ROBUSTNESS STUDIES OF THE FEEDBACK LINEARIZATION METHOD

by

John Mikal Størdal

Sivilingeniør Noregs Tekniske Høgskule (1988)

Submitted to the Department of
Aeronautics and Astronautics in Partial Fulfillment of
the Requirements for the
Degree of

MASTER OF SCIENCE
in Aeronautics and Astronautics

at the

Massachusetts Institute of Technology

June 1992

Signature of Author _____
Department of Aeronautics and Astronautics
May 18, 1992

Certified by _____
Lena Valavani
Associate Professor
Department of Aeronautics and Astronautics
Thesis Supervisor

Accepted by _____
Harold Y. Wachman
Professor
Department of Aeronautics and Astronautics
Chairman, Department Graduate Committee

ROBUSTNESS STUDIES OF THE FEEDBACK

LINEARIZATION METHOD

by

John Mikal Størdal

Submitted to the Department of Aeronautics and Astronautics

on June 1, 1992

in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE in Aeronautics and Astronautics

Control designs based on the feedback linearization method are applied to a variety of second order nonlinear dynamic systems and the resulting performance is compared to more classical approaches utilizing Taylor linearization models. A number of different nonlinear effects have been studied such as Coulomb friction, piecewise linear functions, mass on nonlinear spring (Duffing's equation), the Mathieu equation etc. Robustness of the methods with respect to both structured- and unstructured model errors is explored. Structured modeling error is simulated by introducing a mismatch between the nominal parameter value and the actual parameter value. Unstructured modeling errors are simulated by introducing low damped high order dynamics in the model simulating the actual plant. The effect of measurement error is also studied. In the design of the controllers effort has been focused into putting them on comparable specification with respect to bandwith, dc gain etc.

The comparison is done by Monte Carlo simulations of the different nonlinear systems with the two different classes of controllers. A performance measure is defined and the results compared.

To Mona

Acknowledgements

I would first and foremost like to thank Professor Lena Valavani, my thesis supervisor, for all the help along the way. Her support on many fronts has made things easier.

Thanks also to my good friends Mauricio, Jose, Mark, Mak, Dave. As a source of support, entertainment and good company you were/are invaluable.

My stay at MIT was funded by Statens Lånekasse for Utdanning and the U.S. Educational Foundation in Norway in partnership with the Institute of International Education which awarded me a Fulbright Scholarship, with special thanks to Mrs. Barbara Lysholt Pedersen and Patricia DeShong.

Contents

List of Figures

Nomenclature

M	uppercase letter denotes a matrix
v	lowercase boldface letter denotes a vector
$\nabla(.)$	gradient
$L_f(.)$	Lie derivative
[,]	Lie bracket
R^n	n-dimensional Euclidian vector space
$\hat{p}$	nominal parameter value
TL-LQG/LTR	Taylor Linearization Linear Quadratic Gaussian Loop Transfer Recovery Controller
FL-LQG/LTR	Feedback Linearization Linear Quadratic Gaussian Loop Transfer Recovery Controller

1. Introduction

This thesis studies via examples the robustness of the feedback linearization method. Using Monte Carlo simulations of well known, second order, nonlinear dynamic systems, robustness with respect to both structured- (or parametric-) and unstructured- (unmodeled dynamics) modeling uncertainties is studied. A performance measure is defined and the results based on the feedback linearization method are compared to a more classical approach using Taylor series linearized models and linear control design algorithms. A variety of different nonlinear effects such as nonlinear springs, Coulomb friction, discontinuous functions, saturation, etc. are included.

1.1. Motivation and Background

Feedback linearization is a control design (and linearization) method which has attracted a great deal of interest in recent years. Necessary and sufficient conditions for (local) feedback linearizability as well as a general method of construction of the linearizing transformations have been obtained (See [1] and [2] for a summary of the results obtained). The central idea of the approach is to algebraically transform a nonlinear system into a (fully or partly) linear one; linear control techniques can then also be applied in an 'outer loop' setting.

The method is especially appealing from the point of view of control design. One way to control such systems is to build a controller for the equivalent linear system and use the linearizing transformation to obtain the nonlinear controller in terms of the original (physical) coordinates. The method has been used successfully to address some practical control problems (control of helicopters, high performance aircraft, industrial robots etc) [2] and [4].

On the other hand, there are some shortcomings and limitations associated with the approach. One of the most important problems is that the feedback linearization procedure relies on exact cancellation of nonlinear terms and, therefore, highly accurate models are required. There exist theories for constructing controllers which, given a bound on the model uncertainty, guarantee that the state of the system enters and remains within a neighborhood of the equilibrium state after a finite interval of time. This result, however, is based on some very restrictive assumptions on the structure of the uncertainty. These are the so-called structure matching conditions (See [5], [6], [7], and [8] for details). Other important limitations are that the method cannot be used for all nonlinear systems (not all classes are feedback linearizable) and, that in general, the full state has to be measured. Throughout this study it is assumed ,where necessary, that the state vector is available.

Discussion often occurs whether this approach is better (in some sense) than methods based on Taylor linearization of the nonlinear system.The motivation for this study is to compare the performance of the two approaches for a number of examples when model uncertainties are present. .

1.2. Overview of Thesis

The report is structured so that the introductory material briefly discusses the derivation of the feedback linearization method together with some linear control system design tools. The derivation is done to an extent needed for the latter. Later the simulation environment and the simulation plan is described. Finally the simulation results are presented. A brief sketch of the contents of the thesis is given below.

Chapter 2	introduces the feedback linearization method together with some tools from differential geometry such as Lie algebra. Furthermore the LQG/LTR method is briefly discussed here.
Chapter 3	presents the different nonlinear systems together with their equations, that are used in the simulations.
Chapter 4	takes a look at the simulation procedure.
Chapter 5	presents the results of the simulations.

2. Mathematical Preliminaries

First a brief summary of the necessary mathematical tools is given. Then the theory of feedback linearization is outlined to the extent needed in the latter. Finally the LQG/LTR method is presented. This method is used for designing linear controllers.

2.1. Mathematical Notation and Tools

In order to develop the feedback linearization theory, we need some mathematical tools from differential geometry. This section gives a brief overview of the tools needed.

A *vector function*

$$\mathbf{f}: R^n \rightarrow R^n \tag{2.1.1}$$

is called a *vector field* in R^n. To every vector function **f** there corresponds a field of vectors in an n-dimensional space. It is called a *smooth vector field* if **f** has continuous partial derivatives of any required order. Only smooth vector fields will be considered here.

Let h(**x**) be a scalar function of the state **x**, the *gradient* of h is then denoted as $\nabla(h)$ and is represented by the row-vector

$$\nabla(h) = \frac{\partial h}{\partial \mathbf{x}} = [\frac{\partial h}{\partial x_1}, \cdots, \frac{\partial h}{\partial x_n}] \tag{2.1.2}$$

Let **f**(**x**) be a vector field, the *Jacobian* of **f** is then represented by by the nxn matrix

$$\nabla(\mathbf{f}) = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{2.1.3}$$

The *Lie derivative* of a scalar function h with respect to a vector field **f** is a new scalar

function defined by

$$L_{\mathbf{f}} h = \nabla(h)\, \mathbf{f} = \left[\frac{\partial h}{\partial x_1} \cdots \frac{\partial h}{\partial x_n}\right] \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix} \tag{2.1.4}$$

Repeated Lie derivatives can be defined recursively as

$$\begin{aligned} L_{\mathbf{f}}^0 h &= h \\ L_{\mathbf{f}}^i h &= L_{\mathbf{f}}(L_{\mathbf{f}}^{i-1} h) = \nabla(L_{\mathbf{f}}^{i-1} h)\, \mathbf{f} \end{aligned} \tag{2.1.5}$$

Let **g** be another vector field on R^n, then the *Lie bracket* of **f** and **g** is a third vector field defined by

$$[\mathbf{f}, \mathbf{g}] = ad_{\mathbf{f}}\, \mathbf{g} = \nabla(\mathbf{g})\, \mathbf{f} - \nabla(\mathbf{f})\, \mathbf{g} \tag{2.1.6}$$

Repeated Lie brackets can be defined recursively as

$$\begin{aligned} L_{\mathbf{f}}^0 h &= h \\ L_{\mathbf{f}}^i h &= L_{\mathbf{f}}(L_{\mathbf{f}}^{i-1} h) = \nabla(L_{\mathbf{f}}^{i-1} h)\, \mathbf{f} \end{aligned} \quad i=1,2,\cdots \tag{2.1.5}$$

A linearly independent set of vector fields $\{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n\}$ is said to be *involutive* if, and only if, there are scalar functions α_{ijk}: $\mathbf{R}^n \rightarrow \mathbf{R}$ such that

$$[\mathbf{f}_i, \mathbf{f}_j] = \sum_{k=1}^{m} \alpha_{ijk}(\mathbf{x})\, \mathbf{f}_k(\mathbf{x}) \qquad \forall\, i, j \tag{2.1.6}$$

Furthermore we need the definition of a special coordinate transformation called a *diffeomorphism.*. A function Φ is called a diffeomorphism (transformation) in $R^n \supseteq S$ if it is smooth and invertible. If $S=R^n$ then then it's called a *global diffemorphism.* If $R^n \supset S$ and $\nabla\Phi$ is nonsingular at a point $\mathbf{x}=\mathbf{x}_0$ then it's called a *local diffeomorphism.*. By using a diffeomorphism, a nonlinear system can be transformed into another nonlinear system with a new set of states. This is quite similar to what is done in linear systems.

2.2. Feedback Linearization

The basic idea of feedback linearization is to transform a nonlinear system into a linear one in order to utilize the tools developed for linear control system design. A broad class of nonlinear systems can be transformed using methods based on differential geometry.

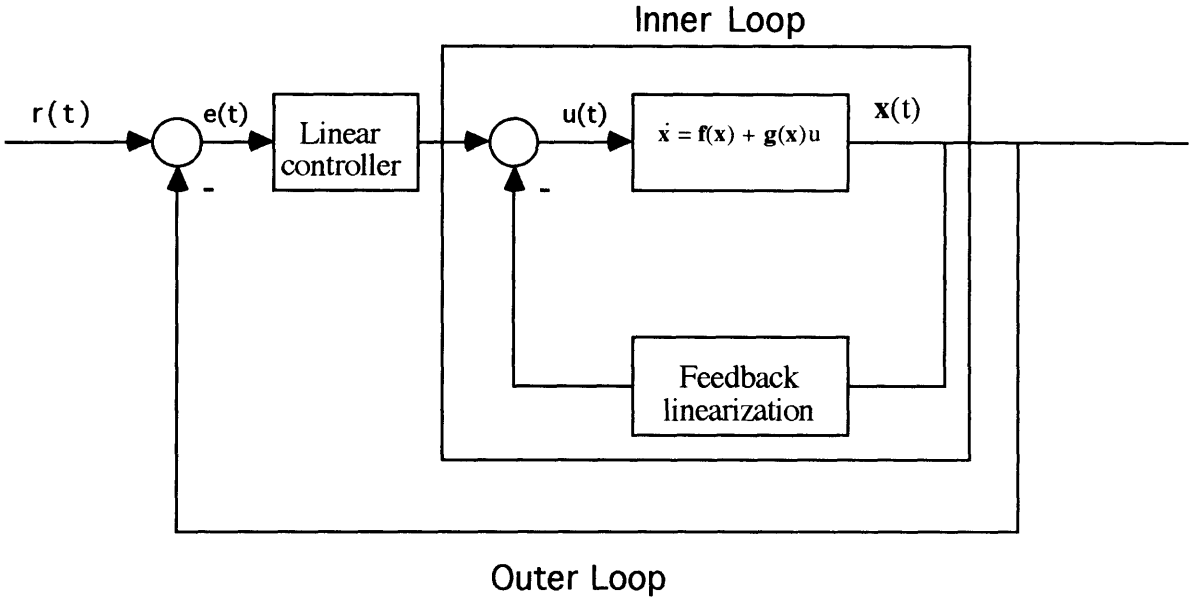

Figure 2.2.1: Feedback linearized nonlinear system.

The idea is to construct a feedback loop such that the system from v to x (the inner loop) in figure 2.2.1 behaves as a linear system. Another more classical linearizing method is, for instance, the Taylor series approximation. Linear control theory can then be utilized to design an outer loop (figure 2.2.1). Feedback linearization is also known as Linearization by Feedback or Nonlinear Inversion. Two different approaches are considered: *Input-State Linearization* and *Input Output Linearization*..

2.2.1. SISO Input-State Linearization

Assume that a nonlinear system

$$\dot{\mathbf{x}}=\mathbf{f}(\mathbf{x},u) \tag{2.2.1}$$

can be written as

$$\dot{\mathbf{x}}=\mathbf{f}(\mathbf{x})+\mathbf{g}(\mathbf{x})u \tag{2.2.2}$$

i.e. the system is linear or affine in u. Furthermore, it is assumed that **f** and **g** are smooth vector fields. The system above is said to be feedback linearizable if, in a region S, there exists a transformation (diffeomorphism)

$$T(\mathbf{x}): S \rightarrow R^n \tag{2.2.3}$$

and a nonlinear feedback control law

$$u(\mathbf{x}) = \alpha(\mathbf{x}) + \beta(\mathbf{x})\, u \tag{2.2.4}$$

such that the new state variables

$$\mathbf{z} = T(\mathbf{x}) \tag{2.2.5}$$

satisfy the equations

$$\dot{\mathbf{z}} = A\mathbf{z} + \mathbf{b}v \tag{2.2.6}$$

where

$$A = \begin{bmatrix} 0_{(n-1)x(n-1)} & I_{(n-1)x(n-1)} \\ 0 & 0 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 0_{(n-1)x1} \\ 1 \end{bmatrix} \tag{2.2.7}$$

It can be shown [1], [2] that in order for this to be possible the following must hold:

1) the vectorfields $[\mathbf{g}, ad_f\mathbf{g}, \ldots, ad_f^{n-1}\mathbf{g}]$ are linearly independent in S

2) the set $[\mathbf{g}, ad_f\mathbf{g}, \ldots, ad_f^{n-2}\mathbf{g}]$ is involutive in S

Condition one above is a generalized controllability condition for the nonlinear system (2.2.1). Based on the above input-state linearization of a nonlinear system can be performed in the following way:

i) form the vector fields $\mathbf{g}, ad_f\mathbf{g}, \ldots, ad_f\mathbf{g}^{(n-1)}$.

ii) check linear independence and involutivity in S.

iii) find $T_1(\mathbf{x})$ by solving

$$L_{\mathbf{g}}T_1 = L_{ad_{\mathbf{f}}\,\mathbf{g}}T_1 = \ldots = L_{ad_{\mathbf{f}}^{n-2}\mathbf{g}}T_1 = 0 \qquad (2.2.7)$$

or ,equivalently,

$$\begin{aligned} \nabla T_1\, ad_{\mathbf{f}}^{i}\mathbf{g} &= 0 \quad i=0, \ldots, n-2 \\ \nabla T_1\, ad_{\mathbf{f}}^{n-1}\mathbf{g} &\neq 0 \end{aligned} \qquad (2.2.8)$$

iv) compute

$$T_i = L_{\mathbf{f}}^{i-1}T_1 \qquad i=2, \ldots, n \qquad (2.2.9)$$

v) compute

$$\alpha(\mathbf{x}) = -\frac{L_{\mathbf{f}}^{n}T_1}{L_{\mathbf{g}}L_{\mathbf{f}}^{n-1}T_1} \qquad (2.2.8)$$

$$\beta(\mathbf{x}) = \frac{1}{L_{\mathbf{g}}L_{\mathbf{f}}^{n-1}T_1} \qquad (2.2.9)$$

2.2.2. SISO Input-Output Linearization

Input-output linearization considers systems described by the state space representation

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u \\ y &= h(\mathbf{x}) \end{aligned} \qquad (2.2.14)$$

where y is the output. We want to generate a linear differential relation between the output y and a new input v. To obtain this linear relationship the output function y is differentiated r times until the input function u explicitly appears

$$y^{(r)} = L_{\mathbf{f}}^{r}\, h(\mathbf{x}) + L_{\mathbf{g}}L_{\mathbf{f}}^{r-1}\, h(\mathbf{x}) \qquad (2.2.15)$$

Then, in a region S the following holds:

$$\begin{aligned} L_{\mathbf{g}}L_{\mathbf{f}}^{i}\, h(\mathbf{x}) &= 0 \quad i = 0, 1, \ldots, r-2 \\ L_{\mathbf{g}}L_{\mathbf{f}}^{r-1}h(\mathbf{x}) &\neq 0 \end{aligned} \qquad (2.2.16)$$

By choosing a control law given by

$$u = \frac{1}{L_g L_f^{r-1} h(\mathbf{x})} (-L_f^r h(\mathbf{x}) + v) \tag{2.2.17}$$

the original nonlinear system can be written as

$$y^{(r)} = v \tag{2.2.18}$$

The number of necessary differentiations r is called the *relative degree* of the system. If r < n, the nonlinear system (2.2.14) can be transformed into a so-called *normal form.* However in our examples r = n and the input output linearization will lead to input-state linearization. Working on the transformed system, tools from linear control systems can be utilized.

2.3. The LQG-LTR Method

The so-called *Linear-Quadratic-Gaussian method with Loop-Transfer-Recovery* (LQG/LTR) is a systematic procedure for designing feedback control systems for both single input single output (SISO) and multi-input multi-output (MIMO) systems [10]. In the following it is used to design SISO controllers only.

Suppose our plant is described by

$$\begin{aligned} \dot{\mathbf{x}}_p &= A_p \mathbf{x}_p + b_p u_p \\ y &= c_p \mathbf{x}_p \end{aligned} \tag{2.3.1}$$

where $\mathbf{x}_p \in \mathbf{R}^n$, $u_p \in \mathbf{R}$, $y \in \mathbf{R}$, $A_p \in \mathbf{R}^{nxn}$, $b_p \in \mathbf{R}^{nx1}$, $c_p \in \mathbf{R}^{nx1}$. First what is called the *design plant model* (DPM) is defined. This model includes the nominal model of the dynamics of the physical process possibly with a scaling of the variables. Augmentation dynamics such as integrators are also included here. In our case one integrator in the control channel is added defined by

$$\dot{u}_p = u_{dpm} \tag{2.3.2}$$

The DPM is then defined by the augmented dynamics

$$\begin{aligned} \dot{\mathbf{x}}_{dpm} &= A_{dpm}\mathbf{x}_{dpm} + b_{dpm}u_{dpm} \\ y &= c_{dpm}\mathbf{x}_{dpm} \end{aligned} \tag{2.3.3}$$

where

$$A_{dpm} = \begin{bmatrix} 0 & 0 \\ \mathbf{b}_p & A_p \end{bmatrix}, \mathbf{b}_{dpm} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \mathbf{c}_{dpm} = \begin{bmatrix} 0 & \mathbf{c}_p \end{bmatrix} \tag{2.3.4}$$

The transfer function (TF) g(s) of the DPM is given by

$$g(s) = \mathbf{c}_{dpm}\Phi_{dpm}(s)\mathbf{b}_{dpm} \tag{2.3.5}$$

where

$$\Phi_{dpm}(s) = (sI - A_{dpm})^{-1} \tag{2.3.5}$$

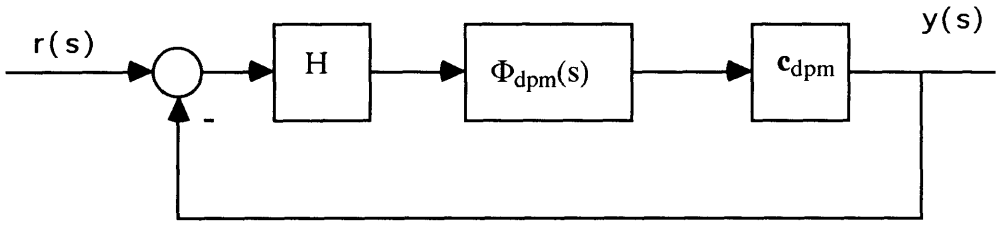

Figure 2.3.1: The structure of the target feedback loop.

Based on (2.3.5) we define the so-called *target feedback loop* (TFL) (figure 2.3.1) which satisfies all our specifications related to nominal stability, stability-robustness to modeling errors, and performance. Later we will try to recover the main properties of this loop with our actual control system.

The design of the TFL boils down to selecting H. An easy and straightforward method to do so is based on the solution of a fictitious continuous time Kalman Filter. Consider the stochastic dynamics

$$\dot{\mathbf{x}}_{dpm} = A_{dpm}\mathbf{x}_{dpm} + \mathbf{l}\xi$$
$$y = \mathbf{c}_{dpm}\mathbf{x}_{dpm} + \theta \tag{2.3.7}$$

where ξ, θ is white zero mean noise with intensity I and μI respectively. The matrix H is then given by

$$H = (\frac{1}{\mu})\Sigma\mathbf{c}_{dpm}^T \tag{2.3.8}$$

where Σ is the solution of the algebraic Ricatti equation

$$0 = A_{dpm}\Sigma + \Sigma A_{dpm}^T + \mathbf{l}\mathbf{l}^T - (\frac{1}{\mu})\Sigma\mathbf{c}_{dpm}^T\mathbf{c}_{dpm}\Sigma \tag{2.3.9}$$

L and μ are now our design parameters for finding a 'good' H. If we choose the matrix L such that the Bode plot of the TFL is identical at high and low frequencies it can be shown [10] that the decomposed L

$$\mathbf{l} = \begin{bmatrix} \mathbf{l}_L \\ \mathbf{l}_H \end{bmatrix} \tag{2.3.10}$$

is given by

$$\mathbf{l}_L = -(\mathbf{c}_p A_p^{-1}\mathbf{b}_p)^{-1} \tag{2.3.11}$$

$$\mathbf{l}_H = \mathbf{c}_p^T(\mathbf{c}_p\mathbf{c}_p^T)^{-1} \tag{2.3.12}$$

The crossover frequency can now be adjusted by a proper selection of μ.The only remaining design parameter in K(s) is the control gain vector **g**. This matrix is computed via the solution of the cheap-control Linear-Quadratic Regulator problem

$$0 = -K_\rho A_{dpm} - A_{dpm}^T K_\rho - \mathbf{c}_{dpm}^T\mathbf{c}_{dpm} + (\frac{1}{\rho})K_\rho\mathbf{b}_{dpm}\mathbf{b}_{dpm}^T K_\rho \tag{2.3.13}$$

and then compute **g** from

$$\mathbf{g} = (\frac{1}{\rho})\mathbf{b}_{dpm}^T K_\rho \tag{2.3.14}$$

It can be shown that as ρ goes to zero our actual plant with the MBC controller approaches

the TFL.[11].

The LQG/LTR compensator K(s) belongs to the so-called model based compensators (MBC). An block diagram of this compensator can be seen in figure 2.3.2.

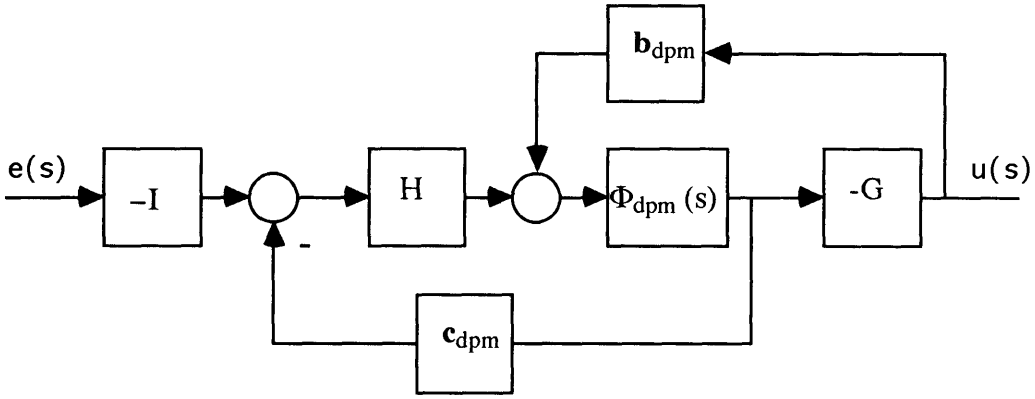

Figure 2.3.2: The structure of the complete compensator.

The transfer function of the controller is defined by

$$u(s) = K(s) \tag{3.2.15}$$

where

$$K(s) = G(sI - A_{dpm} + \mathbf{b}_{dpm}G + H\mathbf{c}_{dpm})^{-1} H \tag{3.2.16}$$

3. Sample Problems

3.1. Duffing's Equation / Mass on a Nonlinear Spring

The mechanical system consisting of a nonlinear spring, a mass, and a time varying force applied to the mass (figure 3.1.1) can be modelled by the equation

$$\ddot{w} + \omega^2 w + hw^3 = gu \tag{3.1.1}$$

where

$$\omega^2 = \frac{k_1}{m},\ h = \frac{k_1 a}{m},\ g = \frac{1}{m} \tag{3.1.2}$$

The parameter h is positive for a hard spring and negative for a soft spring.

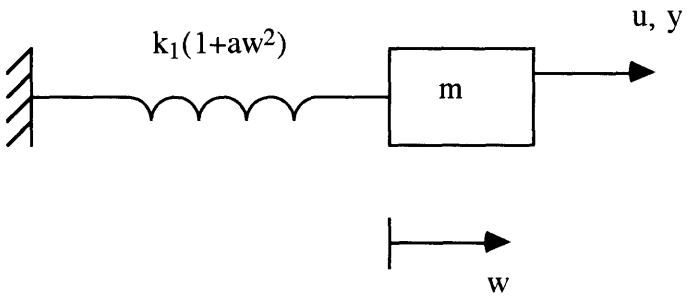

Figure 3.1.1: Mass on nonlinear spring.

Equation 3.1.1 is a classic equation of the theory of nonlinear systems and is known as Duffing's equation [9]. Besides describing the mechanical system of a mass on a nonlinear spring, it also describes an electrical system consisting of the series combination of a capacitor and inductor, one of which is nonlinear.

The equation in state space form with $x = [\, w,\ dw/dt \,]^T$ becomes

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{k_1}{m}x_1 - \frac{k_1 a}{m}x_1^3 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})u \tag{3.1.3}$$

Linearization of this equation based on Taylor series is straightforward and can be written as

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k_1}{m} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u = A\mathbf{x} + Bu \qquad (3.1.4)$$

The measurement model is given by

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad (3.1.5)$$

i.e. position measurement only.

To simulate the effect of unstructured modeling error we can add some additional low damped high order dynamics to the actual system by adding a second mass in series with the first one as shown in figure 3.1.2

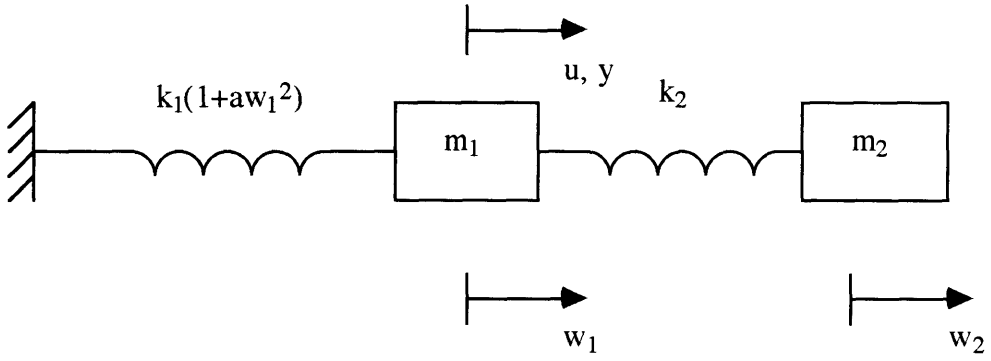

Figure 3.1.2: Mass on nonlinear spring with high order dynamics.

The equations of this system in state space form with $x = [\, w_1, dw_1/dt, w_2, dw_2/dt \,]^T$ are

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{k_1}{m_1}(1+ax_1^2)x_1 - \frac{k_2}{m_1}(x_1-x_3) \\ x_4 \\ -\frac{k_2}{m_2}(x_3-x_1) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m_1} \\ 0 \\ 0 \end{bmatrix} u \qquad (3.1.6)$$

The Taylor linearization of the augmented system is given by

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_1+k_2}{m_1} & 0 & \frac{k_2}{m_1} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k_2}{m_2} & 0 & -\frac{k_2}{m_2} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m_1} \\ 0 \\ 0 \end{bmatrix} u \qquad (3.1.7)$$

We assume collocated control i.e. the measurement is done at the same position as the control is acting

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad (3.1.8)$$

The transfer functions for the Taylor linearized system with and without high order dynamics are shown in figure 3.1.3. The parameter values used to obtain the plot are

$$k_1 = 1,\ k_2 = 10,\ m = 1,\ m_1 = m_2 = 0.5,\ a = 0.2 \qquad (3.1.9)$$

The system without high order dynamics is used as our nominal model for designing two controllers. The first one based on a feedback linearization of (3.1.3) and the second based on (3.1.4). In both cases the nominal measurement model is (3.1.5). A LQG-LTR approach is used in the design of the controllers for the linearized systems. The nominal parameter values used in the controllers are

$$\hat{k}_1 = 1, \hat{m} = 1, \hat{a} = 0.2 \tag{3.1.10}$$

When the effect of umodeled high order dynamics is studied, (3.1.6) is used to simulate the actual plant.

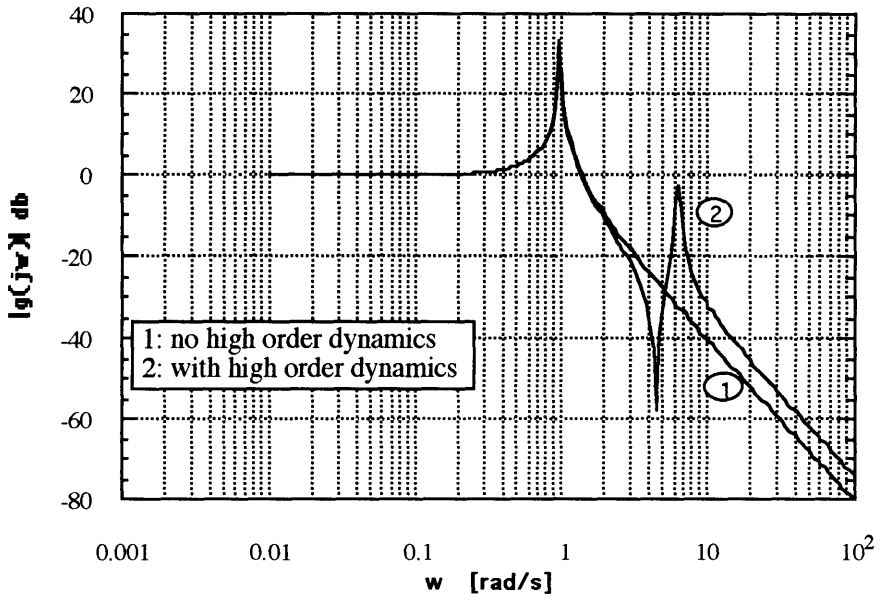

Figure 3.1.3: Open loop transfer functions with and without high order dynamics.

3.1.1. The Controller Based on a Feedback Linearized System

To generate the relationship between the output y and the input u the output is differentiated until u appears explicitly.

$$\begin{aligned} \dot{y} &= \dot{x}_1 = x_2 \\ \ddot{y} &= \dot{x}_2 = -\left(\frac{k_1}{m} + \frac{k_1 a}{m} x_1^2\right)x_1 + \frac{1}{m}u \end{aligned} \tag{3.1.11}$$

Thus the relative degree of the system r = n = 2 and we have no internal dynamics. We also know that in this case the input-output approach is equivalent to the input-state approach.

The control input is chosen to be of the form

$$u = (\widehat{k_1} + \widehat{k_1}\hat{a}x_1^2)x_1 + v \tag{3.1.12}$$

where v is the new input to be determined. The relationship between the output and the new input is now given by

$$\ddot{y} = v \tag{3.1.13}$$

This is our linearized system and will be the basis for the design of the controller. As mentioned above, we use the LQG-LTR approach to do this. The linear, time invariant plant is now given by

$$\begin{aligned} \dot{\mathbf{x}}_p = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v = A_p\mathbf{x}_p + \mathbf{b}_p v \\ y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \mathbf{c}_p\mathbf{x}_p \end{aligned} \tag{3.1.14}$$

We notice that the system matrix A_p is not invertible.

An integrator in the control channel is needed at a later stage but, since we already have a double integrator, it is not necessary to add one in this case; we can use one of the built in integrators. This means that we have to redefine our plant model. The new plant model is now given by the matrices

$$A_p = [0],\ B_p = [1],\ C_p = [1],\ D_p = [0] \tag{3.1.15}$$

The system matrix is still singular. The design plant model (dpm) is now given by [10]

$$A_{dpm} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix},\ B_{dpm} = \begin{bmatrix} 1 \\ 0 \end{bmatrix},\ C_{dpm} = \begin{bmatrix} 0 & 1 \end{bmatrix},\ D_{dpm} = [0] \tag{3.1.16}$$

We want the controller to track reference inputs with frequencies up to 1 rad/s. By design a set of parameters for the LQG-LTR controller is found to be

$$L_L = 100,\ L_H = 1,\ \mu = 1,\ \rho = 1x10e^{-6} \tag{3.1.17}$$

and the resulting control gains are

$$H = \begin{bmatrix} 100 \\ 14.18 \end{bmatrix}$$
$$\mathbf{g} = [4.47\ \ 1000] \tag{3.1.18}$$

The open loop-, sensitivity-, and complementary sensitivity (or closed loop-), transfer functions for the design are shown in figure 3.1.4.

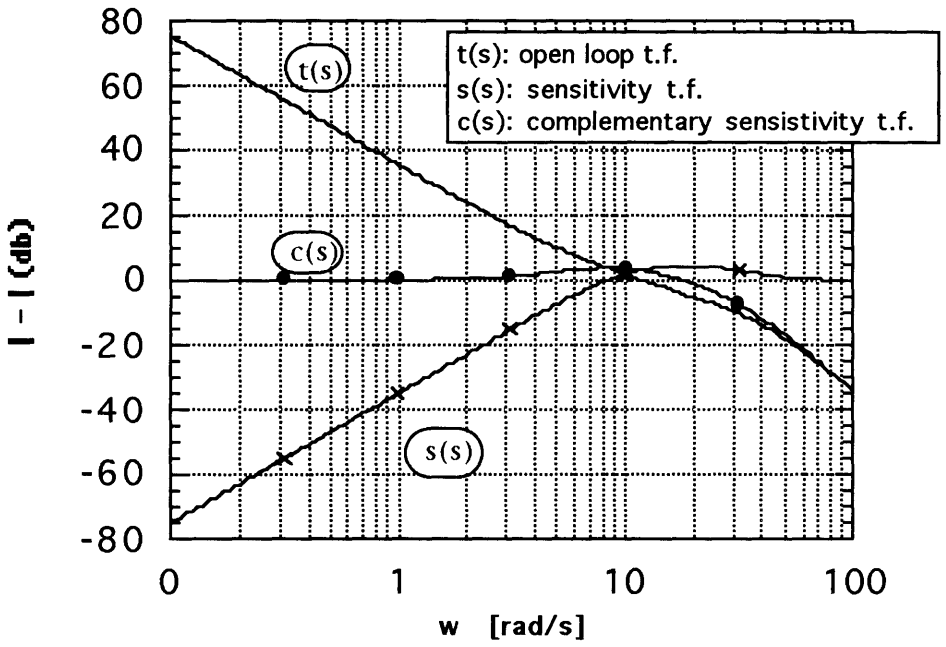

Figure 3.1.4: Transfer functions Feedback linearization/LQG-LTR design.

3.1.2. The LQG/LTR Controller Based on Taylor Linearization

We now use (3.1.4) as our starting point. The plant model is then given by the matrices

$$A_p = \begin{bmatrix} 0 & 1 \\ -\frac{k_1}{m} & 0 \end{bmatrix}, \mathbf{b}_p = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \mathbf{c}_p = [\,1 \quad 0\,], D_p = [0] \tag{3.1.19}$$

Augment the system with an integrator in the control channel and arrive at the design plant model (dpm)

$$A_{dpm} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & -\frac{k_1}{m} & 0 \end{bmatrix}, \mathbf{b}_{dpm} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \mathbf{c}_{dpm} = [\,0 \quad 1 \quad 0\,], D_{dpm} = [0] \tag{3.1.20}$$

We want the controller to track reference inputs with frequencies up to 1 rad/s. With the nominal parameter values from (3.1.9) and by trial and error a set of parameters for the LQG-LTR controller is found to be

$$\mu = 1\text{x}10e^{-4}, \rho = 1\text{x}10e^{-9} \tag{3.1.21}$$

and the resulting gains are

$$H = \begin{bmatrix} 100 \\ 100 \\ 0 \end{bmatrix}$$
$$\mathbf{g} = [\quad 63 \quad 31.560 \quad 2000\ ] \tag{3.1.22}$$

The open loop-, sensitivity-, and complementary sensitivity (or closed loop-), transfer functions are shown in figure 3.1.5.

