Nordic Matlab Conference, Oslo 17–18 Oct, 2001

# Optimal control teaching using Matlab: Have we reached the turning point yet?

David I. Wilson
Department of Electrical Engineering, Karlstad University,
SE 651-88 Sweden

**Keywords:** Matlab/Simulink, optimal control, BVPs, teaching, laboratories

**Abstract**

The title of this article could be interpreted in two ways, and that is deliberate. The teaching of automatic control has undergone a renaissance in the last decade primarily due to Matlab. It is possible to deliver useful courses in automatic control utilising powerful techniques and we can finally rid ourselves of the tedium of manual, predominantly frequency design methods. Perhaps now, our teaching of this material is coming close to optimality.

The second interpretation of the title is the teaching of more advanced topics such as optimal control in undergraduate courses to engineers. Traditionally these topics have suffered from the requirement of higher mathematics (variational calculus, advanced linear algebra, stochastic control), non-trivial topics from numerical analysis (constrained optimisation, QPs, boundary value problems) and all manner of implementation problems. This results in a substantial learning period for the students before they are able to test these schemes on actual equipment. This lack of application, coupled with at times obscure theory is worrying to many students. Simultaneously I aim to show how we at Karlstad University address both these issues.

## 1 Background to control teaching

Many, if not most, universities now use Matlab in their teaching of automatic control. Over the last decade, we at Karlstad University have moved from a plethora of naive design tools, crude graphics, rudimentary symbolics and home-grown interfaces running under DOS with our laboratory equipment. Teaching introductory control courses, particularly to chemical engineering students unused to minimal help screens, was extremely inefficient. The best we could manage in a 5-week introductory control course was one step test, and a PID controller to be built in Pascal. Nowadays the situation has changed due to the standardising to one software environment, and the avoidance of tedious interface programming that offers little to the understanding of control.

The first big improvements came in 1995 when we interfaced Simulink to our collection of laboratory plants using the Real-time toolbox[1] and National Instruments[2] analogue to digital converter cards. Now for example the chemical engineering students whom typically have had little prior exposure to computers outside word-processing, can interface to a laboratory plant and implement and tune a PID controller from scratch. We feel that the improved understanding is due to that the diagram of the PID controller and transfer function model closely follows the block diagram in any standard control textbook, and that the Simulink diagram connected to the true plant is identical to a simulation version except that the transfer function is now replaced with input and output plugs as shown in Fig. 1. One example we use in our first undergraduate control course is to build a PID controller with integral windup protection following [1, Fig 8.10, p310].

What is particularly convenient is that we can easily proto-type a proposed controller design

[1] Available from Humusoft, Czech Republic `http://www.humusoft.cz/rt/irt.htm`

[2] LabPC 1200 available from `http://www.ni.com`

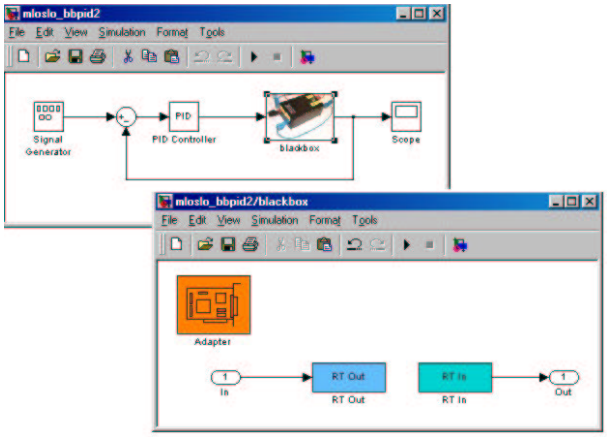


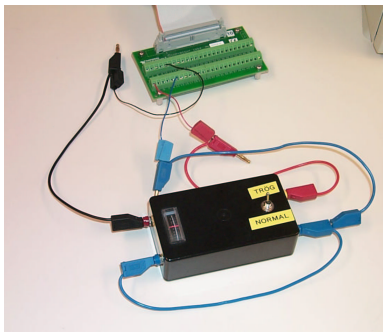

Figure 1: Real-time PID control using SIMULINK and the real-time toolbox from Humusoft. Any number of laboratory plants can be connected to the input and output plugs.

on a model, and then rapidly swap the model for the actual plant, be it a RC filter network, a cascaded series of liquid-level tanks, an electromagnetic balance arm, or even a model helicopter for real tests. Even more interesting for the students is that we can rapidly swap from one plant to a duplicate to test the robustness of the controller, or even swap from one plant to a different type of plant as shown in Fig. 2 and see exactly what components of the controller need re-designing. Since our advanced control courses are optional, only the motivated and interested students pursue them, many of who have had some prior industrial experience. They quickly appreciate that industrial controller design must be flexible and, as far as possible automated.

