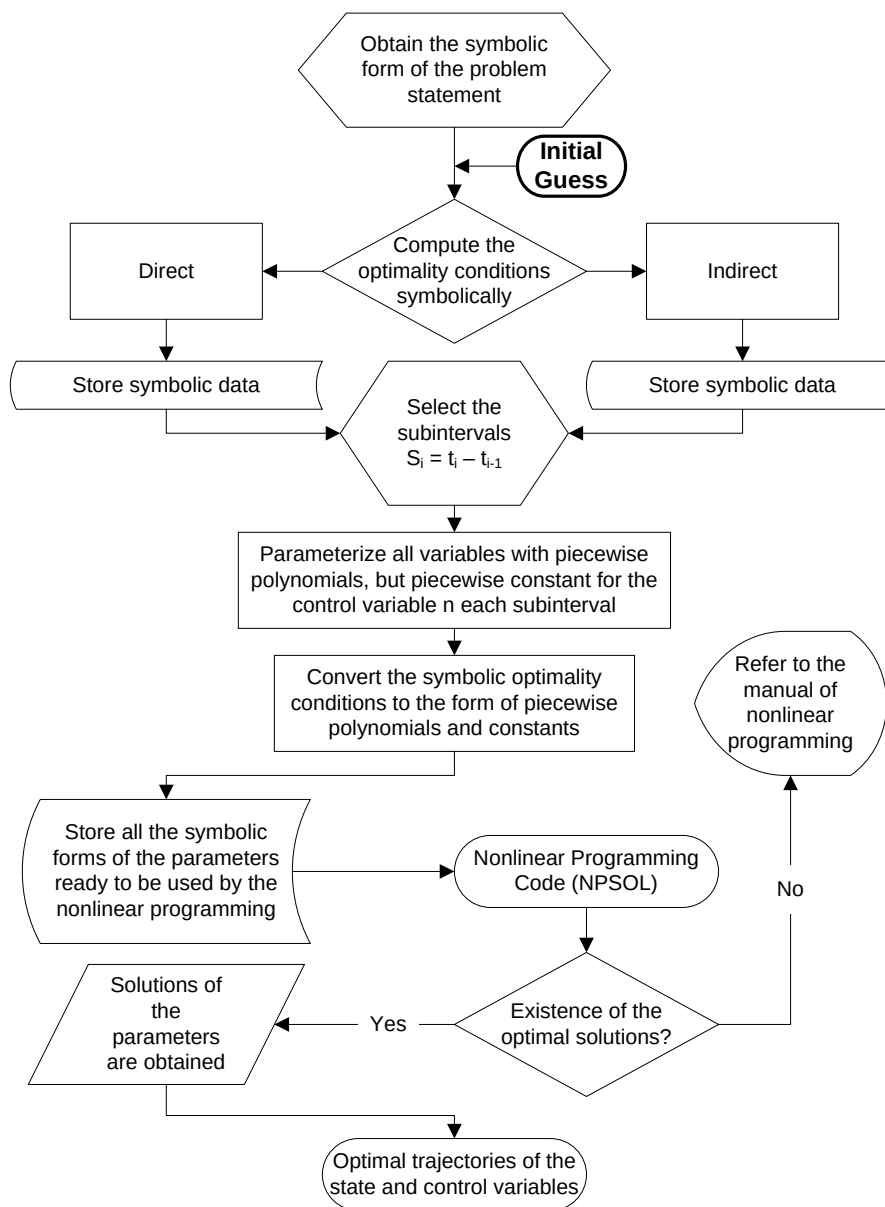
แปลงเงื่อนไขให้เป็นโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) ซึ่งจะคล้ายคลึงกันกับวิธีโดยตรง (Direct Method) วิธีการโปรแกรมแบบไม่เชิงเส้น (Nonlinear Programming) จะลดค่าสเกลของฟังก์ชันให้ต่ำที่สุด และใช้ฟังก์ชันต้นทุน (Cost Function) จากสมการ (2.109) เป็นฟังก์ชันวัตถุประสงค์ (Objective Function) และความต่อเนื่องของตัวแปรสถานะ (State Variables) ที่เข้าใกล้จุดและเพียงพอในการทำให้ตัวแปรลากรางจ์มีความต่อเนื่อง

การคงสถานะที่มีความสัมพันธ์กับความต่อเนื่องของวิธีที่เหมาะสม (Optimal Trajectory) ความต่อเนื่องไม่ใช่เพียงตัวแปรสถานะ (State Variables) เท่านั้น อนุพันธ์ลำดับที่ $(p-1)$ ยังเป็นตัวช่วยในการคำนวณความแปรปรวนซึ่งเรียกว่า เงื่อนไขขอบเขตธรรมชาติ (Natural Boundary Conditions) เมื่อขอบเขตของเวลาถูกแบ่งออกเป็น n ช่วงด้วยจุดระหว่างกลางจำนวน $(n-1)$ จุด โดยมีเงื่อนไขที่จำเป็นในการหาค่าระหว่างจุดด้วยการใช้การวิเคราะห์ตัวแปร δJ_i ดังสมการ (2.119) เมื่อ $h_i = \delta x_i$ และ $J = \sum_{i=1}^I J_i$ จำนวนทุกๆ ช่วงถูกตั้งค่าให้เป็น 0 ทั้งนี้จากสมการ (2.115) จะได้ผลลัพธ์ความต่อเนื่องของสถานะ (Continuity of States Results) ตามสมการ (2.120)

2. การออกแบบโปรแกรมคอมพิวเตอร์ (Computer Programming)

ในการออกแบบโปรแกรมจะเป็นการเพิ่มเติมโปรแกรมที่ใช้ในการหาค่าเริ่มต้น (Initial Guess) ก่อนและหลังจากนั้น จะนำตัวอย่างปัญหาทางด้านพลศาสตร์มาทดลองประมาณ 2 – 3 ตัวอย่าง โดยใช้โปรแกรมที่ปรับปรุงใหม่ และโปรแกรมเดิม แล้วนำค่า CPU Runtime มาเปรียบเทียบกัน และนำมาวิเคราะห์เพื่อทำการสรุปผลงานวิจัย

2.1 ขั้นตอนในการแก้ปัญหาโปรแกรม (Flowchart)



ภาพประกอบ 4 แสดงรูปแบบของขั้นตอน (Flowchart) ของการพัฒนาโปรแกรม โดยมีการเพิ่มขั้นตอนของการหาค่าเริ่มต้น (Initial Guess) เข้ามาในโปรแกรมเดิม

