การหาค่าที่เหมาะสมในการออกแบบระบบควบคุมแบบ PID สำหรับวงจรแปลงผันบั๊กด้วยอัลกอริทึม HFPA

Optimization of PID Design for Buck Converter Using HFPA

จักราวุธ ศิรพัชรางกูร
บัณฑิตศึกษา สาขาวิชาวิศวกรรมระบบสมองกลฝังตัว คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีมหานคร
Email: konbahmcu@gmail.com

ธีรยศ เวียงทอง
ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

สุชาดา สิทธิ์จงสถาพร
สถาบันนวัตกรรมมหานคร คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีมหานคร

Manuscript received November 13, 2016
Revised February 13, 2017

บทคัดย่อ

งานวิจัยนำเสนอการพัฒนาการหาค่าที่เหมาะสมที่สุดของระบบควบคุมแบบสัดส่วน ปริพันธ์ อนุพันธ์ (Proportional–Integral–Derivative controller, PID) โดยการใช้อัลกอริทึมการผสมเกสร (Flower Pollination Algorithm, FPA) และอัลกอริทึมการค้นหาแบบนกดุเหว่า (Cuckoo Search Algorithm, CSA) เป็นการค้นหาค่าที่เหมาะสมที่สุดด้วยการค้นหาค่าสัมประสิทธิ์แบบทำซ้ำ สำหรับนำไปใช้งานวงจรแปลงผันบั๊ก (Buck Converter) โดยกระบวนการทั้งหมดได้ออกแบบและจำลองการทำงานทั้งหมดด้วยโปรแกรม Matlab/Simulink โดยเปรียบเทียบผลการทดลองด้วยเวลาการค้นหาและประสิทธิภาพซึ่งพบว่า FPA มีประสิทธิภาพที่ดีกว่า นอกจากนี้ผู้วิจัยได้เพิ่มประสิทธิภาพ FPA ด้วยการเพิ่มเติมลำดับชั้นการค้นหาให้เรียกว่า (Hierarchical Flower Pollination Algorithm, HFPA) ซึ่งทำให้ปรับปรุงคุณภาพของการค้นหาดีขึ้นโดยเทียบจากจำนวนการค้นหาที่เท่ากัน

ABSTRACT

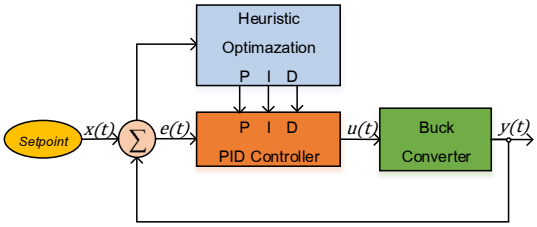
This paper presents the development of PID design optimization using Flower Pollination Algorithm (FPA) and Cuckoo Search Algorithm (CSA). The optimal PID coefficients are repeatedly searched for using in a DC Buck Converter. The process including the PID controller and the conventional buck converter are modelled and implemented in Matlab/Simulink. Simulation results, comparing in terms of searching time and solution quality, reveal that FPA performs better. Additionally, we extend the capability FPA by adding hierarchical search, called HFPA. The quality of the results can be improved in the same searching time.

1. บทนำ

การควบคุมแบบ PID เป็นพื้นฐานการควบคุมที่นิยมใช้งานโดยแพร่หลายในปัจจุบัน เนื่องจากมีโครงสร้างของอัลกอริทึมที่ใช้งานง่าย มีความเสถียร จึงสามารถนำไปใช้งานที่เกี่ยวกับการควบคุมในระดับโรงงานอุตสาหกรรม การทำงานของระบบควบคุมแบบ PID ต้องอาศัยการป้อนกลับทางด้านเอาต์พุตเพื่อใช้ในการประมวลผลหาค่าความผิดพลาด (Error) เพื่อปรับแต่งค่าพารามิเตอร์ให้ได้ค่า

เอาต์พุตตามความต้องการของผู้นำไปใช้งาน ซึ่งวิธีการหาค่าพารามิเตอร์มีหลากหลายวิธี

ในบทความนี้ได้นำเสนอปัญหาของวิธีการหาค่าความผิดพลาด ที่ใช้ในการทดลองเพื่อเปรียบเทียบหาวิธีการที่เหมาะสม สำหรับอัลกอริทึมการปรับแต่งค่าพารามิเตอร์ โดยการใช้อัลกอริทึมการผสมเกสรดอกไม้ เปรียบเทียบกับวิธีการค้นหาแบบนกดุเหว่า และพัฒนาอัลกอริทึมการผสมเกสรดอกไม้แบบลำดับขั้น เพื่อนำมาใช้งานร่วมกับระบบควบคุมแบบ PID เพื่อควบคุมวงจรแปลงผันบั๊ก (Buck converter) ที่เป็นวงจรแปลงระดับแรงดันไฟฟ้ากระแสตรงทางด้านอินพุต ที่มีระดับแรงดันคงที่ให้มีแรงดันที่ลดลงตามความต้องการของผู้ออกแบบหรือผู้ใช้งานหรือที่เรียกกันว่าวงจรลดแรงดันไฟฟ้ากระแสตรง ดังรูปที่1