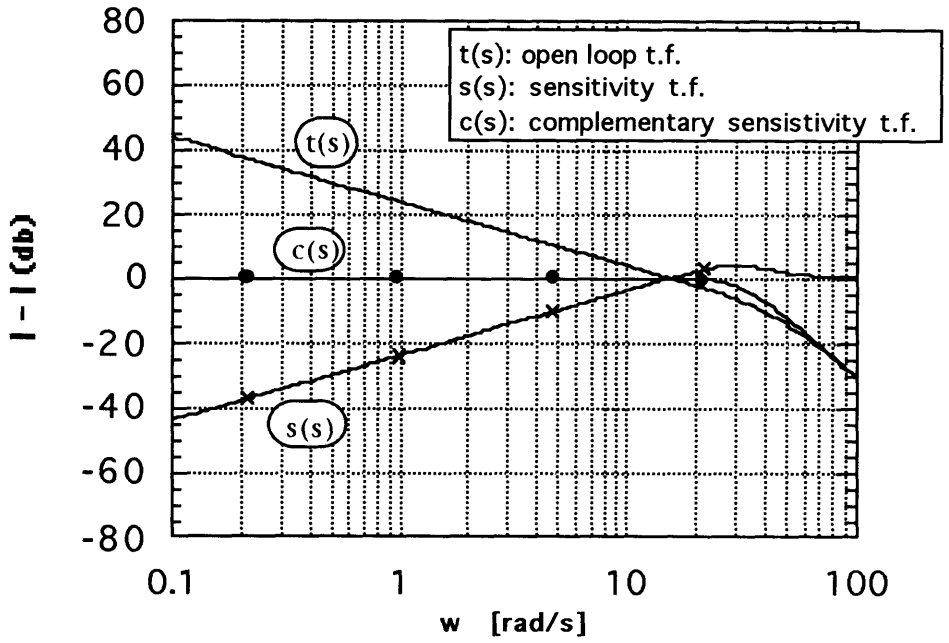

Figure 3.1.5: Transfer functions Taylor linearization/LQG-LTR design.

3.2. The Mathieu Equation

A simple form of the so-called Hill's equation is known as the Mathieu equation. The standard form of the Mathieu equation driven by a time varying input u can be written [9]

$$\frac{d^2w}{dt^2} + (a-2q\cos(2t))\, w = u \tag{3.2.1}$$

where w is the dependent variable, t is the independent variable, and a and q are constants. The equation in state space form with $x = [\, w,\ dw/dt \,]^T$ can be written

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ (2q\cos(2t) - a)x_1 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u = \mathbf{f}(\mathbf{x}) + g(\mathbf{x})u \tag{3.2.2}$$

A Taylor based linearization of this equation can be written

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 2q - a & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u = A\mathbf{x} + Bu \tag{3.2.3}$$

The measurement model is given by

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{3.2.4}$$

We can add some additional low damped high order dynamics to the system in the same way as before. The equations of this augmented system in state space form are

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ (2q\cos(2t) - a)x_1 - k_1(x_1 - x_3) \\ x_4 \\ -k_1(x_3 - x_1) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} u \tag{3.2.5}$$

The Taylor linearized augmented system becomes

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 2q\text{-}a\text{-}k_2 & 0 & k_1 & 0 \\ 0 & 0 & 0 & 1 \\ k_1 & 0 & -k_1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} u \qquad (3.2.6)$$

The same assumption as before of collocated control gives the measurement equation

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad (3.2.7)$$

The parameter values used are

$$k_1 = 10,\ m = 1,\ m_1 = m_2 = 0.5,\ q = 0.5, a = 2 \qquad (3.2.8)$$

With this choice of parameter values the transfer functions for (3.2.3) and (3.2.6) are identical to the transfer functions obtained in figure 3.1.3.

As above the we can now design two controllers, one based on a feedback linearization of (3.2.2), and the other on (3.2.3) directly. The nominal parameter values used in the design of the controllers are

$$\hat{m} = 1,\ \hat{q} = 0.5, \hat{a} = 2 \qquad (3.2.9)$$

3.2.1. The Feedback Linearized Controller

We follow the same steps as above to generate the relationship between the output y and the input u.

$$\begin{aligned} \dot{y} &= \dot{x}_1 = x_2 \\ \ddot{y} &= \dot{x}_2 = (2q\cos(2t) - a)x_1 + u \end{aligned} \qquad (3.2.10)$$

The relative degree of this system is r = n = 2, and we have no internal dynamics.

Furthermore the input-output approach equals the input-state approach. The control input is chosen to be of the form

$$u = - (2\hat{q}\cos(2t) - \hat{a})x_1 + v \qquad (3.2.11)$$

where v is the new input to be determined. The relationship between the output and the new input is now given by

$$\ddot{y} = v \qquad (3.2.12)$$

This is naturally exactly the same equation that we arrived at above so we can use the same LQG/LTR controller with the gain values in (3.1.18). The open loop-, sensitivity-, and complementary sensitivity (or closed loop-), transfer functions are shown in figure 3.1.4.

3.2.2. The LQG/LTR Controller Based on Taylor Linearization

The plant model for design of the LQG/LTR controller is

$$A_p = \begin{bmatrix} 0 & 1 \\ 2\hat{q} - \hat{a} & 0 \end{bmatrix}, \mathbf{b}_p = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \mathbf{c}_p = [\,1 \;\; 0\,], D_p = [0] \qquad (3.2.13)$$

With the choice of nominal parameters in (3.2.9) this linearized model is exactly the same as the control design model for the LQG/LTR controller in 3.1.1 before. Therefore the same gain values (3.1.22) can be used here.

3.3. Mass w/Discontinuous Stiffness

The mechanical oscillating system considered next combines a constant mass with a spring for which the stiffness varies in a discontinuous manner. If the spring becomes stiffer symmetrically, it may be described by the curve of figure 3.3.1.

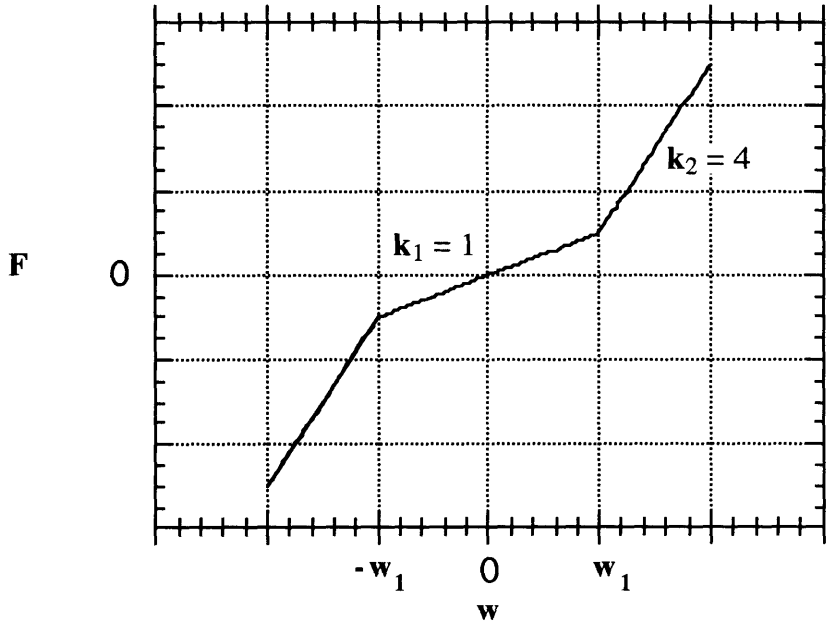

Figure 3.3.1: Spring force as a function of displacement.

The relations applying to such a spring are

$$\begin{aligned} w \le -w_1: &\quad F = k_2(w+w_2) \\ -w_1 \le w \le w_1: &\quad F = k_1 w \\ w \ge w_1: &\quad F = k_2(w-w_2) \end{aligned} \tag{3.3.1}$$

For each of the piecewise linear force equations there is a corresponding system equation.

The three equations of motion are

$$\begin{aligned} w \le -w_1: &\qquad \ddot{w} + \frac{k_2}{m}(w + w_2) = \frac{1}{m}u \\ -w_1 \le w \le w_1: &\qquad \ddot{w} + \frac{k_1}{m}w = \frac{1}{m}u \\ w \ge w_1: &\qquad \ddot{w} + \frac{k_2}{m}(w - w_2) = \frac{1}{m}u \end{aligned} \tag{3.3.2}$$

where w1 and w2 are constants. The equations in state space form with $x = [\, w, dw/dt \,]^T$ can be written

$$\begin{aligned} x \le -w_1: &\qquad \dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{k_2}{m}(x_1 + w_2) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u \\ -w_1 \le x \le w_1: &\qquad \dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{k_1}{m}x_1 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u \\ x \ge w_1: &\qquad \dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{k_2}{m}(x_1 - w_2) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u \end{aligned} \tag{3.3.3}$$

A Taylor linearization of these equations can be written

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k_1}{m} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u = A\mathbf{x} + Bu \tag{3.3.4}$$

The measurement model is given by

$$y = [\, 1 \;\; 0 \,] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{3.3.5}$$

We can add some additional high order dynamics to the system in the same way as before. The Taylor linearized equations of this augmented system in state space form are

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_1+k_3}{m_1} & 0 & \frac{k_3}{m_1} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k_3}{m_2} & 0 & -\frac{k_3}{m_2} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m_1} \\ 0 \\ 0 \end{bmatrix} u \qquad (3.3.6)$$

The assumption of collocated control gives

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad (3.3.7)$$

As before two controllers are designed, based on a feedback linearization of (3.3.3), and on (3.3.4) respectively. The nominal parameter values used for the controllers are

$$\hat{k}_1 = 1, \hat{k}_2 = 4, \hat{m} = 1, \qquad (3.3.8)$$

3.3.1. The Feedback Linearized Controller

We follow the same steps as before to generate the relationship between the output y and the input u

$$\begin{aligned} \dot{y} &= \dot{x}_1 = x_2 \\ \ddot{y} &= \dot{x}_2 = F(x_1) + \frac{1}{m} u \end{aligned} \qquad (3.3.9)$$

where F is as given above. The relative degree of this system is r = n = 2 and we have no internal dynamics. Furthermore, the input-output approach is equivalent to the input-state approach.

Choose the control input to be of the form

$$u = \hat{m}(-\hat{F}(x_1) + v) \tag{3.3.10}$$

where v is the new input to be determined. The relationship between the output and the new input is now given by

$$\ddot{y} = v \tag{3.3.11}$$

This is exactly the same equation that we arrived at above so we can use the same LQG/LTR controller as in 3.1.1. The open loop-, sensitivity-, and complementary sensitivity (or closed loop-), transfer functions are shown above in figure 3.1.4.

3.3.2. The LQG/LTR Controller Based on Taylor Linearization.

The plant model for design of the LQG/LTR controller is

$$A_p = \begin{bmatrix} 0 & 1 \\ -\frac{\hat{k}_1}{\hat{m}} & 0 \end{bmatrix}, \mathbf{b}_p = \begin{bmatrix} 0 \\ \frac{1}{\hat{m}} \end{bmatrix}, \mathbf{c}_p = [\,1 \quad 0\,], D_p = [0] \tag{3.3.12}$$

With the nominal parameter values from (3.3.8) this model is exactly the same as the one in (3.3.2) and the same controller with the same gains can be used.

3.4. Mass w/Dry Friction

To a simple approximation the magnitude of the force required to cause one dry metal surface to slide along another is independent of the relative velocity of the two surfaces. The magnitude of the force depends upon the nature of the surfaces and the pressure one exerts upon the other. This is often called Coulomb or "dry" friction. Consider a mechanical oscillator with a constant mass and a linear spring. The equation of motion for this system is

$$m\,\ddot{w} + h\,\mathrm{sign}(\dot{w}) + k\,w = u \tag{3.4.1}$$

where h and k are constants. The equation in state space form with $x = [\, w,\ dw/dt \,]^T$ can be written as

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{h}{m}\mathrm{sign}(\dot{x}_1) - \frac{k}{m}x_1 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u = \mathbf{f}(\mathbf{x}) + g(\mathbf{x})u \tag{3.4.2}$$

A Taylor based linearization of this equation, ignores the friction and is written

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u = A\mathbf{x} + Bu \tag{3.4.3}$$

The measurement model is given by

$$y = [\, 1 \;\; 0 \,] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \tag{3.4.4}$$

The high order dynamics is added to the system in the same way as above. The equations of this system in state space form are

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{k_1}{m_1}x_1 - \frac{h}{m_1}\text{sign}(\dot{x}_1) - \frac{k_2}{m_1}(x_1 - x_3) \\ x_4 \\ -\frac{k_2}{m_2}(x_3 - x_1) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m_1} \\ 0 \\ 0 \end{bmatrix} u \qquad (3.4.5)$$

The augmented Taylor linearized system becomes

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_1+k_2}{m_1} & 0 & \frac{k_2}{m_1} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{k_2}{m_2} & 0 & -\frac{k_2}{m_2} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m_1} \\ 0 \\ 0 \end{bmatrix} u \qquad (3.4.6)$$

The assumption of collocated control gives

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad (3.4.7)$$

Again two controllers are designed, based on a feedback linearization of (3.4.2), and on (3.4.3) respectively.The nominal parameter values used in the controllers are

$$\hat{k}_1 = 1, \hat{h} = 0.2, \hat{m} = 1 \qquad (3.4.8)$$

3.4.1. The Feedback Linearized Controller

The relationship between the output y and the input u:

$$\begin{aligned} \dot{y} &= \dot{x}_1 = x_2 \\ \ddot{y} &= \dot{x}_2 = -\frac{h}{m}\,\mathrm{sign}(\dot{x}_1) - \frac{k_1}{m}x_1 + \frac{1}{m}u \end{aligned} \tag{3.4.9}$$

The relative degree of this system is also r = n = 2 and the control input is chosen to be of the form

$$u = \hat{h}\,\mathrm{sign}(\dot{x}_1) + \hat{k}_1\,x_1 + v \tag{3.4.10}$$

where v is the new input to be determined. The relationship between the output and the new input is now given by

$$\ddot{y} = v \tag{3.4.11}$$

and the same LQG/LTR controller as designed in 3.1.1 is used. The open loop-, sensitivity-, and complementary sensitivity (or closed loop-), transfer functions are shown above in figure 3.1.4.

3.4.2. The LQG/LTR Controller Based on a Taylor Linearization.

The plant model for design of the LQG/LTR controller is

$$A_p = \begin{bmatrix} 0 & 1 \\ -\frac{\hat{k}_1}{\hat{m}} & 0 \end{bmatrix},\ \mathbf{b}_p = \begin{bmatrix} 0 \\ 1 \end{bmatrix},\ \mathbf{c}_p = [\,1 \quad 0\,],\ D_p = [0] \tag{3.4.12}$$

With the choice of parameters indicated in (3.4.9) this model is exactly the same as the one we arrived at for the mass on nonlinear spring and again the same LQG/LTR controller that is designed in 3.1.2 with exactly the same gains can be used.

3.5. Duffing's Equation w/Input Saturation

A typical nonlinear effect encountered in many practical control systems is the saturation of the actuators. This is due to physical limitations of, for instance, a control valve. The impact of such effects on the overall control system is complicated, but it is known that it may lead to such phenomena as limit cycles and even instability. It is, therefore, interesting to study how a saturation in the control channel will affect the performance of the two controllers compared in this study. In order to do this a saturation element is included in the simulation diagram for the Duffing's equation. The saturation element characteristics are given in (3.5.1) and shown in figure 3.5.1.

$$\begin{aligned} x \le -x_0 \qquad & u = -u_0 = -x_0 \\ -x_0 \le x \le x_0 \qquad & u = x \\ x \ge x_0 \qquad & u = u_0 = x_0 \end{aligned} \tag{3.5.1}$$

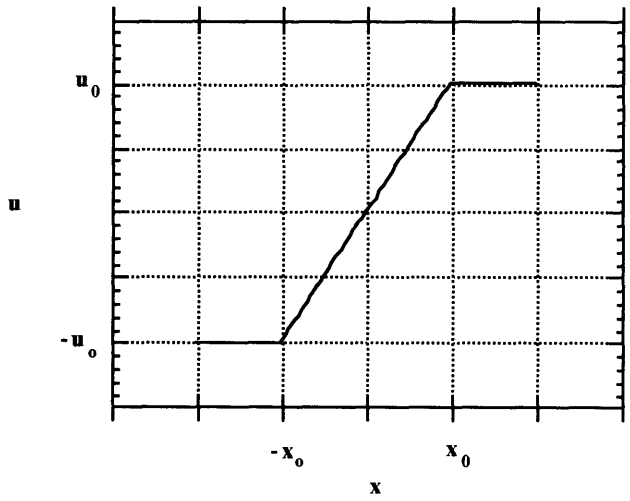

Figure 3.5.1: Saturation element characteristics.

This is considered part of the unmodeled dynamics so both controllers remains the same as in section 3.1.1 and 3.1.2.

3.6. Duffing's Equation w/Input Deadband

A similar effect as saturation is deadband in the control actuator. This is a result of long time use, or bad design and often occurs in practical control problems. To study this effect a deadband is included in the control channel for Duffing's equation in the same way as the saturation element was included. The characteristics of the deadband is given in 3.6.1 and shown in figure 3.6.1

$$\begin{aligned} x \le -x_0 \quad & u = x + x_0 \\ -x_0 \le x \le x_0 \quad & u = 0 \\ x \ge x_0 \quad & u = x - x_0 \end{aligned} \tag{3.6.1}$$

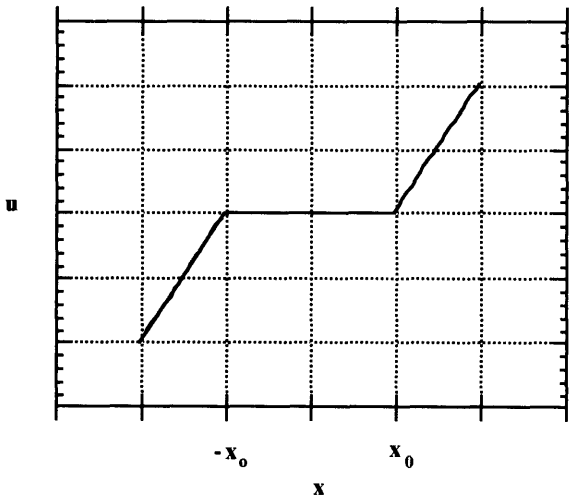

Figure 3.6.1: Deadband element characteristics.

This is also considered part of the unmodeled dynamics so the controllers are the same as in section 3.1.

3.7. One Link Robot

Finally a one link robot in a gravity field (figure 3.7.1) is studied.

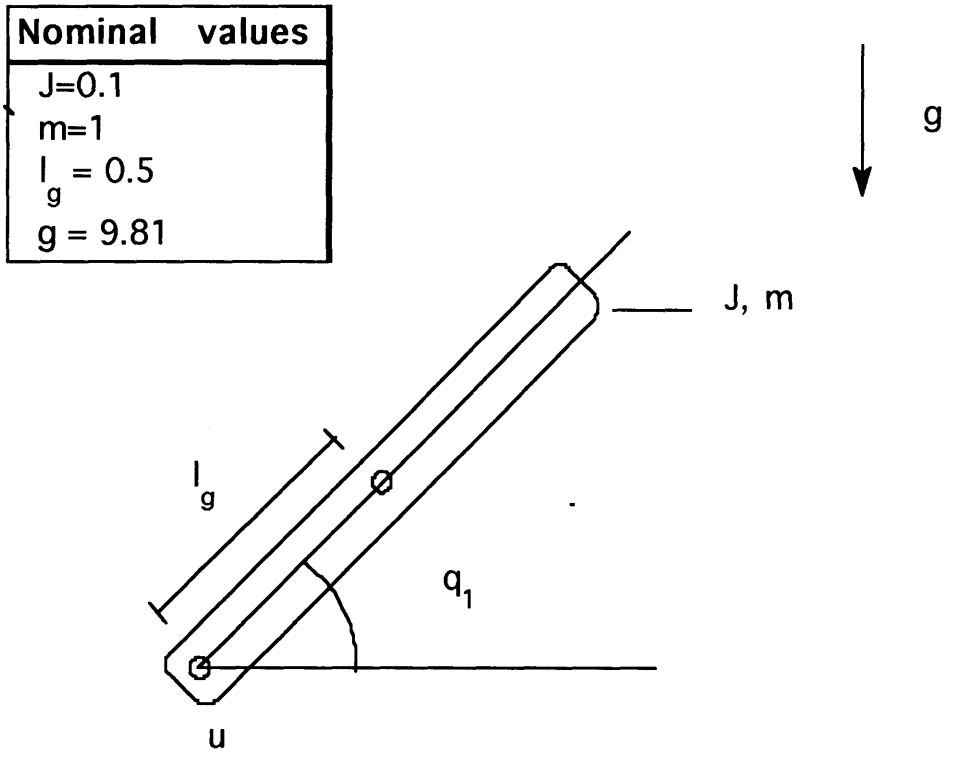

Figure 3.7.1: Rigid robotarm.

From figure 3.7.1 using Newton's law of momentum we find the dynamic equation.

$$J\ddot{q}_1 + mgl_g \sin q_1 = u \tag{3.7.1}$$

Writing the equation in state space form by setting

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} q_1 \\ \dot{q}_1 \end{bmatrix} \tag{3.7.2}$$

we get

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\frac{mgl_g}{J}\sin(x_1) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{J} \end{bmatrix} u = f(\mathbf{x}) + g(\mathbf{x})\, u \qquad (3.7.3)$$

The Taylor linearized system is given by

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{mgl_g}{J} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{J} \end{bmatrix} u = A\mathbf{x} + bu \qquad (3.7.4)$$

The measurement is assumed to be

$$\mathbf{y} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = C\mathbf{x} \qquad (3.7.5)$$

i.e. both position and velocity is available for measurement.

Low damped unmodeled dynamics is added in as indicated in figure 3.7.2.

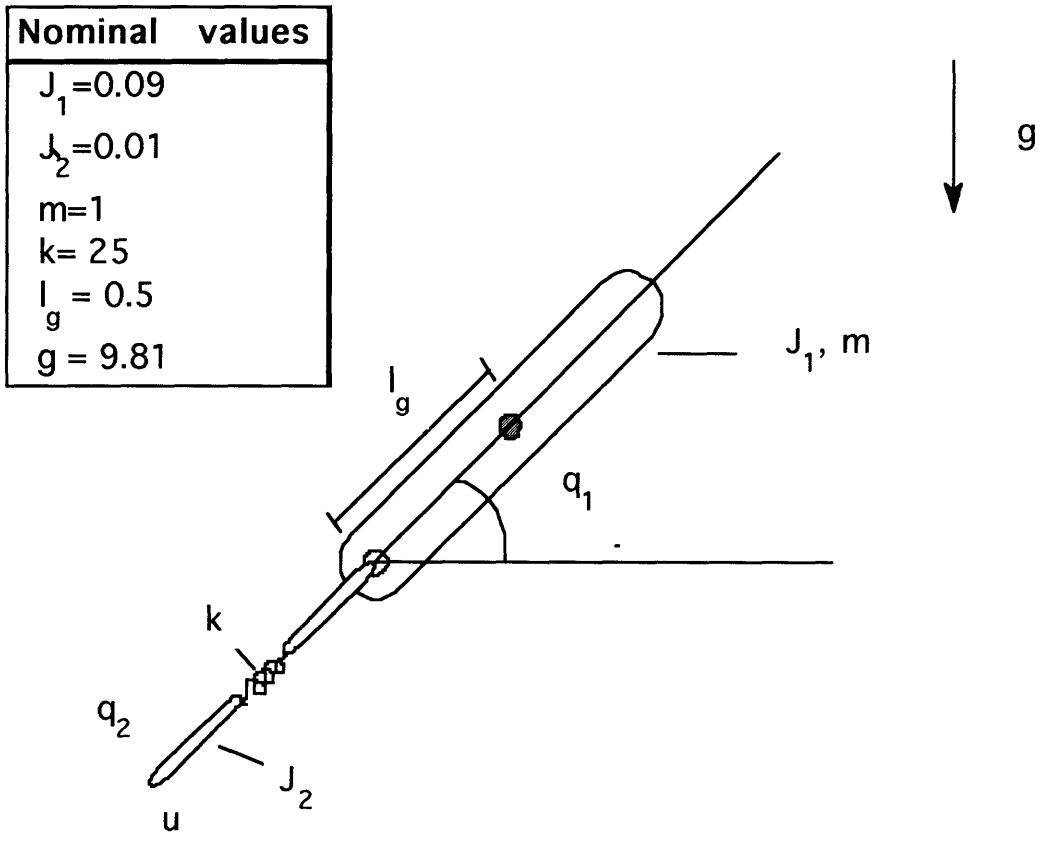

Figure 3.7.2: One link flexible robot.

From the figure, using Newton's law of momentum, we find the equations

$$\begin{aligned} J_1\ddot{q}_1 + m_1 g l_g \sin q_1 + k(q_1 - q_2) &= 0 \\ J_2\ddot{q}_2 - k(q_1 - q_2) &= u \end{aligned} \tag{3.7.6}$$

Writing the model in state space form by setting

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} q_1 \\ \dot{q}_1 \\ q_2 \\ \dot{q}_2 \end{bmatrix} \tag{3.7.7}$$

we get

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ -\dfrac{m_1 g l_g \sin(x_1) + k(x_1 - x_3)}{J_1} \\ x_4 \\ \dfrac{k(x_1 - x_3)}{J_2} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \dfrac{1}{J_2} \end{bmatrix} u = f(\mathbf{x}) + g(\mathbf{x})\, u \tag{3.7.8}$$

The Taylor linearized system can be written

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\dfrac{m_1 g l_g + k}{J_1} & 0 & \dfrac{k}{J_1} & 0 \\ 0 & 0 & 0 & 1 \\ \dfrac{k}{J_2} & 0 & -\dfrac{k}{J_2} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \dfrac{1}{J_2} \end{bmatrix} u \tag{3.7.9}$$

The measurement is assumed to be

$$y = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} q_1 \\ \dot{q}_1 \\ q_2 \\ \dot{q}_2 \end{bmatrix} \tag{3.7.10}$$

i.e. collocated position and rate measurement.

The transfer functions for the linearized systems, using the following parameter values;

$$J = 0.1,\ m = 1,\ l_g = 0.5,\ J_1 = 0.09,\ J_2 = 0.01 \qquad (3.7.11)$$

are given in figure 3.7.3. We observe in particular that there are two resonant frequencies for the flexible model.

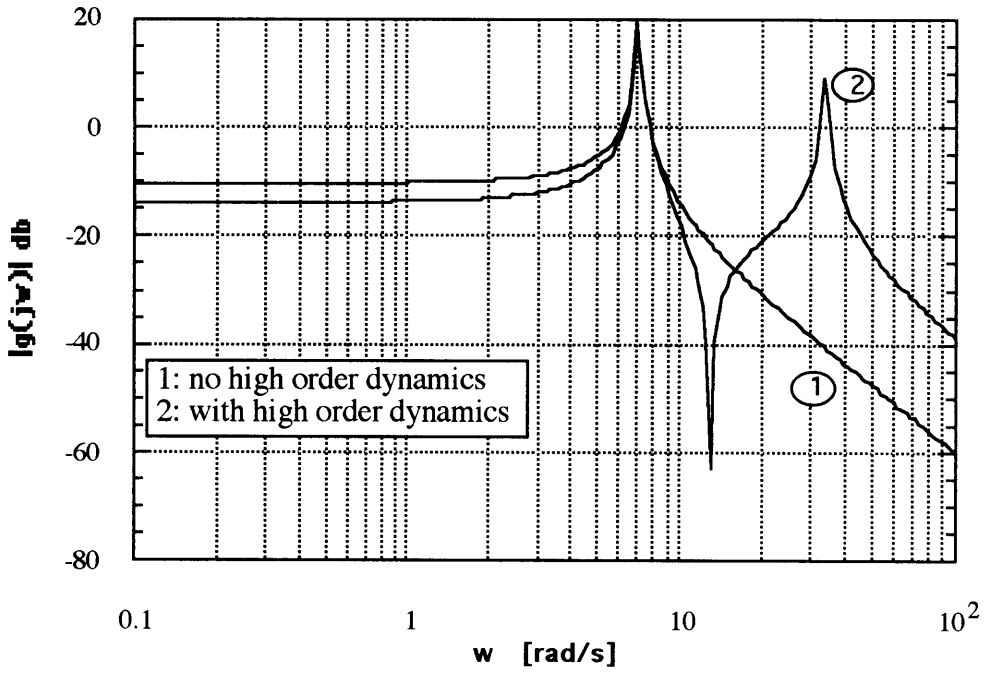

Figure 3.7.3: The transfer functions for the linearized rigid- and flexible robot models.

In order to obtain good command following in this case we have to use both position and rate measurement, therefore, PD controllers instead of LQG/LTR controllers are used.

3.7.1. The Feedback Linearized Controller

Find a diffeomorphism $\mathbf{z}=\Phi(\mathbf{x})$ to transform the linear system in to a linear one. In this case this is trivial.

$$\begin{aligned} z_1 &= x_1 \\ z_2 &= x_2 \end{aligned} \tag{3.7.12}$$

The linearizing control is then given by

$$v = \frac{1}{\hat{J}} u + \frac{\hat{m}\hat{l}_g g}{\hat{J}} \sin(z_1) \tag{3.7.13}$$

and the transformed system is given by

$$\begin{aligned} \dot{z}_1 &= z_2 \\ \dot{z}_2 &= v \end{aligned} \tag{3.7.14}$$

Assume that the derivative of the desired trajectory is available. We must transform the desired trajectory into the z-plane. In this case this is particularly easy

$$\begin{aligned} z_d &= x_d \\ \dot{z}_d &= \dot{x}_d \end{aligned} \tag{3.7.15}$$

Design a PD controller of the form

$$v = k_2 \dot{e} + k_1 e \tag{3.7.16}$$

where

$$\begin{aligned} e &= z_d - z_1 \\ \dot{e} &= \dot{z}_1 - \dot{z}_d \end{aligned} \tag{3.7.17}$$

A set of parameter values is found from classical Bode design to be

$$\begin{aligned} k_1 &= 100 \\ k_2 &= 20 \end{aligned} \tag{3.7.18}$$

3.7.2. The PD controller Based on a Taylor Linearization

The PD controller is of the form

$$u = p_1 e + p_2 \dot{e} \tag{3.7.19}$$

where

$$e = r - x_1$$
$$\dot{e} = \dot{r} - \dot{x}_1 = \dot{r} - x_2 \tag{3.7.20}$$

and p_1, p_2 are constants. Based on the nominal parameter values and a classical Bode design a set of values are found to be

$$p_1 = 145$$
$$p_2 = 8 \tag{3.7.21}$$

The open loop gain plot for the system with the controller is shown in figure 3.7.4

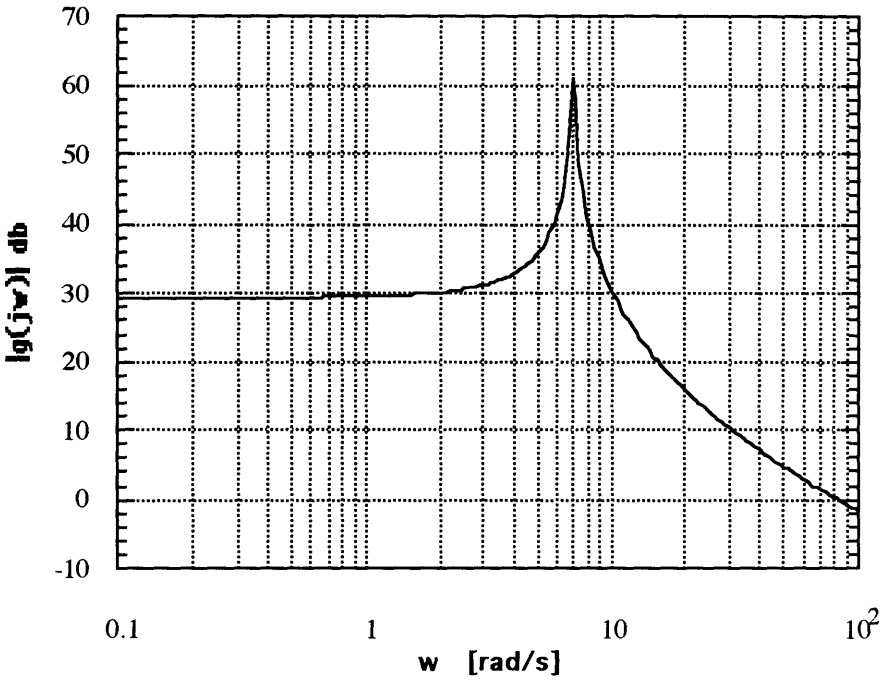

Figure 3.7.4: Gain plot for nominal robot model with PD controller.

4. Simulations

The type of simulations done are often called Monte Carlo simulations i.e. the same algorithm is repeated over and over again with slightly different sets of parameter values. The performance is measured each time and in the end one can plot the performance as a function of the parameter values.

The task of the controllers compared in this study has been to track a reference signal of the type

$$r(t) = A \sin(\omega t) \tag{4.1.1}$$

over a fixed interval of time. The frequency ω of the reference signal has been chosen to be lower than the first mode of the corresponding system. The amplitude A is supposed to excite the nonlinear effects in the equations in a moderate way. Finally, the interval is chosen to be long enough to reduce the effect of the transients but short enough to make the simulations possible in terms of time considerations. The values used throughout the simulations are $A = 2$, $\omega = 0.5$rad/s, and the interval has been 20s.

As a measure on how well a specific controller is doing, the discrete 2-norm of the error signal is used i.e.

$$J = \| e(k) \|_2 = \| r(k) - x_1(k) \|_2 = \sum_{k=1}^{N} (r(k) - x_1(k)) \tag{4.1.2}$$

where J is the performance measure, k is an abbreviation for k*T where T is the sampling interval.

In the simulations the effect of both structured- (parameter error (or parameter mismatch between the model used for designing a controller and the actual model)) and unstructured, (process noise, and unmodeled dynamics) model errors are investigated. The effect of the different types of errors is studied both separately and combined.

The typical simulation diagram looks like the one given in figure 4.1.1. The measurement

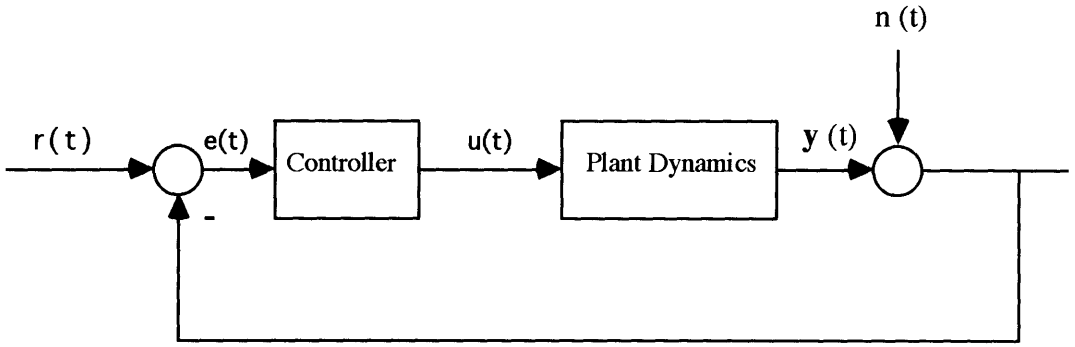

Figure 4.1.1: The simulation scheme.

noise is white noise with unit intensity scaled by a factor n (figure 4.1.2). The value of n has been chosen to be n = 0.05 when nothing else is mentioned.

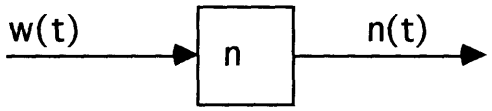

Figure 4.1.2: The measurement noise.

The integration algorithm used is a built in SIMULAB algorithm called LINSIM. According to [12] it is designed for systems that are primarily linear but contains a few non-linear blocks and is particularly good for so-called stiff systems i.e. large numerical difference between the fastest and slowest modes of the system. This occurs when unmodeled low damped high order dynamics are introduced in the equations. The time step used in the simulations is T = 0.01s except when the effect of low damped high order dynamics is studied. Then the time step used is T = 0.0025s.

The simulations are done on a Macintosh IIsi and the algorithms are implemented using

SIMULAB/MATLAB software. See Appendix A for a detailed listing of the programs and [12] for details about the SIMULAB/MATLAB environment.

5. Results

This chapter contains the simulation results. The results are given for each system at a time. First the robustness of the controllers with respect to parameter error is explored. The actual values of the parameters are varied linearly in the range of 20% - 400% of the nominal value. Second the robustness with respect to the level of measurement noise is studied. The scaling factor n is varied linearly between 0.01 and 0.1. Third the robustness with respect to unmodeled dynamics is compared. The unmodeled dynamics is simulated using the augmented models with low damped high order dynamics as the actual plants. The value of the 'second spring constant' is varied between 1 and 200 following a logarithmic scale. Finally the effect of parameter error is explored with both measurement error and unmodeled dynamics present. The values used are n = 0.05 for the measurement noise and 'second spring constant' = 10.

The corresponding MATLAB/SIMULAB programs can be found in Appendix A. When reading the programs for the feedback linearization based controllers, it is important to bear in mind the idea of an inner loop and an outer loop as explained in chapter 2.2 even if they appear as one controller.

When comparing the controllers, only a relative comparison can be done. We must look at trends in the performance with increasing modeling error and not the absolute performance itself. The slopes of the graphs will give us this information.

In the following, TL-controller will refer to the controller based on a Taylor linearization of the actual system, and FL-controller will refer to the controller based on the feedback linearization of the actual system.

5.1. Duffing's Equation / Mass on a Nonlinear Spring

Figures 5.1.1 - 5.1.8 present the Monte Carlo simulation results for Duffing's equation using the controllers designed in 3.1. The controllers use the same measurement information i.e. only position measurement is used for both the FL- and TL-controller.

5.1.1. Parameter Error

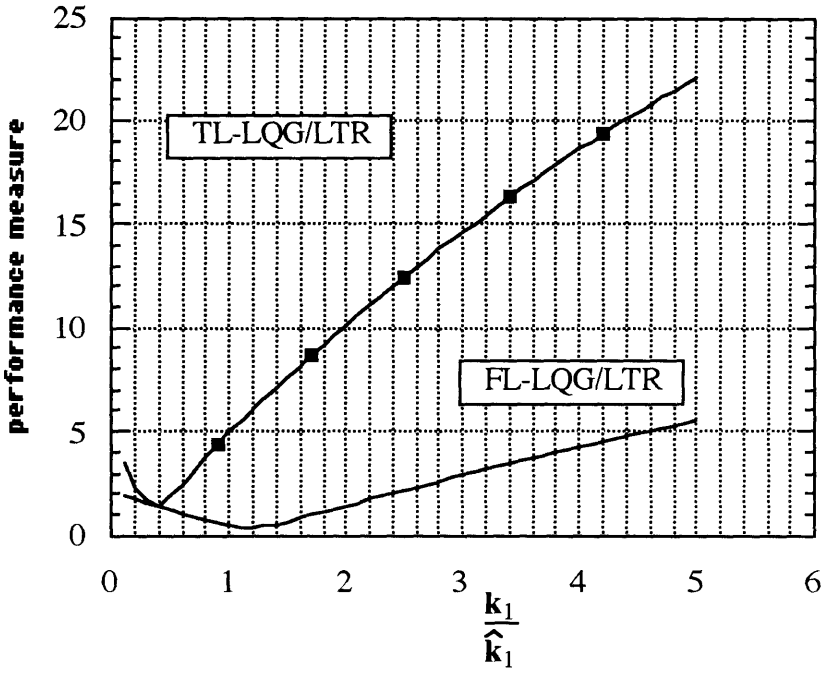

Figure 5.1.1: Performance as a function of parameter error in k1.

