Velocity-Based Gain-Scheduling Demo

D.J.Leith, W.E.Leithead
Department of Electronics & Electrical Engineering,
University of Strathclyde, Glasgow G1 1QE
Tel 0141 548 2407, Fax. 0141 548 4203 Email. doug@icu.strath.ac.uk

1. Introduction

This demonstration provides an example of the application of the velocity-based framework to analyse the dynamics of a simple nonlinear system and to design a nonlinear controller which achieves specified closed-loop performance requirements. The velocity-based gain-scheduling approach is a general framework for the analysis and design of nonlinear systems. This framework retains the continuity with linear methods which is the principle advantage of conventional gain-scheduling methods. However, in contrast to conventional methods, the velocity-based gain-scheduling approach is not restricted to near equilibrium operation and does not involve any form of inherent slow variation requirement whatsoever; for example, global dynamic inversion controllers may be designed. The velocity-based approach generalises the conventional series expansion linearisation about an equilibrium point. A linear system, namely the velocity-based linearisation, is associated with every operating point of a nonlinear system, both equilibrium and non-equilibrium. The resulting velocity-based linearisation family is an alternative representation of the nonlinear system which involves no loss of information; for example, the solutions to the members of the velocity-based linearisation family may be pieced together to approximate arbitrarily accurately the solution to the nonlinear system.

2. Nonlinear Plant

Consider the nonlinear plant with dynamics

$$\dot{x} = G(\rho), \qquad y = x$$

where $\rho = r - 10x$, $G(s)=\tanh(s)+0.01s$. At an equilibrium operating point,

$$G(\rho_o)=0$$

which requires that

$$\rho_o = r_o - 10\, x_o = 0$$

Hence, the series expansion linearisation of the plant relative to the equilibrium operating point, (r_o, x_o, y_o), is

$$\delta\dot{x} = -10\nabla G(0)\delta x + \nabla G(0)\delta r, \quad \delta y = \delta x$$

$$\delta r = r - r_o, \delta x = x - x_o, y = \delta y + y_o$$

where $\nabla G(s) = 1.01 - \tanh^2 s$. The series expansion linearisation is the *same* at every equilibrium operating point.

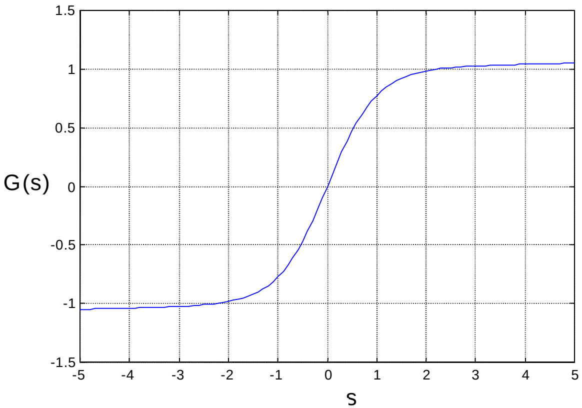
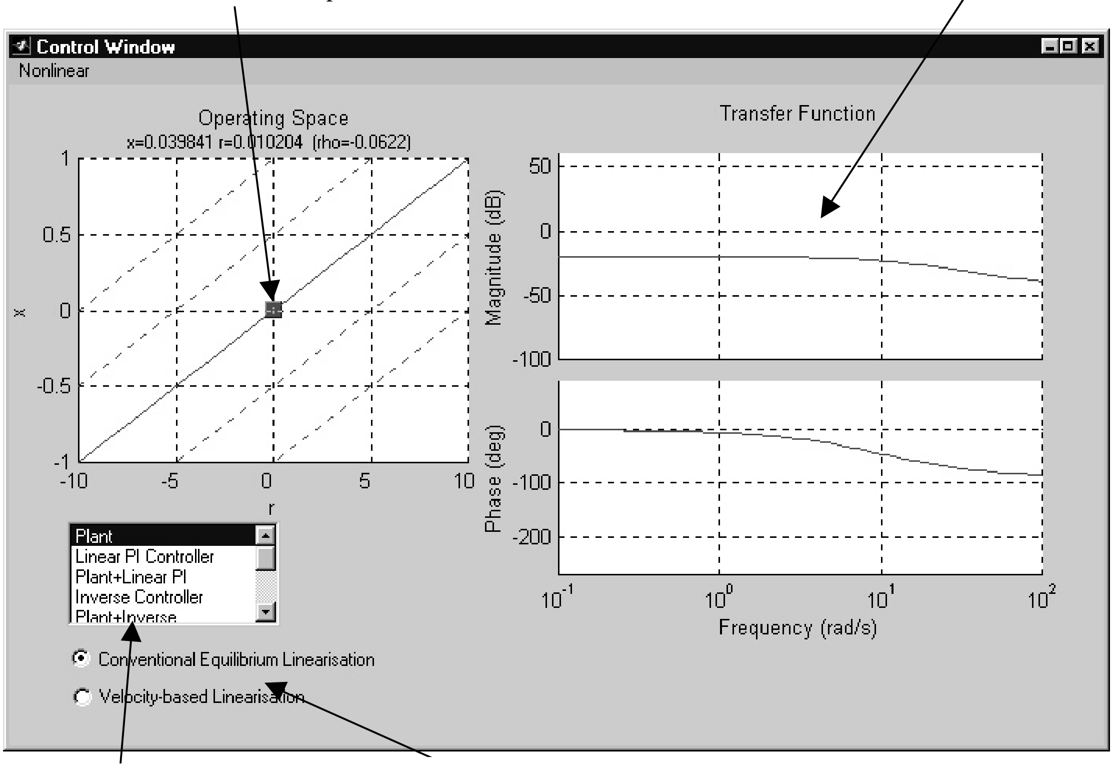