รูปที่ 1 *ผังจำลองการทำงานทั้งระบบ*

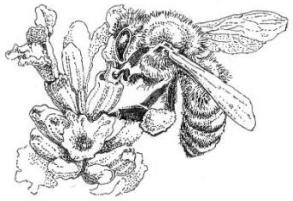
2. อัลกอริทึมแบบฮิวริสติก

เนื่องจากการหาค่าสัมประสิทธิ์ (Coefficient) ของตัวแปรใน PID นั้นมีความสำคัญในการกำหนดการทำงานของระบบ ซึ่งมีหลายๆ วิธีในการหา บทความนี้เลือกใช้อัลกอริทึมที่เป็นลักษณะของฮิวริสติกอัลกอริทึม (Heuristic algorithms) ซึ่งสามารถหาคำตอบที่ดีในเวลาที่ยอมรับได้

2.1 อัลกอริทึมผสมเกสรดอกไม้

พื้นฐานของอัลกอริทึมผสมเกสรดอกไม้ได้แรงบันดาลใจมาจากวิธีการธรรมชาติ [3]. ถูกพัฒนาขึ้นมาโดย Xin-She Yang ในปี 2012 จุดประสงค์ของอัลกอริทึมการผสมเกสรดอกไม้คือ การสืบพันธุ์ของพืชดอกโดยอาศัยการถ่ายโอนข้อมูลที่เป็นละอองเกสรดอกไม้ และกระบวนการสืบพันธุ์ดังกล่าวมักจะมีการเชื่อมโยง และเกี่ยวข้องกับการเคลื่อนย้ายของละอองเกสร ซึ่งจำเป็นต้องอาศัยภาหะถ่ายเรณู (Pollinator) ในการขนส่งละอองเกสรไปยังพืชดอกอื่นๆ เช่น นก ค้างคาว ผีเสื้อ และผึ้ง

วิธีการผสมเกสรดอกไม้สามารถใช้ได้สองวิธีหลักคือ วิธีการผสมเกสรแบบอาศัยสิ่งไม่มีชีวิต (Abiotic Pollination) เป็นวิธีการผสมเกสรโดยใช้สิ่งรอบ ๆ ตัวที่ไม่มีชีวิต เช่น ใช้ลมในการแพร่กระจายเกสรยกตัวอย่างเช่น ดอกหญ้าที่ปลิวตามลม เพื่อแพร่กระจายเกสรไปในที่ต่าง ๆ ถือว่าเป็นการผสมเกสรแบบท้องถิ่น (Local) และอีกวิธีคือการผสมเกสรแบบอาศัยสิ่งมีชีวิต (Biotic Pollination) ยกตัวอย่างเช่นผึ้งในรูปที่ 2 มากินน้ำหวานที่ดอกไม้และทำให้ขาหรือลำตัวมีเกสรของดอกไม้นั้นๆติดตามตัว เกสรเหล่านี้ก็จะถูกผีเสื้อนำพาไปยังดอกไม้อื่นๆเพื่อผสมเกสรต่อไปถือว่าเป็นการผสมเกสรแบบครอบคลุม (Global) โดยประมาณ 90% ของการพืชดอกทั่วไปจะใช้วิธีการผสมโดยวิธี Biotic และอีกประมาณ 10% จะใช้รูปแบบการผสมแบบ Abiotic



รูปที่ 2 *การผสมเกสรโดยอาศัยสิ่งมีชีวิต*
(ที่มา : Worker with pollen ball, Craig Latker)

เห็นได้ว่าการผสมแบบใช้การพึ่งพาสิ่งมีชีวิตในการผสมพันธ์นั้นมีสัดส่วนในการทำงานที่มากกว่าเพราะ การผสมเกสรข้ามดอกโดยพึ่งพาสิ่งมีชีวิตยกตัวอย่างผึ้ง เป็นการผสมเกสรข้ามดอกที่ดีและยังสามารถพัฒนาสายพันธุ์ที่ดีให้กับดอกไม้ได้ เช่น ผึ้งมีแนวโน้มที่จะนำเกสรดอกไม้ก่อนหน้าที่ได้ไปเก็บน้ำหวานมาแล้วไปยังดอกไม้ที่แข็งแรงซึ่งจะทำให้เกิดความมั่นคงของดอกไม้ (Flower Constancy) คือความมั่นคงที่มีแนวโน้มการโอนถ่ายละอองเกสรของผึ้งจะไปยังดอกอื่น ๆ ไม่ว่าจะเป็นดอกไม้สายพันธุ์เดียวกันมีน้ำหวานที่มากกว่า หรือดอกไม้สายพันธ์ที่พิเศษอื่นๆจึงทำให้คุ้มค่าและได้เปรียบเรื่องวิวัฒนาการของสายพันธุ์ที่ดีขึ้น

