เอกสารอ้างอิง

ภาษาไทย

สุรพล คำสุภา, การสร้างแบบจำลองกระบวนการไม่เชิงเส้นโดยการใช้ข่ายงานนิวรัล, วิทยานิพนธ์ปริญญามหาบัณฑิตจุฬาลงกรณ์มหาวิทยาลัย, 2538

ภาษาอังกฤษ

Baht, N. V. and McAvoy, T.J., **Use of neural nets for dynamic modelling and control of chemical process systems**, *Computers chem.Eng.*, P.573-582,Vol.14, 1990.

Botcher, J. C., **On Runge Kutta Process of High Order**, *Journal of the Australian Mathematical Society*, P.179-194, Vol. 4, 1964

Brown, Martin and Harris, C., **Neurofuzzy Adaptive Modelling and Control**, Prentice Hall International Inc., NJ, 1994.

Chitra, S.P., **Neural Net Applications in Chemical Engineering**, *AI Expert*, Nov, 1992.

Coughanowr, Donald R., **Process Systems Analysis and Control**, McGraw-Hill inc., 1991

Emmanouilides, C.and Petrou, L., **Identification and Control of Anaerobic Digesters Using Adaptive, On-line Trained Neural Networks**, *Computers chem. Eng.*, P.113-143, Vol.21, 1997.

Hernandez, E. and Arkun, Y., **Neural Networks Modeling and An Extended DMC Algorithm to Control Nonlinear Systems**, *Proc. Am. Control Conf.*, P.2454-2459, 1990

Hunt, K. J. and Sbarbaro, D., **Neural Networks for Nonlinear Internal Model Control,** *IEEE Proc.-D*, P.413, Vol. 138, no. 5, 1991

Hunt, K. J., Sbarbaro, D., Zbikowski, R. and Growthorp, P. J., **Neural Networks for Control System-A Survey**, *Automatica*, P.1083-1112, Vol. 28, 1992

Jordan, M. and Rumelhart, D. E., **Internal World Models and Supervised Learning**, In Machine Learning : Proc. 8th Int. Workshop, 1991

Kawato, M., Furukawa, K. and Suzuki, R., **A hierachical neural network model for control and voluntary**, *Biological Cybernetics*, Vol. 57, pp.169-185, 1987

Khalid, M. and Omatu, S., **Neural Network Controller for a Temperature Control System,** *IEEE Control Systems*, P.58-64,Vol.12, No.3, 1992.

Khalid, M., Omatu, S. and Yusof, R., **MIMO Furnace Control with Neural Networks,** *IEEE Trans. Control Syst.Tech.*, P.238-245, Vol.4, 1993.

Kramer, M.A., and Leonard, J.A., **Diagonosis using Backpropagation Neural Networks,** *Computers chem. Eng.*, P.1323-1338, Vol. 14, 1990

Kung, S.Y., **Digital Neural Networks**, Prentice Hall International Inc., NJ, 1993.

Lee, Moonyong. and Park, Sunwon, **A New Scheme Combining Neural Feedforward Control with Model-Predictive Control**, *AIChE Journal*, Vol. 38, No. 2,1992

Leonard, J., Kramer, M.A., **Improve of The backpropagation algorithm for Training Neural Networks**, *Computers chem. Eng.*, P.337-341, Vol.14,1990.

Lindfield, G. and Penny, J., **Numerical Methods Using MATLAB**, Ellis Horwood Limited, NY, 1995

Luyben, W. L., **Process Modelling Simulation and Control for Chemical Engineerings,** McGraw-Hill, Inc., 1989

McAvoy, T. J., Wang, N. S., Naidu, S.,Bath, N. and Simmons, M., **Interpreting biosensor data with Backpropagation,** *Proc. IEEE Int. Joint Conf. on Neural Networks*, Vol. 1, P.593-605, 1989.

McClelland, J., Rumelhart, D. and PDP Research Group, **Parallel Distributed Processing**, Vol. 1, Cambridge, MA : MIT Press, 1986

McCulloch, W. and Pitts, W., **A Logical Calculus of The Ideas Immanent in Neural Activity**, *Bulletin of Mathematical Biophysics*, P.115-133, Vol. 7, 1943

Merson, R. H., **An Operational Method for The Study of Integration Process**, *Proc. Conf. on Data Processing and Automatic Computing Machines*, Weapons Research Establishment Salisbury, South Australia, 1957

Minsky, M. and Papret, S., **Perceptrons**, Cambridge : MIT Press, 1969

Nahas, E.P., Henson, M.A. and Seborg, D.E., **Nonlinear internal model control strategy for Neural Networks Models**, *Computers chem. Eng.*, P.1039-1057, Vol.16, 1992.

Narendra, K. S. and Parthasarathy, K., **Gradient Methods for The Optimization of Dynamical Systems Containing Neural Networks**, *IEEE Trancactions on Neural Networks*, P.252-262, Vol. 2(2)

Narendra, K. S. and Parthasarathy, K., **Identification and Control of Dynamical Systems Using Neural Networks**, *IEEE Trans Neural Networks*, Vol. 37(1), 1990

Psaltis, D., Sideris, A. and Yamamura, A., **A Multilayered Neural Networks Controller**, *IEEE Control Syst. Mag.*, P.44-48, Vol. 10, no. 3, Apr., 1989

RayChaudhuri, T., Hamey, Leonard G.C. and Bell, Rodney D., **From Conventional Control to Autonomous Intelligent Methods**, *IEEE Control Systems*, P.78-84,Vol.16, No.5, 1996.

Rosenblatt, F., **The Peraptron : A Probabilistic Model for Information Storage and Organization in The Brain**, *Psy Chology Review*, P.92-99, Vol. 15, 1958

Rumelhart, D. E., Hinton, G. E. and Williams, R. J., **Learning Internal Representation by Error Propagation. In Parallel Distributed Processing,** MIT Press, Cambridge, MA., 1986

Seborg, D.E., Egdggar, T.F. and Mellichamp, D.A., **Process Dynamic and Control**, Wiley, New York, 1989

Shepanski, J. F., **Fast Learning in Artificial Neural System : A Multilayer Perceptron Traning Using Optimal Estimation,** *IEEE Second Int. Neural Netwarks, San Diego*, P.-465-472, 1988

Spieker, A., Najim, K., Chtourou, M. and Thibault, J., **Neural networks for thermal processes,** *J. Porc. Cont.*, P.223-239, Vol. 3, No. 4, 1993

Ydstie, B.E., **Forcasting and control using adaptive connectionist networks,** *Computers chem. Eng.*, P.583-599, Vol.14, 1990.

Wasserman, Philip D., **Advanced Methods in Neural Computing,** Van Nostrand Reinhold, New York, 1993

Werbos, P. J., **Beyond Regression : New Tools for Prediction and Analysis in The Behaviord Sciences**, PhD. Thesis, Harvard University Committee in Applied Mathematics, 1974

Willis, M.J., Massimo, C.D., Montague, G.A., Tham, M.T. and Morris, A.J., **Artificial neural networks in process engineering**, *IEE Proceedings-d*, Vol. 138, No. 3, 1991

ภาคผนวก ก.

โปรแกรมแม็ทแลบ (Matlab)

โปรแกรมแม็ทแลบ (Matlab) เป็นโปรแกรมที่ใช้การคำนวณทางคณิตศาสตร์ขั้นสูง และใช้แก้ปัญหาที่เป็นแมทริกซ์ และเวคเตอร์ การเขียนโปรแกรมด้วยโปรแกรมแม็ทแลบง่ายกว่าโปรแกรมภาษาอื่น ๆ เนื่องจากโปรแกรมแม็ทแลบได้รวมคุณสมบัติที่ดีของโปรแกรมภาษาอื่นเข้ามาเช่นการที่ไม่ต้องประกาศชื่อและชนิดของตัวแปรที่เหมือนกับโปรแกรมภาษาเบสิก และการเขียนโปรแกรมที่เป็นโครงสร้างที่เข้าใจง่าย (procedure language) ที่ใช้ในโปรแกรมภาษาซี และภาษาปาสคาล

ก.1 การพัฒนาของโปรแกรมแม็ทแลบ

โปรแกรมแม็ทแลบ (MATLAB) เป็นชื่อย่อของ "MATrixLABoratory" ซึ่งได้ถูกพัฒนาขึ้นครั้งแรกที่มหาวิทยาลัยนิวแม็กซิโก และมหาวิทยาลัยสแตนฟอร์ดในปลายทศวรรษที่ 1970 เพื่อใช้สอนทฤษฎีเกี่ยวกับ แมทริกซ์ (matrix), ฟังก์ชันพีชคณิตแบบเชิงเส้น (linear algebra) และการวิเคราะห์เชิงตัวเลข (numerical analysis) โปรแกรมแม็ทแลบถูกเขียนขึ้นเป็นครั้งแรกโดยใช้ภาษาฟอร์แทรน (fortran)โดยคลี่ฟ โมลเลอร์ (Clkeve Moller) จากนั้นก็ได้รับการพัฒนาจากโปรแกรมเมอร์อีกหลายท่าน ในโครงการ "LINPACK and EISPACK" ปัจจุบันโปรแกรมแม็ทแลบถูกเขียนขึ้นโดยใช้ภาษาซี (C language)

ก.2 ความสามารถของโปรแกรมแม็ทแลบ

โปรแกรมแม็ทแลบเป็นโปรแกรมที่สามารถโต้ตอบกับผู้ใช้งานแบบทันทีทันใดและได้มีการพัฒนามาอย่างต่อเนื่อง ในการศึกษาโปรแกรมแม็ทแลบจะถูกใช้ไปในการวิจัยและการสอนทางคณิตศาสตร์ ในทางอุตสาหกรรมโปรแกรมแม็ทแลบจะใช้ในการวิจัยทางวิศวกรรม และการแก้ปัญหาทางคณิตศาสตร์เช่น การควบคุมกระบวนการแบบอัตโนมัติ และการวิจัยสัญญาณของกระบวน การทำงานของโปรแกรมแม็ทแลบจะเป็นฟังก์ชันของคำสั่งที่อยู่ในรูปของโปรแกรม "M-file" ซึ่งคำสั่งเหล่านี้สามารถนำมาใช้แก้ปัญหาทางคณิตศาสตร์คือ

ก. การคำนวณเกี่ยวกับแมทริกซ์

โปรแกรมแม็ทแลบสามารถทำแมทริกซ์ทรานส์โพส, การคูณแมทริกซ์, การหาดีเทอร์มิแนนท์, การทำอินเวอร์สแมทริกซ์, ค่าไอเกน, การแก้สมการเชิงเส้น, และการประมาณค่าพารามิเตอร์

ข. การคำนวณโพลีโนเมียล

โปรแกรมแม็ทแลบสามารถใช้ในการคำนวณเกี่ยวกับโพลีโนเมียล เช่น การหารากของโพลีโนเมียล การหาคอนโวลูชัน (convolution) และดีคอนโวลูชัน (deconvolution) การหารโพลีโนเมียล และการหาสมการถดถอยแบบโพลีโนเมียล

ค. การจัดการเกี่ยวกับเวคเตอร์ และการวิเคราะห์ข้อมูล

โปรแกรมแม็ทแลบสามารถนำมาคำนวณผลรวมแบบเวคเตอร์, การหาค่าเฉลี่ย, การหาค่าเบี่ยงเบนมาตราฐาน, ค่าโคแวเรียนซ์, และการหาค่าสูงสุดต่ำสุดของข้อมูล