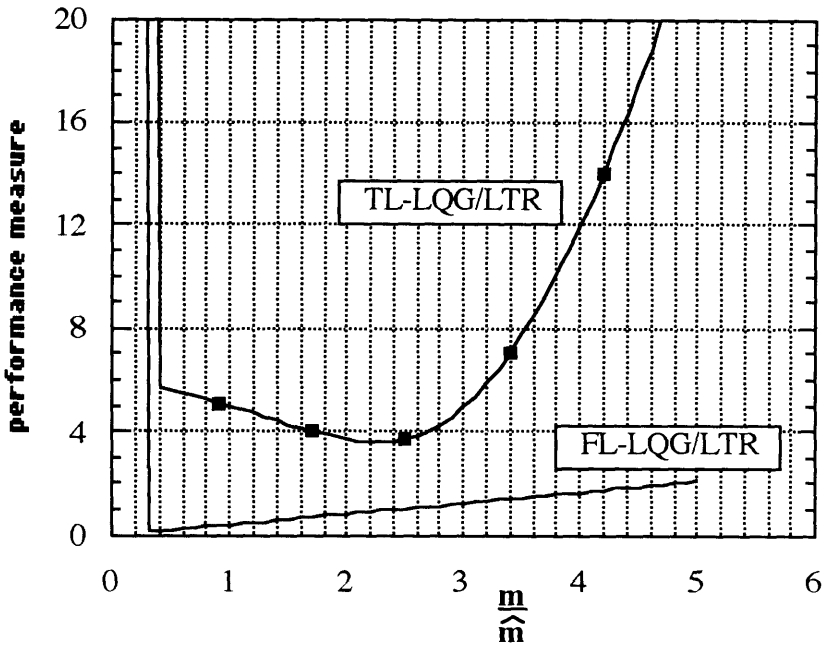

Figure 5.1.2: Performance as a function of parameter error in m.

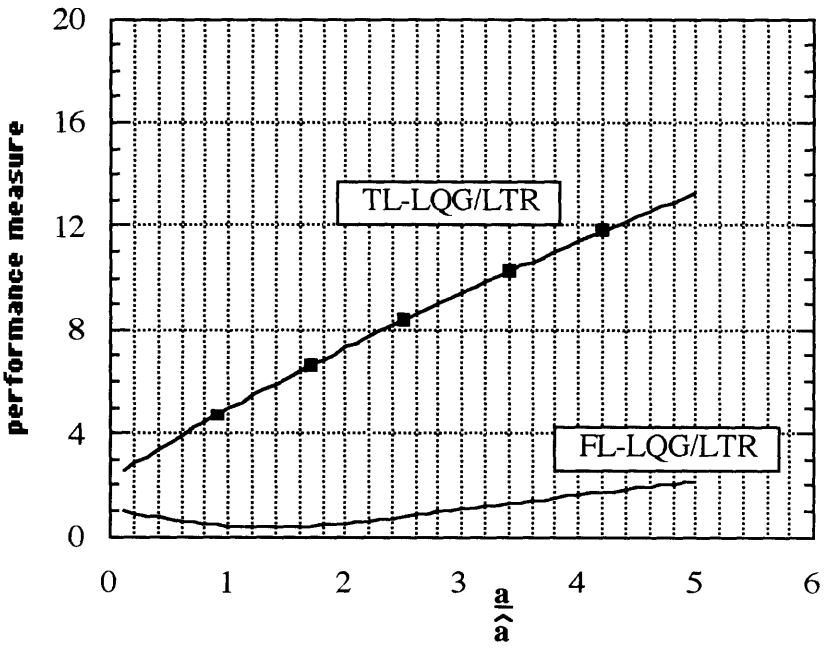

Figure 5.1.3: Performance as a function of parameter error in a.

5.1.2. Measurement Noise

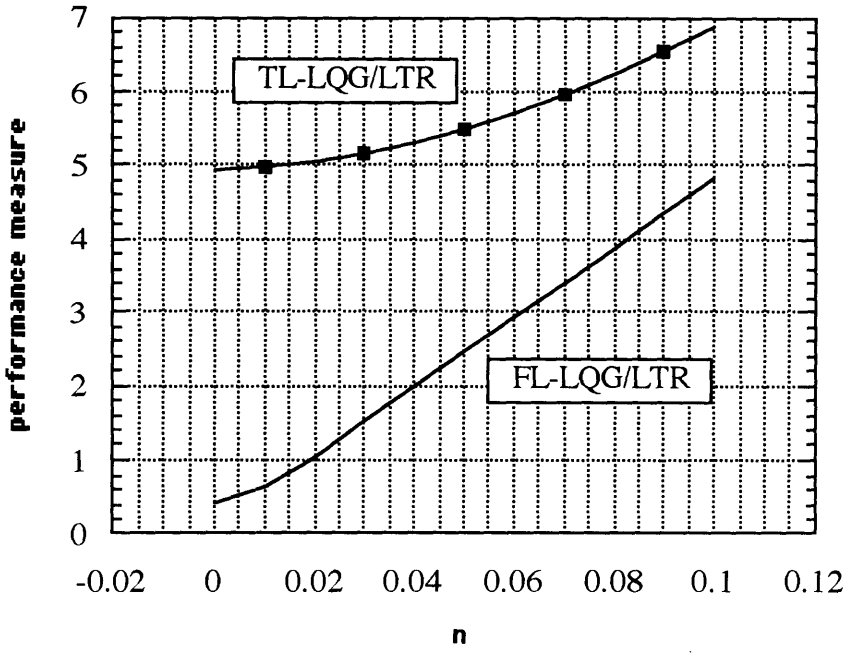

Figure 5.1.4: Performance dependence on measurement noise.

5.1.3. Unmodeled Dynamics

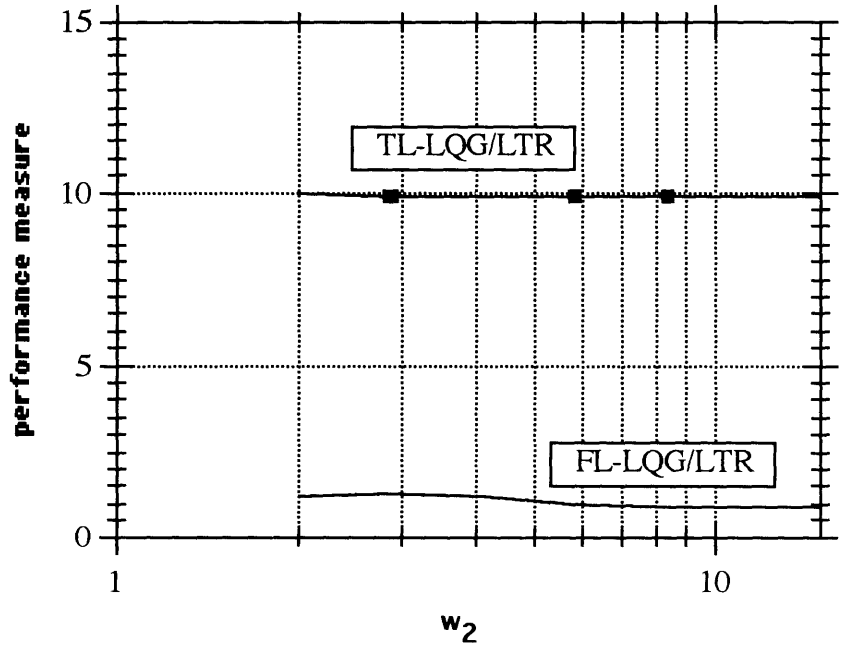

Figure 5.1.5: Performance as a function of frequency for unmodeled dynamics.

5.1.4. Parameter Error with Measurement Noise and Unmodeled Dynamics.

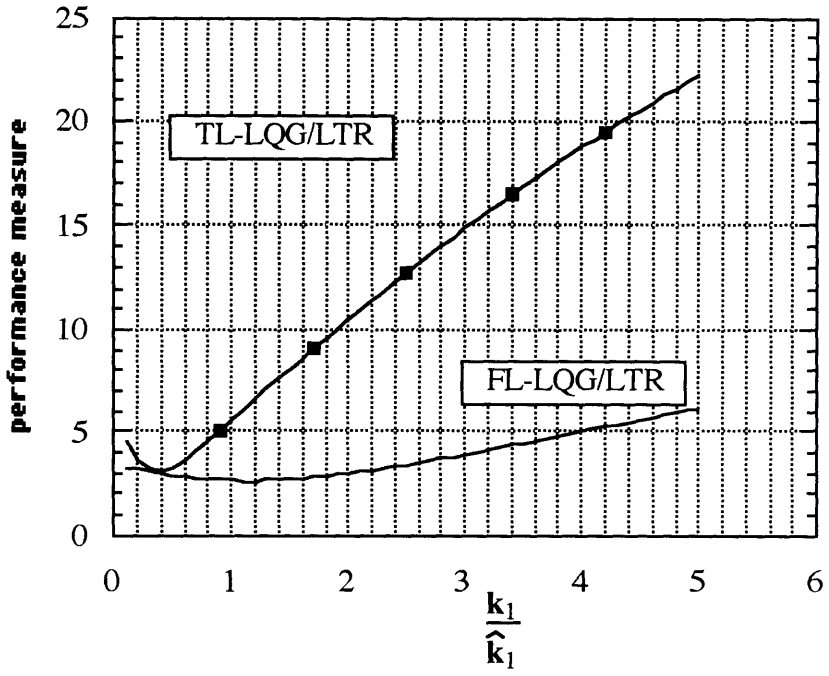

Figure 5.1.6: Performance as a function of parameter error in k1 with measurement noise and unmodeled dynamics at w=5 [rad/s].

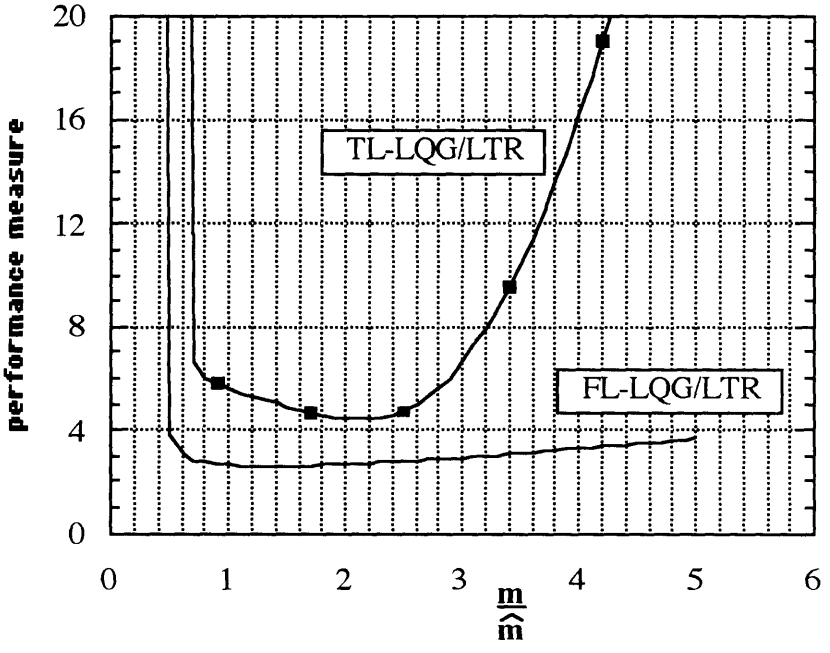

Figure 5.1.7: Performance as a function of parameter error in m with measurement noise and unmodeled dynamics at w=5 [rad/s].

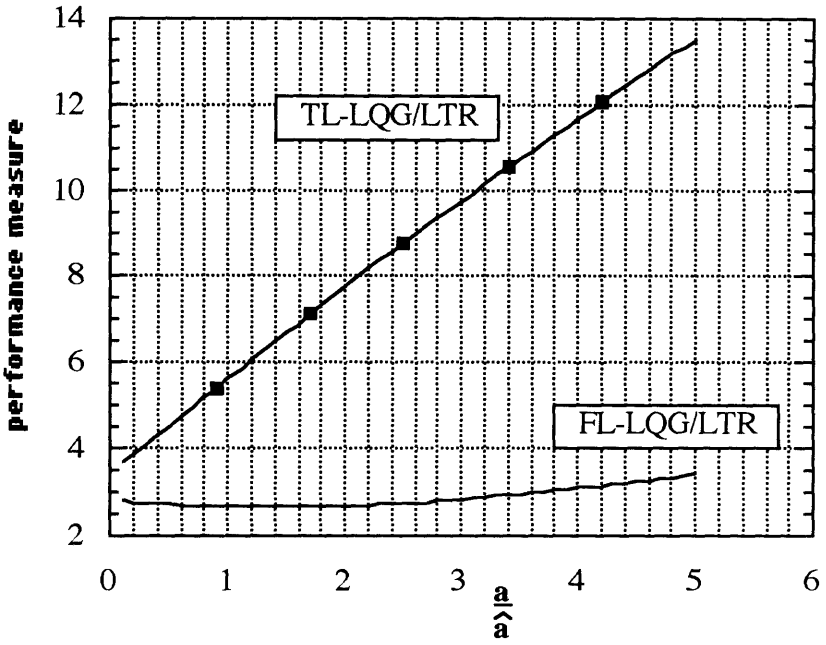

Figure 5.1.8: Performance as a function of parameter error in m with measurement noise and unmodeled dynamics at w=5 [rad/s].

For parameter errors only (figures 5.1.1 - 5.1.3) the FL-controller is more robust. This is especially true for parameter errors in the mass. For small actual values of the mass ($\frac{m}{\hat{m}} <$ 0.4) both systems are unstable, but the system with the TL-controller goes unstable first ($\frac{m}{\hat{m}} \approx 0.4$) (figure 5.1.2). As expected the TL-controller is more robust with respect to measurement noise (figure 5.1.4). The frequency at which the unmodeled dynamics occur seems to have little influence on the performance (figure 5.1.5). With measurement noise (n = 0.05) and unmodeled dynamics ($k_2 = 10$) present (figures 5.1.6 - 5.1.8) the FL-controller is still more robust with respect to parameter error. In fact, the relative performance of the controllers is very much the same as in figures (5.1.1 - 5.1.3).

5.2. Mathieu Equation

Figures 5.2.1 - 5.2.6 present the Monte Carlo simulation results for the Mathieu equation using the controllers designed in 3.2. The controllers use the same measurement information i.e. only position measurement is used for both the FL- and TL-controller.

5.2.1. Parameter Error

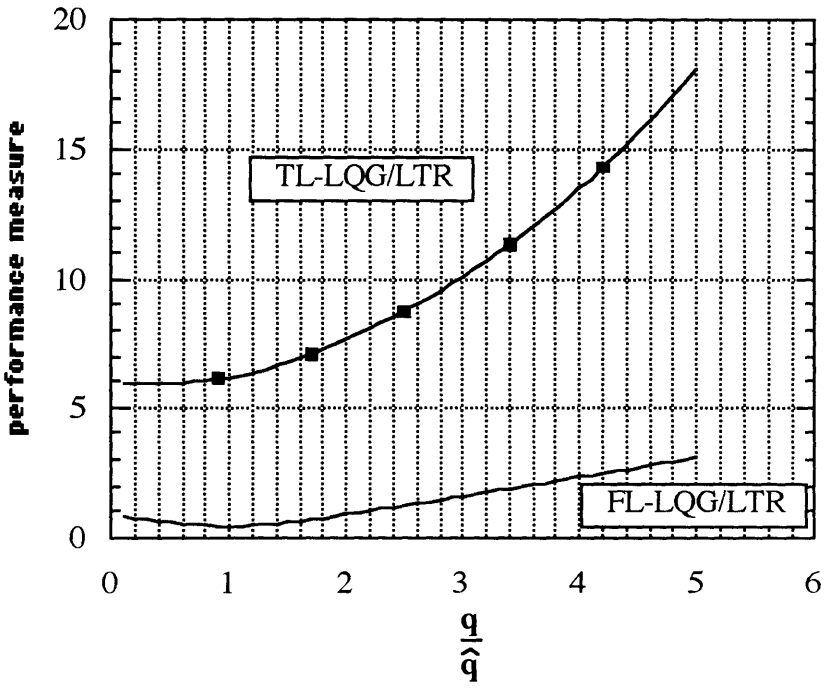

Figure 5.2.1: Performance as a function of parameter error in q.

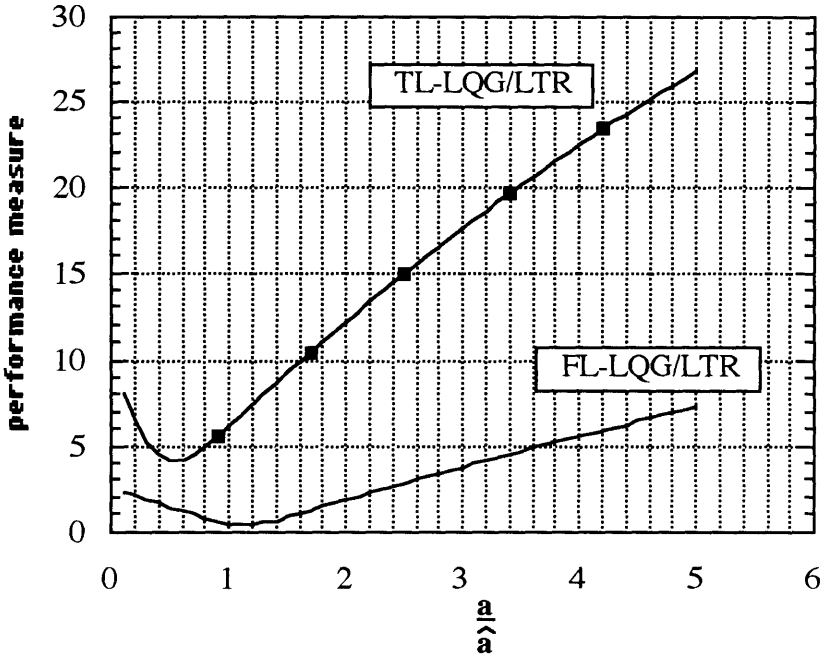

Figure 5.2.2: Performance as a function of parameter error in a.

5.2.2. Measurement Noise

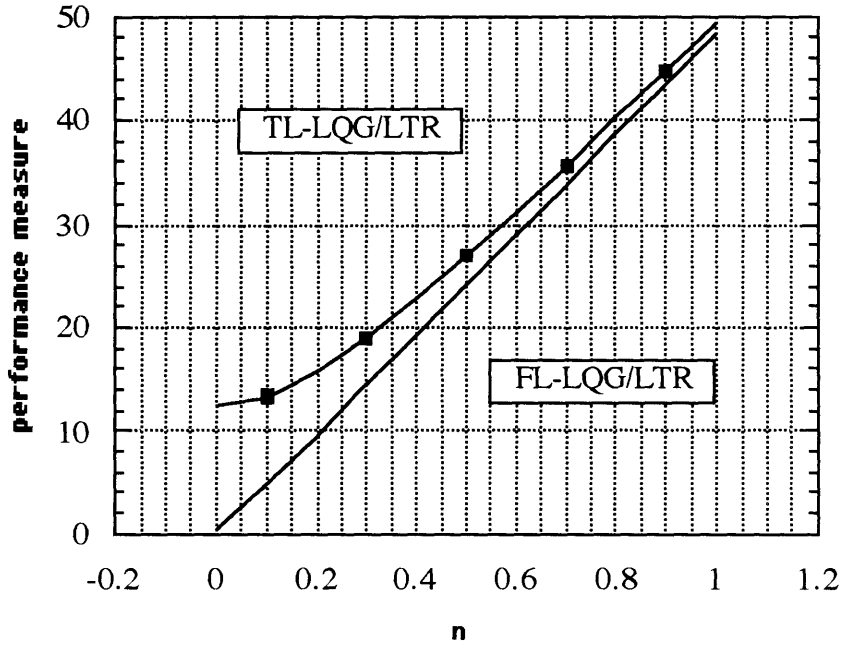

Figure 5.2.3: Performance dependence on measurement noise.

5.2.3. Unmodeled Dynamics

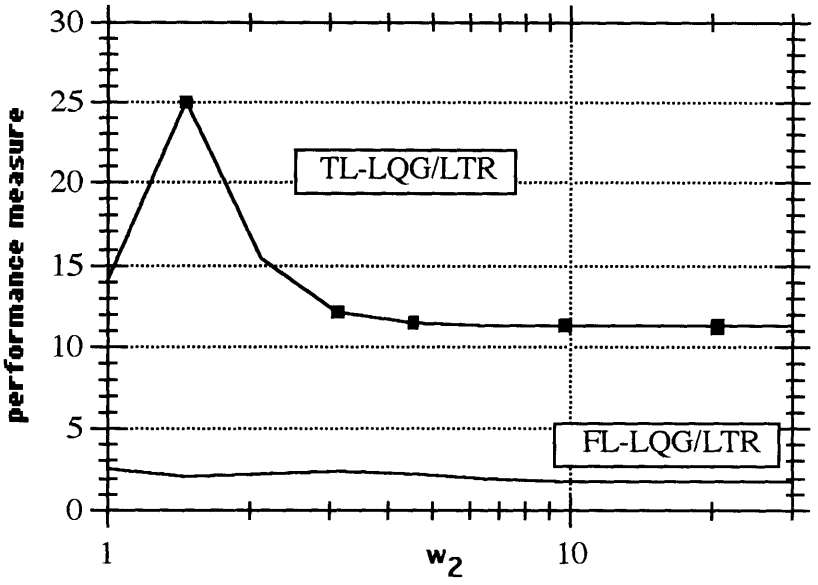

Figure 5.2.4: Performance as a function of frequency for unmodeled dynamics.

5.2.4. Parameter Error with Measurement Noise and Unmodeled Dynamics

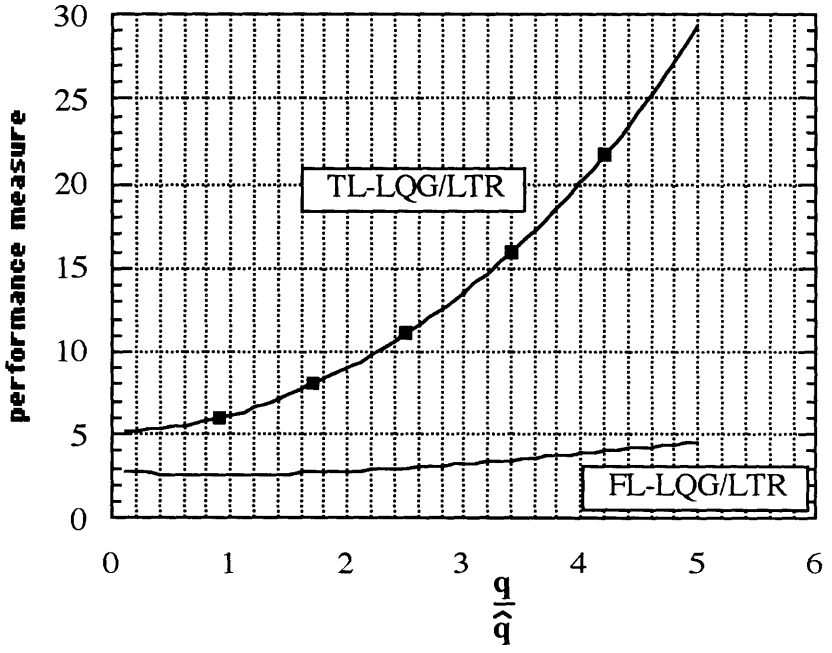

Figure 5.2.5: Performance as a function of parameter error in q with measurement noise and unmodeled dynamics at w=5 [rad/s].

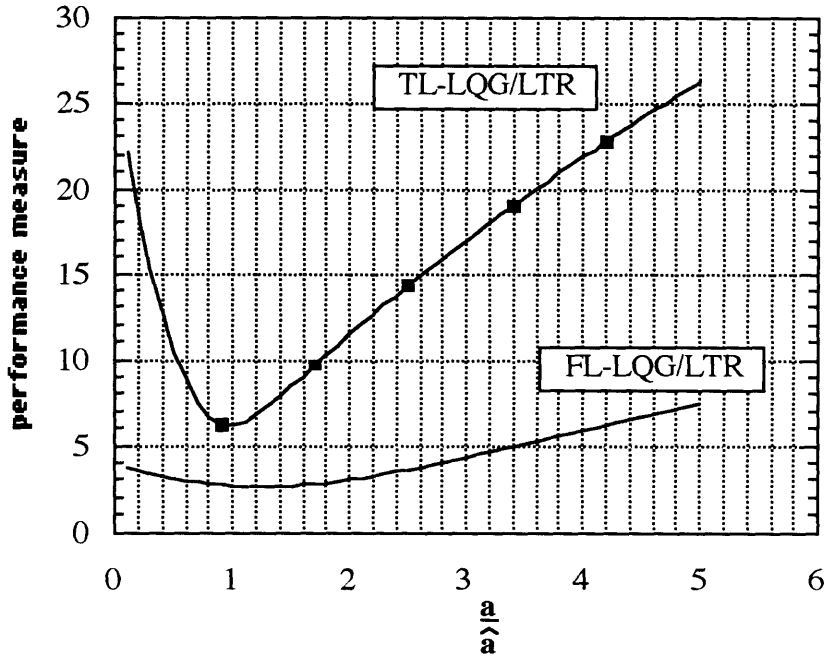

Figure 5.2.6: Performance as a function of parameter error in a with measurement noise and unmodeled dynamics at w=5 [rad/s].

For parameter errors only (figures 5.2.1 - 5.2.2) the FL-controller is more robust. With respect to measurement noise (figure 5.2.3) the performance is approximately the same. The frequency at which the unmodeled dynamics occur seems to have little influence on the performance of the FL-controller (figure 5.2.4) .The TL-controller is sensitive to unmodeled dynamics with frequencies close to the first mode (1 rad/s). With measurement noise (n = 0.05) and unmodeled dynamics (k_1 = 10) present (figures 5.2.5 - 5.2.6) the FL-controller is still more robust with respect to parameter error. In fact, the relative performance of the controllers is very much the same as in figures (5.2.1 - 5.2.2).

5.3. Mass with Discontinuous Stiffness

Figures 5.3.1 - 5.3.8 present the Monte Carlo simulation results for the mass/spring system with discontinuous stiffness using the controllers designed in 3.3. The controllers use the same measurement information i.e. only position measurement is used for both the FL- and TL-controller.

5.3.1. Parameter Error

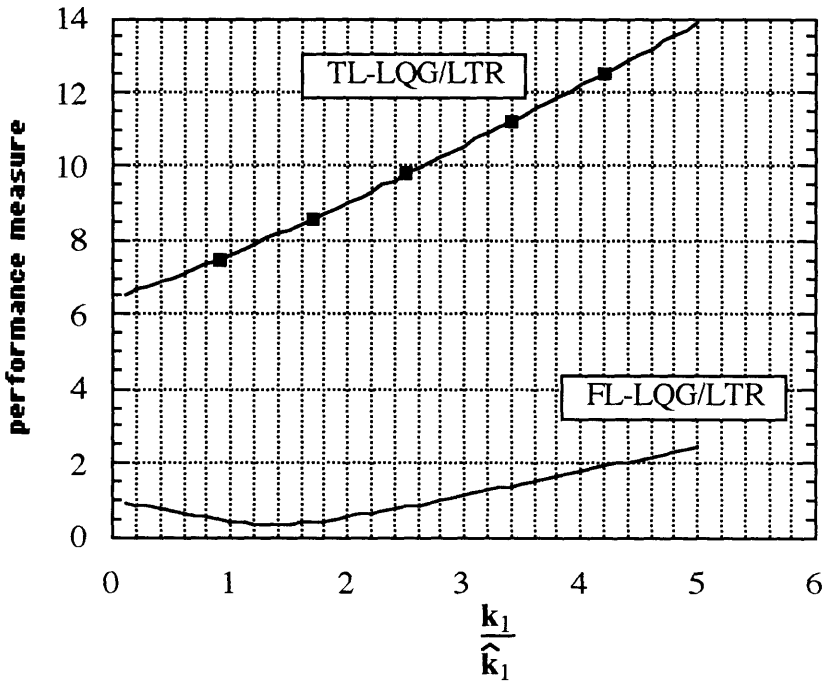

Figure 5.3.1: Performance as a function of parameter error in k1.

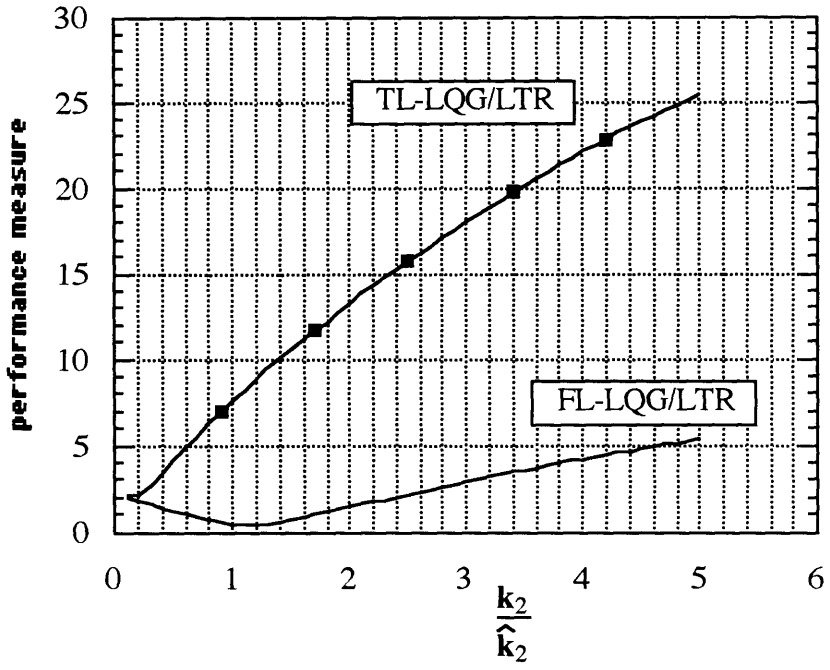

Figure 5.3.2: Performance as a function of parameter error in k2.

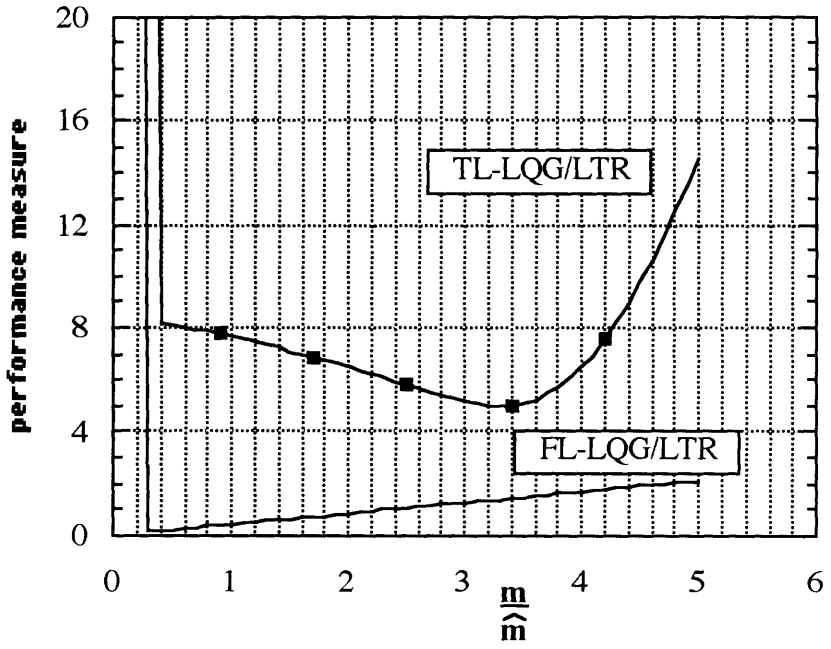

Figure 5.3.3: Performance as a function of parameter error in m.

5.3.2. Measurement Noise

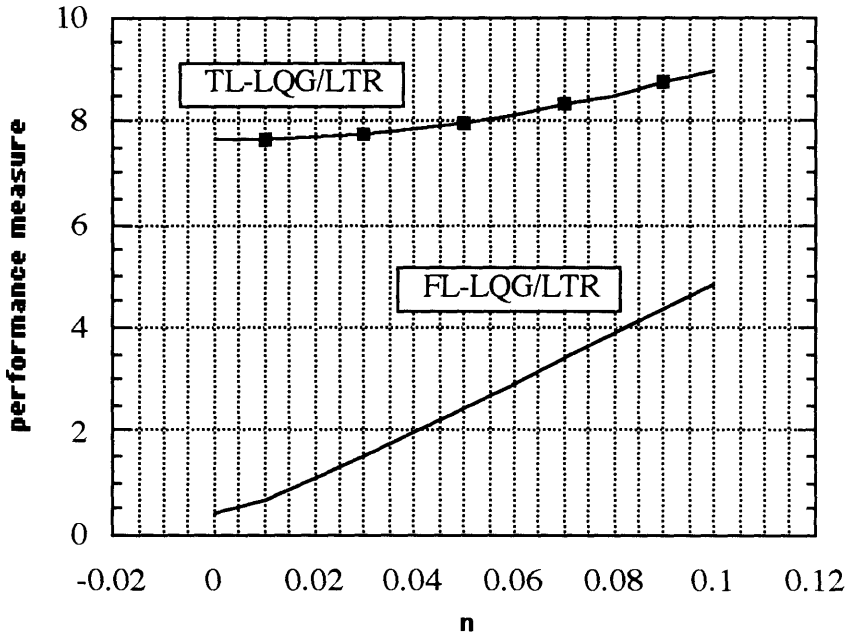

Figure 5.3.4: Performance dependence on measurement noise.

5.3.3. Unmodeled Dynamics

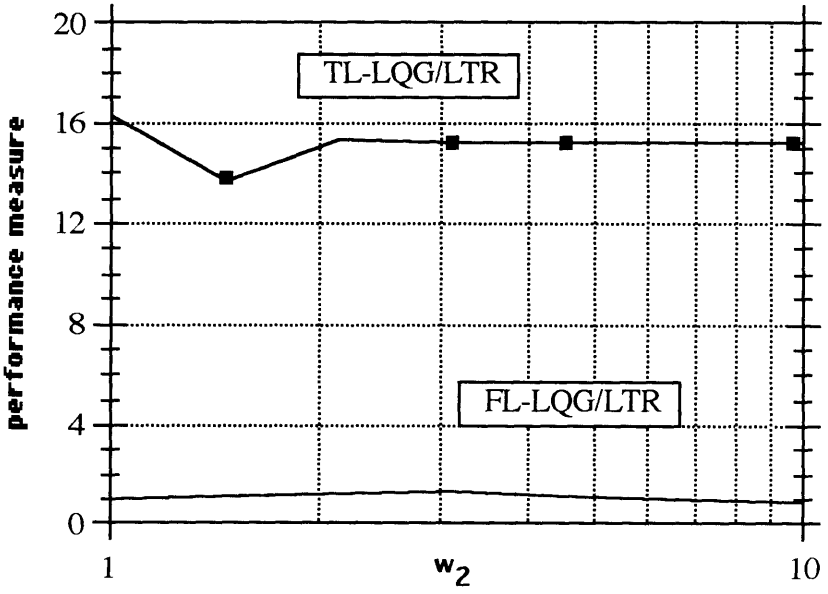

Figure 5.3.5: Performance as a function of frequency for unmodeled dynamics.

5.3.4. Parameter Error with Measurement Noise and Unmodeled Dynamics

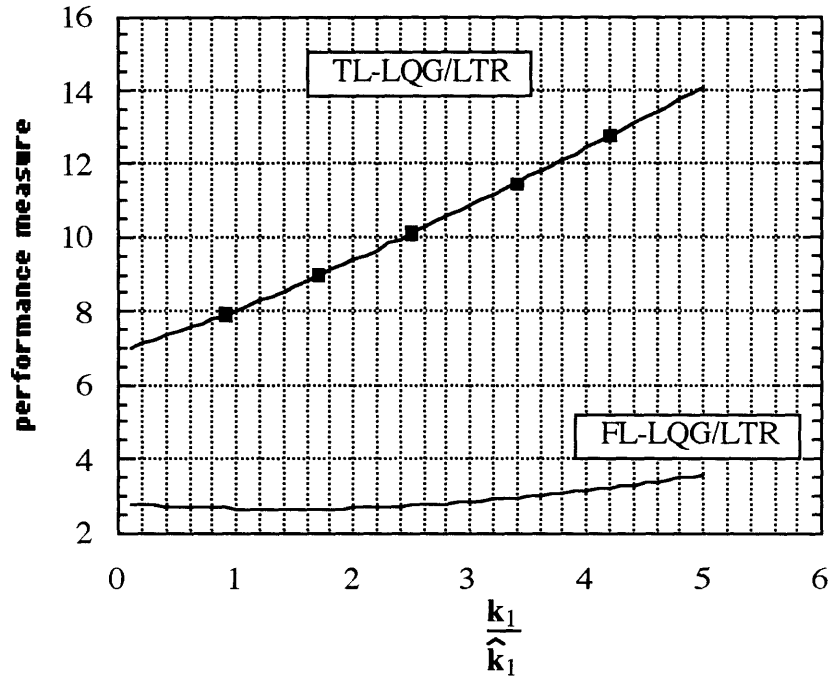

Figure 5.3.6: Performance as a function of parameter error in k1 with measurement noise and unmodeled dynamics at w=5 [rad/s].

