Gust Alleviation using direct Gust Measurement

Sven Marco Hoppe

Flight Systems Research Center
Department of Electrical Engineering
University of California, Los Angeles

March 10, 2000

Contents

1 Introduction **3**

2 Problem Definition **6**
2.1 System 8
2.2 Observer 8

3 Minimization of Accelerations **10**
3.1 Derivation of the optimal control law 12
3.2 Necessary assumptions 17

4 Minimization of Jerkiness **22**
4.1 Transformation of the problem to the previous case 24
4.2 Necessary assumptions 26

5 Computational Performance Requirements **28**
5.1 Minimization of accelerations 28
5.2 Minimization of jerkiness 29
5.3 Possible limitations 30

6 Implementation with Matlab/Simulink **31**
6.1 Dryden model of gust 32
6.2 Airplane state space model 32
6.3 Controller 32
6.3.1 Minimization of accelerations 33
6.3.2 Minimization of jerkiness 35
6.3.3 Bounds of the controller output 37
6.3.4 Performance indices by means of H-infinity 40
6.4 Evaluation module 43
6.5 M-files 45

7 Preliminary Simulation **46**
7.1 Test configuration, system and actuator 46
7.2 Minimization of accelerations 49
7.3 Minimization of jerkiness 53
7.3.1 PT1 actuator 53
7.3.2 PT2 actuator 58

8 Application on a Wing-Fuselage System **63**
8.1 Test configuration, system and actuator 63
8.2 Minimization of accelerations 70
8.3 Minimization of jerkiness 76
8.4 Discussion of appropriate sensor positions 83

9 Final Remarks **85**
9.1 Conclusions 85
9.2 Acknowledgments 86

A Matlab M-Files **88**
A.1 *const_def.m* 88
A.2 *const_calc_1.m* 89
A.3 *const_calc_2.m* 91
A.4 *Hinf_calc.m* 92
A.5 *pzmap_calc.m* 94

Chapter 1

Introduction

The increasing competition in the market of civil aircraft leads to operating efficiency and passenger comfort being very important sales arguments. Continuous developments in jet propulsion technology helped to reduce energy consumption, as well as noise and vibrations due to the engines. The main problem with respect to ride comfort is, however, the transmittance of accelerations and jerkiness imposed by atmospheric turbulence from the wings to the fuselage. This "gust" is also a design constraint: Light airplane structures help to save energy, but are more critical to resist the loads imposed by turbulence. For both reasons, efficient gust alleviation is necessary to improve the performance of modern aircraft.

Gust can be seen as a change in the angle of attack or as an additional varying vertical component of the headwind. The effect of gust can be very strong, since the same aerodynamic forces that keep the airplane flying are involved. Event though the frequency range of those changes is quite low, it is impossible for the pilot to alleviate gust manually. Besides, most of the time during the flight, the autopilot maintains course and the attitude of flight. Certainly, most autopilots should be capable of damping the roughest parts of turbulence, but they are unable to provide satisfactory results in that field. A promising extension should be the application of subsidiary control, where the inner (faster) control loop alleviates turbulence and the outer (slower) loop controls the attitude of flight.

Besides the mentioned ride comfort, another reason for gust alleviation with respect to the fuselage is the sensibility of electrical devices to vibration and high values of acceleration. Many modern airplane designs – especially inherently instable military aircraft – are highly dependent on avionics. The life time and the reliability of these systems is thus essential. To give a vivid

example, I would like to refer to a paper on *Vibration Fatigue of Surface Mount Technology (SMT) Solder Joints* [1] by S. Liguore and D. Followell. According to the graphs provided by the authors, the number of cycles to failure can be estimated by $c \cdot e^{-p/m}$, where p is the vibration level in Grms and c, m are positive constants. Figure 1.1 shows that if the vibrations are reduced by 50% the life time will increase by a factor 100–1000.

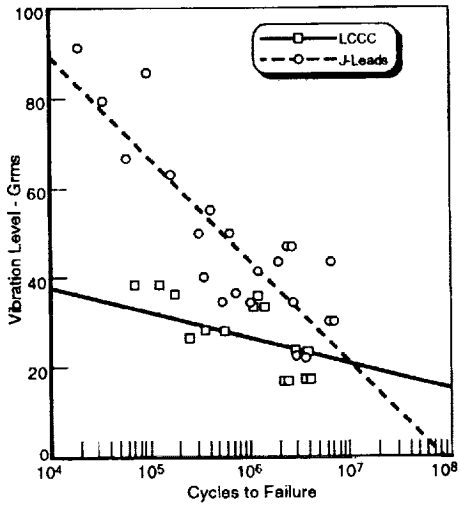

Figure 1.1: Fatigue Life Correlation (Graph taken from [1])

Many papers propose controller designs based on structural measurement, e.g. by accelerometers, strain gauges or – more sophisticated – PZTs to detect gust by its effect. Some results show an improvement, but still, without more information, they will not be satisfactory. A very promising approach is the use of direct gust measurement. Then, information about turbulence is available *before* it has a significant effect on the airplane. As a result, the control system has more time for efficient countermeasures, especially if the sensors are located ahead of the wings. Investigation in that field already took place in the 1960s, e.g. for the F-100 Rough Rider turbulence measurement system [2].

Since then, the available measurement devices have undergone further development from simple mechanical vanes, as used for the F-100 experiments, to sophisticated laser/lidar systems. I would like to mention the research work of O.S. Alvarez-Salazar and G.M. Wang, *A Novel Gust Monitoring Device* [3] which is based on the forward scattering of a laser beam, as well as the paper of D. Soreide, R.K. Bogue, L.J. Ehernberger and H. Bagley

on *Coherent Lidar Turbulence Measurement for Gust Load Alleviation* [4].

The reliability of these gust measurement data is essential for the performance of the alleviating system. Imprecise measurement can even make the controller worsen the effect of turbulence. Obviously, when the reading point is too far ahead of the wings, it is very likely that the turbulence changes between measurement and the encounter with the airplane. This problem is a matter of correlation time and will not be subject to my investigation. Besides, I believe that there is a practical borderline anyway, beyond which additional – even reliable – data have no further beneficial effect.

Although the motivation for my work is the gust alleviation problem, I could also imagine further applications beside airplanes. In the field of automotive electronics, an active undercarriage for a sport utility vehicle is imaginable: Small radar sensors implemented in the front bumper detect the ground conditions and "road holes" in front of the tires and adapt the wheel suspensions for better ride comfort.

Chapter 2

Problem Definition

In the course of the following, methods for controller design using data of direct gust measurement will be developed. Aside from a theoretical point of view the benefit of such measurement has to be weighed against the additional expenses incurred.
In order to find the upper bound of possible improvements, a "perfect" system is considered. If the results are not satisfactory in this case, they won't be applied to a real airplane (with uncertainties about the model) at all.

The following assumptions are made:

1. the whole system is exactly known, it can be described by linear differential equations
2. the differential equations for structure and aerodynamics are at least second order
3. the system is observable and controllable
4. perfect accuracy of the measurement devices, particularly of the gust sensors
5. no change of turbulence between measurement and encounter with the wings

My investigation will cover both situations:

1. sensor right at the location of the wing and its actuators (collocated system)

2. sensor ahead of the wings, e.g. abreast of the nose
   → information about future gust is available ca. 0.1s before encounter

The purpose of the controller is to reduce the effect of the gust encountering the wing by providing suitable actuator signals for countermeasures. As opposed to many designs minimizing the displacements, I will deal with the accelerations, as the displacements (and also the velocity) themselves have no noticeable effect on the "interior" of the fuselage. Thus, I will consider the accelerations for the cost function as a first approach. Then I will also seize the suggestion of A. Tewari [5] who proposes optimal control considering the time-rate change of normal acceleration. He regards the "passenger/crew comfort or weapons aiming and delivery considerations" to be sensitive especially to this "jerkiness of the motion". Both approaches seem applicable in view of electronic devices installed in the fuselage and the passenger comfort.

As the differential equations of aerodynamic and structure for the generalized displacements $\mathbf{w}$ are assumed to be at least second order, $\mathbf{x}$ can be defined as a state vector containing at least $\mathbf{w}$ and $\dot{\mathbf{w}}$. Therefore $\dot{\mathbf{x}}$ includes the acceleration $\ddot{\mathbf{w}}$. The time-rate change of normal acceleration (equals $\dddot{\mathbf{w}}$) is not that easily accessible in any case (at least if I want to avoid $\dot{\mathbf{z}}$) but a sufficient approximation is possible.

Hereafter I will make a discrete-time approach: The Riccati equation for a continuous-time control system is a serious problem, as there is no analytic solution for general system and weight matrices as well as general "gust functions". In discrete-time a closed solution for a general case is possible. Necessary assumptions for the existence of this solution will be discussed.

I will consider the location of the gust sensors variable to obtain a generally valid result. Moving the measurement device to the nose of the airplane enables the engineer to get information about future gust encounters. Of course, modern lidar systems allow measurement far ahead of the airplane increasing the interval of prediction, however, when going too far, the reliability of these data decreases due to jitter (uncertainty of time due to changes in aircraft speed etc.) and especially changes of turbulence before it encounters the airplane. For a theoretical analysis I consider the gust sensor data as absolutely reliable. I expect that the performance – starting from the measurement right at the wings – increases (the "cost" decrease) with the distance from the wings very fast at the beginning, but then will come to a saturation, since the knowledge of gust far in the future will certainly not have a remarkable beneficial effect on the control problem. I would like

to point out that – within this investigation – the expected behavior will not be due to the increasing uncertainty (as discussed above) of future gust data which I assume to be certain. However, for a real application the reliability of measurement is a constraint for the sensor position. The interval between measurement and encounter is τ, so that $\mathbf{z}(\bar{t})$ is known for $t \leq \bar{t} \leq t + \tau$.

2.1 System

The system can be described as a state space model in continuous-time. The matrices are assumed to be known exactly.

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{G}\mathbf{z} \tag{2.1}$$

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} \tag{2.2}$$

$\mathbf{z}(t)$ gust (velocity of vertical wind) directly at the wing. General case: Gust is represented by several reading points (where also the sensors are located) and a truly known shape function included in $\mathbf{G}$. For the easier case of (in spanwise direction) homogeneous gust, $\mathbf{z}$ is simply a scalar.

$\mathbf{y}(t)$ measurement vector of structural displacements (plunge, slope, torsion etc.)

$\mathbf{u}(t)$ controller output

2.2 Observer

In view of the assumption that the model and the process noise (gust) are perfectly known for current time as well as the fact that there is no measurement noise, a Kalman filter is not required. A standard Luenberger observer is applicable:

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} + \mathbf{G}\mathbf{z} + \mathbf{K}\left(\mathbf{y} - \mathbf{C}\hat{\mathbf{x}}\right)$$

If $(\mathbf{A} - \mathbf{KC})$ leads to a stable system with $\mathbf{K}$ sufficiently high, the initial error will approximate zero after a short time with the limit:

$$\lim_{t \to \infty} (\hat{\mathbf{x}} - \mathbf{x}) = \mathbf{0}$$

The observer will thus lead to almost perfect estimations soon after the beginning of the experiment. As in this case, a computer simulation is

processed, the state vector is accessible anyway. To keep the equations more simple and to focus on the controller design itself I leave out the observer (upper boundary of performance).

Chapter 3

Minimization of Accelerations

In view of the underlying differential equations, I assume that the system is provided for continuous-time in state space, that is the matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{G}$ are given. The aim is to minimize the time rate of acceleration of the systems, especially of the fuselage. I suppose that the sampling rate is sufficiently high, so that my goal is achieved by minimizing the acceleration at the sample times.
The current time is t_k, thus the purpose of the calculation is to find an optimal control output $\mathbf{u_k}$. To insure that all available information is considered for the control output, the whole calculation leading to $\mathbf{u_k}$ will be made for every (discrete) time t_k. The rest of the sequence of control outputs will not be used. State of the art signal processors should be able to provide the result without a remarkable delay. The discrete-time approach easily allows to keep the sensor position variable.

An optimal control approach for discrete-time systems is described in detail by K. Ogata [6] for a system without a (known) disturbance and using the state vector itself for the cost function. Due to these two differences and the fact that there are various cross terms within the cost function, I have to change the transformations resulting in another kind of Riccati equation.

Cost function:

$$J = \frac{1}{2}\mathbf{x_{k+N}^T S x_{k+N}} + \frac{1}{2}\sum_{i=k}^{k+N-1}\left[\dot{\mathbf{x}}_\mathbf{i}^\mathbf{T}\mathbf{Q_1}\dot{\mathbf{x}}_\mathbf{i} + \mathbf{u_i^T Q_2 u_i}\right] \qquad (3.1)$$

$$\mathbf{x_k} = \mathbf{x}(t)|_{t=t_k} \qquad \text{same notation for } \mathbf{u_k} \text{ and } \mathbf{z_k}$$

$$
\begin{aligned}
\dot{\mathbf{x}}_\mathbf{k} &= \dot{\mathbf{x}}(t)|_{t=t_k} = \Big(\mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{G}\mathbf{z}(t)\Big)\Big|_{t=t_k} \\
&= \mathbf{A}\mathbf{x}_\mathbf{k} + \mathbf{B}\mathbf{u}_\mathbf{k} + \mathbf{G}\mathbf{z}_\mathbf{k} \qquad (3.2) \\
\mathbf{x}_{\mathbf{k+1}} &= \mathbf{\Phi}\mathbf{x}_\mathbf{k} + \mathbf{\Gamma}\mathbf{u}_\mathbf{k} + \mathbf{\Theta}\mathbf{z}_\mathbf{k} \qquad (3.3) \\
\mathbf{\Phi} &= e^{\mathbf{A}T} \\
\mathbf{\Gamma} &= \int_0^T e^{\mathbf{A}\tau} d\tau\, \mathbf{B} = \mathbf{A}^{-1}\left[e^{\mathbf{A}T} - \mathbf{I}\right]\mathbf{B} \\
\mathbf{\Theta} &= \int_0^T e^{\mathbf{A}\tau} d\tau\, \mathbf{G} = \mathbf{A}^{-1}\left[e^{\mathbf{A}T} - \mathbf{I}\right]\mathbf{G} \\
T &= t_{k+1} - t_k \quad \forall k \geq 0 \qquad \text{sampling period}
\end{aligned}
$$

N, or to be precise NT, is the length of the considered finite horizon.

As mentioned above, I can assume that $\dot{\mathbf{x}}$ contains $\dot{\mathbf{w}}$ and $\ddot{\mathbf{w}}$ where $\dot{\mathbf{w}}$ represents the vector of velocities. With regard to my approach I do not want to consider velocities within my cost functions and would choose $\mathbf{Q_1}$ in such a way that only the terms dealing with $\ddot{\mathbf{w}}$ are unequal to zero. However, as for the following calculations, I will not go on such restrictions, especially not when discussing the necessary assumptions.

Depending on the location of the gust sensors and on the horizontal velocity of the airplane, I will have (for a fixed T) a different number n of gust data $\mathbf{z_k} \ldots \mathbf{z_{k+n-1}}$ available referring to the two cases as described in chapter 2:

1. gust sensor right at the location of the wing and its actuators, that is $n = 1$, no future gust data
2. gust sensor ahead of the wings. The larger $n > 1$ is, the more is the measurement device moved towards the nose. I assume that $n \leq N$

For the case of $n > 1$, it is necessary to know the time delay τ between measurement and effect of the turbulence. Therefore, I propose to use a sensor right at the wings in addition. Thus, τ – and finally n – can be calculated as the maximum of the correlation between the sensor signals. Each measured value can be written in a shift register of length N where the variable write position is n.

Of course, the question arises what values for the unknown gust $\mathbf{z_{k+n}} \ldots \mathbf{z_{k+N-1}}$ will be used within the cost function. Obviously, gust can be seen as a process noise (unbiased filtered white noise) for these times. For this additive-disturbance stochastic problem, the optimal control is identical to the deterministic case, that is $\mathbf{z_i} = \mathbf{0}$ for $i = k + n \ldots k + N - 1$ (certainty-equivalence principle) [7]. The reason why I do not just set $N = n$ is that

even if I don't have information about gust for these times, I have to consider the dynamic of the system to suppress accelerations (e.g. due to oscillations). I will derive the optimal control law for somehow known gust $\mathbf{z_k} \ldots \mathbf{z_{k+N-1}}$ and will later, when it comes to applying the controller, set the unknown parts to zero as described above. This method is possible since there are no partial derivatives with respect to $\mathbf{z_i}$ involved hereafter. In doing so I can avoid unpractical fall differentiations. The current state vector $\mathbf{x_k}$ is known.

3.1 Derivation of the optimal control law

Using the Lagrange's method, equation (3.3) provides the necessary constraints:

$$\begin{aligned} L &= \frac{1}{2}\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S}\mathbf{x}_{\mathbf{k+N}} + \frac{1}{2}\sum_{i=k}^{k+N-1}\Big[\dot{\mathbf{x}}_{\mathbf{i}}^{\mathbf{T}}\mathbf{Q_1}\dot{\mathbf{x}}_{\mathbf{i}} + \\ &\quad + \mathbf{u}_{\mathbf{i}}^{\mathbf{T}}\mathbf{Q_2}\mathbf{u_i} + \boldsymbol{\lambda}_{\mathbf{i+1}}^{\mathbf{T}}\left(-\mathbf{x_{i+1}} + \boldsymbol{\Phi}\mathbf{x_i} + \boldsymbol{\Gamma}\mathbf{u_i} + \boldsymbol{\Theta}\mathbf{z_i}\right)\Big] \qquad (3.4) \\ L &= \frac{1}{2}\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S}\mathbf{x}_{\mathbf{k+N}} + \\ &\quad + \frac{1}{2}\sum_{i=k}^{k+N-1}\Big[\left(\mathbf{A}\mathbf{x_i} + \mathbf{B}\mathbf{u_i} + \mathbf{G}\mathbf{z_i}\right)^T\mathbf{Q_1}\left(\mathbf{A}\mathbf{x_i} + \mathbf{B}\mathbf{u_i} + \mathbf{G}\mathbf{z_i}\right) + \\ &\quad + \mathbf{u}_{\mathbf{i}}^{\mathbf{T}}\mathbf{Q_2}\mathbf{u_i} + \boldsymbol{\lambda}_{\mathbf{i+1}}^{\mathbf{T}}\left(-\mathbf{x_{i+1}} + \boldsymbol{\Phi}\mathbf{x_i} + \boldsymbol{\Gamma}\mathbf{u_i} + \boldsymbol{\Theta}\mathbf{z_i}\right)\Big] \qquad (3.5) \\ L &= \frac{1}{2}\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S}\mathbf{x}_{\mathbf{k+N}} + \\ &\quad + \frac{1}{2}\sum_{i=k}^{k+N-1}\Big[\mathbf{x}_{\mathbf{i}}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}\mathbf{Q_1}\mathbf{A}\mathbf{x_i} + \mathbf{x}_{\mathbf{i}}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}\mathbf{Q_1}\mathbf{B}\mathbf{u_i} + \mathbf{x}_{\mathbf{i}}^{\mathbf{T}}\mathbf{A}^{\mathbf{T}}\mathbf{Q_1}\mathbf{G}\mathbf{z_i} + \\ &\quad + \mathbf{u}_{\mathbf{i}}^{\mathbf{T}}\mathbf{B}^{\mathbf{T}}\mathbf{Q_1}\mathbf{A}\mathbf{x_i} + \mathbf{u}_{\mathbf{i}}^{\mathbf{T}}\mathbf{B}^{\mathbf{T}}\mathbf{Q_1}\mathbf{B}\mathbf{u_i} + \mathbf{u}_{\mathbf{i}}^{\mathbf{T}}\mathbf{B}^{\mathbf{T}}\mathbf{Q_1}\mathbf{G}\mathbf{z_i} + \\ &\quad + \mathbf{z}_{\mathbf{i}}^{\mathbf{T}}\mathbf{G}^{\mathbf{T}}\mathbf{Q_1}\mathbf{A}\mathbf{x_i} + \mathbf{z}_{\mathbf{i}}^{\mathbf{T}}\mathbf{G}^{\mathbf{T}}\mathbf{Q_1}\mathbf{B}\mathbf{u_i} + \mathbf{z}_{\mathbf{i}}^{\mathbf{T}}\mathbf{G}^{\mathbf{T}}\mathbf{Q_1}\mathbf{G}\mathbf{z_i} + \mathbf{u}_{\mathbf{i}}^{\mathbf{T}}\mathbf{Q_2}\mathbf{u_i} + \\ &\quad + \boldsymbol{\lambda}_{\mathbf{i+1}}^{\mathbf{T}}\left(-\mathbf{x_{i+1}} + \boldsymbol{\Phi}\mathbf{x_i} + \boldsymbol{\Gamma}\mathbf{u_i} + \boldsymbol{\Theta}\mathbf{z_i}\right)\Big] \\ L &= \frac{1}{2}\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S}\mathbf{x}_{\mathbf{k+N}} + \frac{1}{2}\sum_{i=k}^{k+N-1}\Big[\begin{pmatrix}\mathbf{x}_{\mathbf{i}}^{\mathbf{T}} & \mathbf{u}_{\mathbf{i}}^{\mathbf{T}} & \mathbf{z}_{\mathbf{i}}^{\mathbf{T}}\end{pmatrix}\cdot \\ &\quad \cdot\begin{pmatrix}\mathbf{A}^{\mathbf{T}}\mathbf{Q_1}\mathbf{A} & \mathbf{A}^{\mathbf{T}}\mathbf{Q_1}\mathbf{B} & \mathbf{A}^{\mathbf{T}}\mathbf{Q_1}\mathbf{G} \\ \mathbf{B}^{\mathbf{T}}\mathbf{Q_1}\mathbf{A} & \mathbf{B}^{\mathbf{T}}\mathbf{Q_1}\mathbf{B} + \mathbf{Q_2} & \mathbf{B}^{\mathbf{T}}\mathbf{Q_1}\mathbf{G} \\ \mathbf{G}^{\mathbf{T}}\mathbf{Q_1}\mathbf{A} & \mathbf{G}^{\mathbf{T}}\mathbf{Q_1}\mathbf{B} & \mathbf{G}^{\mathbf{T}}\mathbf{Q_1}\mathbf{G}\end{pmatrix}\begin{pmatrix}\mathbf{x_i} \\ \mathbf{u_i} \\ \mathbf{z_i}\end{pmatrix} + \\ &\quad + \boldsymbol{\lambda}_{\mathbf{i+1}}^{\mathbf{T}}\left(-\mathbf{x_{i+1}} + \boldsymbol{\Phi}\mathbf{x_i} + \boldsymbol{\Gamma}\mathbf{u_i} + \boldsymbol{\Theta}\mathbf{z_i}\right)\Big] \end{aligned}$$

Using these substitutions:

$$
\begin{aligned}
L \;=\; & \frac{1}{2}\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S}\mathbf{x}_{\mathbf{k+N}} + \\
& + \frac{1}{2}\sum_{i=k}^{k+N-1}\Big[\begin{pmatrix} \mathbf{x}_i^{\mathbf{T}} & \mathbf{u}_i^{\mathbf{T}} & \mathbf{z}_i^{\mathbf{T}} \end{pmatrix}\begin{pmatrix} \mathbf{S_{11}} & \mathbf{S_{12}} & \mathbf{S_{13}} \\ \mathbf{S_{21}} & \mathbf{S_{22}} & \mathbf{S_{23}} \\ \mathbf{S_{31}} & \mathbf{S_{32}} & \mathbf{S_{33}} \end{pmatrix}\begin{pmatrix} \mathbf{x}_i \\ \mathbf{u}_i \\ \mathbf{z}_i \end{pmatrix} + \\
& + \boldsymbol{\lambda}_{\mathbf{i+1}}^{\mathbf{T}}\left(-\mathbf{x_{i+1}} + \boldsymbol{\Phi}\mathbf{x}_i + \boldsymbol{\Gamma}\mathbf{u}_i + \boldsymbol{\Theta}\mathbf{z}_i\right)\Big]
\end{aligned}
$$

Note: $\mathbf{Q_1} = \mathbf{Q_1^T}$ and $\mathbf{Q_2} = \mathbf{Q_2^T}$, thus $\mathbf{S_{ij}^T} = \mathbf{S_{ji}}\ \forall i,j \quad \mathbf{S^T} = \mathbf{S}$
$\mathbf{Q_1}$, $\mathbf{Q_2}$ and $\mathbf{S}$ are real symmetric weight matrices.

The necessary conditions for L to have a minimum read:

$$\frac{\delta L}{\delta \mathbf{x}_i} \stackrel{!}{=} \mathbf{0} \;:\; \mathbf{x}_i^{\mathbf{T}}\mathbf{S_{11}} + \mathbf{u}_i^{\mathbf{T}}\mathbf{S_{21}} + \mathbf{z}_i^{\mathbf{T}}\mathbf{S_{31}} + \boldsymbol{\lambda}_{\mathbf{i+1}}^{\mathbf{T}}\boldsymbol{\Phi} - \boldsymbol{\lambda}_{\mathbf{i}}^{\mathbf{T}} = \mathbf{0} \qquad (3.6)$$

$$\text{for } i = k+1 \ldots k+N-1$$

$$\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S} - \boldsymbol{\lambda}_{\mathbf{k+N}}^{\mathbf{T}} = \mathbf{0} \qquad (3.7)$$

$$\text{for } i = k+N$$

$$\frac{\delta L}{\delta \mathbf{u}_i} \stackrel{!}{=} \mathbf{0} \;:\; \mathbf{x}_i^{\mathbf{T}}\mathbf{S_{12}} + \mathbf{u}_i^{\mathbf{T}}\mathbf{S_{22}} + \mathbf{z}_i^{\mathbf{T}}\mathbf{S_{32}} + \boldsymbol{\lambda}_{\mathbf{i+1}}^{\mathbf{T}}\boldsymbol{\Gamma} = \mathbf{0} \qquad (3.8)$$

$$\text{for } i = k \ldots k+N-1$$

$$\frac{\delta L}{\delta \boldsymbol{\lambda}_i} \stackrel{!}{=} \mathbf{0} \;:\; -\mathbf{x}_i + \boldsymbol{\Phi}\mathbf{x_{i-1}} + \boldsymbol{\Gamma}\mathbf{u_{i-1}} + \boldsymbol{\Theta}\mathbf{z_{i-1}} = \mathbf{0} \qquad (3.9)$$

$$\text{for } i = k+1 \ldots k+N$$

Transformations that will be used later:

$$\boldsymbol{\lambda}_i = \boldsymbol{\Phi}^{\mathbf{T}}\boldsymbol{\lambda}_{\mathbf{i+1}} + \mathbf{S_{13}}\mathbf{z}_i + \mathbf{S_{12}}\mathbf{u}_i + \mathbf{S_{11}}\mathbf{x}_i \qquad (3.10)$$

$$\text{for } i = k+1 \ldots k+N-1$$

$$\mathbf{x_{i+1}} = \boldsymbol{\Phi}\mathbf{x}_i + \boldsymbol{\Gamma}\mathbf{u}_i + \boldsymbol{\Theta}\mathbf{z}_i \qquad (3.11)$$

The set of equations represents a two-point boundary-value problem: $\mathbf{x_k}$ is known for first boundary and (3.7) provides information about the last. In order to solve this problem a kind of Riccati transformation is applicable. I would like to point out again that the Riccati equation for the discrete-time case is much easier to handle than the one for continuous-time and a closed solution is possible. Assuming that $\boldsymbol{\lambda}_i$ can be written as follows:

$$\boldsymbol{\lambda}_i = \mathbf{P}_i\mathbf{x}_i + \mathbf{p}_i \qquad \text{where } \mathbf{P_i^T} = \mathbf{P}_i \qquad (3.12)$$