กระบวนการผสมเกสรแบบครอบคลุม และ กระบวนการผสมเกสรแบบท้องถิ่น การผสมเกสรดอกไม้แบบครอบคลุมนั้นละอองเกสรได้ถูกโอนย้ายไปได้ระยะที่ไกลกว่าเนื่องจากใช้สิ่งมีชีวิตเป็นภาหะในการเคลื่อนย้ายมีสมาการดังต่อไปนี้

$$X_i^{t+1} = X_i^t + \gamma L(\lambda)(X_i^t - g*), \qquad (1)$$

โดยให้ X_i^t : คือละอองเกสรที่ i หรือ ผลของเวคเตอร์ X_i ที่จำนวนทำซ้ำในรอบที่ t

$g*$: ผลลัพธ์ที่ดีที่สุดโดยใช้วิธีวนทำซ้ำ

$L(\lambda)$: ขั้นตอนการเดินทางเพื่อหาเกสรที่แข็งแรงโดยกำหนดให้ L คือวิธีเดินทางแบบ L'evy flight

$$L \sim \frac{\lambda\Gamma(\lambda)\sin(\frac{\pi\lambda}{2})}{\pi}\frac{1}{s^{1+\lambda}}, s \gg s_0 > 0. \qquad (2)$$

และสมการของการเคลื่อนย้ายโดยอาศัยสิ่งไม่มีชีวิตคือ

$$X_i^{t+1} = X_i^t + e(X_i^t - X_k^t), \quad (3)$$

ขั้นตอนการทำงานอัลกอริทึมผสมเกสรจากรูปที่ 4 เริ่มจากกำหนดฟังก์ชั่นวัตถุประสงค์ $f(x)$, $x = (x_1, x_2, \ldots, x_d)$ โดยให้มีจำนวน d มิติ สร้างจำนวนประชากรเริ่มต้นด้วยการกำหนดพารามิเตอร์ ขนาดของประชากร n และสุ่มค่าเริ่มต้นให้กับประชากรเพื่อหาค่าที่ดีที่สุด $g*$ กำหนดค่าเริ่มต้นสำหรับการสวิตช์ความน่าจะเป็น $p \in [0,1]$ และตรวจสอบจำนวนครบรอบการทำงาน โดยกำหนดรอบการทำงานเท่ากับ (จำนวนประชากรเริ่มต้น x จำนวนการทำซ้ำ) กำหนดรอบการทำงาน i = 1 ถึง n ถ้าค่าที่สุ่มได้ มากกว่า ค่าความน่าจะเป็น p ถ้าเป็นจริงให้ค้นหาค่าที่ดีสุดโดยใช้วิธีแบบครอบคลุมตามสมการที่ (1) แต่ถ้าไม่เป็นจริงให้ใช้วิธีค้นหาค่าที่ดีที่สุดโดยใช้วิธีแบบท้องถิ่นตามสมการที่ (2) ประเมินผลลัพธ์ที่ได้มาใหม่ ถ้าผลลัพธ์ที่ได้ดีกว่าเดิมให้นำผลลัพธ์ที่ได้มาทำเป็นผลลัพธ์อันใหม่ จนเสร็จสิ้นกระบวนการจะได้ผลลัพธ์ที่ดีที่สุด $g*$

Flower Pollination Algorithm (or simply Flower Algorithm)

Objective min *or* max $f(\mathbf{x})$, $\mathbf{x} = (x_1, x_2, ..., x_d)$
Initialize a population of n flowers/pollen gametes with random solutions
Find the best solution $\mathbf{g}_*$ *in the initial population*
Define a switch probability $p \in [0, 1]$
while (*t* <*MaxGeneration*)
  for $i = 1 : n$ *(all n flowers in the population)*
    if *rand* < p,
      Draw a (d-dimensional) step vector L which obeys a Lévy distribution
      Global pollination via $\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + L(\mathbf{g}_* - \mathbf{x}_i^t)$
    else
      Draw ϵ *from a uniform distribution in [0,1]*
      Randomly choose j and k among all the solutions
      Do local pollination via $\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \epsilon(\mathbf{x}_j^t - \mathbf{x}_k^t)$
    end if
    Evaluate new solutions
    If new solutions are better, update them in the population
  end for
    Find the current best solution $\mathbf{g}_*$
end while

รูปที่ 3 *รหัสเทียมอัลกอริทึมการผสมเกสรดอกไม้*
(ที่มา : Flower Pollination Algorithm for Global Optimization, 2012)

2.2 อัลกอริทึมค้นหาแบบนกดุเหว่า

การค้นหาแบบนกดุเหว่าพัฒนาขึ้นมาโดย Xin-She Yang ในปี 2009 เป็นวิธีการหาค่าที่เหมาะสมที่สุดโดยได้แรงบันดาลใจมาจากวิธีที่นกดุเหว่าไปฟากไข่ไว้กับรังนกสายพันธุ์อื่น รังที่นกดุเหว่าเลือกวางไข่ลงไปจะถูกสุ่มเลือกเพื่อหาคุณภาพของรัง ถ้ารังที่ถูกเลือกมีคุณภาพที่ดีก็จะยังนำมาใช้เป็นรังสำหรับไข่ในรุ่นถัดไปดังสมการที่ (4) ซึ่งมีรหัสเทียมการทำงานในรูปที่ 5 มีขั้นตอนการทำงานเริ่มจากกำหนดฟังก์ชั่นวัตถุประสงค์ $f(x)$, $x = (x_1, x_2, \ldots, x_d)$ เริ่มต้นประชากรด้วยจำนวนรัง n ตรวจสอบครบรอบการทำงานหรือไม่ ถ้าไม่ให้สุ่มผลลัพธ์ใหม่โดยใช้ L'evy Flights และประเมินคุณภาพของผลลัพธ์ด้วยฟังก์ชั่น Fitness หลังจากนั้นให้สุ่มเลือกรังที่มีอยู่ขึ้นมาถ้าคุณภาพของผลลัพธ์ใหม่ดีกว่าผลลัพธ์จากรังที่สุ่มขึ้นมาให้นำผลลัพธ์นั้นใส่ในรังที่สุ่มขึ้นมาจากนั้นทิ้งรังที่มีคุณภาพแย่และสร้างรังขึ้นมาใหม่ นำผลลัพธ์ที่มีคุณภาพที่ดีที่สุดให้มีอันดับสูงที่สุด