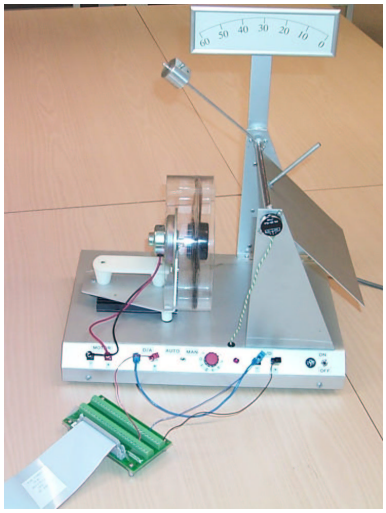
## 2 Optimal control

Karlstad University also offer a number of advanced courses in control intended for graduate or final year undergraduate students covering topics such as digital control, adaptive control and optimal control. There is something inherently satisfying about the nature of optimal control; perhaps it is the combination of elegant theory with the knowledge that no controller will be better — at least in theory.

However the optimal control course has demanding pre-requisites. Most undergraduate text books (e.g. [2–4]) pay scant attention to



(a) Cascaded series of RC filters or 'Blackbox'



(b) Fan and flapper

(c) Toy helicopter

Figure 2: Plants suitable for control laboratories

optimal control, and those that do, [1, 5, 6], primarily restrict themselves to linear optimal control; LQR and LQG. Graduate texts such as [7–9], naturally cover the underlying mathematics competently, but still leave gaps for the implementation although there are rare examples, [10], which manage both the theory and to convey how one could realistically implement such an optimal scheme. The problem is not the author's intention, but rather the inescapable fact that implementing a robust constrained nonlinear multivariable optimiser inside a nonlinear boundary-value problem is a nontrivial computational task. To implement this in realtime, ensuring that when the sampling interval is up you have a sensible control input ready at your disposal, and all in limited, or perhaps even fixed-precision requires serious attention to detail which has little to do with optimal control. On the other hand, without this care, your optimal controller is unlikely to be optimal or even close to it.

## 2.1 The problem of the 'lean period'

A more serious problem in demonstrating many advanced control topics in the laboratory is the requirement for an adequate plant model and state-feedback. In our education, this means that after the students get to control our laboratory plant equipment using PID controllers in a first control course, they cannot manage any more convincing laboratories until they have mastered discrete-time control, system identification (for state-space models), linear optimal control and state estimation. This results in substantial lean period in the education in terms of practical applications and is disturbing for the more practically oriented students. Given their healthy suspicion of simulated responses, without application, they are often unconvinced that this material is worth learning.

So the onus is on the lecturer to provide a demonstration that out-performs competing techniques. First there is the choice of plant. Ideally we would like a multivariable with differentiable nonlinearities preferably in the dynamics (as opposed to Wiener/Hammerstein forms), low noise and no input or output saturation. For practical reasons, we desire time constants in the order of 1–10 seconds. We wish to avoid troublesome nonlinear elements such as stiction, hysteresis and excessive deadtime. In the case of our blackbox (refer Fig. 2(a)), we find it convenient to 'add' the nonlinearities in software using a wrapper around the plant block.

## 2.2 Linear optimal control

Our first optimal control laboratory is to design a Kalman estimator and a linear-quadratic regulator (LQR) controller given a previously identified state-space plant model. The SIMULINK configuration shown in Fig. 3 closely follows any standard textbook diagram such as [5, Fig. 6–5]. An extension to this exercise involves including model adaption and setpoint-following capabilities, [5, p729]. An example of the controlled response is given in Fig. 4. This laboratory highlights the importance of selecting appropriate sampling times, model structure and forgetting factors, and adjusting the control weighting in the optimal formulation to prevent input saturation. Only plant identification is performed in the first 30 samples.