Taking (3.6) as my starting point:

$$\begin{aligned}
\mathbf{0} &= \mathbf{x}_i^T\mathbf{S}_{11} + \mathbf{u}_i^T\mathbf{S}_{21} + \mathbf{z}_i^T\mathbf{S}_{31} + \boldsymbol{\lambda}_{i+1}^T\boldsymbol{\Phi} - \boldsymbol{\lambda}_i^T \\
\mathbf{0} &= \mathbf{x}_i^T\mathbf{S}_{11}\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} + \mathbf{u}_i^T\mathbf{S}_{21}\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} + \mathbf{z}_i^T\mathbf{S}_{31}\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} + \\
&\quad + \underbrace{\boldsymbol{\lambda}_{i+1}^T\boldsymbol{\Gamma}}_{-\mathbf{x}_i^T\mathbf{S}_{12}-\mathbf{u}_i^T\mathbf{S}_{22}-\mathbf{z}_i^T\mathbf{S}_{32}} - \boldsymbol{\lambda}_i^T\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} \\
\boldsymbol{\lambda}_i^T\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} &= \mathbf{x}_i^T\left(\mathbf{S}_{11}\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} - \mathbf{S}_{12}\right) + \mathbf{u}_i^T\left(\mathbf{S}_{21}\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} - \mathbf{S}_{22}\right) + \\
&\quad + \mathbf{z}_i^T\left(\mathbf{S}_{31}\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} - \mathbf{S}_{32}\right) \\
&\quad \text{for } i = k+1 \ldots k+N-1
\end{aligned}$$

and thus:

$$\begin{aligned}
\mathbf{u}_i^T &= \boldsymbol{\lambda}_i^T\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma}\left(\mathbf{S}_{21}\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} - \mathbf{S}_{22}\right)^{-1} - \\
&\quad - \mathbf{x}_i^T\left(\mathbf{S}_{11}\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} - \mathbf{S}_{12}\right)\left(\mathbf{S}_{21}\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} - \mathbf{S}_{22}\right)^{-1} - \\
&\quad - \mathbf{z}_i^T\left(\mathbf{S}_{31}\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} - \mathbf{S}_{32}\right)\left(\mathbf{S}_{21}\boldsymbol{\Phi}^{-1}\boldsymbol{\Gamma} - \mathbf{S}_{22}\right)^{-1} \\
\mathbf{u}_i &= \left(\boldsymbol{\Gamma}^T\boldsymbol{\Phi}^{-T}\mathbf{S}_{12} - \mathbf{S}_{22}\right)^{-1}\boldsymbol{\Gamma}^T\boldsymbol{\Phi}^{-T}\boldsymbol{\lambda}_i - \\
&\quad - \left(\boldsymbol{\Gamma}^T\boldsymbol{\Phi}^{-T}\mathbf{S}_{12} - \mathbf{S}_{22}\right)^{-1}\left(\boldsymbol{\Gamma}^T\boldsymbol{\Phi}^{-T}\mathbf{S}_{11} - \mathbf{S}_{21}\right)\mathbf{x}_i - \\
&\quad - \underbrace{\left(\boldsymbol{\Gamma}^T\boldsymbol{\Phi}^{-T}\mathbf{S}_{12} - \mathbf{S}_{22}\right)^{-1}}_{\mathbf{M}_{inv}}\left(\boldsymbol{\Gamma}^T\boldsymbol{\Phi}^{-T}\mathbf{S}_{13} - \mathbf{S}_{23}\right)\mathbf{z}_i \\
\mathbf{u}_i &= \mathbf{M}_{inv}\boldsymbol{\Gamma}^T\boldsymbol{\Phi}^{-T}\boldsymbol{\lambda}_i^T - \mathbf{M}_{inv}\left(\boldsymbol{\Gamma}^T\boldsymbol{\Phi}^{-T}\mathbf{S}_{11} - \mathbf{S}_{21}\right)\mathbf{x}_i - \\
&\quad - \mathbf{M}_{inv}\left(\boldsymbol{\Gamma}^T\boldsymbol{\Phi}^{-T}\mathbf{S}_{13} - \mathbf{S}_{23}\right)\mathbf{z}_i \qquad (3.13) \\
&\quad \text{for } i = k+1 \ldots k+N-1
\end{aligned}$$

Substitution of (3.12) into (3.10):

$$\begin{aligned}
\mathbf{P}_i\mathbf{x}_i + \mathbf{p}_i &= \boldsymbol{\Phi}^T\mathbf{P}_{i+1}\underbrace{\mathbf{x}_{i+1}}_{\boldsymbol{\Phi}\mathbf{x}_i+\boldsymbol{\Gamma}\mathbf{u}_i+\boldsymbol{\Theta}\mathbf{z}_i} + \boldsymbol{\Phi}^T\mathbf{p}_{i+1} + \mathbf{S}_{13}\mathbf{z}_i + \mathbf{S}_{12}\mathbf{u}_i + \mathbf{S}_{11}\mathbf{x}_i \\
\mathbf{P}_i\mathbf{x}_i + \mathbf{p}_i &= \left(\boldsymbol{\Phi}^T\mathbf{P}_{i+1}\boldsymbol{\Phi} + \mathbf{S}_{11}\right)\mathbf{x}_i + \left(\boldsymbol{\Phi}^T\mathbf{P}_{i+1}\boldsymbol{\Theta} + \mathbf{S}_{13}\right)\mathbf{z}_i + \\
&\quad + \boldsymbol{\Phi}^T\mathbf{p}_{i+1} + \left(\boldsymbol{\Phi}^T\mathbf{P}_{i+1}\boldsymbol{\Gamma} + \mathbf{S}_{12}\right)\mathbf{u}_i \qquad (3.14) \\
&\quad \text{for } i = k+1 \ldots k+N-1
\end{aligned}$$

Substitution of (3.13) into (3.14):

$$\mathbf{P}_i\mathbf{x}_i + \mathbf{p}_i = \Big[\boldsymbol{\Phi}^T\mathbf{P}_{i+1}\boldsymbol{\Phi} + \mathbf{S}_{11} -$$

$$
\begin{aligned}
&- \left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Gamma} + \mathbf{S}_{12}\right)\mathbf{M}_{\mathbf{inv}}\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S}_{11} - \mathbf{S}_{21}\right)\Big]\,\mathbf{x}_i + \\
&+ \Big[\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Theta} + \mathbf{S}_{13} - \\
&- \left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Gamma} + \mathbf{S}_{12}\right)\mathbf{M}_{\mathbf{inv}}\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S}_{13} - \mathbf{S}_{23}\right)\Big]\,\mathbf{z}_i + \\
&+ \mathbf{\Phi}^{\mathbf{T}}\mathbf{p}_{i+1} + \left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Gamma} + \mathbf{S}_{12}\right)\mathbf{M}_{\mathbf{inv}}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\left(\mathbf{P}_i\mathbf{x}_i + \mathbf{p}_i\right)
\end{aligned}
$$

$$
\begin{aligned}
&\Big[-\mathbf{P}_i + \mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Phi} + \mathbf{S}_{11} - \left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Gamma} + \mathbf{S}_{12}\right)\mathbf{M}_{\mathbf{inv}}\cdot \\
&\quad\cdot\left(\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S}_{11} - \mathbf{S}_{21}\right) - \mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{P}_i\right)\Big]\,\mathbf{x}_i + \\
&\quad+ \Big[\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Theta} + \mathbf{S}_{13} - \left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Gamma} + \mathbf{S}_{12}\right)\mathbf{M}_{\mathbf{inv}}\cdot \\
&\quad\cdot\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S}_{13} - \mathbf{S}_{23}\right)\Big]\,\mathbf{z}_i + \mathbf{\Phi}^{\mathbf{T}}\mathbf{p}_{i+1} + \\
&\quad+ \Big[\left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Gamma} + \mathbf{S}_{12}\right)\mathbf{M}_{\mathbf{inv}}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}} - \mathbf{I}\Big]\,\mathbf{p}_i = \mathbf{0} \qquad (3.15) \\
&\quad\text{for } i = k+1 \ldots k+N-1
\end{aligned}
$$

Equation (3.15) must be satisfied for all $\mathbf{x}_i$, $i = k+1 \ldots k+N-1$
$\Rightarrow$ set term in first squared brackets to zero:

$$
\begin{aligned}
&-\mathbf{P}_i + \mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Phi} + \mathbf{S}_{11} + \mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Gamma}\mathbf{M}_{\mathbf{inv}}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{P}_i - \\
&\quad- \mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Gamma}\mathbf{M}_{\mathbf{inv}}\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S}_{11} - \mathbf{S}_{21}\right) + \\
&\quad+ \mathbf{S}_{12}\mathbf{M}_{\mathbf{inv}}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{P}_i - \mathbf{S}_{12}\mathbf{M}_{\mathbf{inv}}\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S}_{11} - \mathbf{S}_{21}\right) \stackrel{!}{=} \mathbf{0} \\
&\Big[\mathbf{I} - \left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Gamma} + \mathbf{S}_{12}\right)\mathbf{M}_{\mathbf{inv}}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\Big]\,\mathbf{P}_i = \\
&\quad\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\left(\mathbf{\Phi} - \mathbf{\Gamma}\mathbf{M}_{\mathbf{inv}}\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S}_{11} - \mathbf{S}_{21}\right)\right) + \mathbf{S}_{11} - \\
&\quad- \mathbf{S}_{12}\mathbf{M}_{\mathbf{inv}}\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S}_{11} - \mathbf{S}_{21}\right)
\end{aligned}
$$

$$
\begin{aligned}
&\mathbf{P}_i = \Big[\mathbf{I} - \left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\mathbf{\Gamma} + \mathbf{S}_{12}\right)\mathbf{M}_{\mathbf{inv}}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\Big]^{-1}\cdot \\
&\quad\cdot\Big[\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{i+1}\left(\mathbf{\Phi} - \mathbf{\Gamma}\mathbf{M}_{\mathbf{inv}}\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S}_{11} - \mathbf{S}_{21}\right)\right) + \mathbf{S}_{11} - \\
&\quad- \mathbf{S}_{12}\mathbf{M}_{\mathbf{inv}}\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S}_{11} - \mathbf{S}_{21}\right)\Big] \qquad (3.16) \\
&\quad\text{for } i = k+1 \ldots k+N-1
\end{aligned}
$$

Equation (3.16) is a kind of Riccati equation, that allows a backward calculation of $\mathbf{P}_i$ for $i = k+N-1 \ldots k+1$. The necessary final value $\mathbf{P}_{k+N}$ can be obtained by substituting (3.12) into (3.7):

$$\mathbf{P}_{k+N}\mathbf{x}_{k+N} + \mathbf{p}_{k+N} = \mathbf{S}\mathbf{x}_{k+N}$$

Since there is no explicit constraint for $\mathbf{x_{k+N}}$ (this is not applicable in view of a permanent disturbance by the gust), this equation must be satisfied for all $\mathbf{x_{k+N}}$. As a result:

$$\mathbf{P_{k+N}} = \mathbf{S} \tag{3.17}$$

$$\mathbf{p_{k+N}} = \mathbf{0} \tag{3.18}$$

Now, $\mathbf{P_{k+N}} \dots \mathbf{P_{k+1}}$ are known and can be used to calculate $\mathbf{p_{k+N-1}} \cdots \mathbf{p_{k+1}}$. For this reason, I would like to remind of (3.15). The last two rows must equal zero as well:

$$\begin{aligned}\left[\mathbf{\Phi}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Theta} + \mathbf{S_{13}} - \left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Gamma} + \mathbf{S_{12}}\right)\mathbf{M_{inv}}\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S_{13}} - \mathbf{S_{23}}\right)\right]\mathbf{z_i} + \\ + \mathbf{\Phi}^{\mathbf{T}}\mathbf{p_{i+1}} + \left[\left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Gamma} + \mathbf{S_{12}}\right)\mathbf{M_{inv}}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}} - \mathbf{I}\right]\mathbf{p_i} = \mathbf{0}\end{aligned}$$

Solving for $\mathbf{p_i}$:

$$\begin{aligned}\mathbf{p_i} = &\left[\mathbf{I} - \left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Gamma} + \mathbf{S_{12}}\right)\mathbf{M_{inv}}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\right]^{-1} \cdot \\ &\cdot \left[\mathbf{\Phi}^{\mathbf{T}}\mathbf{p_{i+1}} + \left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Theta} + \mathbf{S_{13}} - \right.\right. \\ &\left.\left. - \left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Gamma} + \mathbf{S_{12}}\right)\mathbf{M_{inv}}\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S_{13}} - \mathbf{S_{23}}\right)\right)\mathbf{z_i}\right] \\ &\text{for } i = k+1 \dots k+N-1\end{aligned} \tag{3.19}$$

As $\mathbf{z_k} \dots \mathbf{z_{k+N-1}}$ are (somehow) known, $\mathbf{p_{k+N-1}} \cdots \mathbf{p_{k+1}}$ are available. When it comes to calculating the required control output $\mathbf{u_k}$, equation (3.13) is not applicable, since it only holds for $i = k+1 \dots k+N-1$. Therefore I refer to (3.8):

$$\begin{aligned}
\mathbf{0} &= \mathbf{x_i^T}\mathbf{S_{12}} + \mathbf{u_i^T}\mathbf{S_{22}} + \mathbf{z_i^T}\mathbf{S_{32}} + \mathbf{\lambda_{i+1}^T}\mathbf{\Gamma} \\
\mathbf{0} &= \mathbf{x_i^T}\mathbf{S_{12}} + \mathbf{u_i^T}\mathbf{S_{22}} + \mathbf{z_i^T}\mathbf{S_{32}} + \left(\mathbf{x_{i+1}^T}\mathbf{P_{i+1}} + \mathbf{p_{i+1}^T}\right)\mathbf{\Gamma} \\
\mathbf{0} &= \mathbf{x_i^T}\mathbf{S_{12}} + \mathbf{u_i^T}\mathbf{S_{22}} + \mathbf{z_i^T}\mathbf{S_{32}} + \left(\left(\mathbf{\Phi}\mathbf{x_i} + \mathbf{\Gamma}\mathbf{u_i} + \mathbf{\Theta}\mathbf{z_i}\right)^{\mathbf{T}}\mathbf{P_{i+1}} + \mathbf{p_{i+1}^T}\right)\mathbf{\Gamma} \\
\mathbf{0} &= \mathbf{x_i^T}\left(\mathbf{S_{12}} + \mathbf{\Phi}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Gamma}\right) + \mathbf{u_i^T}\left(\mathbf{S_{22}} + \mathbf{\Gamma}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Gamma}\right) + \\
&\quad + \mathbf{z_i^T}\left(\mathbf{S_{32}} + \mathbf{\Theta}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Gamma}\right) + \mathbf{p_{i+1}^T}\mathbf{\Gamma} \\
\mathbf{0} &= \left(\mathbf{S_{21}} + \mathbf{\Gamma}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Phi}\right)\mathbf{x_i} + \left(\mathbf{S_{22}} + \mathbf{\Gamma}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Gamma}\right)\mathbf{u_i} + \\
&\quad + \left(\mathbf{S_{23}} + \mathbf{\Gamma}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Theta}\right)\mathbf{z_i} + \mathbf{\Gamma}^{\mathbf{T}}\mathbf{p_{i+1}} \\
&\quad \text{for } i = k \dots k+N-1 \\
\mathbf{u_i} &= -\left(\mathbf{S_{22}} + \mathbf{\Gamma}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Gamma}\right)^{-1}\left[\left(\mathbf{S_{21}} + \mathbf{\Gamma}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Phi}\right)\mathbf{x_i} + \right. \\
&\quad \left. + \left(\mathbf{S_{23}} + \mathbf{\Gamma}^{\mathbf{T}}\mathbf{P_{i+1}}\mathbf{\Theta}\right)\mathbf{z_i} + \mathbf{\Gamma}^{\mathbf{T}}\mathbf{p_{i+1}}\right]
\end{aligned} \tag{3.20}$$

Finally the optimal control law reads:

$$\begin{aligned}\mathbf{u_k} &= -\left(\mathbf{S_{22}}+\mathbf{\Gamma^T P_{k+1}\Gamma}\right)^{-1}\left[\left(\mathbf{S_{21}}+\mathbf{\Gamma^T P_{k+1}\Phi}\right)\mathbf{x_k}+\right.\\ &\left.+\left(\mathbf{S_{23}}+\mathbf{\Gamma^T P_{k+1}\Theta}\right)\mathbf{z_k}+\mathbf{\Gamma^T p_{k+1}}\right] \qquad (3.21)\end{aligned}$$

To compare the performance of the gust alleviation system a "random" gust function, or to be precise a sequence of data, should be previously recorded for a period of time to guarantee equal conditions for the tests. Then, for different values of n and also of $\mathbf{Q_1}$, $\mathbf{Q_2}$ and $\mathbf{S}$ a given system (represented by the matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{G}$) will be controlled. In order to obtain comparable results I introduce a set of cost functions:

$$J_1 = \int_{t_0}^{t_e} \dot{\mathbf{x}}^{\mathbf{T}}(t)\mathbf{RC}\,\dot{\mathbf{x}}(t)dt \qquad \text{ride comfort} \qquad (3.22)$$

$$J_2 = \int_{t_0}^{t_e} \mathbf{x}^{\mathbf{T}}(t)\mathbf{AD}\,\mathbf{x}(t)dt \qquad \text{"aggregated" displacements} \qquad (3.23)$$

$$J_3 = \int_{t_0}^{t_e} \mathbf{u}^{\mathbf{T}}(t)\mathbf{CE}\,\mathbf{u}(t)dt \qquad \text{control energy} \qquad (3.24)$$

The state vectors used for these functions refer to the case of the controller applied on the continuous-time model as described in (2.1). This is closer to a real application, in which case the accelerations between t_i and t_{i+1} have to be taken into account as well. I would like to point out that $\mathbf{Q_1}$, $\mathbf{Q_2}$ and $\mathbf{S}$ are not applicable to J_1, J_2 and J_3 as they only establish priorities within the controller design and may very. For comparable results, the weight matrices need to be constant throughout the tests.

3.2 Necessary assumptions

The above-mentioned approach is only applicable if and only if all of the following matrices are nonsingular:

A

$\mathbf{A^{-1}}$ exists if and only if $\det(\mathbf{A}) \neq 0$. In view of an observable and controllable system this should be the case. At least if I assume that the system was already stabilized by a slower feedback controller (the main purpose of the controller designed above is the minimization of accelerations) the eigenvalues of $\mathbf{A}$ can then be assumed to be all negative. As a result, $\mathbf{A}$ would be positive definite and thus nonsingular.

$\mathbf{S_{22}}$

$$\mathbf{S_{22}} = \mathbf{B^T Q_1 B} + \mathbf{Q_2}$$

If $(\mathbf{B^T Q_1 B} + \mathbf{Q_2})$ is a positive definite matrix then its determinant is unequal to zero.
If a matrix $\hat{\mathbf{G}}$ is positive definite, that is $J = \mathbf{v^T \hat{G} v} \geq 0 \, \forall \, \mathbf{v}$ and $J = 0$ only for the case of $\mathbf{v} = \mathbf{0}$, then $J = (\mathbf{Ty})^{\mathbf{T}} \hat{\mathbf{G}} (\mathbf{Ty}) = \mathbf{y^T (T^T \hat{G} T) y} = \mathbf{y^T \bar{G} y}$ is a positive definite form, too.
If $\hat{\mathbf{A}}$ is positive definite and $\hat{\mathbf{B}}$ is at least positive semidefinite, then $\mathbf{v^T \hat{A} v} + \mathbf{v^T \hat{B} v} = \mathbf{v^T} (\hat{\mathbf{A}} + \hat{\mathbf{B}}) \mathbf{v}$ is a positive definite form, so that $(\hat{\mathbf{A}} + \hat{\mathbf{B}})$ is a positive definite matrix.
If the terms of $\mathbf{Q_1}$ that refer to nonzero parts of $\mathbf{B}$ are at least positive semidefinite then $\mathbf{B^T Q_1 B}$ will be, too. Assuming that $\mathbf{Q_2}$ is positive definite, $\mathbf{S_{22}^{-1}}$ exists.

$\mathbf{\Phi}$

$$\mathbf{\Phi} = e^{\mathbf{A}T} = \sum_{k=0}^{\infty} \frac{(\mathbf{A}T)^k}{k!}$$

It can be shown [6] that the sum converges absolutely for all finite T, so that $\mathbf{\Phi}$ exists. Therefore, the discrete time approach is possible for any $\mathbf{A}$
According to [6], $e^{\mathbf{A}T}$ is nonsingular and $\mathbf{\Phi^{-1}} = e^{-\mathbf{A}T}$.

$(\mathbf{S_{21} \Phi^{-1} \Gamma - S_{22}})$
I have to prove that the determinant of $(\mathbf{S_{21} \Phi^{-1} \Gamma - S_{22}})$ is unequal to zero. I will use $|\ |$ equivalent to det().

$$|\mathbf{S_{21} \Phi^{-1} \Gamma - S_{22}}| \overset{!}{\neq} 0$$

$$\left|\mathbf{B^T Q_1 \; A \Phi^{-1} A^{-1} \; [\Phi - I] B - B^T Q_1 B - Q_2}\right| \overset{!}{\neq} 0$$

Interim calculation:

$$\mathbf{A \Phi^{-1} A^{-1}} = \mathbf{A} \left[\mathbf{I} + \mathbf{A}(-T) + \frac{\mathbf{A^2}(-T)^2}{2!} + \frac{\mathbf{A^3}(-T)^3}{3!} + \cdots\right] \mathbf{A^{-1}}$$

$$
\begin{aligned}
&= \left[\mathbf{A} + \mathbf{A^2}(-T) + \frac{\mathbf{A^3}(-T)^2}{2!} + \frac{\mathbf{A^4}(-T)^3}{3!} + \cdots\right]\mathbf{A^{-1}} \\
&= \left[\mathbf{I} + \mathbf{A}(-T) + \frac{\mathbf{A^2}(-T)^2}{2!} + \frac{\mathbf{A^3}(-T)^3}{3!} + \cdots\right] \\
&= \mathbf{\Phi^{-1}}
\end{aligned}
$$

The simplified equation reads:

$$
\begin{aligned}
\left|\mathbf{B^T Q_1 \Phi^{-1}}\left[\mathbf{\Phi} - \mathbf{I}\right]\mathbf{B} - \mathbf{B^T Q_1 B} - \mathbf{Q_2}\right| &\overset{!}{\neq} 0 \\
\left|\mathbf{B^T Q_1 B} - \mathbf{B^T Q_1 \Phi^{-1} B} - \mathbf{B^T Q_1 B} - \mathbf{Q_2}\right| &\overset{!}{\neq} 0 \\
\left|\mathbf{B^T Q_1 \Phi^{-1} B} + \mathbf{Q_2}\right| &\overset{!}{\neq} 0
\end{aligned}
$$

If $(\mathbf{B^T Q_1 \Phi^{-1} B} + \mathbf{Q_2})$ is a positive definite matrix its determinant is unequal to zero. This is the case if one addend is positive definite and the other one is at least positive semidefinite. $\mathbf{Q_2}$ is assumed to be positive definite (see $\mathbf{S_{22}}$).
If $(\mathbf{Q_1 \Phi^{-1}})$ is a positive definite matrix, $(\mathbf{B^T Q_1 \Phi^{-1} B})$ will be, too, so that the sum is positive definite. However, I think the criteria of positive definite property is too restrictive in this case.
Simplifying the expression:

$$
\begin{aligned}
\left|\mathbf{B^T Q_1 \Phi^{-1} B} + \mathbf{Q_2}\right| &\overset{!}{\neq} 0 \\
|\mathbf{Q_2}|\left|\mathbf{I} + \mathbf{Q_2^{-1} B^T Q_1 \Phi^{-1} B}\right| &\overset{!}{\neq} 0
\end{aligned}
$$

$|\mathbf{Q_2}| \neq 0$ as $\mathbf{Q_2}$ is positive definite, so that the resulting condition reads:

$$
\left|\mathbf{I} + \mathbf{Q_2^{-1} B^T Q_1 \Phi^{-1} B}\right| \overset{!}{\neq} 0
$$

Using the following relationship:

$$
\begin{aligned}
|\mathbf{I_n} + \mathbf{AB}| &= |\mathbf{I_r} + \mathbf{BA}| \\
\mathbf{A} = \text{(n,r)-matrix}, &\qquad \mathbf{B} = \text{(r,n)-matrix}
\end{aligned}
$$

Then:

$$\left|\mathbf{I}+\mathbf{\Phi}^{-1}\mathbf{B}\mathbf{Q}_2^{-1}\mathbf{B}^{\mathbf{T}}\mathbf{Q}_1\right| \overset{!}{\neq} 0$$

Because of $\det(\mathbf{\Phi}) \neq 0$ I can multiply with $\mathbf{\Phi}$

$$\left|\mathbf{\Phi}+\mathbf{B}\mathbf{Q}_2^{-1}\mathbf{B}^{\mathbf{T}}\mathbf{Q}_1\right| \overset{!}{\neq} 0$$

As $\mathbf{Q_1}$ and $\mathbf{Q_2}$ are almost free parameters, it should be possible to fulfil this condition, so that $\left(\mathbf{S_{21}}\mathbf{\Phi}^{-1}\mathbf{\Gamma}-\mathbf{S_{22}}\right)^{-1} = \mathbf{M}_{\mathbf{inv}}^{\mathbf{T}}$ (and thus also $\mathbf{M_{inv}}$) exists.

$\left[\mathbf{I}-\left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{\mathbf{i+1}}\mathbf{\Gamma}+\mathbf{S_{12}}\right)\mathbf{M_{inv}}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\right]$

Again, I have to prove that the determinant is unequal to zero. Using the relationship $|\mathbf{I_n}+\mathbf{AB}| = |\mathbf{I_r}+\mathbf{BA}|$:

$$\left|\mathbf{I}-\left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{\mathbf{i+1}}\mathbf{\Gamma}+\mathbf{S_{12}}\right)\mathbf{M_{inv}}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\right| \overset{!}{\neq} 0$$

$$\left|\mathbf{I}-\mathbf{M_{inv}}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{\mathbf{i+1}}\mathbf{\Gamma}+\mathbf{S_{12}}\right)\right| \overset{!}{\neq} 0$$

$$\left|\mathbf{I}-\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S_{12}}-\mathbf{S_{22}}\right)^{-1}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{\mathbf{i+1}}\mathbf{\Gamma}+\mathbf{S_{12}}\right)\right| \overset{!}{\neq} 0$$

$$\underbrace{\left|\left(\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S_{12}}-\mathbf{S_{22}}\right)^{-1}\right|}_{\neq 0}\cdot$$

$$\cdot\left|\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{S_{12}}-\mathbf{S_{22}}-\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{\mathbf{i+1}}\mathbf{\Gamma}+\mathbf{S_{12}}\right)\right| \overset{!}{\neq} 0$$

$$\left|\mathbf{S_{22}}+\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{\mathbf{i+1}}\mathbf{\Gamma}\right| \overset{!}{\neq} 0$$

$$\left|\mathbf{S_{22}}+\mathbf{\Gamma}^{\mathbf{T}}\mathbf{P}_{\mathbf{i+1}}\mathbf{\Gamma}\right| \overset{!}{\neq} 0$$

$\mathbf{S_{22}}$ was chosen to be positive definite above. For completely state controllable systems, it can be shown that $\mathbf{P_{i+1}}$ is positive definite or positive semidefinite [6]. The sum of a positive definite and an at least positive semidefinite matrix is positive definite. Therefore $\left|\mathbf{S_{22}}+\mathbf{\Gamma}^{\mathbf{T}}\mathbf{P}_{\mathbf{i+1}}\mathbf{\Gamma}\right|$ is unequal to zero and the matrix $\left[\mathbf{I}-\left(\mathbf{\Phi}^{\mathbf{T}}\mathbf{P}_{\mathbf{i+1}}\mathbf{\Gamma}+\mathbf{S_{12}}\right)\mathbf{M_{inv}}\mathbf{\Gamma}^{\mathbf{T}}\mathbf{\Phi}^{-\mathbf{T}}\right]^{-1}$ exists.

$\left(\mathbf{S_{22}} + \mathbf{\Gamma^T P_{k+1} \Gamma}\right)$

Nonsingular as $\mathbf{S_{22}}$ is positive definite and $\mathbf{P_{k+1}}$ is at least positive semidefinite (see above).

Chapter 4

Minimization of Jerkiness

The idea of this approach is that time rate changes of acceleration – the jerkiness – have a direct bearing on the passengers comfort and on the strains for electronic devices mounted in the fuselage which are delicate to such vibrations. In simple words: Fast changes of acceleration are considered to be worse than even larger but almost constant accelerations. However, as already mentioned above, the time-rate change of acceleration is not easy to handle. To illustrate this problem, I would like to give an example:

$$\begin{aligned} y &= w \\ \ddot{w} &= a_1 w + a_2 \dot{w} + z + u \\ \dddot{y} &= a_1 \ddot{w} + a_2 \dot{w} + \dot{z} + \dot{u} \end{aligned}$$

If I want to minimize the jerk $\dddot{y}$ of the system using u provided by a digital controller I will run into trouble since u(t) is a "staircase function" which is in principle not differentiable for all $t = t_k$ and will lead to high peaks for these points in time. From a pure mathematical point of view I could use the generalized derivative that deals with δ-functions for these times, but one the one hand I would obtain an integral of quadratic terms of $\delta(t)$ within my cost function which is really not easy to handle and on the other hand it would remain an ill-posed problem since *any* change of the controller output leads to new and even higher values of jerkiness than the gust (as a kind of lowpass filtered noise) itself.
More generally speaking, the difference of order – the first derivative of y with respect to time on which u has an instantaneous influence – between input u and output y of the whole system needs to be at least three. If, for example, the differential equations of structure and aerodynamics are second order, the used actuator or a smoothing filter for the controller output must

be at least of first order.