$$x_i^{t+1} = x_i^{(t)} + \alpha \oplus L(\lambda), \quad (4)$$

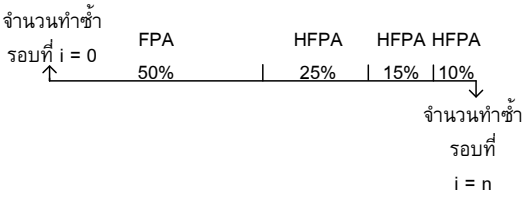
Objective function $f(\boldsymbol{x})$, $\boldsymbol{x} = (x_1, ..., x_d)^T$;
Initial a population of n host nests $\boldsymbol{x}_i$ $(i = 1, 2, ..., n)$;
while (t <MaxGeneration) or (stop criterion);
  Get a cuckoo (say i) randomly by Lévy flights;
  Evaluate its quality/fitness F_i;
  Choose a nest among n (say j) randomly;
  if $(F_i > F_j)$,
    Replace j by the new solution;
  end
  Abandon a fraction (p_a) of worse nests
    [and build new ones at new locations via Lévy flights];
  Keep the best solutions (or nests with quality solutions);
  Rank the solutions and find the current best;
end while
Postprocess results and visualisation;

รูปที่ 4 *รหัสเทียมอัลกอริทึมการค้นหาแบบนกดุเหว่า*
(ที่มา : Engineering Optimisation by Cuckoo Search, 2010)

2.3 อัลกอริทึม HFPA

จากปัญหาการกำหนดค่าเริ่มต้นในการค้นหาของอัลกอริทึม FPA ที่หากไม่เหมาะสมแล้ว จะทำให้คุณภาพของผลลัพธ์ที่ได้ไม่ดีนัก ในบทความนี้นำเสนอวิธีการพัฒนาอัลกอริทึมการผสมเกสรมาปรับปรุงให้มีลำดับขั้นของกลุ่มการค้นหาผลลัพธ์ในแต่ละรอบ (Hierarchical searching) โดยนำผลลัพธ์ที่ดีที่สุดในรอบปัจจุบัน มากำหนดเป็นค่ากลางในการหาในรอบต่อไป โดยมีระยะการค้นหาลดลงตามลำดับในแต่ละรอบ

จากรูปที่ 6 เป็นช่วงแบ่งการทำงานของ HFPA กำหนดให้รอบที่ i จนถึง n จำนวนรอบทำซ้ำสูงสุดโดยแบ่งช่วงการทำงานในรอบ 50% แรกเป็นของ FPA หลังจากนั้น HFPA จะนำค่าที่ดีที่สุดในรอบแรกของ FPA มาลดขนาดเพื่อหาค่าให้ละเอียดยิ่งขึ้น โดยการแบ่งค่าที่ดีที่สุดดังตัวอย่างในตารางที่ 1 ให้แบ่งค่าที่ดีที่สุดที่พบในช่วงค่าสูงสุดและค่าต่ำสุด ให้มีค่าตามเปอร์เซ็นต์ช่วงแบ่งการทำงาน



รูปที่ 5 *ช่วงการทำงาน HFPA*

***ตารางที่** 1 วิธีการคำนวณระยะการค้นหา*

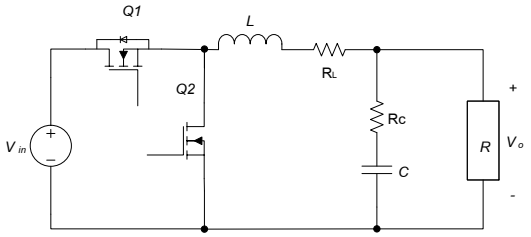
ระยะการค้นหา	ค่า [P,I,D]
25%	$Sol_{new1} = Sol_{best1} \pm (Sol_{best1}\, x\, \frac{25}{100})$ $Sol_{new2} = Sol_{best2} \pm (Sol_{best2}\, x\, \frac{25}{100})$ $Sol_{new3} = Sol_{best2} \pm (Sol_{best3}\, x\, \frac{25}{100})$
15%	$Sol_{new1} = Sol_{best1} \pm (Sol_{best1}\, x\, \frac{15}{100})$ $Sol_{new2} = Sol_{best2} \pm (Sol_{best2}\, x\, \frac{15}{100})$ $Sol_{new3} = Sol_{best3} \pm (Sol_{best3}\, x\, \frac{15}{100})$
10%	$Sol_{new1} = Sol_{best1} \pm (Sol_{best1}\, x\, \frac{10}{100})$ $Sol_{new2} = Sol_{best2} \pm (Sol_{best2}\, x\, \frac{10}{100})$ $Sol_{new3} = Sol_{best3} \pm (Sol_{best3}\, x\, \frac{10}{100})$