Figure 1 – Plot of G(s)

To begin the demo, start MATLAB and then type `Example` followed by <`enter`>. The Control Window shown in figure 2 will appear (without the annotation).



● Conventional Equilibrium Linearisation

○ Velocity-based Linearisation

Current operating point of nonlinear plant- altered by clicking and dragging marker with the mouse. Note, solid red line marks the equilibrium points of the plant and dashed red lines mark parallel contours.

Bode plot of the transfer function of the linearisation associated with the current operating point.

Selects system considered.

Selects type of linearisation: either conventional series expansion linearisation about an equilibrium point or the velocity-based linearisation (which exists for every operating point, not just equilibrium points).

Figure 2 – Control Window

By moving the marker to a number of different equilibrium points (points lying on the solid red line in the operating space display), it can be seen that the transfer function of the series expansion linearisation is indeed the same at every equilibrium point.

3. Linear PI Control

The requirement is to design a controller such that the closed-loop system has a rise time of around 0.5 seconds with less than 25% overshoot in response to demanded step changes in the plant output, y, of magnitude less than 100 units.

The conventional gain-scheduling approach is to design a linear controller on the basis of the series expansion linearisation of the plant about a particular equilibrium operating point. This is repeated for a number of equilibrium points in order to obtain a number of linear controllers. These linear controllers are interpolated, in some appropriate manner, to obtain a continuous linear controller family. A nonlinear controller realisation is then selected which inherits, in some sense, the dynamic characteristics of the members of the linear controller family.

Based on the conventional series expansion linearisation of the plant at an equilibrium operating point, an appropriate local controller is the PI-type controller

$$\begin{bmatrix}\delta\dot{x}_{c_1}\\ \delta\dot{x}_{c_2}\end{bmatrix}=\begin{bmatrix}-50 & 0\\ 1 & 0\end{bmatrix}\begin{bmatrix}\delta x_{c_1}\\ \delta x_{c_2}\end{bmatrix}+\begin{bmatrix}50\\ 0\end{bmatrix}\delta e,\quad \delta r=\begin{bmatrix}K_0 & K_1\end{bmatrix}\begin{bmatrix}\delta x_{c_1}\\ \delta x_{c_2}\end{bmatrix} \tag{1}$$

$$\delta e = e - e_o,\, r = \delta r + r_o \tag{2}$$

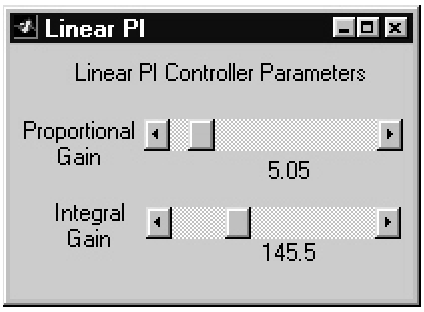
with $e=y_{ref}-y$, $K_o=4.0$, $K_1=100.0$. The transfer function of the above linear controller is $\left(K_o+\frac{K_1}{s}\right)\frac{50}{s+50}$. In this example, the plant dynamics are the same at every equilibrium operating point and so the same controller dynamics, (1), may be employed at every equilibrium operating point. It should be noted that whilst the controller dynamics, (1), are the same at every equilibrium point, the input/output transformations, (2), may vary with the equilibrium operating point.