The structure of **A** and **x** is not prescribed. To maintain this generality I will use the following substitutions:

$$\underbrace{\begin{bmatrix} \vdots \\ \dot{\mathbf{w2}} \\ \vdots \end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix} \vdots \\ \mathbf{R} \\ \vdots \end{bmatrix}}_{\mathbf{A}} \mathbf{x} + \underbrace{\begin{bmatrix} \vdots \\ \mathbf{O} \\ \vdots \end{bmatrix}}_{\mathbf{B}} \mathbf{u} + \underbrace{\begin{bmatrix} \vdots \\ \mathbf{G2} \\ \vdots \end{bmatrix}}_{\mathbf{G}} \mathbf{z}$$

where **w2** is the vector of generalized displacements for which I want to minimize the jerk. Note: **w1** refers to the actorator(s), see the m-file *const_def.m* at the appendix (A.1). With respect to the pre-condition of order, **u** has no instantaneous effect on **w2**. Thus

$$\dddot{\mathbf{w2}} = \mathbf{R}\dot{\mathbf{x}} + \mathbf{G2}\,\dot{\mathbf{z}}$$

The cost function for the approach of minimizing the time-rate change of acceleration reads:

$$J = \frac{1}{2}\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S}\mathbf{x}_{\mathbf{k+N}} + \frac{1}{2}\sum_{i=k}^{k+N-1}\left[\dddot{\mathbf{w2}}_{\mathbf{i}}^{\mathbf{T}}\mathbf{Q_1}\dddot{\mathbf{w2}}_{\mathbf{i}} + \mathbf{u}_{\mathbf{i}}^{\mathbf{T}}\mathbf{Q_2}\mathbf{u}_{\mathbf{i}}\right] \tag{4.1}$$

Note: $\dddot{\mathbf{w2}}_{\mathbf{i}} = \dddot{\mathbf{w2}}(t)\Big|_{t=t_i}$

Inserting the substitutions into the cost function:

$$J = \frac{1}{2}\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S}\mathbf{x}_{\mathbf{k+N}} + \frac{1}{2}\sum_{i=k}^{k+N-1}\left[(\mathbf{R}\dot{\mathbf{x}}_{\mathbf{i}} + \mathbf{G2}\,\dot{\mathbf{z}}_{\mathbf{i}})^T\mathbf{Q_1}\,(\mathbf{R}\dot{\mathbf{x}}_{\mathbf{i}} + \mathbf{G2}\,\dot{\mathbf{z}}_{\mathbf{i}}) + \mathbf{u}_{\mathbf{i}}^{\mathbf{T}}\mathbf{Q_2}\mathbf{u}_{\mathbf{i}}\right]$$

The corresponding Lagrange's function reads:

$$\begin{aligned} L \;=\; & \frac{1}{2}\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S}\mathbf{x}_{\mathbf{k+N}} + \frac{1}{2}\sum_{i=k}^{k+N-1}\Big[(\mathbf{R}\dot{\mathbf{x}}_{\mathbf{i}} + \mathbf{G2}\,\dot{\mathbf{z}}_{\mathbf{i}})^T\mathbf{Q_1}\,(\mathbf{R}\dot{\mathbf{x}}_{\mathbf{i}} + \mathbf{G2}\,\dot{\mathbf{z}}_{\mathbf{i}}) + \\ & + \mathbf{u}_{\mathbf{i}}^{\mathbf{T}}\mathbf{Q_2}\mathbf{u}_{\mathbf{i}} + \boldsymbol{\lambda}_{\mathbf{i+1}}^{\mathbf{T}}\,(-\mathbf{x}_{\mathbf{i+1}} + \boldsymbol{\Phi}\mathbf{x}_{\mathbf{i}} + \boldsymbol{\Gamma}\mathbf{u}_{\mathbf{i}} + \boldsymbol{\Theta}\mathbf{z}_{\mathbf{i}})\Big] \end{aligned} \tag{4.2}$$

4.1 Transformation of the problem to the previous case

It would be a convenience to benefit from the calculations and finally from the control law derived in the previous chapter. This is in fact possible as I will show hereafter.
First of all, I have to find an approximation for $\dot{\mathbf{z}}_\mathbf{i}$ since this information is not available. Knowing that gust can be seen as lowpass filtered white noise, I can assume that $\mathbf{z}(t)$ is smooth and, especially, does not jump. Therefore the approximation

$$\dot{\mathbf{z}}_\mathbf{i} \approx \frac{\mathbf{z_i} - \mathbf{z_{i-1}}}{T}$$

with T sufficiently small is applicable.
Then I will introduce

$$\tilde{\mathbf{z}}_\mathbf{i} = \begin{bmatrix} \mathbf{z_i} \\ \mathbf{z_{i-1}} \end{bmatrix}$$

so that I can express $\dot{\mathbf{z}}_\mathbf{i}$ in terms of $\tilde{\mathbf{z}}_\mathbf{i}$:

$$\dot{\mathbf{z}}_\mathbf{i} = \frac{1}{T} \begin{bmatrix} \mathbf{I} & -\mathbf{I} \end{bmatrix} \tilde{\mathbf{z}}_\mathbf{i}$$

To avoid dealing with two variables for the gust I will also rewrite the system equations:

$$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu} + \begin{bmatrix} \mathbf{G} & \mathbf{O} \end{bmatrix} \tilde{\mathbf{z}} \tag{4.3}$$

$$\mathbf{x_{k+1}} = \mathbf{\Phi x_k} + \mathbf{\Gamma u_k} + \underbrace{\begin{bmatrix} \mathbf{\Theta} & \mathbf{O} \end{bmatrix}}_{\tilde{\mathbf{\Theta}}} \tilde{\mathbf{z}}_\mathbf{k} \tag{4.4}$$

Using these substitutions within the Lagrange's cost function (4.2):

$$\begin{aligned} L = {} & \frac{1}{2}\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S}\mathbf{x}_{\mathbf{k+N}} + \\ & + \frac{1}{2}\sum_{i=k}^{k+N-1}\Bigg[\left(\mathbf{RAx_i} + \mathbf{RBu_i} + \mathbf{R}\begin{bmatrix} \mathbf{G} & \mathbf{O} \end{bmatrix}\tilde{\mathbf{z}}_\mathbf{i} + \mathbf{G2}\frac{1}{T}\begin{bmatrix} \mathbf{I} & -\mathbf{I} \end{bmatrix}\tilde{\mathbf{z}}_\mathbf{i}\right)^T \cdot \\ & \cdot \mathbf{Q_1}\left(\mathbf{RAx_i} + \mathbf{RBu_i} + \mathbf{R}\begin{bmatrix} \mathbf{G} & \mathbf{O} \end{bmatrix}\tilde{\mathbf{z}}_\mathbf{i} + \mathbf{G2}\frac{1}{T}\begin{bmatrix} \mathbf{I} & -\mathbf{I} \end{bmatrix}\tilde{\mathbf{z}}_\mathbf{i}\right) + \\ & + \mathbf{u_i^T Q_2 u_i} + \boldsymbol{\lambda}_{\mathbf{i+1}}^{\mathbf{T}}\left(-\mathbf{x_{i+1}} + \mathbf{\Phi x_i} + \mathbf{\Gamma u_i} + \tilde{\mathbf{\Theta}}\tilde{\mathbf{z}}_\mathbf{i}\right)\Bigg] \end{aligned}$$

The last substitution I will make combines the coefficients for $\tilde{\mathbf{z}}_\mathbf{i}$:

$$\tilde{\mathbf{G}} = \mathbf{R}\begin{bmatrix} \mathbf{G} & \mathbf{O} \end{bmatrix} + \mathbf{G2}\frac{1}{T}\begin{bmatrix} \mathbf{I} & -\mathbf{I} \end{bmatrix}$$

This simplifies the Lagrange's function:

$$\begin{aligned} L &= \frac{1}{2}\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S}\mathbf{x}_{\mathbf{k+N}} + \frac{1}{2}\sum_{i=k}^{k+N-1}\Big[\left(\mathbf{R}\mathbf{A}\mathbf{x}_\mathbf{i} + \mathbf{R}\mathbf{B}\mathbf{u}_\mathbf{i} + \tilde{\mathbf{G}}\tilde{\mathbf{z}}_\mathbf{i}\right)^T \mathbf{Q}_\mathbf{1} \cdot \\ &\quad \cdot \left(\mathbf{R}\mathbf{A}\mathbf{x}_\mathbf{i} + \mathbf{R}\mathbf{B}\mathbf{u}_\mathbf{i} + \tilde{\mathbf{G}}\tilde{\mathbf{z}}_\mathbf{i}\right) + \mathbf{u}_\mathbf{i}^{\mathbf{T}}\mathbf{Q}_\mathbf{2}\mathbf{u}_\mathbf{i} + \\ &\quad + \boldsymbol{\lambda}_{\mathbf{i+1}}^{\mathbf{T}}\left(-\mathbf{x}_{\mathbf{i+1}} + \boldsymbol{\Phi}\mathbf{x}_\mathbf{i} + \boldsymbol{\Gamma}\mathbf{u}_\mathbf{i} + \tilde{\boldsymbol{\Theta}}\tilde{\mathbf{z}}_\mathbf{i}\right)\Big] \end{aligned}$$

Obviously, the last equation is similar to (3.5) if I replace $\mathbf{RA}$ by $\mathbf{A}$, $\mathbf{RB}$ by $\mathbf{B}$, $\tilde{\mathbf{G}}$ by $\mathbf{G}$ and all $\tilde{\boldsymbol{\Theta}}$, $\tilde{\mathbf{z}}_\mathbf{i}$ by $\boldsymbol{\Theta}$, $\mathbf{z}_\mathbf{i}$. This will lead to modified matrices $\mathbf{S}_{\mathbf{ij}}$:

$$\begin{aligned} L &= \frac{1}{2}\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S}\mathbf{x}_{\mathbf{k+N}} + \frac{1}{2}\sum_{i=k}^{k+N-1}\Big[\begin{pmatrix} \mathbf{x}_\mathbf{i}^{\mathbf{T}} & \mathbf{u}_\mathbf{i}^{\mathbf{T}} & \tilde{\mathbf{z}}_\mathbf{i}^{\mathbf{T}} \end{pmatrix} \cdot \\ &\quad \cdot \begin{pmatrix} \mathbf{A}^{\mathbf{T}}\mathbf{R}^{\mathbf{T}}\mathbf{Q}_\mathbf{1}\mathbf{R}\mathbf{A} & \mathbf{A}^{\mathbf{T}}\mathbf{R}^{\mathbf{T}}\mathbf{Q}_\mathbf{1}\mathbf{R}\mathbf{B} & \mathbf{A}^{\mathbf{T}}\mathbf{R}^{\mathbf{T}}\mathbf{Q}_\mathbf{1}\tilde{\mathbf{G}} \\ \mathbf{B}^{\mathbf{T}}\mathbf{R}^{\mathbf{T}}\mathbf{Q}_\mathbf{1}\mathbf{R}\mathbf{A} & \mathbf{B}^{\mathbf{T}}\mathbf{R}^{\mathbf{T}}\mathbf{Q}_\mathbf{1}\mathbf{R}\mathbf{B} + \mathbf{Q}_\mathbf{2} & \mathbf{B}^{\mathbf{T}}\mathbf{R}^{\mathbf{T}}\mathbf{Q}_\mathbf{1}\tilde{\mathbf{G}} \\ \tilde{\mathbf{G}}^{\mathbf{T}}\mathbf{Q}_\mathbf{1}\mathbf{R}\mathbf{A} & \tilde{\mathbf{G}}^{\mathbf{T}}\mathbf{Q}_\mathbf{1}\mathbf{R}\mathbf{B} & \tilde{\mathbf{G}}^{\mathbf{T}}\mathbf{Q}_\mathbf{1}\tilde{\mathbf{G}} \end{pmatrix} \begin{pmatrix} \mathbf{x}_\mathbf{i} \\ \mathbf{u}_\mathbf{i} \\ \tilde{\mathbf{z}}_\mathbf{i} \end{pmatrix} + \\ &\quad + \boldsymbol{\lambda}_{\mathbf{i+1}}^{\mathbf{T}}\left(-\mathbf{x}_{\mathbf{i+1}} + \boldsymbol{\Phi}\mathbf{x}_\mathbf{i} + \boldsymbol{\Gamma}\mathbf{u}_\mathbf{i} + \tilde{\boldsymbol{\Theta}}\tilde{\mathbf{z}}_\mathbf{i}\right)\Big] \\ L &= \frac{1}{2}\mathbf{x}_{\mathbf{k+N}}^{\mathbf{T}}\mathbf{S}\mathbf{x}_{\mathbf{k+N}} + \\ &\quad + \frac{1}{2}\sum_{i=k}^{k+N-1}\Big[\begin{pmatrix} \mathbf{x}_\mathbf{i}^{\mathbf{T}} & \mathbf{u}_\mathbf{i}^{\mathbf{T}} & \tilde{\mathbf{z}}_\mathbf{i}^{\mathbf{T}} \end{pmatrix} \begin{pmatrix} \mathbf{S_{11}} & \mathbf{S_{12}} & \mathbf{S_{13}} \\ \mathbf{S_{21}} & \mathbf{S_{22}} & \mathbf{S_{23}} \\ \mathbf{S_{31}} & \mathbf{S_{32}} & \mathbf{S_{33}} \end{pmatrix} \begin{pmatrix} \mathbf{x}_\mathbf{i} \\ \mathbf{u}_\mathbf{i} \\ \tilde{\mathbf{z}}_\mathbf{i} \end{pmatrix} + \\ &\quad + \boldsymbol{\lambda}_{\mathbf{i+1}}^{\mathbf{T}}\left(-\mathbf{x}_{\mathbf{i+1}} + \boldsymbol{\Phi}\mathbf{x}_\mathbf{i} + \boldsymbol{\Gamma}\mathbf{u}_\mathbf{i} + \tilde{\boldsymbol{\Theta}}\tilde{\mathbf{z}}_\mathbf{i}\right)\Big] \end{aligned}$$

Again: $\mathbf{Q_1} = \mathbf{Q_1^T}$ and $\mathbf{Q_2} = \mathbf{Q_2^T}$, thus $\mathbf{S_{ij}^T} = \mathbf{S_{ji}}\ \forall i,j \quad \mathbf{S^T} = \mathbf{S}$
$\mathbf{Q_1}$, $\mathbf{Q_2}$ and $\mathbf{S}$ are real symmetric weight matrices.

Keeping these substitutions in mind I can benefit from all results of the previous chapter. Thus, I will only state the results for the sake of completeness:

$$\begin{aligned} \mathbf{P}_\mathbf{i} &= \left[\mathbf{I} - \left(\boldsymbol{\Phi}^{\mathbf{T}}\mathbf{P}_{\mathbf{i+1}}\boldsymbol{\Gamma} + \mathbf{S_{12}}\right)\mathbf{M}_{\mathbf{inv}}\boldsymbol{\Gamma}^{\mathbf{T}}\boldsymbol{\Phi}^{-\mathbf{T}}\right]^{-1} \cdot \\ &\quad \cdot \Big[\boldsymbol{\Phi}^{\mathbf{T}}\mathbf{P}_{\mathbf{i+1}}\left(\boldsymbol{\Phi} - \boldsymbol{\Gamma}\mathbf{M}_{\mathbf{inv}}\left(\boldsymbol{\Gamma}^{\mathbf{T}}\boldsymbol{\Phi}^{-\mathbf{T}}\mathbf{S_{11}} - \mathbf{S_{21}}\right)\right) + \mathbf{S_{11}} - \end{aligned}$$

$$- \mathbf{S_{12}} \mathbf{M_{inv}} \left(\mathbf{\Gamma}^{\mathbf{T}} \mathbf{\Phi}^{-\mathbf{T}} \mathbf{S_{11}} - \mathbf{S_{21}} \right) \Big] \qquad (4.5)$$
$$\text{for } i = k+1 \ldots k+N-1$$

$$\mathbf{p_i} = \Big[\mathbf{I} - \left(\mathbf{\Phi}^{\mathbf{T}} \mathbf{P_{i+1}} \mathbf{\Gamma} + \mathbf{S_{12}} \right) \mathbf{M_{inv}} \mathbf{\Gamma}^{\mathbf{T}} \mathbf{\Phi}^{-\mathbf{T}} \Big]^{-1} \Big[\mathbf{\Phi}^{\mathbf{T}} \mathbf{p_{i+1}} + \left(\mathbf{\Phi}^{\mathbf{T}} \mathbf{P_{i+1}} \tilde{\mathbf{\Theta}} + \right.$$
$$\left. + \mathbf{S_{13}} - \left(\mathbf{\Phi}^{\mathbf{T}} \mathbf{P_{i+1}} \mathbf{\Gamma} + \mathbf{S_{12}} \right) \mathbf{M_{inv}} \left(\mathbf{\Gamma}^{\mathbf{T}} \mathbf{\Phi}^{-\mathbf{T}} \mathbf{S_{13}} - \mathbf{S_{23}} \right) \right) \tilde{\mathbf{z}}_{\mathbf{i}} \Big] \qquad (4.6)$$
$$\text{for } i = k+1 \ldots k+N-1$$

$$\mathbf{u_k} = - \left(\mathbf{S_{22}} + \mathbf{\Gamma}^{\mathbf{T}} \mathbf{P_{k+1}} \mathbf{\Gamma} \right)^{-1} \Big[\left(\mathbf{S_{21}} + \mathbf{\Gamma}^{\mathbf{T}} \mathbf{P_{k+1}} \mathbf{\Phi} \right) \mathbf{x_k} +$$
$$+ \left(\mathbf{S_{23}} + \mathbf{\Gamma}^{\mathbf{T}} \mathbf{P_{k+1}} \tilde{\mathbf{\Theta}} \right) \tilde{\mathbf{z}}_{\mathbf{k}} + \mathbf{\Gamma}^{\mathbf{T}} \mathbf{p_{k+1}} \Big] \qquad (4.7)$$

In order to evaluate how good my goal is achieved depending on the sensor positions, I will use a set of cost functions like in the previous chapter, however, with the "ride comfort" as defined for this approach:

$$\tilde{J}_1 = \int_{t_0}^{t_e} \dddot{\mathbf{w2}}^{\mathbf{T}}(t) \, \mathbf{RC} \, \dddot{\mathbf{w2}}^{\mathbf{T}}(t) dt \qquad \text{ride comfort} \qquad (4.8)$$

4.2 Necessary assumptions

Even though I can "recycle" the whole calculation of the previous chapter by using the above mentioned substitutions, I have to make sure that the necessary matrix inversions are possible within the new system as well. Again, I can benefit from the preceding investigation, but have to consider the modified definitions:

$\mathbf{S_{22}}$

$$\mathbf{S_{22}} = \mathbf{B}^{\mathbf{T}} \mathbf{R}^{\mathbf{T}} \mathbf{Q_1} \mathbf{RB} + \mathbf{Q_2}$$

If $(\mathbf{B}^{\mathbf{T}} \mathbf{R}^{\mathbf{T}} \mathbf{Q_1} \mathbf{RB} + \mathbf{Q_2})$ is a positive definite matrix then its determinant is unequal to zero.
Similar to the previous argumentation for $\mathbf{S_{22}}$, if the terms of $\mathbf{Q_1}$ that refer to nonzero parts of $\mathbf{RB}$ are at least positive semidefinite then $\mathbf{B}^{\mathbf{T}} \mathbf{R}^{\mathbf{T}} \mathbf{Q_1} \mathbf{RB}$ will be, too. Again, I will assume that $\mathbf{Q_2}$ is positive definite. $\Rightarrow \mathbf{S_{22}^{-1}}$ exists

$(\mathbf{S_{21}} \mathbf{\Phi}^{-1} \mathbf{\Gamma} - \mathbf{S_{22}})$
The transformations are similar to the previous case although there

are some terms of $\mathbf{R}$ and $\mathbf{R^T}$ involved. The two alternative conditions finally read:

$$\left|\mathbf{B^T R^T Q_1 R \Phi^{-1} B + Q_2}\right| \overset{!}{\neq} 0$$

$$\left|\mathbf{\Phi + R B Q_2^{-1} B^T R^T Q_1}\right| \overset{!}{\neq} 0$$

$\left[\mathbf{I} - \left(\mathbf{\Phi^T P_{i+1} \Gamma + S_{12}}\right) \mathbf{M_{inv} \Gamma^T \Phi^{-T}}\right]$

The argumentation is equal to the previous chapter since there are no decompositions of $\mathbf{S_{ij}}$ and no $\tilde{\mathbf{\Theta}}$ involved.

$\left(\mathbf{S_{22} + \Gamma^T P_{k+1} \Gamma}\right)$

The argumentation is equal to the previous chapter since there are no decompositions of $\mathbf{S_{ij}}$ and no $\tilde{\mathbf{\Theta}}$ involved.

Chapter 5

Computational Performance Requirements

At first sight, the various matrix inversions seem to be a serious problem when it comes to implementing the optimal control law on a DSP. This is, however, not the case, as will be shown in this chapter.

5.1 Minimization of accelerations

Equations (3.16) and (3.17) as well as the included substitution $\mathbf{M_{inv}}$ do not contain the variables $\mathbf{x_i}$, $\mathbf{u_i}$ or $\mathbf{z_i}$. As a result, the whole set of matrices $\mathbf{P_{k+N}} \dots \mathbf{P_{k+1}}$ can be computed offline and saved.
Then, (3.19) can be simplified as follows:

$$\mathbf{p_i} = \mathbf{F1_i}\left(\mathbf{\Phi^T p_{i+1}} + \mathbf{F2_i}\,\mathbf{z_i}\right) \tag{5.1}$$

where

$$\begin{aligned}
\mathbf{F1_i} &= \left[\mathbf{I} - \left(\mathbf{\Phi^T P_{i+1}\Gamma} + \mathbf{S_{12}}\right)\mathbf{M_{inv}}\mathbf{\Gamma^T\Phi^{-T}}\right]^{-1} \\
\mathbf{F2_i} &= \mathbf{\Phi^T P_{i+1}\Theta} + \mathbf{S_{13}} - \left(\mathbf{\Phi^T P_{i+1}\Gamma} + \mathbf{S_{12}}\right)\mathbf{M_{inv}}\left(\mathbf{\Gamma^T\Phi^{-T}S_{13}} - \mathbf{S_{23}}\right)
\end{aligned}$$

for $i = k+1 \dots k+N-1$
Obviously, **F1** and **F2** only depend on constant matrices and the known $\mathbf{P_{k+N}} \dots \mathbf{P_{k+1}}$ so that those matrices can be pre-computed and stored as well. Furthermore, the fact that $\mathbf{p_{k+N}}$ and the unknown gust $\mathbf{z_{k+n}} \dots \mathbf{z_{k+N-1}}$ are set to zero allows another "short cut" for the calculation: $\mathbf{p_i}$ will (starting the recursion with $i = k+N-1$) be zero as long as $\mathbf{z_i} = \mathbf{0}$, that is for $i = k+n \dots k+N-1$. To save time, for

- $n = 1$ set directly

$$\mathbf{p_{k+1}} = \mathbf{0}$$

- $n > 1$ start the calculation with

$$\mathbf{p_{k+n-1}} = \mathbf{F1_{k+n-1}}\,\mathbf{F2_{k+n-1}}\,\mathbf{z_{k+n-1}} \tag{5.2}$$

The optimal control law (3.21) can be simplified as well:

$$\begin{aligned}
\mathbf{u_k} &= \mathbf{F3}\left(\mathbf{F4}\,\mathbf{x_k} + \mathbf{F5}\,\mathbf{z_k} + \mathbf{\Gamma^T p_{k+1}}\right) \\
\mathbf{F3} &= -\left(\mathbf{S_{22}} + \mathbf{\Gamma^T P_{k+1}\Gamma}\right)^{-1} \\
\mathbf{F4} &= \mathbf{S_{21}} + \mathbf{\Gamma^T P_{k+1}\Phi} \\
\mathbf{F5} &= \mathbf{S_{23}} + \mathbf{\Gamma^T P_{k+1}\Theta}
\end{aligned} \tag{5.3}$$

Result: $\mathbf{F1} \ldots \mathbf{F5}$ and $\mathbf{P_{k+N}} \ldots \mathbf{P_{k+1}}$ can be calculated offline (before the experiment) and stored in the controller. During the experiment only $\mathbf{p_i}$, $i = k+n-1 \ldots k+1$ (or none for $n = 1$) need to be calculated online using simple and fast additions and multiplications. Slow matrix inversions are not required at that time. $\mathbf{u_k}$ is finally derived using such easy operation as well.

5.2 Minimization of jerkiness

Also for this approach, offline calculations for $\mathbf{P_{k+N}} \ldots \mathbf{P_{k+1}}$ are possible (equation (4.5)). The difference concerning the online part is (apart from the modified definitions of $\mathbf{S_{ij}}$) the fact that now $\tilde{\mathbf{\Theta}}$ and $\tilde{\mathbf{z}}_\mathbf{i}$ come into play.

$$\begin{aligned}
\mathbf{F1_i} &= \left[\mathbf{I} - \left(\mathbf{\Phi^T P_{i+1}\Gamma} + \mathbf{S_{12}}\right)\mathbf{M_{inv}}\mathbf{\Gamma^T\Phi^{-T}}\right]^{-1} \\
\mathbf{F2_i} &= \mathbf{\Phi^T P_{i+1}\tilde{\Theta}} + \mathbf{S_{13}} - \left(\mathbf{\Phi^T P_{i+1}\Gamma} + \mathbf{S_{12}}\right)\mathbf{M_{inv}}\left(\mathbf{\Gamma^T\Phi^{-T}S_{13}} - \mathbf{S_{23}}\right)
\end{aligned}$$

As a result, I have to be very careful with the above mentioned "short cut" for $\mathbf{p_{k+1}}$, as:

$$\mathbf{p_i} = \mathbf{F1_i}\left(\mathbf{\Phi^T p_{i+1}} + \mathbf{F2_i}\begin{bmatrix}\mathbf{z_i}\\ \mathbf{z_{i-1}}\end{bmatrix}\right) \tag{5.4}$$

The fact that $\mathbf{p_{k+N}}$ and the unknown gust $\mathbf{z_{k+n}} \ldots \mathbf{z_{k+N-1}}$ are set to zero requires to start the downward calculation for $i = k+n$, as opposed to the

previous case. For $n = 1$, $\mathbf{p_{k+1}}$ cannot be set to zero any more. The other matrices $\mathbf{Fi}$ for this approach read

$$\begin{aligned} \mathbf{F3} &= -\left(\mathbf{S_{22}} + \mathbf{\Gamma^T P_{k+1} \Gamma}\right)^{-1} \\ \mathbf{F4} &= \mathbf{S_{21}} + \mathbf{\Gamma^T P_{k+1} \Phi} \\ \mathbf{F5} &= \mathbf{S_{23}} + \mathbf{\Gamma^T P_{k+1} \tilde{\Theta}} \end{aligned}$$

leading to the simplified control law:

$$\mathbf{u_k} = \mathbf{F3}\left(\mathbf{F4}\,\mathbf{x_k} + \mathbf{F5}\,\tilde{\mathbf{z}}_\mathbf{k} + \mathbf{\Gamma^T p_{k+1}}\right) \tag{5.5}$$

5.3 Possible limitations

Depending on the algorithm for matrix multiplications the computation time will increase with $O(q^3)$ in the worst case, where q is the dimension of two multiplied (square) matrices. For system models of very high order, the multiplication could become a problem. As shown above, the number of steps for the recursive calculation of $\mathbf{p_{k+1}}$ depends on n, that is the "distance" between the sensor(s) and the wing. For measurement far ahead of the airplane, the recursion could require too much time and would not be applicable any more.

Chapter 6

Implementation with Matlab/Simulink

Now, I would like to implement the above-derived optimal control algorithms with Matlab/Simulink, which is well-established in the field of control engineering. These "experiments" will evaluate how good my goal of gust alleviation is achieved depending on the parameters involved. I will split the whole model in several subsystems which are described hereafter.