ง. การจัดการเกี่ยวกับการแสดงผลกราฟ

โปรแกรมแม็ทแลบมีการแสดงผลเป็นกราฟให้เลือกได้ 7 แบบคือ

- การพล็อตกราฟ x-y บนสเกลเส้นตรง
- การพล็อตกราฟ x-y บนสเกลล็อก-ล็อก
- การพล็อตกราฟ x-y บนสเกลกึ่งล็อกบนแกน x
- การพล็อตกราฟ x-y บนสเกลกึ่งล็อกบนแกน y
- การพล็อตกราฟแบบโพลาร์
- การพล็อตกราฟแบบตะแกรง 3 มิติ
- การพล็อตกราฟแบบคอนทัวร์

โปรแกรมแม็ทแลบสามารถใช้งานร่วมกับโปรแกรมภาษาอื่น ๆ เช่น โปรแกรมภาษาซี และโปรแกรมภาษาฟอร์แทรน นอกจากนี้โปรแกรมแม็ทแลบ มี Toolboxes ที่ประกอบไปด้วยฟังก์ชันต่าง ๆ ซึ่งใช้ในสาขาต่าง ๆ เช่น

- Control System Toolbox
- SIMULINK
- Neural Network Toolbox
- Fuzzy Logic Toolbox
- Image Processing Toolbox
- Model Predictive Control Toolbox
- Nonlinear Control Design Toolbox

- Optimization Toolbox
- Signal Processing Toolbox
- Statistics Toolbox
- System Identification Toolbox
- Spline Toolbox
- Robust Control Toolbox
- Mu-Analysis and Synthesis Toolbox

ก.3 การเขียนโปรแกรมด้วยคำสั่งในโปรแกรมแม็ทแลบ

โปรแกรมแม็ทแลบ จะรับคำสั่งทีละ 1 บรรทัด (command line) ในพื้นที่หน้าต่างของโปรแกรม หลังจากรับคำสั่งแล้วโปรแกรมแม็ทแลบจะทำการประมวลผลและแสดงผลทางหน้าต่างของการทำงาน และสามารถสร้างไฟล์ที่ประกอบไปด้วยชุดของคำสั่งที่เหมือนกับการสร้างไฟล์ Autoexec.bat ใน dos โดยไฟล์ของคำสั่งจะเก็บอยู่ในรูป "ชื่อไฟล์.M" คือมีนามสกุลของไฟล์เป็น "เอ็ม (.M)" หรือเรียกว่าเอ็มไฟล์ การประมวลผลจะทำการประมวลผลทีละคำสั่งตามลำดับก่อนหลัง การเขียนเอ็มไฟล์ มี 2 รูปแบบคือ สคริปไฟล์ และฟังก์ชันไฟล์

ก.3.1 สคริปไฟล์

สคริปไฟล์เป็นไฟล์ซึ่งเป็นลำดับของคำสั่ง คำอธิบายในสคริปไฟล์สามารถเขียนได้โดยใช้ "%" นำหน้าข้อความที่อธิบายโดยคำอธิบายนี้จะไม่มีผลต่อการทำงานของสคริปไฟล์ โอปะเรเตอร์ในสคริปไฟล์ได้แก่ ยกกำลัง (^), คูณ (*), หาร (/), บวก (+), และ ลบ (-) ตัวอย่างการใช้โอปะเรเตอร์เช่น ไฟล์ OPERATOR.M

```
% Matrix calculation for two matrices A and B
A=[1 2 3; 4 5 6; 7 8 9];
B=[5 -6 -9; 1 1 0;24 1 0];
% Addition result assigned to C
D=A*B;
disp(D);
% Division result assigned to E
E=A\B;
disp(E);
```

นอกจากโอปะเรเตอร์แล้ว โปรแกรมแม็ทแล็บรองรับการทำงานที่วนซ้ำ ๆ หลายครั้ง ซึ่งได้แก่คำสั่ง for loop และคำสั่ง while loop นอกจากนี้โปรแกรมแม็ทแล็บมีคำสั่งเงื่อนไข if ซึ่งคำสั่งทั้ง 3 เป็นคำสั่งที่โปรแกรมภาษาทุกภาษาต้องมี โดยมีการใช้คำสั่งเหล่านี้ ดังนี้คือ

1. คำสั่ง for loop จะใช้เมื่อต้องการทำการคำนวณคำสั่งเดิมวนซ้ำหลาย ๆ ครั้งโดยอยู่ในรูปแบบดังนี้คือ

```
for loopvariable =  loopexpression
statements
end
```

ตัวอย่างของคำสั่ง for loop คือ

```
for i=1:n
   for j=1:m
   C(i , j)=A(i , j)+cos((i+j)*pi/(n+m))*B(i , j);
  end
end
```

2. คำสั่ง while loop จะใช้เมื่อต้องการคำนวณคำสั่งเดิมวนซ้ำโดยมีเงื่อนไขการสิ้นสุดการวนซ้ำโดยอยู่ในรูปแบบดังนี้คือ

```
while while_expression
 statements
end
```

ใน while_expression โอปะเรเตอร์ที่ใช้ในการเปรียบเทียบคือ เท่ากับ (= =), น้อยกว่าหรือเท่ากับ (< =), มากกว่าหรือเท่ากับ (> =), ไม่เท่ากับ (~ =), น้อยกว่า (<), และมากกว่า (>)

ตัวอย่างของคำสั่ง while loop คือ

```
dif = 1;
while dif>0.005
   x1=x2-cos(x2)/(1+x2);
   dif=abs(x2-x1);
end
```

3. คำสั่งเงื่อนไข (if statement) มีรูปแบบดังนี้คือ

```
if if_expression
  statements
elseif if_expression
```

```
  statements
elseif if_expression
  statements
. . .
. . .
else
  statements
end
```

ตัวอย่างของคำสั่ง if คือ

```
for k = 1:n
  for p =1:m
   if k = = p
        z (k,p) =1;
        tota l = total + z (k,p);
    else if k < p
        z (k,p) = -1;
        total = total + z (k,p)
     else
        z(k,p) = 0;
     end
    end
end
if (x ~= 0)&(x < y)
    b = sqrt(y - x)/x;
     disp(b);
  end
```

การเรียกใช้สคริปไฟล์สามารถเรียกใช้ได้โดยพิมพ์ชื่อเอ็มไฟล์ที่หน้าต่างของโปรแกรมแม็ทแลบ หรืออาจเขียนเรียกใช้ในสคริปไฟล์ หรือฟังก์ชันไฟล์อื่น ๆ

ก.3.2 ฟังก์ชันไฟล์

ฟังก์ชันไฟล์คือ ไฟล์ที่เริ่มต้นบรรทัดแรกด้วยคำว่า "function" ตัวอย่างเช่น

```
function y = mean(x)
% MEAN Average or mean value.
% For vectors,MEAN(X) RETURNS THE MEAN VALUE.
% For matrices, MEAN(X) IS A ROW VECTOR
% containing the mean value of each column.
[m,n]=size(x);
if m==l;
m=n;
end
y=sum(x)/m;
```

จากตัวอย่างข้างต้นจะเห็นว่าเอ็มไฟล์ทั้งสองแตกต่างกันที่ แบบสคริปไฟล์ไม่มีการส่งค่าของตัวแปรทั้งเข้าและออก ดังนั้นค่าต่าง ๆ ที่เกิดขึ้นในเอ็มไฟล์จะไม่มีผลกระทบต่อเอ็มไฟล์อื่น ๆ ในกรณีที่มีชื่อตัวแปรเหมือนกัน ส่วนแบบฟังก์ชันไฟล์มีการส่งค่าเข้าและออกเหมือนกับการเขียนเป็นฟังก์ชันโดยทั่ว ๆ ไปในภาษาการเขียนโปรแกรมแบบอื่น ๆ เช่น ภาษาซี และภาษาปาสคาล โดยสัญลักษณ์" % " ในฟังก์ชันไฟล์คือคำอธิบายการคำนวณและการเขียนโปรแกรมในฟังก์ชันไฟล์ เหมือนกับที่ใช้ในสคริปไฟล์

ภาคผนวก ข.

การแก้สมการคณิตศาสตร์ในกระบวนการวิศวกรรมเคมี

โดยใช้วิธีเชิงตัวเลข (Numerical method)

ในกระบวนการวิศวกรรมเคมีหน่วยปฏิบัติการ (unit operation) ต่าง ๆ เช่น ถังผสม, ถังปฏิกรณ์ต่าง ๆ , หอกลั่น ที่มีการเปลี่ยนแปลงทางพลศาสตร์ของมวล (dynamic mass) และสมดุลพลังงานของระบบ สามารถแทนได้ด้วยแบบจำลองทางคณิตศาสตร์ (mathematical model) ที่อยู่ในรูปสมการอนุพันธ์แบบธรรมดา (ordinary differential equation) ซึ่งที่มีรูปแบบสมการทั่วไปสำหรับสมการอนุพันธ์อันดับที่ n ใด ๆ เป็น

$$a_n \frac{d^n y}{dt^n} + a_{n-1} \frac{d^{n-1} y}{dt^{n-1}} + \ldots + a_1 \frac{dy}{dt} + a_0 y = f(t) \qquad (ข.1)$$

โดยที่ค่า a_0 , a_1 , a_2 ,..., a_n เป็นค่าคงที่

$f(t)$ เป็นค่าของอินพุทหรือตัวรบกวนที่ใส่ลงไปในระบบ (forcing function or distribance)

$y(t)$ เป็นค่าที่ตอบสนองออกมาจากระบบที่เวลา t

โดยทั่วไปจะสนใจระบบที่มีสมการอนุพันธ์อันดับที่ n =1 และ n=2

สมการอนุพันธ์อันดับที่ 1 (first order differential equation) มีรูปแบบสมการทั่วไปเป็น

$$a_1 \frac{dy}{dt} + a_0 y = f(t) \qquad (ข.2)$$

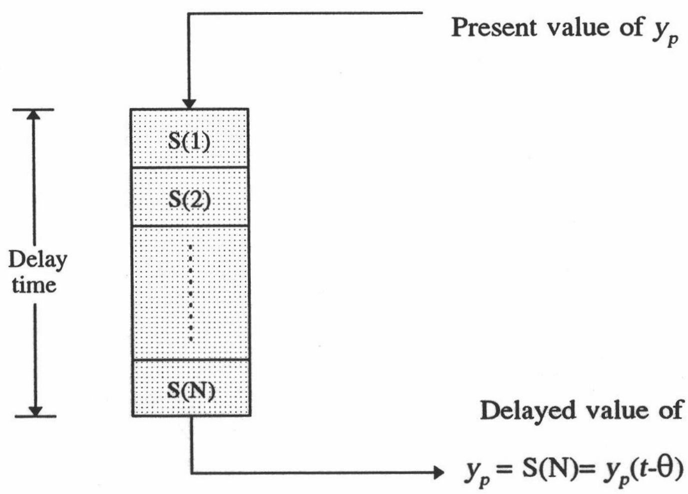
สมการอนุพันธ์อันดับที่ 2 (second order differential equation) มีรูปแบบสมการทั่วไปเป็น

$$a_2 \frac{d^2 y}{dt^2} + a_1 \frac{dy}{dt} + a_0 dy = f(t) \qquad (ข.3)$$

ข.1 การหน่วงเวลา (time delay)