บทที่ 4

ผลการทดสอบและวิเคราะห์ผลการทดสอบ

ในการวิจัยครั้งนี้ ผู้วิจัยนำเสนอผลงานดังนี้

1. ผลการทดสอบ
2. วิเคราะห์ผลการทดสอบ

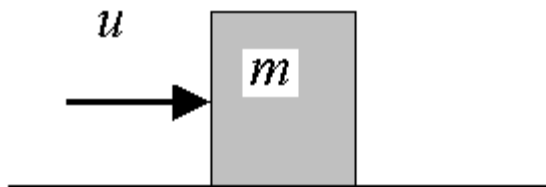
1. ผลการทดสอบ

จากการทดสอบรันโปรแกรมที่ได้มาจากการดำเนินการปรับปรุง และออกแบบโปรแกรมคอมพิวเตอร์ (Computer Programming) ตามวิธีการดำเนินงานวิจัยในบทที่ 3 โดยใช้ตัวอย่างปัญหาดังต่อไปนี้

1.1 ตัวอย่างปัญหาการเคลื่อนที่แบบมวลเดียว (One Mass System)

1.1.1 ลักษณะของปัญหา (Problem Statement)

ตัวอย่างปัญหานี้ จะพิจารณาระบบที่มีมวลเพียงมวลเดียวคือมวล m โดยมีอินพุตเดียวคือ u ที่ถูกกำหนดให้เป็นส่วนที่ป้อนเข้า (Actuator input) ดังภาพประกอบ 5



ภาพประกอบ 5 แสดงรูปแบบการเคลื่อนที่แบบมวลเดียว (One Mass System)

$$mx^{(2)} = u \quad (4.1)$$

จากปัญหาทางด้านระบบพลศาสตร์ข้างต้น ฟังก์ชันที่กำหนดตามสมการที่ (4.1) ที่ซึ่ง $u(t)$ เป็นอินพุต (actuator input) และมวล $m = 1$ โดยที่รูปของ state space คือ

$$\dot{x}^{(1)} = x_1, \quad (4.2)$$

$$\dot{x}_1^{(1)} = u, \quad (4.3)$$

โดยที่

$$x(0) = 0 \quad (4.4)$$

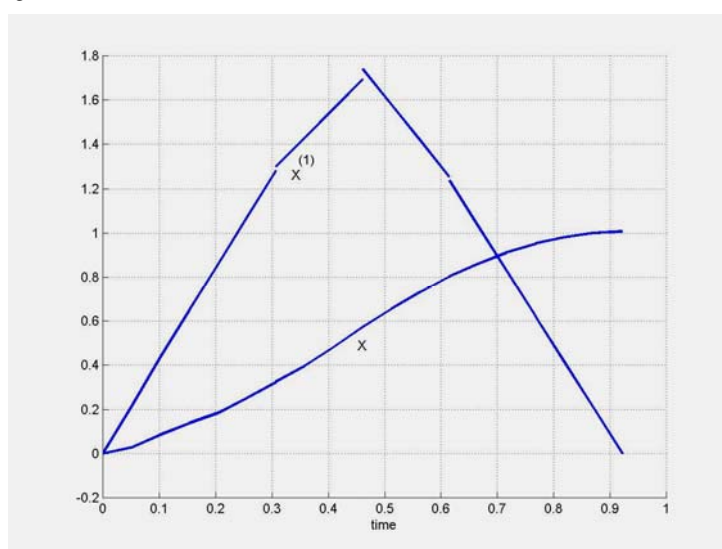
$$x^{(1)}(0) = 0 \quad (4.5)$$

$$x(1) = 1 \quad (4.6)$$

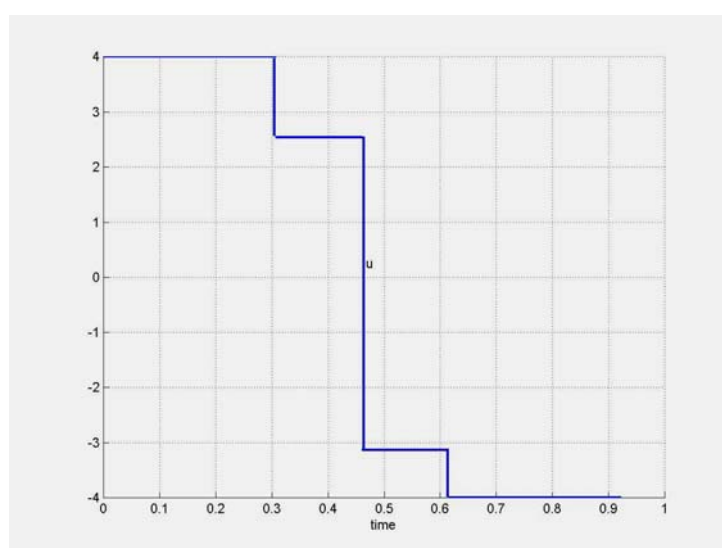
$$x^{(1)}(1) = 0 \quad (4.7)$$

1.1.2 ผลที่ได้จากการรันโปรแกรม

จากปัญหาข้างต้น จะได้ผลการรันโปรแกรมหังภาพประกอบ 6 และ 7



ภาพประกอบ 6 แสดงรูปเส้น Curve ของ $x^{(1)}$ และ x



ภาพประกอบ 7 แสดงรูปเส้น Curve ของ u

1.1.3 ผลการรันโปรแกรมเปรียบเทียบ CPU Run-time

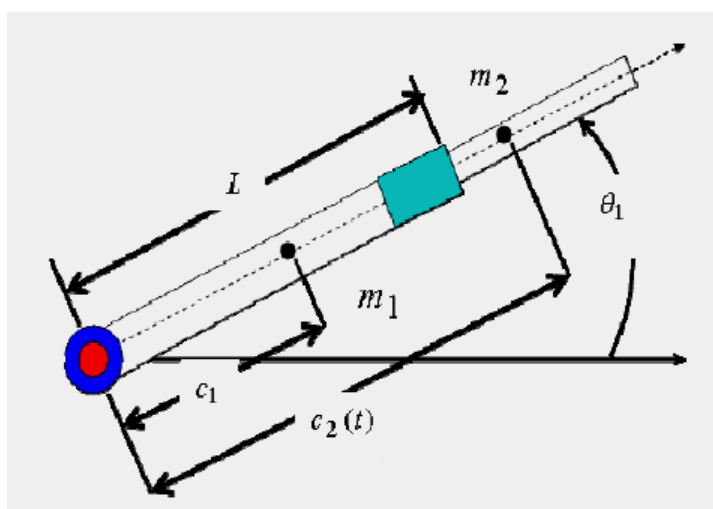
ตาราง 1 ผลการรันโปรแกรมเปรียบเทียบ CPU Run-time (One mass system)

ลำดับ ที่	รายละเอียด	Elapsed time (Second)		Time Reduced
		Zero Guess	New Guess	
1	One mass system (Direct)	34.320	5.880	82.86%
2	One mass system (Indirect)	38.110	37.510	1.57%

1.2 ตัวอย่างปัญหาหุ่นยนต์แขนกลแบบโพลาร์ (The Polar Manipulator)

1.2.1 ลักษณะของปัญหา (Problem Statement)