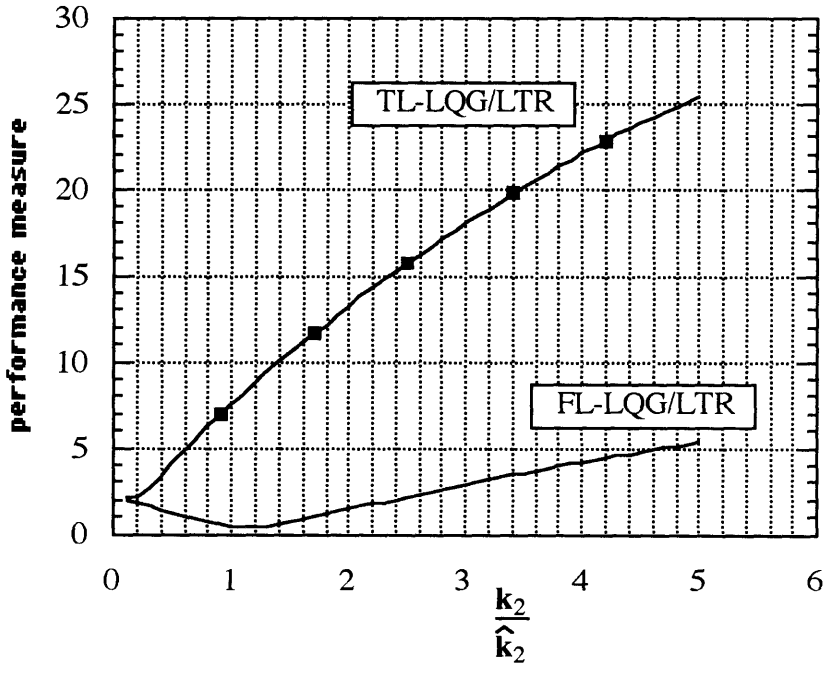

Figure 5.3.7: Performance as a function of parameter error in k2 with measurement noise and unmodeled dynamics at w=5 [rad/s].

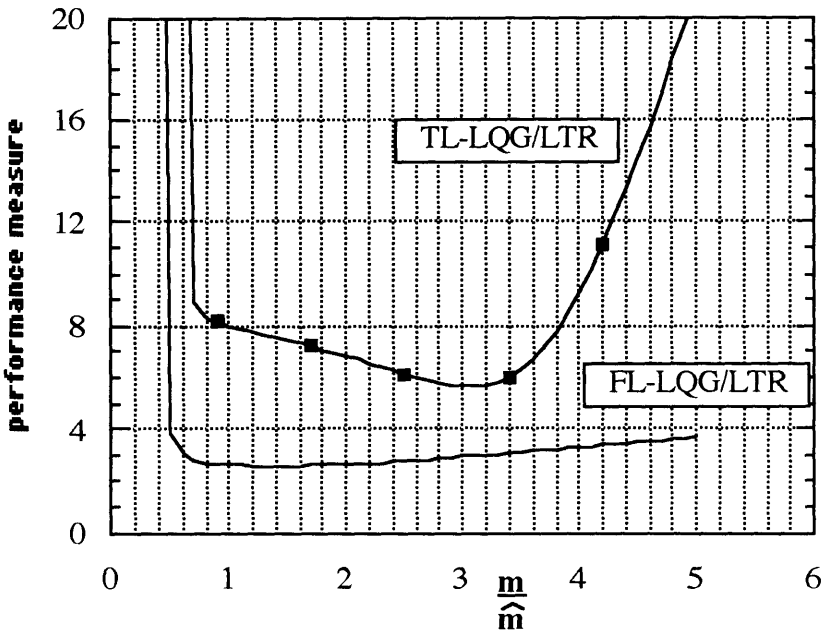

Figure 5.3.8: Performance as a function of parameter error in m with measurement noise and unmodeled dynamics at w=5 [rad/s].

For parameter errors only (figures 5.3.1 - 5.3.3) the FL-controller is more robust. This is especially true for parameter errors in the mass. For small actual values of the mass ($\frac{m}{\hat{m}} < 0.4$) both systems are unstable, but the system with the TL-controller goes unstable ($\frac{m}{\hat{m}} \approx 0.4$) first (figure 5.3.2). As expected the TL-controller is more robust with respect to measurement noise (figure 5.3.4). The frequency at which the unmodeled dynamics occur seems to have little influence on the performance (figure 5.3.5). With measurement noise ($n = 0.05$) and unmodeled dynamics ($k_3 = 10$) present (figures 5.3.6 - 5.3.8) the FL-controller is still more robust with respect to parameter error. In fact, the relative performance of the controllers as a function of parameter error, with unmodeled dynamics present in the actual plant, is very much the same as without unmodeled dynamics

5.4. Mass w/Dry Friction

Figures 5.4.1 - 5.4.8 present the Monte Carlo simulation results for the mass/spring system with dry friction using the controllers designed in 3.4. In this case the FL-controller uses rate information in addition to position information in the inner loop (feedback linearization loop), but since the reference signal is relatively slowly varying the advantage should be limited.

5.4.1. Parameter Error

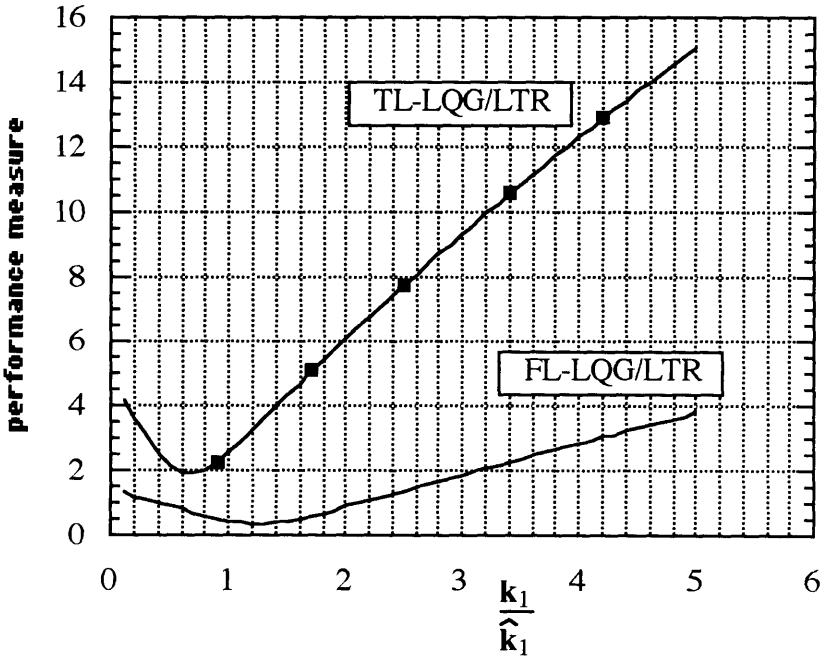

Figure 5.4.1: Performance as a function of parameter error in k1.

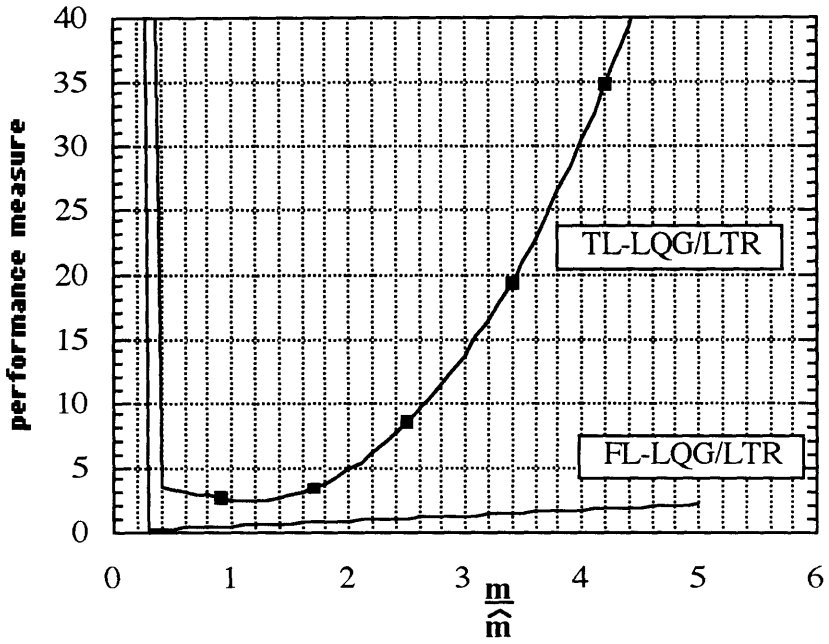

Figure 5.4.2: Performance as a function of parameter error in m.

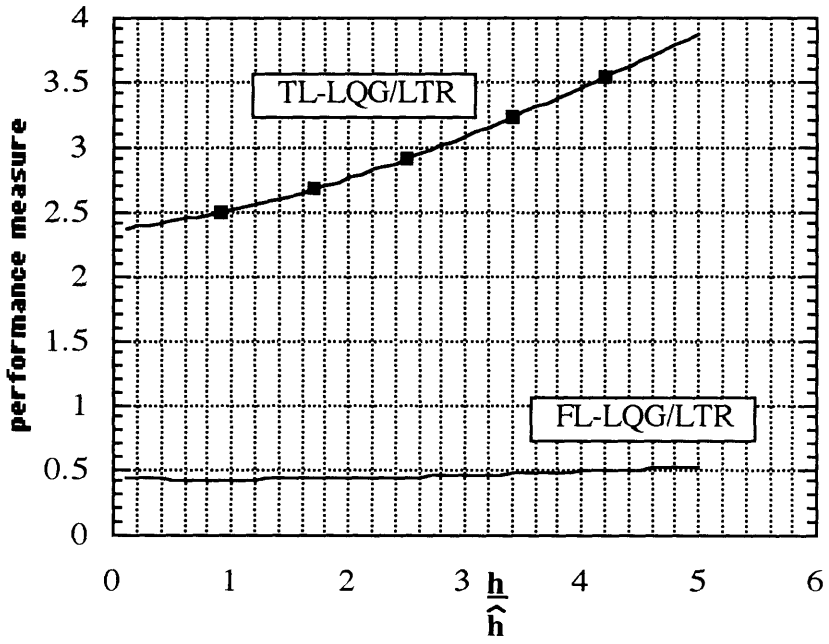

Figure 5.4.3: Performance as a function of parameter error in h.

5.4.2. Measurement Noise

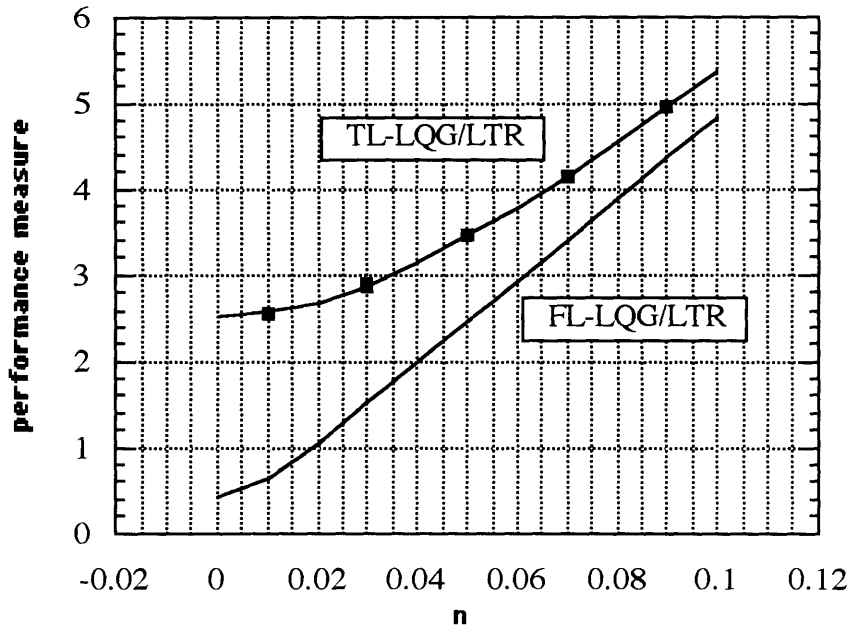

Figure 5.4.4: Performance dependence on measurement noise.

5.4.3. Unmodeled Dynamics

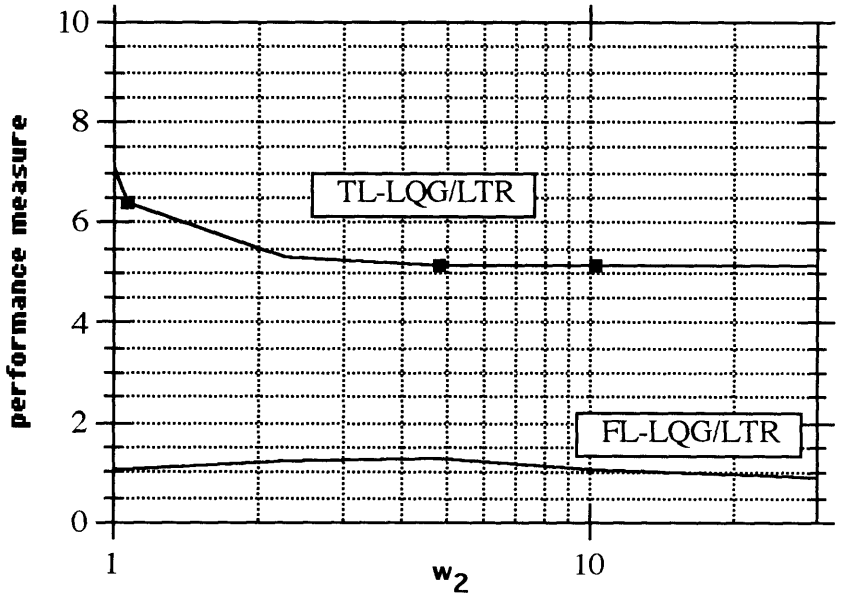

Figure 5.4.5: Performance as a function of frequency for unmodeled dynamics.

5.4.4. Parameter Error with Measurement Noise and Unmodeled Dynamics

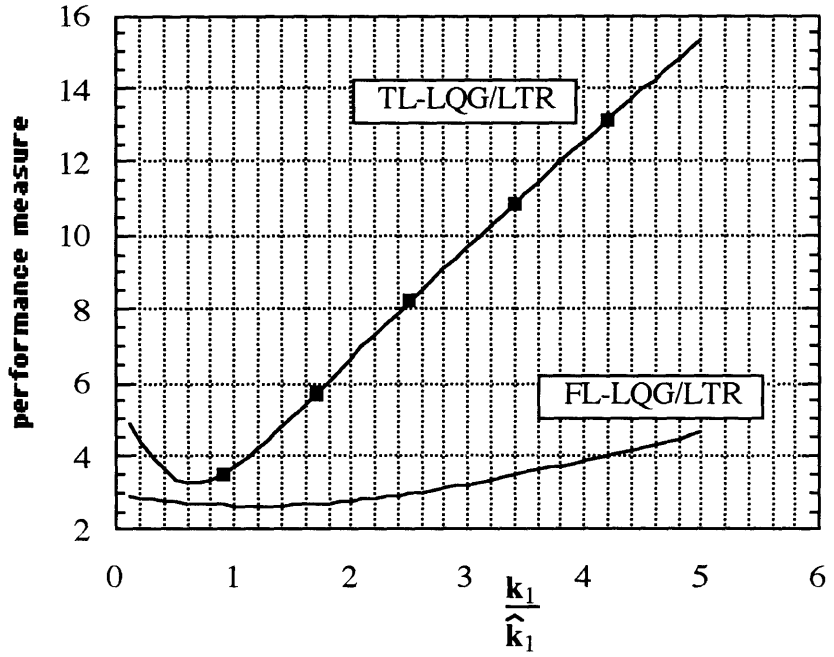

Figure 5.4.6: Performance as a function of parameter error in k1 with measurement noise and unmodeled dynamics at w=5 [rad/s].

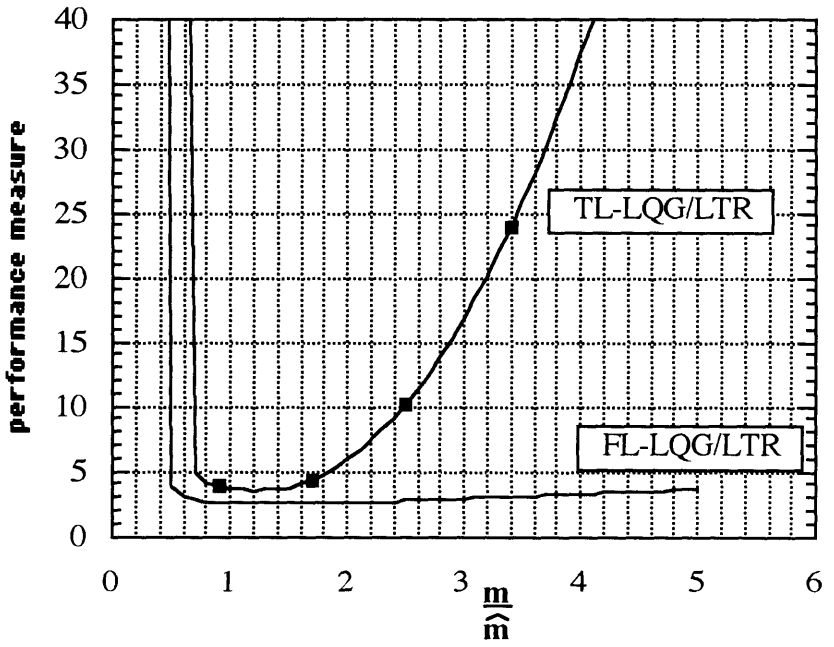

Figure 5.4.7: Performance as a function of parameter error in m with measurement noise and unmodeled dynamics at w=5 [rad/s].

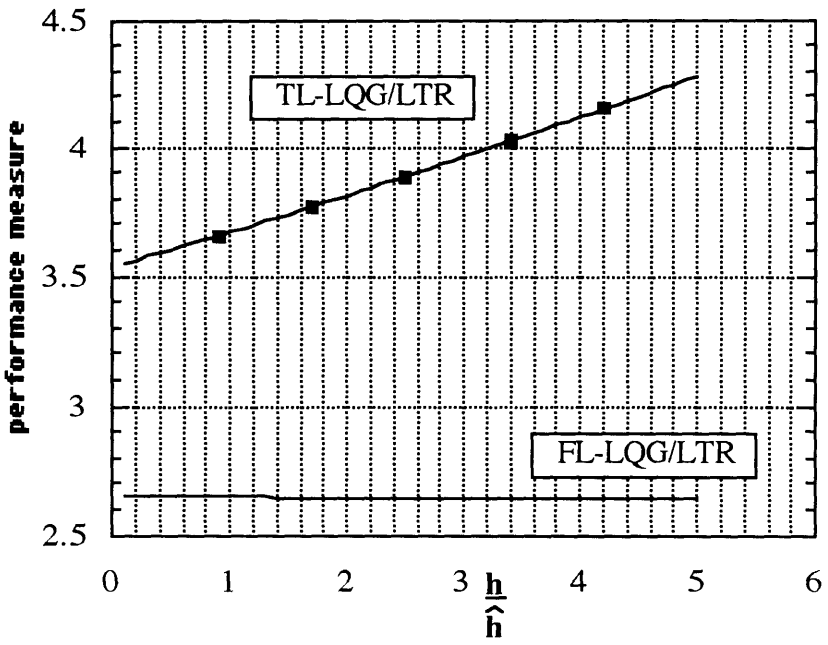

Figure 5.4.8: Performance as a function of parameter error in h with measurement noise and unmodeled dynamics at w=5 [rad/s].

For parameter errors only (figures 5.4.1 - 5.4.3) the FL-controller is more robust. This is especially true for parameter errors in the mass. For small actual values of the mass both systems are unstable ($\frac{m}{\hat{m}} < 0.4$), but the system with the TL-controller goes unstable first ($\frac{m}{\hat{m}} \approx 0.4$) compared to ($\frac{m}{\hat{m}} \approx 0.3$) for the FL-controlled system (figure 5.4.2). Also for parameter errors in h the FL-controller shows better robustness performance. With respect to measurement noise (figure 5.4.4) the controllers seem to perform about the same. The frequency at which the unmodeled dynamics occur seems to have little influence on the performance (figure 5.4.5). With measurement noise (n = 0.05) and unmodeled dynamics ($k_2 = 10$) present (figures 5.4.6 - 5.4.8) the FL-controller is still more robust with respect to parameter error. In fact, the relative performance of the controllers is very much the same as in figures (5.4.1 - 5.4.3).

5.5. Duffing's Equation w/Input Saturation

Figures 5.5.2 - 5.5.8 present the Monte Carlo simulation results for Duffing's equation with a saturation element in the control channel (figure 5.5.1) using the same controllers as in 5.1.

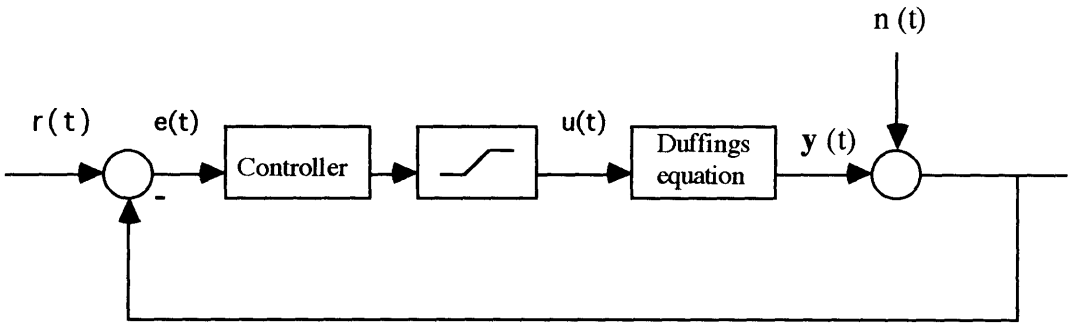

Figure 5.5.1: Simulation diagram.

5.5.1. Saturation Limit

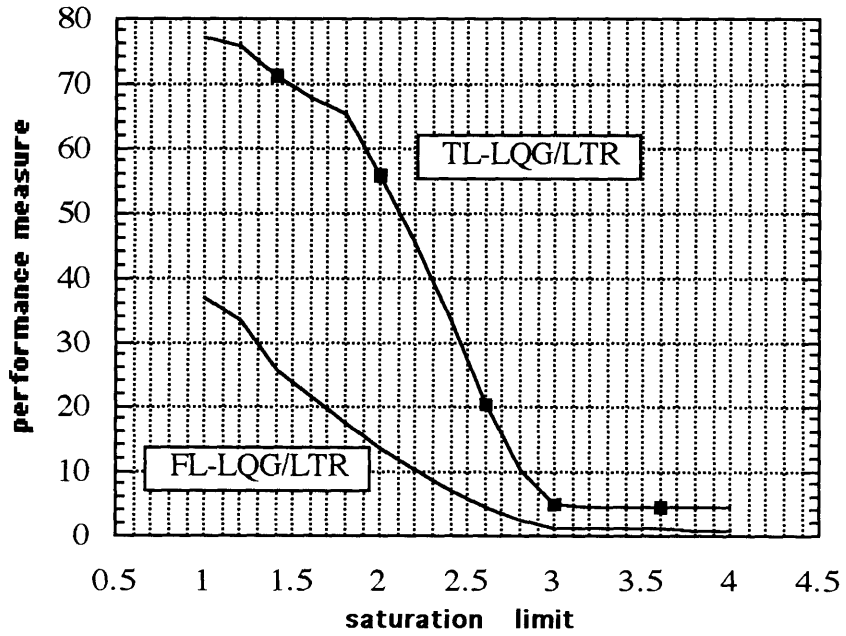

Figure 5.5.2: Performance dependence on saturation limit.

5.5.2. Parameter Error

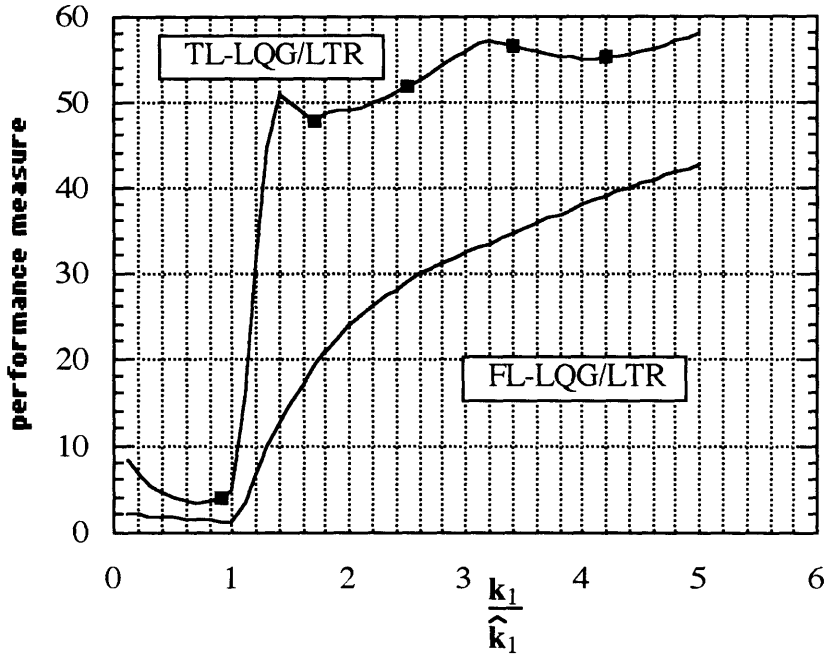

Figure 5.5.3: Performance as a function of parameter error in k1.

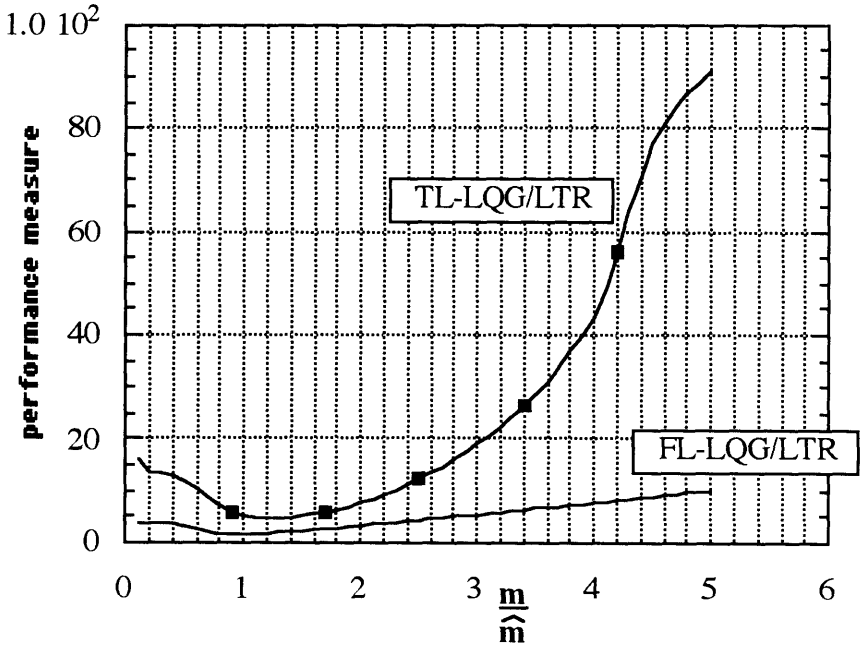

Figure 5.5.4: Performance as a function of parameter error in m.

5.5.3. Measurement Noise

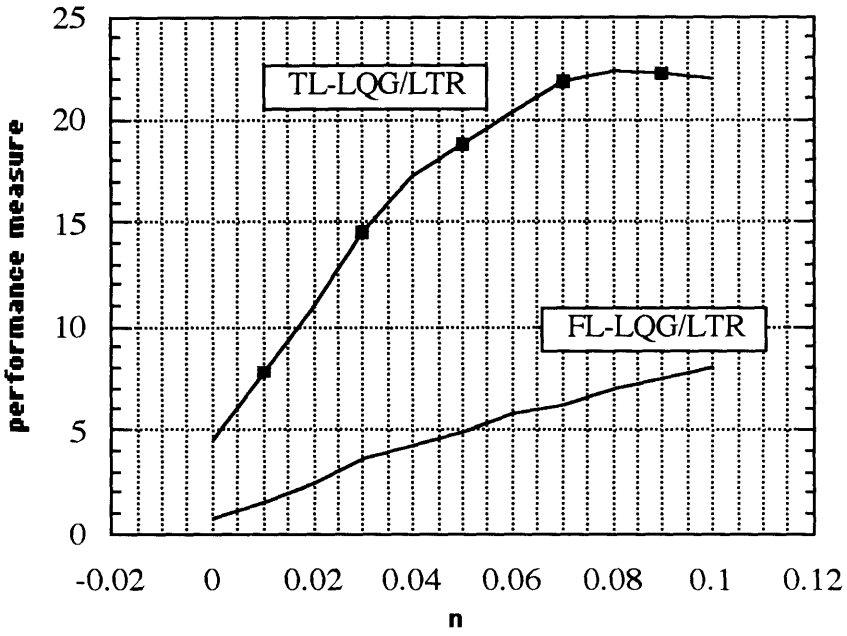

Figure 5.5.5: Performance dependence on measurement noise.

5.5.4. Unmodeled Dynamics

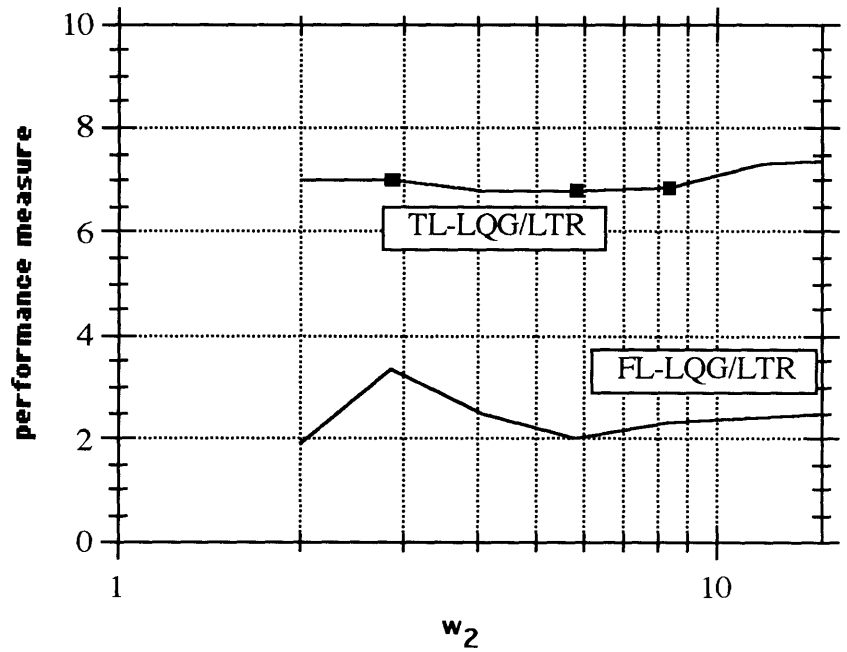

Figure 5.5.6: Performance as a function of frequency for high order dynamics.

5.5.5. Parameter Error with Measurement Noise and Unmodeled Dynamics

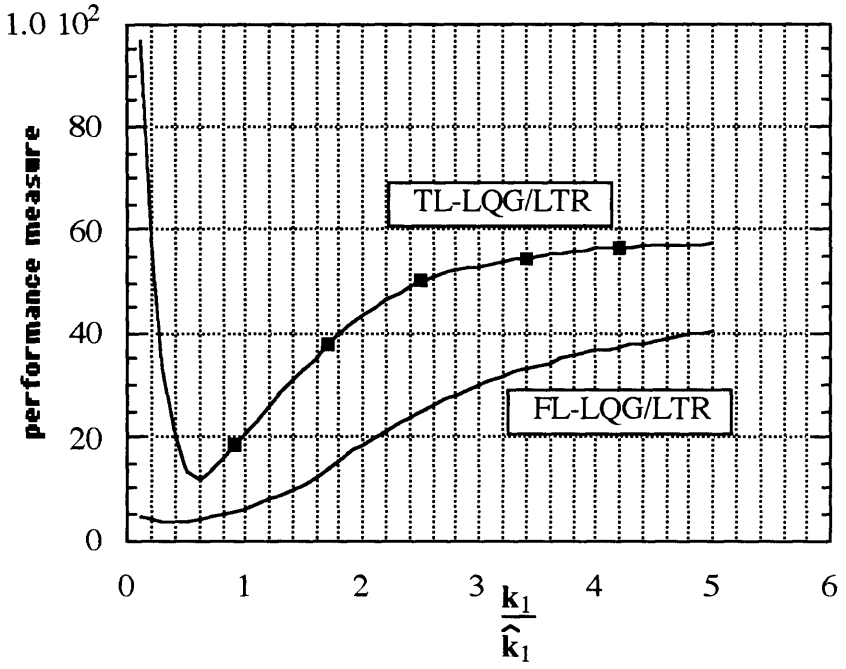

Figure 5.5.7: Performance as a function of parameter error in k1 with measurement noise and unmodeled dynamics at w=5 [rad/s].

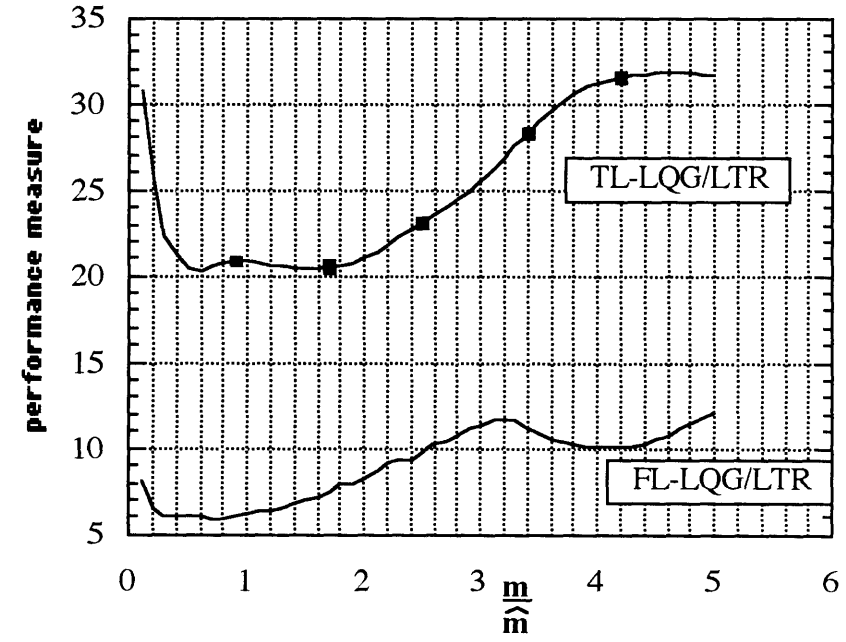

Figure 5.5.8: Performance as a function of parameter error in m with measurement noise and unmodeled dynamics at w=5 [rad/s].

First the effect of the saturation limit is explored (figure 5.5.1). The FL-controller is less sensitive to saturation in the control channel than the TL-controller.

Then the saturation limit is set to 3 and the simulations from 5.1 are repeated (figures 5.5.2 - 5.5.8). For parameter errors only (figures 5.5.3 - 5.5.4) the FL-controller is more robust. This is especially true for parameter errors in the mass (figure 5.5.3). It is interesting that for small actual values of the mass both systems remain stable in this case, but went unstable without the saturation element in the control channel. It looks a limit in the allowable control signals is somehow stabilizing in this case. With respect to measurement noise (figure 5.5.5) the controllers seem to perform about the same. The frequency at which the unmodeled dynamics occur seems to have little influence on the performance (figure 5.5.6). With measurement noise (n = 0.05) and unmodeled dynamics ($k_2 = 10$) present (figures 5.5.7 - 5.5.8), the FL-controller is still more robust with respect to parameter error. In fact, the relative performance of the controllers is very much the same as in figures (5.5.3 - 5.5.4).

5.6. Duffing's Equation with Deadband in Input

Figures 5.6.2 - 5.6.8 present the Monte Carlo simulation results for Duffing's equation with a deadband element in the control channel (figure 5.6.1) using the same controllers as in 5.1.

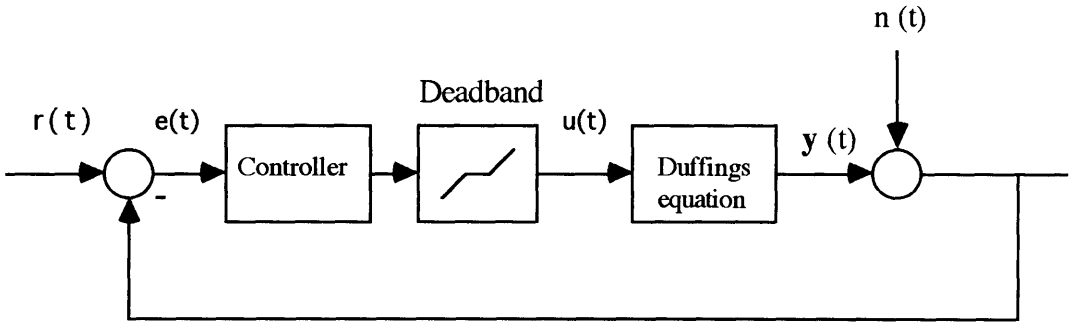

Figure 5.6.1: Simulation diagram.