กระบวนการจริงโดยทั่วไปในทางวิศวกรรมเคมีจะมีการหน่วงเวลาเสมอ เวลาในการหน่วงคือเวลาที่ป้อนสัญญาณเข้าสู่กระบวนการจนให้ผลลัพธ์, y_p ออกมา ดังนั้นแบบจำลองของกระบวนการที่สร้างขึ้นจึงต้องทำการหน่วงเวลาให้เหมือนกับกระบวนการจริง สำหรับในการซิมมูเลทสามารถแก้ปัญหาโดย

1. ใส่ฟังก์ชันการหน่วงเวลารวมเข้าไปในสมการอนุพันธ์ของระบบและ แก้สมการหาผลลัพธ์ออกมา



รูปที่ ข.1 การเก็บผลลัพธ์ที่ได้จากกระบวนการไว้ในหน่วยความจำ

2. เก็บสัญญาณผลลัพธ์ที่ได้เรียงไว้ในหน่วยความจำ จำนวนของผลลัพธ์ที่ต้องเก็บเรียงไว้ขึ้นกับค่าการหน่วงเวลาของกระบวนการ, θ และเวลาที่ใช้ในหนึ่งคาบ, dt ซึ่งมีความสัมพันธ์กันดังนี้

$$\text{จำนวนตำแหน่งที่ต้องสำรอง}, N = \theta / dt \qquad \text{(ข.4)}$$

การเก็บผลลัพธ์ที่ได้จากข่ายงานนิวรัลไว้ในหน่วยความจำ แสดงได้ดังรูปที่ ข.1

ข.2 การแก้สมการอนุพันธ์

สมการอนุพันธ์แบบธรรมดา (ODE) ในรูปแบบจำลองทางคณิตศาสตร์ สามารถหาคำตอบได้ด้วยวิธีเชิงตัวเลข (numerical method) วิธีที่นิยมใช้กันอย่างกว้างขวางคือวิธีของรังกัตตาอันดับ 4 ซึ่งสามารถแก้สมการอนุพันธ์แบบไม่เชิงเส้นได้เป็นอย่างดี

ข.2.1 สมการอนุพันธ์อันดับ 1

$$a_1 \frac{dy}{dt} + a_0 y = f(t) \qquad \text{(ข.5)}$$

จัดสมการให้อยู่ในรูป

$$\frac{dy}{dt} = f(t, y) \qquad \text{(ข.6)}$$

กำหนดสภาวะเริ่มต้น (initial condition): $t = t_0$, $y = y_0$

อัลกอริธึมของรังกัตตาอันดับที่ 4 เพื่อแก้สมการหา $y(t)$ คือ

$$k_1 = f(y_0, t_0)dt \qquad \text{(ข.7)}$$

$$k_2 = f(y_0 + \frac{k_1}{2}, t_0 + \frac{dt}{2})dt \qquad \text{(ข.8)}$$

$$k_3 = f(y_0 + \frac{k_2}{2}, t_0 + \frac{dt}{2})dt \qquad \text{(ข.9)}$$

$$k_4 = f(y_0 + k_3, t_0 + dt)dt \qquad (ข.10)$$

$$y_1 = y_0 + (k_1 + 2k_2 + 2k_3 + k_4)/6 \qquad (ข.11)$$

$$t_1 = t_0 + dt \qquad (ข.12)$$

dt คือช่วงเวลาที่เพิ่มขึ้นของตัวแปรอิสระ t

ข.2.2 สมการอนุพันธ์อันดับ 2

$$a_2 \frac{d^2 y}{dt^2} + a_1 \frac{dy}{dt} + a_0 dy = f(t) \qquad (ข.13)$$

แปลงสมการอนุพันธ์อันดับ 2 ให้อยู่ในรูปสมการอนุพันธ์อันดับ 1 สองสมการ โดยกำหนดให้ $y_1 = y$ และ $y_2 = dy_1/dt = dy/dt$ ดังนั้น

$$\frac{dy_1}{dt} = y_2 \qquad (ข.14)$$

$$\frac{dy_2}{dt} = -\frac{a_0}{a_2} y_1 - \frac{a_1}{a_2} y_2 + \frac{1}{a_2} f(t) \qquad (ข.15)$$

กรณีที่ตัวแปรตามเป็น y_1, y_2 และตัวแปรอิสระเป็น t จัดสมการให้อยู่ในรูป

$$\frac{dy_1}{dt} = f(y_1, y_2, t) \qquad (ข.16)$$

$$\frac{dy_2}{dt} = f(y_1, y_2, t) \qquad (ข.17)$$

$$k_1 = f_1(y_{10}, y_{20}, t_0)dt \qquad (ข.18)$$

$$l_1 = f_2(y_{10}, y_{20}, t_0)dt \qquad (ข.19)$$

$$k_2 = f_1(y_{10} + \frac{k_1}{2}, y_{20} + \frac{l_1}{2}, t_0 + \frac{dt}{2})dt \qquad (ข.20)$$

$$l_2 = f_2(y_{10} + \frac{k_1}{2}, y_{20} + \frac{l_1}{2}, t_0 + \frac{dt}{2})dt \qquad (ข.21)$$

$$k_3 = f_1(y_{10} + \frac{k_2}{2}, y_{20} + \frac{l_2}{2}, t_0 + \frac{dt}{2})dt \qquad (ข.22)$$

$$l_3 = f_2(y_{10} + \frac{k_2}{2}, y_{20} + \frac{l_2}{2}, t_0 + \frac{dt}{2})dt \quad (ข.23)$$

$$k_4 = f_1(y_{10} + k_3, y_{20} + l_3, t_0 + dt)dt \quad (ข.24)$$

$$l_4 = f_2(y_{10} + k_3, y_{20} + l_3, t_0 + dt)dt \quad (ข.25)$$

$$y_{11} = y_{10} + (k_1 + 2k_2 + 2k_3 + k_4)/6 \quad (ข.26)$$

$$y_{21} = y_{20} + (l_1 + 2l_2 + 2l_3 + l_4)/6 \quad (ข.27)$$

$$t_1 = t_0 + dt \quad (ข.28)$$

ข.3 การแก้สมการอนุพันธ์ โดยใช้โปรแกรมแม็ทแลบ

โปรแกรมแม็ทแลบมีฟังก์ชัน (ODE23.M และ ODE45.M) ที่ใช้ในการแก้สมการอนุพันธ์แบบธรรมดา (ODE) โดยใช้วิธีรังกัตตา ฟังก์ชัน ODE45 .M จะมีความแม่นยำมากกว่า ODE23.M เนื่องจากฟังก์ชัน ODE45.M มีอันดับสูงกว่า ODE23.M แต่ฟังก์ชันทั้งคู่นี้ไม่สามารถนำไปใช้ในงานวิจัยนี้ได้เนื่องจากฟังก์ชันทั้งสองนี้มีการกำหนดเวลาในหนึ่งคาบ (sampling time) ไม่เท่ากันในแต่ละสเต็พของการซิมมูเลท ในช่วงใดของสมการอนุพันธ์ที่การเปลี่ยนแปลงน้อยหรือมีความซับซ้อนน้อยมันจะเพิ่มเวลาในหนึ่งคาบมากขึ้น ในทางกลับกันถ้าในช่วงที่สมการอนุพันธ์มีการเปลี่ยนแปลงมากมันจะลดเวลาในหนึ่งคาบลง

ข.3.1 การใช้ฟังก์ชัน ODE23.M และ ODE45.M

ตัวอย่างของการใช้ฟังก์ชัน ODE23.M และ ODE45.M จะใช้เหมือนกันทุกประการเพียงแต่มีความแม่นยำที่ต่างกันเท่านั้น ตัวอย่างของสมการอนุพันธ์ที่ใช้ในการแก้คือ

$$\frac{dy}{dt} = -y \quad (ข.29)$$

สมการอนุพันธ์นี้สามารถเขียนอยู่ในรูปฟังก์ชัน, f500.M ได้ดังนี้

```
function yprime=f500(t,y)
yprime = -y;
```

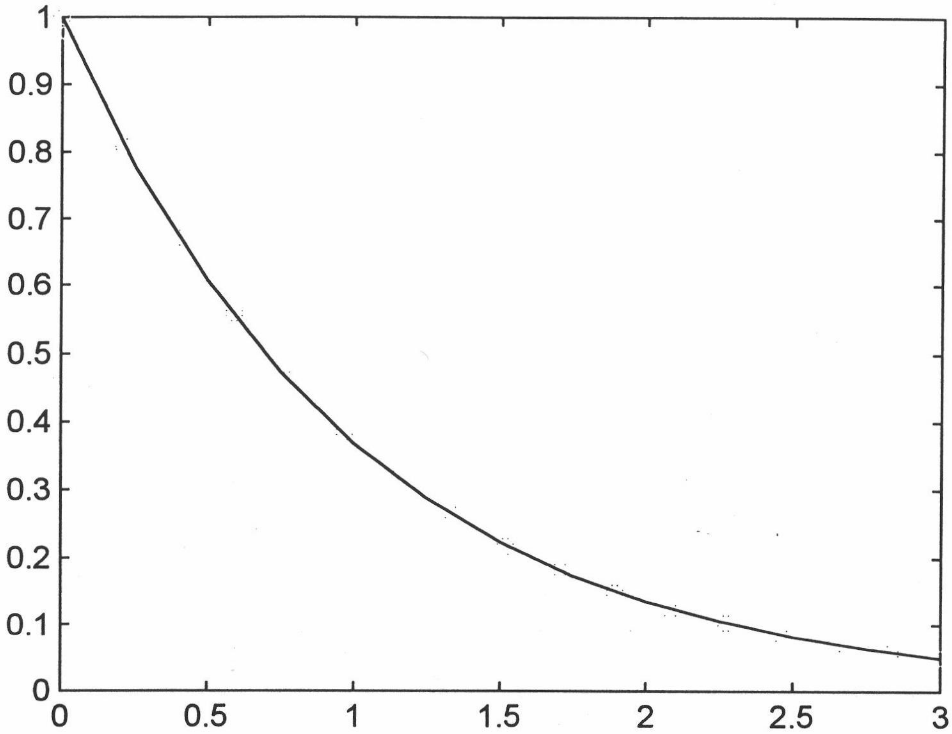
ตัวอย่างการเรียกใช้ฟังก์ชัน ODE45.M เพื่อแก้สมการอนุพันธ์ (ข.29) ได้แก่

```
%run time
simtime = 3;
%accuracy value
acc = 0.001;
%initial value
initx=1;
[t x]=ode45('f500',0,simtime,initx,acc);
plot(t1,x1);
title('ODE45')
xlabel('time');
ylabel('x ');
```

ในการเรียกใช้ฟังก์ชัน ODE45.M ตัวกำหนดเวลาในการซิมมูเลท (simtime), ค่าความแม่นยำ (acc), และค่าเริ่มต้น (initx) และผลของการซิมมูเลทแสดงในรูปที่ ข.2