ตัวอย่างปัญหานี้ จะพิจารณาระบบพลศาสตร์ของหุ่นยนต์แขนกลแบบโพลาร์ ส่วนที่เป็นส่วนป้อนเข้า (Input) ระบบมี 1 องศาอิสระดังภาพประกอบ 8



ภาพประกอบ 8 แสดงรูปหุ่นยนต์แขนกลแบบโพลาร์ (Polar Manipulator)

และ มีสมการการเคลื่อนที่คือ

$$\begin{aligned}
 u_1 = & m_1 g c_1 \cos(\theta_1) + m_2 g c_2 \cos(\theta_1) \\
 & + (c_1^2 m_1 + I_{g1} + I_{g2} + m_2 c_2^2) \ddot{\theta}_1 + 2m_2 c_2 \dot{c}_2 \dot{\theta}_1
 \end{aligned} \tag{4.8}$$

$$u_1 = -m_2 g \sin(\theta_1) - m_2 \ddot{c}_2 + m_2 c_2 \dot{\theta}_1^2 \quad (4.9)$$

เมื่อ $u_1(t)$ และ $u_2(t)$ คือตัวป้อนเข้า (Actuator input) ที่มีความสัมพันธ์เกี่ยวเนื่องกับตัวแปร $\theta_1(t)$ และ $c_2(t)$ ตามลำดับ โดยที่รูปของ state space คือ

$$\begin{aligned} \dot{\theta}_1 &= \theta_2, \\ \dot{\theta}_2 &= \frac{-m_1 g c_1 \cos(\theta_1) - m_2 g c_2 \cos(\theta_1) - 2m_2 c_2 c_3 \theta_2 + u_1}{c_1^2 m_1 + I_{g1} + I_{g2} + m_2 c_2 \dot{\theta}_1} \end{aligned} \quad (4.10)$$

โดยที่

$$\begin{aligned} \dot{c}_2 &= c_3, \\ c_3 &= -g \sin(\theta_1) + c_2 \theta_2^2 - \frac{u_2}{m_2} \end{aligned} \quad (4.11)$$

ในการควบคุมระบบจากการกำหนดเซตของเงื่อนไขเริ่มต้นของ $\theta_1, \dot{\theta}_1, c_2$, และ $\dot{c}_2$ ที่เวลาเริ่มต้น t_0 โดยระบุจุดเป้าหมายไว้ที่ t_f ขณะที่ค่าคอสต่ำสุด (Minimum Cost) คือ

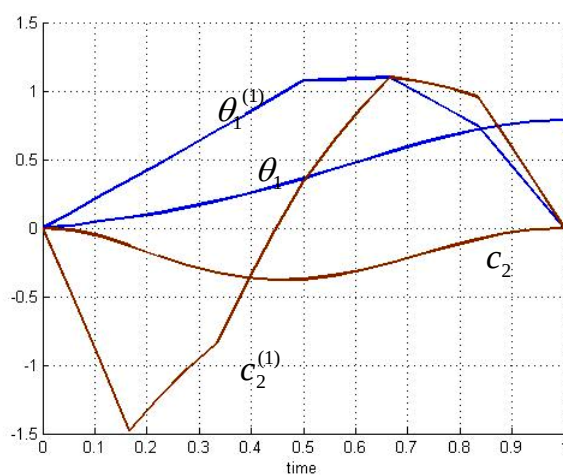
$$J = \int_{t_0}^{t_f} (u_1^2 + u_2^2) dt \quad (4.12)$$

วิธีการเคลื่อนที่ที่จะต้องเป็นไปตามเงื่อนไขบังคับ $-50 \leq u_1 \leq 50$ และ $-50 \leq u_2 \leq 50$ ในระหว่างการเคลื่อนที่ ค่าพารามิเตอร์ที่ใช้ในหน่วยระบบ MKS ดังนี้ $L = 1.0$, $m_1, m_2 = 1.0, c_1 = 3, g = 9.8, I_{g1} = I_{g2} = 1.0$, เงื่อนไขขอบเขต ทั้ง 2 จะกำหนดโดย

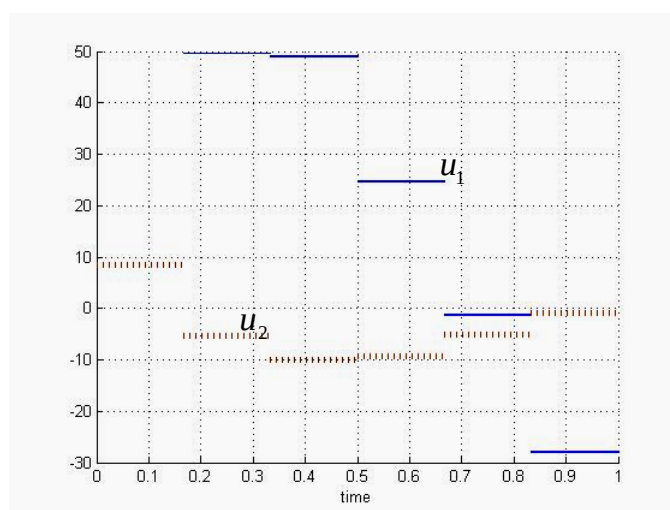
$$\theta_1(0) = 0, \dot{\theta}_1(0) = 0, c_2(0) = 0, \dot{c}_1(0) = 0, \quad (4.13)$$

1.2.2 ผลที่ได้จากการรันโปรแกรม

จากปัญหาข้างต้น จะได้ผลการรันโปรแกรกดังภาพประกอบ 9 และ 10



ภาพประกอบ 9 แสดงรูปเส้น Curve ของ θ_1 , $\theta_1^{(1)}$, c_2 และ $c_2^{(1)}$



ภาพประกอบ 10 แสดงรูปเส้น Curve ของ u_1 และ u_2

1.2.3 ผลการรันโปรแกรมเปรียบเทียบ CPU Run-time

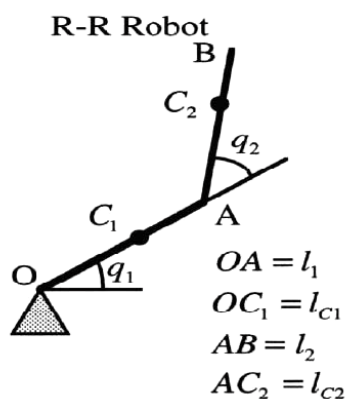
ตาราง 2 ผลการรันโปรแกรมเปรียบเทียบ CPU Run-time (The Polar Manipulator)

ลำดับ ที่	รายละเอียด	Elapsed time (Second)		Time Reduced
		Zero Guess	New Guess	
1	Polar Manipulator (Direct)	11.100	5.880	47.03%
2	Polar Manipulator (Indirect)	26.030	24.120	7.34%