It remains to determine a controller realisation corresponding to the family of linear controllers. Owing to the integral action in the controller, e_o is zero and r_o is implicitly generated by the feedback. Hence, on the basis of the family of linearisations at the equilibrium operating points, a linear PI-type controller seems to be appropriate; namely,

$$\begin{bmatrix}\dot{x}_{c_1}\\ \dot{x}_{c_2}\end{bmatrix}=\begin{bmatrix}-50 & 0\\ 1 & 0\end{bmatrix}\begin{bmatrix}x_{c_1}\\ x_{c_2}\end{bmatrix}+\begin{bmatrix}50\\ 0\end{bmatrix}e,\quad r=\begin{bmatrix}K_0 & K_1\end{bmatrix}\begin{bmatrix}x_{c_1}\\ x_{c_2}\end{bmatrix}$$

Since the series expansion linearisation of the plant is the same at every equilibrium operating point, it is perhaps unsurprising that the conventional gain-scheduling approach should lead to a linear design of controller.

In the Control Window, choose Plant+Linear PI to analyse. An additional window will appear which permits the controller parameters K_o and K_1 to be specified (see figure 3) and the transfer function displayed will change to show the closed-loop dynamics of the series expansion linearisation of the plant with the linear PI controller. Adjust the controller parameters to obtain a satisfactory closed-loop transfer function with bandwidth around 5 rad/s, peak less than 6 dB (*e.g.* $K_o=5$, $K_1=50$).



Linear PI

Linear PI Controller Parameters

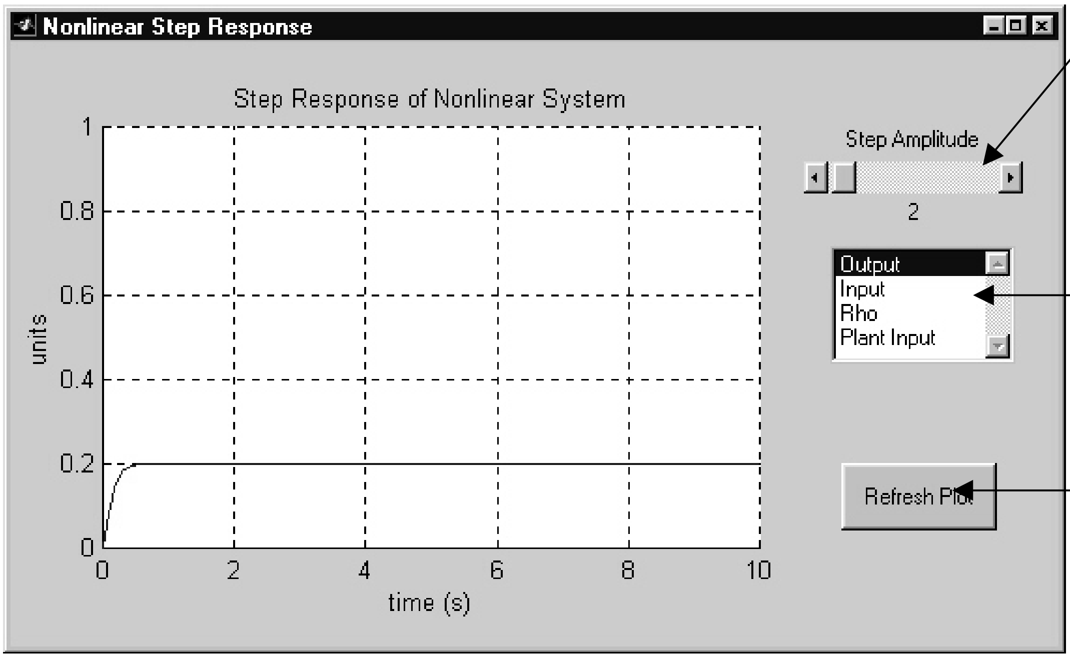
Proportional Gain 5.05

Integral Gain 145.5

Figure 3 – Linear PI controller parameter window

Recall that this transfer function only describes the dynamics in the vicinity of an equilibrium point. To see the step response of the nonlinear plant when the linear PI controller is employed, from the Nonlinear menu choose Step Response. A new window will appear showing the step response (see figure 4). The magnitude of the step input applied can be varied by adjusting the slider on this window. It can be seen that for inputs greater than around 0.3 units, the PI controller is unable to achieve the overshoot requirements.