ข.3.2 การแก้สมการอนุพันธ์โดยวิธีรังกัตตาที่พัฒนาขึ้นเอง

จากเหตุผลที่กล่าวมาแล้วที่ไม่สามารถใช้ฟังก์ชัน ODE23.M และ ODE45.M ในการแก้สมการอนุพันธ์ในงานวิจัยนี้ การแก้สมการอนุพันธ์ในงานวิจัยนี้จะพัฒนาต่อจาก Lindfield และ Penny (1995) Lindfield ได้แก้สมการอนุพันธ์โดยวิธีรังกัตตา อันดับที่ 4, 5 และ 6 โดยจากรูปสมการทั่วไปของวิธีรังกัตตาอันดับที่ 4



รูปที่ ข.2 แสดงผลของการซิมมูเลทสมการอนุพันธ์ f500.Mโดยใช้ฟังก์ชัน ODE45.M

$$k_1 = f(t_n, y_n)dt \tag{ข.40}$$

$$k_2 = f(t_n + \frac{dt}{2}, y_n + \frac{k_1}{2})dt \tag{ข.41}$$

$$k_3 = f(t_n + \frac{dt}{2}, y_n + \frac{k_2}{2})dt \tag{ข.42}$$

$$k_4 = f(t_n + dt, y_n + k_3)dt \tag{ข.43}$$

$$y_{n+1} = y_n + (k_1 + 2k_2 + 2k_3 + k_4)/6 \tag{ข.44}$$

Lindfield จัดสมการ (ข.40) - (ข.44) ให้อยู่ในรูปทั่วไป สำหรับแต่ละสเต็พ n = 0, 1,

2,...

$$k_1 = f(t_n, y_n) \quad (ข.45)$$

$$k_i = f(t_n + d_i dt, y_n + \sum_{j=1}^{i-1} c_{ij} k_j) \quad (ข.46)$$

$$y_{n+1} = y_n + \sum_{j=1}^{p} b_j k_j \quad (ข.47)$$

โดยที่ p คือจำนวนอันดับ (order)

สมการ (ข.45) - (ข.47) เป็นรูปแบบทั่วไป สำหรับวิธีรังกัตตาอันดับ 4, 5 และ 6 จะมีค่าของ b, c และ d ที่แตกต่างกันคือ

1. วิธีรังกัตตาอันดับ 4 ซึ่งเป็นวิธีมาตราฐานที่ใช้กันทั่วไปโดยมีค่า b, c, , และ d คือ

b=[1/6 1/3 1/3 1/6];

c=[0 0 0 0; 0.5 0 0 0; 0 0.5 0 0;0 0 1 0];

d=[0 0.5 0.5 1];

2. วิธีรังกัตตาอันดับ 5 ซึ่งเสนอโดย Merson (1957) โดยมีค่า b, c, , และ d คือ

b=[1/6 0 0 2/3 1/6];

c=[0 0 0 0 0;1/3 0 0 0 0;1/6 1/6 0 0 0;1/8 0 3/8 0 0;1/2 0 -3/2 2 0];

d=[0 1/3 1/3 1/2 1];

3. วิธีรังกัตตาอันดับ 6 ซึ่งเสนอโดย Butcher (1964) โดยมีค่า b, c และ d คือ

b=[0.07777777778 0 0.355555556 0.13333333 0.355555556 0.0777777778];

c(1:4,:)=[0 0 0 0 0 0;0.25 0 0 0 0 0;0.125 0.125 0 0 0 0;0 -0.5 1 0 0 0];

c(5,:)=[0.1875 0 0 0.5625 0 0];

c(6,:)=[-0.4285714 0.2857143 1.714286 -1.714286 1.1428571 0];

```
d=[0 0.25 0.25 0.5 0.75 1];
```

Lindfield เขียนฟังก์ชันสำหรับแก้สมการอนุพันธ์ 1 สมการ, RKGEN.M ซึ่งมีรายละเอียดดังนี้

```
function[tvals,yvals]=rkgen(f,start,finish,startval,step,order)
%Solves dy/dt=f(t,y).start, finish are initial, final values of t
%startval is initial value of y, step is the increment in t
%method (1, 2 or 3) selects Order=4->Classical RK, Order=5->Merson RK
%OR Order=5->Butcher RK.
b=[ ] ; c=[ ]; d=[ ];
if order < 4 | order > 6
  disp('Method number unknown so using order=4');
  order=4;
end;
if order==4
   b=[1/6 1/3 1/3 1/6]; d=[0 0.5 0.5 1];
   c=[0 0 0 0; 0.5 0 0 0; 0 0.5 0 0;0 0 1 0];
   disp('Order=4 ,Classical method selected');
elseif  order==5
   b=[1/6 0 0 2/3 1/6];d=[0 1/3 1/3 1/2 1];
   c=[0 0 0 0 0;1/3 0 0 0 0;1/6 1/6 0 0 0;1/8 0 3/8 0 0;...
      1/2 0 -3/2 2 0];
   disp('Order=5,Merson method selected');
else
   b=[0.07777777778 0                0.355555556 0.13333333...
      0.355555556    0.0777777778];
   d=[0 0.25 0.25 0.5 0.75 1];
   c(1:4,:)=[0 0 0 0 0 0;0.25 0 0 0 0 0;0.125 0.125 0 0 0 0;...
            0 -0.5 1 0 0 0];
```

```
    c(5,:)=[0.1875 0 0 0.5625 0 0];
    c(6,:)=[-0.4285714 0.2857143 1.714286 -1.714286 1.1428571 0];
    disp('Order=6,Bucher method selected');
end;
steps=(finish-start)/step+1;
y=startval; t=start;
yvals=startval; tvals=start;
for j=2:steps
   k(1)=step*feval(f,t,y);
   for i=2:order
      k(i)=step*feval(f, t+step*d(i), y+c(i,1:i-1)*k(1:i-1)');
   end;
   y1=y+b*k'; t1=t+step;
   %collect values together for output
   tvals=[tvals, t1]; yvals=[yvals, y1];
   t=t1; y=y1;
end;
```

ข้อจำกัดของวิธีของ Lindfield คือสามารถแก้สมการอนุพันธ์เพียง 1 สมการเท่านั้น ในงานวิจัยนี้กระบวนการที่ใช้ในการทดลองมีสมการอนุพันธ์มากกว่า 1 สมการ ดังนั้นจึงจำเป็นต้องพัฒนาวิธีของ Lindfield โดยฟังก์ชันที่พัฒนานี้จะใช้วิธีรังกัตตา อันดับ 4 เนื่องจากในการแก้สมการอนุพันธ์หลายสมการจำเป็นต้องมีค่า k_1 , k_2 , k_3 , k_4 หลายชุด ในหัวข้อ ข.2.2 เป็นการแก้สมการอนุพันธ์ 2 สมการจะต้องมีค่า k_1 , k_2 , k_3 , k_4 และ l_1 , l_2 , l_3 , l_4 เนื่องจากโปรแกรมแม็ทแล็บสามารถคำนวณแมทริกซ์ได้ ดังนั้นจึงกำหนดตัวแปร k_1 , k_2 , k_3 , k_4 เป็น

ตัวแปรประเภทแมทริกซ์ ฟังก์ชันที่เขียนขึ้นเป็น รังกัตตาอันดับ 4 (RKSTEP4.M) มีรายละเอียดดังนี้

```
function[t1 ,y1]=rkstep4(f,start,startval,step)
%Solves dy/dt=f(t,y).start, finish are initial, final values of t
%startval is initial value of y, step is the increment in t
%Order=4 ,Classical method selecte
%10 Jan. 1997, BY SANTI LIMPORNCHAIJAROEN
b=[1/6 1/3 1/3 1/6];
d=[0 0.5 0.5 1];
c=[0 0 0 0; 0.5 0 0 0; 0 0.5 0 0;0 0 1 0];
y=startval; t=start;
k1=step*feval(f,t, y);
k2=step*feval(f,t+step*d(2),y+c(2,1)*k1);
k3=step*feval(f,t+step*d(3),y+c(3,1)*k1+c(3,2)*k2);
k4=step*feval(f,t+step*d(4),y+c(4,1)*k1+c(4,2)*k2+c(4,3)*k3);
y1=y +(b(1)*k1+b(2)*k2+b(3)*k3 +b(4)*k4);
t1=t+step;
```

จากฟังก์ชัน RKSTEP.M f คือเซ็ตของสมการอนุพันธ์ (ODE) โดยมีเวลาเริ่มต้น (start) ค่าเริ่มต้น (startval) และเวลาในหนึ่งสเต็พ (step)

ข.4 ตัวอย่างการทดสอบการแก้สมการอนุพันธ์ระหว่างฟังก์ชัน ODE45.M กับฟังก์ชันที่เขียนขึ้น (RKSTEP4.M)

