บทที่ 6 ภาคผนวก

6.1. ประวัตินักวิจัย

Benjamas Panomruttanarug

CONTACT INFORMATION

King Mongkut's University of Technology Thonburi
Department of Control system and Instrumentation Engineering
Bangkok, Thailand 10140

E-mail: benjamas.pan@kmutt.ac.th

Tel: 02.470.9096

Web: www.inc.eng.kmutt.ac.th/~yoodyut

RESEARCH INTERESTS

Iterative Learning Control, Repetitive Control, Kalman Filtering, Servo Control in HDD, Autonomous Parallel Parking System, Self Balancing System.

EDUCATION

Columbia University New York, NY

Ph.D., Electrical Engineering, October 2006

- Dissertation Topic: **"Simple and Effective Optimization Based FIR Repetitive Controller Design and Conversion to Learning Control Using Initial Condition Updates"**
- Advisor: Professor Richard W. Longman

M.S., Electrical Engineering, May 2002

Mahidol University Nakornprathom, Thailand

B.S., Electrical Engineering, May 1999

ACTIVITIES AND HONORS

Received junior research fellowship from the French Embassy (10 out of 50 approx.), 2011

Received visiting research fellowship from University of Electro-Communication (UEC) (3 researchers are selected from worldwide applications), 2010

Royal Thai Government scholarship to pursue M.S. and Ph.D., 2000

Completed Tripetch Isuzu's Junior Executive Program (program accepted 30 out of approx. 2000 candidates from all colleges in Thailand), 1999

Department assistant and treasurer for Mahidol Student Engineers Association, 1996

SELECTED PUBLICATIONS

M. Q. Phan, R. W. Longman, B. Panomruttanarug, and S. C. Lee, **"Robustification of iterative learning control and repetitive control by averaging,"** *International Journal of Control*, Vol. 86, No. 5, pp. 855-868, 2013.

B. Panomruttanarug and K. Higuchi, **"Fuzzy Logic Based Autonomous Parallel Parking System with Kalman Filtering,"** *SICE Journal of Control, Measurement, and System Integration*, Vol. 3, No. 4, pp. 266-271, July 2010.

B. Panomruttanarug and R.W.Longman, **"Repetitive Controller Design Using Optimization in Frequency Domain,"** *Proceedings of the 2004 AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, Providence, Rhode Island, August, 2004.

R. W. Longman, K. Xu, and B. Panomruttanarug, **"Designing Learning Control that is Close to Instability for Improved Parameter Identification."** *Proceedings of the 2006 Conference on High Performance Scientific Computing*, Springer Verlag, 2008, pp.359-370, ISBN 3-540-79408-0.

B. Panomruttanarug, and R. W. Longman, **"Using a Kalman Filter to Attenuate Noise in Learning and Repetitive Control can Easily Degrade Performance,"** *IEEE Proceeding of International Conference on Instrumentation, Control and Information Technology (SICE)*, 2008, pp. 3453-3458, ISBN 978-4-907764-30-2.

B. Cheowait, B. Panomruttanarug, W. Lenwari, **"Design and Analysis of Current Control**

for Shunt Active Filter Based on Repetitive Control Technique Using Optimization in the Frequency Domain," *2008 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2009, pp. 871-876, ISBN 978-1-4244-2678-2.

P. Thongsuk, and B. Panomruttanarug, **"FIR Optimization Control for External Vibration in Self Servo Track Writer,"** *IEEE Proceeding of International Conference on Instrumentation, Control and Information Technology (SICE)*, 2010, pp. 13-17, ISBN 978-1-4244-7642-8.

T. Jatturongkapolkul and B. Panomruttanarug, **"Frequency Domain System Identification for Servo System in HGA Testing System Using Particle Swarm Optimization,"** *Proceedings of the IEEE Industrial Electronics Society (IECON)*, 2011, pp. 587-591, ISBN 978-1-61284-969-0.

TALKS

April, 2011 Ecole Nationale de l'Aviation Civile (ENAC), Toulouse, France.

October, 2010 Department of Advanced Energy, Graduate School of Frontier Sciences, University of Tokyo, Chiba, Japan.

August, 2008 Department of Aerospace Engineering, Kanagawa Institute of Technology (KAIT), Kanagawa, Japan.

May, 2007 Department of Mechanical Engineering, Hong Kong University of Technology and Science, Kowloon, Hong Kong.

November, 2006 Department of Aerospace Engineering, Tokyo Metropolitan Institute of Technology, Tokyo, Japan.

November, 2006 Department of Mechanical Engineering, Osaka University, Osaka, Japan.

June, 2006 Interdisciplinary Center for Scientific Computing (IWR), University of Heidelberg, Heidelberg, Germany.

June, 2004 Center for Self-Organizing and Intelligent Systems, Utah State University, USA.

PROFESSIONAL SERVICE

Reviewer for *International Journal of Machine Tools & Manufacture*, *Automatica*, *IEEE Transactions on Circuits and Systems I*, Springer Verlag, *IEEE/ASME AIM*, IEEE/RSJ International Conference on Intelligent Robots and Systems, **EECON** Electrical Engineering Conference, **ECTI Transaction**, Asian Journal of Control, European Journal of Control, etc.

PROFESSIONAL EXPERIENCE

Thai Engineering & Business Co., Ltd., Bangkok, Thailand

Design/Sales Engineer **May, 1999 - January, 2000**

Designed and marketed ventilation devices for buildings and factories, for Matsushita Electric Industrial Co., Ltd. Primary liaison between Thai and Japan-based engineering teams. Responsible as a consultant and technical interpreter.

King Mongkut's University of Technology Thonburi, Bangkok, Thailand

Lecturer **November, 2006 - Present**

Teaching: grad/undergrad courses in control, circuits, and mathematics. Researching: learning and repetitive control, optimization, signal processing. Advising grad/undergrad students and managing grants.

I/U CRC Advanced Manufacturing, KMUTT, Bangkok, Thailand

Deputy Director **Febuary, 2008 - Present**

Administering and managing hard disk drive fundings for researchers in universities, collaborating with HDD industries for solving on-site problems.

Asian Institute of Technology, Pathumthani, Thailand

Adjunt Faculty **June, 2009 - July, 2009**

Co-teaching the course "**Robust and Adaptive Control**".

RESEARCH GRANTS **KMUTT** **July, 2007 - June, 2009**

The Advantages and Disadvantages of Kalman Filtering in Learning and Repetitive Control.

NSTDA **Jan, 2009 - Feb, 2010**

Parallel Parking System Based on Fuzzy Logic Control.

TRF **July, 2010 - June, 2012**

A Use of Kalman Filtering in Relinearizations of Nonlinear Systems in Iterative Learning Control System.

NRCT **October, 2010 - September, 2011**

A Design of Self-Balancing Wheelchair.

6.2 ผลงานตีพิมพ์จากงานวิจัย

- M. Q. Phan, R. W. Longman, B. Panomruttanarug, and S. C. Lee, "**Robustification of Iterative Learning Control and Repetitive Control by Averaging**," *International Journal of Control*, 2013, Vol. 86, Issue 5, **pp. 855-868.**

- B. Panomruttanarug, R. W. Longman, and M.Q. Phan, "**Designing Stable Iterative Learning Control Systems from Frequency Based Repetitive Control Designs**," *AAS/AIAA Astrodynamics Specialist Conference*, Alaska, USA, Aug, 2011.

6.3.เอกสารแนบหมายเลข 2 รูปแบบ Abstract (บทคัดย่อ)

รูปแบบ Abstract (บทคัดย่อ)

Project Code : MRG5380063
(รหัสโครงการ)

Project Title : การทำให้ระบบควบคุมแบบซ้ำไปซ้ำมามีเสถียรภาพมากขึ้นโดยการใช้ตัวควบคุมแบบเหมาะสม
(ชื่อโครงการ)

Investigator : ผศ.ดร. เบญจมาศ พนมรัตนรักษ์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี
(ชื่อนักวิจัย)

E-mail Address : benjamas.pan@kmutt.ac.th

Project Period : 3 ปี
(ระยะเวลาโครงการ)

วัตถุประสงค์: เพื่อออกแบบตัวควบคุมแบบเรียนรู้ที่มีการย้อนกลับไปสู่ภาวะเริ่มต้นให้มีประสิทธิภาพดีเหมาะสมกับระบบจริงที่จะถูกควบคุม

วิธีทดลอง: เริ่มจากการออกแบบตัวควบคุมควบคุมแบบเรียนรู้ที่มีการย้อนกลับไปสู่ภาวะเริ่มต้นจากตัวควบคุมแบบเรียนรู้ที่ทำแบบต่อเนื่องที่ออกแบบโดยใช้ข้อมูลเชิงความถี่จากระบบซึ่งเป็นตัวควบคุมที่มีประสิทธิภาพ พร้อมทดสอบดูว่าตัวควบคุมควบคุมแบบเรียนรู้ที่มีการย้อนกลับไปสู่ภาวะเริ่มต้นที่เกิดขึ้นนั้นมีประสิทธิภาพในการลดค่าความผิดพลาดลงได้หรือไม่ จากนั้นจะทำการปรับปรุงประสิทธิภาพของตัวควบคุมควบคุมแบบเรียนรู้ที่มีการย้อนกลับไปสู่ภาวะเริ่มต้นโดยการปรับค่าเกนซึ่งจะอาศัยการทำ optimization แบบ steepest descent method โดยดูทิศทางการ update ค่าเกนของเมตริกซ์การเรียนรู้จาก sensitivity function ซึ่งปรับค่าเกนนั้นสามารถปรับได้ 1 ค่า (มุมซ้ายบนของเมตริกซ์) หรือหลายๆค่าก็ได้ จากนั้นจึงนำไปทดสอบประสิทธิภาพของตัวควบคุมต่อไป
ต่อมามีการจำลองสถานการณ์ว่าถ้าสมการทางคณิตศาสตร์ของระบบที่จะนำมาใช้ในการออกแบบตัวควบคุมมีค่าตัวแปรไม่แน่นอน เราจะสามารถออกแบบตัวควบคุมควบคุมแบบเรียนรู้ที่มีการย้อนกลับไปสู่ภาวะเริ่มต้นที่มีประสิทธิภาพได้อย่างไร ซึ่งในงานวิจัยได้เสนอวิธีการหาค่าเฉลี่ยที่นำมาใช้ในการออกแบบตัวควบคุม เพื่อทดสอบดูว่าตัวควบคุมควบคุมแบบเรียนรู้ที่มีการย้อนกลับไปสู่ภาวะเริ่มต้นที่ออกแบบมาจากการหาค่าเฉลี่ยนั้นมีประสิทธิภาพดีกว่าตัวควบคุมควบคุมแบบเรียนรู้ที่มีการย้อนกลับไปสู่ภาวะเริ่มต้นที่ออกแบบไม่มีการหาค่าเฉลี่ยอย่างไร

ผลการทดลอง: จากผลการจำลองจะเห็นได้ว่าตัวควบคุมควบคุมแบบเรียนรู้ที่มีการย้อนกลับไปสู่ภาวะเริ่มต้นที่ออกแบบโดยการทำ optimization สามารถลดค่าความผิดพลาด (RMS error) ที่เกิดขึ้นจากการทำซ้ำได้ดีกว่าตัวควบคุมควบคุมแบบเรียนรู้ที่มีการย้อนกลับไปสู่ภาวะเริ่มต้นแบบที่ไม่มีการปรับค่าเกนใดๆ และในกรณีที่ระบบมีตัวแปรที่ไม่แน่นอนนั้น เราก็ยังสามารถออกแบบตัวควบคุมควบคุมแบบเรียนรู้ที่มีการย้อนกลับไปสู่ภาวะเริ่มต้นได้จากวิธีการหาค่าเฉลี่ย ซึ่งมีเสถียรภาพมากกว่าตัวควบคุมควบคุมแบบเรียนรู้ที่มีการย้อนกลับไปสู่ภาวะเริ่มต้นแบบไม่มีการหาค่าเฉลี่ย

สรุปและวิจารณ์ผลการทดลอง: ในงานวิจัยได้กล่าวถึงการออกแบบตัวควบคุมแบบเรียนรู้ที่มีการย้อนกลับไปสู่ภาวะเริ่มต้นและวิธีการทดสอบความมีเสถียรภาพของระบบ โดยมีเทคนิคการออกแบบตัวควบคุม 2 วิธี คือวิธีที่หนึ่งเป็นการนำเทคนิคการทำ optimization แบบ steepest descent method มาใช้โดยดูทิศทางการ update ค่าเกนของเมตริกซ์การเรียนรู้จาก sensitivity function ซึ่งปรับค่าเกนนั้นสามารถปรับได้ 1 ค่า (มุมซ้ายบนของเมตริกซ์) หรือหลายๆค่าก็ได้ แต่ผลการทดลองที่ได้ไม่แตกต่างกันมากนัก ซึ่งนักวิจัยแนะนำว่าปรับเกนเพียงแค่ 1 ค่าก็เพียงพอเพื่อลดความซับซ้อนจากการคำนวณ ซึ่งผลการทดลองได้นำเปรียบเทียบให้เห็นว่าตัวควบคุมที่ออกแบบจากการปรับค่าเกนเพียง 1 ตัว ได้ผลดีกว่าตัวควบคุมแบบที่ไม่มีการปรับค่าเกนเลย นอกจากนี้มีการศึกษาถึงกรณีที่เมื่อไม่ต้องมีการเรียนรู้ค่าความผิพลาดใน time step แรกๆ เราสามารถกำหนดให้มีการปรับค่าเกนในตำแหน่งแถวที่ 2 หรือ 3 แทนแถวแรกก็ได้ ซึ่งกำหนดได้จากค่า cutoff number หรือค่า c ส่วนการออกแบบตัวควบคุมอีกวิธีหนึ่งนั้นอาศัยหลักการหาค่าเฉลี่ยจากโมเดลที่จำลองสร้างขึ้นโดยให้มีค่าพารามิเตอร์ของระบบเปลี่ยนแปลงไป ซึ่งผลการทดลองแสดงให้เห็นว่าตัวควบคุมที่ออกแบบด้วยวิธีการหาค่าเฉลี่ยสามารถทำให้จำนวนระบบที่มีเสถียรภาพมีมากกว่าการใช้ตัวควบคุมที่ออกแบบมาจากระบบเพียงระบบเดียว ซึ่งในการทดลองได้ทดสอบกับตัวควบคุม 3 ประเภทคือ contraction mapping, partial isometry, และ quadratic cost ซึ่งผลของตัวควบคุมทั้ง 3 ประเภทก็ออกมาในลักษณะเดียวกัน

ข้อเสนอแนะสำหรับงานวิจัยในอนาคต: นำการออกแบบตัวควบคุมที่ได้มาทดสอบกับระบบควบคุมจริง เช่น หุ่นยนต์แขนกล เพื่อทดสอบประสิทธิภาพของตัวควบคุมว่าสามารถนำไปใช้งานได้อย่างมีประสิทธิภาพดังที่แสดงในผลการจำลองหรือไม่

Keywords : Iterative Learning Control, Repetitive Control, Robustification
(คำหลัก)

6.4. เอกสารแนบหมายเลข 3 Output จากโครงการวิจัยที่ได้รับทุนจาก สกว.

Output จากโครงการวิจัยที่ได้รับทุนจาก สกว.

1. ผลงานตีพิมพ์ในวารสารวิชาการนานาชาติรายละเอียดดังนี้
   M. Q. Phan, R. W. Longman, B. Panomruttanarug, and S. C. Lee, "**Robustification of Iterative Learning Control and Repetitive Control by Averaging,**" *International Journal of Control*, 2013, Vol. 86, Issue 5, pp. 855-868.

2. การนำผลงานวิจัยไปใช้ประโยชน์
   - เชิงสาธารณะ (มีเครือข่ายความร่วมมือ/สร้างกระแสความสนใจในวงกว้าง)
     ผลงานวิจัยจากโครงการวิจัยนี้ได้รับความสนใจจากนักวิจัยที่อยู่ในสาขาเดียวกันเป็นอย่างมาก ดังจะเห็นได้จากเมื่อมีการตีพิมพ์ผลงานไปแล้ว นักวิจัยได้รับเชิญจากมหาวิทยาลัย Southampton ประเทศ อังกฤษ ให้นำผลงานวิจัยไปแสดงพร้อมกับหารือเกี่ยวกับความร่วมมือที่จะทำในอนาคต ซึ่งจะนำผลงานวิจัยไปต่อยอดเพื่อให้มีการใช้งานกับเครื่องมือด้านการแพทย์ต่อไป
   - เชิงวิชาการ (มีการพัฒนาการเรียนการสอน/สร้างนักวิจัยใหม่)
     ผลงานวิจัยที่เกิดขึ้นถูกนำไปถ่ายทอดให้กับนักศึกษาระดับปริญญาโทได้เรียนรู้และนำไปประยุกต์ใช้กับงานทางด้านต่างๆ เช่นการควบคุมแขนกลหุ่นยนต์ในโรงงานอุตสาหกรรม เป็นต้น

3. อื่นๆ (เช่น ผลงานตีพิมพ์ในวารสารวิชาการในประเทศ การเสนอผลงานในที่ประชุมวิชาการ หนังสือ การจดสิทธิบัตร)
   - B. Panomruttanarug, R. W. Longman, and M.Q. Phan, "**Designing Stable Iterative Learning Control Systems from Frequency Based Repetitive Control Designs,**" *AAS/AIAA Astrodynamics Specialist Conference*, Alaska, USA, Aug, 2011.