กำหนดให้: Sol_{new} = ผลลัพธ์ใหม่ Sol_{best} = ค่าที่ดีที่สุดในรอบปัจจุบัน

3. การออกแบบระบบ

3.1 วงจรแปลงผันบั๊ก

วงจรแปลผันบั๊กดังรูปที่ 7 เป็นวงจรที่ลดทอนแรงดันไฟตรงจากแหล่งจ่ายให้มีระดับแรงดันที่ต่ำลง โดยใช้การสวิตช์มอสเฟส Q ด้วยสัญญาณพัลส์ทำให้มอสเฟสเสมือนปิดวงจรทำให้เกิดกระแสไหลไปยังเอาต์พุตแต่เมื่อสวิตช์ปิดที่มอสเฟสกระแสยังคงไหลอย่างต่อเนื่องจากตัวเหนี่ยวนำ L สมการที่ 5 และตัวเก็บประจุ C สมการที่ 6 ซึ่งคำนวณแรงดันเอาต์พุตได้ดังสมการที่ 7 และสมการถ่ายโอนของวงจรได้ดังสมการที่ 8



***รูปที่ 6** วงจรสมมูลของวงจรแปลงผันบั๊ก*

$$\frac{di_L}{dt} = \frac{1}{L}(V_g * d - i_L * R_L - v_o) \quad (5)$$

$$\frac{dv_c}{dt} = \frac{1}{C}(i_L - i_o) \quad (6)$$

$$v_o = v_C + R_c(i_L - i_o) \quad (7)$$

โดยที่ V_g= แรงดันอินพุต, i_o= กระแสโหลด และ i_L= กระแสที่ไหลผ่านตัวเหนี่ยวนำ

$$G_{vd}(s) = \frac{v_i(sR_cC\ 1)}{\{s^2LC\left(\frac{R\ R_c}{R}\right) + s(R_cC\frac{R\ R_c}{R} + \frac{L}{R} + R_LC + \frac{R\ R_L}{R}\} \quad (8)$$

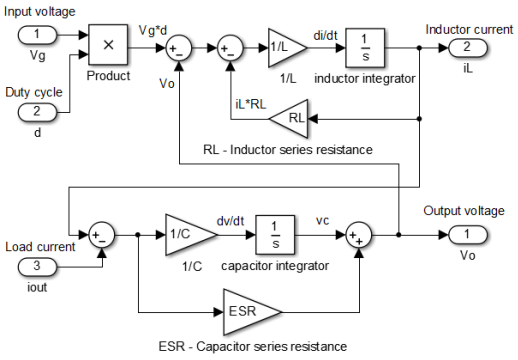
3.2 ระบบควบคุมแบบ PID

ระบบควบคุมแบบ PID อาศัยเทคนิคการควบคุมพื้นฐาน 3 แบบ ดังสมการที่ (9) ได้แก่ แบบที่ 1 การควบคุมแบบสัดส่วน (Proportional control) เป็นการนำสัญญาณควบคุมที่เกิดจากค่าความผิดพลาดของผลตอบสนองมาคูณกับค่าคงที่ Kp แบบที่ 2 การควบคุมแบบปริพันธ์ (Integral control) เป็นการควบคุมแบบใช้หลักการปริพันธ์ความคลาดเคลื่อนเพื่อกำจัดออฟเซตที่เกิดขึ้นเมื่อสัญญาณอ้างอิงเป็นค่าคงที่ แบบที่ 3 การควบคุมแบบอนุพันธ์ (derivative control) เป็นการควบคุมค่าการเปลี่ยนแปลงของค่าความคลาดเคลื่อนที่สภาวะอยู่ตัวสามารถรู้ได้ล่วงหน้าจากแนวโน้มความชันในการเปลี่ยนแปลง ณ เวลาต่างๆ ของสัญญาณค่าคลาดเคลื่อน โดยหลักการทางแคลคูลัส ความชัน ณ จุดเวลาต่างๆ ของฟังก์ชั่นใด ๆ

$$u(t) = K_p e(t) + \mathrm{K_i} \int_0^t \mathrm{e}(t)\, \mathrm{d}\tau + K_d \frac{d}{dt} e(t) \quad (9)$$

3.3 แบบจำลองวงจร

เพื่อทดสอบประสิทธิภาพของอัลกอริทึมที่ใช้ในการหาค่าสัมประสิทธิ์ของ PID ที่ใช้ในวงจรแปลงผันบั๊ก [4] ระบบได้ถูกสร้างเพื่อจำลอง (8) ในโปรแกรม Simulink ดังแสดงในรูปที่ 8 โดยกำหนดค่าภายในแบบจำลองให้ $V_{in} = 12V, V_{out} = 5v, L = 4.1\ uH, C = 376uF, R_L = 80m\Omega, R_c = 5m\Omega,\ fs = 100\ KHz$ นำแบบจำลองที่กำหนดมาเชื่อมต่อกับแบบจำลองการควบคุมแบบ PID โดยใช้การควบคุมแบบลูปปิดและกำหนดค่าแรงดันอ้างอิงให้เท่ากับ $5v$ กำหนดให้โหลด $R = 5\Omega$