สมการอนุพันธ์ที่ใช้ทดสอบเป็นสมการอนุพันธ์อันดับ 1 สองสมการคือ

$$\frac{dx}{dt} = 0.5(-s - \frac{x^3}{3} + px) \qquad (ข. 48)$$

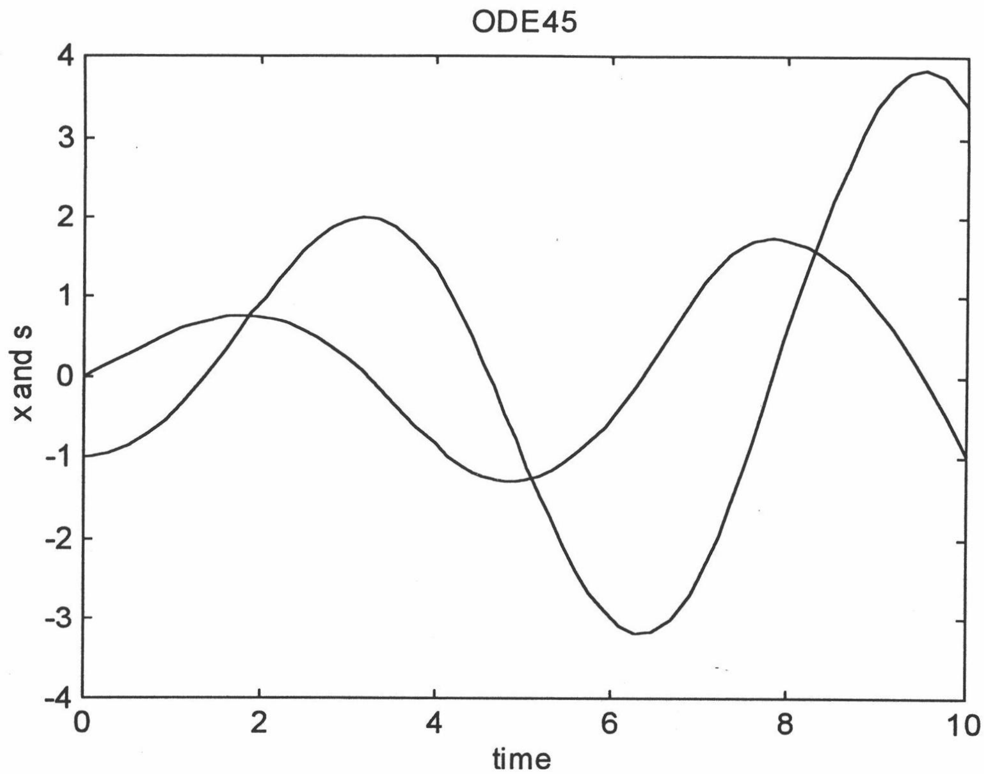
$$\frac{ds}{dt} = 2x \qquad (ข. 49)$$

จากสมการทั้งสองนี้เขียนให้อยู่ในรูปฟังก์ชันได้ดังนี้

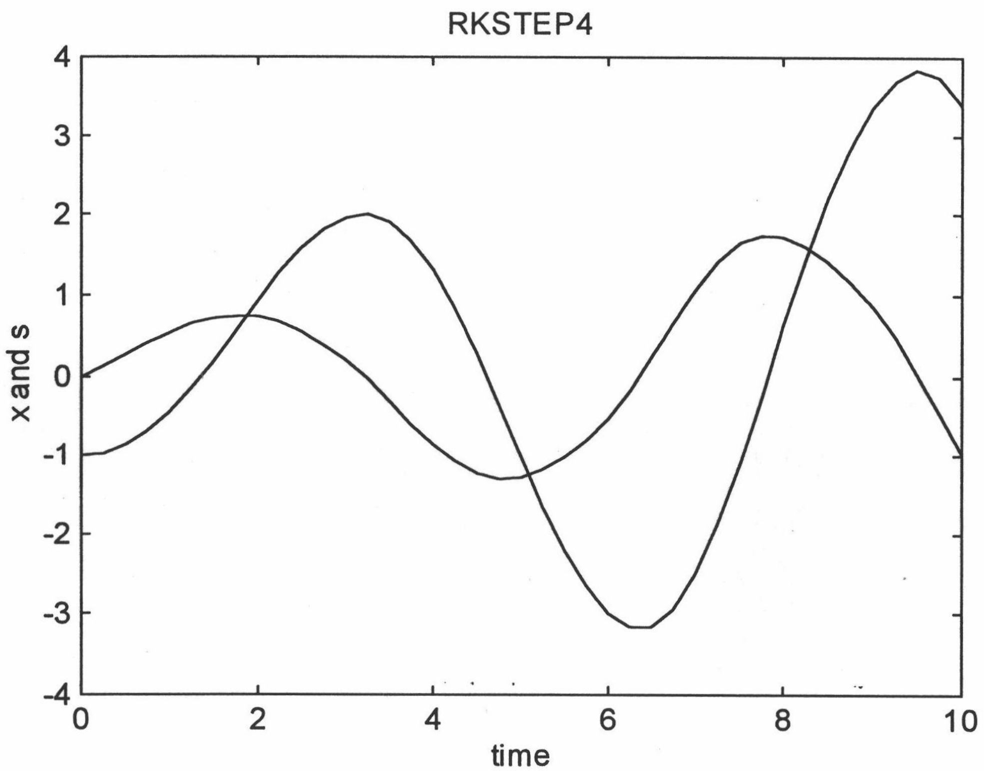
```
function fv=f504(t ,x)
%note that x and s are represented by x(1) and x(2)
global p
fv=zeros(2 ,1);
fv(1)= 0.5*( -x(2) -x(1)^3/3+p*x(1));
fv(2)=2*x(1);
```

โปรแกรมที่เปรียนเทียบฟังก์ชัน ODE45.M กับ ฟังก์ชันที่เขียนขึ้น (RKSTEP4.M) คือ

```
clear all
global p
p=1;
initx=[0 -1]';
[t1 x1]=ode45('f504',0,10,initx,0.001);
figure(1)
plot(t1,x1);
title('ODE45')
xlabel('time');
ylabel('x and s');
t2=0;
x2=initx;
while t2<10
[t2 x2]=rkstep_4('f504',t2,initx,.25);
initx=x2;
x3=[x3 ,x2];
t3=[t3, t2];
```



รูปที่ ข. 3 แสดงผลการซิมมูเลทโดยใช้ฟังก์ชัน ODE45.M



รูปที่ ข.4 แสดงผลการซิมมูเลทโดยใช้ฟังก์ชัน RKSTEP4.M

```
end
figure(4)
plot(t3,x3);
title('RKSTEP4')
xlabel('time');
ylabel('x and s');
```

ผลการทดสอบพบว่าทั้ง ODE45.M และRKSTEP4.M จะให้ผลใกล้เคียงกันซึ่งแสดงในรูปที่ ข. 3 และ ข.4 ตามลำดับ

ภาคผนวก ค.

ตัวควบคุมแบบป้อนกลับแบบพีไอดี

และหลักเกณฑ์ในการตัดสินสมรรถนะของระบบควบคุม

ในบทนี้กล่าวถึงตัวควบคุมแบบป้อนกลับแบบพีไอดี และการจูนตัวควบคุมแบบป้อนกลับแบบพีไอดี ในตอนท้ายกล่าวถึงหลักเกณฑ์ในการตัดสินสมรรถนะของระบบควบคุมซึ่งมีอยู่ 2 วิธีคือ พิจารณาจากผลการตอบสนองของกระบวนการ และพิจารณาจากอินทรีกรัลของค่าผิดพลาด

ค.1 ทฤษฎีของตัวควบคุมแบบป้อนกลับแบบพีไอดี

การควบคุมแบบป้อนกลับแบบพีไอดีเป็นการควบคุมแบบพื้นฐานที่เข้าใจง่ายและใช้ในการควบคุมกระบวนการทางอุตสาหกรรมทั่วไป โครงสร้างของระบบควบคุมแบบป้อนกลับดังแสดงในรูปที่ ค.1 ตัวควบคุมใช้ค่าตัวแปรควบคุมจากกระบวนการ c และเปรียบเทียบกับเซ็ทพอยท์ sp และสร้างสัญญาณเอาท์พุท u เพื่อปรับสภาพกระบวนการ โดยใช้ค่าความผิดพลาดในการควบคุม $e = sp - c$ ในการตัดสินใจ สำหรับความสัมพันธ์ระหว่าง ความผิด

พลาดในการควบคุม e กับสัญญาณเอาท์พุท u ของตัวควบคุมจะมีลักษณะอย่างไรจะขึ้นกับชนิดและคุณสมบัติของตัวควบคุม

เราสามารถแบ่งตัวควบคุมได้เป็น 3 ประเภทได้แก่ ตัวควบคุมแบบนิวแมติค ซึ่งทำงานโดยใช้ลม ตัวควบคุมแบบอิเล็กทรอนิค ทำงานโดยใช้วงจรอิเล็คทรอนิคแบบเชิงเส้น (linear circuit) หรือไมโครโปรเซสเซอร์สร้างสัญญาณเอาท์พุทโดยการเลียนแบบ และจำลองการทำงานของตัวควบคุมแบบนิวแมติค และตัวควบคุมแบบอิเล็กทรอนิคจากแบบจำลองทางคณิตศาสตร์ของตัวควบคุมแบบเดิม ตัวควบคุมแบบพื้นฐานที่ใช้ในระบบควบคุมแบบป้อนกลับในกระบวนการอุตสาหกรรมโดยทั่วไปมี 4 ประเภทคือ ตัวควบคุมแบบพี ตัวควบคุมแบบพีไอ ตัวควบคุมแบบพีดี และตัวควบคุมแบบพีไอดี

ก. ตัวควบคุมแบบพี (proportional controller หรือ P controller)

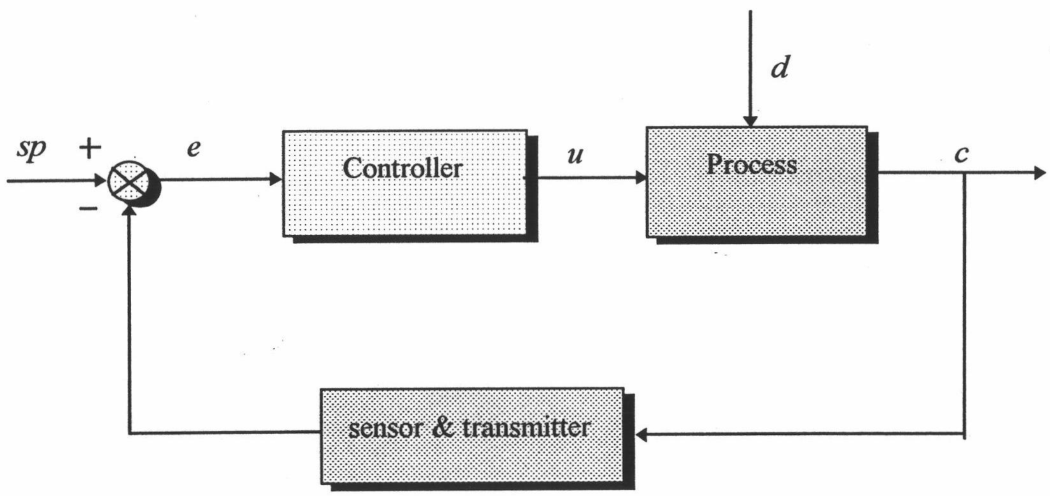
สัญญาณเอาท์พุทจะแปรผันตรงตามค่าความผิดพลาดในการควบคุม

$$u(t) = K_c e(t) + u_s \qquad \text{(ค.1)}$$

K_c คือเกนสัดส่วน (proportional gain) ของตัวควบคุม, u_s คือ ค่าไบอัส (bias signal) ของตัวควบคุม, และ $u(t)$ เป็นค่าเอาท์พุทของตัวควบคุม ตัวควบคุมแบบพีสามารถแสดงอัตราการควบคุม 2 วิธี คือเกนสัดส่วน K_c และ แบนด์สัดส่วน PB (porportional band) โดย PB = 100 / K_c แบนด์สัดส่วนหมายถึงค่าความผิดพลาดในการควบคุม e ที่ทำให้สัญญาณเอาท์พุทมีค่าสูงสุด ตัวอย่างเช่น ถ้าตัวควบคุมแบบพีมีค่าแบนด์สัดส่วน PB = 200% จะมีค่า K_c =100/200=0.5 แทนค่า K_c ลงในสมการที่ (ค.1) โดยสมมติค่า u_s=0

$$u(t) = 0.5*e(t) \qquad (ค.2)$$

แสดงว่าความผิดพลาดในการควบคุม $e(t)$ จะต้องมีค่า 2 หรือ 200% จึงจะทำให้สัญญาณเอาท์พุทมีค่าสูงสุด คือ 1 หรือ 100 %



รูปที่ ค.1 ระบบการควบคุมแบบป้อนกลับ (feedback control system)

กระบวนการอุตสาหกรรมทั่วไปสามารถใช้ตัวควบคุมที่มีเกนสัดส่วน $0.2 \le Kc \le 100$ หรือค่าแบนด์สัดส่วนน้อยจะมีความไวในการตอบสนองต่อค่าความผิดพลาดในการควบคุมสูง ทรานส์เฟอร์ฟังก์ชันของตัวควบคุมแบบพีคือ

$$G_c(s) = K_c \qquad (ค.3)$$

ข. ตัวควบคุมแบบพีไอ (Proportional-Integral controller หรือ PI controller)

สัญญาณเอาท์พุทมีความสัมพันธ์กับค่าความผิดพลาดในการควบคุมตามสมการ

$$u(t) = K_c e(t) + \frac{K_c}{\tau_I}\int_0^t e(t)dt + u_s \qquad (ค.4)$$

เมื่อ τ_I คือ ค่าคงที่เวลาอินทีกรัล (integral time constant) หรือเวลารีเซ็ท (reset time) ของตัวควบคุมแบบไอ ตัวควบคุมแบบไอแสดงอัตราการควบคุมโดยใช้ค่าคงที่เวลาอินทรีกรัล หรือเวลารีเซ็ท มีหน่วยเป็นครั้ง/นาทีซึ่งมีค่าเป็น $1 / \tau_I$