Exercise Vary the controller parameters to see if performance specification can be satisfied.



sets amplitude of step input

quantity to be displayed

refreshes the step response after changes to the controller or controller parameters

Figure 4 Step response window

4. Velocity-based linearisation

In order to incorporate information about the plant dynamics at non-equilibrium operating points into the controller design, reformulate the nonlinear plant, by differentiating, as

$$\dot{w} = -10\nabla G(\rho)w + \nabla G(\rho)\dot{r}, \qquad \dot{y} = w \tag{3}$$

where $\rho = r - 10x$. The velocity-based linearisation associated with the operating point at which ρ equals ρ_1 is simply the corresponding "frozen" form of (3)

$$\dot{\hat{w}} = -10\nabla G(\rho_1)\hat{w} + \nabla G(\rho_1)\dot{r}, \qquad \dot{y} = \hat{w}$$

where $\rho_1 = r_1 - 10x_1$. Recalling that ρ equals zero at the equilibrium operating points, it can be seen that the velocity-based linearisation at an equilibrium operating point is directly related to the conventional series expansion linearisation at an equilibrium point. However, in contrast to the conventional linearisation, a velocity-based linearisation is defined for every operating point, not just the equilibrium points. The solution to a particular velocity-based linearisation approximates the solution to the nonlinear system in the vicinity of the relevant operating point. As the nonlinear system moves from one operating point to another, the solution can be approximated arbitrarily accurately by piecing together the solutions to appropriate velocity-based linearisations.

Select Plant and Velocity-based Linearisation in the Control Window to display the transfer function of the plant velocity-based linearisation associated with the current operating point. Since a velocity-based linearisation is defined for every operating point, it is now possible to move the operating point marker off the equilibrium manifold. It can be seen that the bandwidth of the plant velocity-based linearisation rapidly decreases as the operating point moves away from the equilibrium manifold and ρ increases (it can be seen from figure 1 that the slope, or "gain", of the plant nonlinearity rapdily decreases as ρ changes from zero). Selecting Plant+Linear PI, the variation in the transfer function of the velocity-based linearisation of the closed-loop system is displayed. It can be seen that the bandwidth rapidly decreases as ρ increases and a large resonance peak develops in the transfer function. The velocity-based linearisation thereby provides insight into the failure of the linear PI controller.

5. Velocity-based Gain-scheduling

The velocity-based gain-scheduling methodology can be employed to incorporate information regarding the plant dynamics away from equilibrium points into the controller design. This design approach involves the following steps

1. Determine the velocity-based linearisation family associated with the nonlinear plant dynamics.
2. Based on the plant velocity-based linearisation family, determine the required controller velocity-based linearisation family such that the resulting closed-loop family achieves the performance requirements. Since each member of the plant family is linear, conventional *linear* design methods can be utilised to design each corresponding member of the controller family.
3. Realise a nonlinear controller corresponding to the family of linear controllers designed at step 2. The velocity-based form of the controller can be obtained directly from the family of linear controllers by simply permitting the ρ to vary with the operating point.

This design procedure retains a divide and conquer philosophy and maintains the continuity with linear design methods which is an important feature of the conventional gain-scheduling approach. However, in contrast to the conventional gain-scheduling approach, the resulting nonlinear controller is valid throughout the operating envelope of the plant, not just in the vicinity of the equilibrium operating points. This extension is a direct consequence of employing the velocity-based linearisation framework rather than the conventional series expansion linearisation about an equilibrium operating point.

5.1 Velocity-based Dynamic Inversion Control

A dynamic inversion controller is obtained when, in step 2, the controller linearisation family is designed such that the closed-loop combination of each plant velocity-based linearisation with the corresponding controller velocity-based linearisation has the same transfer function at every operating point and, in addition, the state of the controller is selected appropriately. This is a direct generalisation of linear pole-zero inversion and is quite distinct from Input-Output Linearisation. Similarly to the linear case, when the relative degrees of the plant velocity-based linearisations are greater than zero (in the SISO case, number of poles is greater than the number of zeroes), the dynamic inverse is unrealisable. However, an arbitarily accurate realisable approximation can be obtained by augmenting the exact inverse with sufficiently fast dynamics (analagous to adding high frequency poles in the linear case).