1.3 ตัวอย่างปัญหาหุ่นยนต์แบบ 2 ชั้นต่อโยง (The Planar Two-link Manipulator)

1.3.1 ลักษณะของปัญหา (Problem Statement)

ตัวอย่างปัญหานี้ จะพิจารณาระบบพลศาสตร์ของหุ่นยนต์ 2 ชั้นต่อโยงแบบหมุน (Two-link Revolute Robot) โดยพิจารณาว่าแขนกลไม่มีโมเมนต์ความเฉื่อย ระบบมี 2 องศาอิสระ ดังภาพประกอบ 8



ภาพประกอบ 11 แสดงรูปแขนกล 2 ชั้นต่อโยง (Planar Two-link Manipulator)

และ มีสมการการเคลื่อนที่คือ

$$\begin{aligned}
 & (m_2 l_{c_2}^2 + m_2 l_{c_2} l_1 \cos(q_2)) \ddot{q}_2 + m_2 l_1^2 + m_1 l_{c_1}^2 \\
 & + m_2 l_{c_2}^2 + 2m_2 l_{c_2} l_1 \cos(q_2)) \ddot{q}_1 + m_2 g l_1 \cos(q_1) \\
 & + m_2 g l_2 \cos(q_1 + q_2) - 2m_2 l_{c_2} l_1 \sin(q_2) \dot{q}_2 \dot{q}_1 \\
 & - m_2 l_{c_2} l_1 \sin(q_2) \dot{q}_2^2 + m_1 g l_1 \cos(q_1) = u_1
 \end{aligned} \tag{4.14}$$

โดยที่รูปของ state space คือ

$$\dot{q}_1 = q_3 \tag{4.15}$$

$$\dot{q}_2 = q_4 \tag{4.16}$$

ในการควบคุมระบบจากการกำหนดเซตของเงื่อนไขเริ่มต้นบน $q_1, \dot{q}_1, q_2$, และ $\dot{q}_2$ ที่เวลาเริ่มต้น t_0 โดยระบุจุดเป้าหมายไว้ที่ t_f ขณะที่ค่าคอสต่ำสุด (Minimum Cost) คือ

$$J = \int_{t_0}^{t_f} (u_1^2 + u_2^2) dt \tag{4.17}$$

วิธีการเคลื่อนที่ที่จะต้องเป็นไปตามเงื่อนไขบังคับ $-100 \leq u_1 \leq 100$ และ $-100 \leq u_2 \leq 100$ ในระหว่างการเคลื่อนที่ ค่าพารามิเตอร์ที่ใช้ในหน่วยระบบ MKS ดังนี้ $m_1 = 1.3, m_2 = 1.6, l_1 = l_2 = 1.0, g = 9.8, l_{c_1} = l_{c_2} = 1.0$, เงื่อนไขขอบเขต ทั้ง 2 จะกำหนดโดย

$$q_1(0) = \frac{\pi}{6}, \dot{q}_1 = 0 \tag{4.18}$$

$$q_2(0) = \frac{\pi}{7}, \dot{q}_2(0) = 0 \tag{4.19}$$

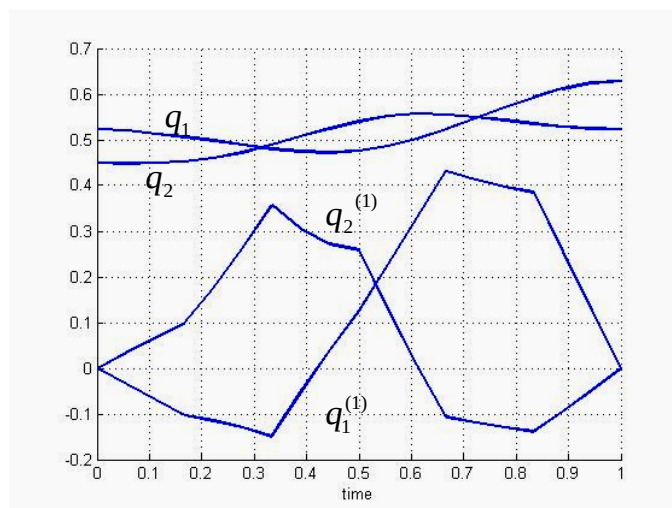
$$q_1(0) = \frac{\pi}{5}, \dot{q}_1 = 0 \tag{4.20}$$

$$q_2(1) = \frac{\pi}{6} \tag{4.21}$$

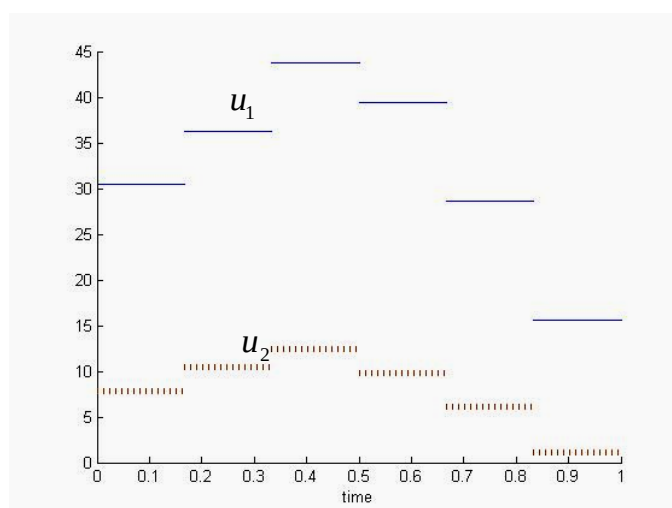
$$q_2(1) = 0 \tag{4.22}$$

1.3.2 ผลที่ได้จากการรันโปรแกรม

จากปัญหาข้างต้น จะได้ผลการรันโปรแกรมหังภาพประกอบ 12 และ 13



ภาพประกอบ 12 แสดงรูปเส้น Curve ของ q_1 , $q_1^{(1)}$, q_2 และ $q_2^{(1)}$



ภาพประกอบ 13 แสดงรูปเส้น Curve ของ u_1 และ u_2

1.3.3 ผลการรันโปรแกรมเปรียบเทียบ CPU Run-time

ตาราง 3 ผลการรันโปรแกรมเปรียบเทียบ CPU Run-time (Planar Two-link Manipulator)

ลำดับ ที่	รายละเอียด	Elapsed time (Second)		Time Reduced
		Zero Guess	New Guess	
1	Planar Two-link Manipulator (Direct)	63.220	19.770	68.73%
2	Planar Two-link Manipulator (Indirect)	195.81	126.39	35.45%