กระบวนการทางอุตสาหกรรมทั่วไปสามารถใช้ตัวควบคุมที่มีค่าคงที่เวลาอินทีกรัล $0.1 \le \tau_I \le 50$ นาที รูปที่ ค.2 แสดงการตอบสนองของตัวควบคุมแบบพีไอเริ่มต้น $t = 0$ สัญญาณเอาท์พุทจากการควบคุมแบบพีอย่างเดียวจะมีค่า $u = K_c{*}e(t)$ เมื่อเวลาผ่านไป τ_I สัญญาณเอาท์พุทจากการควบคุมแบบไอจะมีค่าเท่ากับสัญญาณเอาท์พุทแบบพีอย่างเดียว

$$\frac{K_c}{\tau_I}\int_0^{\tau} e(t)dt = \frac{K_c}{\tau_I} e\tau_I = K_c \quad e \qquad (ค.5)$$

ค่าคงที่เวลาอินทีกรัล τ_I หมายถึงช่วงเวลาที่ตัวควบคุมใช้ในการเพิ่มสัญญาณเอาท์พุทแบบไอจนมีค่าเท่ากับสัญญาณที่ได้จากการควบคุมแบบพีเพียงอย่างเดียวหนึ่งครั้ง หรือตัวควบคุมแบบไอต้องใช้เวลา τ_I เพื่อเพิ่มสัญญาณเอาท์พุทให้มีค่าเท่ากับสัญญาณที่ได้จากการควบคุมแบบพีหนึ่งครั้ง ตัวควบคุมแบบไอมีคุณสมบัติต่างจากตัวควบคุมแบบพีคือ สัญญาณเอาท์พุทจะมีการเปลี่ยนแปลงค่าเพิ่มขึ้นหรือลดลงตลอดเวลาที่ยังมีค่าความผิดพลาดในการควบคุมอยู่ จากสมการที่ (ค.4) ทรานส์เฟอร์ฟังก์ชันของตัวควบคุมแบบพีไอคือ

$$G_c(s) = K_c\left(1 + \frac{1}{\tau_I s}\right) \qquad (ค.6)$$

ค. ตัวควบคุมแบบพีไอดี (Proportional-Integral-Derivative controller หรือ PID controller)

สัญญาณเอาท์พุทของตัวควบคุมคือ

$$u(t) = K_c e(t) + \frac{K_c}{\tau_I}\int_0^t e(t)dt + K_c \tau_D \frac{de(t)}{dt} + u_s \qquad (ค.7)$$

เมื่อ τ_D คือค่าคงที่เวลาอนุพันธ์ (derivative time constant) ของตัวควบคุมแบบดี ตัวควบคุมสร้างสัญญาณเอาท์พุทแบบดีจากเทอม $de(t)/dt$ ในสมการที่ (ค.7) ซึ่งมีค่าเปลี่ยนแปลงทันทีที่ค่าความผิดพลาดในการควบคุม $e(t)$ มีการเปลี่ยนแปลงไม่ว่าจะมีค่าเพิ่มขึ้นหรือลดลง และสัญญาณการควบคุมแบบดีจะถูกสร้าง เพื่อลดความผิดพลาดในการควบคุมล่วงหน้า

ทรานส์เฟอร์ฟังก์ชันของตัวควบคุมแบบพีไอดี คือ

$$G_c(s) = K_c\left(1 + \frac{1}{\tau_I s} + \tau_D s\right) \qquad (ค.8)$$

ง. ตัวควบคุมแบบพีดี (Proportional-Derivative controller หรือ PD controller)

สัญญาณการควบคุมแบบพีดีจะแสดงด้วยสมการ

$$u(t) = K_c e + K_c \tau_D \frac{de(t)}{dt} + u_s \qquad (ค.9)$$

การควบคุมแบบพีดีมีการเปลี่ยนแปลงที่เป็นสัดส่วนกับอนุพันธ์ของความผิดพลาด การควบคุมแบบนี้จะเรียกได้ว่าเป็นการควบคุมแบบคาดการณ์ล่วงหน้า (anticipatory control) พฤติกรรมของตัวควบคุมแบบพีดีแสดงได้โดยพิจารณาจากการตอบสนองต่อการเปลี่ยนแปลง

แบบเชิงเส้นของความผิดพลาดในรูปที่ ค.3 ซึ่งเป็นผลตอบสนองต่อฟังก์ชัน $e(t) = At$ ของสมการที่ (ค.9) จะได้เป็นสมการ (ค.10)

$$u(t) = A_c t + A_c \tau_D + u_s \qquad (ค.10)$$

จะเห็นได้ว่า u เปลี่ยนแปลงอย่างทันทีทันใดด้วยปริมาณ $AK_c\tau_D$ ซึ่งเป็นผลจากการกระทำของเทอมอนุพันธ์ จากนั้นเปลี่ยนแปลงอย่างเชิงเส้นด้วยอัตรา AK_c ทรานส์เฟอร์ฟังก์ชันของตัวควบคุมแบบพีดีคือ

$$G_c(s) = K_c(1+\tau_D s) \qquad (ค.11)$$

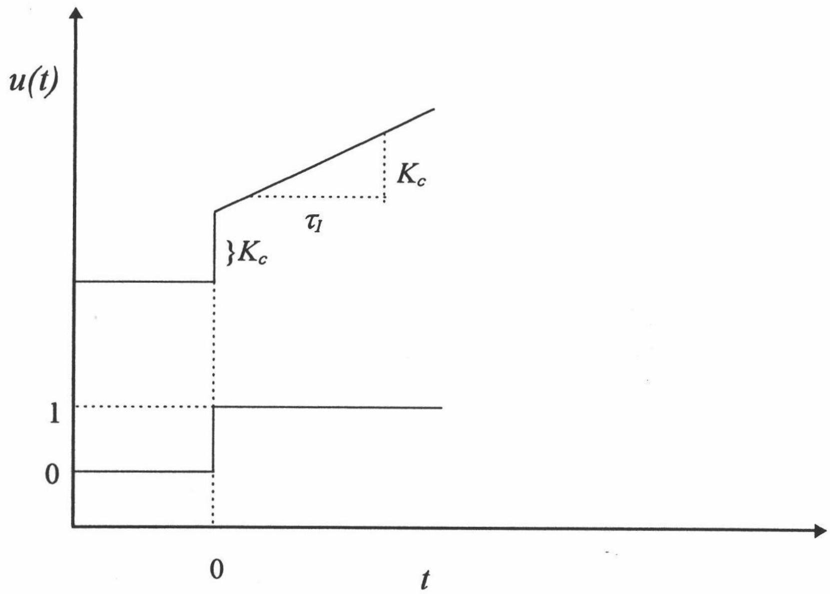
ตัวควบคุมป้อนกลับแบบพื้นฐานที่กล่าวมานี้คุณลักษณะที่สำคัญคือลักษณะของเกน (K_c) และพารามิเตอร์ที่คงที่ ซึ่งลักษณะแบบนี้จะเหมาะสำหรับการควบคุมกระบวนการที่ไม่ซับซ้อน

ค.2 การจูนตัวควบคุมแบบป้อนกลับแบบพีไอดี

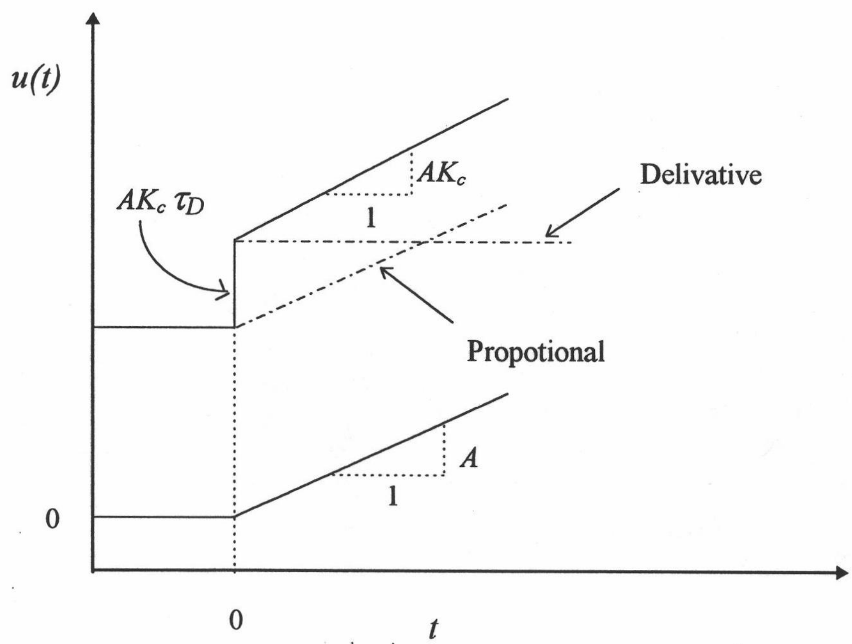
ในการจูนค่าพารามิเตอร์ของตัวควบคุมแบบพีไอดี จะใช้วิธีของ "ซีเกลอร์-นิโคลส์" (Ziegler-Nichols) ซึ่งเป็นวิธีการไซเคิลแบบต่อเนื่อง (continuous cycling method) ซึ่งการควบคุมประเภทนี้อาศัยการจูนแบบลองผิดลองถูก (trial and error tunning) สามารถแบ่งออกเป็นขั้นตอนต่าง ๆ ได้ดังนี้คือ

ขั้นตอนที่1 : การจูนตัวควบคุมแบบลองผิดลองถูก (trial and error tunning)

เป็นการจูนตัวควบคุมเพื่อหาค่า K_C ซึ่งสามารถแบ่งเป็นขั้นตอนย่อยได้ดังนี้



รูปที่ ค.2 การตอบสนองของตัวควบคุมแบบพีไอ (Coughanowr, 1991)



รูปที่ ค.3 การตอบสนองของตัวควบคุมแบบพีดี (Coughanowr, 1991)

1. กำจัดค่าอินทีกรัล (integral) และ ดีไรเวทีฟ (derivative) ออก โดยการตั้งค่า τ_I ให้มีค่าสูงที่สุด และค่า τ_D ให้มีค่าต่ำสุด

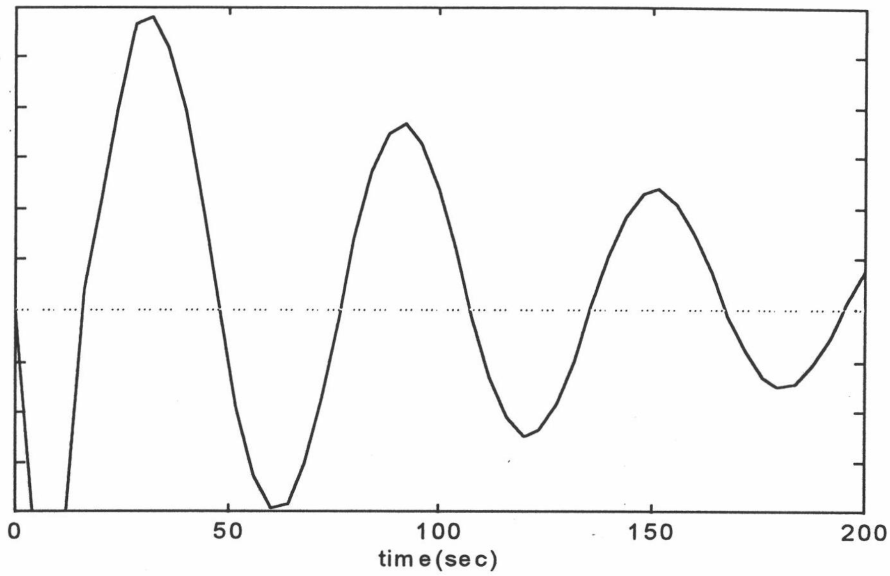
2. สุ่มค่า K_C ขึ้นมาค่าหนึ่ง (เริ่มที่ค่า K_C น้อย ๆ)