International Journal of Control, 2013
Vol. 86, No. 5, 855–868, http://dx.doi.org/10.1080/00207179.2013.764576

Taylor & Francis
Taylor & Francis Group

Robustification of iterative learning control and repetitive control by averaging

Minh Q. Phan[a], Richard W. Longman[b], Benjamas Panomruttanarug[c,*] and Soo Cheol Lee[d]

[a]*Thayer School of Engineering, Dartmouth College, Hanover, NH 03755, USA;* [b]*Department of Mechanical Engineering, Columbia University, New York, NY 10027, USA;* [c]*Department of Control System and Instrumentation Engineering, King Mongkut's University of Technology Thonburi, Bangkok, Thailand 10140;* [d]*Department of Mechanical and Automotive Engineering, Daegu University, Daegu, Korea 750-712*

(*Received 11 February 2010; final version received 6 January 2013*)

This paper describes a recently developed averaging technique to robustify iterative learning and repetitive controllers. The robustified controllers are found by minimising cost functions that are averaged over either multiple analytical time-domain models or experimental frequency-domain data. The aim is to produce a technique that is simple and general, and can be applied to any iterative learning control (ILC) or repetitive control (RC) design that involves the minimisation of a cost function. Substantial improvement in convergence to zero tracking error in the presence of model uncertainties has been observed for both ILC and RC by this averaging technique.

Keywords: iterative learning control; repetitive control; robustification; averaging cost function

1. Introduction

Iterative learning control (ILC) starts with a fundamental recognition that repeated practice is a common mode of human learning. Given a goal (regulation, tracking, or optimisation), ILC refers to the mechanism by which the system improves its performance by repeated trials. As the task is performed repeatedly, the system ultimately compensates for any repeatable sources of error and disturbances without necessarily knowing what they are or where they come from. Learning control can be applied to systems that are designed to return to the same initial condition before each new execution of the task, as in the case of a robot performing a task on each item that arrives one by one on an assembly line. The same learning concept can also be applied to the situation where there is no resetting between periods such as turning or milling operations, in which case repetitive control (RC) is used. ILC and RC are thus most suitable for repetitive processes such as manufacturing, Owens (1977), Uchiyama (1978), Arimoto, Kawamura, and Miyazaki (1984), Rogers (1987), Moore (1993), Bien and Xu (1998), Frueh and Phan (2000), Phan, Longman, and Moore (2000), Elci, Longman, Phan, Juang, and Ugoletti (2002) and Dijkstra and Bosgra (2002). In many applications, tracking accuracy can be several orders of magnitude less than hardware repeatability. ILC and RC are perhaps the most cost-effective technologies that are capable of bringing tracking accuracy up to the repeatability level of the hardware.

In order to reach zero tracking error, the iterations made by RC or ILC must be convergent, or equivalently, they must make an asymptotically stable process. The aim is to converge to zero tracking error in hardware, but we must make use of a model in designing the control laws, and the model will not be perfect. If one can solve the inverse problem of finding the input needed by the model in order to make the model output track the desired trajectory, applying this input to the hardware will give results that are only as good as the model. Experimental experience on a robot showed that inversion of a model reduced the tracking error by a factor of 50. But when ILC was applied instead, the final error level was reduced by a factor of 1000 after 12 iterations Longman (2000) and Elci, Longman, Phan, Juang, and Ugoletti (1994b). ILC and RC designs are based upon a model, but they iterate with the real world behaviour. Therefore, to accomplish this additional improvement over inverting a model requires the ILC law or RC law to have substantial stability robustness to model error. Indeed, robust ILC has been considered in the literature, e.g. Ishihara, Abe, and Takeda (1992), Amman, Owens, Rogers, and Wahl (1996) and Tayebi and Zaremba (2000), and continues to be an important research topic. One of the approaches to handle parameter uncertainty in ILC makes use of linear matrix inequalities (LMIs). These approaches seek to guarantee stability for a specified range of parameters. Some examples of these papers are Xu, Sun, and Yu (2008), Galkowski et al. (2003), Galkowski, Paszke, Sulikowski, Rogers, and Owens (2002a) and Galkowski, Rogers, Xu, Lam, and Owens (2002b).

One can classify two types of robustness: robustness to high frequency un-modelled dynamics such as residual

*Corresponding author. Email: benjamas.pan@kmutt.ac.th

vibration modes, and robustness to parameter uncertainty in a model of the correct order. To create the first kind of robustness in ILC, batch zero-phase low-pass filtering is introduced in Elci, Phan, Longman, Juang, and Ugoletti (1994a), Elci et al. (1994b) and Plotnik and Longman (1999). For RC, the Q filter of Chew and Tomizuka (1990) serves this same purpose, and Panomruttanarug and Longman (2006) develop enhanced specialised real time FIR (finite impulse response) zero-phase filter cutoffs. Bao and Longman (2008) further develops such filters, making use of a transition zone and bounding the output amplitude response not to go above unity. These filters allow one to cut off the learning process above some frequency for which the model is no longer accurate enough to produce stability. One aims for zero error below the cutoff frequency, and sacrifices zero error above the cutoff in exchange for asymptotic stability in the presence of high frequency model error. Another strategy is through the use of basis functions as formulated in Frueh and Phan (2000).

Robust design in the control literature usually addresses robustness to parameter uncertainties, and aims to design a controller that is guaranteed to be stable for some specified range of parameters. The design process will fail if there does not exist a controller that can stabilise the full parameter range. The approach used here applies to both ILC and RC, and takes whatever distribution of uncertainty applies to the problem, and improves the robustness for this distribution, but does not attempt to guarantee stability for all models in the distribution, something that may not be possible, Takanishi, Phan, and Longman (2005), Lee, Phan, and Longman (2006), Brown, Phan, Lee, and Longman (2007) and Panomruttanarug, Longman, and Phan (2007). The simplicity of the approach makes it very attractive for practical applications. The present paper describes the main ideas and contributions of these four conference publications, and represents the first journal publication on this different kind of robustification. The basic approach is developed in Takanishi (2005). The uncertainties in the model are specified by their probabilistic distribution functions (uniform, Gaussian, etc.). The controller is designed not from these uncertainty distribution functions directly, but rather indirectly from a set of models whose parameters are generated from these distribution functions. Consequently, instead of minimising a single cost function associated with the nominal model, a cost function averaged over the uncertain models is minimised. Since the cost function is nonlinear, minimising the average cost is not the same as minimising the cost for the average or nominal model. The same mathematics used to minimise a single cost function can be easily extended to minimising an average cost function. Remarkably, this simple extension results in a controller that does very substantially improve the stability robustness. Although in theory it would take an impractical number of models to represent all possible uncertainty variations when the number of uncertain parameters is large, it has been observed that significant improvement in robustness can be achieved with a relatively small number of models in the design set. Being a probabilistic design approach, stability cannot be guaranteed for all models that obey the specified distribution functions. However, when taking into account the method's flexibility, simplicity, and the substantial improvement in robustness that it produces, one comes to the conclusion that the benefits it brings far outweigh the probabilistic nature of the solution. The robustness improvement can be quantified rigorously by stochastic stability analysis via Monte Carlo simulation, e.g. Brown (1956), Apostolakis (1990), Henley and Kumamoto (1992), Ray and Stengel (1992) and Phan and Maghami (1996), as done in Brown et al. (2007) for a robustified RC design with monotonic convergence. It is noted here that the averaging technique is general in that it can be applied to any other optimal controller design. Significant improvement in robustness has been observed when this design principle is applied to optimal model predictive controllers as shown in Barlow (2009).

This paper has two main parts. The first part presents a set of effective ILC and RC laws, and then formulates how one robustifies these laws by averaging cost functions. Each law is designed to produce monotonic decay of the Euclidean norm of the error from iteration to iteration in ILC, or to produce monotonic decay of each component of the frequency content of the error from period to period in RC. Each law has a parameter that adjusts the learning rate, with slower learning providing improved robustness, as is understood in the literature. We seek to develop robustness by averaging costs, improving the robustness for any chosen gain. The second part provides a number of numerical examples that illustrate the achieved robustification on a robot joint model.

2. Robustification of ILC

2.1. *Repetition domain formulation*

The following summarises the ILC formulation originally developed in Phan and Longman (1988) and later extended in Phan, Longman, and Moore (2000). Consider a system of the general form

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + v(k) \\ y(k) &= Cx(k), \end{aligned} \tag{1}$$

where A, B, C are the matrices describing the system dynamics. For simplicity, we consider the single-input single-output (SISO) case. Extension to the multiple-input multiple-output (MIMO) case is straightforward. The index k denotes the time $t = kT$, where T is an appropriate sampling interval. The vectors $x(k)$ and $y(k)$ denote the system state and output, respectively. The initial state $x(0)$ and the process disturbance $v(k)$ are the same from one repetition (or trial) to the next.

Define the following time history vectors:

$$\underline{y}_j = \begin{bmatrix} y_j(1) \\ y_j(2) \\ \vdots \\ y_j(p) \end{bmatrix} \quad \underline{y}^* = \begin{bmatrix} y^*(1) \\ y^*(2) \\ \vdots \\ y^*(p) \end{bmatrix} \tag{2}$$

$$\underline{u}_j = \begin{bmatrix} u_j(0) \\ u_j(1) \\ \vdots \\ u_j(p-1) \end{bmatrix} \quad \underline{w} = \begin{bmatrix} w(0) \\ w(1) \\ \vdots \\ w(p-1) \end{bmatrix}.$$

The $\underline{y}^*$ represents the desired output trajectory that is p-time-steps long. In the above definitions, we assume that the time delay through the system is a one-time step, i.e. the product CB is non-zero. For any repetition j, the relationship between an input time history and the resultant output time history is

$$\underline{y}_j = P\underline{u}_j + \underline{w}, \tag{3}$$

where

$$P = \begin{bmatrix} CB & & & & \\ CAB & CB & & & \\ CA^2B & CAB & \ddots & & \\ \vdots & \ddots & \ddots & CB & \\ CA^{p-1}B & \cdots & CA^2B & CAB & CB \end{bmatrix}. \tag{4}$$

This equation packages the convolution sum solution of (1) for all p-time-steps in the input-output histories of a run. The vector $\underline{w}$ incorporates the effect of the unknown initial state $x(0)$ and the unknown disturbance $v(k)$. The tracking error for repetition j is

$$\underline{e}_j = \underline{y}^* - \underline{y}_j. \tag{5}$$

Define a backward difference operator $\delta_j(.)$ applied to any variable $\underline{z}$ to be

$$\delta_j \underline{z} = \underline{z}_j - \underline{z}_{j-1}. \tag{6}$$

Thus, $\delta_j \underline{e} = -\delta_j \underline{y}$. Applying this operator to (3) yields

$$\underline{e}_j = \underline{e}_{j-1} - P\delta_j \underline{u}. \tag{7}$$

Notice that the unknown repeating initial condition and disturbance are eliminated by this operation. Iterative learning controllers designed from (7) will automatically compensate for any unknown repetitive disturbances regardless of any repeating initial condition. This equation forms the basis for the development of various ILC laws using modern state-space techniques.

2.2. *ILC convergence conditions*

A general (first-order) linear iterative learning control law has the form

$$\underline{u}_j = \underline{u}_{j-1} + L\underline{e}_{j-1}, \tag{8}$$

where L is a matrix of control gains chosen by the specific ILC design. Recognising that (8) can be written as $\delta_j \underline{u} = L\underline{e}_{j-1}$ the error propagation equation from one iteration to the next can be derived from (7) as

$$\underline{e}_j = (I - PL)\underline{e}_{j-1}, \tag{9}$$

where I is the identity matrix. One concludes from this that the tracking error at every time step of the p-time-step desired trajectory will go to zero as j tends to infinity, for all initial error histories, if and only if all eigenvalues of the matrix $(I - PL)$ are less than one in magnitude,

$$\max_i |\lambda_i (I - PL)| < 1. \tag{10}$$

Some ILC laws have poor transients during the convergence process even if (10) is satisfied. One can ensure that the Euclidean norm of the error decreases monotonically for every iteration if L satisfies the sufficient stability condition

$$\max_i \sigma_i (I - PL) < 1, \tag{11}$$

where σ_i is the ith singular value of $(I - PL)$, Longman (2000). Although (10) is the true stability condition, (11) is more practical.

2.3. *Quadratic cost ILC*

An optimal learning controller can be found to minimise a quadratic cost function with weighting matrices Q and R, Frueh and Phan (2000), Phan and Juang (1996) and Owens and Amann (1994)

$$J = \underline{e}_j^T Q \underline{e}_j + \delta_j^T \underline{u} R \delta_j \underline{u}. \tag{12}$$

Substituting (7) into (12), taking the derivative of J with respect to $\delta_j \underline{u}$, and setting the result to zero produces an ILC law of the form (8) with the optimal learning gain L,

$$L = (P^T Q P + R)^{-1} P^T Q. \tag{13}$$

Equation (13) above gives the optimal learning gain matrix derived from a given model of the system. For the multiple-model averaging design, we generate M different models A_i, B_i, C_i, $i = 1, 2, \ldots, M$ that are consistent with the specified distribution functions for the uncertain parameters. The averaging controller is one that minimises

the expected value of J over the range of uncertain parameters. When the number of samples is sufficiently large, the expected value of a quantity can be approximated by its average. Hence, the average cost function to minimise is:

$$E\{J\} \approx \frac{1}{M}\sum_{i=1}^{M} J_i \tag{14}$$

$$J_i = (\underline{e}_{j-1} - P_i\delta_j\underline{u})^T Q(\underline{e}_{j-1} - P_i\delta_j\underline{u}) + \delta_j^T\underline{u}R\delta_j\underline{u}, \tag{15}$$

for a sufficiently large M. Each P_i is built from the ith model. Minimising (14) with respect to $\delta_j\underline{u}$ produces the following learning gain,

$$L = \left(\sum_{i=1}^{M} P_i^T Q P_i + MR\right)^{-1}\sum_{i=1}^{M} P_i^T Q. \tag{16}$$

It is clear from this derivation that an ILC law based on the averaging technique can be derived as long as a cost function is specified.

2.4. *Contraction mapping ILC*

The Euclidean norm contraction mapping ILC law, Jang and Longman (1994), sets the ILC gain matrix L to be

$$L = \phi P^T, \tag{17}$$

where ϕ is a scaling factor. According to (9), the convergence of the tracking error is governed by

$$\underline{e}_j = \left(I - \phi P P^T\right)\underline{e}_{j-1}. \tag{18}$$

To see the motivation for this law, write the singular value decomposition of P as $P = USV^T$ where U and V are unitary matrices, and $S = diag(\sigma_1, \sigma_2, \ldots, \sigma_p)$ is a diagonal matrix of all singular values of P. From (18)

$$U^T\underline{e}_j = \begin{bmatrix} 1-\phi\sigma_1^2 & & & \\ & 1-\phi\sigma_2^2 & & \\ & & \ddots & \\ & & & 1-\phi\sigma_p^2 \end{bmatrix} U^T\underline{e}_{j-1}. \tag{19}$$

The Euclidean norm of $U^T\underline{e}_j$ is the same as the Euclidean norm of $\underline{e}_j$, every element of $U^T\underline{e}_j$ will decrease in magnitude every iteration provided that $|1 - \phi\sigma_i^2| < 1$, or equivalently,

$$0 < \phi < 2/\max(\sigma_i^2). \tag{20}$$

To generalise the contraction mapping ILC law using the averaging technique, we need a cost function that produces the learning gain given in (17). This learning gain can be derived from a cost function that is a sum of the squares of all future errors,

$$J = \frac{1}{2}\underline{e}^T\underline{e} = \frac{1}{2}\left[e^2(1) + \cdots + e^2(p)\right]. \tag{21}$$

At the end of each repetition one has an input history $\underline{u}$ for that run, and the resulting error history $\underline{e}$. The derivative of the scalar cost with respect to changes in the input $u(k)$ at time step k is

$$\frac{\partial J}{\partial u(k)} = \left(\frac{\partial J}{\partial \underline{e}}\right)^T\left(\frac{\partial \underline{e}}{\partial u(k)}\right) = \underline{e}^T\left(\frac{\partial \underline{e}}{\partial u(k)}\right) = -(P_k)^T\underline{e}, \tag{22}$$

where we have used the derivative $\partial\underline{e}/\partial u(k) = -P_k$. The subscript k on P denotes the kth column of matrix P. Then taking a step of length ϕ in the steepest descent direction in the next repetition creates the learning law

$$\underline{u}_{j+1} = \underline{u}_j - \phi\left.\frac{\partial J}{\partial \underline{u}}\right|_j = \underline{u}_j + \phi P^T\underline{e}_j, \tag{23}$$

hence the contraction mapping learning gain $L = \phi P^T$ is justified. Taking the average of the cost (21) over different models, and taking a step in the steepest descent direction for this average cost produces the following averaged contraction mapping ILC gain

$$L = \frac{1}{M}\sum_{i=1}^{M} L_i = \frac{\phi}{M}\sum_{i=1}^{M} P_i^T, \tag{24}$$

where $L_i = \phi P_i^T$, $i = 1, 2, \ldots, M$.

2.5. *Partial isometry ILC*

Some of the singular values of P can be small. As a result the error components on the associated singular vectors in U will learn monotonically but slowly. For a sufficiently large p, the singular values of P can be associated with the magnitude frequency response of the system, and this response gets small at high frequencies as in Chen and Longman (2003). There can also be particularly small singular values that are introduced by the conversion of the ordinary differential equation to a discrete-time difference equation. Instead of using $L = \phi P^T = \phi V S U^T$, the partial isometry law, Jang and Longman (1996) uses

$$L = \phi V U^T. \tag{25}$$

The entries in the diagonal matrix of (19) become $1 - \phi\sigma_i$, and convergence is assured if the step length satisfies

$0 < \phi < 2/\max(\sigma_i)$. This law learns substantially faster at high frequencies. One could, of course, aim to speed up the learning process by picking $L = \phi V S^{-1} U^T$, but this is simply creating the inverse of matrix P. This matrix is normally ill-conditioned, and the inverse is very sensitive to model errors. Using this inverse solution is therefore not advisable in practice.