5.6.1. Deadband Limit

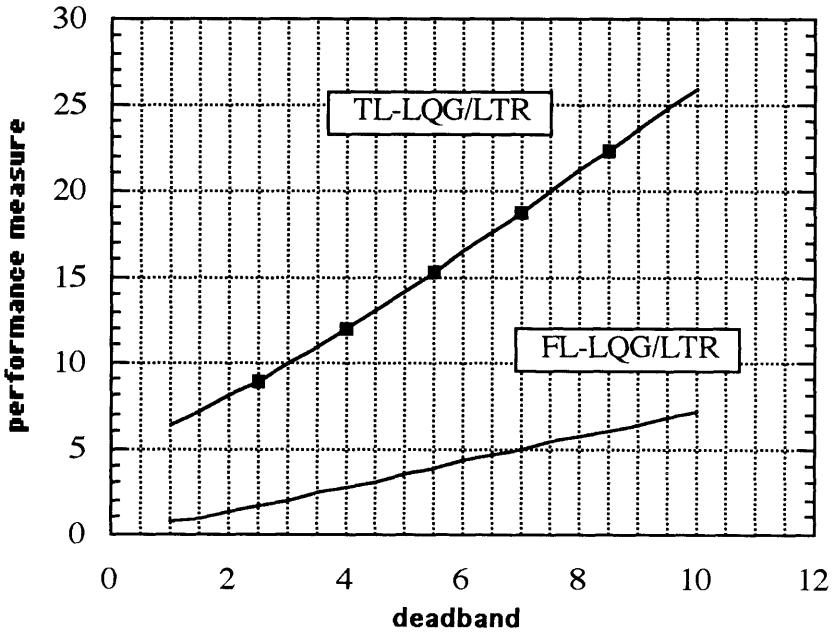

Figure 5.6.2: Performance dependence on deadband.

5.6.2. Parameter Error

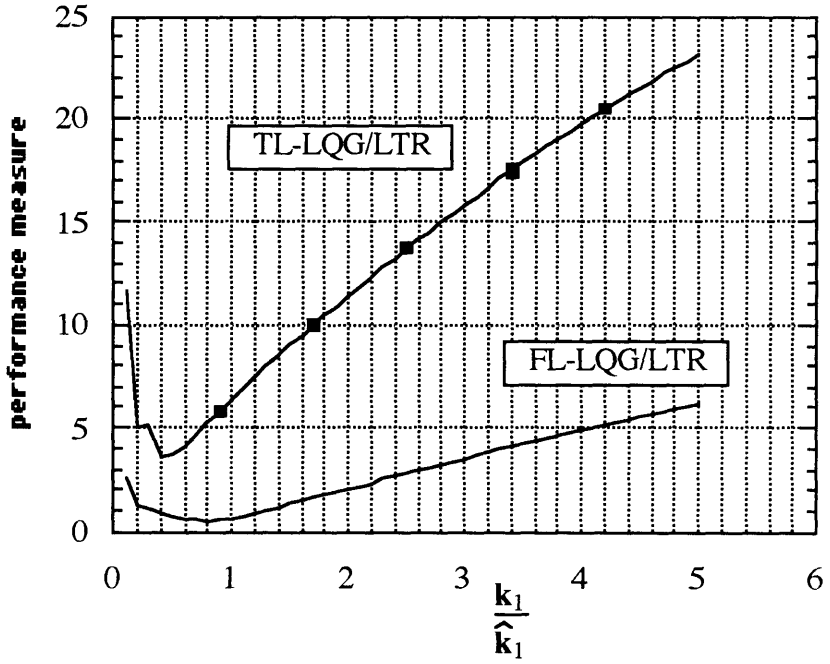

Figure 5.6.3: Performance as a function of parameter error in k1.

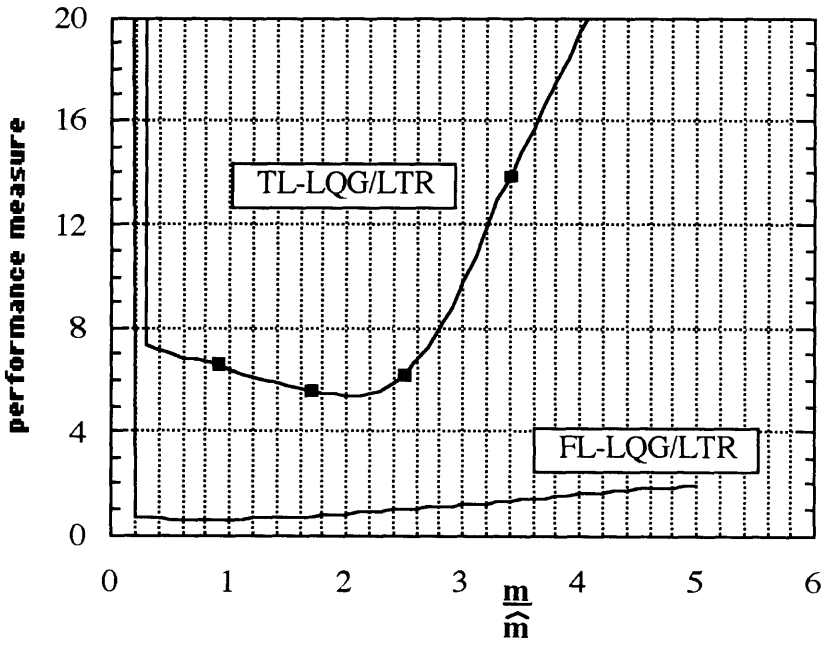

Figure 5.6.4: Performance as a function of parameter error in m.

5.6.3. Measurement Noise

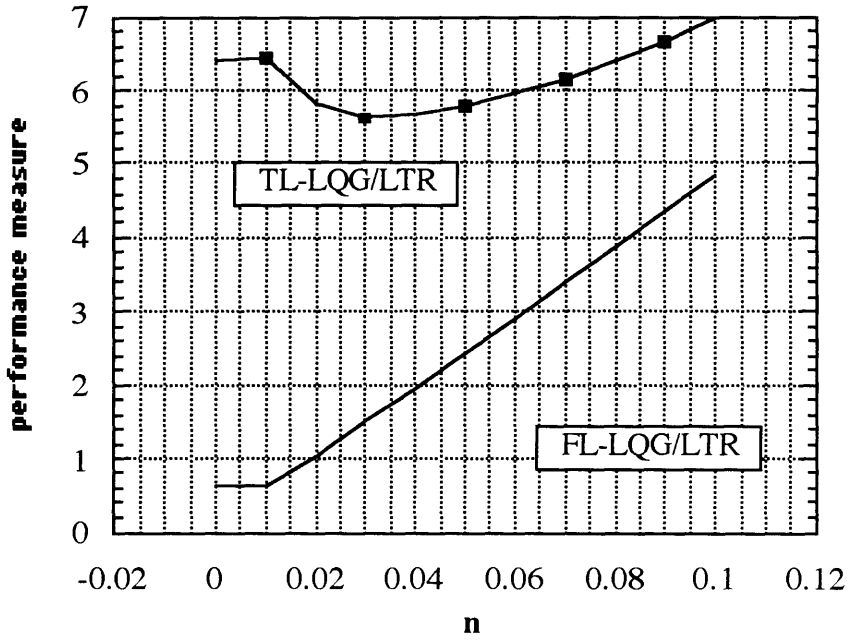

Figure 5.6.5: Performance dependence on measurement noise.

5.6.4.

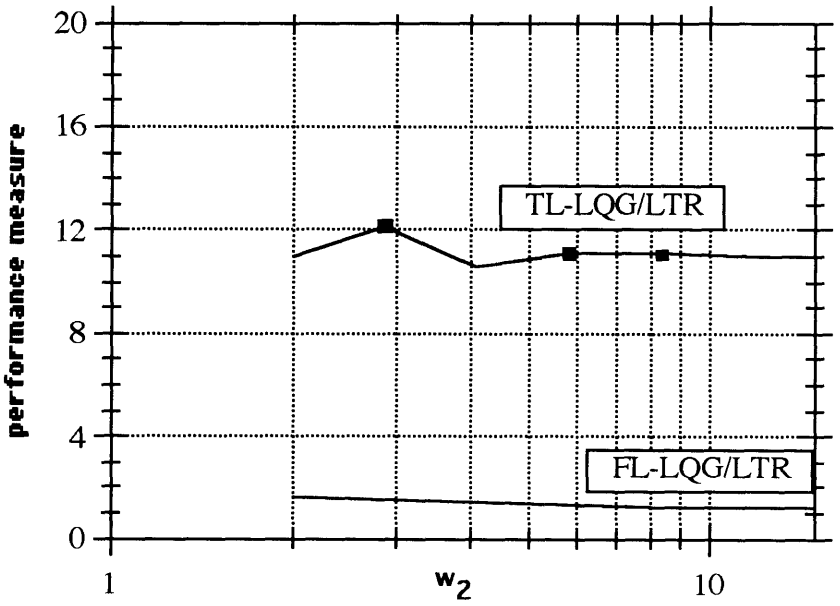

Figure 5.6.6: Performance as a function of frequency for high order dynamics.

5.6.5.

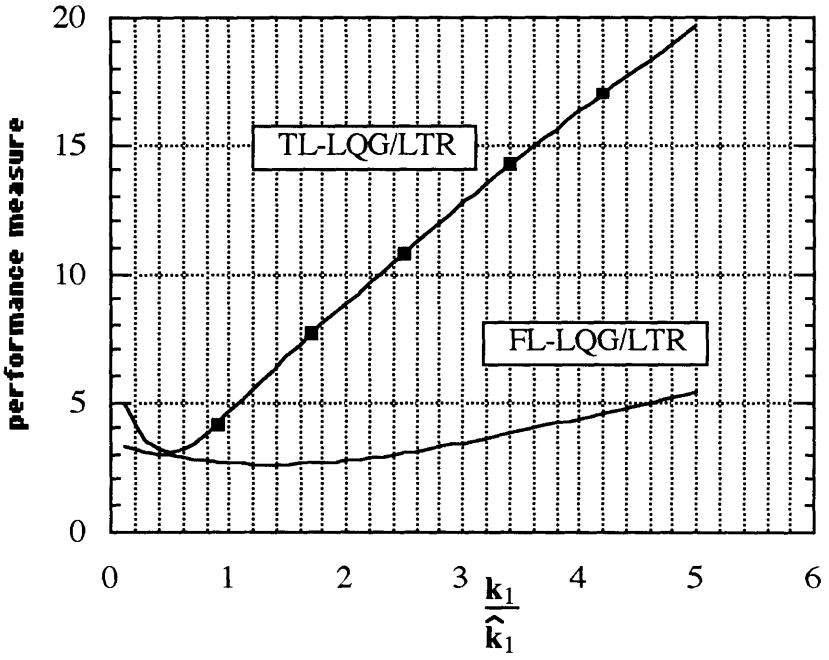

Figure 5.6.7: Performance as a function of parameter error in k1 with measurement noise and unmodeled dynamics at w=5 [rad/s].

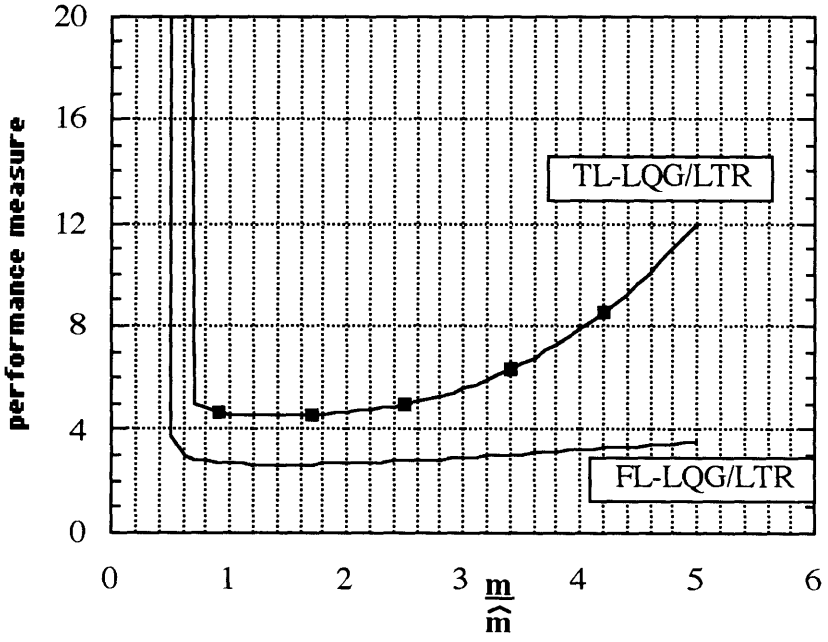

Figure 5.6.8: Performance as a function of parameter error in m with measurement noise and unmodeled dynamics at w=5 [rad/s].

First the effect of the deadband limit is explored (figure 5.6.1). The FL-controller shows less sensitivity to deadband in the control channel than the TL-controller.

Then the deadband limit is set to 1 and the simulations from 5.1 are repeated (figures 5.6.3 - 5.6.8). For parameter errors only (figures 5.5.3 - 5.5.4) the FL-controller is more robust. This is especially true for parameter errors in the mass (figure 5.6.4). With respect to measurement noise (figure 5.6.5) the controllers seem to perform about the same. The frequency at which the unmodeled dynamics occur seems to have little influence on the performance (figure 5.6.6). With measurement noise (n = 0.05) and unmodeled dynamics ($k_2 = 10$) present (figures 5.6.7 - 5.6.8), the picture has changed somewhat from figures 5.6.3 and 5.6.4, and the controllers seem to have about the same performance except at low actual values of k_1 when the TL-controlled system is unstable.

5.7. One Link Robot

Figures 5.7.1 - 5.7.6 present the Monte Carlo simulation results for the one link robot using the controllers designed in 5.7. The controllers use the same measurement information i.e. only position measurement is used for both the FL- and TL-controller.

5.7.1. Parameter Error

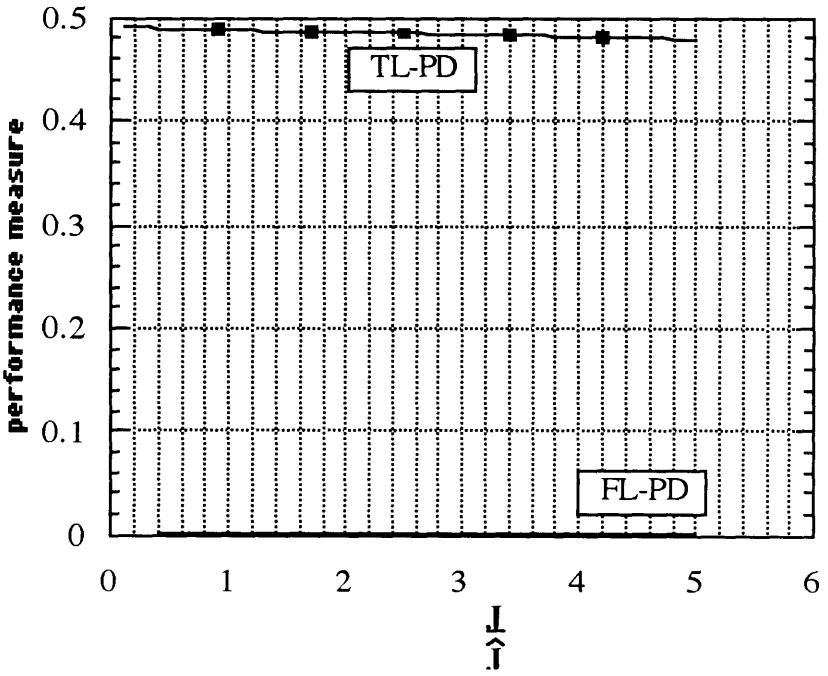

Figure 5.7.1: Performance as a function of parameter error in J.

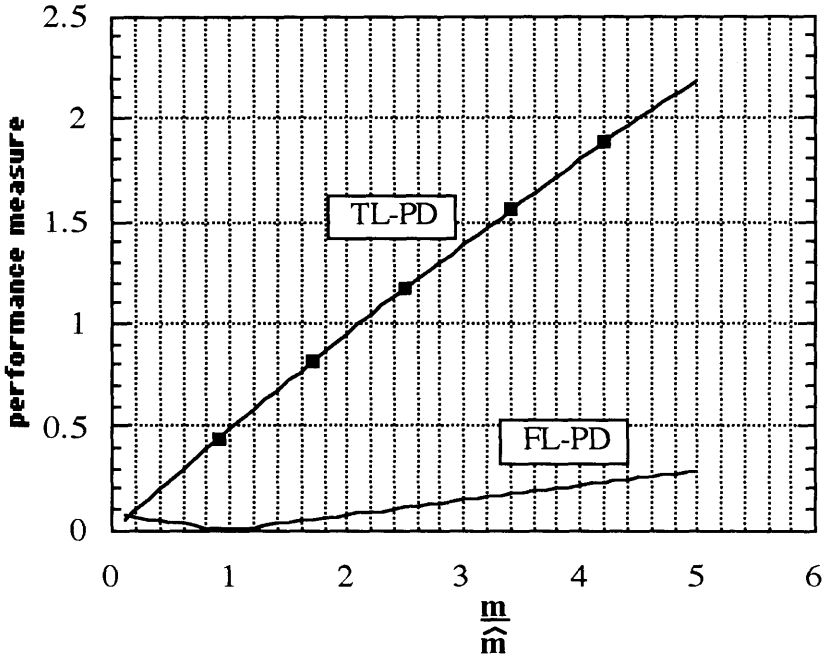

Figure 5.7.2: Performance as a function of parameter error in m.

5.7.2. Measurement Noise

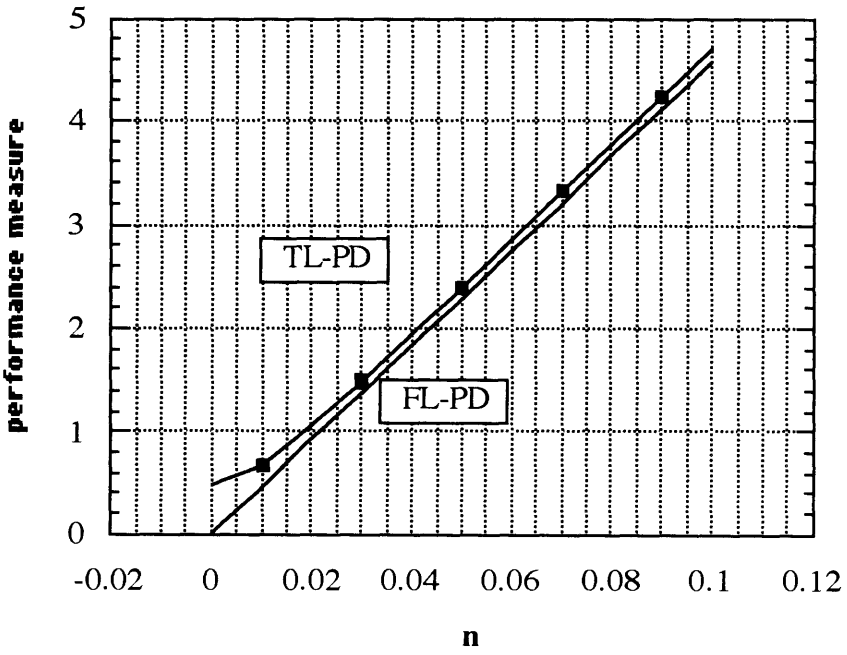

Figure 5.7.3: Performance dependence on measurement noise.

5.7.3. Unmodeled Dynamics

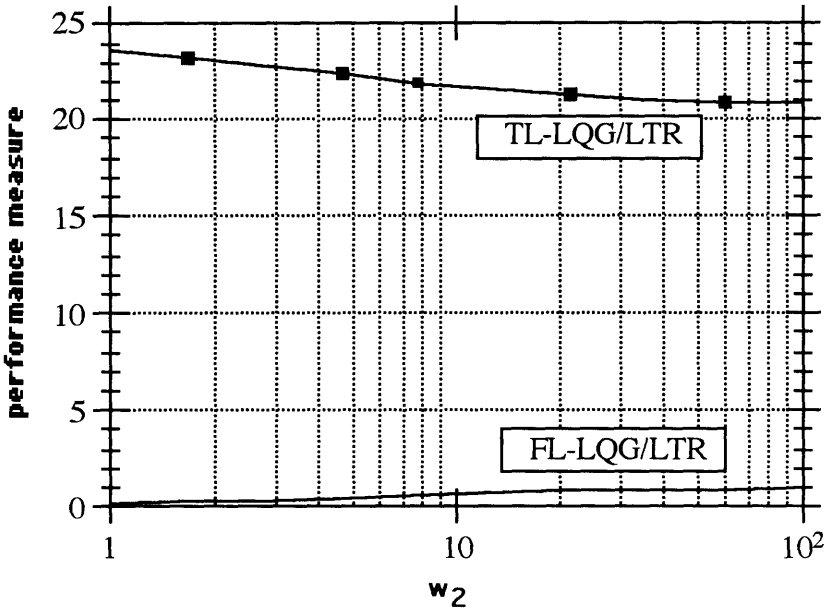

Figure 5.7.4: Performance as a function of frequency for unmodeled dynamics.

5.7.4. Parameter Error with Measurement Noise and Unmodeled Dynamics

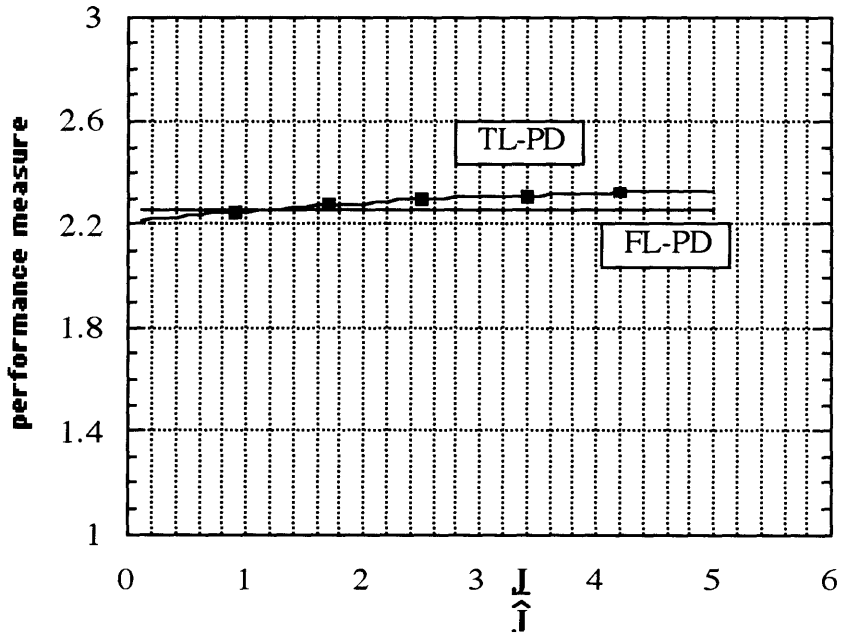

Figure 5.7.5: Performance as a function of parameter error in J with measurement noise and unmodeled dynamics ($k_1 = 10$).

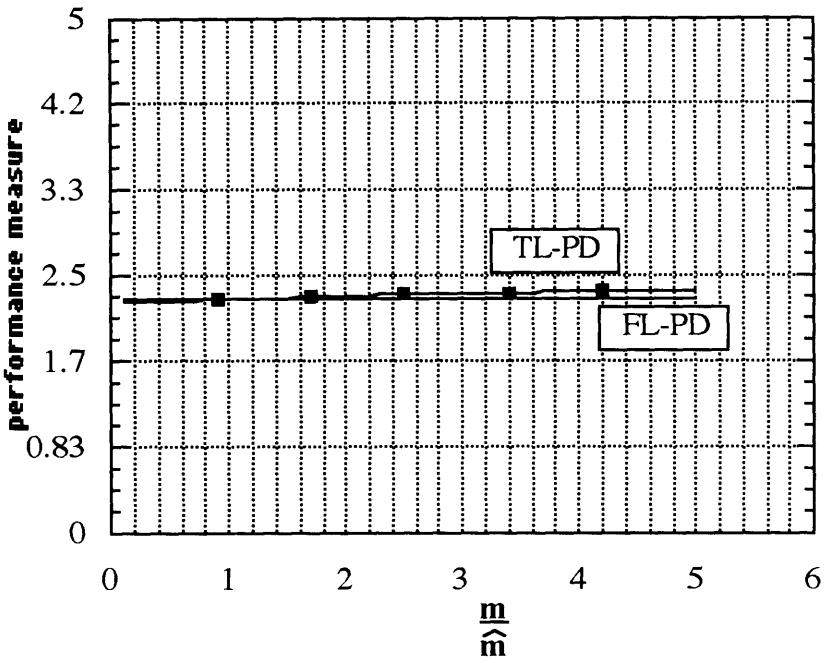

Figure 5.7.6: Performance as a function of parameter error in m with measurement noise and unmodeled dynamics ($k_1 = 10$).

For parameter errors only (figures 5.7.1 - 5.7.2) the picture is mixed. For parameter error in J the controllers show about the same robustness performance. For parameter error in m the TL-controller performs better. With respect to measurement noise (figure 5.7.3) the performance is about the same. The frequency at which the unmodeled dynamics occur seems to have little influence on the performance for both controllers (figure 5.7.4). With measurement noise ($n = 0.05$) and unmodeled dynamics ($k_2 = 10$) present (figures 5.7.5 - 5.7.6) both controllers have about the same robustness performance. The introduction of unmodeled dynamics and measurement noise has affected the FL-controller more than the TL-controller with respect to parameter error in m.

6. Conclusions

This thesis has presented a study of the robustness of the feedback linearization method based on Monte-Carlo simulations. The feedback linearization method has been compared to a more classical approach based on Taylor linearization.

Robustness with respect to both structured- and unstructured model uncertainties has been investigated for a variety of well known second order nonlinear systems. Structured model uncertainties have been simulated by introducing a mismatch between the actual parameters,used in the simulated plant, and the nominal parameters used in the controllers. The actual parameter has been varied through a wide selection of values in a range of 20% - 400% of the nominal value. For each value the performance of the controllers has been measured with a given performance measure. Unstructured model uncertainties have been introduced by adding high order low damped dynamics to the simulated plant. The frequency of the high order unmodeled dynamics has been varied through a range of frequencies and at each point the performance has been measured. Furthermore, the effect of varying intensity of measurement noise is studied as well.

The feedback linearization method gives relatively better robustness to parameter errors as well as high order dynamics. It looks like including the nonlinear information about the model into the control design process gives better robustness to parameter errors without destroying robustness to measurement noise. (No absolute comparison can be done, but the fact that the slopes of the performance graphs are less steep for the controllers based on the feedback linearization indicates better robustness). The robustness to measurement

noise turns out to be about the same. It is especially interesting that the same trend holds in the example when a saturation element is introduced in the control channel.

Bibliography

[1] Valavani L., MIT Course 16.372 Lecture Notes 1991

[2] Slotine J.J.E., Li W.. Applied Nonlinear Control. Prentice Hall 1991

[3] Sastry S.S., Kokotovic P.V., "Feedback Linearization in the Presence of Uncertainties", Center for Intelligent Control Systems 1988.

[4] Akhrif O., Blankenship G.L., "Robust Stabilization of Feedback Linearizable Systems", Proceedings of the 27th Conference on Decision and Control 1988.

[5] Balestrino A., De Maria G., Zinober A.S.I., "Nonlinear Adaptive Model-Following Control", Automatica 1985, 5, p. 559.

[6] Gutman S., Leitmann G., "Stabilizing Control for Linear Systems with Bounded Parameter and Input Uncertainty", Proceedings of the 7th IFIP Conference on Optimization Techniques, Springer Verlag 1976, p. 729.

[7] Narendra K.S., Kudva P., "Stable Adaptive Schemes for System Identification and Control", IEEE Trans. Syst. Man. Cyber., 1974, 4, no. 6, p. 542.

[8] Spong M.W., "Robust Stabilization for a Class of Nonlinear Systems", Theory and Application of Nonlinear Control Systems, 1986, North Holland.

[9] Cunningham W.J., Introduction to Nonlinear Analysis, McGraw-Hill Electrical and Electronic Engineering Series 1958.

[10] Athans M., "A Tutorial on the LQG/LTR Method", Proc. American Control Conference, Seattle, Wa, June 1986.

[11] Stein G., "A Critique of the LQG/LTR Design Method", Proc. American Control Conference Seattle, Wa, June 1986.

[12] The MathWorks, Inc., SIMULAB User's Guide 1991

Appendix A. Program Listing

A.1. Skeleton of Monte-Carlo Simulation.

A.1.1. Batch

```
%This is the main M-file for doing Monte-Carlo
%simulations. Each of the subprograms which also
%are M-files are called sequentially.

%Parameter error
clear;
mcpe;

%Measurement noise
clear;
mcei;

%Parameter with unmodeled dynamics and
%measurement noise
clear;
mcto;

%Unmodeled dynamics
clear;
mcud;
```

A.1.2. Mcpe

```
%Initialize system parameters
springdat;

%No high order dynamics and
%no measurement noise
k2=0;
d=0.0;

%Initialize count variables
k=1;
tfinal=20;

%Initialize simulation parameters
tol=1e-5;
minstep=0.01;
maxstep=0.01;
options=[tol,minstep,maxstep];

%Start simulation
for i = -90:10:400
        k1=(1+i/100)*1;
        x0=[0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springto',tfinal,x0,options);
        mctek1(k,:)=min(norm(y,2),100);
        x0=[0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springtolqgltr',tfinal,x0,options);
```

```
        mctek1lqg(k,:)=min(norm(y,2),100);
        k1=1;

        m=(1+i/100)*1;
        x0=[0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springto',tfinal,x0,options);
        mctem(k,:)=min(norm(y,2),100);
        x0=[0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springtolqgltr',tfinal,x0,options);
        mctemlqg(k,:)=min(norm(y,2),100);
        m=1;

        a=(1+i/100)*0.2;
        x0=[0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springto',tfinal,x0,options);
        mctea(k,:)=min(norm(y,2),100);
        x0=[0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springtolqgltr',tfinal,x0,options);
        mctealqg(k,:)=min(norm(y,2),100);
        a=0.2;

        k=k+1;
end

%Save results
xval=(0.1:0.1:5);
mcpedat = [xval' mctek1 mctek1lqg mctem mctemlqg mctea mctealqg];
```

```
save mcpe.res mcpedat;
save mcpe.txt mcpedat /ascii;
```

A.1.3. Mcei

```
%Initialize system parameters
springdat;

%No high order dynamics
k2=0;

%Initialize count variables
k=1;
tfinal=20;
npts=11;

%Noise scale factor
dvalues=(0:0.01:0.1);

%Initialize simulation parameters
tol=1e-5;
minstep=0.01;
maxstep=0.01;
options=[tol,minstep,maxstep];

%Start simulation
for i = 1:npts
```

```
        d=dvalues(i);
        x0=[0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springto',tfinal,x0,options);
        clear t x
        mcteei(k,:)=min(norm(y,2),100);
        clear y

        x0=[0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springtolqgltr',tfinal,x0,options);
        clear t x
        mcteeilqg(k,:)=min(norm(y,2),100);
        clear y
        k=k+1;
end

%Save results
mceidat = [dvalues' mcteei mcteeilqg ];
save mcei.res mceidat;
save mcei.txt mceidat /ascii;
```

A.1.4. Mcto

```
%Initialize system parameters
springdat;

%Initialize count variables
k=1;
```

```
tfinal=20;
ulimit = 100;

%Initialize simulation parameters
tol=1e-5;
minstep=0.01;
maxstep=0.01;
options=[tol,minstep,maxstep];

%Start simulation
for i = -90:10:400
        k1=(1+i/100)*1;
        x0=[0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springto',tfinal,x0,options);
        clear t x
        tek1(k,:)=min(norm(y,2),ulimit);
        clear y;

        x0=[0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springtopp',tfinal,x0,options);
        clear t x
        tek1pp(k,:)=min(norm(y,2),ulimit);
        clear y;

        x0=[0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springtolqgltr',tfinal,x0,options);
        clear t x
```

```
tek1lqg(k,:)=min(norm(y,2),ulimit);
clear y;
k1=1;

m1=(1+i/100)*0.5;
m2=m1;
x0=[0.0;0.0;0.0;0.0;0.0;0.0];
[t,x,y] = linsim('springto',tfinal,x0,options);
clear t x
tem(k,:)=min(norm(y,2),ulimit);
clear y;

x0=[0.0;0.0;0.0;0.0];
[t,x,y] = linsim('springtopp',tfinal,x0,options);
clear t x
tempp(k,:)=min(norm(y,2),ulimit);
clear y;

x0=[0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0];
[t,x,y] = linsim('springtolqgltr',tfinal,x0,options);
clear t x
temlqg(k,:)=min(norm(y,2),ulimit);
clear y;
m1=0.5;
m2=m1;

a=(1+i/100)*0.2;
```

```
        x0=[0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springto',tfinal,x0,options);
        clear t x
        tea(k,:)=min(norm(y,2),ulimit);
        clear y;

        x0=[0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springtopp',tfinal,x0,options);
        clear t x
        teapp(k,:)=min(norm(y,2),ulimit);
        clear y;

        x0=[0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springtolqgltr',tfinal,x0,options);
        clear t x
        tealqg(k,:)=min(norm(y,2),ulimit);
        clear y;
        a=0.2;

        k=k+1;
end

%Save results
xval=(0.1:0.1:5);
mctodat = [xval' tek1 tek1pp tek1lqg tem tempp temlqg tea teapp tealqg];
save mcto.res mctodat;
save mcto.txt mctodat /ascii;
```

A.1.5. Mcud

```
%Initialize system parameters
springdat;

%No measurement noise
d=0.0;

%Initialize count variables
k=1;
tfinal=20;
npts=10;

%Frequencies for unmodeled dynamics
omg2values=logspace(0.3,1.7,npts);
k2values=0.5*omg2values.^2;

%Initialize simulation parameters
tol=1e-5;
minstep=0.0025;
maxstep=0.0025;
options=[tol,minstep,maxstep];

%Start simulation
for i = 1:npts
        k2=k2values(i);
        x0=[0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springto',tfinal,x0,options);
```

```
        clear t x
        mcteud(k,:)=min(norm(y,2),100);
        clear y

        x0=[0.0;0.0;0.0;0.0;0.0;0.0;0.0;0.0];
        [t,x,y] = linsim('springtolqgltr',tfinal,x0,options);
        clear t x
        mcteudlqg(k,:)=min(norm(y,2),100);
        clear y

        k=k+1;
end

%Save results
mcuddat = [omg2values' mcteud mcteudlqg ];
save mcud.res mcuddat;
save mcud.txt mcuddat /ascii;
```