3. สังเกตดูลักษณะของกราฟของการตอบสนอง (response) ที่ได้ ถ้าเกิดโอเวอร์ชูต (overshoot) แสดงว่าจะต้องลดค่า K_C ลงและถ้ากราฟการตอบสนองมีแนวโน้มเข้าใกล้ค่าเซ็ทพอยท์แสดงว่าจะต้องเพิ่มค่า K_C ให้มากขึ้น ทั้งนี้การเพิ่มหรือลดค่า K_C จะทำจนกระทั่งกราฟของการตอบสนองที่ได้มีลักษณะเป็นกราฟคงที่ ซึ่งจะมีแอมพลิจูด (amplitude) และ เวลา (period) ที่เท่ากันโดยตลอดซึ่งวิธีการตอบสนองแบบนี้จะเรียกว่าวิธีการไซเคิลแบบต่อเนื่อง (continuous cycling method) ค่า K_C ที่ได้จะมีค่าเท่ากับค่า K_{CU} ซึ่งค่า K_{CU} นี้เราจะนำไปใช้ในการหาค่าการควบคุมสัดส่วน (proportional control) ของค่าพีไอดี โดยวิธีของ "ซีเกลอร์-นิโคลส์"

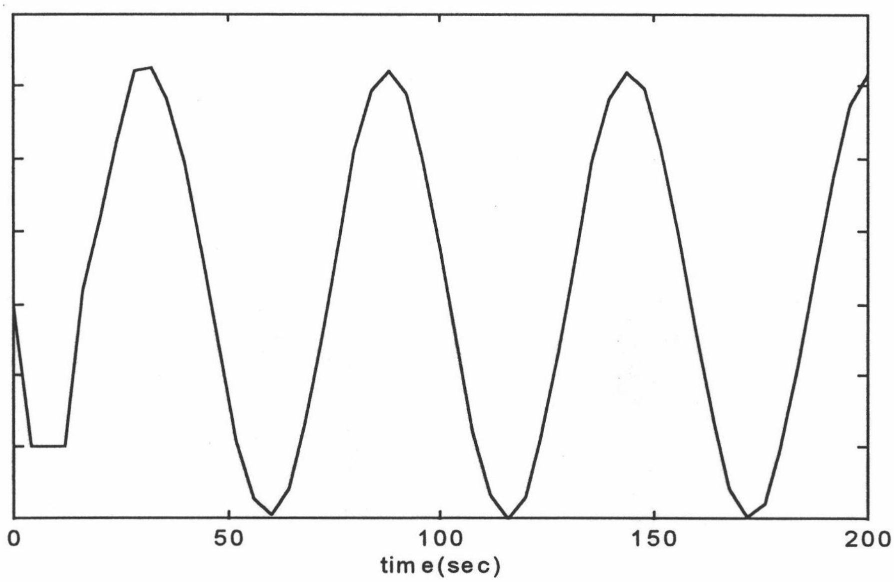
นิยามของ K_{CU} หรือ "utimate gain" คือค่า K_C ที่มากที่สุดของตัวควบคุมกระบวนการที่ทำให้กระบวนการแบบวงจรปิด (close loop) มีเสถียรภาพที่ดีเมื่อใช้ค่าสัดส่วน (proportional) ในการควบคุมกระบวนการเพียงอย่างเดียว

จากรูปที่ ค.4 ถึง ค.6 แสดงถึงลักษณะของการตอบสนองที่ค่า K_{CU} ต่าง ๆ กัน

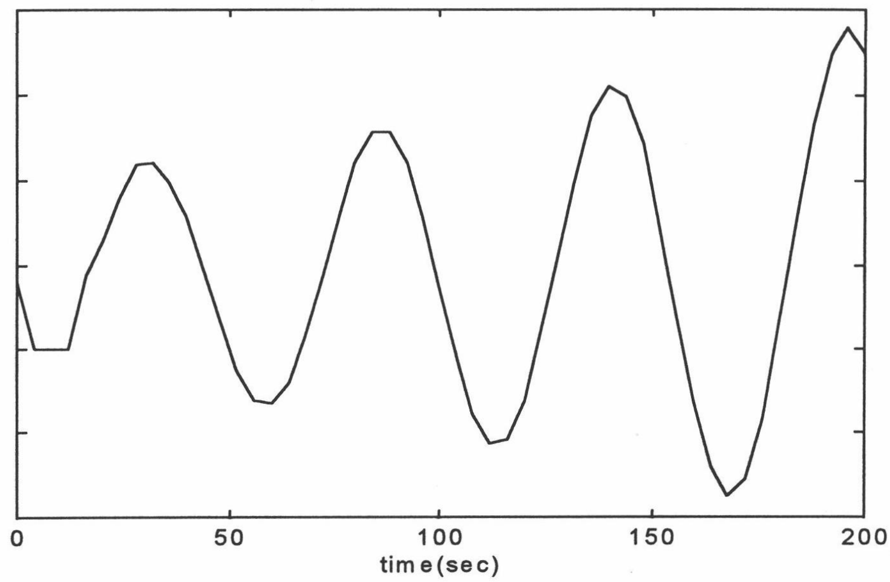
1. $K_C < K_{CU}$ ในกรณีที่การตอบสนอง เกิดแบบโอเวอร์แดม (overdamp) และค่า K_C น้อยกว่าค่า K_{CU} มาก ๆ การตอบสนองจะเกิดการแกว่ง (oscillatory) การแก้ไขก็คือการเพิ่มค่า K_C จนได้การตอบสนองเป็นแบบไซเคิลแบบต่อเนื่อง



รูปที่ ค.4 การจูนตัวควบคุมแบบพีไอดี โดยวิธีของ “ซีเกลอร์-นิโคลส์” เมื่อ $K_C < K_{CU}$



รูปที่ ค.5 การจูนตัวควบคุมแบบพีไอดี โดยวิธีของ “ซีเกลอร์-นิโคลส์” เมื่อ $K_C = K_{CU}$



รูปที่ ค.6 การจูนตัวควบคุมแบบพีไอดี โดยวิธีของ "ซีเกลอร์-นิโคลส์" เมื่อ $K_C > K_{CU}$

ตารางที่ ค.1 การหาค่าพารามิเตอร์ของตัวควบคุมแบบป้อนกลับแบบพีไอดี

Controller	K_C	τ_I	τ_D
P	$0.5K_{CU}$	-	-
PI	$0.45K_{CU}$	$P_U/1.2$	-
PID	$0.6K_{CU}$	$P_U/2$	$P_U/8$

2. $K_C = K_{CU}$ ในกรณีนี้การตอบสนองจะมีค่าแอมพลิจูด และเวลาที่คงที่ในลักษณะนี้เรียกว่าการไซเคิลแบบต่อเนื่อง ค่า K_C ที่ได้จะมีค่าสูงสุด

3. $K_C > K_{CU}$ (without saturation) ในกรณีนี้ กระบวนการจะมีเสถียรภาพดีกว่าแต่ค่า K_{CU} ที่ได้จะมีค่ามากจนเกินไปจนมีผลทำให้การควบคุมกระบวนการทำได้ไม่ดีนัก

ขั้นตอนที่ 2 :วิธีการควบคุมแบบการไซเคิลแบบต่อเนื่อง (continuous cycling method)

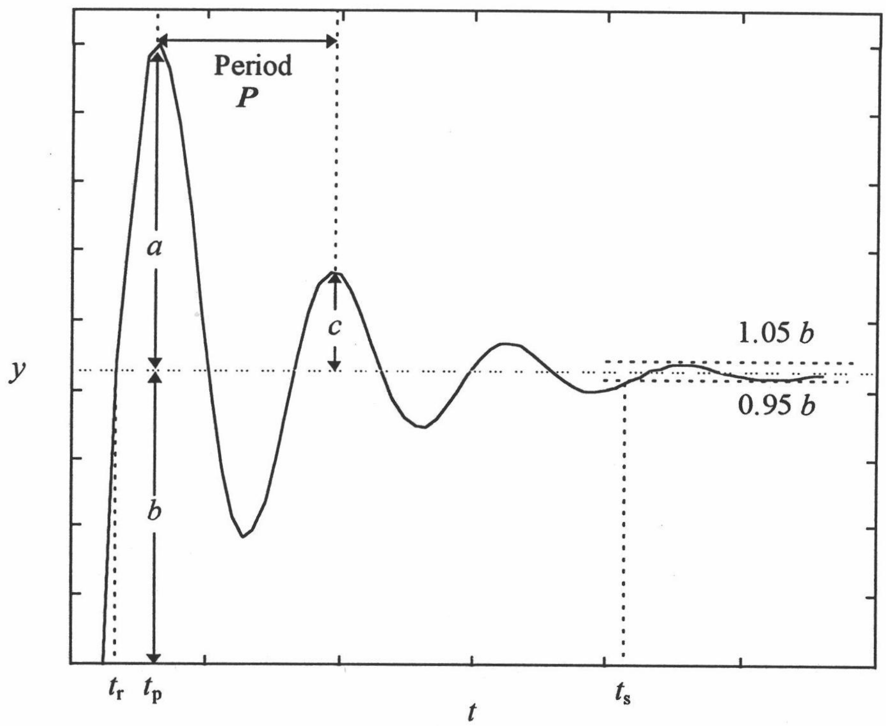
การควบคุมแบบการไซเคิลแบบต่อเนื่องนี้ซีเกลอร์และนิโคลส์ได้เป็นผู้นำเสนอครั้งแรกเมื่อ ค.ศ.1942 โดยการนำค่า K_{CU} ที่ได้จากการทำลองผิดลองถูกมาหาค่าพารามิเตอร์ของตัวควบคุมแบบพีไอดีแสดงดังในตารางที่ ค.1 จากหนังสือ Process Dynamic and Control โดย Seborg (ตารางที่ 13.2 หน้า 298)

ค.3 หลักเกณฑ์ตัดสินสมรรถนะของระบบควบคุม

ผู้ออกแบบระบบควบคุมจะต้องกำหนดหลักเกณฑ์การตัดสินใจในการออกแบบระบบควบคุม เพื่อใช้เป็นมาตราฐานเปรียบเทียบระบบควบคุมหลายระบบว่าระบบใดสามารถให้ผลการควบคุมที่ดีที่สุด และความต้องการที่กำหนดไว้ในการออกแบบ หลักเกณฑ์ที่ใช้ในการตัดสินมี 2 วิธีซึ่งขึ้นอยู่กับวัตถุประสงค์ในการควบคุม

1. พิจารณาจากผลการตอบสนองของกระบวนการ

เกณฑ์การตัดสินคือระบบที่ดีจะต้องสามารถลดค่าโอเวอร์ชูท (a) ลดค่าเวลาเข้าสู่สมดุล (t_s) ลดเวลาขาขึ้น (t_r) ให้มีค่าน้อยที่สุด หรือต้องมีค่าอัตราลดทอน (Decay ratio) หรือ c/a ให้มีค่าเป็น 1/4 ซึ่งแสดงในรูปที่ ค.7