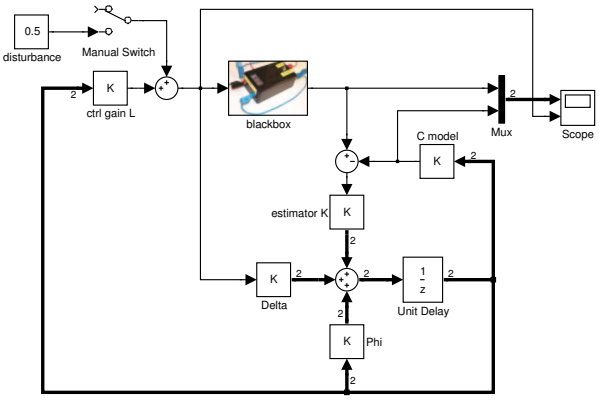



Figure 3: Real-time LQR using observed states for feedback.

## 2.3 Classical optimal control

A good place to start teaching optimal control is with the classical open-loop general optimal control problem if only to exemplify why closed-loop versions based on linear models and quadratic performance indices are so popular. Here we wish to establish inputs $\mathbf{u}(t)$ to min-

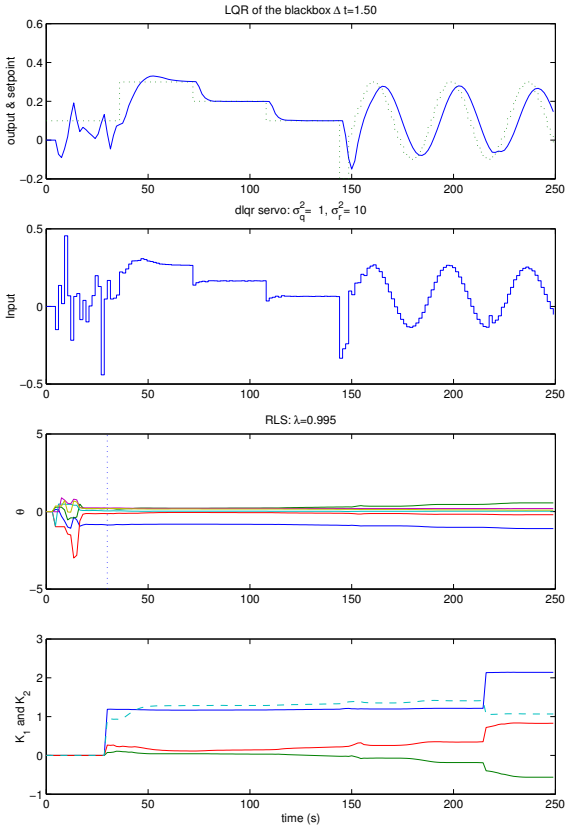



Figure 4: A servo linear quadratic regulator of a laboratory plant with an adaptive RLS plant model. Upper two trends: system output, setpoint and input. Lower two trends: Model parameters and controller gains.

imise the scalar functional

$$\mathcal{J} = \phi\left(\mathbf{x}(t_f)\right) + \int_0^{t_f} L(\mathbf{x}, \mathbf{u}, t)\, dt \qquad (1)$$

given a nonlinear dynamic model

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \qquad \mathbf{x}(t=0) = \mathbf{x}_0 \qquad (2)$$

The solution to Eqn. 1 is obtained through variational calculus to be a two-point boundary value problem,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \quad \mathbf{x}(t_0) = \mathbf{x}_0 \qquad (3)$$

$$\dot{\boldsymbol{\lambda}} = -\frac{\partial H}{\partial \mathbf{x}}, \quad \boldsymbol{\lambda}(t_f) = \left.\frac{\partial \phi}{\partial \mathbf{x}}\right|_{t_f} \qquad (4)$$

where $\boldsymbol{\lambda}$ are the co-state variables, $H \stackrel{\text{def}}{=} L + \boldsymbol{\lambda}^T \mathbf{f}$ is the Hamiltonian and the control input is determined as the solution to

$$\frac{\partial L}{\partial \mathbf{u}} + \boldsymbol{\lambda}^T \frac{\partial \mathbf{f}}{\partial \mathbf{u}} = 0 \qquad (5)$$

Solving the possibly nonlinear two-point boundary value problem of Eqns. 3 and 4 with the embedded algebraic constraint, Eqn. 5 is non-trivial. If input saturation constraints are present, then the method due to Pontryagin is appropriate. The issue becomes even worse if we were to attempt this optimisation every sample time. We could parameterise the control moves, and solve the ensuing optimisation problem, but this is an approximation, and still exhibits excessive computation.