A.2. System Simulation Programs

A.2.1. Duffing's Equation / Mass on a Nonlinear Spring

```
function [ret,x0,str]=springto(t,x,u,flag);
%SPRINGTO      is the M-file description of the SIMULAB system
named SPRINGTO.
%       The block-diagram can be displayed by typing: SPRINGTO.
%
%       SYS=SPRINGTO(T,X,U,FLAG) returns depending on FLAG certain
%       system values given time point, T, current state vector, X,
%       and input vector, U.
```

```
%       FLAG is used to indicate the type of output to be returned in
SYS.
%
%       Setting FLAG=1 causes SPRINGTO to return state derivitives,
FLAG=2
%       discrete states, FLAG=3 system outputs and FLAG=4 next
sample
%       time. For more information and other options see SFUNC.
%
%       Calling SPRINGTO with a FLAG of zero:
%       [SIZES]=SPRINGTO([],[],[],0),  returns a vector, SIZES, which
%       contains the sizes of the state vector and other parameters.
%               SIZES(1) number of states
%               SIZES(2) number of discrete states
%               SIZES(3) number of outputs
%               SIZES(4) number of inputs.
%       For the definition of other parameters in SIZES, see SFUNC.
%       See also, TRIM, LINMOD, LINSIM, EULER, RK23, RK45, ADAMS,
GEAR.

% Note: This M-file is only used for saving graphical information;
%       after the model is loaded into memory an internal model
%       representation is used.

% the system will take on the name of this mfile:
sys = mfilename;
new_system(sys)
simulab_version(1.01)
if(0 == (nargin + nargout))
    set_param(sys,'Location',[6,51,499,371])
    open_system(sys)
end;
set_param(sys,'algorithm',          'Linear')
set_param(sys,'Start time',    '0.0')
set_param(sys,'Stop time',          '20')
set_param(sys,'Min step size',      '0.05')
set_param(sys,'Max step size',      '0.05')
set_param(sys,'Relative error','1e-5')
set_param(sys,'Return vars', '')


%    Subsystem  'measurement'.
```

```
new_system([sys,'/','measurement'])
set_param([sys,'/','measurement'],'Location',[3,42,509,381])

add_block('built-in/Inport',[sys,'/','measurement/In_2'])
set_param([sys,'/','measurement/In_2'],...
                'position',[25,95],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Demux',[sys,'/','measurement/Demux1'])
set_param([sys,'/','measurement/Demux1'],...
                'position',[70,87],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Outport',[sys,'/','measurement/Out_4'])
set_param([sys,'/','measurement/Out_4'],...
                'position',[285,20],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sum',[sys,'/','measurement/Sum'])
set_param([sys,'/','measurement/Sum'],...
                'position',[235,20],...
                'size',[20,20],...
                'inputs','+-')

add_block('built-in/Inport',[sys,'/','measurement/In_1'])
set_param([sys,'/','measurement/In_1'],...
                'position',[25,25],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Demux',[sys,'/','measurement/Demux'])
set_param([sys,'/','measurement/Demux'],...
                'position',[70,19],...
                'size',[40,32],...
                'outputs','3')
add_line([sys,'/','measurement'],[260,30;280,30])
add_line([sys,'/','measurement'],[50,35;65,35])
add_line([sys,'/','measurement'],[50,105;65,105])
add_line([sys,'/','measurement'],[115,95;185,95;185,35;230,35])
add_line([sys,'/','measurement'],[115,25;230,25])
```

```
%    Finished composite block 'measurement'.

set_param([sys,'/','measurement'],...
                'position',[370,62],...
                'size',[30,56])

add_block('built-in/Outport',[sys,'/','Outport'])
set_param([sys,'/','Outport'],...
                'position',[450,80],...
                'size',[20,20],...
                'Port','1')


%    Subsystem 'Controller'.

new_system([sys,'/','Controller'])
set_param([sys,'/','Controller'],'Location',[19,219,526,388])

add_block('built-in/State-space',[sys,'/','Controller/State-space'])
set_param([sys,'/','Controller/State-space'],...
                'position',[270,5],...
                'size',[60,30],...
                'A','Acfdb',...
                'B','-Hfdb',...
                'C','-Gfdb',...
                'D','0')

add_block('built-in/Inport',[sys,'/','Controller/In_2'])
set_param([sys,'/','Controller/In_2'],...
                'position',[25,80],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Demux',[sys,'/','Controller/Demux1'])
set_param([sys,'/','Controller/Demux1'],...
                'position',[90,72],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Demux',[sys,'/','Controller/Demux'])
set_param([sys,'/','Controller/Demux'],...
```

```
                'position',[90,9],...
                'size',[40,32],...
                'outputs','3')

add_block('built-in/Outport',[sys,'/','Controller/Out_1'])
set_param([sys,'/','Controller/Out_1'],...
                'position',[515,30],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Inport',[sys,'/','Controller/In_1'])
set_param([sys,'/','Controller/In_1'],...
                'position',[25,15],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sum',[sys,'/','Controller/Sum'])
set_param([sys,'/','Controller/Sum'],...
                'position',[200,10],...
                'size',[20,20],...
                'inputs','+-')

add_block('built-in/Note',[sys,'/','Controller/e'])
set_param([sys,'/','Controller/e'],...
                'position',[230,0],...
                'size',[1,1])

add_block('built-in/Sum',[sys,'/','Controller/Sum3'])
set_param([sys,'/','Controller/Sum3'],...
                'position',[450,30],...
                'size',[20,20],...
                'inputs','++')

add_block('built-in/Sum',[sys,'/','Controller/Sum4'])
set_param([sys,'/','Controller/Sum4'],...
                'position',[375,75],...
                'size',[20,20],...
                'inputs','++')

add_block('built-in/Gain',[sys,'/','Controller/Gain'])
set_param([sys,'/','Controller/Gain'],...
                'position',[230,70],...
                'size',[20,20],...
```

```
                'Gain','k1h/mh')

add_block('built-in/Fcn',[sys,'/','Controller/Fcn'])
set_param([sys,'/','Controller/Fcn'],...
                'position',[220,110],...
                'size',[40,20],...
                'Expr','ah*u[1]*u[1]*u[1]')

add_block('built-in/Gain',[sys,'/','Controller/Gain3'])
set_param([sys,'/','Controller/Gain3'],...
                'position',[305,110],...
                'size',[20,20],...
                'Gain','k1h/mh')
add_line([sys,'/','Controller'],[335,20;345,20;345,35;445,35])
add_line([sys,'/','Controller'],[50,90;85,90])
add_line([sys,'/','Controller'],[50,25;85,25])
add_line([sys,'/','Controller'],[135,15;195,15])
add_line([sys,'/','Controller'],[135,80;180,80;180,25;195,25])
add_line([sys,'/','Controller'],[225,20;265,20])
add_line([sys,'/','Controller'],[475,40;510,40])
add_line([sys,'/','Controller'],[400,85;420,85;420,45;445,45])
add_line([sys,'/','Controller'],[180,80;225,80])
add_line([sys,'/','Controller'],[255,80;370,80])
add_line([sys,'/','Controller'],[180,80;180,120;215,120])
add_line([sys,'/','Controller'],[265,120;300,120])
add_line([sys,'/','Controller'],[330,120;340,120;340,90;370,90])


%    Finished composite block 'Controller'.

set_param([sys,'/','Controller'],...
                'position',[165,62],...
                'size',[30,56])

add_block('built-in/White Noise',[sys,'/','White Noise'])
set_param([sys,'/','White Noise'],...
                'position',[75,140],...
                'size',[20,20],...
                'Seed','0')


%    Subsystem  'Reference'.
```

```
new_system([sys,'/','Reference'])
set_param([sys,'/','Reference'],'Location',[0,176,396,362])

add_block('built-in/Mux',[sys,'/','Reference/Mux1'])
set_param([sys,'/','Reference/Mux1'],...
                'position',[275,27],...
                'size',[30,36],...
                'inputs','3')

add_block('built-in/Outport',[sys,'/','Reference/Out_1'])
set_param([sys,'/','Reference/Out_1'],...
                'position',[330,35],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave'])
set_param([sys,'/','Reference/Sine Wave'],...
                'position',[100,25],...
                'size',[20,20],...
                'amplitude','ramp',...
                'frequency','rfreq',...
                'phase','0')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave1'])
set_param([sys,'/','Reference/Sine Wave1'],...
                'position',[100,75],...
                'size',[20,20],...
                'amplitude','ramp*rfreq',...
                'frequency','rfreq',...
                'phase','3.1415926/2')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave2'])
set_param([sys,'/','Reference/Sine Wave2'],...
                'position',[100,125],...
                'size',[20,20],...
                'amplitude','-ramp*rfreq*rfreq',...
                'frequency','rfreq',...
                'phase','0')
add_line([sys,'/','Reference'],[125,35;270,35])
add_line([sys,'/','Reference'],[310,45;325,45])
add_line([sys,'/','Reference'],[125,85;150,85;150,45;270,45])
add_line([sys,'/','Reference'],[125,135;245,135;245,55;270,55])
```

```
%    Finished composite block 'Reference'.

set_param([sys,'/','Reference'],...
                'position',[35,50],...
                'size',[30,50])


%    Subsystem  'springdyn'.

new_system([sys,'/','springdyn'])
set_param([sys,'/','springdyn'],'Location',[3,43,503,345])

add_block('built-in/Gain',[sys,'/','springdyn/Gain5'])
set_param([sys,'/','springdyn/Gain5'],...
                'position',[115,55],...
                'size',[20,20],...
                'Gain','1/m1')

add_block('built-in/Gain',[sys,'/','springdyn/Gain4'])
set_param([sys,'/','springdyn/Gain4'],...
                'position',[375,15],...
                'size',[20,20],...
                'Gain','d')

add_block('built-in/Sum',[sys,'/','springdyn/Sum3'])
set_param([sys,'/','springdyn/Sum3'],...
                'position',[435,57],...
                'size',[20,36],...
                'inputs','++')

add_block('built-in/Gain',[sys,'/','springdyn/Gain3'])
set_param([sys,'/','springdyn/Gain3'],...
                'position',[205,215],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k1/m1')

add_block('built-in/Gain',[sys,'/','springdyn/Gain2'])
set_param([sys,'/','springdyn/Gain2'],...
                'position',[100,190],...
                'size',[20,20],...
                'orientation',3,...
```

```
                'Gain','k2/m1')

add_block('built-in/Note',[sys,'/','springdyn/x1-x3'])
set_param([sys,'/','springdyn/x1-x3'],...
                'position',[445,250],...
                'size',[1,1])

add_block('built-in/Gain',[sys,'/','springdyn/Gain1'])
set_param([sys,'/','springdyn/Gain1'],...
                'position',[230,350],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k2/m2')

add_block('built-in/Sum',[sys,'/','springdyn/Sum2'])
set_param([sys,'/','springdyn/Sum2'],...
                'position',[415,282],...
                'size',[20,36],...
                'inputs','+-')

add_block('built-in/Note',[sys,'/','springdyn/x3'])
set_param([sys,'/','springdyn/x3'],...
                'position',[375,290],...
                'size',[1,1])

add_block('built-in/Note',[sys,'/','springdyn/x4'])
set_param([sys,'/','springdyn/x4'],...
                'position',[270,290],...
                'size',[1,1])

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator3'])
set_param([sys,'/','springdyn/Integrator3'],...
                'position',[345,300],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator2'])
set_param([sys,'/','springdyn/Integrator2'],...
                'position',[230,300],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Mux',[sys,'/','springdyn/Mux'])
```

```
set_param([sys,'/','springdyn/Mux'],...
                'position',[485,72],...
                'size',[30,36],...
                'inputs','2')

add_block('built-in/Outport',[sys,'/','springdyn/Outport'])
set_param([sys,'/','springdyn/Outport'],...
                'position',[545,80],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Inport',[sys,'/','springdyn/Inport'])
set_param([sys,'/','springdyn/Inport'],...
                'position',[25,55],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator'])
set_param([sys,'/','springdyn/Integrator'],...
                'position',[330,100],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Gain',[sys,'/','springdyn/Gain'])
set_param([sys,'/','springdyn/Gain'],...
                'position',[250,180],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k1/m1')

add_block('built-in/Sum',[sys,'/','springdyn/Sum'])
set_param([sys,'/','springdyn/Sum'],...
                'position',[195,44],...
                'size',[20,147],...
                'inputs','+---')

add_block('built-in/Note',[sys,'/','springdyn/x1'])
set_param([sys,'/','springdyn/x1'],...
                'position',[365,86],...
                'size',[1,1])

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator1'])
set_param([sys,'/','springdyn/Integrator1'],...
```

```
                'position',[250,100],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Note',[sys,'/','springdyn/x2'])
set_param([sys,'/','springdyn/x2'],...
                'position',[280,86],...
                'size',[1,1])

add_block('built-in/Fcn',[sys,'/','springdyn/Fcn'])
set_param([sys,'/','springdyn/Fcn'],...
                'position',[265,215],...
                'size',[40,20],...
                'orientation',2,...
                'Expr','a*u[1]*u[1]*u[1]')

add_block('built-in/Inport',[sys,'/','springdyn/Inport1'])
set_param([sys,'/','springdyn/Inport1'],...
                'position',[290,15],...
                'size',[20,20],...
                'Port','2')
add_line([sys,'/','springdyn'],[390,225;310,225])
add_line([sys,'/','springdyn'],[370,110;370,190;275,190])
add_line([sys,'/','springdyn'],[390,110;390,290;410,290])
add_line([sys,'/','springdyn'],[355,110;390,110;390,85;430,85])
add_line([sys,'/','springdyn'],[140,65;190,65])
add_line([sys,'/','springdyn'],[50,65;110,65])
add_line([sys,'/','springdyn'],[400,25;410,25;410,65;430,65])
add_line([sys,'/','springdyn'],[315,25;370,25])
add_line([sys,'/','springdyn'],[460,75;470,75;470,80;480,80])
add_line([sys,'/','springdyn'],[225,360;130,360;130,310;225,310])
add_line([sys,'/','springdyn'],[245,190;170,190;170,170;190,170])
add_line([sys,'/','springdyn'],[200,225;150,225;150,135;190,135])
add_line([sys,'/','springdyn'],[110,185;110,100;190,100])
add_line([sys,'/','springdyn'],[290,110;290,150;425,150;425,100;480,1
00])
add_line([sys,'/','springdyn'],[260,225;230,225])
add_line([sys,'/','springdyn'],[450,300;460,300;460,265;110,265;110,2
15])
add_line([sys,'/','springdyn'],[440,300;450,300;450,360;255,360])
add_line([sys,'/','springdyn'],[370,310;410,310])
add_line([sys,'/','springdyn'],[255,310;340,310])
add_line([sys,'/','springdyn'],[520,90;540,90])
```

```
add_line([sys,'/','springdyn'],[275,110;325,110])
add_line([sys,'/','springdyn'],[220,120;220,110;245,110])


%    Finished composite block 'springdyn'.

set_param([sys,'/','springdyn'],...
                'position',[245,77],...
                'size',[30,51])
add_line(sys,[405,90;445,90])
add_line(sys,[280,105;365,105])
add_line(sys,[280,105;300,105;300,165;130,165;130,105;160,105])
add_line(sys,[70,75;105,75;105,45;335,45;335,75;365,75])
add_line(sys,[70,75;160,75])
add_line(sys,[200,90;240,90])
add_line(sys,[100,150;215,150;215,115;240,115])
% Return any arguments.
if (nargin | nargout)
        % Must use feval here to access system in memory
        if (nargin > 3)
                if (flag == 0)
                        eval(['[ret,x0,xstr]=',sys,'(t,x,u,flag);'])
                else
                        eval(['ret =', sys,'(t,x,u,flag);'])
                end
        else
                [ret,x0,str] = feval(sys);
        end
end
```

A.2.2. Mathieu Equation

```
function [ret,x0,str]=mathieuto(t,x,u,flag);
%MATHIEUTO     is the M-file description of the SIMULAB system
named MATHIEUTO.
%      The block-diagram can be displayed by typing: MATHIEUTO.
%
%      SYS=MATHIEUTO(T,X,U,FLAG) returns depending on FLAG
certain
%      system values given time point, T, current state vector, X,
%      and input vector, U.
%      FLAG is used to indicate the type of output to be returned in
SYS.
%
%      Setting FLAG=1 causes MATHIEUTO to return state derivitives,
FLAG=2
%      discrete states, FLAG=3 system outputs and FLAG=4 next
sample
%      time. For more information and other options see SFUNC.
%
%      Calling MATHIEUTO with a FLAG of zero:
%      [SIZES]=MATHIEUTO([],[],[],0),  returns a vector, SIZES, which
%      contains the sizes of the state vector and other parameters.
%            SIZES(1) number of states
%            SIZES(2) number of discrete states
%            SIZES(3) number of outputs
%            SIZES(4) number of inputs.
%      For the definition of other parameters in SIZES, see SFUNC.
%      See also, TRIM, LINMOD, LINSIM, EULER, RK23, RK45, ADAMS,
GEAR.

% Note: This M-file is only used for saving graphical information;
%       after the model is loaded into memory an internal model
%       representation is used.

% the system will take on the name of this mfile:
sys = mfilename;
new_system(sys)
simulab_version(1.01)
if(0 == (nargin + nargout))
   set_param(sys,'Location',[19,107,529,355])
```

```
    open_system(sys)
end;
set_param(sys,'algorithm',          'Linear')
set_param(sys,'Start time',   '0.0')
set_param(sys,'Stop time',          '20')
set_param(sys,'Min step size',      '0.05')
set_param(sys,'Max step size',      '0.05')
set_param(sys,'Relative error','1e-5')
set_param(sys,'Return vars', '')


%    Subsystem  ['mathieu',13,'equation'].

new_system([sys,'/',['mathieu',13,'equation']])
set_param([sys,'/',['mathieu',13,'equation']],'Location',[3,42,509,381])

add_block('built-in/Note',[sys,'/',['mathieu',13,'equation/x1-x3']])
set_param([sys,'/',['mathieu',13,'equation/x1-x3']],...
                'position',[415,275],...
                'size',[1,1])

add_block('built-in/Gain',[sys,'/',['mathieu',13,'equation/Gain3']])
set_param([sys,'/',['mathieu',13,'equation/Gain3']],...
                'position',[245,280],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k2')

add_block('built-in/Gain',[sys,'/',['mathieu',13,'equation/Gain2']])
set_param([sys,'/',['mathieu',13,'equation/Gain2']],...
                'position',[220,420],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k2')

add_block('built-in/Sum',[sys,'/',['mathieu',13,'equation/Sum2']])
set_param([sys,'/',['mathieu',13,'equation/Sum2']],...
                'position',[395,332],...
                'size',[20,36],...
                'inputs','+-')

add_block('built-in/Note',[sys,'/',['mathieu',13,'equation/x3']])
set_param([sys,'/',['mathieu',13,'equation/x3']],...
```

```
                'position',[355,340],...
                'size',[1,1])

add_block('built-in/Note',[sys,'/',['mathieu',13,'equation/x4']])
set_param([sys,'/',['mathieu',13,'equation/x4']],...
                'position',[250,340],...
                'size',[1,1])

add_block('built-
in/Integrator',[sys,'/',['mathieu',13,'equation/Integrator3']])
set_param([sys,'/',['mathieu',13,'equation/Integrator3']],...
                'position',[325,350],...
                'size',[20,20],...
                'Initial','0')

add_block('built-
in/Integrator',[sys,'/',['mathieu',13,'equation/Integrator2']])
set_param([sys,'/',['mathieu',13,'equation/Integrator2']],...
                'position',[210,350],...
                'size',[20,20],...
                'Initial','0')

add_block('built-
in/Product',[sys,'/',['mathieu',13,'equation/Product']])
set_param([sys,'/',['mathieu',13,'equation/Product']],...
                'position',[315,215],...
                'size',[25,20],...
                'orientation',2,...
                'inputs','2')

add_block('built-in/Clock',[sys,'/',['mathieu',13,'equation/Clock']])
set_param([sys,'/',['mathieu',13,'equation/Clock']],...
                'position',[500,220],...
                'size',[20,20],...
                'orientation',2)

add_block('built-in/Mux',[sys,'/',['mathieu',13,'equation/Mux']])
set_param([sys,'/',['mathieu',13,'equation/Mux']],...
                'position',[485,117],...
                'size',[30,36],...
                'inputs','2')

add_block('built-in/Outport',[sys,'/',['mathieu',13,'equation/Outport']])
```

```
set_param([sys,'/',['mathieu',13,'equation/Outport']],...
                'position',[530,125],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Inport',[sys,'/',['mathieu',13,'equation/control']])
set_param([sys,'/',['mathieu',13,'equation/control']],...
                'position',[70,95],...
                'size',[20,20],...
                'Port','1')

add_block('built-
in/Integrator',[sys,'/',['mathieu',13,'equation/Integrator']])
set_param([sys,'/',['mathieu',13,'equation/Integrator']],...
                'position',[320,120],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Gain',[sys,'/',['mathieu',13,'equation/Gain']])
set_param([sys,'/',['mathieu',13,'equation/Gain']],...
                'position',[240,175],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','a')

add_block('built-in/Sum',[sys,'/',['mathieu',13,'equation/Sum']])
set_param([sys,'/',['mathieu',13,'equation/Sum']],...
                'position',[175,94],...
                'size',[20,67],...
                'inputs','+-+-')

add_block('built-in/Note',[sys,'/',['mathieu',13,'equation/x1']])
set_param([sys,'/',['mathieu',13,'equation/x1']],...
                'position',[355,106],...
                'size',[1,1])

add_block('built-
in/Integrator',[sys,'/',['mathieu',13,'equation/Integrator1']])
set_param([sys,'/',['mathieu',13,'equation/Integrator1']],...
                'position',[240,120],...
                'size',[20,20],...
                'Initial','0')
```

```
add_block('built-in/Note',[sys,'/',['mathieu',13,'equation/x2']])
set_param([sys,'/',['mathieu',13,'equation/x2']],...
                'position',[270,106],...
                'size',[1,1])

add_block('built-in/Fcn',[sys,'/',['mathieu',13,'equation/Fcn']])
set_param([sys,'/',['mathieu',13,'equation/Fcn']],...
                'position',[430,220],...
                'size',[40,20],...
                'orientation',2,...
                'Expr','cos(2*u[1])')

add_block('built-in/Gain',[sys,'/',['mathieu',13,'equation/Gain1']])
set_param([sys,'/',['mathieu',13,'equation/Gain1']],...
                'position',[240,215],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','2*q')

add_block('built-in/Sum',[sys,'/',['mathieu',13,'equation/Sum1']])
set_param([sys,'/',['mathieu',13,'equation/Sum1']],...
                'position',[420,115],...
                'size',[20,20],...
                'inputs','++')

add_block('built-in/Inport',[sys,'/',['mathieu',13,'equation/noise']])
set_param([sys,'/',['mathieu',13,'equation/noise']],...
                'position',[300,10],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Gain',[sys,'/',['mathieu',13,'equation/Gain4']])
set_param([sys,'/',['mathieu',13,'equation/Gain4']],...
                'position',[365,10],...
                'size',[20,20],...
                'Gain','d')
add_line([sys,'/',['mathieu',13,'equation']],[240,290;105,290;105,120;1
70,120])
add_line([sys,'/',['mathieu',13,'equation']],[235,225;135,225;135,135;1
70,135])
add_line([sys,'/',['mathieu',13,'equation']],[235,185;155,185;155,150;1
70,150])
add_line([sys,'/',['mathieu',13,'equation']],[440,350;440,290;270,290])
```

```
add_line([sys,'/',['mathieu',13,'equation']],[380,220;380,340;390,340])
add_line([sys,'/',['mathieu',13,'equation']],[215,430;115,430;115,360;2
05,360])
add_line([sys,'/',['mathieu',13,'equation']],[420,350;440,350;440,430;2
45,430])
add_line([sys,'/',['mathieu',13,'equation']],[350,360;390,360])
add_line([sys,'/',['mathieu',13,'equation']],[235,360;320,360])
add_line([sys,'/',['mathieu',13,'equation']],[425,230;345,230])
add_line([sys,'/',['mathieu',13,'equation']],[495,230;475,230])
add_line([sys,'/',['mathieu',13,'equation']],[380,130;380,220;345,220])
add_line([sys,'/',['mathieu',13,'equation']],[310,225;265,225])
add_line([sys,'/',['mathieu',13,'equation']],[95,105;170,105])
add_line([sys,'/',['mathieu',13,'equation']],[265,130;315,130])
add_line([sys,'/',['mathieu',13,'equation']],[200,130;235,130])
add_line([sys,'/',['mathieu',13,'equation']],[360,131;360,185;265,185])
add_line([sys,'/',['mathieu',13,'equation']],[520,135;525,135])
add_line([sys,'/',['mathieu',13,'equation']],[290,130;290,195;475,195;4
80,145])
add_line([sys,'/',['mathieu',13,'equation']],[345,130;415,130])
add_line([sys,'/',['mathieu',13,'equation']],[325,20;360,20])
add_line([sys,'/',['mathieu',13,'equation']],[390,20;400,20;400,120;415,
120])
add_line([sys,'/',['mathieu',13,'equation']],[445,125;480,125])


%    Finished composite block ['mathieu',13,'equation'].

set_param([sys,'/',['mathieu',13,'equation']],...
                'position',[250,76],...
                'size',[30,53])


%    Subsystem  'Reference'.

new_system([sys,'/','Reference'])
set_param([sys,'/','Reference'],'Location',[0,176,396,362])

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave2'])
set_param([sys,'/','Reference/Sine Wave2'],...
                'position',[100,125],...
                'size',[20,20],...
                'amplitude','-ramp*rfreq*rfreq',...
                'frequency','rfreq',...
```

```
                'phase','0')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave1'])
set_param([sys,'/','Reference/Sine Wave1'],...
                'position',[100,75],...
                'size',[20,20],...
                'amplitude','ramp*rfreq',...
                'frequency','rfreq',...
                'phase','3.1415926/2')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave'])
set_param([sys,'/','Reference/Sine Wave'],...
                'position',[100,25],...
                'size',[20,20],...
                'amplitude','ramp',...
                'frequency','rfreq',...
                'phase','0')

add_block('built-in/Outport',[sys,'/','Reference/Out_1'])
set_param([sys,'/','Reference/Out_1'],...
                'position',[330,35],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Mux',[sys,'/','Reference/Mux1'])
set_param([sys,'/','Reference/Mux1'],...
                'position',[275,27],...
                'size',[30,36],...
                'inputs','3')
add_line([sys,'/','Reference'],[125,135;245,135;245,55;270,55])
add_line([sys,'/','Reference'],[125,85;150,85;150,45;270,45])
add_line([sys,'/','Reference'],[310,45;325,45])
add_line([sys,'/','Reference'],[125,35;270,35])


%    Finished composite block 'Reference'.

set_param([sys,'/','Reference'],...
                'position',[40,50],...
                'size',[30,50])

add_block('built-in/Outport',[sys,'/','Outport'])
set_param([sys,'/','Outport'],...
```

```
                'position',[445,80],...
                'size',[20,20],...
                'Port','1')


%    Subsystem  'measurement'.

new_system([sys,'/','measurement'])
set_param([sys,'/','measurement'],'Location',[3,42,509,381])

add_block('built-in/Demux',[sys,'/','measurement/Demux'])
set_param([sys,'/','measurement/Demux'],...
                'position',[70,19],...
                'size',[40,32],...
                'outputs','3')

add_block('built-in/Inport',[sys,'/','measurement/In_1'])
set_param([sys,'/','measurement/In_1'],...
                'position',[25,25],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sum',[sys,'/','measurement/Sum'])
set_param([sys,'/','measurement/Sum'],...
                'position',[235,20],...
                'size',[20,20],...
                'inputs','+-')

add_block('built-in/Outport',[sys,'/','measurement/Out_4'])
set_param([sys,'/','measurement/Out_4'],...
                'position',[285,20],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Demux',[sys,'/','measurement/Demux1'])
set_param([sys,'/','measurement/Demux1'],...
                'position',[70,87],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Inport',[sys,'/','measurement/In_2'])
set_param([sys,'/','measurement/In_2'],...
                'position',[25,95],...
```

```
                'size',[20,20],...
                'Port','2')
add_line([sys,'/','measurement'],[115,25;230,25])
add_line([sys,'/','measurement'],[115,95;185,95;185,35;230,35])
add_line([sys,'/','measurement'],[50,105;65,105])
add_line([sys,'/','measurement'],[50,35;65,35])
add_line([sys,'/','measurement'],[260,30;280,30])


%    Finished composite block 'measurement'.

set_param([sys,'/','measurement'],...
                'position',[365,62],...
                'size',[30,56])


%    Subsystem  'FL-LQGLTR'.

new_system([sys,'/','FL-LQGLTR'])
set_param([sys,'/','FL-LQGLTR'],'Location',[3,42,509,381])

add_block('built-in/Clock',[sys,'/','FL-LQGLTR/Clock'])
set_param([sys,'/','FL-LQGLTR/Clock'],...
                'position',[160,155],...
                'size',[20,20])

add_block('built-in/Product',[sys,'/','FL-LQGLTR/Product'])
set_param([sys,'/','FL-LQGLTR/Product'],...
                'position',[340,110],...
                'size',[25,20],...
                'inputs','2')

add_block('built-in/State-space',[sys,'/','FL-LQGLTR/MBC Controller'])
set_param([sys,'/','FL-LQGLTR/MBC Controller'],...
                'position',[385,5],...
                'size',[60,30],...
                'A','Acfdb',...
                'B','-Hfdb',...
                'C','-Gfdb',...
                'D','0')

add_block('built-in/Inport',[sys,'/','FL-LQGLTR/In_2'])
set_param([sys,'/','FL-LQGLTR/In_2'],...
```

```
                'position',[25,80],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Demux',[sys,'/','FL-LQGLTR/Demux1'])
set_param([sys,'/','FL-LQGLTR/Demux1'],...
                'position',[90,72],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Demux',[sys,'/','FL-LQGLTR/Demux'])
set_param([sys,'/','FL-LQGLTR/Demux'],...
                'position',[90,9],...
                'size',[40,32],...
                'outputs','3')

add_block('built-in/Outport',[sys,'/','FL-LQGLTR/Out_1'])
set_param([sys,'/','FL-LQGLTR/Out_1'],...
                'position',[585,30],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Inport',[sys,'/','FL-LQGLTR/In_1'])
set_param([sys,'/','FL-LQGLTR/In_1'],...
                'position',[25,15],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sum',[sys,'/','FL-LQGLTR/Sum'])
set_param([sys,'/','FL-LQGLTR/Sum'],...
                'position',[315,10],...
                'size',[20,20],...
                'inputs','+-')

add_block('built-in/Note',[sys,'/','FL-LQGLTR/e'])
set_param([sys,'/','FL-LQGLTR/e'],...
                'position',[300,0],...
                'size',[1,1])

add_block('built-in/Sum',[sys,'/','FL-LQGLTR/Sum3'])
set_param([sys,'/','FL-LQGLTR/Sum3'],...
                'position',[540,30],...
                'size',[20,20],...
```

```
                'inputs','++')

add_block('built-in/Sum',[sys,'/','FL-LQGLTR/Sum4'])
set_param([sys,'/','FL-LQGLTR/Sum4'],...
                'position',[490,75],...
                'size',[20,20],...
                'inputs','+-')

add_block('built-in/Gain',[sys,'/','FL-LQGLTR/Gain'])
set_param([sys,'/','FL-LQGLTR/Gain'],...
                'position',[345,70],...
                'size',[20,20],...
                'Gain','ah')

add_block('built-in/Fcn',[sys,'/','FL-LQGLTR/Fcn'])
set_param([sys,'/','FL-LQGLTR/Fcn'],...
                'position',[230,155],...
                'size',[40,20],...
                'Expr','cos(2*u[1])')

add_block('built-in/Gain',[sys,'/','FL-LQGLTR/Gain3'])
set_param([sys,'/','FL-LQGLTR/Gain3'],...
                'position',[420,110],...
                'size',[20,20],...
                'Gain','2*qh')
add_line([sys,'/','FL-LQGLTR'],[185,165;225,165])
add_line([sys,'/','FL-LQGLTR'],[275,165;285,165;285,125;335,125])
add_line([sys,'/','FL-LQGLTR'],[280,80;280,115;335,115])
add_line([sys,'/','FL-LQGLTR'],[370,120;415,120])
add_line([sys,'/','FL-LQGLTR'],[135,80;340,80])
add_line([sys,'/','FL-LQGLTR'],[450,20;495,20;495,35;535,35])
add_line([sys,'/','FL-LQGLTR'],[515,85;525,85;525,45;535,45])
add_line([sys,'/','FL-LQGLTR'],[280,80;280,25;310,25])
add_line([sys,'/','FL-LQGLTR'],[50,90;85,90])
add_line([sys,'/','FL-LQGLTR'],[50,25;85,25])
add_line([sys,'/','FL-LQGLTR'],[135,15;310,15])
add_line([sys,'/','FL-LQGLTR'],[340,20;380,20])
add_line([sys,'/','FL-LQGLTR'],[565,40;580,40])
add_line([sys,'/','FL-LQGLTR'],[370,80;485,80])
add_line([sys,'/','FL-LQGLTR'],[445,120;455,120;455,90;485,90])


%    Finished composite block 'FL-LQGLTR'.
```

```
set_param([sys,'/','FL-LQGLTR'],...
                'position',[165,62],...
                'size',[30,56])

add_block('built-in/White Noise',[sys,'/','White Noise'])
set_param([sys,'/','White Noise'],...
                'position',[40,135],...
                'size',[20,20],...
                'Seed','0')
add_line(sys,[200,90;245,90])
add_line(sys,[75,75;160,75])
add_line(sys,[285,105;360,105])
add_line(sys,[285,105;300,105;300,165;130,165;130,105;160,105])
add_line(sys,[75,75;105,75;105,45;335,45;335,75;360,75])
add_line(sys,[400,90;440,90])
add_line(sys,[65,145;215,145;215,115;245,115])
% Return any arguments.
if (nargin | nargout)
        % Must use feval here to access system in memory
        if (nargin > 3)
                if (flag == 0)
                        eval(['[ret,x0,xstr]=',sys,'(t,x,u,flag);'])
                else
                        eval(['ret =', sys,'(t,x,u,flag);'])
                end
        else
                [ret,x0,str] = feval(sys);
        end
end
```