รูปที่ ค.7 ลักษณะการตอบสนองของกระบวนการ

2. พิจารณาจากอินทีกรัลของค่าความผิดพลาด

เป็นการตรวจสอบผลการตอบสนองของระบบควบคุมตามเวลาทุกจุด ตั้งแต่กระบวนการเริ่มต้นเปลี่ยนแปลงเมื่อเวลา $t = 0$ จนเข้าสู่สภาวะสมดุล เมื่อเวลา $t \to \alpha$ ได้แก่

ก. อินทีกรัลของกำลังสองของความผิดพลาด (*ISE*)

$$ISE = \int_0^{\alpha} e^2(t)dt \qquad (ค.12)$$

ข. อินทีกรัลของค่าสัมบูรณ์ของความผิดพลาด (*IAE*)

$$IAE = \int_0^{\alpha} e(t)dt \qquad (ค.13)$$

ค. อินทีกรัลเวลาของค่าสัมบูรณ์ของความผิดพลาด (*ITAE*)

$$ITAE = \int_0^{\alpha} te(t)dt \qquad (ค.14)$$

โดย $e(t)= sp(t) - c(t)$

การออกแบบตัวควบคุมที่ดีนั้นจะต้องมีค่าอินทีเกรตของความผิดพลาดน้อยที่สุด หลักเกณฑ์ในการตัดสินทั้งหมดนี้จะมีความเหมาะสมสำหรับระบบควบคุมแต่ละระบบดังนี้

ก. สำหรับค่าความผิดพลาดที่มีค่ามาก ใช้ *ISE* จะดีกว่า *IAE* เนื่องจากค่าความผิดพลาดจะถูกยกกำลังสอง ซึ่งจะทำให้ได้ค่าของการอินทีเกรตมีค่ามากตามไปด้วย

ข. สำหรับความผิดพลาดที่มีค่าน้อย ใช้ *IAE* จะดีกว่า *ISE* เนื่องจากเมื่อทำการยกกำลังสองความผิดพลาดที่มีค่าน้อย ๆ จะทำให้ได้ค่าที่น้อยลงไปอีกทำให้เปรียบเทียบได้ยาก

ค. สำหรับระบบที่มีช่วงการทดสอบที่ยาวนาน หลักเกณฑ์ *ITAE* จะเหมาะสำหรับการออกแบบระบบควบคุม เนื่องจากค่าความผิดพลาดที่ปรากฏในช่วงที่ t มีค่ามากจะถูกขยายให้มีค่ามากขึ้นไปด้วยถึงแม้ว่าจะเป็นค่าผิดพลาดที่น้อย ในช่วงเวลาอินทีเกรต

ภาคผนวก ง.

ตัวอย่างโค้ดโปรแกรมที่สำคัญ

โปรแกรมที่เขียนขึ้นในงานวิจัยที่สำคัญคือ โปรแกรมสำหรับซิมมูเลทกระบวนการเพื่อนำข้อมูลอินพุท และเอาท์พุทที่ได้จากการซิมมูเลทนำไปใช้ในการฝึกข่ายงานนิวรัล และโปรแกรมสำหรับฝึกข่ายงาน

ง.1 โปรแกรมสำหรับซิมมูเลทกระบวนการ

โปรแกรมนี้จะให้ผู้ใช้เลือกเวลาในการซิมมูเลท และอินพุทจะเป็นแบบสเต็พ โดยผู้ใช้สามารถเลือกชนิดของอินพุท ซึ่งอาจเป็นการสุ่มเฉพาะความสูง หรือทั้งความสูง และความกว้าง ในการซิมมูเลทจะใช้วิธีรังกัตตา อันดับ 4 ในการแก้สมการอนุพันธ์ ซึ่งมีรายละเอียดดังนี้

```
% Define global variable
global Fo
global delta
delta=1;
stime=1;
simtime=input('enter runtime');
```

```
% Initial Condition
initx=[3.4 2.05]' ;
%SELECT TYPE OF INPUT
z1=menu('Select type of Input',...
        'Random Amplitude',...
        'Random Amplitude and Range');
disp('')
t=0;
Fo=0;  i=0;   time=0;   flow=0;
a=-1;b=1;  c=1; d=10;
flow(1)=35.1;   time(1)=0;
while (t < (simtime))
  if z1==1
    n=stime;
  else
    r=rand(1,1);
    n=round(((d-c)*r+c));
  end
  r=rand(1,1);
  Fin=((b-a)*r+a)*3.51+35.1;
  for j=1:n
     if t>= simtime
       break
     end
     i=i+1;
     t=t+h;
     flow(i+1)=Fin;
```

```
  end
end

i=1;t=0;
%initial condition
x=[3.4 2.05]' ;
while (t < (simtime))
Fo=flow(i);
initx=x;
[t x]=rkstep_4('fgra',t,initx,delta);
t2(i)=t;
x2(i,1)=x(1);
x2(i,2)=x(2);
i=i+1;
end
```

ง.2 โปรแกรมสำหรับฝึกข่ายงาน

ในโปรแกรมนี้จะให้ผู้ใช้ป้อนว่าจะใช้ฟังก์ชันกระตุ้นประเภทไหน และกำหนดค่าเดดไทม์ของกระบวนการ ข้อมูลที่ใช้ในการฝึกได้จากการซิมมูเลทในหัวข้อ ง.1 โดยเก็บไว้ในชื่อไฟล์ gra00_fo.dat และ gra00_h.dat หลังจากนั้นผู้ใช้เลือกจำนวนของอินพุทที่ต้องการหน่วงเวลา, n_u และจำนวนของเอาท์พุทที่ต้องการหน่วงเวลา, n_y ซึ่งจะได้เป็นอินพุทเวคเตอร์, P ในขั้นตอนต่อไปโปรแกรมจะสุ่มค่าน้ำหนักเริ่มต้นโดยใช้ฟังก์ชัน iniff( ) และทำการฝึกข่ายงานโดยใช้ฟังก์ชัน trainbpx( ) ซึ่งเป็นฟังก์ชันในการฝึกโดยใช้อัลกอริธึมการกระจายย้อนกลับ

ในตอนท้ายจะทำการทดสอบข่ายงานที่ทำการฝึกแล้วมาทดสอบกับข้อมูลอีกชุดหนึ่งคือไฟล์ gra0_fo.dat และ gra0_h.dat รายละเอียดของโปรแกรมมีดังนี้

```
% Select Activation function
z1=menu('Select  Activation function of input-hidden:',...
        'logsig(0 - 1)',...
        'tansig(-1 - 1)');
disp('')
if z1==1
    act_1='logsig';
else  act_1='tansig';
end

z2=menu('Select  Activation function of hidden-output:',...
        'logsig(0 - 1)',...
        'tansig(-1 - 1)');
disp('')
if z2==1
    act_2='logsig';
else    act_2='tansig';
end
 if strcmp(act_1,'logsig')
   min1=0;
```

```
      max1=1;
    end
    if strcmp(act_1,'tansig')
      min1=-1;
      max1=1;
    end
    if strcmp(act_2,'logsig')
      min2=0;
      max2=1;
    end
    if strcmp(act_2,'tansig')
      min2=-1;
      max2=1;
    end
delay=input('enter number of delay times');
load gra00_fo.dat
load gra00_h.dat
 [A,B] = size(gra00_fo);
Xout=gra00_h((1+delay):B);
Xin=gra00_fo(1:(B-delay));
clear A B
clear gra00_fo gra00_h
% Scale range
```

```
Fmax=max(Xin)
Fmin=min(Xin)
hmax=max(Xout)
hmin=min(Xout)
T=min1+((max1-min1)/(hmax-hmin)).*(Xout-hmin);
X=min2+((max2-min2)/(Fmax-Fmin)).*(Xin-Fmin);
%*******************************************
% Recurrent
%*******************************************
n=input('enter number of recurrent of input');
m=input('enter number of recurrent of output');
% initialize X
if n==0
    PP=delaysig(T,1,m);
    P=[X ; PP]
   clear PP
   if m==0 P=X ; end
else
    XX=delaysig(X,1,n);
    PP=delaysig(T,1,m);
    P=[X ; XX ; PP]
     if m==0  P=[X;XX];  end
     clear XX PP
```

```
end

% INITIALIZE NETWORK ARCHITECTURE

%=================================

[R,Q] = size(P); S1 =5; [S2,Q] = size(T);

% INITFF is used to initialize the weights and biases

[W1,b1,W2,b2]=initff(P,S1,act_1,T,act_2);

% TRAINING PARAMETERS

disp_freq = 5;

max_epoch =150;

err_goal =0.000001;

momentum = 0.95;

err_ratio = 1.04;

TP = [disp_freq max_epoch err_goal NaN NaN NaN momentum ];

[ W1,b1,W2,b2,ep,tr] = trainbpx(W1,b1,act_1,W2,b2,act_2,P,T,TP);

% RESULTS

A=simuff(P,W1,b1,act_1,W2,b2,act_2);

SSE = sumsqr(A-T);

hout=((hmax-hmin)/(max1-min1)).*(A-min1)+hmin;

figure(1)

plot(1:length(hout),hout,'-',1:length(Xout),Xout,'--');

xlabel('time(s)');

ylabel('h');

title('TRAINING');
```

```
%*********************************
%Cross Validate
%*********************************
Xin=0;Xout=0;T=0;
load gra0_fo.dat
load gra0_h.dat
[A,B] = size(gra0_fo);
X2out=gra0_h((1+delay):B);
X2in=gra0_fo(1:(B-delay));
clear A B
clear gra0_fo gra0_h
T2=min1+((max1-min1)/(hmax-hmin)).*(X2out-hmin);
X2=min2+((max2-min2)/(Fmax-Fmin)).*(X2in-Fmin);
[R3,Q3]=size(X2in);
%******************************************
% Recurrent
%******************************************
if n==0
   PP2=delaysig(T2,1,m);
   P2=[X2 ; PP2]
   clear PP2
else
   XX2=delaysig(X2,1,n);
```

```
    PP2=delaysig(T2,1,m);
    P2=[X2 ; XX2 ; PP2];
    clear XX2  PP2
end

% RESULTS
A2=simuff(P2,W1,b1,act_1,W2,b2,act_2);
SSE = sumsqr(A2-T2);
h2out=((hmax-hmin)/(max1-min1)).*(A2-min1)+hmin;
figure(2)
plot(1:length(h2out),h2out,'-',1:length(X2out),X2out,'--');
xlabel('time(s)');
ylabel('h');
title('CROSS VALIDATION');
```

ประวัติผู้เขียน

นายสันติ ลิ้มพรชัยเจริญ เกิดเมื่อวันที่ 8 กุมภาพันธ์ พ.ศ. 2513 สำเร็จการศึกษาในระดับชั้นมัธยมศึกษาปีที่ 6 จากโรงเรียนวัดสุทธิวราราม เมื่อ พ.ศ. 2532 สำเร็จการศึกษาระดับปริญญาตรี ปริญญาวิทยาศาสตร์บัณฑิต สาขาเทคโนโลยีชีวภาพ จากมหาวิทยาลัยเกษตรศาสตร์ เมื่อปี พ.ศ. 2536