Solving the full two-point boundary value problem using the shooting method does work, but this places excessive limits on the hardware, and works only if you have a good idea for the initial start guess of the costates at $t = 0$. A more robust method is to use the boundary-value code based on collocation, `bvp4c`, present in MATLAB release 12. With 500MHz PCs, we can run this optimal controller (with a linear state-estimator) down to sampling times of around 1 second. This could be substantially improved with better coding and mex s-functions, but has the disadvantage that we lose the transparency of the algorithm.

## 2.4 Dynamic Matrix Control

Dynamic matrix control or DMC is one example of a collection of predictive controllers known as

receding horizon controllers, [11]. For unconstrained linear plants, the controller requires a simple, albeit large matrix inversion. While the basic algorithm is relatively straight forward, the subtleties and the open-loop nature, and the potential for model-plant mismatch mean that the student gains most only after they have programmed it. It is also a good example where it is much easier to develop the algorithm in raw MATLAB as opposed to SIMULINK.

For nonlinear plants, and/or constraints, the controller calculations require an online optimisation calculation. As in the linear case, the optimisation is repeated every sampling interval, but now requires an iterative search rather than a standard matrix inversion. Fig. 5 shows an example of a model predictive controller of the helicopter shown in Fig. 2(c) implemented using the xPC target. While the controlled response is impressive compared to what we can achieve using PID control, [12], the development effort and cost is substantial, limiting this to a final-year project.

### 2.5 Optimal controllers to minimise the absolute error

Optimal control where we try to minimise the absolute sum of the errors, rather than the classical squared sum results in a linear program, [13]. This approach may be more robust is certainly more natural and it has the added advantage that we can directly take into account saturation in the input variables, or indeed any linear constraint condition.

The optimisation problem is then to choose the set of $N$ future manipulated variables $\mathbf{u}_k$ such that the performance index

$$\mathcal{J} = \sum_{k=0}^{N-1} \sum_{i=1}^{n} |r_i - x_i| \qquad (6)$$

is minimised subject to the discrete process model, $\mathbf{x}_{k+1} = \mathbf{\Phi}\mathbf{x}_k + \mathbf{\Delta}\mathbf{u}_k$ of order $n$. This can be formulated as an admittedly largish linear program for even modest sized problems. Fig. 6 shows a view of the constraint matrix for a 3 input/3 output system with a control horizon of 10 samples. We could use the LP solver `lp.m` from the optimisation toolbox, (now updated to `linprog`), but we note that `lp` does not employ sparse techniques. The lower figure shows the resulting optimal trajectory highlighting the acausal behaviour when using a receding horizon controller knowing future setpoint changes.

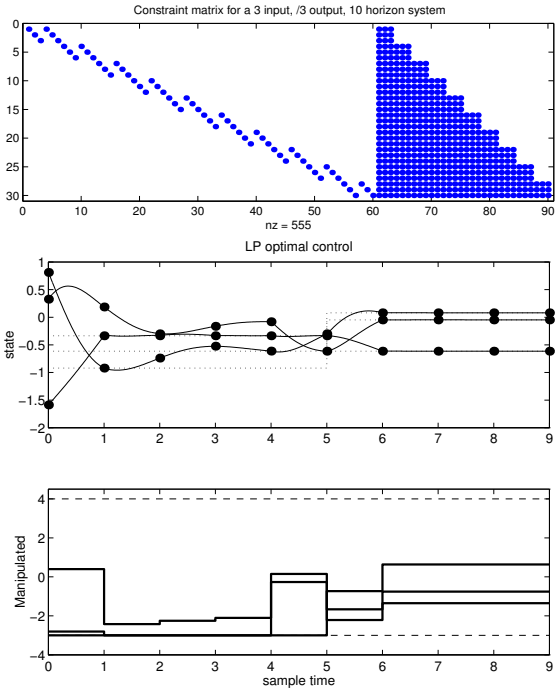



Figure 6: Upper: The constraint matrix (viewed with `spy`) from a linear program used as a optimal control strategy. Lower: Acausal behaviour with input constraints.

## 3 Conclusions

The teaching of automatic control at Karlstad University has benefited from the incorporation of MATLAB, SIMULINK and the real-time toolbox. The main problem we now face in our second course in the subject is that until the students are able to identify discrete state-space models and design observers or Kalman filters, they cannot apply modern control techniques such as LQG or GPC to physical plants. Even once the students manage to demonstrate such optimal control techniques, it is still problematic to convince oneself that the response is actually optimal in any sense.