Again, to derive the iterative learning controller associated with (25) by averaging, we need to find a cost function to produce it. Indeed, (25) can be found by considering the following cost function

$$J = \frac{1}{2} \underline{e}^T \left(U S^{-1} U^T \right) \underline{e} \tag{26}$$

and taking the steepest decent direction as in the contraction mapping law. Taking the average of the cost function in (26) produces the following averaged partial isometry ILC gain

$$L = \frac{1}{M} \sum_{i=1}^{M} L_i = \frac{\phi}{M} \sum_{i=1}^{M} V_i U_i^T, \tag{27}$$

where $L_i = \phi V_i U_i^T$, $i = 1, 2, \ldots, M$.

3. Robustification of RC

3.1. RC convergence condition

The following describes a convergence condition for RC as described in Panomruttanarug and Longman (2004) which later motivates a robustification design. Let p denote the number of time steps in one period. A general linear repetitive control law takes the form,

$$\begin{aligned} u(k) = u(k-p) &+ a_1 e(k-p+m-1) \\ &+ \cdots + a_m e(k-p) + a_{m+1} e(k-p-1) \\ &+ \cdots + a_n e(k-n+m-p). \end{aligned} \tag{28}$$

In (28) there are a total of n scalar gains for a SISO system. The error $e(k-p+m-1)$ corresponds to the most recent past tracking error associated with the gain a_1, and $e(k-n+m-p)$ corresponds to the most distant past tracking error associated with the gain a_n. The control law in (28) can also be written in the z-domain as

$$U(z) = \left[\frac{F(z)}{z^p - 1} \right] E(z), \tag{29}$$

where

$$\begin{aligned} F(z) = a_1 z^{m-1} &+ \cdots a_{m-1} z + a_m \\ &+ a_{m+1} z^{-1} + \cdots + a_n z^{-(n-m)}. \end{aligned} \tag{30}$$

The repetitive controller gains $a_1, a_2, \ldots, a_n$ are to be designed so that the tracking error converges to zero as the number of periods tends to infinity. Furthermore, for practical reasons, we desire the convergence of the tracking error from period to period to be monotonic. To this end, from the block diagram of Figure 1, we can write

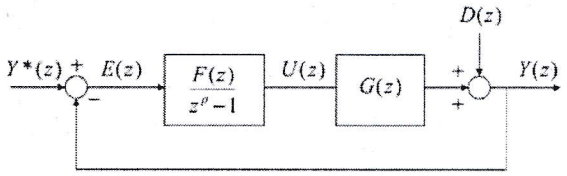

Figure 1. Repetitive control block diagram.

$$\begin{aligned} &\{1 - z^{-p}[1 - G(z)F(z)]\} E(z) \\ &\quad = (1 - z^{-p}) [Y^*(z) - D(z)], \end{aligned} \tag{31}$$

where $G(z)$ is the transfer function of the system being controlled, $G(z) = C(zI - A)^{-1} B$. In (1), $v(k)$ is an input disturbance. Here $d(k)$ is the equivalent output disturbance, and the desired trajectory is $y^*(k)$, both of which are periodic with a period of p- time-steps, hence $(1 - z^{-p})[Y^*(z) - D(z)] = 0$. It follows from (31) that

$$z^p E(z) = [1 - G(z)F(z)] E(z). \tag{32}$$

Suppose that the learning is sufficiently slow compared to the period of p-time-steps that one can consider each period to be in a steady state. Now consider the frequency transfer function version of $1 - G(z)F(z)$ formed by substituting $z = e^{i\omega T}$, where T is the sampling interval. The magnitude of such a frequency transfer function for frequency ω gives the amplitude change from the amplitude of a sinusoidal input of frequency ω to the amplitude of the steady-state sinusoidal output. Under the quasi-steady state assumption above, this magnitude can be interpreted as relating to the frequency components of the error in one period to the frequency components of the error in the following period. Then, having this magnitude less than unity for all frequencies between zero and Nyquist frequency suggests monotonic convergence of all frequency components of the error. Thus, we require

$$|1 - G(z)F(z)| < 1, \qquad z = e^{i\omega T} \tag{33}$$

for all frequencies ω up to Nyquist frequency, $0 \leq \omega_i T \leq \pi$. For a given repetitive controller $F(z)$, $1 - G(e^{i\omega T})F(e^{i\omega T})$ can be plotted on the complex plane as ωT varies between 0 and π. We desire that this polar plot remains inside the unit circle in the complex plane to satisfy the approximate monotonic decay condition of (33) for all frequency components of the tracking error. Note that the argument that generates (33) suggests that any frequency component of the error associated with a frequency outside the unit circle would grow monotonically, corresponding

to an unstable system. Condition (33) is a sufficient condition for stability, but it can be shown that for reasonable values of p, the difference between this condition and the actual stability boundary is so small as to be of no practical significance, as in Songchon and Longman (2003).

3.2. *RC design from model*

Note that a choice for $F(z)$ is $G^{-1}(z)$. This choice is problematic because the inverse of a discrete-time z-transfer function is usually unstable. Let Φ denote the repetitive controller gain vector to be designed,

$$\Phi = [a_1 \;\; a_2 \;\; \ldots \;\; a_{m-1} \;\; a_m \;\; a_{m+1} \;\; \ldots \;\; a_n]^T \tag{34}$$

Then $F(z)$ can be written as

$$F(z) = M(z)\Phi, \tag{35}$$

where $M(z)$ (not to be confused with M, the number of different models) is

$$M(z) = \left[z^{m-1} \;\; z^{m-2} \ldots \; z \;\; 1 \;\; z^{-1} \ldots z^{-(n-m)}\right]. \tag{36}$$

It is established in the previous section that we desire the polar plot of $1 - G(e^{i\omega T})F(e^{i\omega T})$ to remain inside the unit circle for monotonic convergence of the tracking error. Hence, the repetitive controller gains can be found by minimising the shape of this polar plot. This can be accomplished by minimising the cost function Panomruttanarug and Longman (2004)

$$\begin{aligned} J = \sum_{i=0}^{N-1} & W_i \left[1 - G(z_i)M(z_i)\Phi\right] \\ & \times \left[1 - G(z_i)M(z_i)\Phi\right]^* + v\Phi^T\Phi, \end{aligned} \tag{37}$$

where v is a scalar weighting factor, the star * denotes the complex conjugate operation, $z_i = e^{j\omega_i T}$, $i = 0, 1, 2, \ldots, N-1$ where N is the number of discrete points that make up the polar plot, each corresponds to a discrete frequency $\omega_i, 0 \le \omega_i T \le \pi$. Taking the derivative of J with respect to the gain vector Φ produces the RC gains,

$$\Phi = \mathbf{A}^{-1}\mathbf{B}, \tag{38}$$

where

$$\mathbf{A} = \sum_{i=0}^{N-1} W_i \left[\mathrm{Re}\left(Q(z_i)\right) + \mathrm{Re}\left(Q(z_i)\right)^T\right] + 2vI \tag{39}$$

$$\mathbf{B} = \sum_{i=0}^{N-1} W_i \left[\mathrm{Re}\left(S^*(z_i)\right) + \mathrm{Re}\left(S(z_i)\right)^T\right]. \tag{40}$$

In (39), Re(.) denotes the real part of the quantity (.), and

$$Q(z_i) = S^*(z_i)S(z_i) \qquad S(z_i) = G(z_i)M(z_i). \tag{41}$$

In (39) and (40), the superscript T denotes the regular (real) transpose, I is an identity matrix, and the superscript * denotes the complex conjugate transpose. The MIMO version of this design technique is presented in Brown et al. (2007).

3.3. *RC design from frequency response data*

Another version of the above design can be formulated to make use of frequency response data directly instead of computing $G(z)$ from a model like in Panomruttanarug and Longman (2007). From experimental data, let $|G(\omega)|$ and $\varphi(\omega)$ denote the magnitude and phase of the frequency response,

$$G(\exp(j\omega_i T)) = |G(\omega_i)| \exp\left(\varphi(\omega_i)\right) \tag{42}$$

Then the RC gains can be found from (38)

$$\mathbf{A} = \sum_{i=0}^{N-1} W_i \left|G(\omega_i)\right|^2 \left[Z_1 \;\; Z_2 \;\; \ldots \;\; Z_n\right] \tag{43}$$

$$\mathbf{B} = \sum_{i=0}^{N-1} W_i \left|G(\omega_i)\right| Z_m, \tag{44}$$

where

$$Z_1 = \begin{bmatrix} 1 + v \\ \cos(\omega_i T) \\ \vdots \\ \cos((n-1)\omega_i T) \end{bmatrix}, \tag{45}$$

$$Z_2 = \begin{bmatrix} \cos(\omega_i T) \\ 1 + v \\ \vdots \\ \cos((n-2)\omega_i T) \end{bmatrix}, \tag{46}$$

$$Z_n = \begin{bmatrix} \cos((n-1)\omega_i T) \\ \cos((n-2)\omega_i T) \\ \vdots \\ 1 + v \end{bmatrix}, \tag{47}$$

$$Z_m = \begin{bmatrix} \cos((m-1)\omega_i T + \varphi(\omega_i)) \\ \cos((m-2)\omega_i T + \varphi(\omega_i)) \\ \vdots \\ \cos((m-N)\omega_i T + \varphi(\omega_i)) \end{bmatrix}. \tag{48}$$

3.4. *Robustified RC design*

Now consider the case where the system model contains a number of uncertain parameters. For the averaging design,

we generate M different models $G_m(z)$, $m = 1, 2, \ldots, M$, and minimise the average cost function

$$E\{J\} \approx \frac{1}{M}\sum_{m=1}^{M} J_m, \tag{49}$$

$$J_m = \sum_{i=0}^{N-1} W_i\,[1 - G_m(z_i)M(z_i)\Phi] \times [1 - G_m(z_i)M(z_i)\Phi]^* + v\Phi^T\Phi. \tag{50}$$

Note that in this case averaging the cost for all models to produce a controller is different than finding the controllers for all models and averaging them. Minimising this average cost produces the following RC gains computed as in (38) but $\mathbf{A}$, $\mathbf{B}$ are now given as

$$\mathbf{A} = \sum_{m=1}^{M} \mathbf{A}_m + 2MvI, \tag{51}$$

$$\mathbf{B} = \sum_{m=1}^{M} \mathbf{B}_m, \tag{52}$$

$$\mathbf{A}_m = \sum_{i=0}^{N-1} W_i \left[\mathrm{Re}\left(Q_m(z_i)\right) + \mathrm{Re}\left(Q_m(z_i)\right)^T\right], \tag{53}$$

$$\mathbf{B}_m = \sum_{i=0}^{N-1} W_i \left[\mathrm{Re}\left(S_m^*(z_i)\right) + \mathrm{Re}\left(S_m(z_i)\right)^T\right], \tag{54}$$

$$Q_m(z_i) = S_m^*(z_i)S_m(z_i), \tag{55}$$

$$S_m(z_i) = G_m(z_i)M(z_i). \tag{56}$$

The previous averaging is designed for parametric uncertainties. To account for high-frequency un-modelled dynamics, we can design the averaged RC based on frequency response data directly. The counterparts of (51) and (52) are

$$\mathbf{A} = \sum_{m=1}^{M}\sum_{i=0}^{N-1} W_i\,|G_m(\omega_i)|^2\,[Z_1 \;\; Z_2 \;\ldots\; Z_n], \tag{57}$$

$$\mathbf{B} = \sum_{m=1}^{M}\sum_{i=0}^{N-1} W_i\,|G_m(\omega_i)|Z_m, \tag{58}$$

where $G_m(\omega)$ represents the frequency response of the mth experiment, as in Panomruttanarug et al. (2007).

4. Robustified ILC illustrations

The following illustration is performed on a model motivated by the dynamics of each link of the 7-degree-of-freedom robot (Robotics Research Corp. K-series 807iHP manipulator) shown in Figure 2. The robot has a maximum workspace radius of 0.89 m with a maximum payload of 20 lbs. The closed loop control of each link has a bandwidth

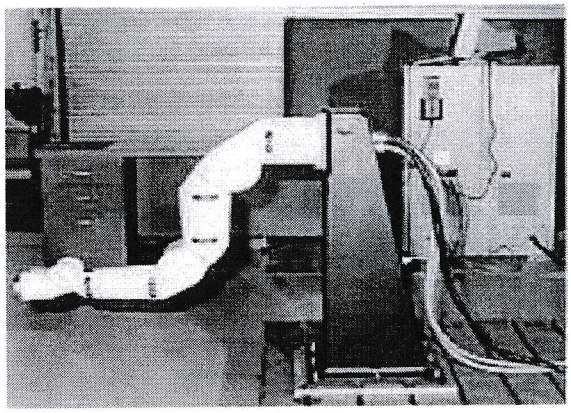

Figure 2. A seven degree-of-freedom robot.

determined by a first-order pole at 1.4 Hz, and exhibits two resonant frequencies. For purposes of illustration, the parameters are modified, and in particular the damping ratios for each mode are taken to be 0.1 which is much lower than the actual damping in the system. This makes the robustification of ILC and RC more difficult. Hence, we take the nominal model to be

$$G_n(s) = G_a(s)G_b(s)G_c(s), \tag{59}$$

where

$$G_a(s) = \left(\frac{a}{s+a}\right) \quad G_b(s) = \left(\frac{\omega_1^2}{s^2 + 2\zeta_1\omega_1 s + \omega_1^2}\right), \tag{60}$$

$$G_c(s) = \left(\frac{\omega_2^2}{s^2 + 2\zeta_2\omega_2 s + \omega_2^2}\right). \tag{61}$$

The model is fed by a zero-order-hold and we choose a sampling frequency of 100 Hz. The nominal values of the parameters are $a = 8.8$, $\omega_1 = 6(2\pi)$, $\omega_2 = 18(2\pi)$, $\zeta_1 = \zeta_2 = 0.1$. We seek to robustify the ILC laws to parametric variations obeying uniform distributions around these nominal values,

$$\begin{gathered} 6.613 < a < 10.977 \\ 27.762 < \omega_1 < 46.234,\;\; 85.022 < \omega_2 < 141.114 \\ 0.0751 < \zeta_1 < 0.1243,\;\; 0.0751 < \zeta_2 < 0.1243. \end{gathered} \tag{62}$$

Each of the ILC laws has the property that the learning gets slow at high frequencies, and this means that the spectral radius, or magnitude of the largest eigenvalue of $I - PL$, will get close to unity in the stability criterion (10). Numerical round off in the computations can easily make the spectral radius go slightly above unity so that one may not be able to make a definitive judgment of whether an ILC system is stable. In other words, numerical simulations of the learning process are made to help establish whether a candidate controller is stable, but it is still possible that an

Table 1. Quadratic cost ILC averaging results.

	Max. Eigenvalues			Max. Singular Values		
Quadratic Cost ILC	>1	>1.001	>1.01	>1	>1.001	>1.01
Nominal	159	21	19	199	42	40
Average	136	0	0	200	9	5
Average with $\phi = 0.5$	154	0	0	200	2	0
Average with $\phi = 0.1$	142	0	0	199	0	0
200 different models: 1^{st} run						
Nominal	160	35	31	200	53	49
Average	156	0	0	200	9	5
200 different models: 2^{nd} run						
Nominal	156	16	16	200	44	43
Average	156	0	0	200	5	4
200 different models: 3^{rd} run						
Nominal	152	27	22	199	41	43
Average	150	0	0	200	8	5

instability is sufficiently small that it is not seen in simulations for a specific initial error when a chosen number of ILC iterations is performed. Here we examine three different threshold levels for the spectral radius, all eigenvalues $|\lambda_i| < 1$, $|\lambda_i| < 1.001$, and $|\lambda_i| < 1.01$ because numerical simulations suggest stability for systems with spectral radius less than 1.001 and instability for systems with larger spectral radius. The monotonic decay condition (11) wants the maximum singular value less than unity. Therefore, we investigate this condition with the same three threshold levels as well. Note that even if the eigenvalue or singular value is truly above unity but less than 1.001, the ILC can be very effective at reducing the tracking error in applications, as in Longman and Huang (1994).

Table 1 studies the robustification using the quadratic cost ILC law. Note that the learning rate at any given frequency for this law decays as the square of the magnitude frequency response of the system, so that if the system response is down at some frequency by a factor of 10, the learning rate is down by a factor of 100. This suggests that the spectral radius should get very close to unity. Two hundred models are created by picking random coefficients from their uniform distributions in (62). The first row labelled 'nominal' corresponds to a controller designed from the nominal model described above and applied to the 200 models. The weighting matrices Q and R are set to be identity matrices. We find that 159 out of the 200 models have a spectral radius above 1, 21 above 1.001, and 19 above 1.01. The second row of Table 1 uses an averaged cost design, averaging over the 200 models. This time 136 models have a spectral radius above 1 but no model has a spectral radius above 1.001. Hence the averaging has made a very substantial improvement. These results use a scaling factor equal to unity (full gain). The third row of the table scales the gain down to 0.5, and the next row scales it down further to 0.1.