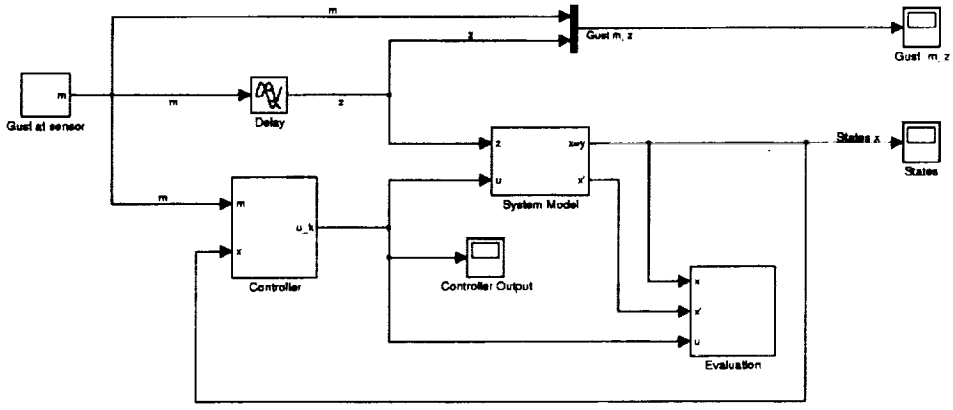

Figure 6.1: The test configuration and its subsystems

6.1 Dryden model of gust

To receive meaningful simulation results, a realistic gust signal needs to be generated. I will use the Dryden model which consists of a white noise generator and the following second order lowpass filter:

$$G(s) = \frac{1 + \sqrt{3}\omega_g}{(1 + \omega_g s)^2}$$

In chapter 3 I pointed out that the gust signal should be equal within

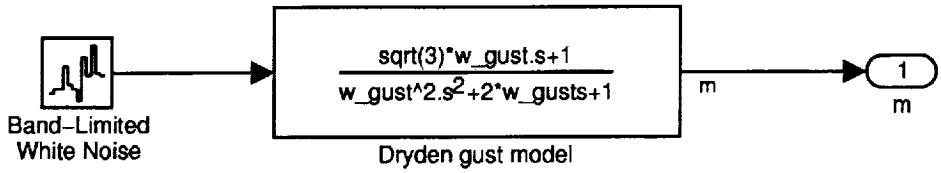

Figure 6.2: The gust source using the Dryden model

the experiments to ensure a maximum of comparability between the results. I proposed to sample and save a random gust function and recall it for every test. The noise generator of Matlab/Simulink does provide the same sequence for each simulation, so that a "gust data file" is not necessary.

6.2 Airplane state space model

Despite the fact that the controller design is based on a discrete-time model (and also a discrete-time cost function), the "real" system of course remains continuous. Both representations lead to the same states for $t = kT$. The continuous-time system, however, also provides information between these points in time and should therefore be used for evaluation.

6.3 Controller

The controller is realized for discrete-time. For this reason, all input signals, that is the gust right at the sensor and the current states, are sampled. Of course, the control algorithm itself could be realized by a m-file that is run for each time step t_k like a program on a DSP. For the application on Simulink, a transfer function, however, fits better into the signal path design. As a side effect, the transfer function implementation allows to have an easier

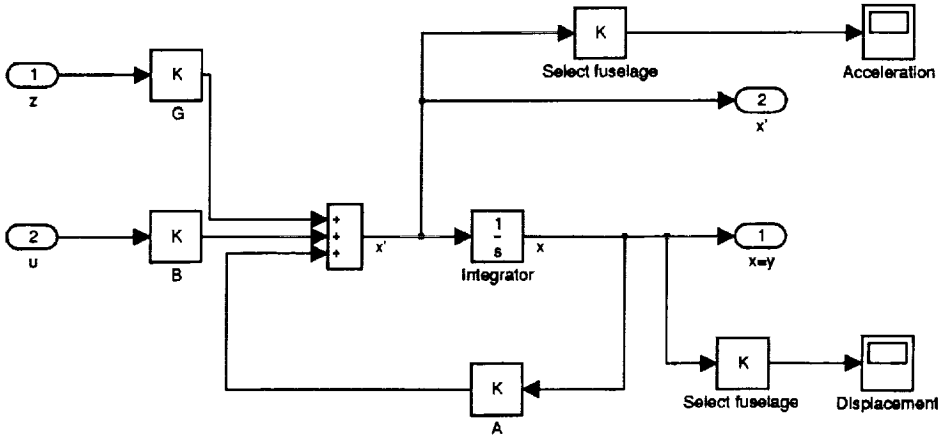

Figure 6.3: The continuous-time model of the airplane

insight into the controller. I will derive this for the approach of chapter 3 first and then discuss the necessary modifications for the algorithm of 4.

6.3.1 Minimization of accelerations

For the case of $n > 1$, (5.1) and (5.2) lead to a set of coupled equations:

$$\begin{aligned}
\mathbf{p_{k+n-1}} &= \mathbf{F1_{k+n-1}}\,\mathbf{F2_{k+n-1}}\,\mathbf{z_{k+n-1}} \\
\mathbf{p_{k+n-2}} &= \mathbf{F1_{k+n-2}}\left(\mathbf{\Phi^T p_{k+n-1}} + \mathbf{F2_{k+n-2}}\,\mathbf{z_{k+n-2}}\right) \\
\mathbf{p_{k+n-3}} &= \mathbf{F1_{k+n-3}}\left(\mathbf{\Phi^T p_{k+n-2}} + \mathbf{F2_{k+n-3}}\,\mathbf{z_{k+n-3}}\right) \\
&\vdots \\
\mathbf{p_{k+2}} &= \mathbf{F1_{k+2}}\left(\mathbf{\Phi^T p_{k+3}} + \mathbf{F2_{k+2}}\,\mathbf{z_{k+2}}\right) \\
\mathbf{p_{k+1}} &= \mathbf{F1_{k+1}}\left(\mathbf{\Phi^T p_{k+2}} + \mathbf{F2_{k+1}}\,\mathbf{z_{k+1}}\right)
\end{aligned}$$

Eleminating $\mathbf{p_{k+n-1}} \cdots \mathbf{p_{k+2}}$ I obtain:

$$\begin{aligned}
\mathbf{p_{k+1}} = \;& \mathbf{F1_{k+1}\Phi^T\cdot}\;\; \mathbf{F1_{k+2}\Phi^T\cdot}\;\; \cdots\;\; \mathbf{F1_{k+n-2}\Phi^T\cdot}\;\; \mathbf{F1_{k+n-1}\cdot}\;\; \mathbf{F2_{k+n-1}\cdot}\;\; \mathbf{z_{k+n-1}+} \\
& \mathbf{F1_{k+1}\Phi^T\cdot}\;\; \mathbf{F1_{k+2}\Phi^T\cdot}\;\; \cdots\;\; \mathbf{F1_{k+n-2}\cdot}\qquad\qquad\quad \mathbf{F2_{k+n-2}\cdot}\;\; \mathbf{z_{k+n-2}+} \\
& \quad\vdots\qquad\qquad\quad\;\; \vdots\qquad\qquad\qquad\qquad\qquad\quad \vdots\qquad\quad \vdots \\
& \mathbf{F1_{k+1}\Phi^T\cdot}\;\; \mathbf{F1_{k+2}\cdot}\qquad\qquad\qquad\qquad\qquad\quad \mathbf{F2_{k+2}\cdot}\;\; \mathbf{z_{k+2}+} \\
& \mathbf{F1_{k+1}\cdot}\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \mathbf{F2_{k+1}\cdot}\;\; \mathbf{z_{k+1}}
\end{aligned} \tag{6.1}$$

Now, I apply a z-transformation on this equation. To avoid confusion I will rename the variables first:

$$\begin{aligned} \mathbf{g_k} &:= \mathbf{z_k} \\ \hat{\mathbf{p}}_\mathbf{k} &:= \mathbf{p_{k+1}} \end{aligned}$$

Using the relation

$$Z\left\{\mathbf{g_{k+i}}\right\} = z^i \mathbf{G_z}(z)$$

Note: $\mathbf{G_z}(z)$ has nothing to do with the matrix $\mathbf{G}$ of the state space model. The z-transformation of (6.1) can be written as follows:

$$\begin{aligned} \hat{\mathbf{P}}_\mathbf{z}(z) &= \sum_{i=1}^{n-1} \left\{ \left[\prod_{\nu=1}^{i} \mathbf{F1_{k+\nu}} \mathbf{\Phi}^\mathbf{T} \right] \mathbf{\Phi}^{-\mathbf{T}} \mathbf{F2_{k+i}}\, z^i \mathbf{G_z}(z) \right\} \\ \hat{\mathbf{P}}_\mathbf{z}(z) &= \underbrace{\sum_{i=1}^{n-1} \left\{ \left[\prod_{\nu=1}^{i} \mathbf{F1_{k+\nu}} \mathbf{\Phi}^\mathbf{T} \right] \mathbf{\Phi}^{-\mathbf{T}} \mathbf{F2_{k+i}}\, z^i \right\}}_{\tilde{\mathbf{PF}}_\mathbf{z}(z)} \mathbf{G_z}(z) \end{aligned}$$

Now, I have a filter $\tilde{\mathbf{PF}}_\mathbf{z}(z)$ with $\mathbf{g_k} = \mathbf{z_k}$ (that is the gust directly at the wing) as the input and $\hat{\mathbf{p}}_\mathbf{k} = \mathbf{p_{k+1}}$ as the output. The controller, however, will receive the signal $\mathbf{m_k}$ of the gust measurement device, that is $n-1$ steps ahead:

$$\begin{aligned} \mathbf{g_k} &= \mathbf{m_{k-(n-1)}} \\ &\Downarrow \quad \text{z-transformation} \\ \mathbf{G_z}(z) &= z^{-(n-1)} \mathbf{M_z}(z) \end{aligned}$$

Finally the filter for $\mathbf{m_k}$ providing $\mathbf{p_{k+1}}$ reads:

$$\mathbf{PF_z}(z) = \frac{\sum_{i=1}^{n-1} \left\{ \left[\prod_{\nu=1}^{i} \mathbf{F1_{k+\nu}} \mathbf{\Phi}^\mathbf{T} \right] \mathbf{\Phi}^{-\mathbf{T}} \mathbf{F2_{k+i}}\, z^i \right\}}{z^{n-1}}$$

Note: For the case of $n = 1$, $\mathbf{p_{k+1}} = \mathbf{0}$ so that this filter has to be set to zero. The z-transformation of the optimal control law in the form of (5.3) with these substitutions reads:

$$\begin{aligned} \mathbf{U_z}(z) &= \mathbf{F3} \left[\mathbf{F4}\, \mathbf{X_z}(z) + \mathbf{F5}\, \mathbf{G_z}(z) + \mathbf{\Gamma}^\mathbf{T} \hat{\mathbf{P}}_\mathbf{z}(z) \right] \\ &= \mathbf{F3} \left[\mathbf{F4}\, \mathbf{X_z}(z) + \mathbf{F5}\, z^{-(n-1)} \mathbf{M_z}(z) + \mathbf{\Gamma}^\mathbf{T} \mathbf{PF_z}(z)\, \mathbf{M_z}(z) \right] \end{aligned}$$

$$
\begin{aligned}
&= \mathbf{F3}\left[\mathbf{F4}\,\mathbf{X_z}(z) + \underbrace{\left(\mathbf{F5}\,z^{-(n-1)} + \mathbf{\Gamma^T P F_z}(z)\right)}_{\mathbf{GF_z}(z)}\mathbf{M_z}(z)\right] \\
\mathbf{U_z}(z) &= \mathbf{F3}\Big[\mathbf{F4}\,\mathbf{X_z}(z) + \mathbf{GF_z}(z)\,\mathbf{M_z}(z)\Big] \qquad (6.2)
\end{aligned}
$$

$\mathbf{GF_z}(z)$ is a kind of "gust filter" and has the effect of a dynamic feed-forward part of the controller.

$$
\begin{aligned}
\mathbf{GF_z}(z) &= \frac{\mathbf{\Gamma^T}\sum_{i=1}^{n-1}\left\{\left[\prod_{\nu=1}^{i}\mathbf{F1_{k+\nu}}\mathbf{\Phi^T}\right]\mathbf{\Phi^{-T}F2_{k+i}}\,z^i\right\} + \mathbf{F5}}{z^{n-1}} \\
&= \frac{\mathbf{Num_{n-1}}z^{n-1} + \mathbf{Num_{n-2}}z^{n-2} + \cdots + \mathbf{Num_2}z^2 + \mathbf{Num_1}z + \mathbf{F5}}{z^{n-1}}
\end{aligned}
$$

$\mathbf{Num_i}$ are matrices of the same size as $\mathbf{G}$. The discrete transfer function module of Simulink which I would like to use is restricted to scalar input. Therefore, $\mathbf{z_k}$ and $\mathbf{m_k}$ have to be a scalar within this evaluation.

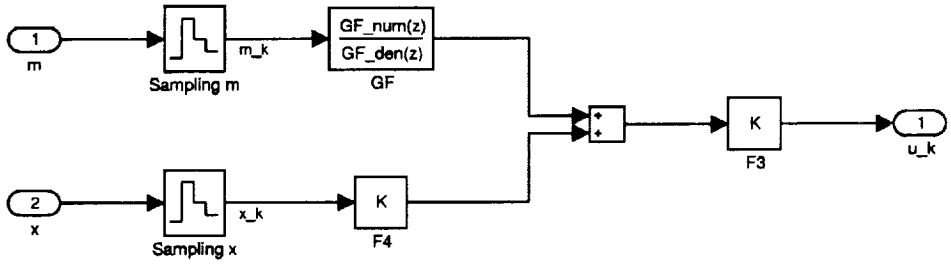

Figure 6.4: The controller including the feed-forward "gust filter" for the case of chapter 3

6.3.2 Minimization of jerkiness

The following calculation for the gust part of the controller is similar to the case above, but I have to consider $\tilde{\mathbf{z}}_\mathbf{i}$. Equation (5.4) allows to describe $\mathbf{p_{k+1}}$ as a linear combination of gust data:

$$
\begin{aligned}
\mathbf{p_{k+1}} = \;& \mathbf{F1_{k+1}\Phi^T\cdot}\;\; \mathbf{F1_{k+2}\Phi^T\cdot}\;\; \cdots\;\; \mathbf{F1_{k+n-1}\Phi^T\cdot}\;\; \mathbf{F1_{k+n}\cdot}\;\; \mathbf{F2_{k+n}\cdot}\;\; \tilde{\mathbf{z}}_{\mathbf{k+n}}+ \\
& \mathbf{F1_{k+1}\Phi^T\cdot}\;\; \mathbf{F1_{k+2}\Phi^T\cdot}\;\; \cdots\;\; \mathbf{F1_{k+n-1}\cdot}\;\; \mathbf{F2_{k+n-1}\cdot}\;\; \tilde{\mathbf{z}}_{\mathbf{k+n-1}}+ \\
& \quad\vdots \qquad\qquad\quad \vdots \qquad\qquad\qquad\qquad\quad \vdots \qquad\quad \vdots \\
& \mathbf{F1_{k+1}\Phi^T\cdot}\;\; \mathbf{F1_{k+2}\cdot}\;\; \mathbf{F2_{k+2}\cdot}\;\; \tilde{\mathbf{z}}_{\mathbf{k+2}}+ \\
& \mathbf{F1_{k+1}\cdot}\;\; \mathbf{F2_{k+1}\cdot}\;\; \tilde{\mathbf{z}}_{\mathbf{k+1}}
\end{aligned} \qquad (6.3)
$$

I have to consider that only $\mathbf{z_k} \dots \mathbf{z_{k+n-1}}$ are available and $\mathbf{z_{k+n}} \dots \mathbf{z_{k+N-1}}$ are set to zero as described at the beginning. Therefore I have to be careful with $\tilde{\mathbf{z}}_{\mathbf{k+n}}$:

$$\tilde{\mathbf{z}}_{\mathbf{k+n}} = \begin{bmatrix} \mathbf{z_{k+n}} \\ \mathbf{z_{k+n-1}} \end{bmatrix} := \begin{bmatrix} \mathbf{0} \\ \mathbf{z_{k+n-1}} \end{bmatrix}$$

Again, I will rename the variables and apply a z-transformation on this equation.

$$\begin{aligned} \mathbf{g_k} &:= \mathbf{z_k} \\ \tilde{\mathbf{g}}_\mathbf{k} &:= \tilde{\mathbf{z}}_\mathbf{k} = \begin{bmatrix} \mathbf{z_k} & \mathbf{z_{k-1}} \end{bmatrix}^T \\ \hat{\mathbf{p}}_\mathbf{k} &:= \mathbf{p_{k+1}} \end{aligned}$$

Using the relation

$$Z\left\{\tilde{\mathbf{g}}_{\mathbf{k+i}}\right\} = \begin{bmatrix} z^i\mathbf{I} \\ z^{i-1}\mathbf{I} \end{bmatrix} \mathbf{G_z}(z)$$

the z-transformation of (6.3) leads to:

$$\begin{aligned} \hat{\mathbf{P}}_\mathbf{z}(z) &= \left\{ \sum_{i=1}^{n-1} \left[\left(\prod_{\nu=1}^{i} \mathbf{F1_{k+\nu}} \mathbf{\Phi^T} \right) \mathbf{\Phi^{-T}} \mathbf{F2_{k+i}} \begin{bmatrix} z^i\mathbf{I} & z^{i-1}\mathbf{I} \end{bmatrix}^T \right] + \right. \\ &\quad \left. + \left(\prod_{\nu=1}^{n} \mathbf{F1_{k+\nu}} \mathbf{\Phi^T} \right) \mathbf{\Phi^{-T}} \mathbf{F2_{k+n}} \begin{bmatrix} \mathbf{O} & z^{n-1}\mathbf{I} \end{bmatrix}^T \right\} \mathbf{G_z}(z) \\ \hat{\mathbf{P}}_\mathbf{z}(z) &= \tilde{\mathbf{PF}}_\mathbf{z}(z) \mathbf{G_z}(z) \end{aligned}$$

Finally the filter for $\mathbf{m_k}$ providing $\mathbf{p_{k+1}}$ reads:

$$\begin{aligned} \mathbf{PF_z}(z) &= \frac{\sum_{i=1}^{n-1} \left[\left(\prod_{\nu=1}^{i} \mathbf{F1_{k+\nu}} \mathbf{\Phi^T} \right) \mathbf{\Phi^{-T}} \mathbf{F2_{k+i}} \begin{bmatrix} z^i\mathbf{I} & z^{i-1}\mathbf{I} \end{bmatrix}^T \right]}{z^{n-1}} + \\ &\quad + \frac{\left(\prod_{\nu=1}^{n} \mathbf{F1_{k+\nu}} \mathbf{\Phi^T} \right) \mathbf{\Phi^{-T}} \mathbf{F2_{k+n}} \begin{bmatrix} \mathbf{O} & z^{n-1}\mathbf{I} \end{bmatrix}^T}{z^{n-1}} \end{aligned}$$

Using these substitutions, the optimal control law in the form of (5.5) reads:

$$\begin{aligned} \mathbf{U_z}(z) &= \mathbf{F3} \left[\mathbf{F4}\, \mathbf{X_z}(z) + \mathbf{F5} \begin{bmatrix} \mathbf{I} \\ z^{-1}\mathbf{I} \end{bmatrix} \mathbf{G_z}(z) + \mathbf{\Gamma^T} \hat{\mathbf{P}}_\mathbf{z}(z) \right] \\ &= \mathbf{F3} \left[\mathbf{F4}\, \mathbf{X_z}(z) + \underbrace{\left(\mathbf{F5} \begin{bmatrix} \mathbf{I} \\ z^{-1}\mathbf{I} \end{bmatrix} z^{-(n-1)} + \mathbf{\Gamma^T} \mathbf{PF_z}(z) \right)}_{\mathbf{GF_z}(z)} \mathbf{M_z}(z) \right] \end{aligned}$$

$$\mathbf{U_z}(z) \;=\; \mathbf{F3}\Big[\mathbf{F4}\,\mathbf{X_z}(z) + \mathbf{GF_z}(z)\,\mathbf{M_z}(z)\Big] \tag{6.4}$$

Obviously the controller layout is similar to the one dealing with accelerations.

$$\mathbf{GF_z}(z) = \frac{\mathbf{\Gamma^T} \sum_{i=1}^{n-1} \left[\left(\prod_{\nu=1}^{i} \mathbf{F1_{k+\nu}} \mathbf{\Phi^T} \right) \mathbf{\Phi^{-T}} \mathbf{F2_{k+i}} \begin{bmatrix} z^i\mathbf{I} & z^{i-1}\mathbf{I} \end{bmatrix}^T \right]}{z^{n-1}} +$$
$$+ \frac{\mathbf{\Gamma^T} \left(\prod_{\nu=1}^{n} \mathbf{F1_{k+\nu}} \mathbf{\Phi^T} \right) \mathbf{\Phi^{-T}} \mathbf{F2_{k+n}} \begin{bmatrix} \mathbf{O} & z^{n-1}\mathbf{I} \end{bmatrix}^T + \mathbf{F5} \begin{bmatrix} \mathbf{I} & z^{-1}\mathbf{I} \end{bmatrix}^T}{z^{n-1}}$$

The restriction of scalar $\mathbf{z_k}$ and $\mathbf{m_k}$ also applies to this case.

6.3.3 Bounds of the controller output

With respect to the application of my controller, it is important to know the upper bound of outputs. Real controllers and actuators can only operate within a limited range, so that a non-linearity comes into play if outputs exceed these limits. This would certainly reduce the performance with respect to my goal. The chief problem, however, is that such non-linearities can jeopardize stability, which is often only verified for the linear case. In course of the following I will determine the maximum controller output for a somehow "worst case". I will assume that the initial value of the states is zero, as for a stable linear system this part (superposition) will decrease to zero anyway. Thus, the states are a linear function of gust data, so that the upper bound for the controller output is proportional to a kind of maximum of gust. To find this upper bound for all possible sequences of gust I will make use of norms theory, especially of H_∞ which is known from robust controls.

As for the definition of the used vector norms I am quite free, as long as some constraints are satisfied. The H_∞ norm for the discrete-time transfer function matrix is defined as

$$\left\| \hat{\mathbf{F}} \right\|_\infty \;=\; \sup_\omega \bar{\sigma} \left(\hat{\mathbf{F}}(e^{j\omega}) \right)$$

$\bar{\sigma}$ is the greatest singular value of $\mathbf{F^*F}$. This norm is not trivial to calculate. Therefore I will make use of the command `normhinf(sys)` provided by the robust control toolbox of Matlab. For this reason I have to transform the whole controlled system, especially the "gust filter" of the controller, to state space. Finally I will be able to formulate the problem as

$$\mathbf{U_z}(z) = \mathbf{F_{u_z}}(z)\,\mathbf{M_z}(z) \;\;\Rightarrow\;\; \|\mathbf{u}\| \le \left\| \hat{\mathbf{F}}_\mathbf{u} \right\|_\infty \cdot \|\mathbf{m}\|$$

$$\mathbf{X_z}(z) = \mathbf{F_{x_z}}(z)\,\mathbf{M_z}(z) \quad \Rightarrow \quad \|\mathbf{x}\| \le \left\|\hat{\mathbf{F}}_{\mathbf{x}}\right\|_{\infty} \cdot \|\mathbf{m}\|$$

I will also determine the maximum for the states of the system which can be quite useful in view of structural displacements.
Taking the control law in the form of (6.2) or (6.4) as my starting point:

$$\begin{aligned}
\mathbf{U_z}(z) &= \mathbf{F3}\,[\mathbf{F4}\,\mathbf{X_z}(z) + \mathbf{GF_z}(z)\,\mathbf{M_z}(z)] \\
&= \mathbf{F3}\,\mathbf{F4}\,\mathbf{X_z}(z) + \mathbf{F3}\,\mathbf{GF_z}(z)\,\mathbf{M_z}(z) \\
&= \mathbf{K}\mathbf{X_z}(z) + \mathbf{F3}\,\mathbf{GF_num_z}(z)\,z^{-(n-1)}\mathbf{M_z}(z) \\
&= \mathbf{K}\mathbf{X_z}(z) + \mathbf{F3}\,\mathbf{GF_num_z}(z)\,\mathbf{G_z}(z) \\
\mathbf{U_z}(z) &= \mathbf{K}\mathbf{X_z}(z) + \sum_{i=-1}^{n-1} \mathbf{B_i} z^i \mathbf{G_z}(z) \\
&\quad \mathbf{B_i} = \mathbf{F3}\,\mathbf{GF_num_i}
\end{aligned}$$

where $\mathbf{GF\_num_i}$ are the coefficients of the numerator polynomial of $\mathbf{GF_z}(z)$ for the denominator being z^{n-1}. The inverse z-transformation leads to

$$\mathbf{u_k} = \mathbf{K}\mathbf{x_k} + \sum_{i=-1}^{n-1} \mathbf{B_i}\mathbf{g_{k+i}} \tag{6.5}$$

As mentioned above, each of the "systems" $\mathbf{F_{u_z}}(z)$ and $\mathbf{F_{x_z}}(z)$ has to be described in state space, which requires to introduce new states for all known gust data. The corresponding dynamic matrix $\mathbf{\Phi_g}$ is a kind of shift register.

$$\begin{aligned}
\mathbf{x_{g_{k+1}}} &= \mathbf{\Phi_g}\mathbf{x_{g_k}} + \mathbf{B_g}\mathbf{m_k} \\
\mathbf{x_{g_{k+1}}} &= \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & & \mathbf{0} \\ \vdots & & & & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}}_{\mathbf{A_{22}}} \begin{bmatrix} \mathbf{g_{k-1}} \\ \mathbf{g_k} \\ \mathbf{g_{k+1}} \\ \vdots \\ \mathbf{g_{k+n-1}} \end{bmatrix} + \underbrace{\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{I} \end{bmatrix}}_{\mathbf{B_g}} \mathbf{m_k}
\end{aligned}$$

Now, I will transform the system equation using (6.5):

$$\begin{aligned}
\mathbf{x_{k+1}} &= \mathbf{\Phi}\mathbf{x_k} + \mathbf{\Gamma}\mathbf{u_k} + \mathbf{\Theta}\mathbf{g_k} \\
&= \mathbf{\Phi}\mathbf{x_k} + \mathbf{\Gamma}\mathbf{K}\mathbf{x_k} + \mathbf{\Gamma}\sum_{i=-1}^{n-1} \mathbf{B_i}\mathbf{g_{k+1}} + \mathbf{\Theta}\mathbf{g_k} \\
\mathbf{x_{k+1}} &= \underbrace{(\mathbf{\Phi} + \mathbf{\Gamma}\mathbf{K})}_{\mathbf{A_{11}}}\mathbf{x_k} + \underbrace{\begin{bmatrix} \mathbf{\Gamma}\mathbf{B_{-1}} & (\mathbf{\Gamma}\mathbf{B_0} + \mathbf{\Theta}) & \mathbf{\Gamma}\mathbf{B_1} & \cdots & \mathbf{\Gamma}\mathbf{B_{n-1}} \end{bmatrix}}_{\mathbf{A_{12}}} \mathbf{x_{g_{k+1}}}
\end{aligned}$$

The resulting fictitious system having $\mathbf{m_k}$ as input reads:

$$\underbrace{\begin{bmatrix} \mathbf{x_{k+1}} \\ \mathbf{x_{g_{k+1}}} \end{bmatrix}}_{\mathbf{x_{all_{k+1}}}} = \underbrace{\begin{bmatrix} \mathbf{A_{11}} & \mathbf{A_{12}} \\ \mathbf{O} & \mathbf{A_{22}} \end{bmatrix}}_{\mathbf{A_{all}}} \underbrace{\begin{bmatrix} \mathbf{x_k} \\ \mathbf{x_{g_k}} \end{bmatrix}}_{\mathbf{x_{all_k}}} + \underbrace{\begin{bmatrix} \mathbf{O} \\ \mathbf{B_g} \end{bmatrix}}_{\mathbf{B_{all}}} \mathbf{m_k} \tag{6.6}$$

The output "y" of the system will be the vector for which I want to determine the maximum norm.

- for $\mathbf{x_k}$

$$\begin{aligned} \mathbf{y_{all_k}} &:= \mathbf{x_k} = \underbrace{\begin{bmatrix} \mathbf{I} & \mathbf{O} \end{bmatrix}}_{\mathbf{C_{all_x}}} \mathbf{x_{all_k}} \\ \mathbf{D_{all_x}} &= \mathbf{O} \end{aligned}$$

- for $\mathbf{u_k}$

$$\begin{aligned} \mathbf{y_{all_k}} &:= \mathbf{u_k} = \mathbf{K x_k} + \begin{bmatrix} \mathbf{B_{-1}} & \mathbf{B_0} & \mathbf{B_1} & \cdots & \mathbf{B_{n-1}} \end{bmatrix} \mathbf{x_{g_k}} \\ \mathbf{y_{all_k}} &= \underbrace{\begin{bmatrix} \mathbf{K} & \mathbf{B_{-1}} & \mathbf{B_0} & \mathbf{B_1} & \cdots & \mathbf{B_{n-1}} \end{bmatrix}}_{\mathbf{C_{all_u}}} \mathbf{x_{all_k}} \\ \mathbf{D_{all_u}} &= \mathbf{O} \end{aligned}$$

These matrices allow to make use of a very efficient function of the robust control toolbox:

```
sysx=ss(Aall,Ball,Callx,Dallx,T);
sysu=ss(Aall,Ball,Callu,Dallu,T);
normx=normhinf(sysx)
normu=normhinf(sysu)
```

The whole m-file (*Hinf_calc.m*) is included in the appendix. Finally I obtain for the boundaries of $\mathbf{x}$ and $\mathbf{u}$:

$$\begin{aligned} \|\mathbf{u}\| &\leq \left\|\hat{\mathbf{F}}_\mathbf{u}\right\|_\infty \cdot \|\mathbf{m}\| \\ \|\mathbf{x}\| &\leq \left\|\hat{\mathbf{F}}_\mathbf{x}\right\|_\infty \cdot \|\mathbf{m}\| \end{aligned}$$

where $\left\|\hat{\mathbf{F}}_\mathbf{u}\right\|_\infty = \texttt{normu}$ and $\left\|\hat{\mathbf{F}}_\mathbf{x}\right\|_\infty = \texttt{normx}$.