***รูปที่ 7** แบบจำลองคณิตศาสตร์วงจรแปลงผันบั๊ก*

4. การทดลอง และผลลัพธ์

บทความนี้ได้เปรียบเทียบการทำงานอัลกอริทึมผสมเกสรดอกไม้กับอัลกอริทึมนกดุเหว่า [5] เพื่อหาค่าประสิทธิภาพของอัลกอริทึมที่

ใช้งาน กำหนดชุดข้อมูลสำหรับการทดลองตามตารางที่ 2 เป็นตารางสำหรับพารามิเตอร์เริ่มต้นของอัลกอริทึมผสมเกสร และตารางที่ 3 เป็นตารางสำหรับพารามิเตอร์เริ่มต้นของอัลกอริทึมค้นหาแบบนกดุเหว่าโดยใช้การทดลองทั้งหมด 5 ครั้งเพื่อหาค่าเฉลี่ยของผลลัพธ์ที่ได้ ซึ่งการทดลองใช้วิธีหาค่าความผิดพลาดโดยใช้สมการหาค่าดรรชนีสมรรถนะดังสมการที่ (10) – (13) โดยใช้วิธีจำลองการทำงานทั้ง 4 สมการ ซึ่งผลจากการทดลองสมการ IAE ให้ผลลัพธ์เหมาะสมที่สุดสำหรับนำไปทดลองการทำงานของอัลกอริทึม HFPA

Integral Absolute Error (IAE) $= \int_0^\infty |e(t)|dt$ (10)

Integral Time-weighted Absolute Error (ITAE) = $\int_0^\infty t|e(t)|dt$ (11)

Integral Squared Error (ISE) $= \int_0^\infty e(t)^2 dt$ (12)

Integral Time-weighted Squared Error ITSE = $\int_0^\infty te(t)^2 dt$ (13)

ตารางที่ 2 *พารามิเตอร์อัลกอริทึมผสมเกสรดอกไม้*

พารามิเตอร์	ค่าเริ่มต้น
ขนาดของประชากร	25
ความน่าจะเป็น P	0.8
จำนวนรอบทำซ้ำ	1500
มิติ	3
ค่าต่ำสุด [P, I, D]	[1x10^{-6}, 1x10^{-6}, 1x10^{-6}]
ค่าสูงสุด [P, I, D]	[100, 10, 0.1]

ตารางที่ 3 *พารามิเตอร์อัลกอริทึมการค้นหาแบบนกดุเหว่า*

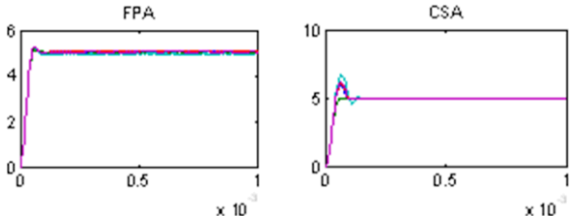
พารามิเตอร์	ค่าเริ่มต้น
ขนาดของประชากร	25
เกณฑ์ตัดสินใจทิ้งรัง	0.25
จำนวนรอบทำซ้ำ	1500
ค่าต่ำสุด [P, I, D]	[1x10^{-6}, 1x10^{-6}, 1x10^{-6}]
ค่าสูงสุด [P, I, D]	[100, 10, 0.1]

4.1 ผลการเปรียบเทียบประสิทธิภาพระหว่าง CSA กับ FPA

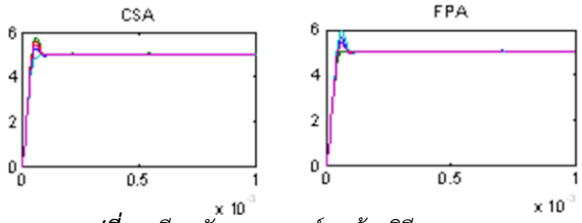
ผลการทดลองแสดงสัญญาณเอาต์พุตซึ่งเป็นค่าเฉลี่ยจากการจำลองการทำงานทั้งหมด 5 ครั้ง ซึ่งเป็นผลลัพธ์ที่ได้จากสมการหาค่าความผิดพลาดตามตารางที่ 4 โดยมีค่าสัญญาณรูปที่ 9 – 12 ซึ่งเป็นผลที่ได้ในแต่ละครั้งแสดงด้วยการใช้สีต่างๆ ดังต่อไปนี้ ครั้งที่ 1 สีน้ำเงิน, ครั้งที่ 2 สีเขียว, ครั้งที่ 3 สีแดง, ครั้งที่ 4 สีฟ้า, ครั้งที่ 5 สีม่วง