A.2.3. Mass with Discontinuous Stiffness

```
function [ret,x0,str]=sprdiscstfto(t,x,u,flag);
%SPRDISCSTFTO  is the M-file description of the SIMULAB system
named SPRDISCSTFTO.
%       The block-diagram can be displayed by typing: SPRDISCSTFTO.
%
%       SYS=SPRDISCSTFTO(T,X,U,FLAG) returns depending on FLAG
certain
%       system values given time point, T, current state vector, X,
%       and input vector, U.
%       FLAG is used to indicate the type of output to be returned in
SYS.
%
%       Setting FLAG=1 causes SPRDISCSTFTO to return state
derivitives, FLAG=2
%       discrete states, FLAG=3 system outputs and FLAG=4 next
sample
%       time. For more information and other options see SFUNC.
%
%       Calling SPRDISCSTFTO with a FLAG of zero:
%       [SIZES]=SPRDISCSTFTO([],[],[],0),  returns a vector, SIZES, which
%       contains the sizes of the state vector and other parameters.
%             SIZES(1) number of states
%             SIZES(2) number of discrete states
%             SIZES(3) number of outputs
%             SIZES(4) number of inputs.
%       For the definition of other parameters in SIZES, see SFUNC.
%       See also, TRIM, LINMOD, LINSIM, EULER, RK23, RK45, ADAMS,
GEAR.

% Note: This M-file is only used for saving graphical information;
%       after the model is loaded into memory an internal model
%       representation is used.

% the system will take on the name of this mfile:
sys = mfilename;
new_system(sys)
simulab_version(1.01)
if(0 == (nargin + nargout))
   set_param(sys,'Location',[73,41,473,246])
```

```
    open_system(sys)
end;
set_param(sys,'algorithm',          'Linear')
set_param(sys,'Start time',   '0.0')
set_param(sys,'Stop time',          '20')
set_param(sys,'Min step size',      '0.05')
set_param(sys,'Max step size',      '0.05')
set_param(sys,'Relative error','1e-5')
set_param(sys,'Return vars', '')


%    Subsystem  'springdyn'.

new_system([sys,'/','springdyn'])
set_param([sys,'/','springdyn'],'Location',[3,42,509,381])

add_block('built-in/Gain',[sys,'/','springdyn/Gain4'])
set_param([sys,'/','springdyn/Gain4'],...
                'position',[85,35],...
                'size',[20,20],...
                'Gain','1/m1')

add_block('built-in/Inport',[sys,'/','springdyn/noise'])
set_param([sys,'/','springdyn/noise'],...
                'position',[330,5],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Look Up Table',[sys,'/','springdyn/Look Up
Table'])
set_param([sys,'/','springdyn/Look Up Table'],...
                'position',[270,145],...
                'size',[25,20],...
                'orientation',2,...
                'Input_Values','[-z3 -z1 z1 z3]',...
                'Output_Values','[-y3 -y1 y1 y3]')

add_block('built-in/Gain',[sys,'/','springdyn/Gain2'])
set_param([sys,'/','springdyn/Gain2'],...
                'position',[90,140],...
                'size',[20,20],...
                'orientation',3,...
                'Gain','k3/m1')
```

```
add_block('built-in/Note',[sys,'/','springdyn/x1-x3'])
set_param([sys,'/','springdyn/x1-x3'],...
                'position',[445,225],...
                'size',[1,1])

add_block('built-in/Gain',[sys,'/','springdyn/Gain1'])
set_param([sys,'/','springdyn/Gain1'],...
                'position',[230,315],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k3/m2')

add_block('built-in/Sum',[sys,'/','springdyn/Sum2'])
set_param([sys,'/','springdyn/Sum2'],...
                'position',[405,227],...
                'size',[20,36],...
                'inputs','+-')

add_block('built-in/Note',[sys,'/','springdyn/x3'])
set_param([sys,'/','springdyn/x3'],...
                'position',[365,235],...
                'size',[1,1])

add_block('built-in/Note',[sys,'/','springdyn/x4'])
set_param([sys,'/','springdyn/x4'],...
                'position',[260,235],...
                'size',[1,1])

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator3'])
set_param([sys,'/','springdyn/Integrator3'],...
                'position',[335,245],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator2'])
set_param([sys,'/','springdyn/Integrator2'],...
                'position',[220,245],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Mux',[sys,'/','springdyn/Mux'])
set_param([sys,'/','springdyn/Mux'],...
```

```
                'position',[515,92],...
                'size',[30,36],...
                'inputs','2')

add_block('built-in/Outport',[sys,'/','springdyn/Outport'])
set_param([sys,'/','springdyn/Outport'],...
                'position',[575,100],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Inport',[sys,'/','springdyn/Inport'])
set_param([sys,'/','springdyn/Inport'],...
                'position',[20,35],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator'])
set_param([sys,'/','springdyn/Integrator'],...
                'position',[320,90],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Gain',[sys,'/','springdyn/Gain'])
set_param([sys,'/','springdyn/Gain'],...
                'position',[200,145],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','1/m1')

add_block('built-in/Sum',[sys,'/','springdyn/Sum'])
set_param([sys,'/','springdyn/Sum'],...
                'position',[175,82],...
                'size',[20,36],...
                'inputs','+--')

add_block('built-in/Note',[sys,'/','springdyn/x1'])
set_param([sys,'/','springdyn/x1'],...
                'position',[355,76],...
                'size',[1,1])

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator1'])
set_param([sys,'/','springdyn/Integrator1'],...
                'position',[240,90],...
```

```
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Note',[sys,'/','springdyn/x2'])
set_param([sys,'/','springdyn/x2'],...
                'position',[270,76],...
                'size',[1,1])

add_block('built-in/Gain',[sys,'/','springdyn/Gain3'])
set_param([sys,'/','springdyn/Gain3'],...
                'position',[385,45],...
                'size',[20,20],...
                'orientation',1,...
                'Gain','d')

add_block('built-in/Sum',[sys,'/','springdyn/Sum1'])
set_param([sys,'/','springdyn/Sum1'],...
                'position',[415,85],...
                'size',[20,20],...
                'inputs','++')
add_line([sys,'/','springdyn'],[110,45;135,45;135,90;170,90])
add_line([sys,'/','springdyn'],[45,45;80,45])
add_line([sys,'/','springdyn'],[225,325;165,325;165,255;215,255])
add_line([sys,'/','springdyn'],[360,100;360,155;300,155])
add_line([sys,'/','springdyn'],[265,155;225,155])
add_line([sys,'/','springdyn'],[345,100;380,100;380,235;400,235])
add_line([sys,'/','springdyn'],[450,245;450,215;100,215;100,165])
add_line([sys,'/','springdyn'],[430,245;450,245;450,325;255,325])
add_line([sys,'/','springdyn'],[360,255;400,255])
add_line([sys,'/','springdyn'],[245,255;330,255])
add_line([sys,'/','springdyn'],[550,110;570,110])
add_line([sys,'/','springdyn'],[265,100;315,100])
add_line([sys,'/','springdyn'],[200,100;235,100])
add_line([sys,'/','springdyn'],[195,155;150,155;150,110;170,110])
add_line([sys,'/','springdyn'],[100,135;100,100;170,100])
add_line([sys,'/','springdyn'],[290,100;290,130;415,130;415,120;510,1
20])
add_line([sys,'/','springdyn'],[395,70;395,90;410,90])
add_line([sys,'/','springdyn'],[380,100;410,100])
add_line([sys,'/','springdyn'],[440,95;450,95;450,100;510,100])
add_line([sys,'/','springdyn'],[355,15;395,15;395,40])
```

```
%    Finished composite block 'springdyn'.

set_param([sys,'/','springdyn'],...
                'position',[260,62],...
                'size',[30,51])


%    Subsystem  'Reference'.

new_system([sys,'/','Reference'])
set_param([sys,'/','Reference'],'Location',[0,176,396,362])

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave2'])
set_param([sys,'/','Reference/Sine Wave2'],...
                'position',[100,125],...
                'size',[20,20],...
                'amplitude','-ramp*rfreq*rfreq',...
                'frequency','rfreq',...
                'phase','0')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave1'])
set_param([sys,'/','Reference/Sine Wave1'],...
                'position',[100,75],...
                'size',[20,20],...
                'amplitude','ramp*rfreq',...
                'frequency','rfreq',...
                'phase','3.1415926/2')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave'])
set_param([sys,'/','Reference/Sine Wave'],...
                'position',[100,25],...
                'size',[20,20],...
                'amplitude','ramp',...
                'frequency','rfreq',...
                'phase','0')

add_block('built-in/Outport',[sys,'/','Reference/Out_1'])
set_param([sys,'/','Reference/Out_1'],...
                'position',[330,35],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Mux',[sys,'/','Reference/Mux1'])
```

```
set_param([sys,'/','Reference/Mux1'],...
                'position',[275,27],...
                'size',[30,36],...
                'inputs','3')
add_line([sys,'/','Reference'],[125,135;245,135;245,55;270,55])
add_line([sys,'/','Reference'],[125,85;150,85;150,45;270,45])
add_line([sys,'/','Reference'],[310,45;325,45])
add_line([sys,'/','Reference'],[125,35;270,35])


%    Finished composite block 'Reference'.

set_param([sys,'/','Reference'],...
                'position',[30,35],...
                'size',[30,50])


%    Subsystem  'FL-LQGLTR'.

new_system([sys,'/','FL-LQGLTR'])
set_param([sys,'/','FL-LQGLTR'],'Location',[22,230,529,399])

add_block('built-in/Gain',[sys,'/','FL-LQGLTR/Gain'])
set_param([sys,'/','FL-LQGLTR/Gain'],...
                'position',[340,70],...
                'size',[20,20],...
                'Gain','1/mh')

add_block('built-in/State-space',[sys,'/','FL-LQGLTR/State-space'])
set_param([sys,'/','FL-LQGLTR/State-space'],...
                'position',[270,5],...
                'size',[60,30],...
                'A','Acfdb',...
                'B','-Hfdb',...
                'C','Gfdb',...
                'D','0')

add_block('built-in/Inport',[sys,'/','FL-LQGLTR/In_2'])
set_param([sys,'/','FL-LQGLTR/In_2'],...
                'position',[25,80],...
                'size',[20,20],...
                'Port','2')
```

```
add_block('built-in/Demux',[sys,'/','FL-LQGLTR/Demux1'])
set_param([sys,'/','FL-LQGLTR/Demux1'],...
                'position',[90,72],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Demux',[sys,'/','FL-LQGLTR/Demux'])
set_param([sys,'/','FL-LQGLTR/Demux'],...
                'position',[90,9],...
                'size',[40,32],...
                'outputs','3')

add_block('built-in/Outport',[sys,'/','FL-LQGLTR/Out_1'])
set_param([sys,'/','FL-LQGLTR/Out_1'],...
                'position',[515,30],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Inport',[sys,'/','FL-LQGLTR/In_1'])
set_param([sys,'/','FL-LQGLTR/In_1'],...
                'position',[25,15],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sum',[sys,'/','FL-LQGLTR/Sum'])
set_param([sys,'/','FL-LQGLTR/Sum'],...
                'position',[200,10],...
                'size',[20,20],...
                'inputs','-+')

add_block('built-in/Note',[sys,'/','FL-LQGLTR/e'])
set_param([sys,'/','FL-LQGLTR/e'],...
                'position',[230,0],...
                'size',[1,1])

add_block('built-in/Look Up Table',[sys,'/','FL-LQGLTR/Look Up
Table'])
set_param([sys,'/','FL-LQGLTR/Look Up Table'],...
                'position',[240,70],...
                'size',[25,20],...
                'Input_Values','[-z3 -z1h z1h z3]',...
                'Output_Values','[-y3h -y1h y1h y3h]')
```

```
add_block('built-in/Sum',[sys,'/','FL-LQGLTR/Sum3'])
set_param([sys,'/','FL-LQGLTR/Sum3'],...
                'position',[450,30],...
                'size',[20,20],...
                'inputs','++')
add_line([sys,'/','FL-LQGLTR'],[365,80;410,80;410,45;445,45])
add_line([sys,'/','FL-LQGLTR'],[270,80;335,80])
add_line([sys,'/','FL-LQGLTR'],[335,20;345,20;345,35;445,35])
add_line([sys,'/','FL-LQGLTR'],[50,90;85,90])
add_line([sys,'/','FL-LQGLTR'],[50,25;85,25])
add_line([sys,'/','FL-LQGLTR'],[135,15;195,15])
add_line([sys,'/','FL-LQGLTR'],[135,80;180,80;180,25;195,25])
add_line([sys,'/','FL-LQGLTR'],[225,20;265,20])
add_line([sys,'/','FL-LQGLTR'],[180,80;235,80])
add_line([sys,'/','FL-LQGLTR'],[475,40;510,40])


%    Finished composite block 'FL-LQGLTR'.

set_param([sys,'/','FL-LQGLTR'],...
                'position',[170,47],...
                'size',[30,56])

add_block('built-in/White Noise',[sys,'/','White Noise'])
set_param([sys,'/','White Noise'],...
                'position',[35,140],...
                'size',[20,20],...
                'Seed','0')


%    Subsystem  'measurement'.

new_system([sys,'/','measurement'])
set_param([sys,'/','measurement'],'Location',[3,42,509,381])

add_block('built-in/Inport',[sys,'/','measurement/In_2'])
set_param([sys,'/','measurement/In_2'],...
                'position',[25,95],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Demux',[sys,'/','measurement/Demux1'])
set_param([sys,'/','measurement/Demux1'],...
```

```
                'position',[70,87],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Outport',[sys,'/','measurement/Out_4'])
set_param([sys,'/','measurement/Out_4'],...
                'position',[285,20],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sum',[sys,'/','measurement/Sum'])
set_param([sys,'/','measurement/Sum'],...
                'position',[235,20],...
                'size',[20,20],...
                'inputs','+-')

add_block('built-in/Inport',[sys,'/','measurement/In_1'])
set_param([sys,'/','measurement/In_1'],...
                'position',[25,25],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Demux',[sys,'/','measurement/Demux'])
set_param([sys,'/','measurement/Demux'],...
                'position',[70,19],...
                'size',[40,32],...
                'outputs','3')
add_line([sys,'/','measurement'],[260,30;280,30])
add_line([sys,'/','measurement'],[50,35;65,35])
add_line([sys,'/','measurement'],[50,105;65,105])
add_line([sys,'/','measurement'],[115,95;185,95;185,35;230,35])
add_line([sys,'/','measurement'],[115,25;230,25])


%    Finished composite block 'measurement'.

set_param([sys,'/','measurement'],...
                'position',[370,47],...
                'size',[30,56])

add_block('built-in/Outport',[sys,'/','Outport'])
set_param([sys,'/','Outport'],...
                'position',[450,65],...
```

```
                'size',[20,20],...
                'Port','1')
add_line(sys,[205,75;255,75])
add_line(sys,[65,60;165,60])
add_line(sys,[295,90;365,90])
add_line(sys,[295,90;300,90;300,185;130,185;130,90;165,90])
add_line(sys,[65,60;105,60;105,20;335,20;335,60;365,60])
add_line(sys,[60,150;220,150;220,100;255,100])
add_line(sys,[405,75;445,75])
% Return any arguments.
if (nargin | nargout)
        % Must use feval here to access system in memory
        if (nargin > 3)
                if (flag == 0)
                        eval(['[ret,x0,xstr]=',sys,'(t,x,u,flag);'])
                else
                        eval(['ret =', sys,'(t,x,u,flag);'])
                end
        else
                [ret,x0,str] = feval(sys);
        end
end
```

A.2.4. Mass w/Dry Friction

```
function [ret,x0,str]=mechto(t,x,u,flag);
%MECHTO  is the M-file description of the SIMULAB system named
MECHTO.
%       The block-diagram can be displayed by typing: MECHTO.
%
%       SYS=MECHTO(T,X,U,FLAG) returns depending on FLAG certain
%       system values given time point, T, current state vector, X,
%       and input vector, U.
%       FLAG is used to indicate the type of output to be returned in
SYS.
%
%       Setting FLAG=1 causes MECHTO to return state derivitives,
FLAG=2
%       discrete states, FLAG=3 system outputs and FLAG=4 next
sample
%       time. For more information and other options see SFUNC.
%
%       Calling MECHTO with a FLAG of zero:
%       [SIZES]=MECHTO([],[],[],0),  returns a vector, SIZES, which
%       contains the sizes of the state vector and other parameters.
%             SIZES(1) number of states
%             SIZES(2) number of discrete states
%             SIZES(3) number of outputs
%             SIZES(4) number of inputs.
%       For the definition of other parameters in SIZES, see SFUNC.
%       See also, TRIM, LINMOD, LINSIM, EULER, RK23, RK45, ADAMS,
GEAR.

% Note: This M-file is only used for saving graphical information;
%       after the model is loaded into memory an internal model
%       representation is used.

% the system will take on the name of this mfile:
sys = mfilename;
new_system(sys)
simulab_version(1.01)
if(0 == (nargin + nargout))
    set_param(sys,'Location',[33,106,493,307])
    open_system(sys)
```

```
end;
set_param(sys,'algorithm',          'Linear')
set_param(sys,'Start time',   '0.0')
set_param(sys,'Stop time',          '20')
set_param(sys,'Min step size',      '0.05')
set_param(sys,'Max step size',      '0.05')
set_param(sys,'Relative error','1e-5')
set_param(sys,'Return vars', '')


%    Subsystem 'Reference'.

new_system([sys,'/','Reference'])
set_param([sys,'/','Reference'],'Location',[0,176,396,362])

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave2'])
set_param([sys,'/','Reference/Sine Wave2'],...
                'position',[100,125],...
                'size',[20,20],...
                'amplitude','-ramp*rfreq*rfreq',...
                'frequency','rfreq',...
                'phase','0')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave1'])
set_param([sys,'/','Reference/Sine Wave1'],...
                'position',[100,75],...
                'size',[20,20],...
                'amplitude','ramp*rfreq',...
                'frequency','rfreq',...
                'phase','3.1415926/2')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave'])
set_param([sys,'/','Reference/Sine Wave'],...
                'position',[100,25],...
                'size',[20,20],...
                'amplitude','ramp',...
                'frequency','rfreq',...
                'phase','0')

add_block('built-in/Outport',[sys,'/','Reference/Out_1'])
set_param([sys,'/','Reference/Out_1'],...
                'position',[330,35],...
                'size',[20,20],...
```

```
                'Port','1')

add_block('built-in/Mux',[sys,'/','Reference/Mux1'])
set_param([sys,'/','Reference/Mux1'],...
                'position',[275,27],...
                'size',[30,36],...
                'inputs','3')
add_line([sys,'/','Reference'],[125,135;245,135;245,55;270,55])
add_line([sys,'/','Reference'],[125,85;150,85;150,45;270,45])
add_line([sys,'/','Reference'],[310,45;325,45])
add_line([sys,'/','Reference'],[125,35;270,35])


%    Finished composite block 'Reference'.

set_param([sys,'/','Reference'],...
                'position',[35,40],...
                'size',[30,50])


%    Subsystem  'FL-LQGLTR'.

new_system([sys,'/','FL-LQGLTR'])
set_param([sys,'/','FL-LQGLTR'],'Location',[5,184,512,353])

add_block('built-in/Sum',[sys,'/','FL-LQGLTR/Sum3'])
set_param([sys,'/','FL-LQGLTR/Sum3'],...
                'position',[450,27],...
                'size',[20,36],...
                'inputs','+++')

add_block('built-in/Note',[sys,'/','FL-LQGLTR/e'])
set_param([sys,'/','FL-LQGLTR/e'],...
                'position',[230,0],...
                'size',[1,1])

add_block('built-in/Sum',[sys,'/','FL-LQGLTR/Sum'])
set_param([sys,'/','FL-LQGLTR/Sum'],...
                'position',[200,10],...
                'size',[20,20],...
                'inputs','-+')

add_block('built-in/Inport',[sys,'/','FL-LQGLTR/In_1'])
```

```
set_param([sys,'/','FL-LQGLTR/In_1'],...
                'position',[25,15],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Outport',[sys,'/','FL-LQGLTR/Out_1'])
set_param([sys,'/','FL-LQGLTR/Out_1'],...
                'position',[515,35],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Demux',[sys,'/','FL-LQGLTR/Demux'])
set_param([sys,'/','FL-LQGLTR/Demux'],...
                'position',[90,9],...
                'size',[40,32],...
                'outputs','3')

add_block('built-in/Demux',[sys,'/','FL-LQGLTR/Demux1'])
set_param([sys,'/','FL-LQGLTR/Demux1'],...
                'position',[90,72],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Inport',[sys,'/','FL-LQGLTR/In_2'])
set_param([sys,'/','FL-LQGLTR/In_2'],...
                'position',[25,80],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Note',[sys,'/','FL-LQGLTR/z1'])
set_param([sys,'/','FL-LQGLTR/z1'],...
                'position',[140,60],...
                'size',[1,1])

add_block('built-in/Gain',[sys,'/','FL-LQGLTR/Gain2'])
set_param([sys,'/','FL-LQGLTR/Gain2'],...
                'position',[345,125],...
                'size',[20,20],...
                'Gain','hh/mh')

add_block('built-in/MATLAB Fcn',[sys,'/','FL-LQGLTR/MATLAB Fcn'])
set_param([sys,'/','FL-LQGLTR/MATLAB Fcn'],...
                'position',[245,120],...
```

```
                'size',[50,30],...
                'MATLAB Fcn','sign',...
                'Output Width','-1')

add_block('built-in/Gain',[sys,'/','FL-LQGLTR/Gain3'])
set_param([sys,'/','FL-LQGLTR/Gain3'],...
                'position',[375,70],...
                'size',[20,20],...
                'Gain','kh/mh')

add_block('built-in/State-space',[sys,'/','FL-LQGLTR/MBC Controller'])
set_param([sys,'/','FL-LQGLTR/MBC Controller'],...
                'position',[260,5],...
                'size',[60,30],...
                'A','Acfdb',...
                'B','-Hfdb',...
                'C','Gfdb',...
                'D','0')
add_line([sys,'/','FL-LQGLTR'],[475,45;510,45])
add_line([sys,'/','FL-LQGLTR'],[180,80;370,80])
add_line([sys,'/','FL-LQGLTR'],[225,20;255,20])
add_line([sys,'/','FL-LQGLTR'],[135,100;195,100;195,135;240,135])
add_line([sys,'/','FL-LQGLTR'],[135,80;180,80;180,25;195,25])
add_line([sys,'/','FL-LQGLTR'],[135,15;195,15])
add_line([sys,'/','FL-LQGLTR'],[50,25;85,25])
add_line([sys,'/','FL-LQGLTR'],[50,90;85,90])
add_line([sys,'/','FL-LQGLTR'],[400,80;420,80;420,45;445,45])
add_line([sys,'/','FL-LQGLTR'],[300,135;340,135])
add_line([sys,'/','FL-LQGLTR'],[370,135;435,135;435,55;445,55])
add_line([sys,'/','FL-LQGLTR'],[325,20;390,20;390,35;445,35])


%    Finished composite block 'FL-LQGLTR'.

set_param([sys,'/','FL-LQGLTR'],...
                'position',[165,52],...
                'size',[30,56])

add_block('built-in/White Noise',[sys,'/','White Noise'])
set_param([sys,'/','White Noise'],...
                'position',[50,145],...
                'size',[20,20],...
                'Seed','0')
```

```
%    Subsystem  ['mass w',13,'dry friction'].

new_system([sys,'/',['mass w',13,'dry friction']])
set_param([sys,'/',['mass w',13,'dry
friction']],'Location',[3,42,509,381])

add_block('built-in/Gain',[sys,'/',['mass w',13,'dry friction/Gain6']])
set_param([sys,'/',['mass w',13,'dry friction/Gain6']],...
                'position',[90,80],...
                'size',[20,20],...
                'Gain','1/m1')

add_block('built-in/Gain',[sys,'/',['mass w',13,'dry friction/Gain4']])
set_param([sys,'/',['mass w',13,'dry friction/Gain4']],...
                'position',[235,220],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k1/m1')

add_block('built-in/Gain',[sys,'/',['mass w',13,'dry friction/Gain3']])
set_param([sys,'/',['mass w',13,'dry friction/Gain3']],...
                'position',[185,175],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','h/m1')

add_block('built-in/MATLAB Fcn',[sys,'/',['mass w',13,'dry
friction/MATLAB Fcn']])
set_param([sys,'/',['mass w',13,'dry friction/MATLAB Fcn']],...
                'position',[240,170],...
                'size',[50,30],...
                'orientation',2,...
                'MATLAB Fcn','sign',...
                'Output Width','-1')

add_block('built-in/Note',[sys,'/',['mass w',13,'dry friction/x2']])
set_param([sys,'/',['mass w',13,'dry friction/x2']],...
                'position',[270,76],...
                'size',[1,1])
```

```
add_block('built-in/Integrator',[sys,'/',['mass w',13,'dry
friction/Integrator1']])
set_param([sys,'/',['mass w',13,'dry friction/Integrator1']],...
                'position',[240,90],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Note',[sys,'/',['mass w',13,'dry friction/x1']])
set_param([sys,'/',['mass w',13,'dry friction/x1']],...
                'position',[355,76],...
                'size',[1,1])

add_block('built-in/Sum',[sys,'/',['mass w',13,'dry friction/Sum']])
set_param([sys,'/',['mass w',13,'dry friction/Sum']],...
                'position',[180,82],...
                'size',[20,36],...
                'inputs','+-')

add_block('built-in/Integrator',[sys,'/',['mass w',13,'dry
friction/Integrator']])
set_param([sys,'/',['mass w',13,'dry friction/Integrator']],...
                'position',[320,90],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Inport',[sys,'/',['mass w',13,'dry friction/Inport']])
set_param([sys,'/',['mass w',13,'dry friction/Inport']],...
                'position',[15,80],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Outport',[sys,'/',['mass w',13,'dry
friction/Outport']])
set_param([sys,'/',['mass w',13,'dry friction/Outport']],...
                'position',[565,100],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Mux',[sys,'/',['mass w',13,'dry friction/Mux']])
set_param([sys,'/',['mass w',13,'dry friction/Mux']],...
                'position',[505,92],...
                'size',[30,36],...
                'inputs','2')
```

```
add_block('built-in/Integrator',[sys,'/',['mass w',13,'dry
friction/Integrator2']])
set_param([sys,'/',['mass w',13,'dry friction/Integrator2']],...
                'position',[220,305],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Integrator',[sys,'/',['mass w',13,'dry
friction/Integrator3']])
set_param([sys,'/',['mass w',13,'dry friction/Integrator3']],...
                'position',[335,305],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Note',[sys,'/',['mass w',13,'dry friction/x4']])
set_param([sys,'/',['mass w',13,'dry friction/x4']],...
                'position',[260,295],...
                'size',[1,1])

add_block('built-in/Note',[sys,'/',['mass w',13,'dry friction/x3']])
set_param([sys,'/',['mass w',13,'dry friction/x3']],...
                'position',[365,295],...
                'size',[1,1])

add_block('built-in/Sum',[sys,'/',['mass w',13,'dry friction/Sum2']])
set_param([sys,'/',['mass w',13,'dry friction/Sum2']],...
                'position',[405,287],...
                'size',[20,36],...
                'inputs','+-')

add_block('built-in/Gain',[sys,'/',['mass w',13,'dry friction/Gain1']])
set_param([sys,'/',['mass w',13,'dry friction/Gain1']],...
                'position',[230,365],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k2/m2')

add_block('built-in/Note',[sys,'/',['mass w',13,'dry friction/x1-x3']])
set_param([sys,'/',['mass w',13,'dry friction/x1-x3']],...
                'position',[445,250],...
                'size',[1,1])
```

```
add_block('built-in/Gain',[sys,'/',['mass w',13,'dry friction/Gain2']])
set_param([sys,'/',['mass w',13,'dry friction/Gain2']],...
                'position',[220,265],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k2/m1')

add_block('built-in/Sum',[sys,'/',['mass w',13,'dry friction/Sum3']])
set_param([sys,'/',['mass w',13,'dry friction/Sum3']],...
                'position',[110,247],...
                'size',[20,36],...
                'orientation',2,...
                'inputs','+++')

add_block('built-in/Inport',[sys,'/',['mass w',13,'dry
friction/Inport1']])
set_param([sys,'/',['mass w',13,'dry friction/Inport1']],...
                'position',[350,10],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Gain',[sys,'/',['mass w',13,'dry friction/Gain5']])
set_param([sys,'/',['mass w',13,'dry friction/Gain5']],...
                'position',[390,45],...
                'size',[20,20],...
                'orientation',1,...
                'Gain','d')

add_block('built-in/Sum',[sys,'/',['mass w',13,'dry friction/Sum1']])
set_param([sys,'/',['mass w',13,'dry friction/Sum1']],...
                'position',[425,85],...
                'size',[20,20],...
                'inputs','++')
add_line([sys,'/',['mass w',13,'dry friction']],[115,90;175,90])
add_line([sys,'/',['mass w',13,'dry friction']],[40,90;85,90])
add_line([sys,'/',['mass w',13,'dry
friction']],[295,100;295,135;425,135;425,120;500,120])
add_line([sys,'/',['mass w',13,'dry friction']],[235,185;210,185])
add_line([sys,'/',['mass w',13,'dry
friction']],[360,100;360,185;295,185])
add_line([sys,'/',['mass w',13,'dry friction']],[205,100;235,100])
add_line([sys,'/',['mass w',13,'dry friction']],[265,100;315,100])
add_line([sys,'/',['mass w',13,'dry friction']],[540,110;560,110])
```

```
add_line([sys,'/',['mass w',13,'dry friction']],[245,315;330,315])
add_line([sys,'/',['mass w',13,'dry friction']],[360,315;400,315])
add_line([sys,'/',['mass w',13,'dry
friction']],[430,305;450,305;450,375;255,375])
add_line([sys,'/',['mass w',13,'dry
friction']],[225,375;105,375;105,315;215,315])
add_line([sys,'/',['mass w',13,'dry
friction']],[345,100;380,100;380,230;260,230])
add_line([sys,'/',['mass w',13,'dry
friction']],[380,230;380,295;400,295])
add_line([sys,'/',['mass w',13,'dry
friction']],[450,305;450,275;245,275])
add_line([sys,'/',['mass w',13,'dry friction']],[215,275;135,275])
add_line([sys,'/',['mass w',13,'dry
friction']],[230,230;200,230;200,265;135,265])
add_line([sys,'/',['mass w',13,'dry
friction']],[180,185;155,185;155,255;135,255])
add_line([sys,'/',['mass w',13,'dry
friction']],[105,265;85,265;85,110;175,110])
add_line([sys,'/',['mass w',13,'dry friction']],[400,70;400,90;420,90])
add_line([sys,'/',['mass w',13,'dry friction']],[375,20;400,20;400,40])
add_line([sys,'/',['mass w',13,'dry friction']],[380,100;420,100])
add_line([sys,'/',['mass w',13,'dry
friction']],[450,95;460,95;460,100;500,100])


%    Finished composite block ['mass w',13,'dry friction'].

set_param([sys,'/',['mass w',13,'dry friction']],...
                'position',[250,67],...
                'size',[30,51])


%    Subsystem  'measurement'.

new_system([sys,'/','measurement'])
set_param([sys,'/','measurement'],'Location',[3,42,509,381])

add_block('built-in/Inport',[sys,'/','measurement/In_2'])
set_param([sys,'/','measurement/In_2'],...
                'position',[25,95],...
                'size',[20,20],...
                'Port','2')
```

```
add_block('built-in/Demux',[sys,'/','measurement/Demux1'])
set_param([sys,'/','measurement/Demux1'],...
                'position',[70,87],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Outport',[sys,'/','measurement/Out_4'])
set_param([sys,'/','measurement/Out_4'],...
                'position',[285,20],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sum',[sys,'/','measurement/Sum'])
set_param([sys,'/','measurement/Sum'],...
                'position',[235,20],...
                'size',[20,20],...
                'inputs','+-')

add_block('built-in/Inport',[sys,'/','measurement/In_1'])
set_param([sys,'/','measurement/In_1'],...
                'position',[25,25],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Demux',[sys,'/','measurement/Demux'])
set_param([sys,'/','measurement/Demux'],...
                'position',[70,19],...
                'size',[40,32],...
                'outputs','3')
add_line([sys,'/','measurement'],[260,30;280,30])
add_line([sys,'/','measurement'],[50,35;65,35])
add_line([sys,'/','measurement'],[50,105;65,105])
add_line([sys,'/','measurement'],[115,95;185,95;185,35;230,35])
add_line([sys,'/','measurement'],[115,25;230,25])


%    Finished composite block 'measurement'.

set_param([sys,'/','measurement'],...
                'position',[370,52],...
                'size',[30,56])
```

```
add_block('built-in/Outport',[sys,'/','Outport'])
set_param([sys,'/','Outport'],...
                'position',[455,70],...
                'size',[20,20],...
                'Port','1')
add_line(sys,[75,155;220,155;220,105;245,105])
add_line(sys,[200,80;245,80])
add_line(sys,[70,65;160,65])
add_line(sys,[285,95;365,95])
add_line(sys,[70,65;105,65;105,45;335,45;335,65;365,65])
add_line(sys,[405,80;450,80])
add_line(sys,[285,95;310,95;310,180;140,180;140,95;160,95])
% Return any arguments.
if (nargin | nargout)
        % Must use feval here to access system in memory
        if (nargin > 3)
                if (flag == 0)
                        eval(['[ret,x0,xstr]=',sys,'(t,x,u,flag);'])
                else
                        eval(['ret =', sys,'(t,x,u,flag);'])
                end
        else
                [ret,x0,str] = feval(sys);
        end
end
```

A.2.5. Duffing's Equation w/Input Saturation

```
function [ret,x0,str]=sprsatto(t,x,u,flag);
%SPRSATTO       is the M-file description of the SIMULAB system
named SPRSATTO.
%       The block-diagram can be displayed by typing: SPRSATTO.
%
%       SYS=SPRSATTO(T,X,U,FLAG) returns depending on FLAG certain
%       system values given time point, T, current state vector, X,
%       and input vector, U.
%       FLAG is used to indicate the type of output to be returned in
SYS.
%
%       Setting FLAG=1 causes SPRSATTO to return state derivitives,
FLAG=2
%       discrete states, FLAG=3 system outputs and FLAG=4 next
sample
%       time. For more information and other options see SFUNC.
%
%       Calling SPRSATTO with a FLAG of zero:
%       [SIZES]=SPRSATTO([],[],[],0),  returns a vector, SIZES, which
%       contains the sizes of the state vector and other parameters.
%               SIZES(1) number of states
%               SIZES(2) number of discrete states
%               SIZES(3) number of outputs
%               SIZES(4) number of inputs.
%       For the definition of other parameters in SIZES, see SFUNC.
%       See also, TRIM, LINMOD, LINSIM, EULER, RK23, RK45, ADAMS,
GEAR.

% Note: This M-file is only used for saving graphical information;
%       after the model is loaded into memory an internal model
%       representation is used.

% the system will take on the name of this mfile:
sys = mfilename;
new_system(sys)
simulab_version(1.01)
if(0 == (nargin + nargout))
    set_param(sys,'Location',[3,223,502,382])
    open_system(sys)
```

```
end;
set_param(sys,'algorithm',          'Linear')
set_param(sys,'Start time',   '0.0')
set_param(sys,'Stop time',          '20')
set_param(sys,'Min step size',      '0.05')
set_param(sys,'Max step size',      '0.05')
set_param(sys,'Relative error','1e-5')
set_param(sys,'Return vars', '')


%    Subsystem  'measurement'.

new_system([sys,'/','measurement'])
set_param([sys,'/','measurement'],'Location',[3,42,509,381])

add_block('built-in/Inport',[sys,'/','measurement/In_2'])
set_param([sys,'/','measurement/In_2'],...
                'position',[25,95],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Demux',[sys,'/','measurement/Demux1'])
set_param([sys,'/','measurement/Demux1'],...
                'position',[70,87],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Outport',[sys,'/','measurement/Out_4'])
set_param([sys,'/','measurement/Out_4'],...
                'position',[285,20],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sum',[sys,'/','measurement/Sum'])
set_param([sys,'/','measurement/Sum'],...
                'position',[235,20],...
                'size',[20,20],...
                'inputs','+-')

add_block('built-in/Inport',[sys,'/','measurement/In_1'])
set_param([sys,'/','measurement/In_1'],...
                'position',[25,25],...
                'size',[20,20],...
```

```
                'Port','1')

add_block('built-in/Demux',[sys,'/','measurement/Demux'])
set_param([sys,'/','measurement/Demux'],...
                'position',[70,19],...
                'size',[40,32],...
                'outputs','3')
add_line([sys,'/','measurement'],[260,30;280,30])
add_line([sys,'/','measurement'],[50,35;65,35])
add_line([sys,'/','measurement'],[50,105;65,105])
add_line([sys,'/','measurement'],[115,95;185,95;185,35;230,35])
add_line([sys,'/','measurement'],[115,25;230,25])


%    Finished composite block 'measurement'.

set_param([sys,'/','measurement'],...
                'position',[370,62],...
                'size',[30,56])

add_block('built-in/Outport',[sys,'/','Outport'])
set_param([sys,'/','Outport'],...
                'position',[450,80],...
                'size',[20,20],...
                'Port','1')


%    Subsystem  'Controller'.

new_system([sys,'/','Controller'])
set_param([sys,'/','Controller'],'Location',[19,219,526,388])

add_block('built-in/State-space',[sys,'/','Controller/State-space'])
set_param([sys,'/','Controller/State-space'],...
                'position',[270,5],...
                'size',[60,30],...
                'A','Acfdb',...
                'B','-Hfdb',...
                'C','-Gfdb',...
                'D','0')

add_block('built-in/Inport',[sys,'/','Controller/In_2'])
set_param([sys,'/','Controller/In_2'],...
```

```
                'position',[25,80],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Demux',[sys,'/','Controller/Demux1'])
set_param([sys,'/','Controller/Demux1'],...
                'position',[90,72],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Demux',[sys,'/','Controller/Demux'])
set_param([sys,'/','Controller/Demux'],...
                'position',[90,9],...
                'size',[40,32],...
                'outputs','3')

add_block('built-in/Outport',[sys,'/','Controller/Out_1'])
set_param([sys,'/','Controller/Out_1'],...
                'position',[515,30],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Inport',[sys,'/','Controller/In_1'])
set_param([sys,'/','Controller/In_1'],...
                'position',[25,15],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sum',[sys,'/','Controller/Sum'])
set_param([sys,'/','Controller/Sum'],...
                'position',[200,10],...
                'size',[20,20],...
                'inputs','+-')

add_block('built-in/Note',[sys,'/','Controller/e'])
set_param([sys,'/','Controller/e'],...
                'position',[230,0],...
                'size',[1,1])

add_block('built-in/Sum',[sys,'/','Controller/Sum3'])
set_param([sys,'/','Controller/Sum3'],...
                'position',[450,30],...
                'size',[20,20],...
```

```
                'inputs','++')

add_block('built-in/Sum',[sys,'/','Controller/Sum4'])
set_param([sys,'/','Controller/Sum4'],...
                'position',[375,75],...
                'size',[20,20],...
                'inputs','++')

add_block('built-in/Gain',[sys,'/','Controller/Gain'])
set_param([sys,'/','Controller/Gain'],...
                'position',[230,70],...
                'size',[20,20],...
                'Gain','k1h/mh')

add_block('built-in/Fcn',[sys,'/','Controller/Fcn'])
set_param([sys,'/','Controller/Fcn'],...
                'position',[220,110],...
                'size',[40,20],...
                'Expr','ah*u[1]*u[1]*u[1]')

add_block('built-in/Gain',[sys,'/','Controller/Gain3'])
set_param([sys,'/','Controller/Gain3'],...
                'position',[305,110],...
                'size',[20,20],...
                'Gain','k1h/mh')
add_line([sys,'/','Controller'],[335,20;345,20;345,35;445,35])
add_line([sys,'/','Controller'],[50,90;85,90])
add_line([sys,'/','Controller'],[50,25;85,25])
add_line([sys,'/','Controller'],[135,15;195,15])
add_line([sys,'/','Controller'],[135,80;180,80;180,25;195,25])
add_line([sys,'/','Controller'],[225,20;265,20])
add_line([sys,'/','Controller'],[475,40;510,40])
add_line([sys,'/','Controller'],[400,85;420,85;420,45;445,45])
add_line([sys,'/','Controller'],[180,80;225,80])
add_line([sys,'/','Controller'],[255,80;370,80])
add_line([sys,'/','Controller'],[180,80;180,120;215,120])
add_line([sys,'/','Controller'],[265,120;300,120])
add_line([sys,'/','Controller'],[330,120;340,120;340,90;370,90])


%    Finished composite block 'Controller'.

set_param([sys,'/','Controller'],...
```

```
                'position',[165,62],...
                'size',[30,56])

add_block('built-in/White Noise',[sys,'/','White Noise'])
set_param([sys,'/','White Noise'],...
                'position',[75,140],...
                'size',[20,20],...
                'Seed','0')


%    Subsystem  'Reference'.

new_system([sys,'/','Reference'])
set_param([sys,'/','Reference'],'Location',[0,176,396,362])

add_block('built-in/Mux',[sys,'/','Reference/Mux1'])
set_param([sys,'/','Reference/Mux1'],...
                'position',[275,27],...
                'size',[30,36],...
                'inputs','3')

add_block('built-in/Outport',[sys,'/','Reference/Out_1'])
set_param([sys,'/','Reference/Out_1'],...
                'position',[330,35],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave'])
set_param([sys,'/','Reference/Sine Wave'],...
                'position',[100,25],...
                'size',[20,20],...
                'amplitude','ramp',...
                'frequency','rfreq',...
                'phase','0')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave1'])
set_param([sys,'/','Reference/Sine Wave1'],...
                'position',[100,75],...
                'size',[20,20],...
                'amplitude','ramp*rfreq',...
                'frequency','rfreq',...
                'phase','3.1415926/2')
```

```
add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave2'])
set_param([sys,'/','Reference/Sine Wave2'],...
                'position',[100,125],...
                'size',[20,20],...
                'amplitude','-ramp*rfreq*rfreq',...
                'frequency','rfreq',...
                'phase','0')
add_line([sys,'/','Reference'],[125,35;270,35])
add_line([sys,'/','Reference'],[310,45;325,45])
add_line([sys,'/','Reference'],[125,85;150,85;150,45;270,45])
add_line([sys,'/','Reference'],[125,135;245,135;245,55;270,55])


%    Finished composite block 'Reference'.

set_param([sys,'/','Reference'],...
                'position',[35,50],...
                'size',[30,50])


%    Subsystem  'springdyn'.

new_system([sys,'/','springdyn'])
set_param([sys,'/','springdyn'],'Location',[7,89,507,391])

add_block('built-in/Fcn',[sys,'/','springdyn/Fcn'])
set_param([sys,'/','springdyn/Fcn'],...
                'position',[285,230],...
                'size',[40,20],...
                'orientation',2,...
                'Expr','a*u[1]*u[1]*u[1]')

add_block('built-in/Gain',[sys,'/','springdyn/Gain3'])
set_param([sys,'/','springdyn/Gain3'],...
                'position',[200,230],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k1/m')

add_block('built-in/Saturation',[sys,'/','springdyn/Saturation'])
set_param([sys,'/','springdyn/Saturation'],...
                'position',[60,55],...
                'size',[25,20],...
```

```
                'Upper Limit','satlim',...
                'Lower Limit','-satlim')

add_block('built-in/Gain',[sys,'/','springdyn/Gain5'])
set_param([sys,'/','springdyn/Gain5'],...
                'position',[115,55],...
                'size',[20,20],...
                'Gain','1/m1')

add_block('built-in/Gain',[sys,'/','springdyn/Gain4'])
set_param([sys,'/','springdyn/Gain4'],...
                'position',[375,15],...
                'size',[20,20],...
                'Gain','d')

add_block('built-in/Sum',[sys,'/','springdyn/Sum3'])
set_param([sys,'/','springdyn/Sum3'],...
                'position',[435,57],...
                'size',[20,36],...
                'inputs','++')

add_block('built-in/Gain',[sys,'/','springdyn/Gain2'])
set_param([sys,'/','springdyn/Gain2'],...
                'position',[100,190],...
                'size',[20,20],...
                'orientation',3,...
                'Gain','k2/m1')

add_block('built-in/Note',[sys,'/','springdyn/x1-x3'])
set_param([sys,'/','springdyn/x1-x3'],...
                'position',[445,250],...
                'size',[1,1])

add_block('built-in/Gain',[sys,'/','springdyn/Gain1'])
set_param([sys,'/','springdyn/Gain1'],...
                'position',[230,350],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k2/m2')

add_block('built-in/Sum',[sys,'/','springdyn/Sum2'])
set_param([sys,'/','springdyn/Sum2'],...
                'position',[415,282],...
```

```
                'size',[20,36],...
                'inputs','+-')

add_block('built-in/Note',[sys,'/','springdyn/x3'])
set_param([sys,'/','springdyn/x3'],...
                'position',[375,290],...
                'size',[1,1])

add_block('built-in/Note',[sys,'/','springdyn/x4'])
set_param([sys,'/','springdyn/x4'],...
                'position',[270,290],...
                'size',[1,1])

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator3'])
set_param([sys,'/','springdyn/Integrator3'],...
                'position',[345,300],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator2'])
set_param([sys,'/','springdyn/Integrator2'],...
                'position',[230,300],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Mux',[sys,'/','springdyn/Mux'])
set_param([sys,'/','springdyn/Mux'],...
                'position',[485,72],...
                'size',[30,36],...
                'inputs','2')

add_block('built-in/Outport',[sys,'/','springdyn/Outport'])
set_param([sys,'/','springdyn/Outport'],...
                'position',[545,80],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Inport',[sys,'/','springdyn/Inport'])
set_param([sys,'/','springdyn/Inport'],...
                'position',[10,55],...
                'size',[20,20],...
                'Port','1')
```

```
add_block('built-in/Integrator',[sys,'/','springdyn/Integrator'])
set_param([sys,'/','springdyn/Integrator'],...
                'position',[330,100],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Gain',[sys,'/','springdyn/Gain'])
set_param([sys,'/','springdyn/Gain'],...
                'position',[250,180],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k1/m1')

add_block('built-in/Sum',[sys,'/','springdyn/Sum'])
set_param([sys,'/','springdyn/Sum'],...
                'position',[195,49],...
                'size',[20,152],...
                'inputs','+---')

add_block('built-in/Note',[sys,'/','springdyn/x1'])
set_param([sys,'/','springdyn/x1'],...
                'position',[365,86],...
                'size',[1,1])

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator1'])
set_param([sys,'/','springdyn/Integrator1'],...
                'position',[250,100],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Note',[sys,'/','springdyn/x2'])
set_param([sys,'/','springdyn/x2'],...
                'position',[280,86],...
                'size',[1,1])

add_block('built-in/Inport',[sys,'/','springdyn/Inport1'])
set_param([sys,'/','springdyn/Inport1'],...
                'position',[290,15],...
                'size',[20,20],...
                'Port','2')
add_line([sys,'/','springdyn'],[195,240;155,240;155,145;190,145])
add_line([sys,'/','springdyn'],[370,190;370,240;330,240])
add_line([sys,'/','springdyn'],[280,240;225,240])
```

```
add_line([sys,'/','springdyn'],[245,190;235,190;235,225;180,225;180,1
85;190,185])
add_line([sys,'/','springdyn'],[35,65;55,65])
add_line([sys,'/','springdyn'],[90,65;110,65])
add_line([sys,'/','springdyn'],[370,110;370,190;275,190])
add_line([sys,'/','springdyn'],[390,110;390,290;410,290])
add_line([sys,'/','springdyn'],[355,110;390,110;390,85;430,85])
add_line([sys,'/','springdyn'],[140,65;190,65])
add_line([sys,'/','springdyn'],[400,25;410,25;410,65;430,65])
add_line([sys,'/','springdyn'],[315,25;370,25])
add_line([sys,'/','springdyn'],[460,75;470,75;470,80;480,80])
add_line([sys,'/','springdyn'],[225,360;130,360;130,310;225,310])
add_line([sys,'/','springdyn'],[110,185;110,105;190,105])
add_line([sys,'/','springdyn'],[290,110;290,150;425,150;425,100;480,1
00])
add_line([sys,'/','springdyn'],[450,300;460,300;460,265;110,265;110,2
15])
add_line([sys,'/','springdyn'],[440,300;450,300;450,360;255,360])
add_line([sys,'/','springdyn'],[370,310;410,310])
add_line([sys,'/','springdyn'],[255,310;340,310])
add_line([sys,'/','springdyn'],[520,90;540,90])
add_line([sys,'/','springdyn'],[275,110;325,110])
add_line([sys,'/','springdyn'],[220,125;220,110;245,110])


%    Finished composite block 'springdyn'.

set_param([sys,'/','springdyn'],...
                'position',[245,77],...
                'size',[30,51])
add_line(sys,[405,90;445,90])
add_line(sys,[280,105;365,105])
add_line(sys,[280,105;300,105;300,165;130,165;130,105;160,105])
add_line(sys,[70,75;105,75;105,45;335,45;335,75;365,75])
add_line(sys,[70,75;160,75])
add_line(sys,[200,90;240,90])
add_line(sys,[100,150;215,150;215,115;240,115])
% Return any arguments.
if (nargin | nargout)
        % Must use feval here to access system in memory
        if (nargin > 3)
                if (flag == 0)
                        eval(['[ret,x0,xstr]=',sys,'(t,x,u,flag);'])
```

```
            else
                eval(['ret =', sys,'(t,x,u,flag);'])
            end
        else
            [ret,x0,str] = feval(sys);
        end
end
```