When the computed spectral radius is above unity, but less than 1.001, it is ambiguous whether the system is unstable, or numerical roundoff in the computation is responsible for the spectral radius exceeding unity. However, even if the system is unstable, the instability is most likely slow enough that the law is still very useful in practice, as discussed in Longman and Huang (1994) because the error may easily reach a very small value before any indication of the instability starts to appear. This means one can apply the law and turn off the learning when the error goes through a minimum. The experiments on the robot in Elci et al. (1994a) achieved an improvement in tracking error by a factor of 50 before any error growth was observed from high- frequency phase error.

The results cited so far use 200 models from the sample distribution, and examine how well the controller averaged over these models work on these same models. A better test of robustification is to create a controller by averaging the cost over these 200 models, but apply it to a different sample of 200 models taken from the distribution. This is done for three different random samples of 200 models in the lower part of the table. The observed robustification by using average cost (14) persists with no models having a spectral radius larger than 1.001 in any of the three random samples of 200 models. Table 2 repeats the same process for the contraction mapping ILC, and Table 3 for the partial isometry ILC. Overall, it is clear that the averaging technique results in a significant robustification for all ILC designs.

5. Robustified RC illustrations

For RC we have two cases to illustrate. The first case uses averaging to robustify a repetitive controller designed directly from experimental frequency response data and applied to a nominal model. The second case shows how the averaged controller performs on variations of the nominal model. Consider the same $5th$ order system in the previous section. The repetitive controller (28) has $n = 20$, $m = 11$.

Table 2. Contraction mapping ILC averaging results.

Euclidean Norm Contraction Mapping Law	Max. Eigenvalues			Max. Singular Values		
	>1	>1.001	>1.01	>1	>1.001	>1.01
Nominal	178	81	80	199	92	92
Average	155	0	0	200	13	12
Average with $\phi = 0.5$	139	0	0	200	5	2
Average with $\phi = 0.1$	148	0	0	200	0	0
200 different models: 1^{st} run						
Nominal	173	89	87	200	99	98
Average	148	0	0	199	22	21
200 different models: 2^{nd} run						
Nominal	169	80	79	200	93	98
Average	157	0	0	200	10	9
200 different models: 3^{rd} run						
Nominal	163	73	71	200	87	86
Average	152	0	0	200	19	17

Table 3. Partial isometry ILC averaging results.

Partial Isometry ILC	Max. Eigenvalues			Max. Singular Values		
	>1	>1.001	>1.01	>1	>1.001	>1.01
Nominal	173	79	64	200	99	83
Average	158	1	1	200	42	33
Average with $\phi = 0.5$	159	0	0	200	15	9
Average with $\phi = 0.217$	151	0	0	200	6	2
Average with $\phi = 0.1$	155	0	0	200	2	0
200 different models: 1^{st} run						
Nominal	171	78	66	200	99	87
Average	147	7	7	200	55	50
200 different models: 2^{nd} run						
Nominal	171	76	62	200	94	80
Average	155	5	4	200	44	36
200 different models: 3^{rd} run						
Nominal	177	70	57	200	87	75
Average	146	4	4	200	39	37

If we know the model perfectly we can make a very effective repetitive control design. In practice we do not know the model perfectly but we can make experiments (say 10 in this case) to compute the frequency response of the system. Because of noise and leakage in identification, these frequency responses will all be different. The conventional approach is to take the average of the 10 identified frequency responses to produce a single frequency response, and then design a controller from this single average model. Instead, in the robustified design, the controller is derived by minimising a cost that is averaged over the 10 experimental frequency responses.

Figure 3 shows the performance of the controller designed from the average of the 10 frequency responses when applied to the actual (nominal) model. The vertical axis of Figure 3 is the left hand side (LHS) of (33) with $G(z)$ being the actual 5-th order model. Above about 35 Hz the plot exceeds unity when $v = 0$, and at least for large enough periods this ensures instability of the RC system and growth of error components above this frequency with successive periods. Increasing v quickly decreases the learning rate and voids instability at higher frequencies. Now consider designing a repetitive controller that minimizes the cost that

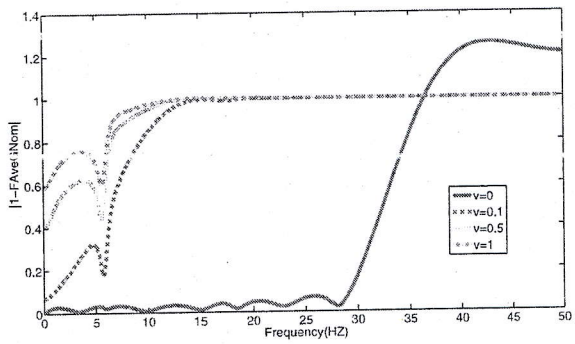

Figure 3. LHS of (33) for RC designed from the frequency response average.

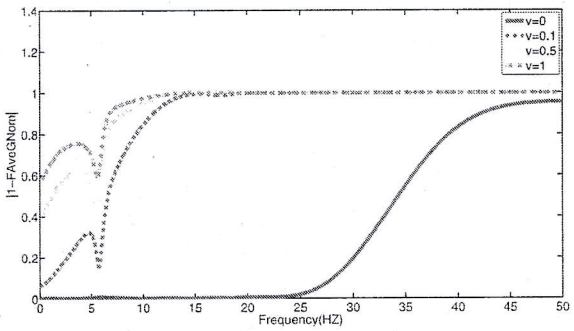

Figure 4. LHS of (33) for RC designed from the average cost function.

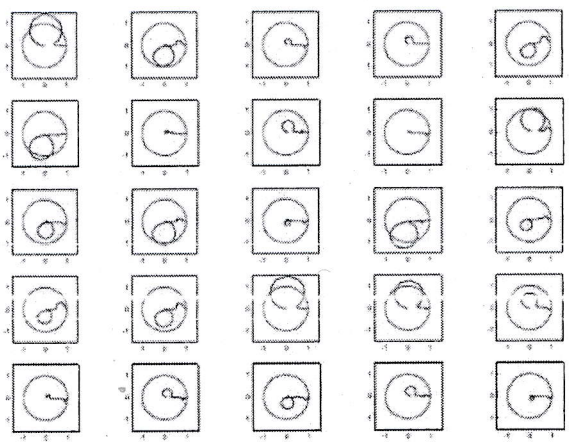

Figure 5. Magnitude and phase of $1 - G(z)F(z)$ for RC designed from the nominal model and applied to models 1–25.

is averaged over all 10 frequency responses. The results when applied to the actual model are shown in Figure 4. Notice now that the magnitude plots have spectral radius less than unity for all frequencies even when $v = 0$. Hence, by the averaging technique one succeeds in getting an asymptotically stable repetitive control system despite not having any model that is accurate at high frequencies.

Finally, we present here the performance of the robustified repetitive controller when it is applied to different models due to parametric variations. The same system is used for the simulation with similar variations in the coefficients. Figure 5 shows the magnitude and phase of $1 - G(e^{i\omega T})F(e^{i\omega T})$ for a repetitive controller designed from the nominal model, but applied to 25 variations of the nominal model that are not used in the design set. The unit circles are shown in the plots. Figure 5 is to be compared directly to Figure 6 which plots the same quantity for a repetitive controller which is designed from the average cost and applied to the same 25 models. The performance of the averaged design can be assessed by comparing each subplot of Figure 5 to the corresponding subplot of Figure 6.

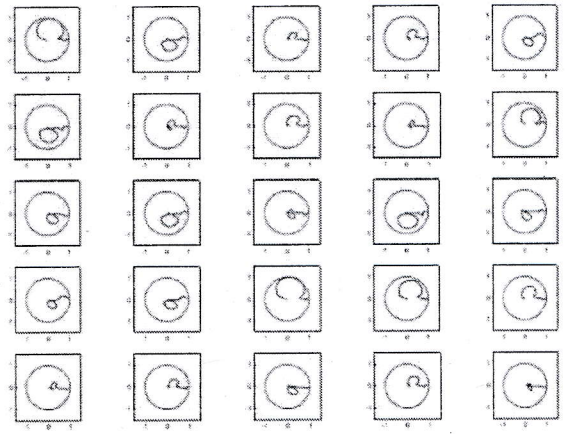

Figure 6. Magnitude and phase of $1 - G(z)F(z)$ for RC designed from the average cost and applied to the same 25 models.

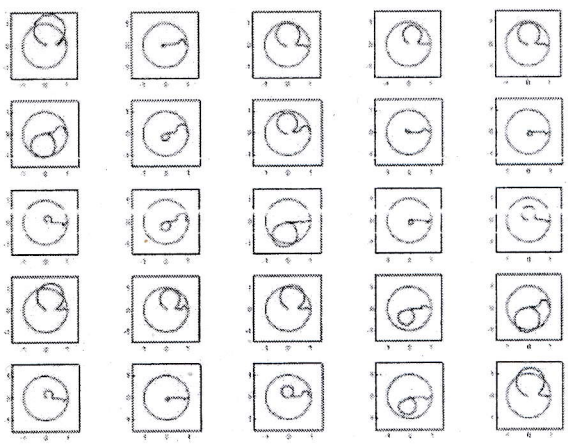

Figure 7. Magnitude and phase of $1 - G(z)F(z)$ for RC designed from the nominal model and applied to another 25 models.

The improvement in the stability robustness can be clearly observed. A total of 100 variations from the nominal model are checked, and another 25 models are shown in Figures 7 and 8. Again, the same robustification effect is observed.

6. Understanding the mechanism of robustification

One can gain some understanding of the mechanism responsible for the robustification by examining the averaging as seen in the frequency domain. Consider the RC design obtained by use of the cost function (37). It aims to pick the RC law (35) to minimise the square of the left-hand side of (33) summed over frequencies from zero to Nyquist, so that the $F(z)$ aims to match the inverse of the magnitude frequency response, and to produce a phase that is the negative of the steady state phase frequency response of the system. Then when averaging is introduced, it tries to minimise this summed over all models. If the minimisation succeeds in

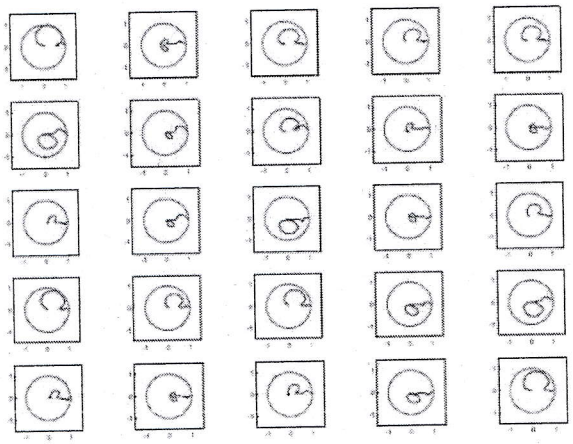

Figure 8. Magnitude and phase of $1 - G(z)F(z)$ for RC designed from the average cost and applied to the same 25 models.

keeping the left-hand side of (33) less than unity at all frequencies for a model, then the RC is stable for that model.

Note that (33) can be thought of as making a circular region in the complex plane representation of the complex numbers $G(z)F(z)$, and that a model is stable if this frequency response representation for each frequency from zero to Nyquist stays inside the circle of unit radius centred at +1. From this, we can conclude that the amount of phase error in the $F(z)$ attempt to match the negative of the phase of $G(z)$, is limited by + or −90 degrees (at least for large enough p so that condition (33) can be considered the stability boundary, as in Songchon and Longman (2003)). If one inserts an overall gain into $F(z)$, then one can approach this limit in phase error as the gain tends to zero. The magnitude error is much less important in the determination of stability.

Consider the same system (59) with the same uniformly distributed uncertainties given in (62), with a 100 Hz sample rate, and $n = 200$ and $m = 101$ in (28). And use 200 models randomly selected from these distributions. Figures 9 and 10 present the magnitude and phase frequency response curves of the models. In order to make the plot intelligible, only 21 of the 200 models are shown. The full frequency response ranges from zero to 50 Hz at Nyquist frequency. For clarity of presentation the figure presents the plots over two frequency intervals showing the behaviours around the two resonant frequencies of the system. The circles represent the frequency response of the nominal model. If one were not robustifying, the $F(z)$ would be designed to match the curve represented by these circles, match in the sense that the magnitude would aim for the reciprocal and the phase would aim for the negative of the phase shown. The design of $F(z)$ when the cost function averages over the 200 modes, instead aims in this same manner to the curve represented by the squares. The squares actually are the points aimed for, but the fit is sufficiently good so that one

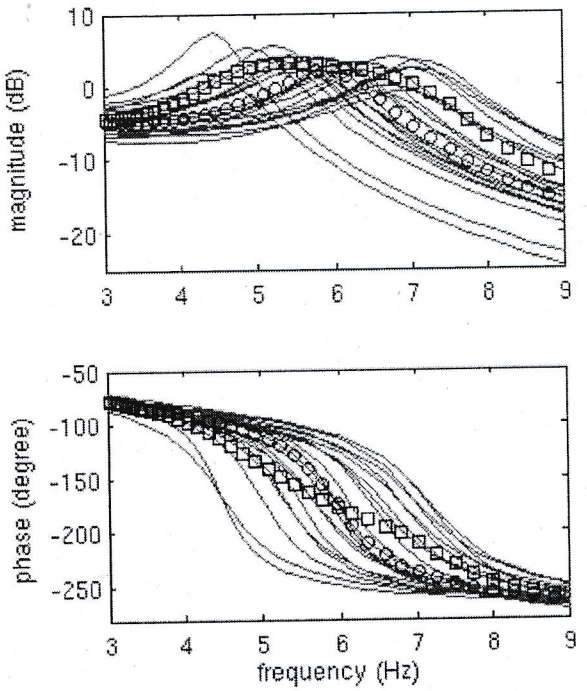

Figure 9. Magnitude and phase Bode plots in the frequency interval 1–9 Hz of 21 sample models, for the nominal model RC design (circles), and for the averaged cost design (squares).

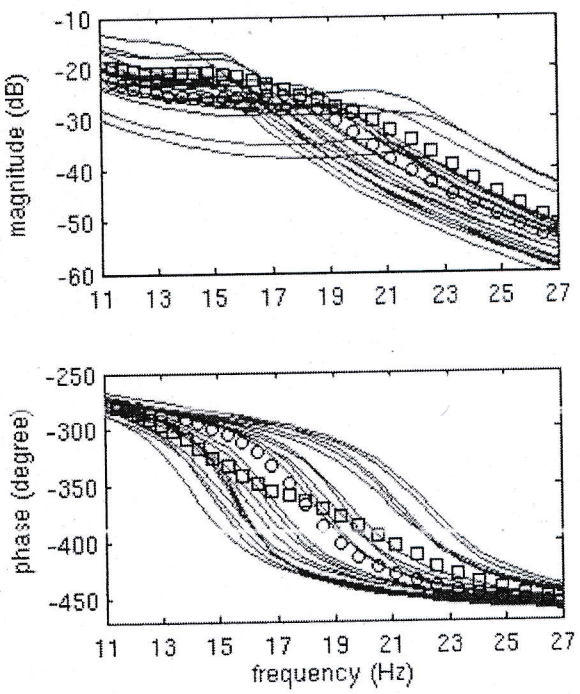

Figure 10. Magnitude and phase Bode plots in the frequency interval 11–27 Hz of 21 sample models, for the nominal model RC design (circles), and for the averaged cost design (squares).

does not see any discrepancy between the desired and the actual designs when the latter is also plotted. Examining the squares in the phase plots for each resonant frequency region, it is clear that the design is keeping the phase of the design much nearer the centre of the range of phases for the set of models. The research that followed considers the direct use of this thinking in producing robustification. Any model among the set of models considered, having a phase difference from the new design that is less than + or −90 degrees can be stabilised by the averaging process with a small enough overall gain. Hence, it is clear that the resulting design stabilises models in the distribution that would not be stable when one designs based on the nominal model. We note that the averaging method used here does not guarantee that all models in the distribution will be stabilised. But we also note that no RC law can stabilise a set of models that have a range of phase larger than the maximum possible tolerance of 180 degrees, and hence design methods that aim to guarantee robustness for a given set of models will fail in such cases. The method presented here is very simple to use, and will stabilise the RC for a wide range of model errors. Hence, we recommend that it be used whenever one makes an ILC or RC design.

The same frequency response thinking can be applied to those parts of ILC problems that can be considered modelled by steady state frequency response thinking, for example the part of the trajectory after a few time constants for transient decay. Note that as the trajectory gets longer, the ILC law (17) makes an error update formula (18) which is converted to frequency response, be one minus ϕ times the product of the magnitude frequency response, and the difference of the phase angle frequency response, of the true system and the model. The ILC law (27) is the same except that the magnitude frequency response of the model is removed. The ILC law (13) produces the negative of the phase, but alters the magnitude differently. Hence, an averaging of these laws creates similar interpretations as in the RC problem above.

7. Conclusions

This paper describes a simple yet highly effective technique to improve the stability robustness of iterative learning and repetitive control laws. The method minimises a cost function that is averaged over either multiple analytical time-domain models or experimental frequency-domain data. For ILC we consider three design methods: quadratic cost, contraction mapping, and partial isometry. For sufficiently long trajectories, these laws also result in a monotonic decay of every frequency component of the tracking error from repetition to repetition. In the RC problem, the approximate frequency-by-frequency monotonic convergence condition from period to period is used to derive the repetitive controllers.