ตารางที่ 4 *ผลการจำลองการทำงาน FPA และ CSA*

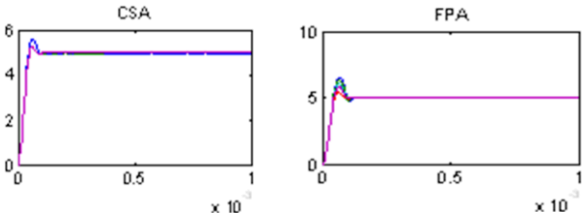
Error Criterion	Over Shoot	Ripple	Settling Time	Algorithm
$\int_0^\infty \|e(t)\|dt$	5.5 V	18.6 mV	0.14 ms	CSA
	5.6 V	18.6 mV	0.12 ms	FPA
$\int_0^\infty e(t)^2 dt$	5.4 V	18.6 mV	0.16 ms	CSA
	5.2 V	18.6 mV	0.16 ms	FPA
$\int_0^\infty t\|e(t)\|dt$	6.2 V	18.6 mV	0.12 ms	CSA
	6.04 V	18.6 mV	0.16 ms	FPA
$\int_0^\infty te(t)^2 dt$	5.4 V	18.6 mV	0.17 ms	CSA
	5.3 V	18.6 mV	0.17 ms	FPA



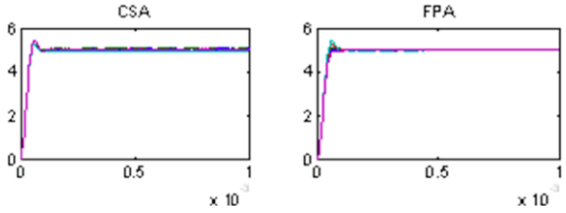
รูปที่ 8 *เทียบสัญญาณเอาต์พุตด้วยวิธี IAE*



รูปที่ 9 *เทียบสัญญาณเอาต์พุตด้วยวิธี ISE*



รูปที่ 10 *เทียบสัญญาณเอาต์พุตด้วยวิธี ITAE*

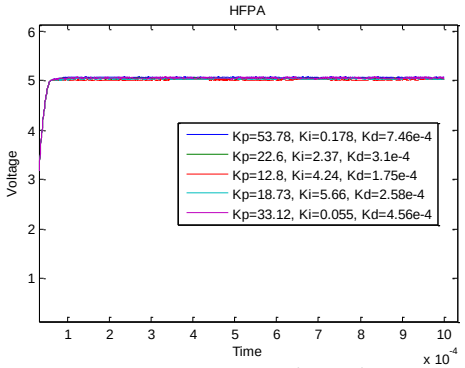


รูปที่ 11 *เทียบสัญญาณเอาต์พุตด้วยวิธี ITSE*

จากตารางที่ 4 ที่เป็นการเปรียบเทียบกันระหว่าง CSA และ FPA ซึ่งจะเห็นได้อย่างชัดเจนว่า FPA โดยวิธี IAE สามารถหาค่าผลลัพธ์ที่สามารถลู่เข้า (Settling time) ที่ 0.12 มิลลิวินาที และมีค่า Overshoot 5.6 โวลต์ ซึ่งมีค่าลู่เข้าเท่ากับ CSA โดยวิธี ITAE ที่ 0.12 มิลลิวินาที แต่ค่า Overshoot เท่ากับ 6.2 โวลต์ มากกว่า 0.6 โวลต์ ส่วน Ripple ของทั้งสองมีค่าเฉลี่ยที่เท่ากัน หากพิจารณาถึงเวลาที่ใช้ในการหา FSA สามารถหาได้โดยเฉลี่ยแต่ละรอบเท่ากับ 3.3นาที ส่วน CSA เท่ากับ 5.5 นาทีซึ่งนานกว่า

4.2ผลการทดลองด้วยอัลกอริทึม HFPA

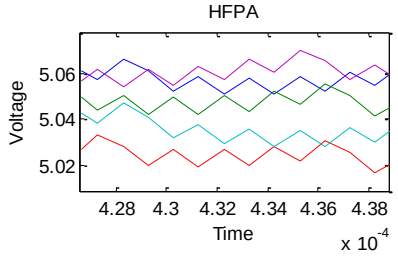
การทดลองนี้ จะใช้อัลกอริทึม HFPA ด้วยวิธีการหาค่าผิดพลาด



แบบ IAE มีผลลัพธ์ที่จำลองการทำงานทั้ง 5 ครั้ง

รูปที่ 12 *ค่า Settling Time และ Overshoot โดยวิธี HFPA*

ค่าสัมประสิทธิ์ Kp Ki, Kd ในแต่ละครั้งของการจำลองการทำงานแสดงตามสีดังต่อไปนี้ ครั้งที่ 1 สีน้ำเงิน, ครั้งที่ 2 สีเขียว, ครั้งที่ 3 สีแดง, ครั้งที่ 4 สีฟ้า, ครั้งที่ 5 สีม่วง จากรูปที่ 13 ซึ่งผลการทดลองจะเห็นได้ว่าค่า Settling Time ของวิธี HFPA มีค่าเฉลี่ยไม่เกิน 0.10 มิลลิวินาที ทำให้ได้ผลลัพธ์ที่ดีขึ้นเมื่อเทียบกับวิธี FPA ที่มีค่าเฉลี่ยมากกว่า 0.12 มิลลิวินาที รวมทั้งวิธี CSA ที่มีค่าเฉลี่ยมากว่า 0.14 มิลลิวินาที และพบว่าสัญญาณทั้งหมดที่ใช้วิธีของ HFPA ไม่เกิดการ Overshoot ของการจำลองทั้ง 5 ครั้ง