2. วิเคราะห์ผลการทดสอบ

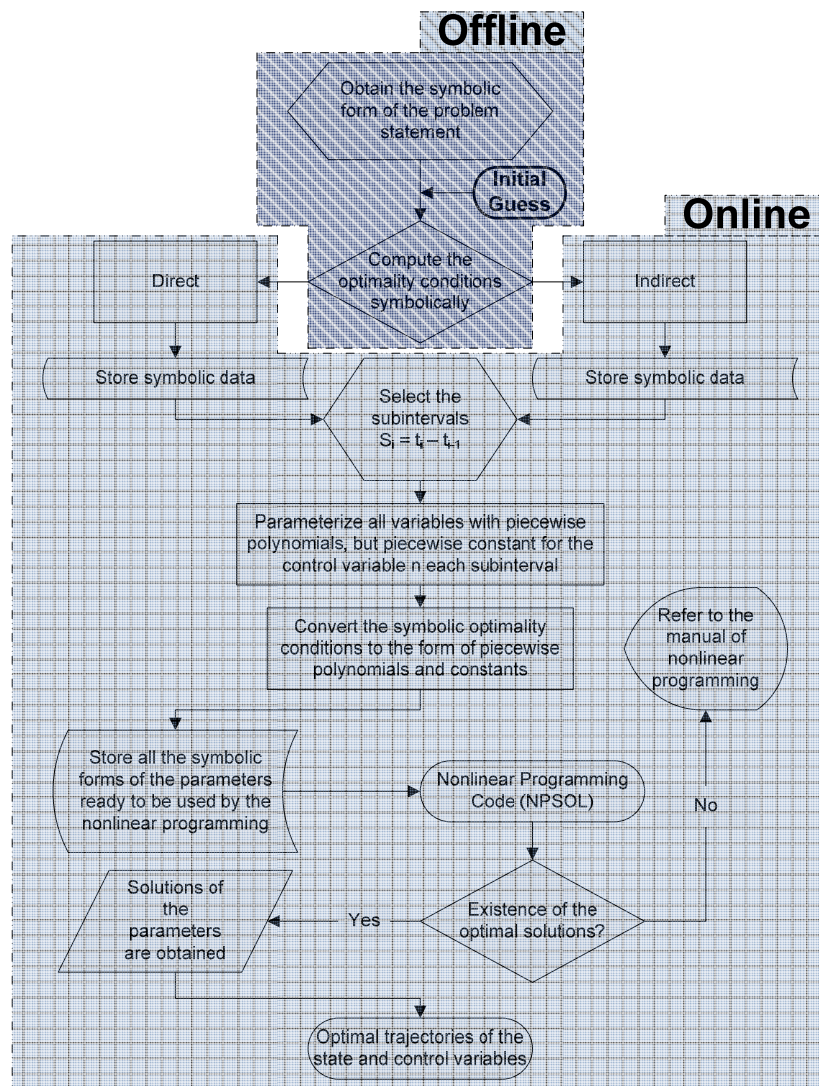
จากการผลการทดสอบรันโปรแกรมที่ได้มาจากการดำเนินการปรับปรุง และออกแบบโปรแกรมคอมพิวเตอร์ (Computer Programming) ตามวิธีการดำเนินงานวิจัยในบทที่ 3 จะเห็นได้ว่า ค่าของ CPU Run-time ของการรันโปรแกรมที่ได้มีการปรับปรุงค่าเริ่มต้น (Initial Guess) ก่อนนำไปคำนวณในขั้นตอนต่อไปตามขั้นตอนและกระบวนการของโปรแกรมเดิมนั้น จะมีค่าน้อยกว่าค่า CPU Run-time ของโปรแกรมเดิมที่ใช้รันโดยกำหนดค่าเริ่มต้นเป็น 0 (Zero Guess)

บทที่ 5

สรุปและวิจารณ์ผลการทดสอบ

1. สรุปและวิจารณ์ผลการทดสอบ

จากขั้นตอนเริ่มต้นซึ่งเป็นขั้นตอนที่ 1 ของโปรแกรมที่เรียกว่าส่วนออฟไลน์ (Offline) จะเป็นการอินพุต (Input) ค่าต่างๆ ที่จำเป็นสำหรับโปรแกรม รวมทั้งการสร้างรูปแบบของปัญหา (Problem Statement) ของปัญหาต่างๆ ซึ่งโปรแกรมจะรับค่าต่างๆ ที่เป็นองค์ประกอบของปัญหาแต่ละปัญหา จากข้อมูลที่ได้จัดเตรียมไว้ในรูปแบบของ M-file ไว้ก่อนแล้ว และนำค่าต่างๆ ดังที่กล่าวไว้ข้างต้น นำไปแทนค่าในส่วนที่ได้เขียนโปรแกรมเพิ่มเติมเพื่อที่จะหาค่าเริ่มต้นที่เหมาะสม



ภาพประกอบ 14 แสดงรูปแบบของขั้นตอนออฟไลน์ (Offline) และออนไลน์ (Online) ของโปรแกรม

หลังจากนั้นในขั้นตอนต่อไป จึงจะนำค่าเริ่มต้นที่ได้ ไปใช้ในส่วนของการแก้ปัญหาของโปรแกรมที่ไม่เป็นเชิงเส้น (Nonlinear Programing) หรือขั้นตอนที่ 2 ของโปรแกรม ซึ่งส่วนนี้เรียกว่าส่วนออนไลน์ (Online) โดยโปรแกรมจะแบ่งออกเป็นการแก้ปัญหาโดยวิธี Direct และ Indirect ดังภาพประกอบ 14

และในการออกแบบส่วนเพิ่มเติมของโปรแกรมเพื่อให้ได้ค่าเริ่มต้น (Initial Guess) ที่มีความเหมาะสมกว่าที่เป็นอยู่เดิมนั้น หลังจากที่ได้นำตัวอย่างปัญหาทางด้านพลศาสตร์มาทดลองจำนวนทั้งสิ้น 3 ตัวอย่าง นำมารันโดยโปรแกรมเดิมและโปรแกรมใหม่ที่ได้ปรับปรุงเพื่อเปรียบเทียบกัน และจากผลการทดลองรันโปรแกรมดังกล่าว จะพบว่ากรณีที่ใช้โปรแกรมเดิมที่ยังไม่มีการปรับปรุงค่าเริ่มต้นก่อน (Zero Guess) จะมีค่า CPU Run-time มากกว่ากรณีที่ใช้โปรแกรมที่ได้รับการออกแบบส่วนเพิ่มเติมในการปรับปรุงค่าเริ่มต้นให้เหมาะสม (Initial Guess) ดังที่แสดงไว้ในตาราง 4

ตาราง 4 สรุปผลการรันโปรแกรมเปรียบเทียบ CPU Run-time ของแต่ละปัญหา

ลำดับที่	รายละเอียด	Elapsed time (Second)		Time Reduced
		Zero Guess	New Guess	
1	One mass system (Direct)	34.320	5.880	82.86%
2	One mass system (Indirect)	38.110	37.510	1.57%
3	Polar Manipulator (Direct)	11.100	5.880	47.03%
4	Polar Manipulator (Indirect)	26.030	24.120	7.34%
5	Planar Two-link Manipulator (Direct)	63.220	19.770	68.73%
6	Planar Two-link Manipulator (Indirect)	195.81	126.39	35.45%