The observed improvement in stability robustness by the averaging technique is found to be significant for all these ILC and RC designs. Consider the stability boundary to be the computed spectral radius equal to 1.001. Such a value produces well behaved learning histories, at least for many repetitions, and may in fact be asymptotically stable if the computation error is eliminated. We point out that even if it is not asymptotically stable, such a law can be very useful in practice, since it most often will produce a very small error level before any effect of instability is observed, as in Longman and Huang (1994). The results here show that the quadratic cost ILC was unstable (i.e. above 1.001) for 10.5% of the 200 models using the nominal model to do the design, while no model exceeded this spectral radius after averaging. Hence, they could be used for learning in practice. The Euclidean Norm Contraction Mapping law had 40% of the 200 models definitely unstable when designed from the nominal model, and again no model exceeded this radius after averaging. The Partial Isometry law had 39.5% definitely unstable before averaging and only one model unstable after averaging. Obtaining this much robustification with very little effort makes the averaging approach presented here have real significance in practice. Finally, although the focus of this paper is for ILC and RC robustification, the averaging technique is quite general and can be applied to any controller design that involves the minimisation of a cost function.

References

Amman, N., Owens, D. H., Rogers, E., & Wahl, A. (1996). An H-infinity approach to linear iterative learning control design. *International Journal of Adaptive Control and Signal Processing*, *10*, 767–781.

Apostolakis, G. (1990). The concept of probability in safety assessments of technological systems. *Science*, *250*, 1359–1364.

Arimoto, S., Kawamura, S., & Miyazaki, F. (1984). Bettering operation of robots by learning. *Journal of Robotic Systems*, *1*(2), 123–140.

Bao, J., & Longman, R. W. (2008). Enhancements of repetitive control using specialized FIR zero-phase filter designs. *Advances in the Astronautical Sciences*, *129*, 1413–1432.

Barlow, J. S. (2009). *Direct data-based model predictive control with applications to structures, robotic swarms and aircraft* (PhD Thesis). Thayer School of Engineering, Dartmouth College, pp. 87–95.

Bien, Z., & Xu, J.-X. (1998). *Iterative learning control: Analysis, design, integration, and applications*. Boston: Kluwer Academic Publishers.

Brown, G. W. Monte carlo methods. In E. F. Beckenbach (Ed.), *Modern mathematics for the engineer*. New York: McGraw-Hill.

Brown, H. M., Phan, M. Q., Lee, S. C., & Longman, R. W. (2007). Robustified repetitive controllers with monotonic convergence for multiple-input multiple-output systems. *Advances in the Astronautical Sciences*, *129*, 1893–1912.

Chen, K., & Longman, R. W. (2003). Creating short time equivalents of frequency cutoff for robustness in learning control. *Advances in the Astronautical Sciences*, *114*, 95–114.

Chew, K. -K., & Tomizuka, M. (1990). Steady-state and stochastic performance of a modified discrete-time prototype repetitive

controller. *ASME Journal of Dynamic Systems, Measurement and Control*, *112*(1), 35–41.

Dijkstra, B. G., & Bosgra, O. H. (2002). Extrapolation of optimal lifted system ILC solution with application to a waferstage. In *Proceedings of the American Control Conference* (2595–2600), Anchorage, AK.

Elci, H., Longman, R. W., Phan, M. Q., Juang, J.-N., & Ugoletti, R. (1994b). Discrete frequency-based learning control for precision motion control. In *Proceedings of the 1994 IEEE International Conference on Systems, Man and Cybernetics* (pp. 2767–2773). San Antonio, TX.

Elci, H., Longman, R. W., Phan, M. Q., Juang, J.-N., & Ugoletti, R. (2002). Simple learning control made practical by zero-phase filtering: Applications to robotics. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, *49*(6), 753–767.

Elci, H., Phan, M. Q., Longman, R. W., Juang, J.-N., & Ugoletti, R. (1994a). Experiments in the use of learning control for maximum precision robot trajectory tracking. In *Proceedings of the 1994 Conference on Information Sciences and Systems* (pp. 951–958). Princeton, NJ.

Frueh, A., & Phan, M. Q. (2000). Linear quadratic optimal learning control (LQL). *International Journal of Control*, *73*(10), 832–839.

Galkowski, K., Lam, J., Rogers, E., Xu, S., Sulikowski, B., Paszke, W., ...Owens, D. H. (2003). LMI based stability analysis and robust controller design for discrete linear repetitive processes. *International Journal of Robust Nonlinear Control*, *13*(13), 1195–1211.

Galkowski, K., Paszke, W., Sulikowski, B., Rogers, E., & Owens, D. H. (2002a). LMI based stability analysis and controller design for a class of 2D continuous-discrete linear systems. In *Proceedings of the 2002 American Control Conference* (p. 2934). Anchorage, AK.

Galkowski, K., Rogers, E., Xu, S., Lam, J., & Owens, D. H. (2002b). LMIs a fundamental tool in analysis and controller design for discrete linear repetitive processes. *IEEE Transactions on Circuits and Systems Part I: Fundamental Theory and Applications*, *49*(6), 768–778.

Henley, E. J., & Kumamoto, H. (1992). *Probabilistic risk assessment: Reliability engineering, design, and analysis*. New York: IEEE Press.

Ishihara, T., Abe, K., & Takeda, H. (1992). A discrete-time design of robust iterative learning controllers. *IEEE Transcations on Systems, Man, and Cybernetics*, *22*(1), 74–84.

Jang, H. S., & Longman, R. W. (1994). A new learning control law with monotonic decay of the tracking error norm. In *Proceedings of the Thirty-Second Annual Allerton Conference on Communication, Control, and Computing* (pp. 314–323). Monticello, Illinois.

Jang H. S., & Longman, R. W. (1996). Design of digital learning controllers using a partial isometry. *Advances in the Astronautical Sciences*, *93*, 137–152.

Lee, S. C., Phan, M. Q., & Longman, R. W. (2006). Multiple-model design of robustified repetitive controllers with optimized convergence rate. In *Proceedings of the 2006 AIAA Guidance, Navigation, and Control Conference*. Keystone, CO.

Longman, R. W. (2000). Iterative learning control and repetitive control for engineering practice. *International Journal of Control, Special Issue on Iterative Learning Control*, *73*(10), 930–954.

Longman, R. W., & Huang, Y.-C. (1994). Use of unstable repetitive control for improved tracking accuracy. *Adaptive Structures and Composite Materials: Analysis and Applications*, ASME, AD-Vol. 45/MD-Vol. 54, 315–324.

Moore, K. L. (1993). *Iterative learning control for deterministic systems: Advances in Industrial Control*. London: Springer-Verlag.

Owens, D. H. (1977). Stability of linear multipass processes. *Proceedings of the IEEE*, *124*, 1079–1082.

Owens, D. H., & Amann, N. (1994). Norm-optimal iterative learning control. Internal Report Series of the Centre for Systems and Control Engineering, University of Exeter.

Panomruttanarug, B., & Longman, R. W. (2004). Repetitive controller design using optimization in the frequency domain. In *Proceedings of the AIAA/AAS Astrodynamic Specialist Conference and Exhibit*. Providence, RI.

Panomruttanarug B., & Longman, R. W. (2006). Frequency based optimal design of FIR zero-phase filters and compensators for robust repetitive control. *Advances in the Astronautical Sciences*, *123*, 219–238.

Panomruttanarug, B., & Longman, R. W. (2007). Designing optimized FIR repetitive controllers from noisy frequency response data. *Advances in the Astronautical Sciences*, *127*, 1723–1742.

Panomruttanarug, B., Longman, R. W., & Phan, M. Q. (2007). Multiple model robustification of iterative learning and repetitive control laws including design from frequency response data. In *Proceedings of the AAS/AIAA Spaceflight Mechanics Meeting*. Savannah, GA.

Phan, M. Q., & Juang, J.-N. (1996). Designs of learning controllers based on an auto-regressive representation of a linear system. *Journal of Guidance, Control, and Dynamics*, *19*(2), 355–362.

Phan, M. Q., & Longman, R. W. (1988). A mathematical theory of learning control for linear discrete multivariable systems. In *Proceedings of the AIAA/AAS Astrodynamics Conference* (pp. 740–746). Minneapolis, Minnesota.

Phan, M. Q., Longman, R. W., & Moore, K. L. (2000). A unified formulation of linear iterative learning control. *Advances in the Astronautical Sciences*, *105*, 93–112.

Phan, M. Q., & Maghami, P. (1996). Estimation of vibration levels in flexible structures under periodic disturbances in the presence of modelling uncertainties. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*. San Diego, CA.

Plotnik, A. M., & Longman, R. W. (1999). Subtleties in the use of zero-phase low-pass filtering and cliff filtering in learning control. *Advances in the Astronautical Sciences*, *103*, 673–692.

Ray, L. R., & Stengel, R. (1992). Stochastic measures of performance robustness in aircraft control systems. *Journal of Guidance, Control, and Dynamics*, *15*(6), 1381–1387.

Rogers, E. (1987). *Feedback and stability theory for linear multipass processes* (Research Report). The Queen's University of Belfast, Department of Electrical and Electronic Engineering.

Songchon, S., & Longman, R. W. (2003). Comparison of the stability boundary and the frequency response stability condition in learning and repetitive control. *International Journal of Applied Mathematics and Computer Science*, *13*(2), 169–177.

Takanishi, K. (2005). *Probabilistic robust iterative learning control* (MS Thesis). Thayer School of Engineering, Dartmouth College.

Takanishi, K., Phan, M. Q., & Longman, R. W. (2005). Multiple-model probabilistic design of robust iterative learning controllers. *Transactions of the North American Manufacturing Research Institution, Society of Manufacturing Engineers*, *33*, 533–540.

Tayebi, A., & Zaremba, M. B. (2000). Internal model based robust iterative learning control for uncertain LTI systems. In *Proceedings of the 39th IEEE Conference on Decision and Control* (pp. 3439–3444). Sydney, Australia.

Uchiyama, M. (1978). Formulation of high-speed motion pattern of a mechanical arm by trial, (in Japanese). *Transactions of the Society for Instrumentation and Control Engineers*, *14*, 706–712.

Xu, J., Sun, M., & Yu, L. (2008). LMI-based synthesis of robust iterative learning controller with current feedback for linear uncertain systems. *International Journal of Control, Automation, and Systems*, *6*(2), 171–179.

AAS 11-587

DESIGNING STABLE ITERATIVE LEARNING CONTROL SYSTEMS FROM FREQUENCY BASED REPETITIVE CONTROL DESIGNS

Benjamas Panomruttanarug,[*] Richard W. Longman,[†] and Minh Q. Phan[‡]

Repetitive control (RC) and iterative learning control (ILC) design control systems that aim for zero tracking error in repeating situations. ILC applies to spacecraft problems that perform repeated scanning maneuvers with a fine pointing instrument. Very effective RC design methods use steady state frequency response approaches. They cannot be applied directly to ILC because ILC is a finite time problem, and asks for zero error not only in the part of the desired trajectory after transients have decayed, but also asks for zero error during the transients as well. This work converts these effective RC design methods so that they apply to the ILC problem, and produce convergence to zero tracking error that is monotonic with iterations in the sense of the Euclidean norm. One first fills the ILC gain matrix with the RC gains, and then adjusts a few gains associated with the first few time steps, using a steepest descent or other similar algorithm. We show that adjusting only one gain can be sufficient to produce asymptotic stability and monotonic convergence. The method is simple to apply, and allows one to make use of a very effective RC design method in ILC applications, and represents a powerful way to apply frequency response ideas to finite time ILC problems.

INTRODUCTION

Repetitive control (RC) and iterative learning control (ILC) are sister fields that aim to converge to zero tracking error in repeating situations. Repetitive control applies to control systems that perform periodic commands, or to systems that are subject to periodic disturbances (see References 1-5). Because time progresses indefinitely, one can make designs based on steady state frequency response, and then the error can asymptotically approach zero as time progresses.

Iterative learning control applies to situations in which one wants to perform a specific tracking command starting from the same initial condition each time, and want to have zero error every time step of the finite time trajectory. Hence, one is asking for zero error not in steady state only, but throughout the initial transients each run. See References 5-9.

[*] Assistant Professor, Department of Control System and Instrumentation Engineering, King Mongkut's University of Technology Thonburi, Bangmod Tungkru, Bangkok, Thailand 10140.
[†] Professor of Mechanical Engineering, Columbia University MC4703, 500 W. 120th St., New York, NY 10027 USA
[‡] Associate Professor of Engineering, Thayer School of Engineering, 8000 Cummings Hall, Dartmouth College, Hanover, NH 03755-8000 USA.

The most important RC application for spacecraft is the isolation of fine pointing equipment from the vibrations caused by slight imbalance in CMG's or reaction wheels (References 10-11). ILC applies when one wants to make repeated scanning maneuvers with fine pointing equipment.

Very effective design methods have been developed to generate repetitive control laws making use of steady state frequency response, as presented in References 12,4. The approach designs an FIR filter that mimics the inverse of the steady state frequency response of the feedback control system. This produces a stable but noncausal transfer function that can be applied to the data from the previous period. This circumvents the difficulty that the inverse of the discrete time z-transfer function that would handle the transient phase too, is almost always unstable and cannot be used in the RC design. The approach can be amazingly effective. Using only a linear combination of 12 errors observed in the previous period of the disturbance or command, one can get an effective inverse of the system that is good to three digits for a third order system, as described in References 12,4 and also cited below. The design places zeros at roughly evenly spaced locations around a circle in the z-plane, in order to cancel the effect of each zero introduced on the negative real axis by the discretization process.

The simplest form of ILC adds to the command at each time step, a constant times the error observed at the corresponding time in the previous run, but shifted forward by the time delay through the feedback control system. This ILC law can be proved to be asymptotically stable and converge to zero tracking error for essentially all linear systems (e.g. Reference 5), and also for nonlinear systems satisfying a Lipschitz condition (Reference 13). Convergence is based on a "transient wave" of zero error that propagates from the start of the trajectory, and eventually reaches the end of the trajectory. But the learning transients can easily become astronomical on the way to zero error, because the behavior in the parts of the trajectory in front of the "wave" are being excited (Reference 5). This includes any parts of the trajectory that can be considered as steady state behavior, i.e. as beyond the settling time of the system. Reference 9 suggested that ILC should be designed to satisfy a monotonic decay condition for the steady state frequency response, in order to avoid such prohibitively bad transients. This paper develops methods to very directly do this, using an RC design based on an FIR filter that mimics the inverse of the steady state frequency response in order to create the learning matrix. Then a few of the gains associated with the initial time steps are adjusted to convert to a stable and monotonically converging ILC algorithm. This successfully converts the very effective RC design method into a very effective and easy to use ILC design method.

This paper follows from two previous works, References 14,15. The first reference makes use of the same RC gains to fill the ILC gain matrix, and iteratively changes the initial conditions to produce convergence. There is no need for adjustment of the initial conditions using the method presented here. The second reference uses a cost function that aims to decrease a quadratic cost during learning that penalizes the sum of singular values squared of the error propagation matrix, together with a penalty on the change in the command input from run to run. The present paper is similar, but focuses directly on the maximum singular value and does a minimal amount of gain adjustment in order to produce stability and monotonic convergence of the ILC design. Examples show that one may be able to produce this monotonic convergence by adjusting only one gain in the upper left corner of the ILC gain matrix, and do this by simply plotting the maximum singular value versus this gain.

ILC PROBLEM FORMULATION AND CONVERGENCE CONDITIONS

Consider a scalar input, scalar output system

$$x(k+1) = Ax(k) + Bu(k)$$
$$y(k) = Cx(k) + v_d(k) \tag{1}$$

Usually this represents a feedback control system, and $u(k)$ then represents the command. The $v_d(k)$ represents any repeating disturbance that occurs every time one gives a command to the system. Wherever the disturbance occurs in the feedback control system, it is converted to its equivalent output disturbance used here. The iterative learning control problem seeks to iteratively adjust the command $u(k)$, $k = 0,1,...,p-1$ from one run to the next, aiming to converge to that command which produces an output $y(k)$ matching the desired output history, $y_D(k)$, $k = 1,2,...,p$. We assume $CB \neq 0$ (simple adjustments can be made when it is zero). The solution of this equation for every time step in the desired trajectory, can be packaged as

$$\underline{y}_j = \bar{A}x(0) + P\underline{u}_j + \underline{v}_d \tag{2}$$

where j is the run number, also called the repetition number or iteration number, and

$$\underline{y}_j = \begin{bmatrix} y_j(1) & y_j(2) & \cdots & y_j(p) \end{bmatrix}^T$$
$$\underline{u}_j = \begin{bmatrix} u_j(0) & u_j(1) & \cdots & u_j(p-1) \end{bmatrix}^T$$
$$P = \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{p-1}B & CA^{p-2}B & \cdots & CB \end{bmatrix} \quad ; \quad \bar{A} = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^p \end{bmatrix} \tag{3}$$

The time arguments in the entries of $\underline{v}_d$, $\underline{y}_D$, $\underline{e}_j = \underline{y}_D - \underline{y}_j$ are the same as those in $\underline{y}_j$, and are shifted by one time step from those in $\underline{u}_j$, reflecting the one time step delay going through the system when $CB \neq 0$. The ILC objective is that as $j \to \infty$, the output $\underline{y}_j \to \underline{y}_D$. The command input must then converge to

$$\underline{u}_D = P^{-1}[\underline{y}_j - \bar{A}x(0) - \underline{v}_d] \tag{4}$$

where the P matrix is the true P matrix associated with the behavior of the hardware, which is likely to differ from the P matrix associated with a system model used for ILC design. The fact that the iterations are made with the real world allows the possibility of getting zero error in hardware while using an imperfect model in the ILC law design.