Now, the question is how good these "upper bounds" of the above stated inequalities are with respect to my application. Therefore I would like to review the results for a general case:

$$\mathbf{Y_z}(z) = \mathbf{F_z}(z)\,\mathbf{U_z}(z) \Rightarrow \|\mathbf{y}\| \leq \left\|\hat{\mathbf{F}}\right\|_\infty \cdot \|\mathbf{u}\|$$

I have to assume that my input signal is band limited:

$$\hat{\mathbf{U}}(\omega) = \mathbf{0} \quad \forall\, \omega > \omega_{max}$$

I would like to give the definition of the H-infinity norm again:

$$\left\|\hat{\mathbf{F}}\right\|_\infty = \sup_\omega \bar{\sigma}\left(\hat{\mathbf{F}}(e^{j\omega})\right) = \bar{\sigma}\left(\hat{\mathbf{F}}(e^{j\omega_{sup}})\right)$$

In case ω_{sup} is greater than ω_{max}, the inequality for $\|\mathbf{y}\|$ is a very rough estimate and a constant c exists such that

$$\begin{aligned} \|\mathbf{y}\| &\leq c \cdot \|\mathbf{u}\| \\ c &< \left\|\hat{\mathbf{F}}\right\|_\infty \end{aligned}$$

Thus $\|\hat{\mathbf{F}}\|_\infty$ is not the supremum of c for this particular input signal. Obviously, it is necessary to regard the results provided by *Hinf_calc.m* in view of the application. If the eigenfrequencies of the controlled system are within the bandwidth of the input signal (gust) the H-infinity norms are a good estimation.

6.3.4 Performance indices by means of H-infinity

With respect to evaluation of the controller designs it is also interesting to compare similar H_∞ norms for acceleration and jerk, respectively, of the controlled system and the system without the controller. These norms can also help to guarantee that specifications e.g. for maximum acceleration are observed.

$$\begin{array}{rcl@{\quad}l} \mathbf{A_{n_z}}(z) = \mathbf{F_{Ac_z}}(z)\,\mathbf{M_z}(z) & \Rightarrow & \|\mathbf{a_n}\| \leq \|\hat{\mathbf{F}}_{\mathbf{an}}\|_\infty \cdot \|\mathbf{m}\| & \text{no contr.} \\ \mathbf{A_{c_z}}(z) = \mathbf{F_{An_z}}(z)\,\mathbf{M_z}(z) & \Rightarrow & \|\mathbf{a_c}\| \leq \|\hat{\mathbf{F}}_{\mathbf{ac}}\|_\infty \cdot \|\mathbf{m}\| & \text{controlled} \\ \mathbf{J_{n_z}}(z) = \mathbf{F_{Jc_z}}(z)\,\mathbf{M_z}(z) & \Rightarrow & \|\mathbf{j_n}\| \leq \|\hat{\mathbf{F}}_{\mathbf{jn}}\|_\infty \cdot \|\mathbf{m}\| & \text{no contr.} \\ \mathbf{J_{c_z}}(z) = \mathbf{F_{Jn_z}}(z)\,\mathbf{M_z}(z) & \Rightarrow & \|\mathbf{j_c}\| \leq \|\hat{\mathbf{F}}_{\mathbf{jc}}\|_\infty \cdot \|\mathbf{m}\| & \text{controlled} \end{array}$$

These calculations are also part of *Hinf_calc.m* and use the fictitious system described by $\mathbf{A_{all}}$, $\mathbf{B_{all}}$ and $\mathbf{x_{all_k}}$ for the controlled system norms. As for

the system without a controller, I replace $\mathbf{A_{all}}$ by $\tilde{\mathbf{A}}_{\mathbf{all}}$. I will set $\mathbf{a} = \mathbf{L}\dot{\mathbf{x}}$ for the first and $\mathbf{j} = \mathbf{L}\,\ddot{\mathbf{w}}\mathbf{2}$ for the second approach. $\mathbf{L}$ selects the states related to the fuselage as this is in line with my design goal, $\|\mathbf{L}\| = 1$ is assumed.

$$\begin{aligned}
\mathbf{x_{g_{k+1}}} &= \underbrace{\mathbf{\Phi_g}}_{\tilde{\mathbf{A}}_{22}}\mathbf{x_{g_k}} + \mathbf{B_g m_k} \qquad \text{as for the controlled case} \\
\mathbf{x_{k+1}} &= \mathbf{\Phi x_k} + \mathbf{\Theta g_k} \\
\mathbf{x_{k+1}} &= \underbrace{\mathbf{\Phi}}_{\tilde{\mathbf{A}}_{11}}\mathbf{x_k} + \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{\Theta} & \mathbf{O} \end{bmatrix}}_{\tilde{\mathbf{A}}_{12}}\mathbf{x_{g_{k+1}}}
\end{aligned}$$

Therefore, for the system without a controller I obtain:

$$\underbrace{\begin{bmatrix} \mathbf{x_{k+1}} \\ \mathbf{x_{g_{k+1}}} \end{bmatrix}}_{\mathbf{x_{all_{k+1}}}} = \underbrace{\begin{bmatrix} \tilde{\mathbf{A}}_{11} & \tilde{\mathbf{A}}_{12} \\ \mathbf{O} & \tilde{\mathbf{A}}_{22} \end{bmatrix}}_{\tilde{\mathbf{A}}_{\mathbf{all}}} \underbrace{\begin{bmatrix} \mathbf{x_k} \\ \mathbf{x_{g_k}} \end{bmatrix}}_{\mathbf{x_{all_k}}} + \underbrace{\begin{bmatrix} \mathbf{O} \\ \mathbf{B_g} \end{bmatrix}}_{\mathbf{B_{all}}} \mathbf{m_k} \tag{6.7}$$

Now, I can determine the various $\mathbf{C}$ and $\mathbf{D}$ matrices so that the output of the fictitious transfer function equals the variable for which I want to calculate an upper bound.

- accelerations of the system without a controller

$$\begin{aligned}
\mathbf{a_{n_k}} &= \mathbf{L}\dot{\mathbf{x}}_\mathbf{k} \\
&= \mathbf{L}\left(\mathbf{Ax_k} + \mathbf{Gg_k}\right) \\
\mathbf{y_{all_k}} &= \underbrace{\begin{bmatrix} \mathbf{LA} & \mathbf{0} & \mathbf{LG} & \mathbf{O} \end{bmatrix}}_{\mathbf{C_{all_{an}}}}\mathbf{x_{all_k}} \\
\mathbf{D_{all_{an}}} &= \mathbf{O}
\end{aligned}$$

  Use $\tilde{\mathbf{A}}_{\mathbf{all}}$ for the dynamic matrix.

- accelerations of the system with the controller

$$\begin{aligned}
\mathbf{a_{c_k}} &= \mathbf{L}\dot{\mathbf{x}}_\mathbf{k} \\
&= \mathbf{L}\left(\mathbf{Ax_k} + \mathbf{Bu_k} + \mathbf{Gg_k}\right) \\
&= \mathbf{LAx_k} + \mathbf{LB}\left(\mathbf{Kx_k} + \begin{bmatrix} \mathbf{B_{-1}} & \mathbf{B_0} & \mathbf{B_1} & \cdots & \mathbf{B_{n-1}} \end{bmatrix}\mathbf{x_{g_k}}\right) + \mathbf{LGg_k} \\
\mathbf{y_{all_k}} &= \underbrace{\mathbf{L}\begin{bmatrix} (\mathbf{A}+\mathbf{BK}) & \mathbf{BB_{-1}} & (\mathbf{BB_0}+\mathbf{G}) & \mathbf{BB_1} & \cdots & \mathbf{BB_{n-1}} \end{bmatrix}}_{\mathbf{C_{all_{ac}}}}\mathbf{x_{all_k}} \\
\mathbf{D_{all_{ac}}} &= \mathbf{O}
\end{aligned}$$

  Use $\mathbf{A_{all}}$ for the dynamic matrix.

- jerkiness of the system without a controller

$$
\begin{aligned}
\mathbf{j_{n_k}} &= \mathbf{L}\,\dddot{\mathbf{w2}}_\mathbf{k} \\
\ddot{\mathbf{w2}} &= -\mathbf{M2}^{-1}\begin{bmatrix}\mathbf{K2} & \mathbf{D2} & \mathbf{O}\end{bmatrix}\mathbf{x_k} \\
\dddot{\mathbf{w2}} &= -\mathbf{M2}^{-1}\begin{bmatrix}\mathbf{K2} & \mathbf{D2} & \mathbf{O}\end{bmatrix}\dot{\mathbf{x}}_\mathbf{k} \\
\mathbf{j_{n_k}} &= -\mathbf{L}\,\mathbf{M2}^{-1}\begin{bmatrix}\mathbf{K2} & \mathbf{D2} & \mathbf{O}\end{bmatrix}\dot{\mathbf{x}}_\mathbf{k} \\
\dot{\mathbf{x}}_\mathbf{k} &= \begin{bmatrix}\mathbf{A} & \mathbf{0} & \mathbf{G} & \mathbf{O}\end{bmatrix}\mathbf{x_{all_k}} \\
\mathbf{y_{all_k}} &= \underbrace{-\mathbf{L}\,\mathbf{M2}^{-1}\begin{bmatrix}\mathbf{K2} & \mathbf{D2} & \mathbf{O}\end{bmatrix}\begin{bmatrix}\mathbf{A} & \mathbf{0} & \mathbf{G} & \mathbf{O}\end{bmatrix}}_{\mathbf{C_{all_{jn}}}}\mathbf{x_{all_k}} \\
\mathbf{D_{all_{jn}}} &= \mathbf{O}
\end{aligned}
$$

  Use $\tilde{\mathbf{A}}_\mathbf{all}$ for the dynamic matrix.

- jerkiness of the system with the controller

$$
\begin{aligned}
\mathbf{j_{c_k}} &= \mathbf{L}\,\dddot{\mathbf{w2}}_\mathbf{k} \\
\dddot{\mathbf{w2}}_\mathbf{k} &= \mathbf{R}\dot{\mathbf{x}}_\mathbf{k} + \mathbf{G2}\dot{\mathbf{g}}_\mathbf{k} \\
&\approx \mathbf{R}\mathbf{A}\mathbf{x_k} + \mathbf{R}\mathbf{B}\mathbf{u_k} + \tilde{\mathbf{G}}\begin{bmatrix}\mathbf{g_k} \\ \mathbf{g_{k-1}}\end{bmatrix} \\
&= \mathbf{R}(\mathbf{A}+\mathbf{B}\mathbf{K})\,\mathbf{x_k} + \mathbf{R}\mathbf{B}\begin{bmatrix}\mathbf{B_{-1}} & \mathbf{B_0} & \mathbf{B_1} & \cdots & \mathbf{B_{n-1}}\end{bmatrix}\mathbf{x_{g_k}} + \\
&\quad + \tilde{\mathbf{G}}\begin{bmatrix}\mathbf{I} \\ \mathbf{O}\end{bmatrix}\mathbf{g_k} + \tilde{\mathbf{G}}\begin{bmatrix}\mathbf{O} \\ \mathbf{I}\end{bmatrix}\mathbf{g_{k-1}} \\
\dddot{\mathbf{w2}}_\mathbf{k} &\approx \Bigg[\mathbf{R}(\mathbf{A}+\mathbf{B}\mathbf{K}) \quad \left(\mathbf{R}\mathbf{B}\mathbf{B_{-1}} + \tilde{\mathbf{G}}\begin{bmatrix}\mathbf{O} & \mathbf{I}\end{bmatrix}^T\right) \quad \left(\mathbf{R}\mathbf{B}\mathbf{B_0} + \tilde{\mathbf{G}}\begin{bmatrix}\mathbf{I} & \mathbf{O}\end{bmatrix}^T\right) \\
&\qquad \mathbf{R}\mathbf{B}\mathbf{B_1} \quad \cdots \quad \mathbf{R}\mathbf{B}\mathbf{B_{n-1}}\Bigg]\mathbf{x_{all_k}} \\
\mathbf{y_{all_k}} &= \mathbf{L}\Bigg[\mathbf{R}(\mathbf{A}+\mathbf{B}\mathbf{K}) \quad \left(\mathbf{R}\mathbf{B}\mathbf{B_{-1}} + \tilde{\mathbf{G}}\begin{bmatrix}\mathbf{O} & \mathbf{I}\end{bmatrix}^T\right) \quad \left(\mathbf{R}\mathbf{B}\mathbf{B_0} + \tilde{\mathbf{G}}\begin{bmatrix}\mathbf{I} & \mathbf{O}\end{bmatrix}^T\right) \\
&\qquad \mathbf{R}\mathbf{B}\mathbf{B_1} \quad \cdots \quad \mathbf{R}\mathbf{B}\mathbf{B_{n-1}}\Bigg]\mathbf{x_{all_k}} \\
\mathbf{y_{all_k}} &= \mathbf{C_{all_{jc}}}\mathbf{x_{all_k}} \\
\mathbf{D_{all_{jc}}} &= \mathbf{O}
\end{aligned}
$$

  Use $\mathbf{A}_\mathbf{all}$ for the dynamic matrix.

The difference between $\|\hat{\mathbf{F}}_{\mathbf{an}}\|_\infty$ and $\|\hat{\mathbf{F}}_{\mathbf{ac}}\|_\infty$ as well as $\|\hat{\mathbf{F}}_{\mathbf{jn}}\|_\infty$ and $\|\hat{\mathbf{F}}_{\mathbf{jc}}\|_\infty$, respectively, provides information about the beneficial effect of the particular controller for the worst case of gust. I will divide these differences by the norm for the controlled case to obtain an index telling how many percent the system behavior is worse without the controller:

$$\text{improvement}(\mathbf{a}) = \frac{\left\|\hat{\mathbf{F}}_{\mathbf{an}}\right\|_\infty - \left\|\hat{\mathbf{F}}_{\mathbf{ac}}\right\|_\infty}{\left\|\hat{\mathbf{F}}_{\mathbf{ac}}\right\|_\infty} \cdot 100\%$$

$$\text{improvement}(\mathbf{j}) = \frac{\left\|\hat{\mathbf{F}}_{\mathbf{jn}}\right\|_\infty - \left\|\hat{\mathbf{F}}_{\mathbf{jc}}\right\|_\infty}{\left\|\hat{\mathbf{F}}_{\mathbf{jc}}\right\|_\infty} \cdot 100\%$$

In the previous section, I discussed the meaningfulness of the boundaries calculated using H-infinity norms. Since the performance indices are based on the same theory, I should consider the bandwidth of the input signal for the particular application as well.

6.4 Evaluation module

This subsystem contains the cost functions J_3, J_2 and J_1 or $\tilde{J}_1$ for each test to provide easy to compare results. The scopes show the increase of cost during the experiment. Hereafter is shown the subsystem for the approach of chapter 3 and chapter 4. To calculate the jerkiness and the ride comfort within the second approach I make use of a derivative element. I would like to point out that such an element is not used for the controller itself.

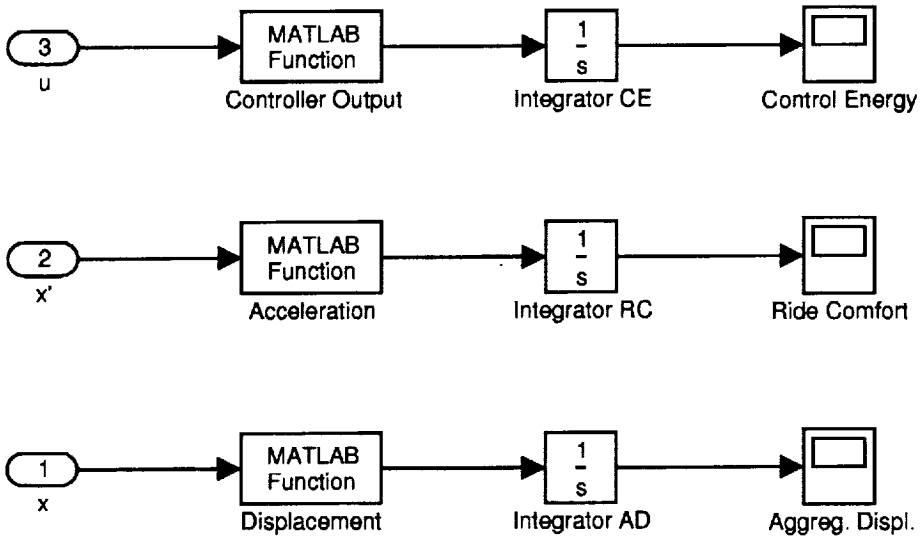

Figure 6.5: The evaluation subsystem for acceleration control

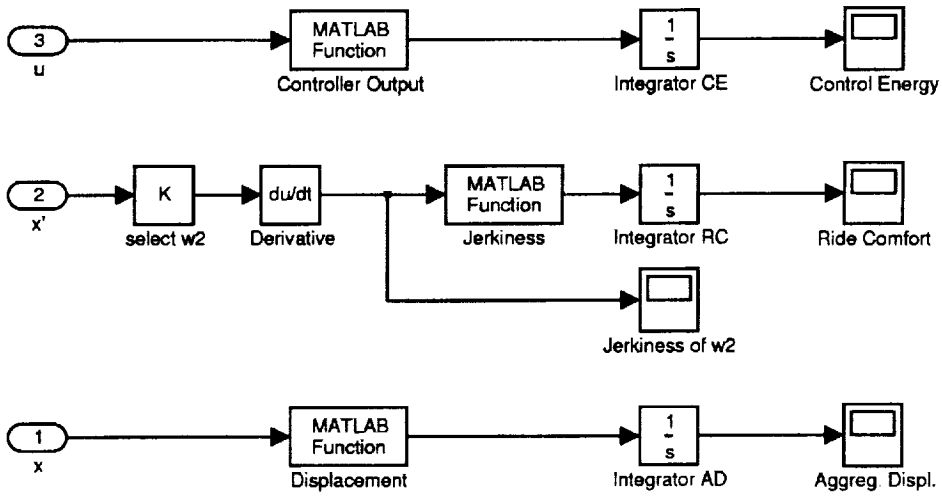

Figure 6.6: The evaluation subsystem for jerkiness control

6.5 M-files

All constants within the evaluation are defined in *const_def.m*, that is: The sytem (described by $\mathbf{A}$, $\mathbf{B}$, $\mathbf{G}$), the weight matrices ($\mathbf{S}$, $\mathbf{Q_1}$, $\mathbf{Q_2}$), the sampling period T, the number of available gust data n, the length of the finite horizon N and the parameter ω_g for the Dryden model of gust.
const_calc_1.m, *const_calc_2.m* respectively, calculate all variables that depend on the definitions in *const_def.m*. The calculation of the various H_∞ norms is implemented in *Hinf_calc.m*. Finally, *pzmap_calc.m* plots the pole-zero map for the system with and without the controller to verify stability. All of these files are included in the appendix.

Chapter 7

Preliminary Simulation

7.1 Test configuration, system and actuator

To examine the performance of the two different controllers and their implementation a very simple oscillator (pendulum) will be subject to control. Its differential equation reads:

$$\ddot{w} + \frac{1}{2}\dot{w} + w = f$$

The step response of the system without a controller is shown in figure 7.1. The disturbance (gust) and the actuator lead to a force f that has influence on the oscillation. Often there is a delay between the occurrence of a disturbance and its effect (here the unwanted oscillation), as well as between the "command" of the controller and the moment the countermeasures really come into effect (because of actuator delays). With regard to this problem information of future disturbance should be able to increase the controller performance. To stress this effect, I chose a slow PT2 actuator. Its transfer function reads:

$$G(S) = \frac{1}{0.04s^2 + 0.2s + 1}$$

The step response is shown in figure 7.2. I also tried a PT1 actuator for the controller dealing with jerkiness, since the difference of order is smaller in that case:

$$G(S) = \frac{1}{s + 1}$$

The step response is shown in figure 7.3.

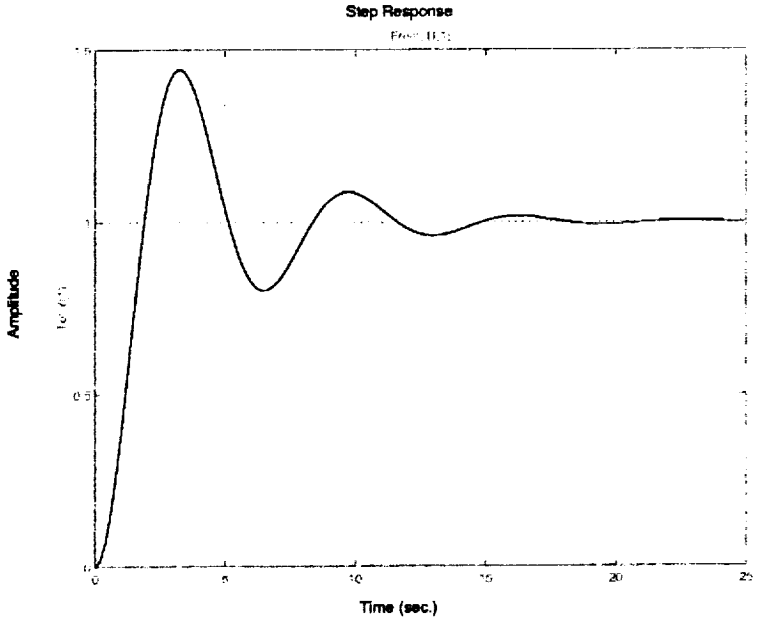

Figure 7.1: Step response of the pendulum itself

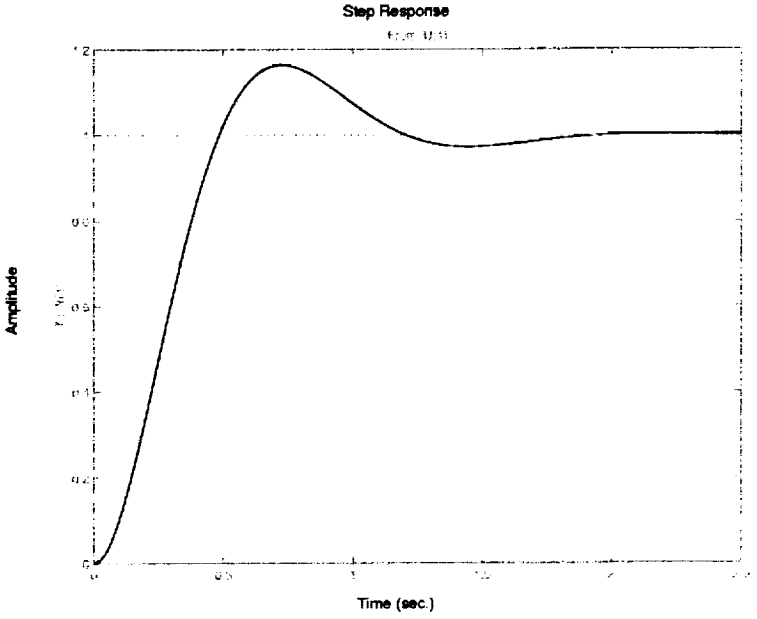

Figure 7.2: Step response of the PT2 actuator

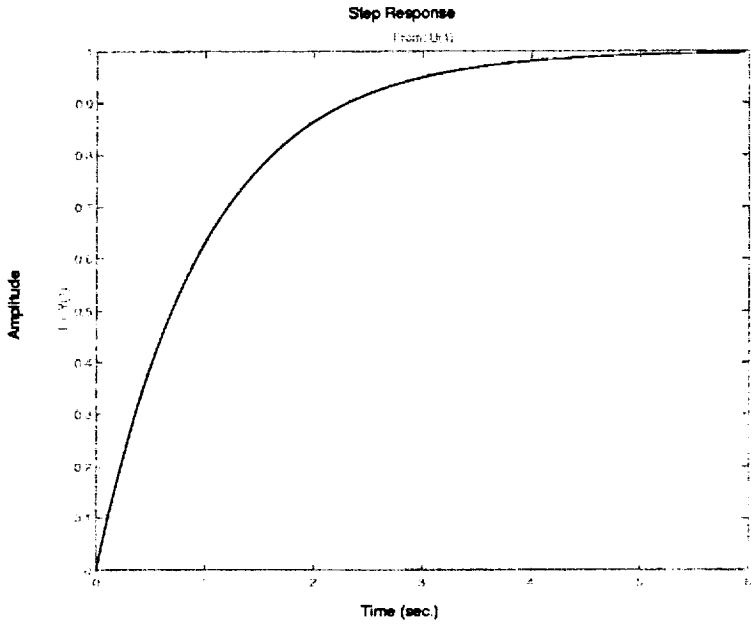

Figure 7.3: Step response of the PT1 actuator

7.2 Minimization of accelerations

Since this approach is not critical regarding the disturbance $\mathbf{g_k}$ (or $\mathbf{z_k}$) I can use a simple step function that allows easily comprehensible results to determine whether the control algorithm works as expected.
The main criteria for the comparison of the cases $n = 1$ and $n = 30$ is of course the ride comfort index at the end of the simulation. I tried to keep the magnitude of the controller outputs within the same range, so that the performance does not depend on that. I also calculated the control energy for the sake of completeness, although the energy will be – as opposed to the output range – not be a problem to most applications. The displacements should of course not become too large, even though it is not the purpose of this controller to keep them as small as possible. Obviously, the results are much better, if more gust data are available. Then, as can be seen from figure 7.6, right graph, the controller begins its countermeasures already before the disturbance encounters the system, what leads to lower accelerations.