In the present example, the relative degree of the plant is unity and an approximate velocity-based dynamic inverse is

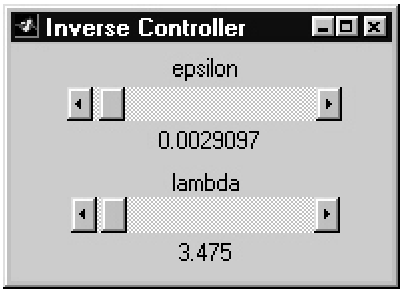
$$\dot{w}^{\mathrm{i}} = -(10+\frac{1}{\varepsilon})\nabla G(\rho) w^{i} - \frac{\nabla G(\rho)}{\varepsilon}\dot{v}, \qquad \dot{r} = \frac{1}{\varepsilon} w^{\mathrm{i}} + \frac{1}{\varepsilon}\dot{v} \tag{4}$$

where v is an auxiliary input and ε determines the fast dynamics included to make the inverse realisable. As ε tends to zero, the approximate inverse tends to the exact inverse in the sense that the output $\dot{y}$ of the plant tends to the input $\dot{v}$. The system, (4), is essentially an open-loop inverse of the plant. In order to reject stready disturbances integral feedback is employed

$$\dot{v} = \lambda(y_{ref} - y) \tag{5}$$

where y_{ref} is the signal which the plant output is required to track and λ determines the speed of the error dynamics. Recall that the velocity-based linearisations of the approximate inverse are obtained by simply "freezing" the dynamics, (4)-(5).

In the Control Window, set the system to Plant+Inverse. An additional window will appear which permits the controller parameters ε and λ to be specified (see figure 5) and the transfer function displayed will change to show the closed-loop dynamics of the velocity-based linearisation of the plant with that of the inverse controller. To see the step response of the nonlinear closed-loop system, from the Nonlinear menu choose Step Response. A new window will appear showing the step response (see figure 4). The magnitude of the step input applied can be varied by adjusting the slider on this window. Adjust the controller parameters to obtain a satisfactory step response (e.g. ε=0.001, λ=5).



epsilon
0.0029097
lambda
3.475

Figure 5 - Inverse controller parameter window

5.2 Blended Multiple Model Approximation of Plant

The velocity-based gain-scheduling approach is not confined to dynamic inversion but may also be employed to design controllers when the desired closed-loop dynamics are nonlinear. In step 2 of the velocity-based gain-scheduling design procedure, the velocity-based linearisation family of the controller is determined based on the velocity-based linearisation family of the plant. Of course, the plant and controller linearisation families have an infinite number of members. The controller family can be determined directly (as in the dynamic inversion situation) when an analytic desription of the plant is available However, an alternative is to design the controller velocity-based linearisation for only a small (finite) number of members of the plant linearisation family and these controller linearisations are then interpolated to obtain a continuous controller velocity-based linearisation family.

Rather than choosing an arbitrary controller interpolation (such as linear interpolation of the controller state-space matrices), an appropriate blended multiple model based representation of the plant is first determined. The velocity-based linearisations of the plant at a small number of operating points ("local models") are interpolated ("blended") to obtain an approximate plant velocity-based linearisation family. It should be noted that this representation is distinct from other blended multiple model representations proposed in the literature in that the local models are *linear* (rather than affine) and the solution to the overall system is *directly* related to the solutions to the local models (simply the interpolation of the solutions to the local models). In the present example, an appropriate approximate velocity-based linearisation family of the plant, which uses only two local models, is

$$\dot{\hat{w}} = \sum_{i=0}^{1}\left(-10\nabla G(\rho_i)\hat{w} + \nabla G(\rho_i)\dot{r}\right)\mu_i(\rho), \qquad \dot{\hat{y}} = \hat{w} \qquad (6)$$

where $\rho_o = 0$ ("equilibrium"), $\rho_1=5$ ("far from equilibrium") and the weighting functions, μ_i, are Gaussian-based and normalised so that $\sum_{i=0}^{1}\mu_i = 1$ (see figure 6).

In the Control Window, set the system to Approx Plant. The transfer function displayed will change to show that of the approximate velocity-based linearisation family, (6), of the plant. Changing the setting to Approx Error displays the difference between the transfer function of the exact and approximate plant velocity-based linearisations.