Define a difference operator in repetitions so that $\delta_j \underline{u} = \underline{u}_j - \underline{u}_{j-1}$ and $\delta_j \underline{y} = \underline{y}_j - \underline{y}_{j-1}$, and apply these to Eq. (2) to produce $\delta_j \underline{e} = -\delta_j \underline{y} = -P\delta_j \underline{u}$. A general linear first order ILC law takes the form $\delta_j \underline{u} = L\underline{e}_{j-1}$, or

$$\underline{u}_{j+1} = \underline{u}_j + L\underline{e}_j \tag{5}$$

where L is the $p \times p$ matrix of learning gains ℓ_{ij}, for row i and column j. Substituting Eq. (5) into the error update equation above Eq. (5) produces the error propagation formula from iteration to iteration

$$\underline{e}_{j+1} = (I - PL)\underline{e}_j \tag{6}$$

The error vector $\underline{e}_j$ will converge to zero for all possible initial error history vectors $\underline{e}_0$ obtained in the first run, if and only if the spectral radius of matrix $(I - PL)$ is less than unity, i.e. all eigenvalues of this matrix must be less than unity in magnitude. Another important condition is: the Euclidean norm of the error will decay monotonically for all possible initial error history vectors $\underline{e}_0$, if and only if the maximum singular value of matrix $(I - PL)$ is less than unity. The first condition is the stability boundary, an if and only if condition, and the second condition is a sufficient condition for stability, and one that has the desirable monotonic error decay property:

$$\left|\lambda_i(I - PL)\right| < 1 \quad i = 1,2,...,p \tag{7}$$

$$\max_i \sigma_i(I - PL) < 1 \quad i = 1,2,...,p \tag{8}$$

It is instructive to examine a transfer function analysis of the ILC equations. Here we ignore the fact that we are dealing with finite time signals, and take the z-transform of Eq. (1) for the jth run

$$\begin{aligned} Y_j(z) &= G(z)U_j(z) + V(z) + C(zI - A)^{-1}zx(0) \\ G(z) &= C(zI - A)^{-1}B \end{aligned} \tag{9}$$

Then $\delta_{j+1}E(z) = -\delta_{j+1}Y(z) = -G(z)\delta_{j+1}U(z)$. Also take the z-transform of Eq. (5)

$$\delta_{j+1}U(z) = L(z)E_j(z) \tag{10}$$

We will be considering the steady state frequency response when we use the transforms, and this is obtained by substituting $z = \exp(i\omega T)$ where T is the sample time interval and ω is the radian frequency. In order for transform methods to make sense with the learning law, L must be something that can be represented as a difference equation. This means that the entries must have the property that every entry in any diagonal must be the same (as in a Toeplitz matrix), and we are looking only at steady state response, which means the matrix must be big enough that much of the trajectory is after the transients of the initial conditions are negligible, and one ignores the right and left ends of the matrix for which the gains become truncated. Under these circumstances, one can describe the learning law as a z-transform $L(z)$. Combining these equations produces

$$E_{j+1}(z) = [1 - G(z)L(z)]E_j(z) \tag{11}$$

which presents a transfer function transforming the error in one iteration into the error in the next iteration. If the magnitude of the frequency response function is less than unity for all frequencies up to Nyquist frequency,

$$\left|1 - G(e^{i\omega T})L(e^{i\omega T})\right| < 1 \quad \forall\ \omega, 0 \le \omega T \le \pi \tag{12}$$

then every steady state frequency component of the error will decay every iteration. This is not a proof of stability. It ignores the fact that ILC is a finite time problem, and that it asks for zero error during the transients as well as any part of the response that might be modeled by steady state response. If the ILC is unstable then there is no portion of the trajectory that can be modeled by the above equation. Nevertheless, when the ILC is stable, the condition suggests good learning transients, and Reference 9 suggests that one should design ILC to satisfy this condition for this

purpose. This paper does exactly that, designs the ILC law based on this condition, and then makes small modifications to a few gains in the upper left corner of L in order to satisfy the sufficient condition for convergence, Eq. (8), which also guarantees monotonic convergence of the error Euclidean norm.

THE RC GAINS INTRODUCED INTO THE ILC GAIN MATRIX L

Repetitive control differs from ILC in that it aims to eliminate tracking error in following a periodic command, or a constant command but in the presence of a periodic disturbance of known period. We take this period to be p time steps. Instead of observing the error in the previous run and modifying the command history for the current run, RC at each time step, in real time, observes the error during the previous period and adjusts the current command. There is no restarting of the system, and time progresses toward infinity. This means that one can rigorously made designs based on steady state frequency response. Reference 4 proves that for repetitive control, a necessary and sufficient condition for convergence to zero steady state error for all possible periods p, is that condition Eq. (12) is satisfied. The effective RC design method of References 12,4 designs an FIR compensator of the form

$$\begin{aligned} L(z) &= a_1 z^{m-1} + a_2 z^{m-2} + \cdots + a_m z^0 + \cdots a_{n-1} z^{-(n-m-1)} + a_n z^{-(n-m)} \\ &= (a_1 z^{n-1} + a_2 z^{n-2} + \cdots + a_m z^{n-m} + \cdots + a_{n-1} z^1 + a_n z^0) / z^{(n-m)} \end{aligned} \tag{13}$$

where the z^0 term applies to the error observed one period back, and the other terms are applied to errors both forward and backward from this point. The designer picks the number of gains n, i.e. the number of errors to include, and also picks how many forward and how many backward gains to consider, by choice of the integer m. The coefficients are designed to minimize the cost functional that aims to satisfy Eq. (12)

$$J = \sum_{j=0}^{N} [1 - G(e^{i\omega_j T}) L(e^{i\omega_j T})] w_j [1 - G(e^{i\omega_j T}) L(e^{i\omega_j T})]^* + v(a_1^2 + a_2^2 + \cdots + a_n^2) \tag{14}$$

The superscript asterisk indicates complex conjugate, and the w_j are adjustable weights which are set to unity here. The v is discussed below, and will normally be set to zero.

The full RC design approach in Reference 4 also asks for a zero-phase low-pass filter cutoff of the learning process, because one cannot usually have good magnitude and phase information to know that Eq. (12) is satisfied all the way up to Nyquist frequency. In RC this is done with an FIR design similar to Eq. (14) (References 16,4). The same is done in ILC but using zero-phase Butterworth filters (References 9,17,5). The cutoff is most likely adjusted based on hardware performance, since one does not normally know the frequency above which one's model becomes too poor to use for design. But it can also be based on how big the corrective action has to be in order to cancel high frequency error, because these corrective actions could saturate the actuators, require too much control effort, or too much control energy.

Numerical examples will be given making use of a third order differential equation model of the feedback control systems for each link of a Robotics Research Corporation robot as in Reference 9, with transfer function from command to response given by

$$G(z) = \left(\frac{a}{s+a}\right)\left(\frac{\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2}\right) \tag{15}$$

where $a = 8.8$, $\zeta = 0.5$, $\omega_0 = 37$, and for the purposes of the current study we usually use the discrete time equivalent when fed by a zero order hold sampling with sample time interval $T = 1/50$. The effectiveness of the design method is illustrated as in References 12,4, where a sample interval of $T = 1/100$ second was used and n was taken as 12, and m was adjusted for best performance with $m = 7$. Figure 1 plots the product $G(e^{i\omega_j T})L(e^{i\omega_j T})$ for frequencies from zero to Nyquist. The magnitude of the deviation from the point +1 is the value of the left hand side of Eq. (12), showing that the FIR compensator using 12 gains is very close to an inverse of the frequency response.

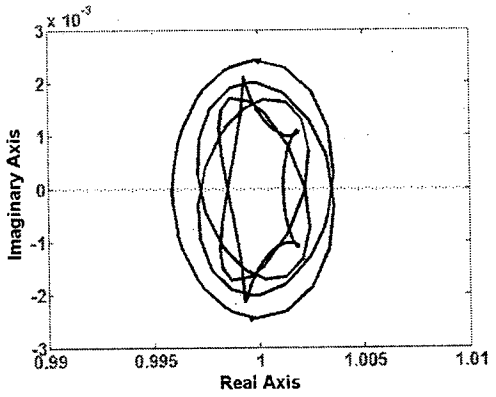

Figure 1. Plot of $G(e^{i\omega_j T})L(e^{i\omega_j T})$ for a 12 gain design $n = 12, m = 7$.

To use the result to fill up the matrix L, one enters the 12 gains at the appropriate locations on each row of the matrix, but truncating the gains at the start and end of the matrix as needed. The coefficient a_m in repetitive control multiplies the error in the previous period precisely one period back. Because of the one step time shift in the definitions of $\underline{u}$ and $\underline{e}$, the gain a_m should fill the subdiagonal, i.e. the diagonal immediately below the main diagonal in the L matrix. And the other coefficients are filled in accordingly.

Here we describe the somewhat arbitrary way we chose to put entries into the matrix L for the computations made in later sections. The number of time steps in an ILC run, also denoted by p, is chosen as an odd number, either 51 or 101 time steps. Then the RC gains are computed for n equal to 51 or 101, supplying enough gains to fill the central row of L. The value of m is chosen to produce 26 coefficients for errors future to the time step of the control being computed, and 25 time steps in the past and at the time step of the current control being computed, for the case of $p = 51$. For $p = 101$, there are 51 future gains and 50 gains for past and current time steps. These numbers are repeated to fill in all elements along their respective diagonals. This leaves triangular areas with zero entries in the upper right and lower left of matrix L. One could fill in the full matrix with computed gains, but as seen below, the gains get small going far enough forward or backward in time and can be truncated. And as seen in Fig. 1 that uses 12 gains, the number of gains required for good performance need not be large

This does not produce a matrix L that is guaranteed stable or has monotonic decay of the error, and we develop methods to modify a small number of gains in the upper left corner of L to produce these desired properties.

THE DESIGN PARAMETER ν IN THE COST FUNCTIONAL

When the sample time interval is short, the gains can become large, with successive gains being of opposite sign. This last property is needed to learn quickly at frequencies near Nyquist because at Nyquist there are only two samples per period. The penalty term with ν in the cost function Eq. (14) was introduced for use in case this causes ill conditioning due to taking the differences of nearly equal numbers.

To study the effects of ν, we start with Figure 2 that shows the gains when ν is zero. It shows gains computed for a 51 gain design for the third order system Eq. (15), which are to be read off the scale at the top of the plot. Also given are gains for a 101 gain design using the bottom scale. Both use the same sample rate of 50Hz. To graphical accuracy the 51 gains of the first design are the same as the corresponding gains in the 101 gain design. One observes that the gains become near zero rather quickly as one deviates from the central gain in the plot. Figure 3 is a zoomed in plot to show how the gains change as one introduces small nonzero values for ν, and one sees that the size of the largest gain decreases quickly when ν is introduced. As it increases, the property of alternating sign from one gain to the next disappears, which makes it difficult to learn fast near Nyquist frequency.

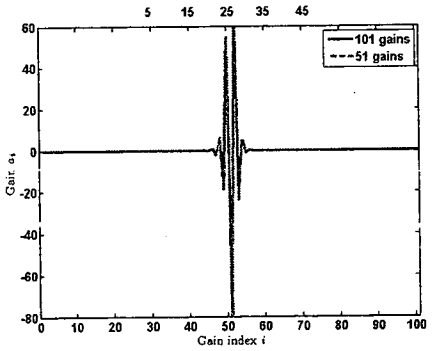

Figure 2. RC gains for 51 and 101 gain designs, with $\nu = 0$, and 50Hz sample rate.

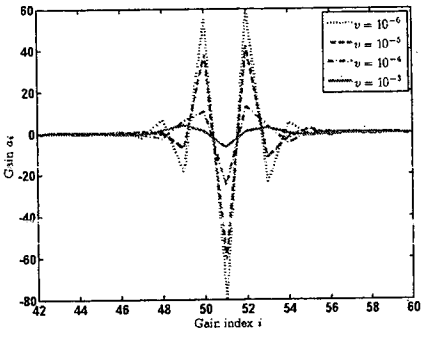

Figure 3. Detail of gains using 101 gain design, and varying ν, 50Hz sample rate.

Figure 4 gives the plot of $G(e^{i\omega_j T})L(e^{i\omega_j T})$ for the 51 gain result with $\nu = 0.1$, analogous to Fig. 1 that used 12 gains and $\nu = 0$. The radial distance from +1 is the amount of decay of the associated frequency point, i.e. the left side of Eq. (12), and it approaches the origin, meaning there is almost no change in the error at the corresponding frequency from one iteration to the next. Figure 5 gives a detailed view near the origin, and it shows that there is decay of the error at all frequencies. Figure 6 however increases ν to 0.2, and we see that now the distance from +1 gets larger than one, and some frequencies get amplified from one iteration to the next. The increased penalty on the size of the gains has now stopped the learning process from working at high frequencies, and produced an instability.

The ν penalty acts like a low pass filter on the learning process. Figure 7 studies this by finding that frequency for which the radial distance from the +1 point reaches 0.95, which the figure chooses to call a cutoff frequency. This plot corresponds to a 50Hz sample rate, producing a 25Hz Nyquist frequency, and the vertical axis gives the "cutoff" frequency in Hz vs. the value of ν. Note that one still needs to use a separate cutoff filter for robustification to model error at high frequency, a filter that is applied to the whole updated signal, not just the error input to the learning law.

When the gains generated by Eq. (14) with nonzero v are used to form the learning gain matrix L, without any adjustment of these gains, the maximum singular values of $[I - PL]$ are plotted in Fig. 8. We see that the maximum singular value goes toward unity, which is consistent with the concept that the learning from one iteration to the next gets very slow at high frequencies. This section has examined how v influences the designs, and the result suggests that one may not want to use a nonzero v unless there really is difficulty associated with having nearly equal large gains with opposite signs creating differences of nearly equal number in the control computation. In the remaining numerical examples, the value of v will be set to zero.

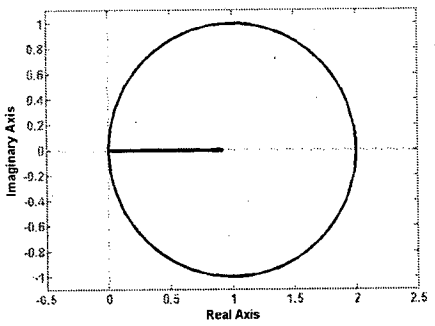

Figure 4. Plot of $G(e^{i\omega_j T})L(e^{i\omega_j T})$ using 51 gains, v = 0.1, with 100Hz sample rate

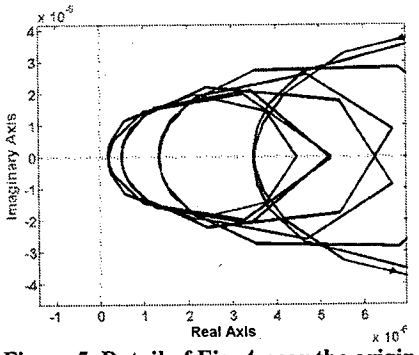

Figure 5. Detail of Fig. 4 near the origin.

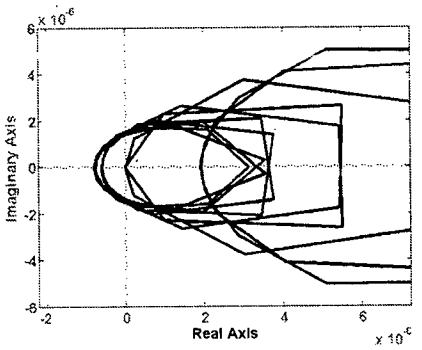

Figure 6. Detail near the origin of Fig. 5 when v is changed to 0.2.

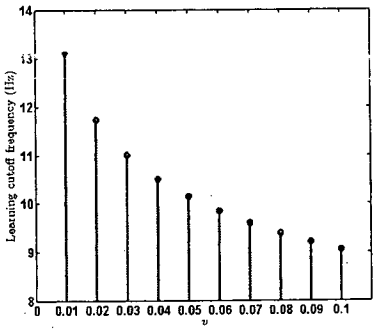

Figure 7. Learning cutoff frequency (Hz) vs. value of v.

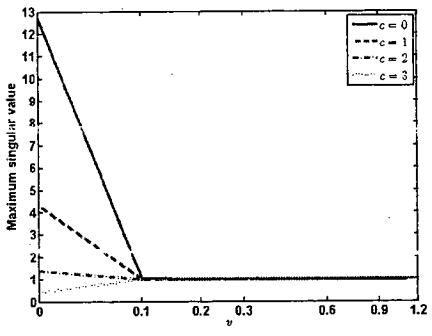

Figure 8. Maximum singular value of $[I - PL]$ for different values of v.

Table 1. Condition number of P_c.

	c=0	c=1	c=2	c=3
50Hz p=51	3.8e+18	241.66	241.59	241.52
50Hz p=101	2.7e+18	250.50	250.49	225.48
100Hz p=51	3.2e+19	1721.22	1720.75	1720.38
100Hz p=101	1.7e+19	1906.27	1906.24	1906.16

ILL CONDITIONING OF THE ILC PROBLEM AND ROW DELETIONS

The ILC problem which asks to obtain zero error every time step of a finite time trajectory by adjusting the zero order hold input, is usually an ill conditioned problem. The matrix P in Eq. (4) is guaranteed full rank if the number CB on the diagonal is nonzero (all eigenvalues are equal to CB), which it is for a SISO system with a one step time delay through the system (larger time delays are easily treated). But the condition number of P can easily be that of a singular matrix as indicated in Table 1 (References 18,19). The numbers for c equal to zero are wrong, the condition number is much worse – the computer is not capable of computing this number correctly, and the value saturates. The actual number grows roughly like 3 to the pth power as matrix size p is increased. This is a result of the fact that the inverse of a discrete time transfer functions obtained from a continuous time system fed by a zero order hold, is unstable for fast enough sample rate provided the pole excess in continuous time is 3 or more (Reference 20). The control action satisfying Eq. (4) is then an exponentially growing control action that alternates sign every time step. This means that the intersample error is also growing exponentially, but at every sample time the error happens to be going through zero. This means that even if one can compute the solution for the control action, and the actuators could actually apply it, it is not a control action that one would want to use. The intersample error is growing exponentially with time step. And this defeats the intended purpose of achieving zero tracking error.