จากตารางสรุปผลการรันโปรแกรมเปรียบเทียบ CPU Run-time ของแต่ละปัญหาในตาราง 4 สามารถสรุปโดยแยกเป็น 2 ส่วนคือ ส่วนที่ใช้โปรแกรมโดยวิธีตรง (Direct Method) และส่วนที่ใช้โปรแกรมโดยวิธีอ้อม (Indirect Method) ดังนี้ ซึ่งเวลาที่ลดลง (Time Reduced) ของ

โปรแกรมโดยวิธีตรงจะลดลงโดยเฉลี่ย ประมาณ 66.20% และโดยวิธีอ้อมจะลดลงโดยเฉลี่ย ประมาณ 15%

2. ข้อเสนอแนะ

จากผลการรันโปรแกรมเพื่อเปรียบเทียบ CPU-Runtime ของโปรแกรมเดิมที่ใช้ค่าเริ่มต้นเป็น 0 (Zero Guess) กับโปรแกรมที่ปรับปรุงใหม่โดยใช้ค่าเริ่มต้น (Initial Guess) ที่ไม่เป็น 0 ของทั้งโดยวิธีตรง (Direct) และวิธีอ้อม (Indirect) จะเห็นว่าเปอร์เซ็นต์ของเวลาที่ลดลงโดยวิธีอ้อม มีเปอร์เซ็นต์น้อยมาก ทั้งนี้ ผู้วิจัยมีข้อเสนอแนะว่า ในอนาคตให้เน้นในส่วนของการแก้ปัญหาโดยวิธีอ้อม (Indirect) โดยการนำเอา Necessary Condition จากกระบวนการทางด้าน Calculus of Variation มาเป็น Initial Guess แทน

บรรณานุกรม

บรรณานุกรม

- Agrawal, S.K., Veeraklaew, T.; & B.C. Fabien. (1998, July). .Direct and Indirect Optimization of Linear Time-Invariant Dynamic Systems Using Transformation. *Optimal Control Applications and Methods*. 19: 393-410.
- Singiresu S.R. (1996). *Engineering Optimization Theory and Practice*. 3rd ed. USA: John Wiley & sons Inc.
- Sunil K.A.; & C.F. Brian. (1999). *Optimization of Dynamic Systems*. Netherlands: Kluwer Academic Publishers.
- Thomas D.G. (1992). *Fundamentals of Vehicle Dynamics*. 4th ed. USA: Society of Automotive Engineers Inc.
- Veeraklaew V. (1999). *Extensions of Optimization Theory and New Computational Approaches for Higher-order Dynamic Systems*. Dissertation Ph.D.(Mechanical Engineering). Delaware: Graduate School of University of Delaware. Photocopied.

ภาคผนวก

ตัวอย่างโปรแกรม

ตัวอย่างโปรแกรม สำหรับกรณี One Mass System

user.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% This matlab file is called user.m which need information from user
%
% in order to run the higher-order optimization algorithm which will
%
% result as optimal state and control trajectories. Therefore, user
%
% has to follow the instruction closely without changing the fixed
%
% variables in this matlab file, also this file needed Matlab version
%
% 5.1 such that the algorithm can be run.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% 1. Number of state variables...format>>> n=?;
%
n=2;
%
%
% 2. Number of control variables...format>>> m=?;
%
m=1;
%
%
% 3. Number of the highest derivative of state variable...
%
format>>> p=?;
%
p=1;
%
ps=[1;1];
%
%
% 4. Number of the intervals***...format>>> N=?;
%
% This interval means that users can set the accuracy of the
%
% solutions. If N is large, it will be more accuracy of the
% solutions,%
% but not always*.
%

```

```

N=6;
%

%
% 5. Initial time...format>>> t0=?;
%
t0=0;
%

%
% 6 Final time...format>>> tf=?.
%
%   where N=N in section4.
%
% tf=1;
%
t_l=0; %assigned lower bound of final time
%
t_u=2; %assigned upper bound of final time
%
t_g=1; %assigned guess of final time
%

%
% 7. This place is where the state equations have to be defined as
%
%format below:
%
%   Format>>> fs(1,1)=sym('?');, ..., fs(n,1)=sym('?');
%
M1=sym(['M1']);
%
fs(1,1)=sym('x11-x20');
%
fs(2,1)=sym('x21-u1');
%

%
% 8. Also, at this point, users need to provide all constraints.
%
%   Format>>> cu(1,1)=sym('?');, ..., cu(m,1)=sym('?'); where m is
%   the number of the control input. cu must be defined for all
%   control variables.*****
%
%   Format>>> cx(1,1)=sym('?');, ..., cx(r,1)=sym('?'); where r is
%   the number of the constraints
%
%   Detail:
%
%   cu means constraints contain only control inputs.
%
%   cx means constraints contain only state or both control and sta-
%   te variables.
%

```

```

% 9.1 Further more for the lower and upper bound of the
inequality%
% constraints. If one has the equality constraints, the lower
%
% and upper bounds must defined as the same values.
%
% Format>>> cu_l(1,1)=?;, ..., cu_l(m,1)=?;
%
% cu_u(1,1)=?;, ..., cu_u(m,1)=?;
%
% and so on where l and u mean upper/lower bound
respectively.%
%*****
%
cu(1,1)=sym('u1');
%
%cu(2,1)=sym('u2');
%

%
cu_l(1,1)=-4;
%
cu_u(1,1)= 4;
%
%cu_l(2,1)=-4;
%
%cu_u(2,1)= 4;
%
%*****
%

%
r=0; % remember this is a number of the constraint of x and u.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%cx(1,1)=sym('x12');
%
%cx_l(1,1)=-4;
%
%cx_u(1,1)=4;
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%

%
% 9. Boundary Conditions
%
% Format>>> x0(i)=?; and xf(i)=?;
%
% where x0(1) =x10dot, x0(2) =x11dot, ..., x0(p) =x1pdot
%
% x0(p+1)=x20dot, x0(p+2)=x21dot, ..., x0(2p)=x2pdot
%
% ...
%
% x0((n-1)p+1)=xn0dot, x0((n-1)p+2)=xn1dot, ..., x0(np)=xnpdot,
%
```

```

%      and repeat with the same logic for xf(1),...,xf(np).
%

%
x0(1,1)=sym('x10');
%
x0(2,1)=sym('x20');
%
%x0(3,1)=sym('x20');
%
%Bound of x0
%
x0_l(1,1)=0;
%
x0_l(2,1)=0;
%
%x0_l(3,1)=0;
%
x0_u(1,1)=0;
%
x0_u(2,1)=0;
%
%x0_u(3,1)=0;
%