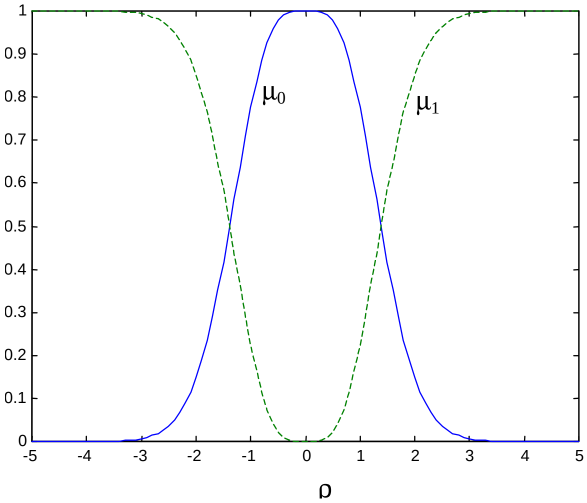


Figure 6 – Weighting functions used in blended multiple model of plant and in velocity-based gain-scheduled controller

5.3 Velocity-based Gain-scheduled Control

A velocity-based gain-scheduled controller can be designed by first determining a linear controller for each of the local models in the approximate blended multiple model of the plant. These linear controllers are then interpolated (using the same weighting functions μ_i as in the plant model) to obtain a continuous controller velocity-based linearisation family. In the present example, only two local models are required and the gain-scheduling design is particularly straightforward. Employing a PI-type controller structure once again, the controllers for, respectively, the ρ_o=0 and ρ_1=5 are

$$\begin{bmatrix} \dot{w}_{c_1} \\ \dot{w}_{c_2} \end{bmatrix} = \begin{bmatrix} -50 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} w_{c_1} \\ w_{c_2} \end{bmatrix} + \begin{bmatrix} 50 \\ 0 \end{bmatrix} \dot{e}, \quad \dot{r} = \begin{bmatrix} K_0^{\,0} & K_1^{\,0} \end{bmatrix} \begin{bmatrix} w_{c_1} \\ w_{c_2} \end{bmatrix}$$

$$\begin{bmatrix} \dot{w}_{c_1} \\ \dot{w}_{c_2} \end{bmatrix} = \begin{bmatrix} -50 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} w_{c_1} \\ w_{c_2} \end{bmatrix} + \begin{bmatrix} 50 \\ 0 \end{bmatrix} \dot{e}, \quad \dot{r} = \begin{bmatrix} K_0^{\,1} & K_1^{\,1} \end{bmatrix} \begin{bmatrix} w_{c_1} \\ w_{c_2} \end{bmatrix}$$

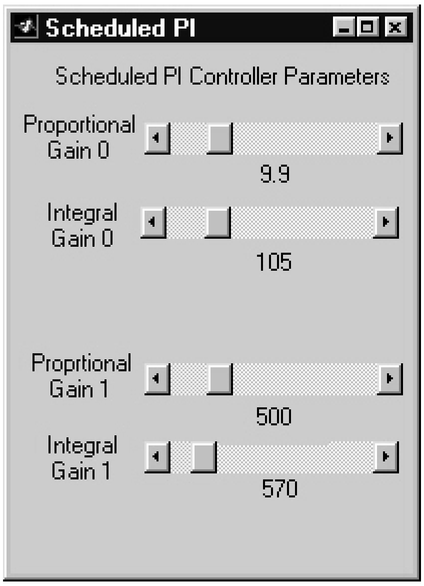
where e=y_{ref}-y. The corresponding scheduled nonlinear controller, obtained by interpolating between these two controllers using the plant weighting functions, is

$$\begin{bmatrix} \dot{w}_{c_1} \\ \dot{w}_{c_2} \end{bmatrix} = \begin{bmatrix} -50 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} w_{c_1} \\ w_{c_2} \end{bmatrix} + \begin{bmatrix} 50 \\ 0 \end{bmatrix} \dot{e}, \quad \dot{r} = \begin{bmatrix} K_0(\rho) & K_1(\rho) \end{bmatrix} \begin{bmatrix} w_{c_1} \\ w_{c_2} \end{bmatrix} \qquad (7)$$

where $K_0 = \sum_{i=0}^{1} K_0{}^i \mu_i(\rho), \quad K_1 = \sum_{i=0}^{1} K_1{}^i \mu_i(\rho)$. The velocity-based linearisations of the controller are simply the "frozen" forms of (7). The controller, (7), cannot be implemented directly since $\dot{e}$ depends on $\dot{y}$ which is not measured. However, the controller may be reformulated equivalently in the realisable form

$$\dot{w}_{c_2} = -50 w_{c_2} + 50 e, \quad \dot{r} = K_0(\rho)\left(-50 w_{c_2} + 50 e\right) + K_1(\rho) w_{c_2}$$