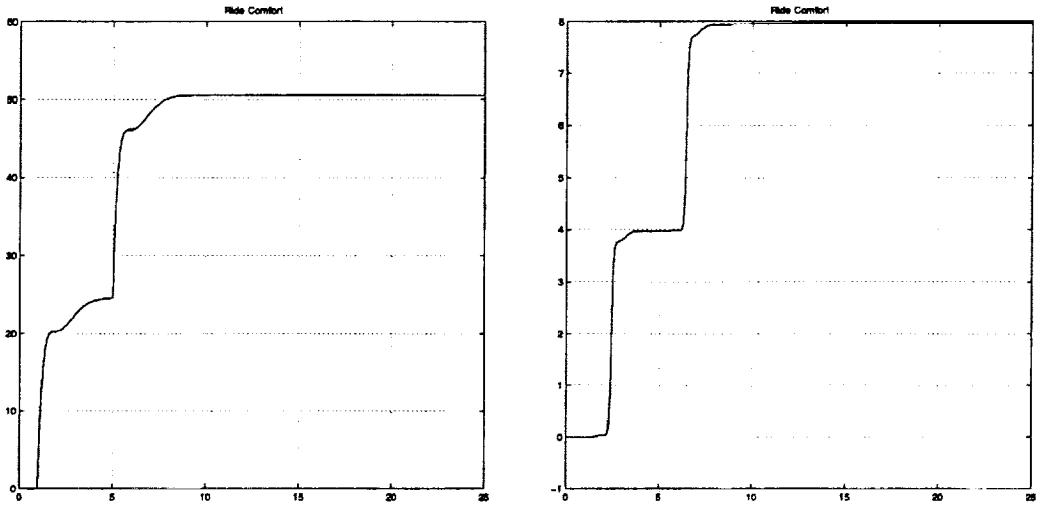

Figure 7.4: "Ride comfort" as the integral of squared accelerations for n=1 and n=30

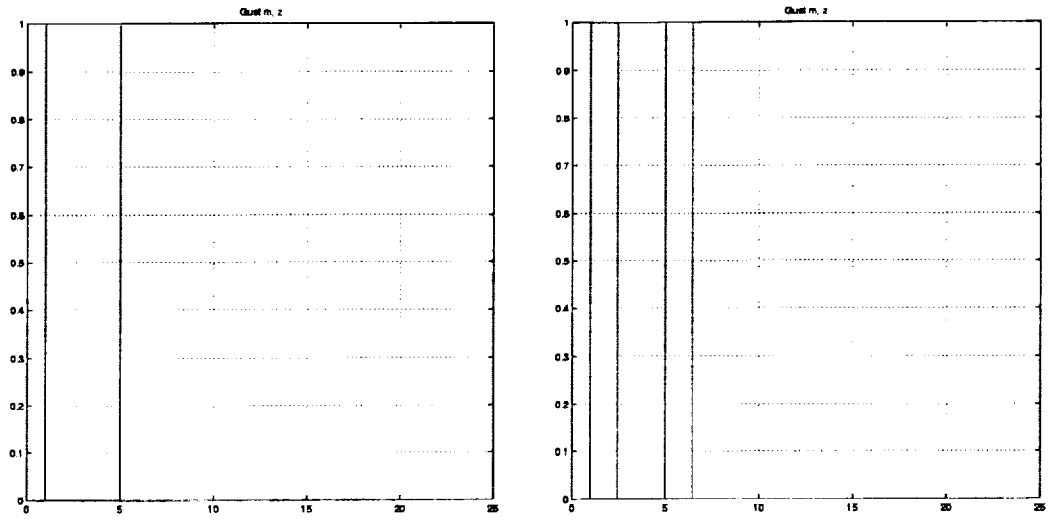

Figure 7.5: Measurement data and disturbance for n=1 and n=30

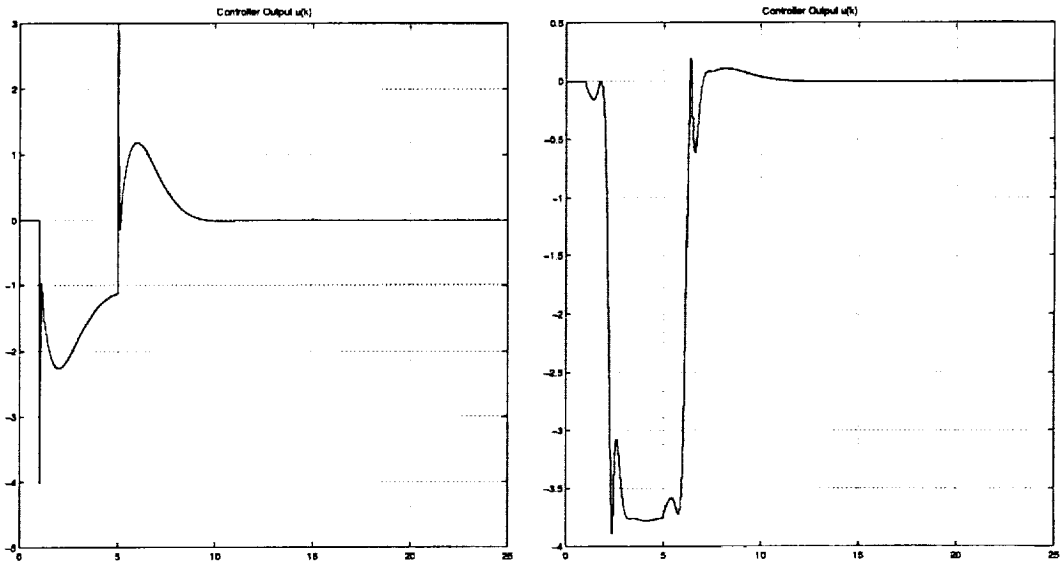

Figure 7.6: Controller output for n=1 and n=30

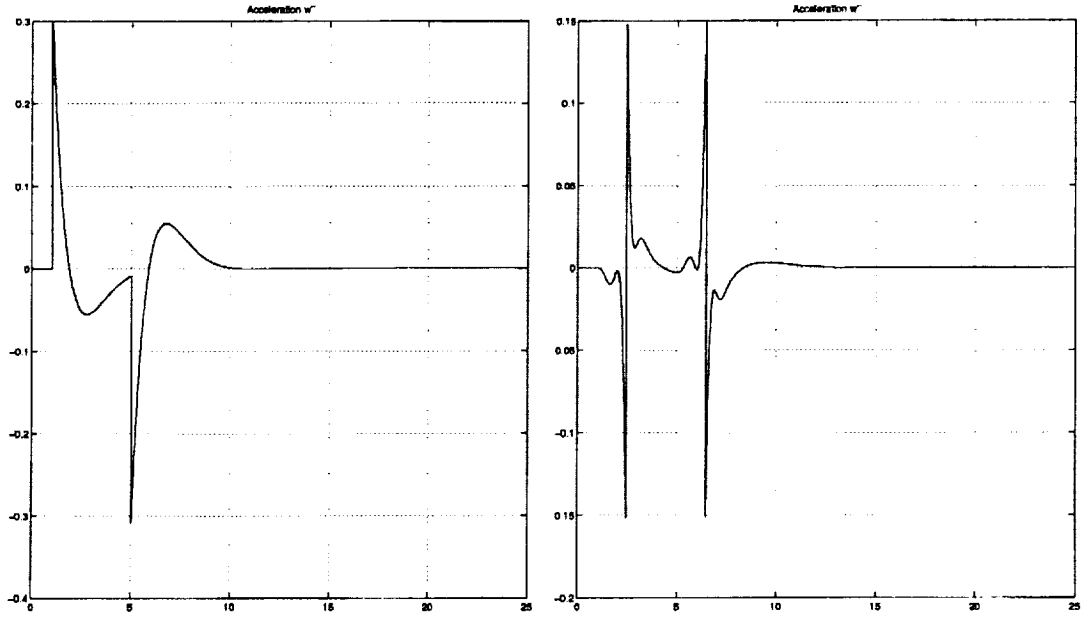

Figure 7.7: Acceleration of the pendulum for n=1 and n=30

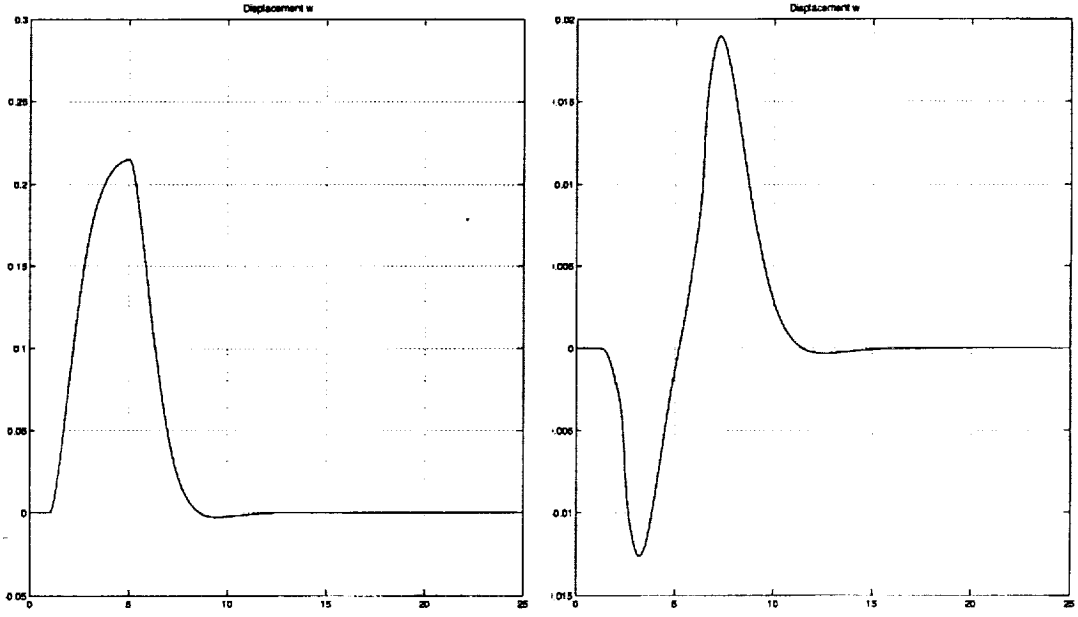

Figure 7.8: Displacement of the pendulum for n=1 and n=30

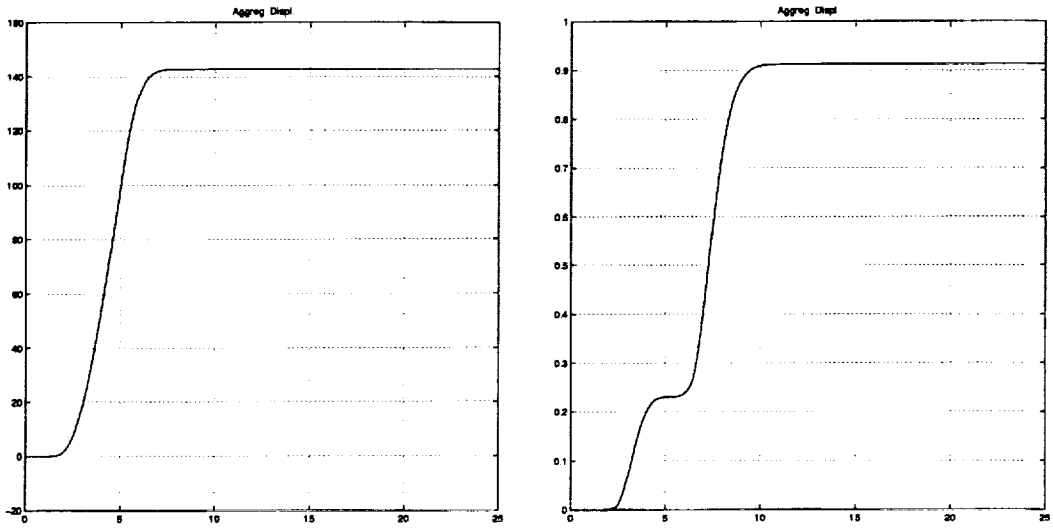

Figure 7.9: "Aggregated displacements" as the integral of squared displacements of the pendulum for n=1 and n=30

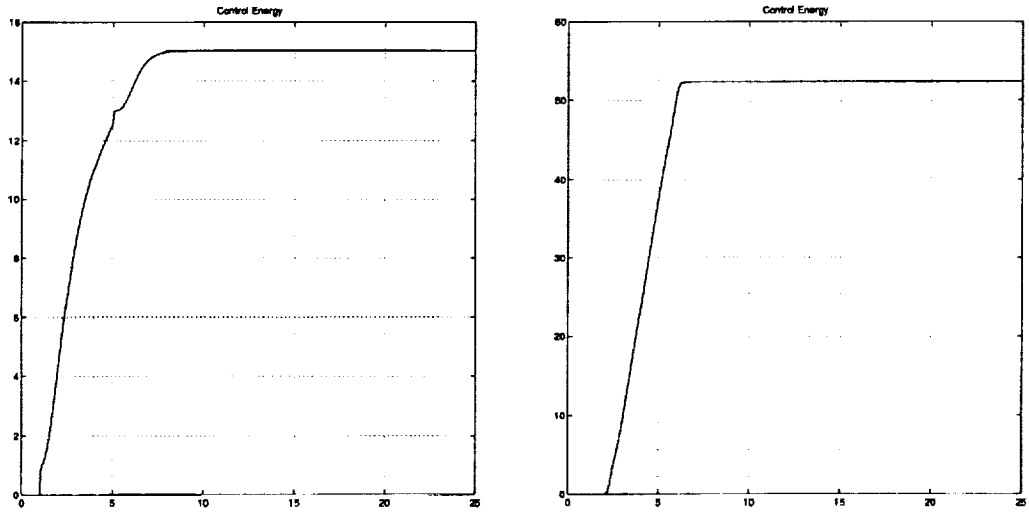

Figure 7.10: Control energy for n=1 and n=30

7.3 Minimization of jerkiness

Dealing with the jerkiness is critical with respect to the time-derivative of the disturbance. It is important that the disturbance function is differentiable for every time. Hence, I choose the first half wave of a $\sin^2(\omega t)$ function, which also has the advantage, that its derivative is zero for $t = 0$.

7.3.1 PT1 actuator

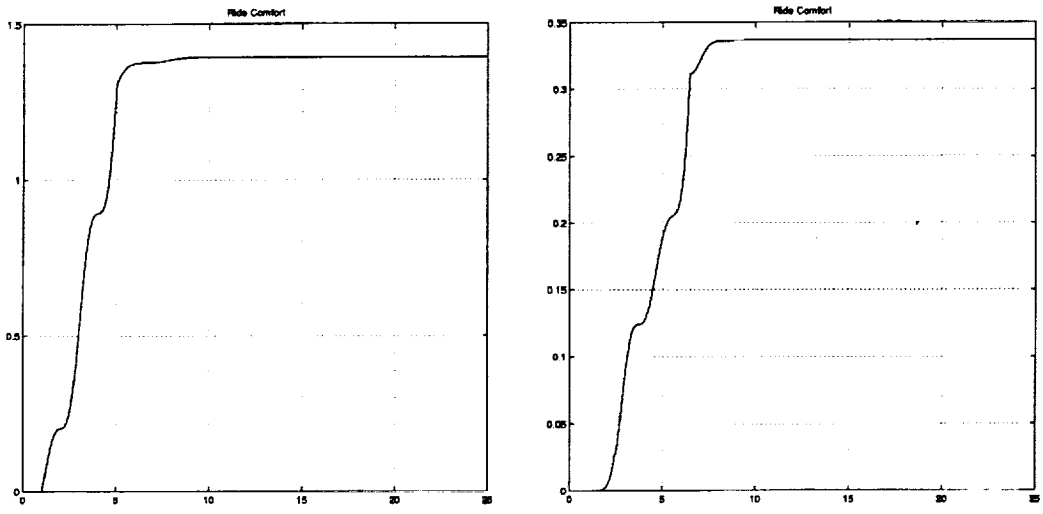

Figure 7.11: "Ride comfort" as the integral of squared jerkiness for n=1 and n=30

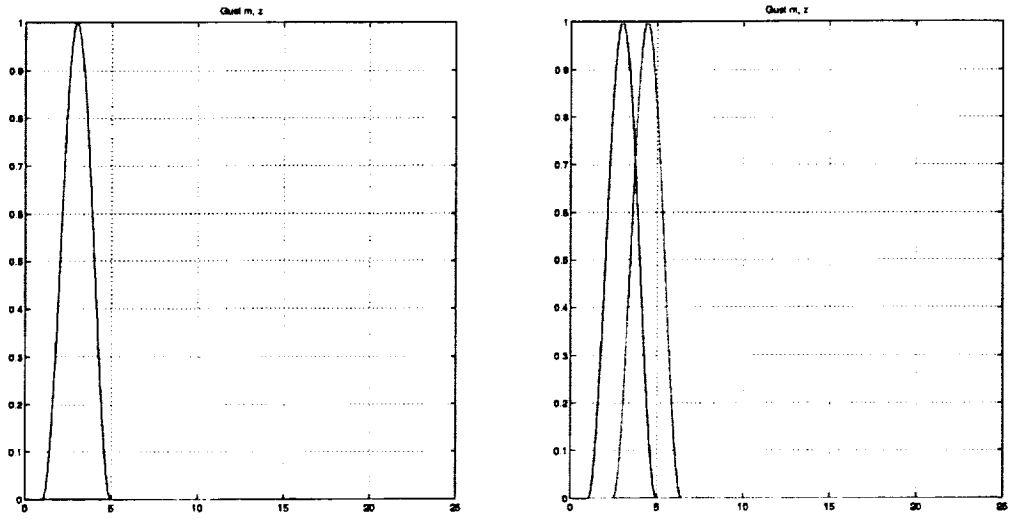

Figure 7.12: Measurement data and disturbance for n=1 and n=30

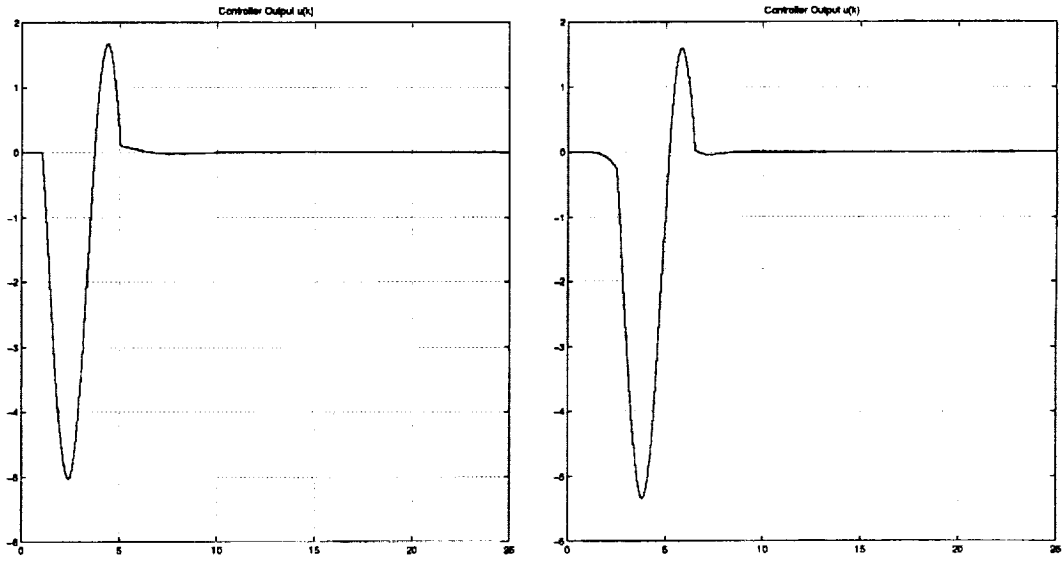

Figure 7.13: Controller output for n=1 and n=30

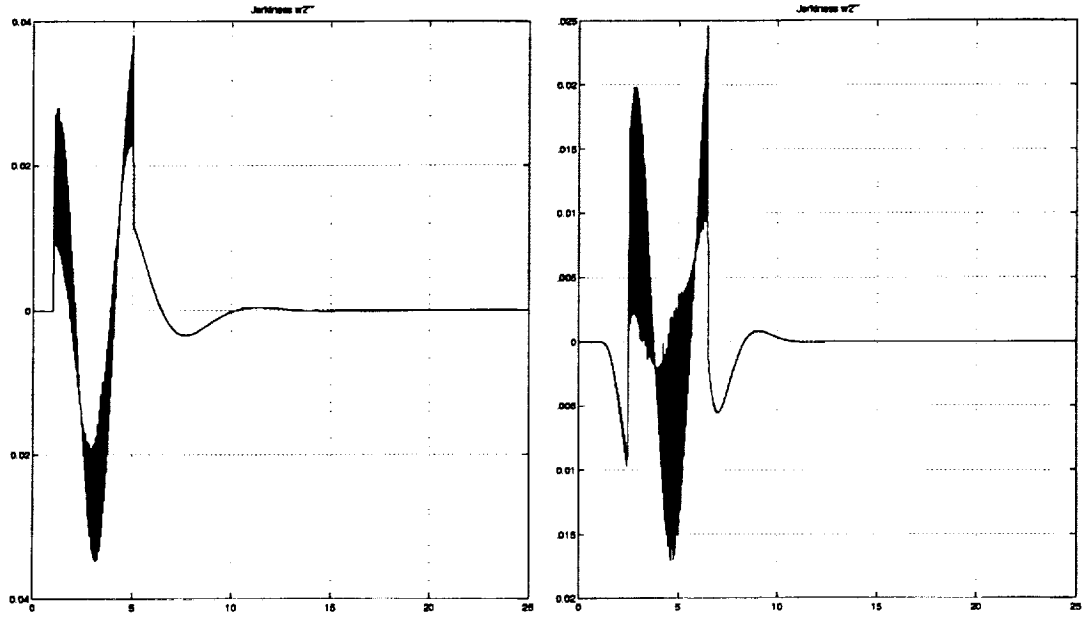

Figure 7.14: Jerkiness of the pendulum for n=1 and n=30

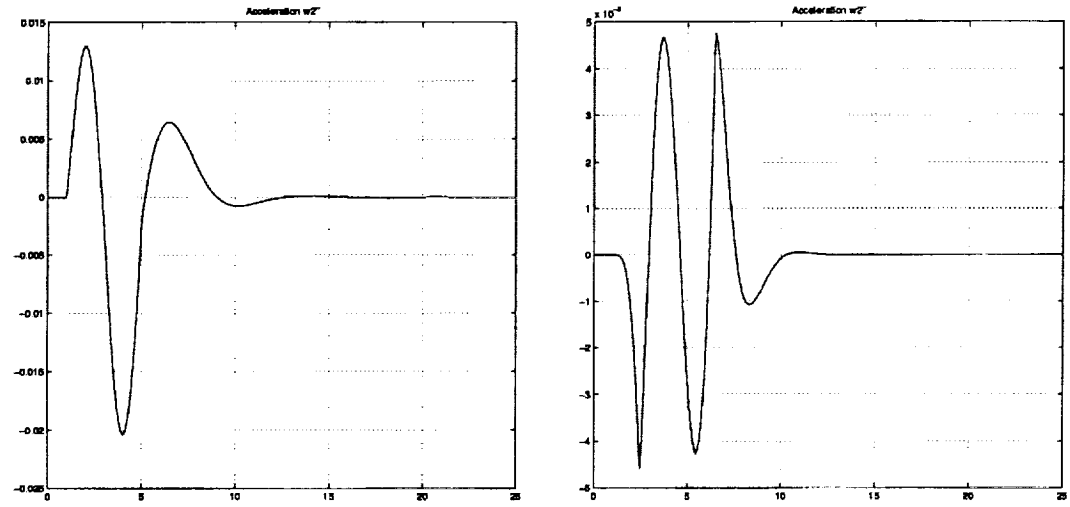

Figure 7.15: Acceleration of the pendulum for n=1 and n=30

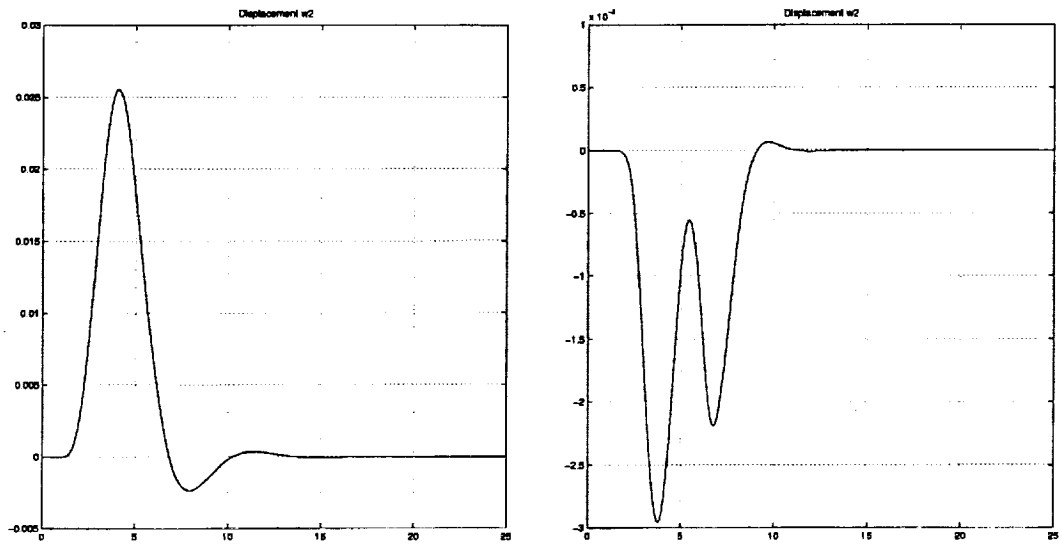

Figure 7.16: Displacement of the pendulum for n=1 and n=30

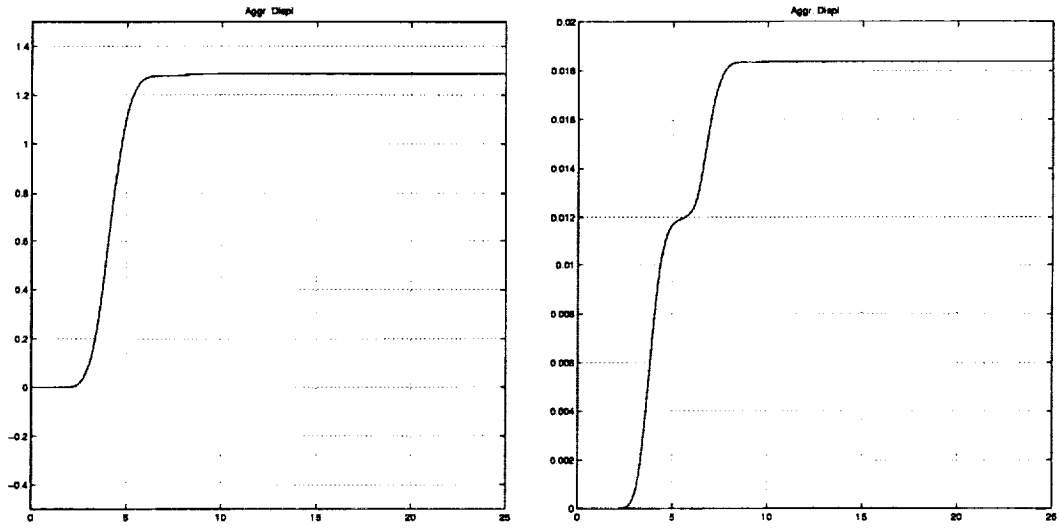

Figure 7.17: "Aggregated displacements" as the integral of squared displacements of the pendulum for n=1 and n=30

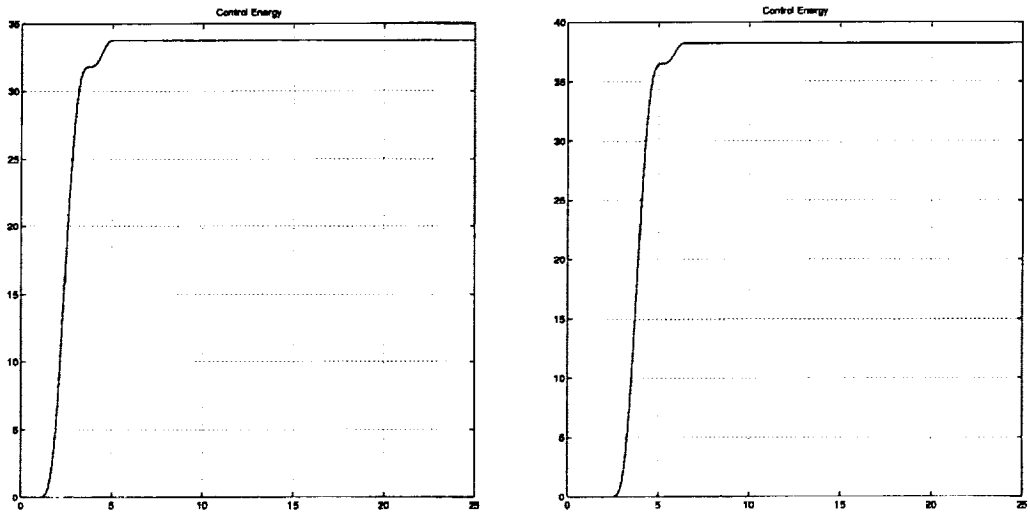

Figure 7.18: Control energy for n=1 and n=30

7.3.2 PT2 actuator

For the jerkiness control with a second order actuator I have to assume at least $n = 2$. The gust filter calculated for this approach (PT1 as well as PT2) is a weighted sum of gust data $\mathbf{m_{k-1}}, \mathbf{m_k}, \ldots, \mathbf{m_{k+n-1}}$ where t_k is the current point in time. As opposed to the case of a first order actuator, here the factorial for $\mathbf{m_{k-1}}$ is always equal to zero, since $\mathbf{B^T R^T} = \mathbf{O}$. In view of the approximation for $\dot{\mathbf{g}}_\mathbf{k}$ I need at least two consecutive gust data. Thus $n = 1$ is not applicable. Also for this simulation, an improvement is visible when increasing the available data.