Despite these difficulties, at the end of our second 5-week control course the students are able to develop advanced control algorithms and ap-

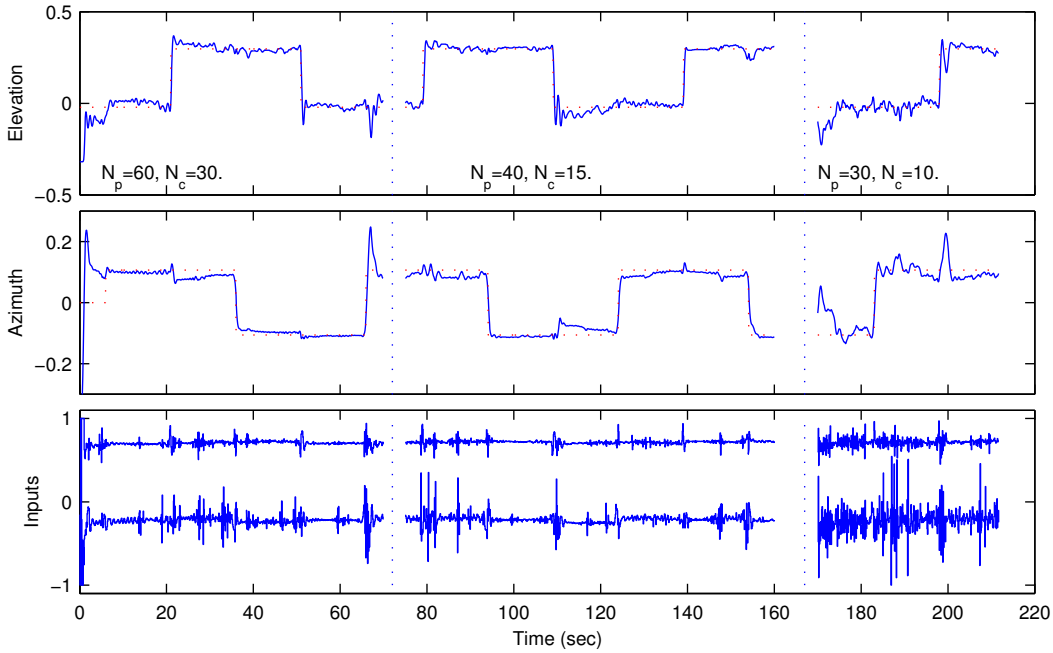



Figure 5: Closed-loop response of the helicopter using a predictive control algorithm with different prediction and control horizons. Data from [12].

ply them to physical plants, something unthinkable even just a few years ago. We do this entirely in MATLAB and some toolboxes, without needing to re-code routines in C or use a compiler.

**Acknowledgements** Thanks are due to Jonas Balderud for the implementation of the helicopter MPC controller.

# References

[1] Karl-Johan Åström and Björn Wittenmark. *Computer-Controlled Systems: Theory and Design*. Prentice–Hall, 3 edition, 1997.

[2] Richard C. Dorf and Robert H. Bishop. *Modern Control Systems*. Addison–Wesley, 8 edition, 1998.

[3] Benjamin C. Kuo. *Automatic Control Systems*. Prentice–Hall, 7 edition, 1995.

[4] Norman S. Nise. *Control Systems Engineering*. Benjamin/Cummings, 2 edition, 1995.

[5] Katsuhiko Ogata. *Discrete Time Control Systems*. Prentice–Hall, 1987.

[6] Thomas Kailath. *Linear Systems*. Prentice–Hall, 1980.

[7] Thomas L. Vincent and Walter J. Grantham. *Nonlinear and Optimal Control Systems*. John Wiley & Sons, 1997.

[8] C.K. Chui and G. Chen. *Linear Systems and Optimal Control*. Springer–Verlag, 1989.

[9] Enid R. Pinch. *Optimal Control and the Calculus of Variations*. Oxford University Press, 1993.

[10] Jr. Arthur E. Bryson. *Dynamic Optimization*. Addison–Wesley, 1999.

[11] C.R. Cutler and B.L. Ramaker. Dynamic matrix control—A computer control algorithm. In *Joint Automatic Control Confr.*, volume 1, pages wp5–B. IEEE, August 1980.

[12] Jonas Balderud and David Wilson. Predictive control of a toy helicopter. In *American Control Conference*, Alaska, USA, 8–10 May 2002. *Submitted*.

[13] Tore K. Gustafsson and Pertti M. Mäkilä. *$L_1$ Identification Toolbox for Matlab*. Åbo Akademi, Finland, August 1994. Ftp: `ftp.abo.fi/pub/rt/l1idtools`.