In the Control Window, set the system to Plant+Scheduled PI. An additional window will appear which permits the controller parameters K_0^0, K_1^0, K_0^1 and K_1^1 to be specified (see figure 7) and the transfer function displayed will change to show the closed-loop dynamics of the velocity-based linearisation of the plant with that of the gain-scheduled controller. Adjust the controller parameters to obtain a satisfactory closed-loop transfer functions when ρ≈0 and ρ≈5 (bandwidth around 5 rad/s, peak value less than 6dB). It should be noted that the transfer function of the closed-loop velocity-based linearisation will vary with ρ even when it is the same at both ρ=0 and ρ=5. This arises because of the rather crude interpolation employed in this example and the variation can be reduced by finer interpolation between a larger number of controller designs. To see the step response of the nonlinear closed-loop system, from the Nonlinear menu choose Step Response. A new window will appear showing the step response (see figure 4). The magnitude of the step input applied can be varied by adjusting the slider on this window. If necessary, adjust the controller parameters further to obtain a satisfactory step response (e.g. $K_0^0 = 10, K_1^0 = 100, K_0^1 = 500, K_1^1 = 550$).



Scheduled PI

Scheduled PI Controller Parameters

Proportional Gain 0: 9.9

Integral Gain 0: 105

Proprtional Gain 1: 500

Integral Gain 1: 570

Figure 7- Scheduled PI controller parameter window

6. Conclusions

For more details, see

LEITH, D.J., LEITHEAD, W.E., 1996, Appropriate Realisation of Gain Scheduled Controllers with Application to Wind Turbine Regulation. *International Journal of Control*, **65**, 223-248.
LEITH, D.J., LEITHEAD, W.E., 1998a, Gain-Scheduled & Nonlinear Systems: Dynamic Analysis by Velocity-Based Linearisation Families. *Int. J. Control*, **70**, 289-317.
LEITH, D.J., LEITHEAD, W.E., 1998b, Gain-Scheduled Controller Design: An Analytic Framework Directly Incorporating Non-Equilibrium Plant Dynamics. *Int. J. Control*, **70**, 249-269.
LEITH, D.J., LEITHEAD, W.E., 1998c, Input-Output Linearisation by Velocity-based Gain-Scheduling. *Int. J. Control*, submitted.
LEITH, D.J., LEITHEAD, W.E., 1998d, Gain-scheduled Control of a Skid-to-Turn Missile: Relaxing Slow Variation Requirements by Velocity-Based Design. *J. Guidance, Control & Dynamics*, submitted.
LEITH, D.J., LEITHEAD, W.E., 1998e, Analytic Framework for Blended Multiple Model Systems Using Linear Local Models. *Int. J. Control*, to appear.
LEITH, D.J., LEITHEAD, W.E., 1998f, Modelling of Nonlinear Integrated Systems: A Velocity-Based Framework Supporting Modular Linearisation-Based Analysis And Design. Internal Report, Industrial Control Centre, University of Strathclyde.

See also reports at http://www.icc.strath.ac.uk/~doug/reports.html

Acknowledgement

D.J.Leith acknowledges the generous support of the Royal Society for the work presented.

Appendix – Installation

Demo requires MATLAB 5.1 or better (but is otherwise standalone and does not require any of the MATLAB toolboxes). Installation is straightforward: create a new directory, copy the demo files there and set the MATLAB path to include the new directory. The demo files are

Example.m
StepResponse.m
LinearPIParas2.m
InvParas.m
ScheduledPIParas.m
Workshop.m
PlantFn.m
PIFn.m
PlantPIFn.m
PlantInverseFn.m
PlantSchedPIFn.m
Example.mat
StepResponse.mat
StepResponse.mat
LinearPIParas2.mat
InvParas.mat
ScheduledPIParas.mat
workshop-demo.pdf