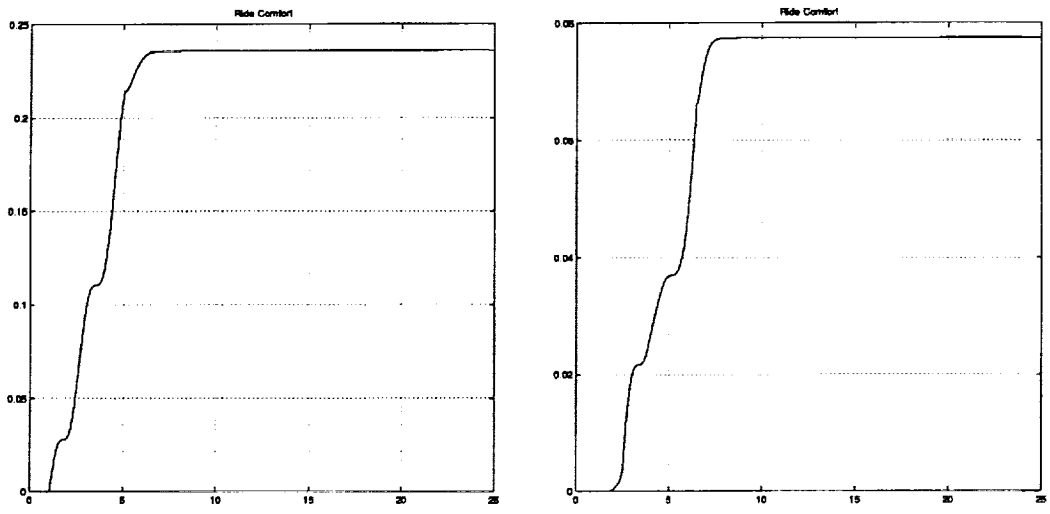

Figure 7.19: "Ride comfort" as the integral of squared jerkiness for n=2 and n=30

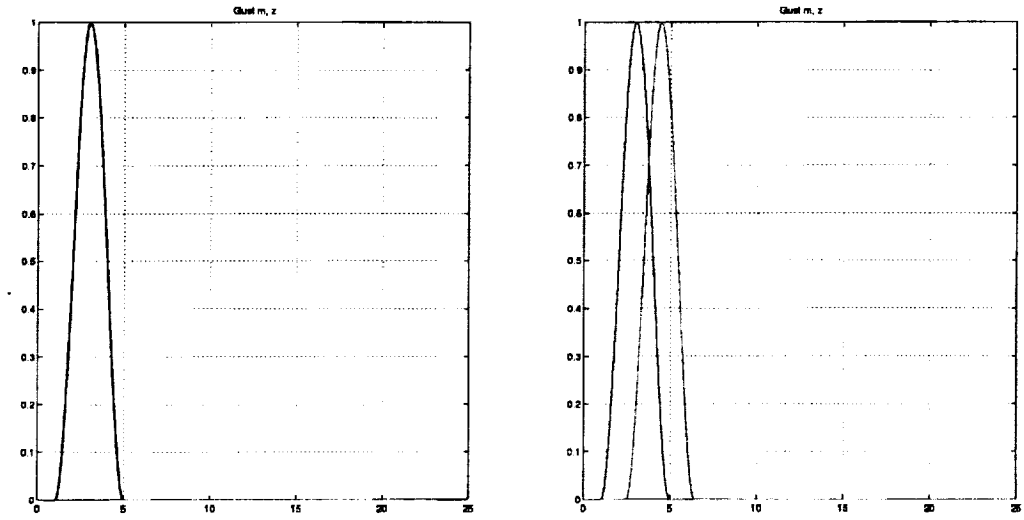

Figure 7.20: Measurement data and disturbance for n=2 and n=30

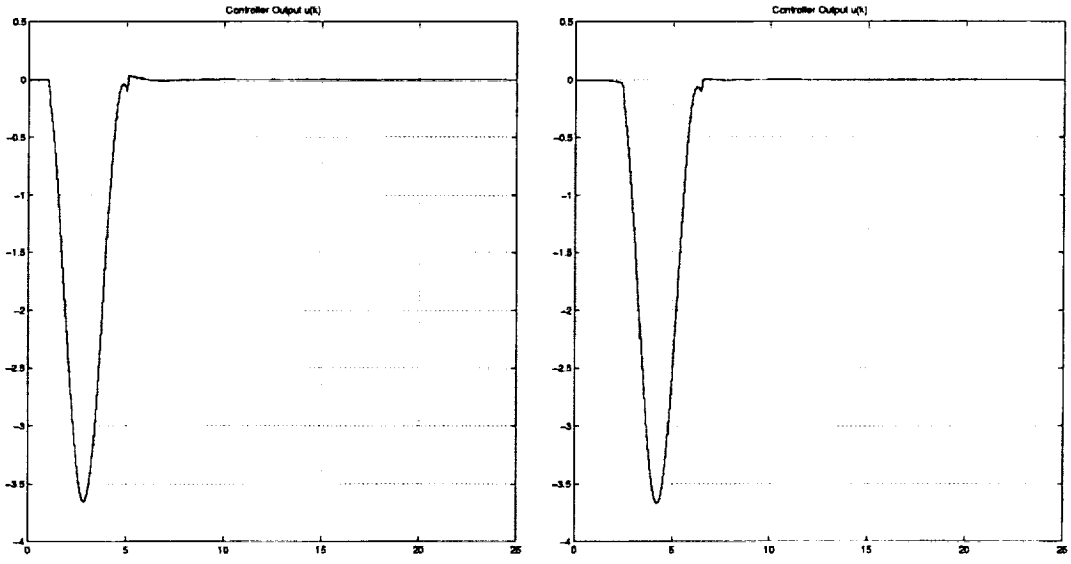

Figure 7.21: Controller output for n=2 and n=30

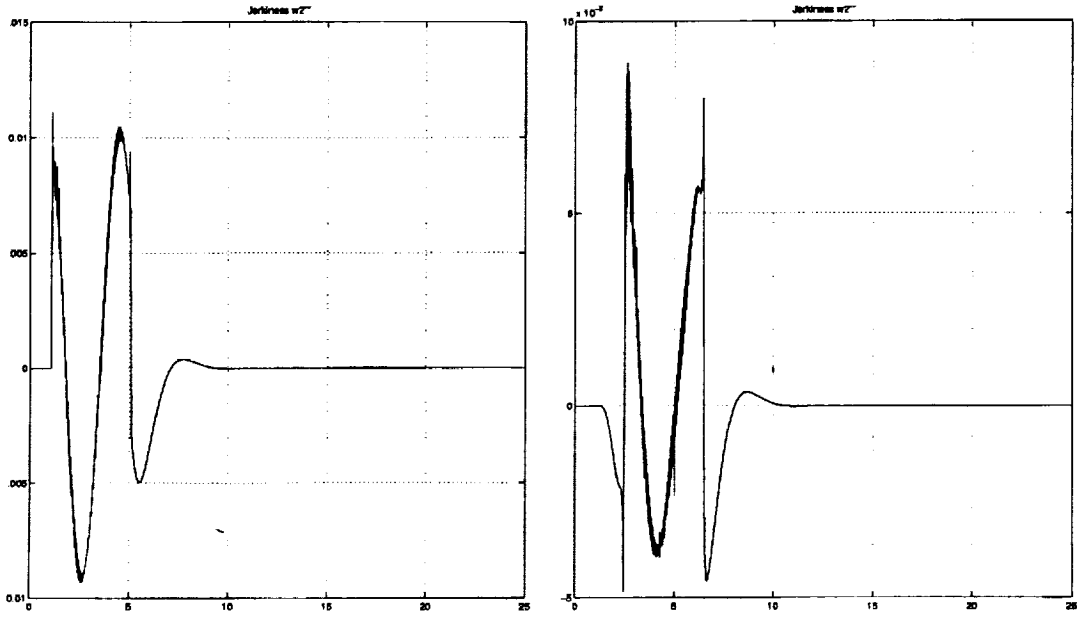

Figure 7.22: Jerkiness of the pendulum for n=2 and n=30

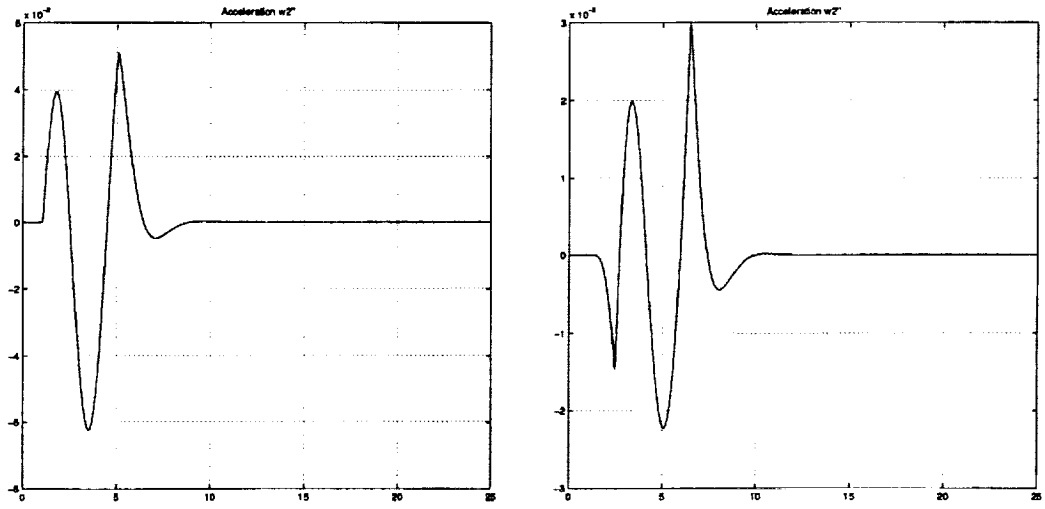

Figure 7.23: Acceleration of the pendulum for n=2 and n=30

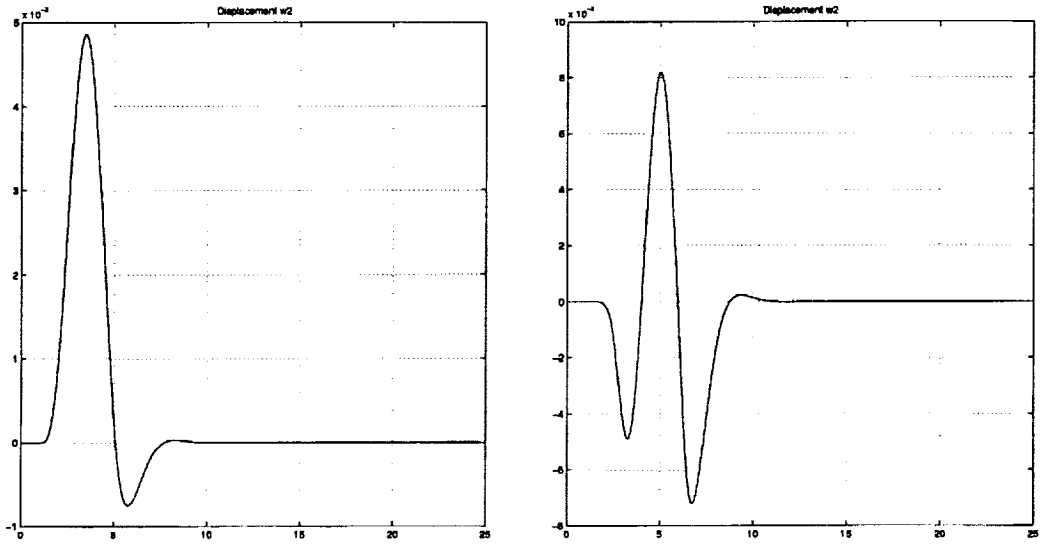

Figure 7.24: Displacement of the pendulum for n=2 and n=30

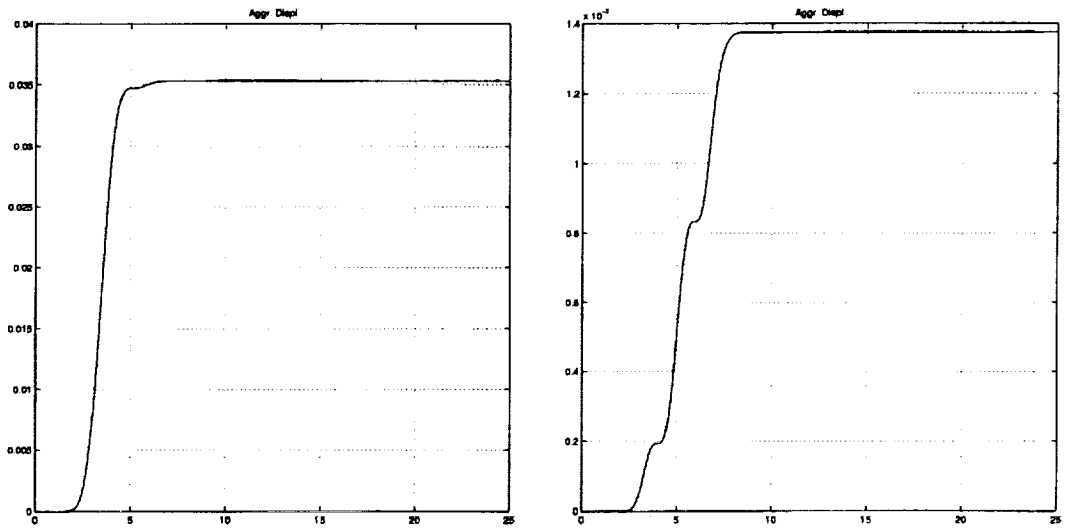

Figure 7.25: "Aggregated displacements" as the integral of squared displacements of the pendulum for n=2 and n=30

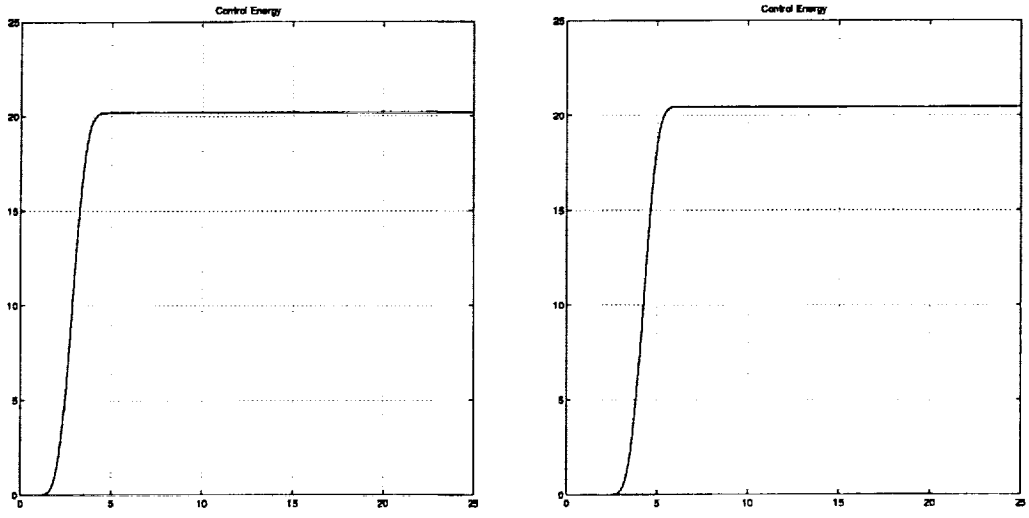

Figure 7.26: Control energy for n=2 and n=30

Chapter 8

Application on a Wing-Fuselage System

8.1 Test configuration, system and actuator

For the main experiment I choose a two degree of freedom system. Of course, with respect to my theory, I could use a more difficult model since neither the number of actuators, nor the degree of freedom is limited. However, this "simplified airplane" consisting of a wing, a fuselage and one first order actuator is sufficient for evaluation, as I am interested in qualitative rather than in exact results. The sketch of figure 8.1 illustrates the variables and constants of the state space representation:

$$\underbrace{\begin{bmatrix} m & 0 \\ 0 & M \end{bmatrix}}_{\mathbf{M_2}} \underbrace{\begin{bmatrix} \ddot{w}_{2_1} \\ \ddot{w}_{2_2} \end{bmatrix}}_{\ddot{\mathbf{w}}_2} + \underbrace{\begin{bmatrix} D_1 & -D_1 \\ -D_1 & D_1 + D_2 \end{bmatrix}}_{\mathbf{D_2}} \underbrace{\begin{bmatrix} \dot{w}_{2_1} \\ \dot{w}_{2_2} \end{bmatrix}}_{\dot{\mathbf{w}}_2} +$$

$$+ \underbrace{\begin{bmatrix} K_1 & -K_1 \\ -K_1 & K_1 + K_2 \end{bmatrix}}_{\mathbf{K_2}} \underbrace{\begin{bmatrix} w_{2_1} \\ w_{2_2} \end{bmatrix}}_{\mathbf{w}_2} = \begin{bmatrix} f_g \\ 0 \end{bmatrix} + \underbrace{\begin{bmatrix} f_a \\ 0 \end{bmatrix}}_{\mathbf{B_2 w_1}}$$

For the determination of the spring and damping constants, I made the following assumptions:

- The mass of the fuselage is 100 times higher than that of the wing which I set to 1.
- The natural frequency (mode) of the wing itself is about 4Hz and the damping is 10%.

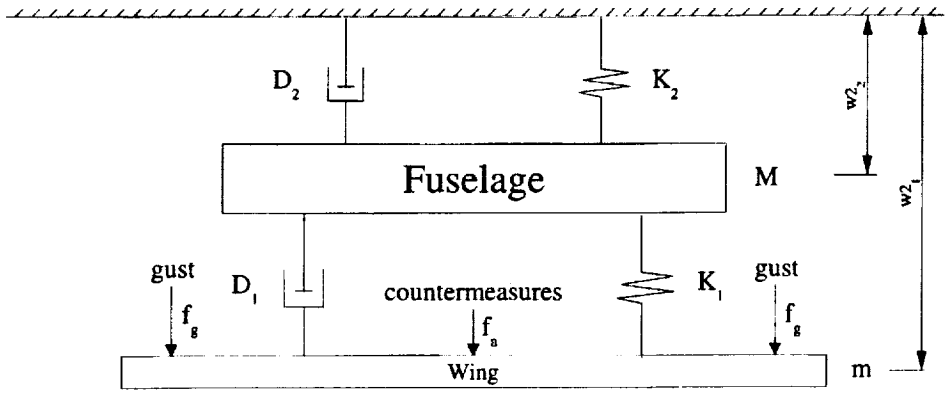

Figure 8.1: Sketch of the wing-fuselage system

- The fuselage itself has a mode at about 1Hz and a damping of 2.5%.

This leads to the following set of constants:

Mass	$m = 1$	$M = 100$
Spring	$K_1 = 700$	$K_2 = 4000$
Damping	$D_1 = 20$	$D_2 = 0.3$

The natural frequency of the fuselage is clearly visible at the bode plot (figure 8.2). For the step response of the coupled system (figure 8.3) the system input is an external force (the gust) to the wing.

The actuator model contains all dynamic parts including the servo-motor and the mechanism moving the ailerons or flaps. Therefore the time constant is higher than for a typical servo itself. The step response is shown in figure 8.4.

I will refer to the Dryden model of gust, as described above. Figure 8.5 shows the Bode plot for the case of $\omega_{gust} = 0.1$.

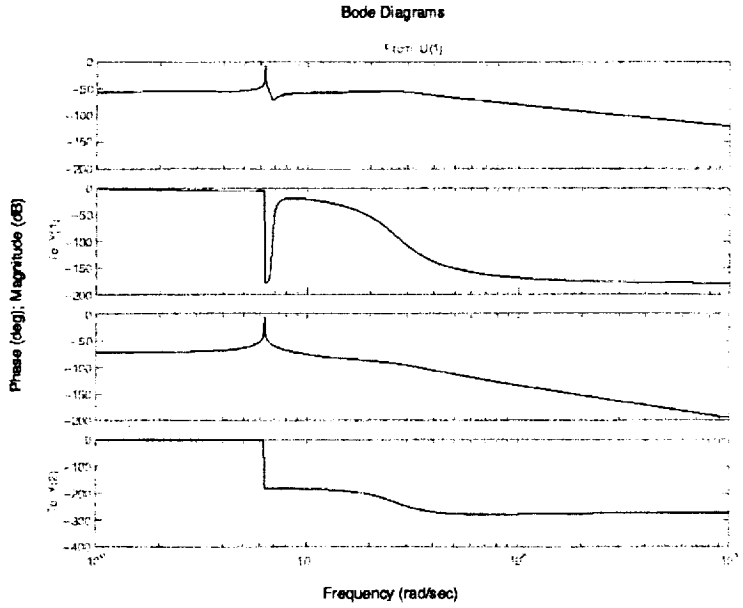

Figure 8.2: Bode plot of the wing-fuselage system

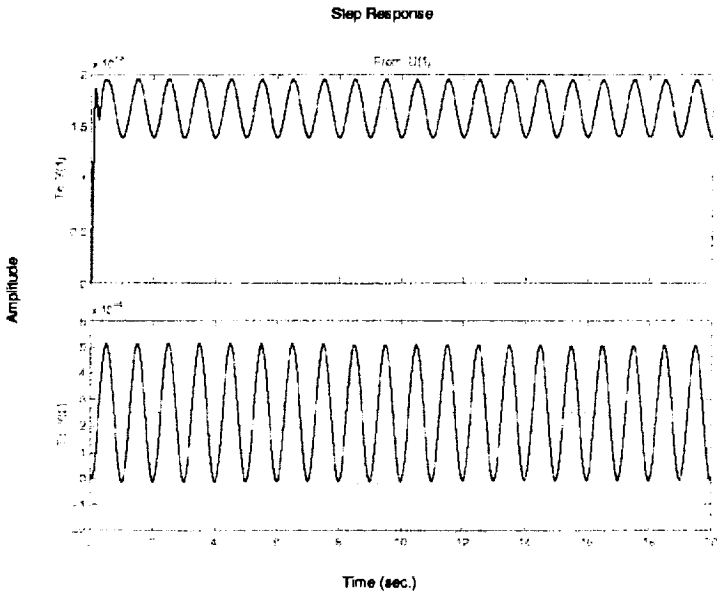

Figure 8.3: Step response for the wing-fuselage system

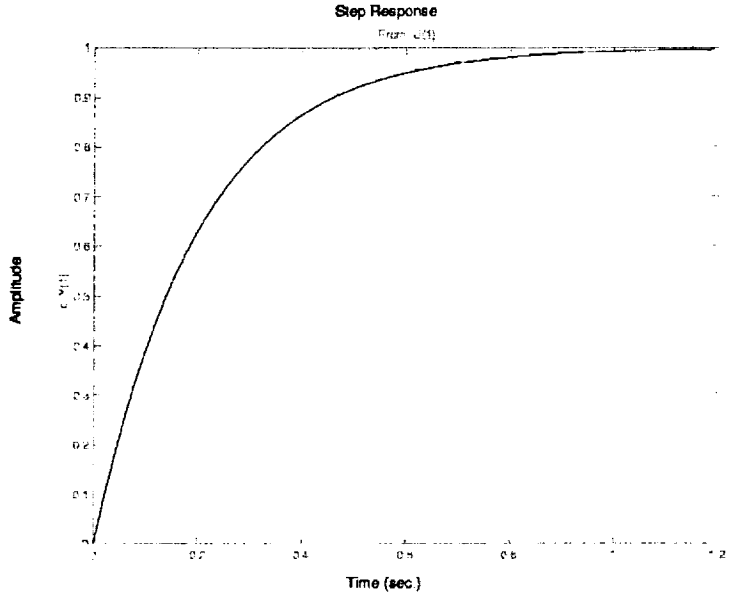

Figure 8.4: Step response of the first order actuator

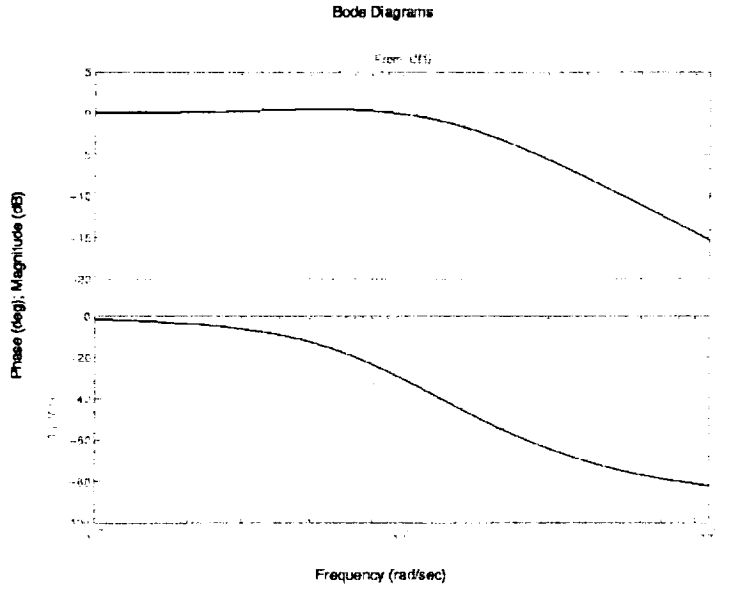

Figure 8.5: Bode plot of the filter for the Dryden model of gust

In order to know the system behavior without the controller, I would like to give the same graphs that will be discussed in the following two experiments also for the system without a controller. Of course, this is not very realistic, as there will always be a controller – even though its purpose is just to reduce the displacements – so that the damping would be higher. However, I think it contributes to the understanding of the system. In anticipation of the main experiments, I would like to give the norms for acceleration and jerkiness for this system without a controller:

$$\begin{aligned} \|\hat{\mathbf{F}}_{\mathbf{a}}\|_\infty &\approx 179 \\ \|\hat{\mathbf{F}}_{\mathbf{j}}\|_\infty &\approx 1129 \end{aligned}$$

The eigenfrequencies of the system are around 1Hz, 4Hz respectively, and thus within the band width of the Dryden model of gust (see 8.5). Although $\|\mathbf{g}\|$ is approximately 8.5 for this simulation, the acceleration and the jerkiness are much smaller than the limit given by means of the corresponding H-infinity norm. On the one hand, the applied "turbulence" was certainly not the worst case, which would be a sinus wave matching a system mode. On the other, the simulation, especially the period of disturbance, was perhaps not long enough: The displacements (see figure 8.8, left graph) seem to be on the increase even at the end of the disturbance, what suggests that a kind of "steady state" on a higher level is not yet reached. Thus, for quantitative comparisons with the following simulations I rather refer to the H-infinity norms. For the evaluation among different sensor positions the test period will prove suitable.

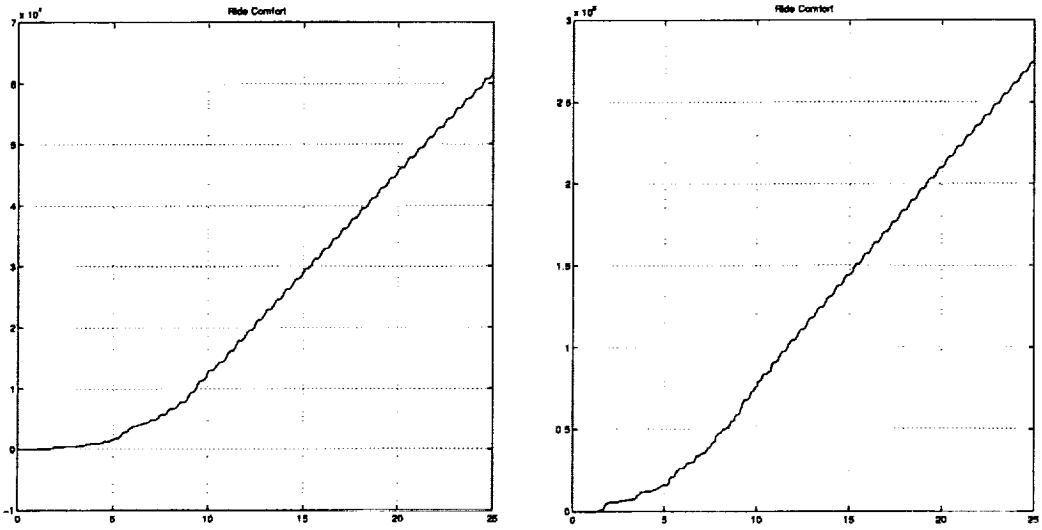

Figure 8.6: "Ride comfort" as the integral of squared accelerations (left) and jerkiness (right)

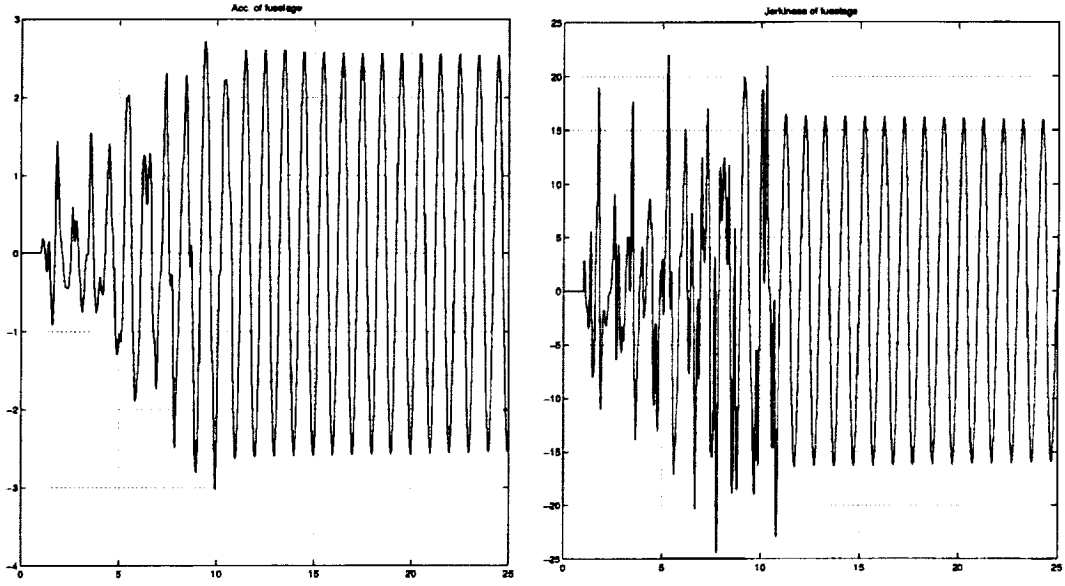

Figure 8.7: Accelerations and jerkiness of the fuselage

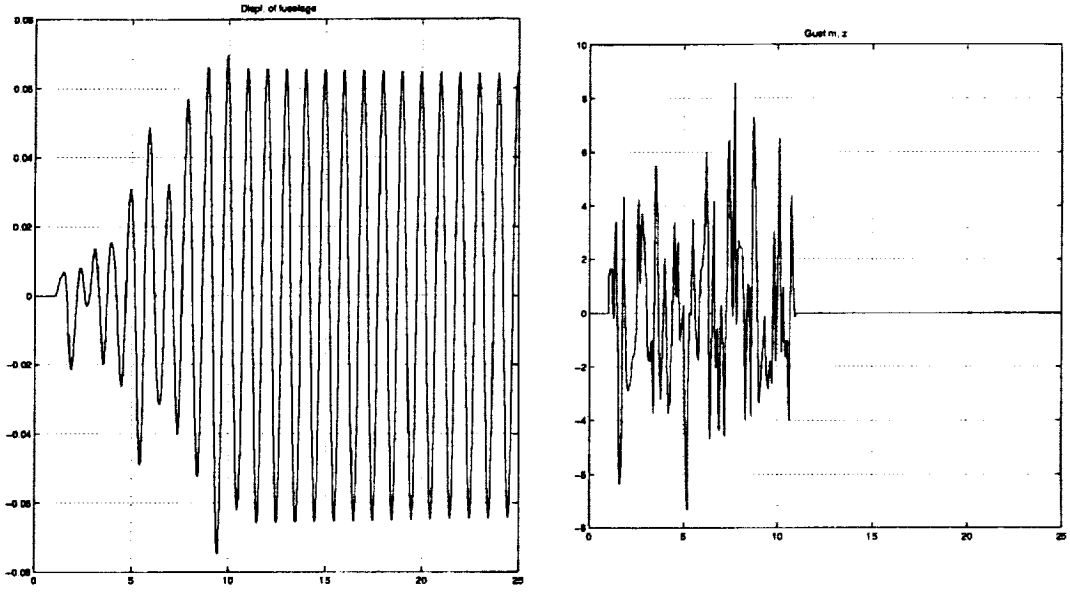

Figure 8.8: Displacement of the fuselage and the gust applied to the wing

8.2 Minimization of accelerations

First of all I would like to refer to the pole-zero maps (figures 8.9 and 8.10) to check for stability. For both values of n the eigenvalues are within the unit circle, the controlled systems are stable. I don't have to worry about the feed forward part of the controller, since for a stable linear system, bounded "disturbances" – the gust and the feed forward component which is only a weighted sum of a finite number of bounded gust data – can never lead to unbounded states and jeopardize (bibo) stability.