A.2.6. Duffing's Equation with Deadband in Input

```
function [ret,x0,str]=sprsatto(t,x,u,flag);
%SPRSATTO       is the M-file description of the SIMULAB system
named SPRSATTO.
%       The block-diagram can be displayed by typing: SPRSATTO.
%
%       SYS=SPRSATTO(T,X,U,FLAG) returns depending on FLAG certain
%       system values given time point, T, current state vector, X,
%       and input vector, U.
%       FLAG is used to indicate the type of output to be returned in
SYS.
%
%       Setting FLAG=1 causes SPRSATTO to return state derivitives,
FLAG=2
%       discrete states, FLAG=3 system outputs and FLAG=4 next
sample
%       time. For more information and other options see SFUNC.
%
%       Calling SPRSATTO with a FLAG of zero:
%       [SIZES]=SPRSATTO([],[],[],0),  returns a vector, SIZES, which
%       contains the sizes of the state vector and other parameters.
%               SIZES(1) number of states
%               SIZES(2) number of discrete states
%               SIZES(3) number of outputs
%               SIZES(4) number of inputs.
%       For the definition of other parameters in SIZES, see SFUNC.
%       See also, TRIM, LINMOD, LINSIM, EULER, RK23, RK45, ADAMS,
GEAR.

% Note: This M-file is only used for saving graphical information;
%       after the model is loaded into memory an internal model
%       representation is used.

% the system will take on the name of this mfile:
sys = mfilename;
new_system(sys)
simulab_version(1.01)
if(0 == (nargin + nargout))
    set_param(sys,'Location',[28,57,527,216])
    open_system(sys)
```

```
end;
set_param(sys,'algorithm',          'Linear')
set_param(sys,'Start time',   '0.0')
set_param(sys,'Stop time',          '20')
set_param(sys,'Min step size',      '0.05')
set_param(sys,'Max step size',      '0.05')
set_param(sys,'Relative error','1e-5')
set_param(sys,'Return vars', '')


%    Subsystem  'springdyn'.

new_system([sys,'/','springdyn'])
set_param([sys,'/','springdyn'],'Location',[16,51,472,392])

add_block('built-in/Dead Zone',[sys,'/','springdyn/Dead Zone'])
set_param([sys,'/','springdyn/Dead Zone'],...
                'position',[60,55],...
                'size',[25,20],...
                'Lower_value','-deadlim',...
                'Upper_value','deadlim')

add_block('built-in/Inport',[sys,'/','springdyn/Inport1'])
set_param([sys,'/','springdyn/Inport1'],...
                'position',[290,15],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Note',[sys,'/','springdyn/x2'])
set_param([sys,'/','springdyn/x2'],...
                'position',[280,86],...
                'size',[1,1])

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator1'])
set_param([sys,'/','springdyn/Integrator1'],...
                'position',[250,100],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Note',[sys,'/','springdyn/x1'])
set_param([sys,'/','springdyn/x1'],...
                'position',[365,86],...
                'size',[1,1])
```

```
add_block('built-in/Sum',[sys,'/','springdyn/Sum'])
set_param([sys,'/','springdyn/Sum'],...
                'position',[195,53],...
                'size',[20,154],...
                'inputs','+---')

add_block('built-in/Gain',[sys,'/','springdyn/Gain'])
set_param([sys,'/','springdyn/Gain'],...
                'position',[250,180],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k1/m1')

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator'])
set_param([sys,'/','springdyn/Integrator'],...
                'position',[330,100],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Inport',[sys,'/','springdyn/Inport'])
set_param([sys,'/','springdyn/Inport'],...
                'position',[10,55],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Outport',[sys,'/','springdyn/Outport'])
set_param([sys,'/','springdyn/Outport'],...
                'position',[545,80],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Mux',[sys,'/','springdyn/Mux'])
set_param([sys,'/','springdyn/Mux'],...
                'position',[485,72],...
                'size',[30,36],...
                'inputs','2')

add_block('built-in/Integrator',[sys,'/','springdyn/Integrator2'])
set_param([sys,'/','springdyn/Integrator2'],...
                'position',[230,300],...
                'size',[20,20],...
                'Initial','0')
```

```
add_block('built-in/Integrator',[sys,'/','springdyn/Integrator3'])
set_param([sys,'/','springdyn/Integrator3'],...
                'position',[345,300],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Note',[sys,'/','springdyn/x4'])
set_param([sys,'/','springdyn/x4'],...
                'position',[270,290],...
                'size',[1,1])

add_block('built-in/Note',[sys,'/','springdyn/x3'])
set_param([sys,'/','springdyn/x3'],...
                'position',[375,290],...
                'size',[1,1])

add_block('built-in/Sum',[sys,'/','springdyn/Sum2'])
set_param([sys,'/','springdyn/Sum2'],...
                'position',[415,282],...
                'size',[20,36],...
                'inputs','+-')

add_block('built-in/Gain',[sys,'/','springdyn/Gain1'])
set_param([sys,'/','springdyn/Gain1'],...
                'position',[230,350],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k2/m2')

add_block('built-in/Note',[sys,'/','springdyn/x1-x3'])
set_param([sys,'/','springdyn/x1-x3'],...
                'position',[445,250],...
                'size',[1,1])

add_block('built-in/Gain',[sys,'/','springdyn/Gain2'])
set_param([sys,'/','springdyn/Gain2'],...
                'position',[100,190],...
                'size',[20,20],...
                'orientation',3,...
                'Gain','k2/m1')

add_block('built-in/Sum',[sys,'/','springdyn/Sum3'])
```

```
set_param([sys,'/','springdyn/Sum3'],...
                'position',[435,57],...
                'size',[20,36],...
                'inputs','++')

add_block('built-in/Gain',[sys,'/','springdyn/Gain4'])
set_param([sys,'/','springdyn/Gain4'],...
                'position',[375,15],...
                'size',[20,20],...
                'Gain','d')

add_block('built-in/Gain',[sys,'/','springdyn/Gain5'])
set_param([sys,'/','springdyn/Gain5'],...
                'position',[115,55],...
                'size',[20,20],...
                'Gain','1/m1')

add_block('built-in/Gain',[sys,'/','springdyn/Gain3'])
set_param([sys,'/','springdyn/Gain3'],...
                'position',[200,230],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k1/m')

add_block('built-in/Fcn',[sys,'/','springdyn/Fcn'])
set_param([sys,'/','springdyn/Fcn'],...
                'position',[285,230],...
                'size',[40,20],...
                'orientation',2,...
                'Expr','a*u[1]*u[1]*u[1]')
add_line([sys,'/','springdyn'],[220,130;220,110;245,110])
add_line([sys,'/','springdyn'],[275,110;325,110])
add_line([sys,'/','springdyn'],[520,90;540,90])
add_line([sys,'/','springdyn'],[255,310;340,310])
add_line([sys,'/','springdyn'],[370,310;410,310])
add_line([sys,'/','springdyn'],[440,300;450,300;450,360;255,360])
add_line([sys,'/','springdyn'],[450,300;460,300;460,265;110,265;110,2
15])
add_line([sys,'/','springdyn'],[290,110;290,150;425,150;425,100;480,1
00])
add_line([sys,'/','springdyn'],[110,185;110,110;190,110])
add_line([sys,'/','springdyn'],[225,360;130,360;130,310;225,310])
add_line([sys,'/','springdyn'],[460,75;470,75;470,80;480,80])
```

```
add_line([sys,'/','springdyn'],[315,25;370,25])
add_line([sys,'/','springdyn'],[400,25;410,25;410,65;430,65])
add_line([sys,'/','springdyn'],[140,65;190,70])
add_line([sys,'/','springdyn'],[355,110;390,110;390,85;430,85])
add_line([sys,'/','springdyn'],[390,110;390,290;410,290])
add_line([sys,'/','springdyn'],[370,110;370,190;275,190])
add_line([sys,'/','springdyn'],[90,65;110,65])
add_line([sys,'/','springdyn'],[35,65;55,65])
add_line([sys,'/','springdyn'],[245,190;235,190;235,225;180,225;180,1
90;190,190])
add_line([sys,'/','springdyn'],[280,240;225,240])
add_line([sys,'/','springdyn'],[370,190;370,240;330,240])
add_line([sys,'/','springdyn'],[195,240;155,240;155,150;190,150])


%    Finished composite block 'springdyn'.

set_param([sys,'/','springdyn'],...
                'position',[245,77],...
                'size',[30,51])


%    Subsystem  'Reference'.

new_system([sys,'/','Reference'])
set_param([sys,'/','Reference'],'Location',[0,176,396,362])

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave2'])
set_param([sys,'/','Reference/Sine Wave2'],...
                'position',[100,125],...
                'size',[20,20],...
                'amplitude','-ramp*rfreq*rfreq',...
                'frequency','rfreq',...
                'phase','0')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave1'])
set_param([sys,'/','Reference/Sine Wave1'],...
                'position',[100,75],...
                'size',[20,20],...
                'amplitude','ramp*rfreq',...
                'frequency','rfreq',...
                'phase','3.1415926/2')
```

```
add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave'])
set_param([sys,'/','Reference/Sine Wave'],...
                'position',[100,25],...
                'size',[20,20],...
                'amplitude','ramp',...
                'frequency','rfreq',...
                'phase','0')

add_block('built-in/Outport',[sys,'/','Reference/Out_1'])
set_param([sys,'/','Reference/Out_1'],...
                'position',[330,35],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Mux',[sys,'/','Reference/Mux1'])
set_param([sys,'/','Reference/Mux1'],...
                'position',[275,27],...
                'size',[30,36],...
                'inputs','3')
add_line([sys,'/','Reference'],[125,135;245,135;245,55;270,55])
add_line([sys,'/','Reference'],[125,85;150,85;150,45;270,45])
add_line([sys,'/','Reference'],[310,45;325,45])
add_line([sys,'/','Reference'],[125,35;270,35])


%    Finished composite block 'Reference'.

set_param([sys,'/','Reference'],...
                'position',[35,50],...
                'size',[30,50])

add_block('built-in/White Noise',[sys,'/','White Noise'])
set_param([sys,'/','White Noise'],...
                'position',[75,140],...
                'size',[20,20],...
                'Seed','0')


%    Subsystem  'Controller'.

new_system([sys,'/','Controller'])
set_param([sys,'/','Controller'],'Location',[19,219,526,388])
```

```
add_block('built-in/Gain',[sys,'/','Controller/Gain3'])
set_param([sys,'/','Controller/Gain3'],...
                'position',[305,110],...
                'size',[20,20],...
                'Gain','k1h/mh')

add_block('built-in/Fcn',[sys,'/','Controller/Fcn'])
set_param([sys,'/','Controller/Fcn'],...
                'position',[220,110],...
                'size',[40,20],...
                'Expr','ah*u[1]*u[1]*u[1]')

add_block('built-in/Gain',[sys,'/','Controller/Gain'])
set_param([sys,'/','Controller/Gain'],...
                'position',[230,70],...
                'size',[20,20],...
                'Gain','k1h/mh')

add_block('built-in/Sum',[sys,'/','Controller/Sum4'])
set_param([sys,'/','Controller/Sum4'],...
                'position',[375,75],...
                'size',[20,20],...
                'inputs','++')

add_block('built-in/Sum',[sys,'/','Controller/Sum3'])
set_param([sys,'/','Controller/Sum3'],...
                'position',[450,30],...
                'size',[20,20],...
                'inputs','++')

add_block('built-in/Note',[sys,'/','Controller/e'])
set_param([sys,'/','Controller/e'],...
                'position',[230,0],...
                'size',[1,1])

add_block('built-in/Sum',[sys,'/','Controller/Sum'])
set_param([sys,'/','Controller/Sum'],...
                'position',[200,10],...
                'size',[20,20],...
                'inputs','+-')

add_block('built-in/Inport',[sys,'/','Controller/In_1'])
set_param([sys,'/','Controller/In_1'],...
```

```
                'position',[25,15],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Outport',[sys,'/','Controller/Out_1'])
set_param([sys,'/','Controller/Out_1'],...
                'position',[515,30],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Demux',[sys,'/','Controller/Demux'])
set_param([sys,'/','Controller/Demux'],...
                'position',[90,9],...
                'size',[40,32],...
                'outputs','3')

add_block('built-in/Demux',[sys,'/','Controller/Demux1'])
set_param([sys,'/','Controller/Demux1'],...
                'position',[90,72],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Inport',[sys,'/','Controller/In_2'])
set_param([sys,'/','Controller/In_2'],...
                'position',[25,80],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/State-space',[sys,'/','Controller/State-space'])
set_param([sys,'/','Controller/State-space'],...
                'position',[270,5],...
                'size',[60,30],...
                'A','Acfdb',...
                'B','-Hfdb',...
                'C','-Gfdb',...
                'D','0')
add_line([sys,'/','Controller'],[330,120;340,120;340,90;370,90])
add_line([sys,'/','Controller'],[265,120;300,120])
add_line([sys,'/','Controller'],[180,80;180,120;215,120])
add_line([sys,'/','Controller'],[255,80;370,80])
add_line([sys,'/','Controller'],[180,80;225,80])
add_line([sys,'/','Controller'],[400,85;420,85;420,45;445,45])
add_line([sys,'/','Controller'],[475,40;510,40])
```

```
add_line([sys,'/','Controller'],[225,20;265,20])
add_line([sys,'/','Controller'],[135,80;180,80;180,25;195,25])
add_line([sys,'/','Controller'],[135,15;195,15])
add_line([sys,'/','Controller'],[50,25;85,25])
add_line([sys,'/','Controller'],[50,90;85,90])
add_line([sys,'/','Controller'],[335,20;345,20;345,35;445,35])


%    Finished composite block 'Controller'.

set_param([sys,'/','Controller'],...
                'position',[165,62],...
                'size',[30,56])

add_block('built-in/Outport',[sys,'/','Outport'])
set_param([sys,'/','Outport'],...
                'position',[450,80],...
                'size',[20,20],...
                'Port','1')


%    Subsystem  'measurement'.

new_system([sys,'/','measurement'])
set_param([sys,'/','measurement'],'Location',[3,42,509,381])

add_block('built-in/Demux',[sys,'/','measurement/Demux'])
set_param([sys,'/','measurement/Demux'],...
                'position',[70,19],...
                'size',[40,32],...
                'outputs','3')

add_block('built-in/Inport',[sys,'/','measurement/In_1'])
set_param([sys,'/','measurement/In_1'],...
                'position',[25,25],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sum',[sys,'/','measurement/Sum'])
set_param([sys,'/','measurement/Sum'],...
                'position',[235,20],...
                'size',[20,20],...
                'inputs','+-')
```

```
add_block('built-in/Outport',[sys,'/','measurement/Out_4'])
set_param([sys,'/','measurement/Out_4'],...
                'position',[285,20],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Demux',[sys,'/','measurement/Demux1'])
set_param([sys,'/','measurement/Demux1'],...
                'position',[70,87],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Inport',[sys,'/','measurement/In_2'])
set_param([sys,'/','measurement/In_2'],...
                'position',[25,95],...
                'size',[20,20],...
                'Port','2')
add_line([sys,'/','measurement'],[115,25;230,25])
add_line([sys,'/','measurement'],[115,95;185,95;185,35;230,35])
add_line([sys,'/','measurement'],[50,105;65,105])
add_line([sys,'/','measurement'],[50,35;65,35])
add_line([sys,'/','measurement'],[260,30;280,30])


%    Finished composite block 'measurement'.

set_param([sys,'/','measurement'],...
                'position',[370,62],...
                'size',[30,56])
add_line(sys,[100,150;215,150;215,115;240,115])
add_line(sys,[200,90;240,90])
add_line(sys,[70,75;160,75])
add_line(sys,[70,75;105,75;105,45;335,45;335,75;365,75])
add_line(sys,[280,105;300,105;300,165;130,165;130,105;160,105])
add_line(sys,[280,105;365,105])
add_line(sys,[405,90;445,90])
% Return any arguments.
if (nargin | nargout)
        % Must use feval here to access system in memory
        if (nargin > 3)
                if (flag == 0)
                        eval(['[ret,x0,xstr]=',sys,'(t,x,u,flag);'])
```

```
            else
                eval(['ret =', sys,'(t,x,u,flag);'])
            end
        else
            [ret,x0,str] = feval(sys);
        end
end
```

A.2.7. One Link Robot

```
function [ret,x0,str]=robotto(t,x,u,flag);
%ROBOTTO is the M-file description of the SIMULAB system named
ROBOTTO.
%       The block-diagram can be displayed by typing: ROBOTTO.
%
%       SYS=ROBOTTO(T,X,U,FLAG) returns depending on FLAG certain
%       system values given time point, T, current state vector, X,
%       and input vector, U.
%       FLAG is used to indicate the type of output to be returned in
SYS.
%
%       Setting FLAG=1 causes ROBOTTO to return state derivitives,
FLAG=2
%       discrete states, FLAG=3 system outputs and FLAG=4 next
sample
%       time. For more information and other options see SFUNC.
%
%       Calling ROBOTTO with a FLAG of zero:
%       [SIZES]=ROBOTTO([],[],[],0), returns a vector, SIZES, which
%       contains the sizes of the state vector and other parameters.
%               SIZES(1) number of states
%               SIZES(2) number of discrete states
%               SIZES(3) number of outputs
%               SIZES(4) number of inputs.
%       For the definition of other parameters in SIZES, see SFUNC.
%       See also, TRIM, LINMOD, LINSIM, EULER, RK23, RK45, ADAMS,
GEAR.

% Note: This M-file is only used for saving graphical information;
%       after the model is loaded into memory an internal model
%       representation is used.

% the system will take on the name of this mfile:
sys = mfilename;
new_system(sys)
simulab_version(1.01)
if(0 == (nargin + nargout))
    set_param(sys,'Location',[100,100,444,311])
    open_system(sys)
```

```
end;
set_param(sys,'algorithm',          'Linear')
set_param(sys,'Start time',   '0.0')
set_param(sys,'Stop time',          '20')
set_param(sys,'Min step size',      '0.01')
set_param(sys,'Max step size',      '0.01')
set_param(sys,'Relative error','1e-3')
set_param(sys,'Return vars', '')

add_block('built-in/White Noise',[sys,'/','White Noise'])
set_param([sys,'/','White Noise'],...
          'position',[35,125],...
          'size',[20,20],...
          'Seed','0')

add_block('built-in/Outport',[sys,'/','Outport'])
set_param([sys,'/','Outport'],...
          'position',[445,65],...
          'size',[20,20],...
          'Port','1')


%    Subsystem  'measurement'.

new_system([sys,'/','measurement'])
set_param([sys,'/','measurement'],'Location',[3,42,509,381])

add_block('built-in/Demux',[sys,'/','measurement/Demux'])
set_param([sys,'/','measurement/Demux'],...
          'position',[70,19],...
          'size',[40,32],...
          'outputs','3')

add_block('built-in/Inport',[sys,'/','measurement/In_1'])
set_param([sys,'/','measurement/In_1'],...
          'position',[25,25],...
          'size',[20,20],...
          'Port','1')

add_block('built-in/Sum',[sys,'/','measurement/Sum'])
set_param([sys,'/','measurement/Sum'],...
          'position',[235,20],...
          'size',[20,20],...
```

```
                'inputs','+-')

add_block('built-in/Outport',[sys,'/','measurement/Out_4'])
set_param([sys,'/','measurement/Out_4'],...
                'position',[285,20],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Demux',[sys,'/','measurement/Demux1'])
set_param([sys,'/','measurement/Demux1'],...
                'position',[70,87],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Inport',[sys,'/','measurement/In_2'])
set_param([sys,'/','measurement/In_2'],...
                'position',[25,95],...
                'size',[20,20],...
                'Port','2')
add_line([sys,'/','measurement'],[115,25;230,25])
add_line([sys,'/','measurement'],[115,95;185,95;185,35;230,35])
add_line([sys,'/','measurement'],[50,105;65,105])
add_line([sys,'/','measurement'],[50,35;65,35])
add_line([sys,'/','measurement'],[260,30;280,30])


%    Finished composite block 'measurement'.

set_param([sys,'/','measurement'],...
                'position',[360,47],...
                'size',[30,56])


%    Subsystem  'Reference'.

new_system([sys,'/','Reference'])
set_param([sys,'/','Reference'],'Location',[0,176,396,362])

add_block('built-in/Mux',[sys,'/','Reference/Mux1'])
set_param([sys,'/','Reference/Mux1'],...
                'position',[275,27],...
                'size',[30,36],...
                'inputs','3')
```

```
add_block('built-in/Outport',[sys,'/','Reference/Out_1'])
set_param([sys,'/','Reference/Out_1'],...
                'position',[330,35],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave'])
set_param([sys,'/','Reference/Sine Wave'],...
                'position',[100,25],...
                'size',[20,20],...
                'amplitude','ramp',...
                'frequency','rfreq',...
                'phase','0')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave1'])
set_param([sys,'/','Reference/Sine Wave1'],...
                'position',[100,75],...
                'size',[20,20],...
                'amplitude','ramp*rfreq',...
                'frequency','rfreq',...
                'phase','3.1415926/2')

add_block('built-in/Sine Wave',[sys,'/','Reference/Sine Wave2'])
set_param([sys,'/','Reference/Sine Wave2'],...
                'position',[100,125],...
                'size',[20,20],...
                'amplitude','-ramp*rfreq*rfreq',...
                'frequency','rfreq',...
                'phase','0')
add_line([sys,'/','Reference'],[125,35;270,35])
add_line([sys,'/','Reference'],[310,45;325,45])
add_line([sys,'/','Reference'],[125,85;150,85;150,45;270,45])
add_line([sys,'/','Reference'],[125,135;245,135;245,55;270,55])


%    Finished composite block 'Reference'.

set_param([sys,'/','Reference'],...
                'position',[35,50],...
                'size',[30,50])
```

```
%    Subsystem 'robotdyn'.

new_system([sys,'/','robotdyn'])
set_param([sys,'/','robotdyn'],'Location',[62,44,503,386])

add_block('built-in/Inport',[sys,'/','robotdyn/noise'])
set_param([sys,'/','robotdyn/noise'],...
                'position',[265,5],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Sum',[sys,'/','robotdyn/Sum1'])
set_param([sys,'/','robotdyn/Sum1'],...
                'position',[380,105],...
                'size',[20,20],...
                'inputs','++')

add_block('built-in/Gain',[sys,'/','robotdyn/Gain4'])
set_param([sys,'/','robotdyn/Gain4'],...
                'position',[335,40],...
                'size',[20,20],...
                'orientation',1,...
                'Gain','d')

add_block('built-in/Mux',[sys,'/','robotdyn/Mux'])
set_param([sys,'/','robotdyn/Mux'],...
                'position',[425,101],...
                'size',[30,53],...
                'inputs','2')

add_block('built-in/Integrator',[sys,'/','robotdyn/Integrator1'])
set_param([sys,'/','robotdyn/Integrator1'],...
                'position',[270,110],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Outport',[sys,'/','robotdyn/Outport'])
set_param([sys,'/','robotdyn/Outport'],...
                'position',[490,120],...
                'size',[20,20],...
                'Port','2')

add_block('built-in/Integrator',[sys,'/','robotdyn/Integrator'])
```

```
set_param([sys,'/','robotdyn/Integrator'],...
                'position',[200,110],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Fcn',[sys,'/','robotdyn/Fcn'])
set_param([sys,'/','robotdyn/Fcn'],...
                'position',[225,165],...
                'size',[40,20],...
                'orientation',2,...
                'Expr','sin(u[1])')

add_block('built-in/Gain',[sys,'/','robotdyn/Gain'])
set_param([sys,'/','robotdyn/Gain'],...
                'position',[60,230],...
                'size',[20,20],...
                'Gain','1/J2')

add_block('built-in/Sum',[sys,'/','robotdyn/Sum'])
set_param([sys,'/','robotdyn/Sum'],...
                'position',[120,102],...
                'size',[20,36],...
                'inputs','--')

add_block('built-in/Gain',[sys,'/','robotdyn/Gain1'])
set_param([sys,'/','robotdyn/Gain1'],...
                'position',[150,165],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','m*g*lg/J1')

add_block('built-in/Inport',[sys,'/','robotdyn/Inport'])
set_param([sys,'/','robotdyn/Inport'],...
                'position',[15,230],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Note',[sys,'/','robotdyn/x1'])
set_param([sys,'/','robotdyn/x1'],...
                'position',[310,105],...
                'size',[1,1])

add_block('built-in/Note',[sys,'/','robotdyn/x2'])
```

```
set_param([sys,'/','robotdyn/x2'],...
                'position',[240,105],...
                'size',[1,1])

add_block('built-in/Integrator',[sys,'/','robotdyn/Integrator2'])
set_param([sys,'/','robotdyn/Integrator2'],...
                'position',[200,240],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Integrator',[sys,'/','robotdyn/Integrator3'])
set_param([sys,'/','robotdyn/Integrator3'],...
                'position',[275,240],...
                'size',[20,20],...
                'Initial','0')

add_block('built-in/Note',[sys,'/','robotdyn/x4'])
set_param([sys,'/','robotdyn/x4'],...
                'position',[235,230],...
                'size',[1,1])

add_block('built-in/Note',[sys,'/','robotdyn/x3'])
set_param([sys,'/','robotdyn/x3'],...
                'position',[315,230],...
                'size',[1,1])

add_block('built-in/Sum',[sys,'/','robotdyn/Sum2'])
set_param([sys,'/','robotdyn/Sum2'],...
                'position',[355,222],...
                'size',[20,36],...
                'inputs','+-')

add_block('built-in/Gain',[sys,'/','robotdyn/Gain2'])
set_param([sys,'/','robotdyn/Gain2'],...
                'position',[250,320],...
                'size',[20,20],...
                'orientation',2,...
                'Gain','k1/J2')

add_block('built-in/Gain',[sys,'/','robotdyn/Gain3'])
set_param([sys,'/','robotdyn/Gain3'],...
                'position',[50,140],...
                'size',[20,20],...
```

```
                'orientation',3,...
                'Gain','k1/J1')

add_block('built-in/Note',[sys,'/','robotdyn/u'])
set_param([sys,'/','robotdyn/u'],...
                'position',[45,220],...
                'size',[1,1])

add_block('built-in/Mux',[sys,'/','robotdyn/Mux1'])
set_param([sys,'/','robotdyn/Mux1'],...
                'position',[430,261],...
                'size',[30,53],...
                'inputs','2')

add_block('built-in/Outport',[sys,'/','robotdyn/Outport1'])
set_param([sys,'/','robotdyn/Outport1'],...
                'position',[490,280],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sum',[sys,'/','robotdyn/Sum3'])
set_param([sys,'/','robotdyn/Sum3'],...
                'position',[120,232],...
                'size',[20,36],...
                'inputs','++')
add_line([sys,'/','robotdyn'],[345,65;345,110;375,110])
add_line([sys,'/','robotdyn'],[320,120;375,120])
add_line([sys,'/','robotdyn'],[405,115;420,115])
add_line([sys,'/','robotdyn'],[290,15;345,15;345,35])
add_line([sys,'/','robotdyn'],[245,250;245,300;425,300])
add_line([sys,'/','robotdyn'],[225,120;245,120;245,151;330,151;330,1
40;420,140])
add_line([sys,'/','robotdyn'],[460,130;485,130])
add_line([sys,'/','robotdyn'],[295,120;320,120;320,175;270,175])
add_line([sys,'/','robotdyn'],[225,120;265,120])
add_line([sys,'/','robotdyn'],[220,175;175,175])
add_line([sys,'/','robotdyn'],[145,120;195,120])
add_line([sys,'/','robotdyn'],[40,240;55,240])
add_line([sys,'/','robotdyn'],[225,250;270,250])
add_line([sys,'/','robotdyn'],[300,250;350,250])
add_line([sys,'/','robotdyn'],[320,175;320,230;350,230])
add_line([sys,'/','robotdyn'],[380,240;395,240;395,330;275,330])
add_line([sys,'/','robotdyn'],[395,240;395,200;60,200;60,165])
```

```
add_line([sys,'/','robotdyn'],[335,250;335,275;425,275])
add_line([sys,'/','robotdyn'],[465,290;485,290])
add_line([sys,'/','robotdyn'],[145,175;100,175;100,130;115,130])
add_line([sys,'/','robotdyn'],[60,135;60,110;115,110])
add_line([sys,'/','robotdyn'],[145,250;195,250])
add_line([sys,'/','robotdyn'],[85,240;115,240])
add_line([sys,'/','robotdyn'],[245,330;105,330;105,260;115,260])


%    Finished composite block 'robotdyn'.

set_param([sys,'/','robotdyn'],...
                'position',[250,77],...
                'size',[30,51])


%    Subsystem  'FL-LQGLTR'.

new_system([sys,'/','FL-LQGLTR'])
set_param([sys,'/','FL-LQGLTR'],'Location',[3,42,514,213])

add_block('built-in/Sum',[sys,'/','FL-LQGLTR/Sum1'])
set_param([sys,'/','FL-LQGLTR/Sum1'],...
                'position',[375,12],...
                'size',[20,36],...
                'inputs','+++')

add_block('built-in/Gain',[sys,'/','FL-LQGLTR/Gain1'])
set_param([sys,'/','FL-LQGLTR/Gain1'],...
                'position',[320,70],...
                'size',[20,20],...
                'Gain','mh*g*lgh')

add_block('built-in/Fcn',[sys,'/','FL-LQGLTR/Fcn'])
set_param([sys,'/','FL-LQGLTR/Fcn'],...
                'position',[245,70],...
                'size',[40,20],...
                'Expr','sin(u[1])')

add_block('built-in/Inport',[sys,'/','FL-LQGLTR/In_2'])
set_param([sys,'/','FL-LQGLTR/In_2'],...
                'position',[25,80],...
                'size',[20,20],...
```

```
                'Port','2')

add_block('built-in/Demux',[sys,'/','FL-LQGLTR/Demux1'])
set_param([sys,'/','FL-LQGLTR/Demux1'],...
                'position',[90,72],...
                'size',[40,36],...
                'outputs','2')

add_block('built-in/Demux',[sys,'/','FL-LQGLTR/Demux'])
set_param([sys,'/','FL-LQGLTR/Demux'],...
                'position',[90,9],...
                'size',[40,32],...
                'outputs','3')

add_block('built-in/Outport',[sys,'/','FL-LQGLTR/Out_1'])
set_param([sys,'/','FL-LQGLTR/Out_1'],...
                'position',[415,15],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Inport',[sys,'/','FL-LQGLTR/In_1'])
set_param([sys,'/','FL-LQGLTR/In_1'],...
                'position',[25,15],...
                'size',[20,20],...
                'Port','1')

add_block('built-in/Sum',[sys,'/','FL-LQGLTR/Sum'])
set_param([sys,'/','FL-LQGLTR/Sum'],...
                'position',[175,10],...
                'size',[20,20],...
                'inputs','+-')

add_block('built-in/Note',[sys,'/','FL-LQGLTR/e'])
set_param([sys,'/','FL-LQGLTR/e'],...
                'position',[205,5],...
                'size',[1,1])

add_block('built-in/Sum',[sys,'/','FL-LQGLTR/Sum2'])
set_param([sys,'/','FL-LQGLTR/Sum2'],...
                'position',[175,45],...
                'size',[20,20],...
                'inputs','+-')
```

```
add_block('built-in/Gain',[sys,'/','FL-LQGLTR/Gain2'])
set_param([sys,'/','FL-LQGLTR/Gain2'],...
                'position',[245,10],...
                'size',[20,20],...
                'Gain','100')

add_block('built-in/Gain',[sys,'/','FL-LQGLTR/Gain3'])
set_param([sys,'/','FL-LQGLTR/Gain3'],...
                'position',[245,45],...
                'size',[20,20],...
                'Gain','20')
add_line([sys,'/','FL-LQGLTR'],[400,30;410,25])
add_line([sys,'/','FL-LQGLTR'],[290,80;315,80])
add_line([sys,'/','FL-LQGLTR'],[150,80;240,80])
add_line([sys,'/','FL-LQGLTR'],[50,90;85,90])
add_line([sys,'/','FL-LQGLTR'],[50,25;85,25])
add_line([sys,'/','FL-LQGLTR'],[135,15;170,15])
add_line([sys,'/','FL-LQGLTR'],[135,80;150,80;150,25;170,25])
add_line([sys,'/','FL-LQGLTR'],[200,20;240,20])
add_line([sys,'/','FL-LQGLTR'],[135,100;160,100;160,60;170,60])
add_line([sys,'/','FL-LQGLTR'],[135,25;145,25;145,50;170,50])
add_line([sys,'/','FL-LQGLTR'],[200,55;240,55])
add_line([sys,'/','FL-LQGLTR'],[270,20;370,20])
add_line([sys,'/','FL-LQGLTR'],[345,80;365,80;370,40])
add_line([sys,'/','FL-LQGLTR'],[270,55;310,55;310,30;370,30])


%    Finished composite block 'FL-LQGLTR'.

set_param([sys,'/','FL-LQGLTR'],...
                'position',[165,62],...
                'size',[30,56])
add_line(sys,[60,135;225,135;225,115;245,115])
add_line(sys,[285,90;355,90])
add_line(sys,[285,90;295,90;295,155;130,155;130,105;160,105])
add_line(sys,[70,75;100,75;100,30;325,30;325,60;355,60])
add_line(sys,[395,75;440,75])
add_line(sys,[200,90;245,90])
add_line(sys,[70,75;160,75])
% Return any arguments.
if (nargin | nargout)
        % Must use feval here to access system in memory
        if (nargin > 3)
```

```
        if (flag == 0)
            eval(['[ret,x0,xstr]=',sys,'(t,x,u,flag);'])
        else
            eval(['ret =', sys,'(t,x,u,flag);'])
        end
    else
        [ret,x0,str] = feval(sys);
    end
end
```