References 18,19 present methods to address this problem, including doing so by not asking for zero error during the first few time steps. The number of time steps that are deleted from the start of the desired trajectory history, should be at least equal to the number of zeros outside the unit circle. For fast enough sample rate, this is one for pole excesses of 3 and 4, two for pole excesses of 5 or 6, etc. Table 1 shows the resulting condition numbers for the P matrix including the effect of removing $c = 1, 2,$ or 3 initial rows of P. We do this to make the ILC problem into a well posed inverse problem to be solved. Equation (4) now has an infinite number of solutions, one of which is the original inverse solution, but another is the pseudo inverse solution, which will be well behaved with the condition numbers in the table.

Note that because there is no attempt to achieve zero error for c time steps at the start of the trajectory, it is best whenever possible to prescribe a desired trajectory that is continuous with a number of derivatives that are also continuous, as one transitions from the physical state of the hardware before and at the initial time, to the desired output trajectory. A common situation is the system being at rest before and at the initial time, meaning all derivatives are zero. This is illustrated below by examples.

The equations and convergence conditions can be adjusted to handle this modified problem. The input history vector $\underline{u}_j$ remains unchanged since we still make use of all time steps of input, but $P, L, \underline{y}_j, \underline{y}_D, \underline{e}_j, \underline{v}_d$ become $P_c, L_c, \underline{y}_{c,j}, \underline{y}_{c,D}, \underline{e}_{c,j}, \underline{v}_{cd}$ which are the original matrices or column vectors with c initial rows removed, except in the case of L_c where c initial columns are removed. The convergence and monotonic decay conditions Eqs. (7,8), are changed by replacing P, L by P_c, L_c, and noting that the number of eigenvalues and singular values is reduced by c.

SENSITIVITY OF MAXIMUM SINGULAR VALUE OF $[I - PL]$ TO EACH GAIN IN L

We are not only interested in making the ILC law stable, we also want to have good learning transients in the sense that the Euclidean norm of the error decays monotonically. So our design objective is to satisfy the inequality of Eq. (8). When we simply use the RC gains computed as described above and insert them into the ILC law gain matrix L in the manner described above,

the result does not usually satisfy the sufficient condition Eq. (8), and also does not satisfy the necessary and sufficient condition for stability Eq. (7). Of course the process of inserting the gains into the matrix, means that near the top left corner of matrix L and also near the bottom right corner, a number of the substantial gains will be truncated because they seek to use error information from the previous run that apply to time steps before the start of the ILC finite time problem, or after the end of the problem. Experience in Reference 14 suggests that one may need to adjust the gains in the first few columns of L. To see which gains might be most effective at reducing the maximum singular value of $[I - P_c L_c]$ to below unity, we compute the sensitivity of the maximum singular value to ℓ_{ij}, the entry of L in row i and column j. For notational convenience we count rows/columns starting with the first row/column when none have been deleted from P and L, and use these same row numbers no matter what value of c is being considered.

Denote the singular value decomposition of $[I - P_c L_c]$ for whatever value of c is of interest (including $c = 0$) by USV^T where $S = diag(\sigma_1, \sigma_2, ..., \sigma_{p-c})$ is the matrix of singular values, and the kth column of the unitary matrix V is the column vector v_k. Define

$$H = [I - P_c L_c]^T [I - P_c L_c] \tag{17}$$

Then $V^T H V = S^2$ which shows that V diagonalizes matrix H with eigenvalues σ_k^2. A standard result for the derivative of an eigenvalue with respect to a parameter gives for symmetric matrices diagonalized by a unitary matrix is

$$\frac{\partial \sigma_k^2}{\partial \ell_{ij}} = v_k^T \frac{\partial H}{\partial \ell_{ij}} v_k \tag{18}$$

Note that $\partial \sigma_k^2 / \partial \ell_{ij} = 2\sigma_k (\partial \sigma_k / \partial \ell_{ij})$, so we can compute the sensitivity of any singular value to changes in the gain ℓ_{ij} in the learning gain matrix L, according to

$$\frac{\partial \sigma_k}{\partial \ell_{ij}} = \left(v_k^T \frac{\partial H}{\partial \ell_{ij}} v_k \right) / (2\sigma_k) \tag{19}$$

$$\frac{\partial H}{\partial \ell_{ij}} = [I - P_c L_c]^T \left[-P_c \frac{\partial L_c}{\partial \ell_{ij}} \right] + \left[-P_c \frac{\partial L_c}{\partial \ell_{ij}} \right]^T [I - P_c L_c] \tag{20}$$

and $\partial L_c / \partial \ell_{ij}$ is a matrix of the same dimension as L_c that is all zero except for unity in the location of the gain ℓ_{ij}.

Figures 9, 10, and 11 present the sensitivity results for all entries in matrix L_c using 51 gains produced by RC, using 50Hz sample rate, and setting $v = 0$ in the cost function. Figures 9, 10, and 11 are for $c = 0$, 1, and 2 respectively. The first three columns in all cases are by far the most sensitive, with the sensitivity decaying as one goes down the column. Although there is truncation of substantial gains as one approaches the bottom right corner of matrix L, there does not appear to be any special sensitivity of the maximum singular value for gains in that part of the matrix. The signs of the sensitivities appear to alternate from one column to the next, and also alternate as the value of c goes from 0 to 1 to 2. In all cases the most sensitive entry in the entire matrix is the entry in the top left corner. As c increases to 2, the sensitivity to this entry decreases by roughly a factor of 10.

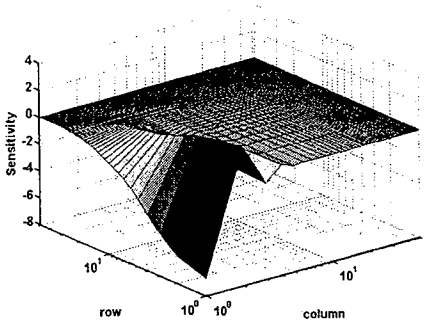

Figure 9. Sensitivity of maximum singular value to all entries in L, $c = 0$, 50Hz sample rate.

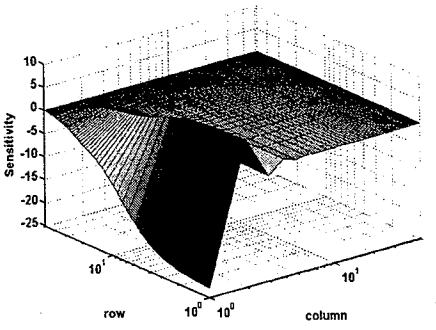

Figure 12. Sensitivity of maximum singular value to all entries in matrix L, $c = 0$, 100Hz sample rate.

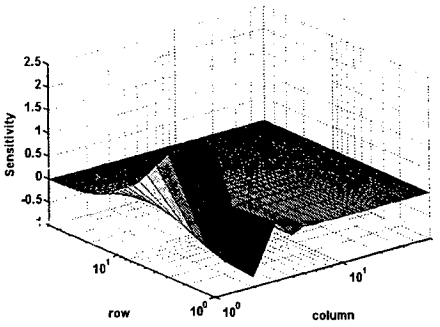

Figure 10. Same as Figure 9, for c = 1.

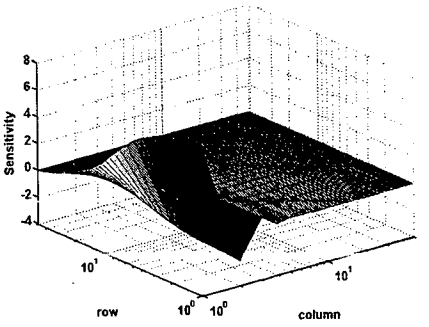

Figure 13. Same as Figure 12 with $c = 1$.

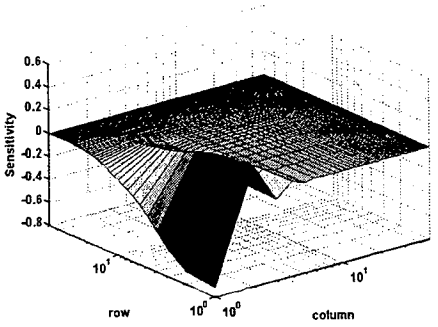

Figure 11. Same as Figure 9 for c = 2.

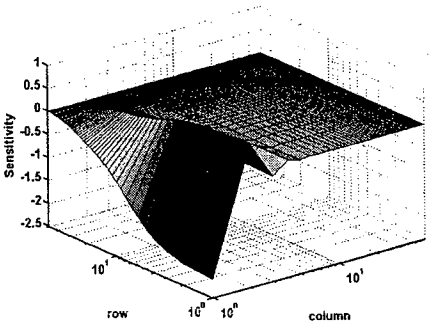

Figure 14. Same as Figure 12 with c = 2

Figures 12, 13, and 14 repeat the same sensitivities except this time the 51 gains are computed for a 100Hz sample rate. Although the first three columns are now associated with half the amount of time as in the previous figures, it is still the first three columns that have the most sensitivity, and in a very similar pattern. The main difference is that the sensitivities are increased by roughly a factor of 3.

ADJUSTING ONE GAIN AND THE DELETED ROW NUMBER c

Since the gain in the upper left corner of the ILC gain matrix L is the one with the largest influence on the maximum singular value, consider adjusting just this one gain. This makes a very simple optimization because it is a one dimensional search. Figures 15, 16, and 17 plot the maximum singular value as a function of the one gain in the upper left corner of L_c, denoted by $\ell_{11}, \ell_{12}, \ell_{13}$ for $c = 0, 1, 2$ respectively.

In Figure 15 for the case of $c = 0$, the original ℓ_{11} gain is -79.9166. The gain that minimizes the maximum singular value is ℓ_{11} = -37.5000. The spectral radius comes close to satisfying the stability condition Eq. (7), but the minimum possible maximum singular value is roughly 4, violating monotonic decay condition Eq. (8). Neighboring values of ℓ_{ij} are $l_{21} = 59.7207, l_{31} = -23.9624, l_{12} = 54.9593, l_{22} = -79.9166$. We conclude that one cannot satisfy the monotonic decay condition Eq. (8) by adjustment of this one gain alone, provided c = 0. Of course, we also want a nonzero value of c in order to have a well posed inverse problem.

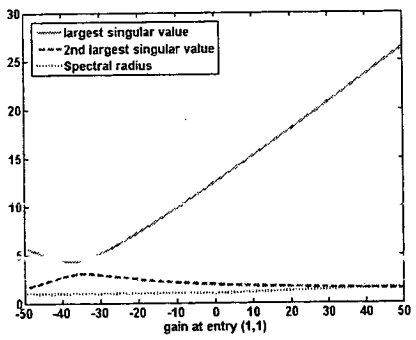

Figure 15. The largest and second largest singular values and the spectral radius as a function of gain ℓ_{11} for $c = 0$.

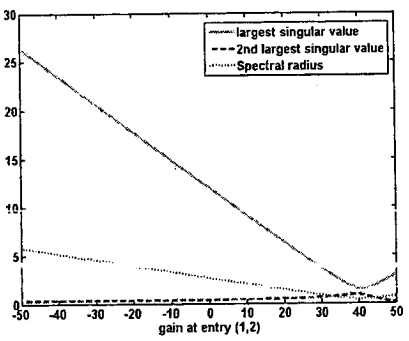

Figure 16. Same as Figure 15 for gain ℓ_{12} for c = 1.

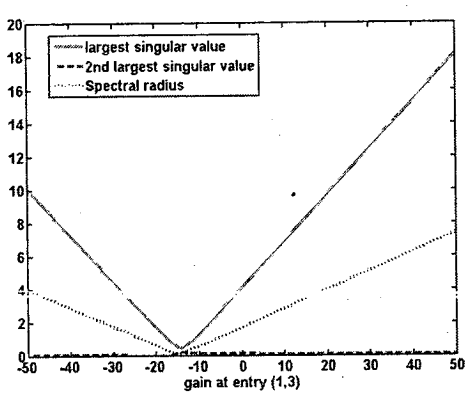

Figure 17. Same as Figure 15 for gain ℓ_{13} with c = 2.

Figure 16 is the corresponding plot for the case $c = 1$. This time stability is achievable by adjusting the one gain in the upper left corner, but the maximum singular value gets much closer to satisfying Eq. (8) but does not get below unity. Figure 17 presents the case $c = 2$, and this time

one can have not only asymptotic stability, but guaranteed monotonic decay. The original ℓ_{13} gain is -18.9316, the minimizing gain is -14.5000, and the neighboring gains are $l_{23} = 54.9593, l_{33} = -79.9166, l_{14} = 6.5213, l_{24} = -18.9316$. We see that by picking c = 2, adjusting only one out of the approximately 1951 gains can produce stability and monotonic convergence. The maximum singular value achieved is 0.4808 which guarantees that the Euclidean norm of the error will decrease by more than a factor of 2 every iteration. This is very fast convergence.

ERROR BEHAVIOR AT INITIAL UNADDRESSED TIME STEPS AND TRAJECTORIES WITH SMOOTH STARTUP

As described above, not addressing the tracking error for a few initial time steps helps to create a well posed inverse problem. The value of c allows a stable in time control action, and as observed in the previous section, and can produce asymptotic stability with monotonic decay. When employing c for this purpose, it is of interest to know what the errors are like for the c unaddressed time steps.

To study this we start with the same 3rd order system and use 51 gains from the RC design with v = 0, and adjust only the single top left entry in the learning gain matrix L_c as in the previous section, picking that gain that gives the smallest maximum singular value of $[I - P_c L_c]$. We consider four possible desired trajectories with $\underline{y}_D$ set to $\underline{y}_{D1}, \underline{y}_{D2}, \underline{y}_{D3}, \underline{y}_{D4}$

$$
\begin{aligned}
&\underline{y}_{D1} = (\pi/2)\sin(\omega t) \quad ; \quad \omega = 2\pi/8 \quad ; \quad t = kT \\
&\underline{y}_{D2} = (\pi/4)[1-\cos(\omega t)] \quad ; \quad \omega = 2\pi/4 \\
&\underline{y}_{D3} = \pi\left[5t^3/(100T)^3 - 7.5t^4/(100T)^4 + 3t^5/(100T)^5\right] \\
&\underline{y}_{D4} = (\pi/8)[1-\cos(\omega t)]^2 \quad ; \quad \omega = 2\pi/4 \\
&k = 0,1,2,\ldots,100
\end{aligned}
\tag{20}
$$

which are shown in Figure 18. Time k = 0 corresponds to the initial condition, and times $k = 1,2,...,100$ define the error history vector before any initial rows are deleted. We consider the 3rd order system to be at rest before and at time zero, i.e. all initial derivatives are zero before and at time zero. The desired trajectory $\underline{y}_{D1}$ makes a continuous function going through time zero, but the first derivative has a jump discontinuity. The trajectory $\underline{y}_{D2}$ is continuous and its first derivative is continuous, but the second derivative has a jump discontinuity. Then $\underline{y}_{D3}$ is continuous through the second derivative, while $\underline{y}_{D4}$ is continuous through the 3rd derivative. The ILC learning process is affected by discontinuities in derivative whether they are at the start of the trajectory or in the middle of a trajectory.

Figure 19 considers the case c = 0, and plots the root mean square (RMS) of the error history for all 100 time steps for all four trajectories since all time steps are addressed, using square P and L matrices that are 100 by 100. The spectral radius for this case is 1.000 and the maximum singular value is 4.3769. Note that the decay of the error is not monotonic. Because P is numerically singular there is an error subspace that cannot learn, and we see the error fails to go to zero. For future comparison, Figures 20, 21, 22, and 23 plot the error at time steps $k = 1,2,3,4$ for each of the desired trajectories, as a function of iteration number. The errors stop changing after a rela-

tively small number of iterations, and do not converge to zero, although smoother updates produce much smaller error levels at these steps.

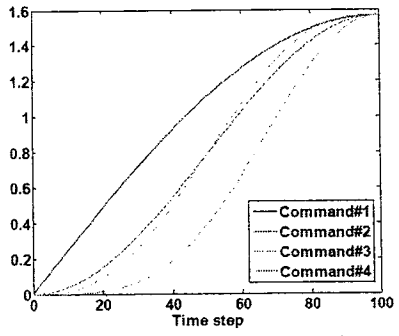

Figure 18. The 4 desired trajectories.

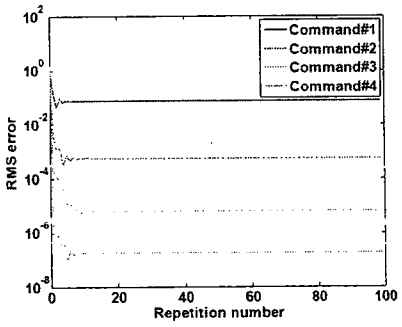

Figure 19. The RMS error for all time steps vs. iterations for $c = 0$.

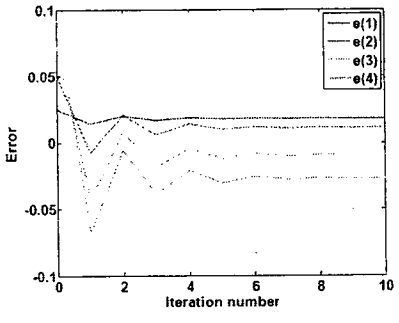

Figure 20. Error at the first 4 time steps vs. iterations for Command #1.