รูปที่ 14 *ค่า Ripple โดยวิธี HFPA*

ผลของค่า Ripple จากรูปที่ 14 อยู่ที่ประมาณไม่เกิน 0.02 โวลต์ คิดเป็น 0.40% ของแรงดันที่ต้องการ 5 โวลต์

5. สรุปผลงานวิจัย

ในบทความนี้นำเสนอ การเปรียบเทียบประสิทธิภาพของอัลกอริทึม CSA และ FPA ที่ใช้ในการหาค่าพารามิเตอร์ที่เหมาะสมสำหรับระบบ PID ที่ใช้ในวงจรแปลงผันบั๊ก โดยใช้วิธีการหาค่าความผิดพลาดแบบต่าง ๆ ซึ่งจากผลการทดลองเห็นได้ว่า FPA มีผลลัพธ์ที่ดีกว่า CSA ในด้านของเวลาที่ใช้ในการลู่เข้า และได้นำเสนอการพัฒนาอัลกอริทึม FPA โดยให้มีลำดับขั้นของกลุ่มการค้นหาผลลัพธ์ในแต่ละรอบ (Hierarchical searching) เป็นการลดขนาดของการหาค่าให้มีจำนวนที่ละเอียดขึ้น ซึ่งทำให้ค่าที่เหมาะสมที่สุดก่อนเข้าสู่ขั้นตอนลำดับขั้นเป็นค่าที่ดีที่สุด โดยเรียกอัลกอริทึมนี้ว่า HFPA ซึ่งจะเห็นได้ว่าผลลัพธ์ที่เป็นค่าสัมประสิทธิ์ของ PID ที่ได้หาได้จาก HFPA เป็นผลลัพธ์ที่ดีที่สุด โดยที่ไม่มีการ Overshoot ของสัญญาณ และ Settling Time มีค่าน้อยที่สุดซึ่งหมายถึงระบบเข้าสู่สถานะ Steady State ได้เร็วขึ้นนั่นเอง

เอกสารอ้างอิง

[1] Xin-She Yang, Flower pollination algorithm for global optimization, in: *Unconventional Computation and Natural Computation 2012*, Lecture Notes in Computer Science, Vol. 7445, pp. 240-249, 2012.

[2] Yang, X. S, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, United Kingdom, 2010.

[3] Yang, X.-S., and Deb, S, "Engineering Optimisation by Cuckoo Search", *Int. J. Mathematical Modelling and Numerical Optimisation*, Vol. 1, No. 4, 330–343, 2010.

[4] Sarun Soman, Sangeetha.T.S, Bindu.S, "Development of Digital Controller for Synchronous Buck Converter", *2015 International Conference on Signal Processing, Computing and Control (ISPCC)*, 2015.

[5] Muhammad Ruswandi Djlal, Imam Robadi, "Optimization of PID Controler Design for DC Motor Based on Flower Pollination Algorithm", *The 2015 International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM 2015)*, Aryaduta Hotel Medan, 2015.

จักราวุธ ศิรพัชรางกูร สำเร็จการศึกษาวิศวกรรมศาสตรบัณฑิต สาขา อิเล็กทรอนิกส์ จากมหาวิทยาลัยเทคโนโลยีมหานคร งานวิจัยที่สนใจ ได้แก่ การหาค่าที่เหมาะสมสำหรับระบบสมองกลฝังตัว

ธีรยศ เวียงทอง จบปริญญาตรีสาขาอิเล็กทรอนิกส์ (เกียรตินิยมอันดับหนึ่ง) จากสถาบันเทคโนโลยีพระจอมเกล้าลาดกระบัง ปริญญาโทสาขา Satellite Communication จาก University of Surrey และปริญญาเอกทางด้าน Digital System Design, Co-design จาก Imperial College, London เคยเป็นอาจารย์ประจำภาควิชาอิเล็กทรอนิกส์ มหาวิทยาลัย เทคโนโลยีมหานคร ปัจจุบันเป็นอาจารย์ประจำภาควิชาวิศวกรรมไฟฟ้า สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบัง มีความสนใจ มีเชี่ยวชาญทางด้านการออกแบบวงจรรวมดิจิตอล การออกแบบร่วมกันระหว่างฮาร์ดแวร์และซอฟท์แวร์ (Co-design) การประยุกต์ใช้งานเอฟพีจีเอ และระบบสมองกลฝังตัว ในระบบควบคุมและประมวลผลต่าง ๆ

สุชาดา สิทธิ์จงสถาพร จบการศึกษาวิศวกรรมศาสตรบัณฑิต (อิเล็กทรอนิกส์) เกียรตินิยมอันดับที่หนึ่ง และวิศวกรรมศาสตรดุษฎีบัณฑิต (วิศวกรรมไฟฟ้า) จากมหาวิทยาลัยเทคโนโลยีมหานคร ปัจจุบันเป็นอาจารย์ประจำภาควิชาวิศวกรรมอิเล็กทรอนิกส์ มหาวิทยาลัยเทคโนโลยีมหานครและได้รับตำแหน่งผู้ช่วยศาสตราจารย์ สาขาวิศวกรรมอิเล็กทรอนิกส์ งานวิจัยที่สนใจ ได้แก่ การประมวลสัญญาณขั้นสูง แบบปรับตัวได้ และการประมวลสัญญาณเชิงสถิติ