%
xf(1,1)=sym('x10');
%
xf(2,1)=sym('x20');
%
%xf(3,1)=sym('x20');
%
%Bound of xf
%
xf_l(1,1)=1;
%
xf_l(2,1)=0;
%
%xf_l(3,1)=1;
%
xf_u(1,1)=1;
%
xf_u(2,1)=0;
%
%xf_u(3,1)=1;
%

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%

%
% 11. Define the minmax values of all parameters.
%

%
absmax=1e3;
%
```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%

```

```

%*****end of user.m
file*****

```

```

guess.m

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% This M-file named guess is the most important file to provide an
%
%initial guesses for the code...
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% 1. calculate ti except t0 and tf.
%
tf=t_g;
%
nn0=num2str(N);
%
eval(['t' nn0 ' = tf;']);
%
for i=1:N-1,
%
    nn1=num2str(i);
%
    nn2=t0+i*(tf-t0)/N;
%
    eval(['t' nn1 ' = nn2;']);
%
end
%
pp=sum(ps);
%
Nn1=(pp*(N+1));
%
Nn2=(n*nv*N);
%
L=sym([zeros(Nn1,Nn2)]);
%
for i=1:N+1,
%
    nn1=num2str(i-1);
%
    for j=1:n,
%
        for l=1:nv,
%
            nn2=num2str(l-1);
%
            if i==N+1,
%

```

```

L(j+(i-1)*pp, (j-1)*nv+l+(i-2)*nv*n)=sym(['t'nn1 '^nn2]);
%
d=L(j+(i-1)*pp, (j-1)*nv+l+(i-2)*nv*n);
%
for k=1:ps(j)-1,
%
    L(j+(i-1)*pp+k*n, (j-1)*nv+l+(i-2)*nv*n)=diff(d);
%
    d=L(j+(i-1)*pp+k*n, (j-1)*nv+l+(i-2)*nv*n);
%
end
%
else
%
L(j+(i-1)*pp, (j-1)*nv+l+(i-1)*nv*n)=sym(['t'nn1 '^nn2']);
%
d=L(j+(i-1)*pp, (j-1)*nv+l+(i-1)*nv*n);
%
for k=1:ps(j)-1,
%
    L(j+(i-1)*pp+k*n, (j-1)*nv+l+(i-1)*nv*n)=diff(d);
%
    d=L(j+(i-1)*pp+k*n, (j-1)*nv+l+(i-1)*nv*n);
%
end
%
end
%
end
%
end
%
for i=1:N-1,
%
    nn1=num2str(i);
%
    for j=1:n,
%
        for l=1:nv,
%
            nn2=num2str(l-1);
%
            L(j+(i)*pp, (j-1)*nv+l+(i-1)*nv*n)=sym(['-t'nn1 '^nn2']);
%
            d=L(j+(i)*pp, (j-1)*nv+l+(i-1)*nv*n);
%
            for k=1:ps(j)-1,
%
                L(j+(i)*pp+k*n, (j-1)*nv+l+(i-1)*nv*n)=diff(d);
%
                d=L(j+(i)*pp+k*n, (j-1)*nv+l+(i-1)*nv*n);
%
            end
%
        end
%
    end
%
end

```

```

        end
    %
end
%
%*****
%
for i=1:Nn1,
%
    for j=1:Nn2,
%
        L1(i,j)=eval(L(i,j));
%
    end
%
end
%
%
%
Linv=pinv(L1);
%
Nnx=pp*(N-1);
%
xxx=zeros(Nnx,1);
%
%
%
R=[x0_l;
%
    xxx;
%
    xf_l];
%
%
%
yc=Linv*R;
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
pre.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   This M-file called pre.m which will provide constraints and objec-
%
%   tive function for funcon.m and funobj.m in the symbolic form at
%
%   %r_c.m and r_o.m respectively.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%*****
clear          %
funcont        %
clear          %
funobjt        %

```

```

%*****

%*****End of
pre.m*****
main_guess.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%
% This M-file called main_guess.m which is the main program that
user %
%need to run for their optimal solutions. The optimal solutions
%
%are obtained using the nonlinear programming called NPSOL package.
%
%INSTRUCTION HOW TO USE THIS CODE:
%
% a) User need to modify M-file named user.m by following
%
% the instruction that embeded inside closely.
%
% b) The second step is preparing all symbolic code for
%
% all constraints and objective functions by typing
%
% command "pre" on Matlab cammand window and disregard all
%
% warnings that appeared on the screen.
%
% c) The last step to run this code is by typing
%
% command "main" on Matlab cammand window.
%
%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%
clear
close all

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 1. Obtaining all inputs from user.m, initial guesses, and %
% # of variables. %
user %
    nv1=(1*n*p*(N+1)+2*n*N+2*r*N)/((1*n)*N); %
    nv2=floor((1*n*p*(N+1)+2*n*N+2*r*N)/((1*n)*N)); %
    if nv1==nv2, %
        nv=nv1; %
    else %
        nv=nv2+1; %
    end %
if m==0, %
    cu=[]; %
    cu_l=[]; %
    cu_u=[]; %
else %
    cu=cu; %
    cu_l=cu_l; %
    cu_u=cu_u; %

```


funcont.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
allnec %This M-file will prepare all symbolic computations.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% 1. calculate ti except t0 and tf.
%
t=sym('t');
%
tf=sym('tf');
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%2.Preparing F(n_states),
%
% XX(np_continuities_state), CX(r_constraints), B0(np_initial),
%
% and Bf(np_final).
%
for i=1:n,
%
    nnn1=num2str(i);
%
    if fs(i)==0,
%
        d1=sym([zeros(N,1)]);
%
    else
%
        d1=sym([zeros(N,1)]);
%
        for ii=1:N,
%
            df(ii,1)=fs(i);
%
        end
%
        for j=1:N,
%
            nn3=num2str(j);
%
            for k1=1:n,
%
                for k2=1:p+1,
%
                    nn1=num2str(k1);
%
                    k22=k2-1;
%
                    nn2=num2str(k22);
%
                    d1(j)=subs(df(j),sym(['x'nn1 nn2]),sym(['x'nn1 nn2 '('nn3 ')']));
%

```

```

        df(j)=d1(j);
    %
        end
    %
    end
    %
    for k1=1:m,
    %
        nn1=num2str(k1);
    %
        nn3=num2str(j);
    %
    d1(j)=subs(d1(j),sym(['u'nn1]),sym(['u'nn1 '('nn3 ')']));
    %
        end
    %
    end
    %
    df=d1;
    %
    end
    %
    for ii=1:N,
    %
        d1(ii)=eval(d1(ii));
    %
    end
    %
    F((i-1)*N+1:i*N,1)=d1;
    %
    end
    %
    %*****
    *
    ddp=0;
    %
    for ii=1:n,
    %
        dp=ps(ii);
    %
        for i=1:dp,
    %
            XX(ddp*N+(i-1)*N+1:ddp*N+(i)*N,1)=Xdp((ii-1)*N+1:ii*N,i);
    %
        end
    %
        ddp=dp+ddp;
    %
    end
    %
    %*****
    *
    if r==0,
    %
        CX=[];
    %
    else
    %

```