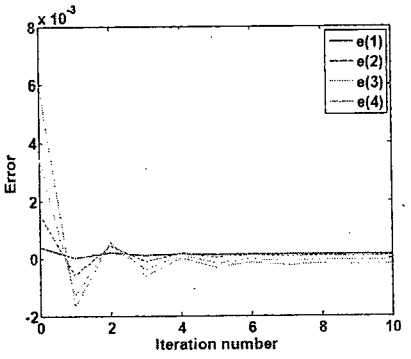

Figure 21. As in Figure 20 for Command #2.

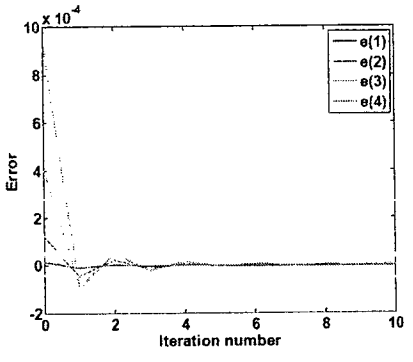

Figure 22. As in Figure 20 for Command #3.

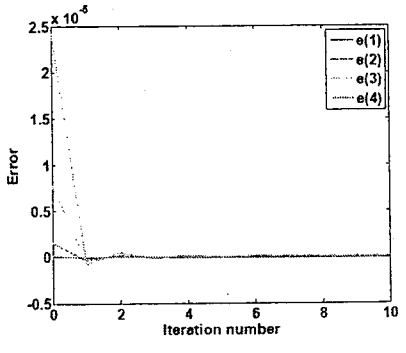

Figure 23. As in Figure 20 for Command #4.

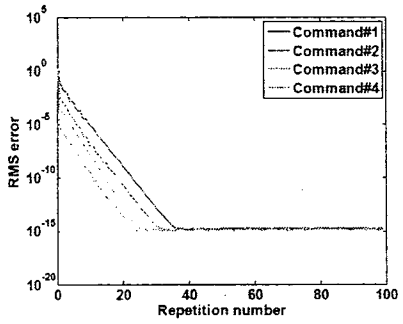

Figure 24. RMS error at addressed time steps using $c = 1$.

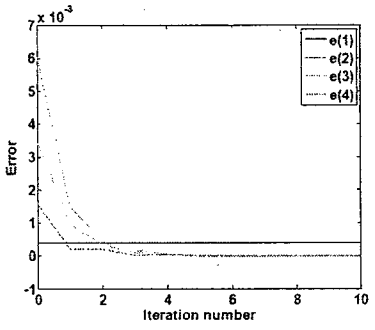

Figure 26. Same as Figure 25 for Command #2.

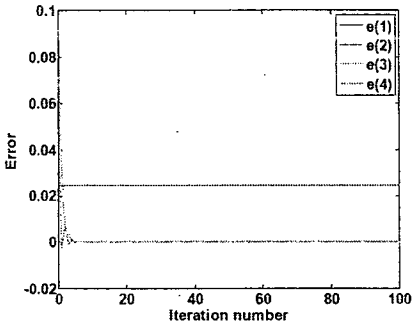

Figure 25. Error at the first 4 time steps using $c = 0$ and Command #1.

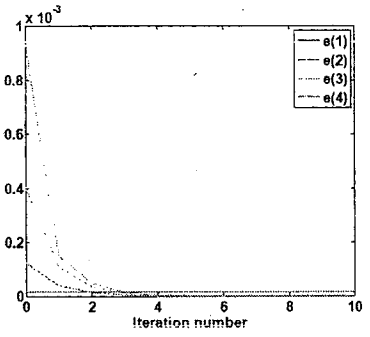

Figure 27. Same as Figure 25 for Command #3

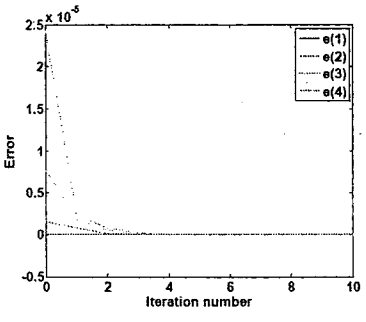

Figure 28. Same as Figure 25 for Command #4.

Figure 24 repeats Figure 19 but this time c is set to 1, making P_c a 99 by 100 matrix, and L_c a 100 by 99 matrix. The spectral radius for this case is 0.4132 and the maximum singular value is 1.4870. The RMS error is given for the addressed time steps, without time step $k = 1$. Figures 25, 26, 27, 28 repeat 20 through 23 for this c. No attempt is made to reduce the error at time step $k = 1$ to zero and it converges to some nonzero value which is smaller for the trajectories having continuity of more derivatives. Now the errors at addressed time steps 2, 3, and 4 approach zero, allowing the RMS error at addressed times to reach 10^{-15} for all trajectories.

Figures 29 through 33 are the analogous figures for $c = 2$. This time P_c is 98 by 100 and L_c is 100 by 98, and the errors at time steps $k = 1$ and 2 are not addressed. The RMS error at addressed points in Figure 29 decays more smoothly and faster than when $c = 1$.

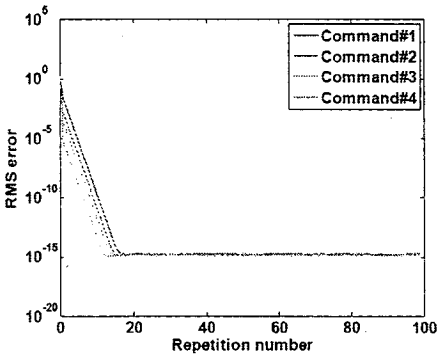

Figure 29. RMS error for addressed time steps using c =2.

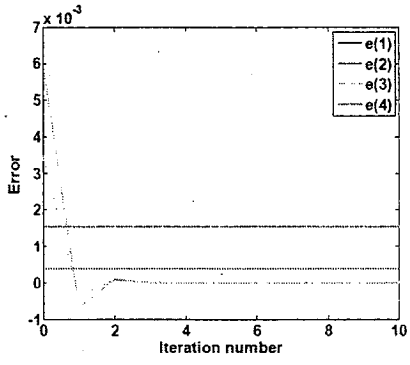

Figure 31. Same as Figure 30 for Command #2.

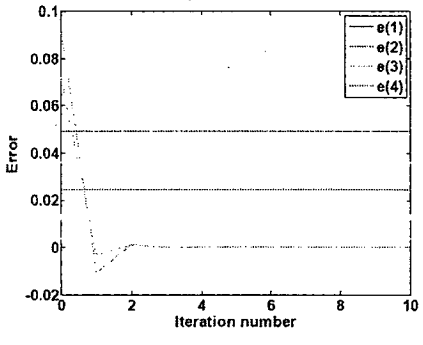

Figure 30. Error for the first 4 time steps vs. iterations for Command #1.

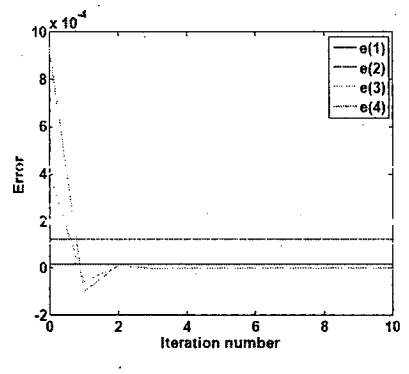

Figure 32. Same as Figure 30 for Command #3.

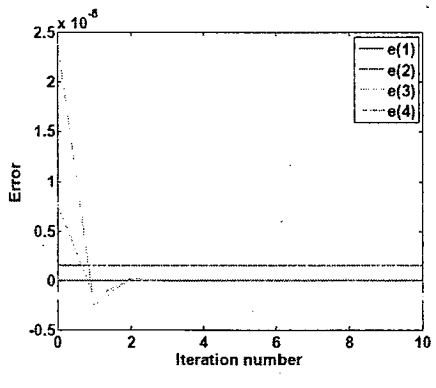

Figure 33. Same as Figure 30 for Command #4.

For these desired trajectories with both $c = 1$ and with $c = 2$, the ILC laws give very good performance after adjusting only one gain in matrix L_c. And the error level at the one or two time

steps that are not being learned can be very small if one picks a desired trajectory with a "smooth startup", i.e. with several derivatives being continuous going through the initial time.

ADJUSTING MORE THAN ONE GAIN IN MATRIX *L*

From the sensitivity transfer function plots in Figures 9 through 14 a number of gains in the upper left corner of the gain matrix L_c have a particularly strong influence on the maximum singular value, and adjusting a number of such gains could achieve better results. We can do this using a steepest descent method. Let $\underline{\ell}$ be a column matrix of the gains we choose to adjust and the maximum singular value of $[I-P_cL_c]$ can now be thought of as a function of these gains, $\sigma_{\max}(\underline{\ell})$. Let $\underline{\ell}_i$ be a specific set of values for the ith gain adjustment. By picking

$$\Delta\underline{\ell}_i = -\gamma_i \left(\frac{d\sigma_{\max}}{d\underline{\ell}}\bigg|_i \right) \tag{21}$$

the linearized change in the maximum singular value becomes

$$\sigma_{\max}(\underline{\ell}_i + \Delta\underline{\ell}_i) \approx \sigma_{\max}(\underline{\ell}_i) + \left(\frac{d\sigma_{\max}}{d\underline{\ell}}\bigg|_i \right)^T \Delta\underline{\ell}_i = \sigma_{\max}(\underline{\ell}_i) - \gamma_i \left(\frac{d\sigma_{\max}}{d\underline{\ell}}\bigg|_i \right)^T \left(\frac{d\sigma_{\max}}{d\underline{\ell}}\bigg|_i \right) \tag{22}$$

which will decrease the maximum singular value provided γ_i is not too large and the derivative of the singular value is not zero. Then one can think of the maximum singular value as a function $\sigma_{\max}(\gamma_i)$ of one parameter for this line search. And the minimizing value $\gamma_i = \gamma_i{}^*$ can be found as in the previous section that varied only one gain. Then set

$$\underline{\ell}_{i+1} = \underline{\ell}_i - \gamma_i{}^* \left(\frac{d\sigma_{\max}}{d\underline{\ell}}\bigg|_i \right) \tag{23}$$

re-evaluate the maximum singular value and the sensitivities for this new set of gains, and repeat. Note that the derivative of H in Eq. (19) is a linear function of the gains and hence, so is the sensitivity in Eq. (23).

Figures 34, 35, and 36 give an example of using this algorithm to optimize the first three gains in the first column of the learning matrix. No rows are removed from P for these results, $c=0$, and we see to what extent using more gains can improve the value of the maximum singular value. Figure 34 gives the result of a line search along the initial steepest descent direction for the three gains. The initial maximum singular value was above 12 as before. Varying γ along this direction did not produce as good a minimum as when only the first gain in the upper left corner was adjusted, compare the minima in Figure 15 and in Figure 34. Using the value corresponding to this minimum, the new steepest descent direction for the three gains was determined, and Figure 35 presents the line search for this direction. Figure 36 gives the best value for 9 successive line searches, and we see that the final result is essentially the same as the result adjusting only one gain in Figure 15.

Considering the strong dependence of the maximum singular value on the first entry in the top left corner of the learning gain matrix, one might wonder whether the optimized results require some gain in this position that is wildly different than neighboring gains. One case using 51 gains at 50Hz sample rate with c set to zero, has the following 7 gains going down the first column: -37.5, 59.7, -24.0. 5/21. -1.23. 0.246. -0.0536 (the 51st gain is 1.11×10^{-9}, justifying the

truncation of gains after the 51st). It decays more slowly going across the first row, again alternating sign, and ending at -0.267 at the 7th entry. The next element on the diagonal is -79.9166 which is the entry for the rest of the diagonal. So the optimized first entry does not appear to be wildly different, does not require something very special for this one step. Consider now the effect of using 100Hz sample rate instead of 50Hz, but keeping the same matrix size, and using 51 gains. Before optimizing the one gain in the upper left corner, the first seven gains going down the first column are: -627.8, 545.0, -238.1, 57.17, -13.74, 3.300, -0.7924. After optimizing, the first entry became -349. Optimizing changed the gain by less than a factor of 2. We do note, however, that the decay of the gain values going down the column is much more correlated to the time step than to the physical time. A fast sample rate will want fast changes in gain values in time.

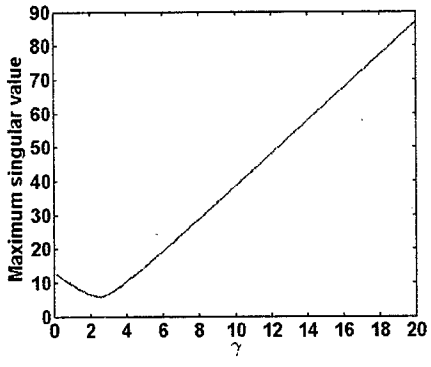

Figure 34. Line search along initial steepest descent direction varying the tip three gains in the first column.

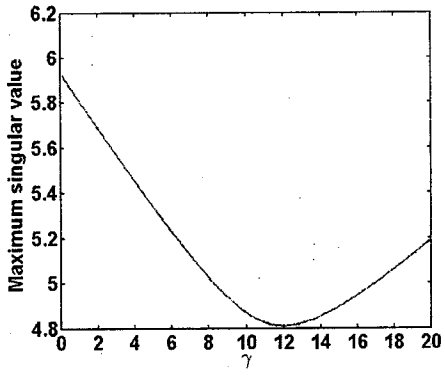

Figure 35. Second line search using first three gains in first column.

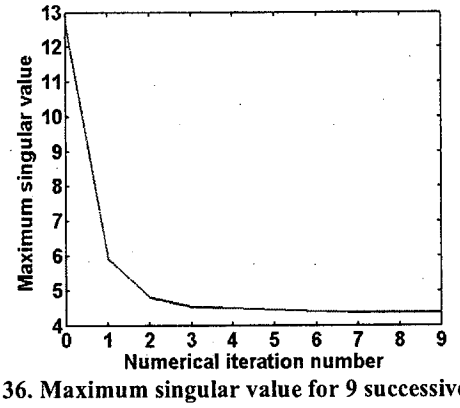

Figure 36. Maximum singular value for 9 successive directions along updated steepest descent directions

OTHER POSSIBLE COST FUNCTIONS

In the adjustment of the single gain we monitored the largest singular value and the second largest singular value. It did not occur that these two crossed and switched roles as the gain was adjusted, but it could have occurred, and this would have caused trouble to the steepest descent method when the derivative of the largest singular value is not continuous.

If such difficulties are encountered, one could decide to minimize a cost function that includes for example, the largest and the second largest singular values. Let $\underline{\sigma}$ be a column vector of the singular values one chooses to adjust, use subscript i for the ith updated value associated with $\underline{\ell}_i$,

and create the sensitivity matrix with sensitivities for each singular value, $S_i = \left(d\underline{\sigma}/d\underline{\ell}\right)\Big|_i$, whose kth column is the column vector of the derivative of the kth σ with respect to $\underline{\ell}$. Then Eq. (22) generalizes to $\underline{\sigma}(\underline{\ell}_i + \Delta\underline{\ell}_i) \approx \underline{\sigma}(\underline{\ell}_i) + S_i^T \Delta\underline{\ell}_i$. The update $\Delta\underline{\ell}_i$ can then be chosen to minimize the cost function

$$J_{i+1} = [\underline{\sigma}(\underline{\ell}_i + \Delta\underline{\ell}_i)]^T [\underline{\sigma}(\underline{\ell}_i + \Delta\underline{\ell}_i)] + r[\Delta\underline{\ell}_i]^T [\Delta\underline{\ell}_i] \tag{24}$$

which results in $\Delta\underline{\ell}_i = -[S_i S_i^T + rI]^{-1} S_i \underline{\sigma}(\underline{\ell}_i)$. A similar approach was used in Reference 15 but the cost function penalized the eigenvalues of H instead, i.e. the quadratic term was quadratic in the squares of the singular values.

Note that there are various numerical algorithms available for eigenvalue optimization of symmetric matrices, and one can of course appeal to these methods. See for example, Reference 21.

CONCLUSIONS

A method is presented here to design iterative learning control systems using the very effective repetitive control design method of Reference 12 which is based on steady state frequency response. Because ILC is a finite time problem that asks for zero tracking error at every time step, including time steps during the initial transients, steady state frequency response modeling does not immediately apply to the ILC problem. It is shown in examples that by adjusting only one gain in the ILC gain matrix generated using the steady state frequency response based RC design, one is able to achieve asymptotic stability and also monotonic decay of the Euclidean norm of the error. This is accomplished by making a graph of the maximum singular value as a function of the one gain. A steepest descent method is presented in order to achieve still better performance adjusting multiple gains. Additional methods are suggested to handle cases where the largest singular value is non differentiable.

The iterative learning control problem is very often an ill posed problem, because the system matrix for the convolution sum solution is numerically singular even though analytically it is guaranteed full rank. This occurs when a continuous time system with at least 3 more poles than zeros is fed by a zero order hold (with sufficiently fast sample rate). This property applies to all ILC laws. In the design approach used here, it is handled by not asking for zero error at a few time steps at the beginning of the trajectory. The error associated with these unaddressed time steps is studied, and it is suggested that whenever possible one should prescribe the desired trajectory in such a way that it is continuous together with continuity of several derivatives going through the time of start up.

The resulting design approach for ILC is an effective method that is easy to use. It is particularly easy to use when one only adjusts one gain in the learning matrix to achieve stability and monotonic convergence to zero tracking error.

The approach allows one to consider any sort of design for ILC that is based on frequency response thinking. After creating such a design, one adjusts a few gains to get asymptotic stability and monotonic decay of the error, making the design based on steady state response thinking apply to the finite time ILC problem including the transient phase.