```

for i=1:r,
%
    nn1=num2str(i);
%
    if cx(i)==0,
%
        d1=sym([zeros(N,1)]);
%
    else
%
        d1=sym([zeros(N,1)]);
%
        for ii=1:N,
%
            dc(ii,1)=cx(i);
%
        end
%
        for j=1:N,
%
            nn3=num2str(j);
%
            for k1=1:n,
%
                for k2=1:2*p+1,
%
                    nn1=num2str(k1);
%
                    k22=k2-1;
%
                    nn2=num2str(k22);
%
                d1(j)=subs(dc(j),sym(['x'nn1 nn2]),sym(['x'nn1 nn2 '('nn3 ')']));
%
                dc(j)=d1(j);
%
            end
%
        end
%
        for k1=1:m,
%
            nn1=num2str(k1);
%
            nn3=num2str(j);
%
            d1(j)=subs(d1(j),sym(['u'nn1]),sym(['u'nn1 '('nn3 ')']));
%
        end
%
    end
%
    dc=d1;
%
end
%
for ii=1:N,
%

```

```

        d1(ii)=eval(d1(ii));
%
    end
%
    CX((i-1)*N+1:i*N,1)=d1;
%
end
%
end
%
%*****
%
pm=n*p-sum(ps);
%
pp=n*p-pm;
%
clear d1
%
clear d2
%
ddp=0;
%
for ii=1:n,
%
    dp=ps(ii);
%
    for i=1:dp,
%
        k=i+ddp;
%
        dc0(k,1)=x0(k);
%
        dcf(k,1)=xf(k);
%
        for k1=1:n,
%
            for k2=1:p+1,
%
                nn1=num2str(k1);
%
                k22=k2-1;
%
                nn2=num2str(k22);
%
                nn3=num2str(1);
%
                nn4=num2str(N);
%
            d1(k,1)=subs(dc0(k,1),sym(['x'nn1 nn2]),sym(['x'nn1 nn2...
%
                '('nn3 ')']),0);
%
            d2(k,1)=subs(dcf(k,1),sym(['x'nn1 nn2]),sym(['x'nn1 nn2...
%
                '('nn4 ')']),0);
%
            dc0(k,1)=d1(k,1);
%

```

```

    dcf(k,1)=d2(k,1);
%
        end
%
    end
%
end
%
ddp=dp+ddp;
%
end
%
for j=1:pp,
%
    dd1(j,1)=eval(d1(j));
%
    dd2(j,1)=eval(d2(j));
%
end
%
B0=dd1;
%
Bf=dd2;
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% 3. Substitute with all applicable ti.
%
%*****F to be
F_C*****%
F_C=sym([zeros(2*n*N,1)]);
%
for i=1:n,
%
    for j=1:N,
%
        k=j+(i-1)*N;
%
        for ii=1:2,
%
            ij=(ii-1)+(j-1);
%
            nn3=num2str(ij);
%
            d=sym(['t'nn3]);
%
            F_C((i-1)*2*N+ii+(j-1)*2)=subs(F(k),sym('t'),d);
%
        end
%
    end
%
end
end
%

```

```

%***Note for future works***
%
% d=sym(['t'num2str(1)])
%
% d=eval(sym(['t'num2str(1)]))
%
% subs(F_C(1),sym('t0'),d)
%
%*****
%
%*****CX to be
CX_C*****%
CX_C=sym([zeros(2*r*N,1)]);
%
for i=1:r,
%
    for j=1:N,
%
        k=j+(i-1)*N;
%
        for ii=1:2,
%
            ij=(ii-1)+(j-1);
%
            nn3=num2str(ij);
%
            d=sym(['t'nn3]);
%
            CX_C((i-1)*2*N+ii+(j-1)*2)=subs(CX(k),sym('t'),d);
%
        end
%
    end
%
end
%
%*****XX to be
XX_C*****%
for i=1:pp,
%
    for j=1:N,
%
        k=j+(i-1)*(N);
%
        nn1=num2str(j);
%
        nn2=num2str(j-1);
%
        d=sym(['t'nn1]);
%
        XXX(k,1)=subs(XX(k),sym('t'),d);
%
        dd=sym(['t'nn2]);
%
        XXXX(k,1)=subs(XX(k),sym('t'),dd);
%
    end
%
end

```



```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 4. Form the constraint matrix c.
ca=[B0_C;
    Bf_C;
    XX_C;
    F_C;
    CX_C];
si1=size(ca);
si2=si1(1);
in1=1*nv*n*N+N*m+1; % #of all variables including mu.
nn1=num2str(N);
nn2=num2str(in1);
for i=1:si2,
    cb(i,1)=subs(ca(i),sym(['t'nn1]),sym(['y'nn2]),0);
end
for i=1:si2,
    c(i,1)=subs(cb(i),sym(['tf']),sym(['y'nn2]),0);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

%***This is an option to provide the gradient of c*****
% This is a correct version, but beware of large size error. %
concj %
%*****

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%End of
funcont.m%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

funobjt.m

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
allnec %This M-file will prepare all symbolic computations.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% 1. calculate ti except t0 and tf.
%
t=sym('t');
%
tf=sym('tf');
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 2.Preparing He. The Hamiltonian Equation.
in1=1*nv*n*N+N*m+1; % #of all variables.
nn1=num2str(in1);
H=sym(['y'nn1]);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%*** This is an option for obtaining jacobian (objgrd)***
conoj %
%*****

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%End
funobjt.m%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

ประวัติย่อผู้วิจัย

ประวัติย่อผู้วิจัย

ชื่อ - ชื่อสกุล	นายธนาคม สวาสติพันธ์
วัน เดือน ปีเกิด	23 ธันวาคม 2521
สถานที่เกิด	อุบลราชธานี
สถานที่อยู่ปัจจุบัน	36/29 ซ.ศูนย์วิจัย 2 แขวงบางกะปิ เขตห้วยขวาง กรุงเทพมหานคร 10310

ประวัติการศึกษา

พ.ศ. 2537	มัธยมศึกษาตอนต้น จาก โรงเรียนอัสสัมชัญอุบลราชธานี
พ.ศ. 2540	มัธยมศึกษาตอนปลาย จาก โรงเรียนอัสสัมชัญอุบลราชธานี
พ.ศ. 2544	วิศวกรรมศาสตรบัณฑิต (วศ.บ. เครื่องกล) จาก มหาวิทยาลัยอุบลราชธานี
พ.ศ. 2552	บัณฑิตศึกษาวิศวกรรมศาสตรมหาบัณฑิต (วศ.ม. เครื่องกล) จาก มหาวิทยาลัยศรีนครินทรวิโรฒ