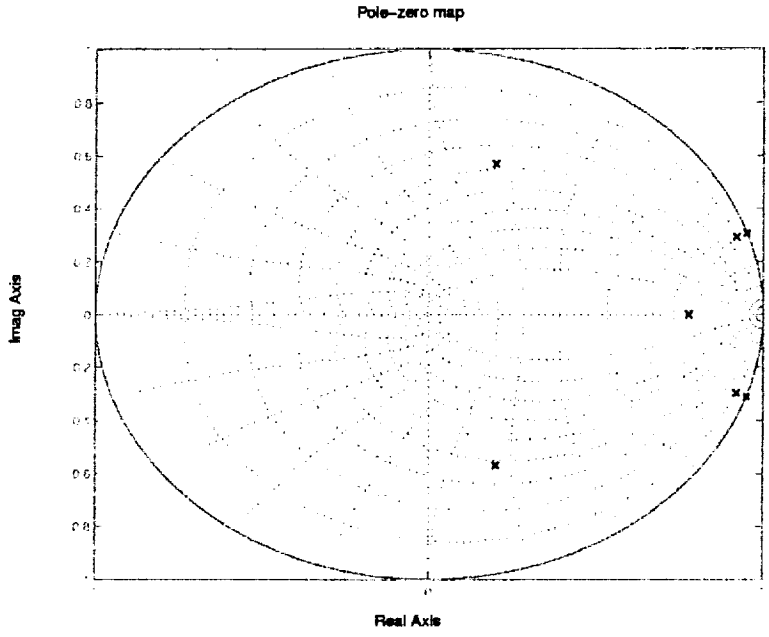

Figure 8.9: Pole-zero map: system without (green) and with the controller for n=1 (blue)

Now, I would like to discuss the H-infinity norms as calculated by *Hinf_calc.m*:

Index	n=1	n=30
$\|\hat{\mathbf{F}}_{\mathbf{ac}}\|_\infty$	0.5070	0.1737
$\|\hat{\mathbf{F}}_{\mathbf{an}}\|_\infty$	179.6118	179.2839
$\|\hat{\mathbf{F}}_{\mathbf{u}}\|_\infty$	32.2925	32.3339
$\|\hat{\mathbf{F}}_{\mathbf{x}}\|_\infty$	10.2074	10.0359
Improvement a	353.2302	1031.2

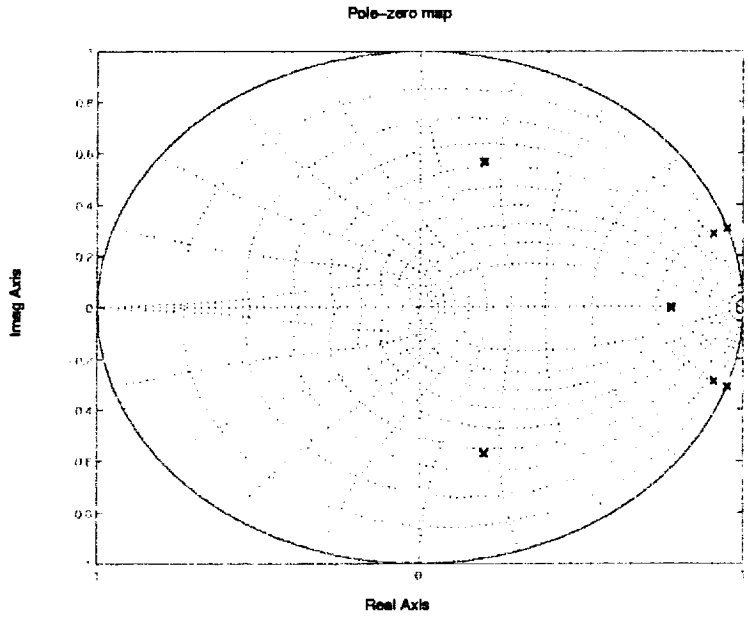

Figure 8.10: Pole-zero map: system without (green) and with the controller for n=30 (blue)

The values of $\|\hat{\mathbf{F}}_{\mathbf{an}}\|_\infty$ for $n = 1$ and $n = 30$ should be equal, the difference is due to inaccuracies of the calculation. For both, the system with and without a controller, the modes are around 1Hz, 4Hz respectively, and thus within the band width of gust. I would like to give the following upper bounds for $\|\mathbf{g}\| \approx 8.5$ with respect to the inequalities stated in section 6.3.3:

bound for	n=1	n=30
$\|\mathbf{a_c}\|$	4.31	1.48
$\|\mathbf{u}\|$	274.49	274.84

The controller output remains clearly below these limits within my simulation, but for a more critical gust function some peaks could be higher. The accelerations for $n = 30$ are already close to the theoretical borderline for this particular controller but for the case of $n = 1$ the performance will deteriorate in the worst case making the advantage of additional gust data even more obvious.

The main criterion for my evaluation is the "ride comfort" which I defined (for this approach) as the integral of the squared accelerations. Ev-

ery simulation lasts 25 seconds and I apply the same gust sequence for $1s \leq t \leq 11s$ to the system, so that the results are easily comparable.
It is unquestionable that already the controller for $n = 1$ leads to a significant improvement of all considered criteria. Thus, I will concentrate on the question whether the additional information available in the case of $n = 30$, which equals data almost about the next 1.5 seconds (T=0.05s), leads to a remarkable increase in performance.

I will give the results of this comparison in tabular form. I would like to remind of the fact that a comfortable ride has a low (!) ride comfort index. Increase/Decrease of the results for n=30 compared with the case of n=1:

Ride comfort index	-58%
Controller Output	-1%
Accelerations	-24%
Displacements	-47%
Aggregated displacements	-69%

Note: The percentages of the table for the accelerations, the displacements and the controller output refer to the maximum absolute value of each graph.

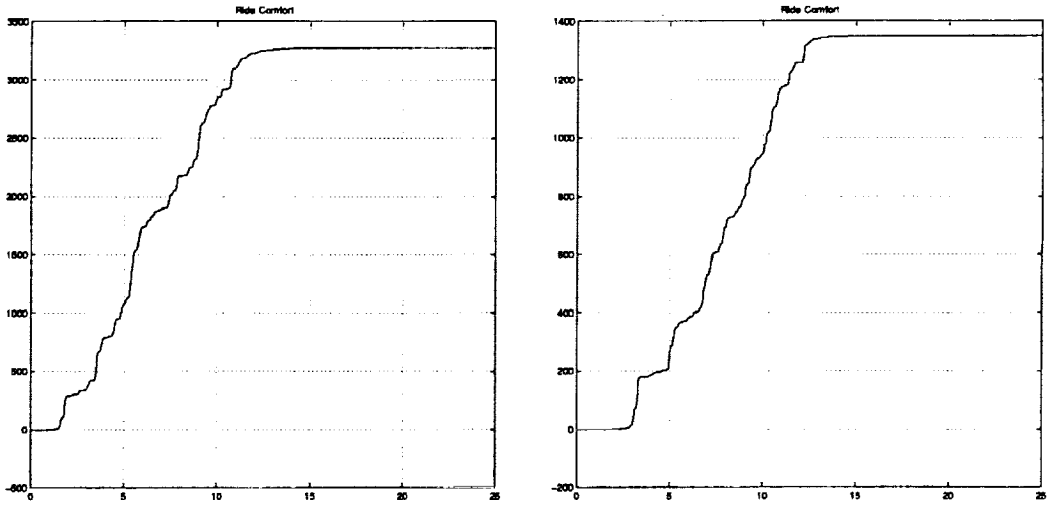

Figure 8.11: "Ride comfort" as the integral of squared accelerations for n=1 and n=30

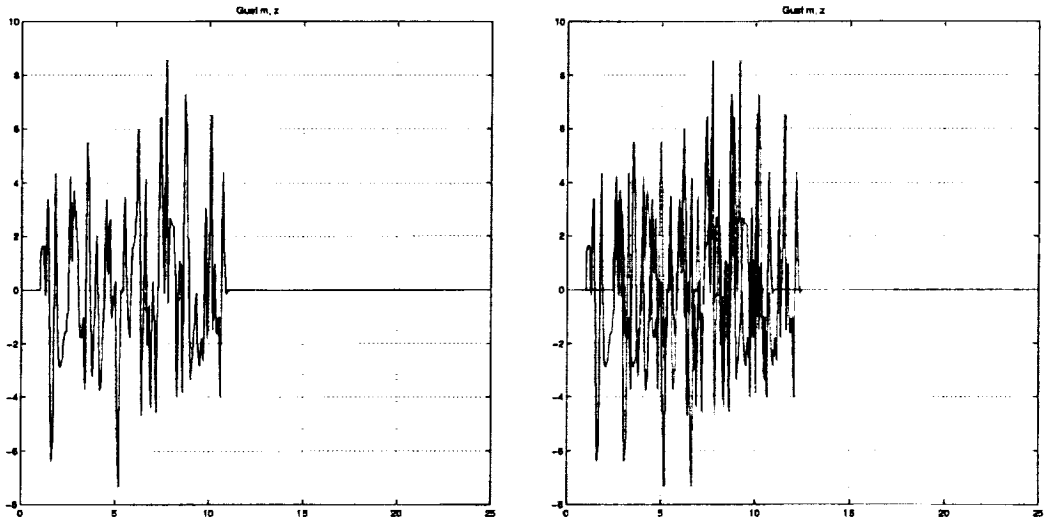

Figure 8.12: Measurement data and disturbance for n=1 and n=30

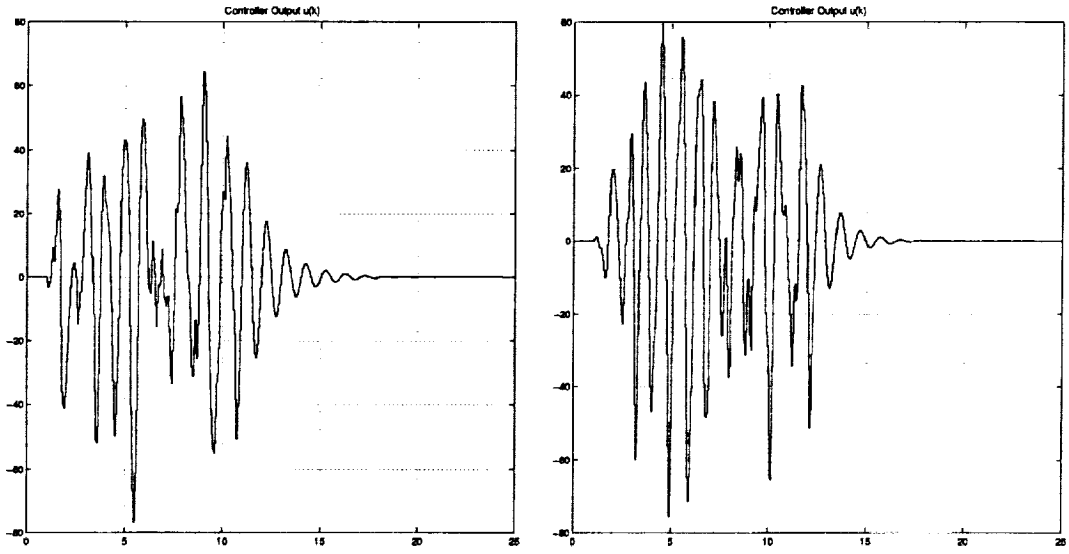

Figure 8.13: Controller output for n=1 and n=30

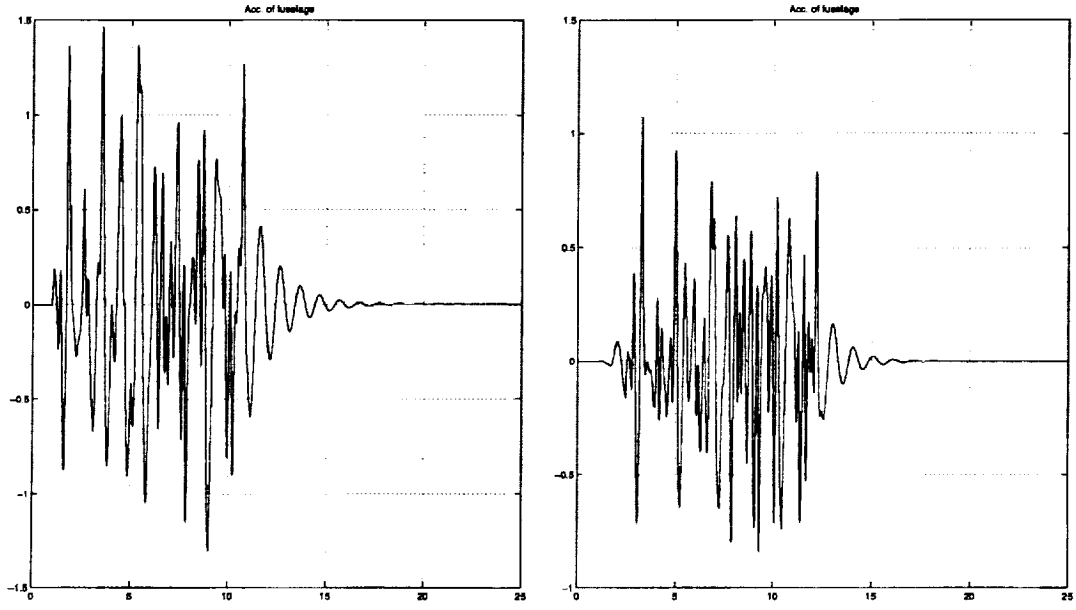

Figure 8.14: Acceleration of the fuselage for n=1 and n=30

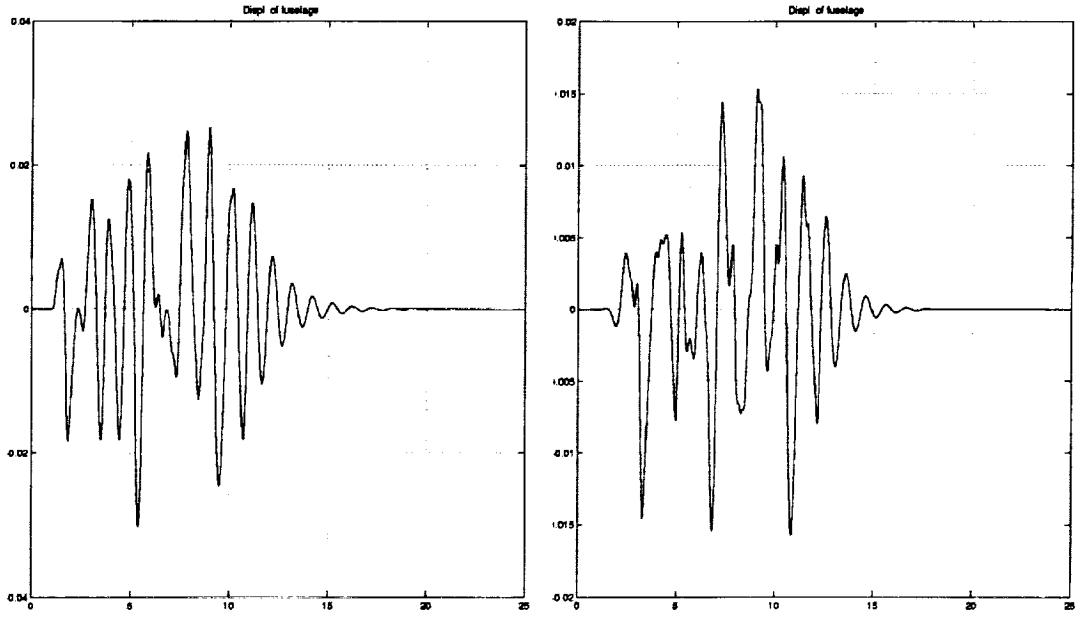

Figure 8.15: Displacement of the fuselage for n=1 and n=30

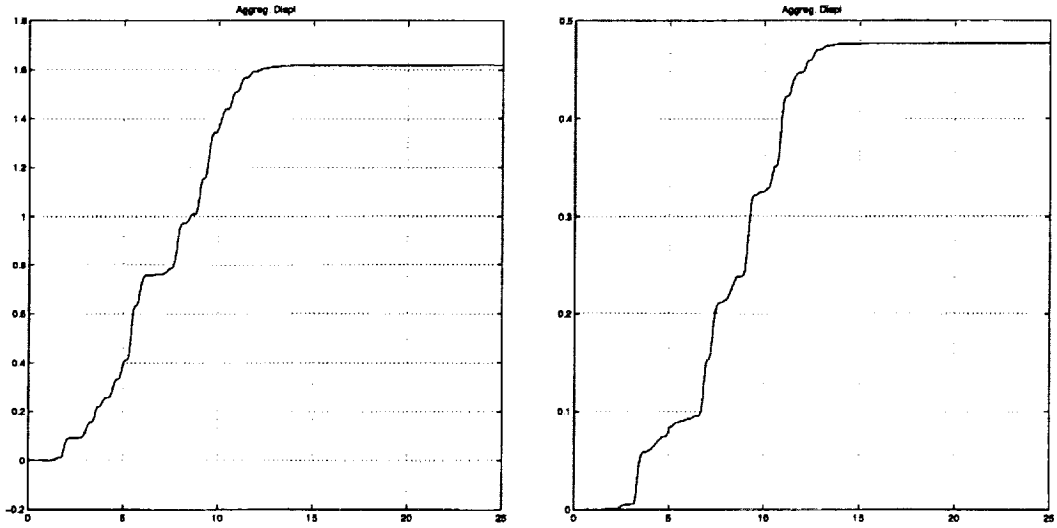

Figure 8.16: "Aggregated displacements" as the integral of squared displacements of the fuselage for n=1 and n=30

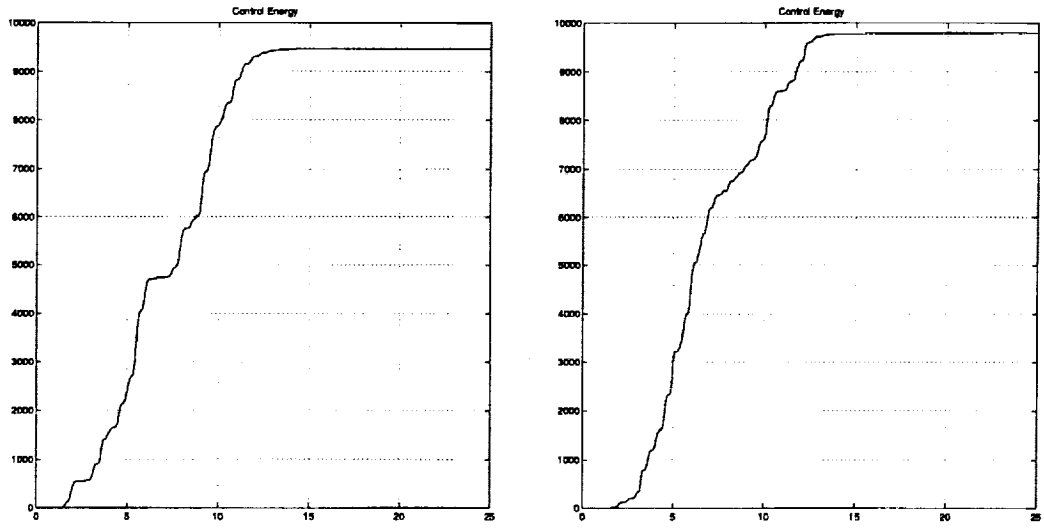

Figure 8.17: Control energy for n=1 and n=30

8.3 Minimization of jerkiness

Also for this approach, I will check for stability in the first place. As can be seen from the pole-zero maps (figures 8.18 and 8.19) for both values of n the eigenvalues are within the unit circle and the controlled systems are stable.

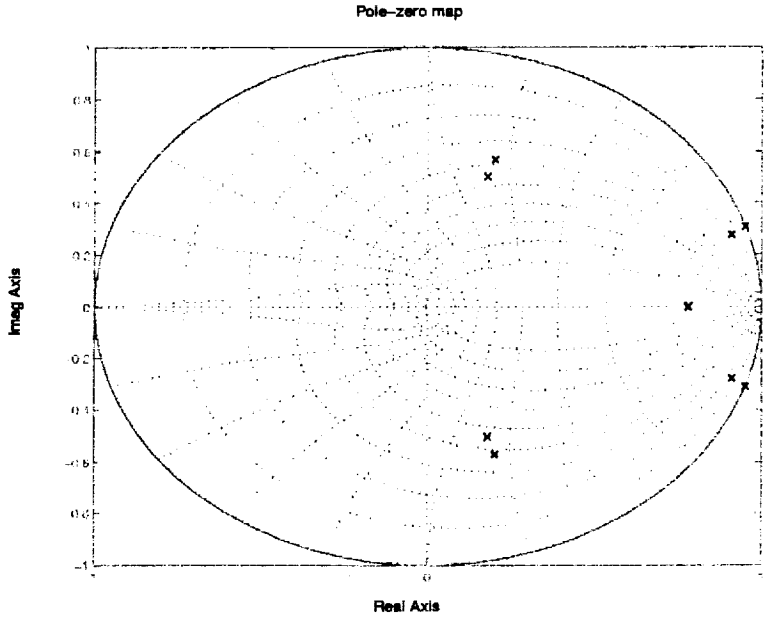

Figure 8.18: Pole-zero map: system without (green) and with the controller for n=1 (blue)

The H-infinity norms for my two approaches show the same trend. The results provided by *Hinf_calc.m* read:

Index	n=1	n=30
$\|\hat{\mathbf{F}}_{\mathbf{jc}}\|_\infty$	4.0383	3.2450
$\|\hat{\mathbf{F}}_{\mathbf{jn}}\|_\infty$	1129.9	1128.9
$\|\hat{\mathbf{F}}_{\mathbf{u}}\|_\infty$	21.3451	19.1240
$\|\hat{\mathbf{F}}_{\mathbf{x}}\|_\infty$	14.4614	10.0953
Improvement j	278.7991	346.9065

The values of $\|\hat{\mathbf{F}}_{\mathbf{jn}}\|_\infty$ are slightly different due to inaccuracies of the calculation. The modes are off course still around 1Hz, 4Hz respectively. The upper bounds for $\|\mathbf{j_c}\|$ and $\|\mathbf{u}\|$ are given for $\|\mathbf{g}\| \approx 8.5$:

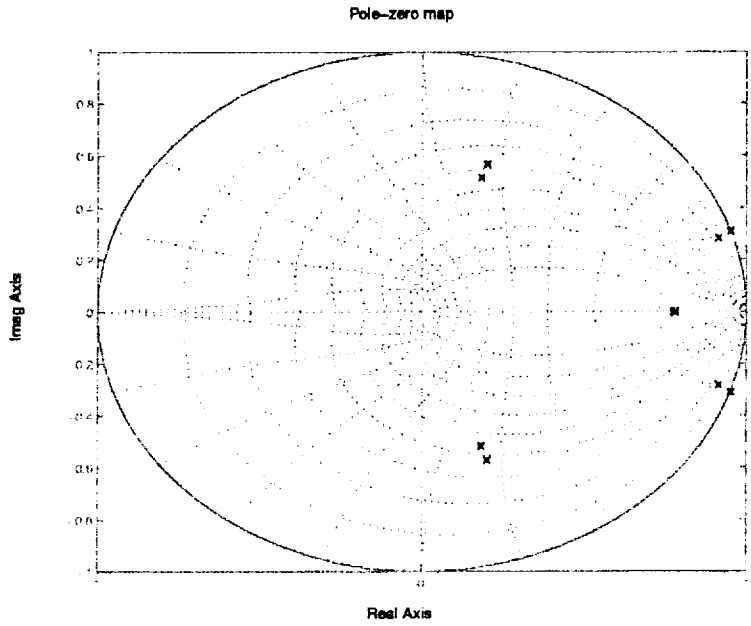

Figure 8.19: Pole-zero map: system without (green) and with the controller for n=30 (blue)

bound for	n=1	n=30
$\|\mathbf{j_c}\|$	34.33	27.58
$\|\mathbf{u}\|$	181.43	162.55

Again, the controller outputs remain far below these limits, but the maximum of jerkiness within my simulation reaches about 50% for both values of n.

Comparing the graphs for $n = 1$ with the results of the system on its own, there is no doubt that the controller is able to reduce jerkiness. Again, the question is whether the extension towards $n = 30$ allows a significant increase in performance. The results of the comparison between n=30 and n=1 read as follows:

Ride comfort index	-51%
Controller Output	-2%
Jerkiness	-32%
Accelerations	-40%
Displacements	-48%
Aggregated displacements	-73%

Note: The percentages for jerkiness, accelerations, displacements and controller output within the given table refer to the maximum absolute value of each graph.

I would like to recall the "Fatigue Life Correlation" diagram that I discussed at my introduction. Although the improvement due to the additional gust data is not as evident as in the case of the first approach or within my preliminary simulations, for the jerkiness, a reduction of 32% can still lead to a substantial increase in life time. For the example of vibration fatigue of SMT solder joints it is more than a factor 100.

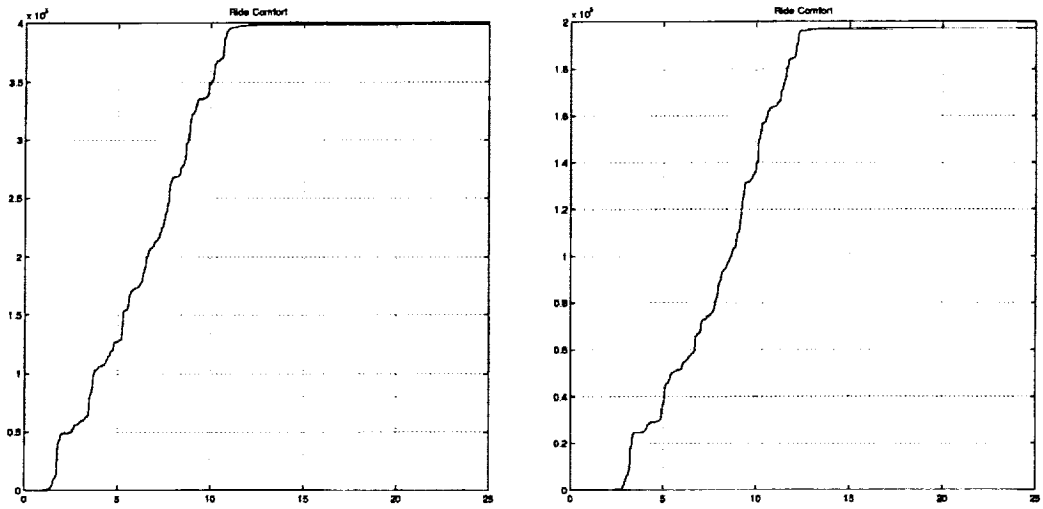

Figure 8.20: "Ride comfort" as the integral of squared jerkiness for n=1 and n=30

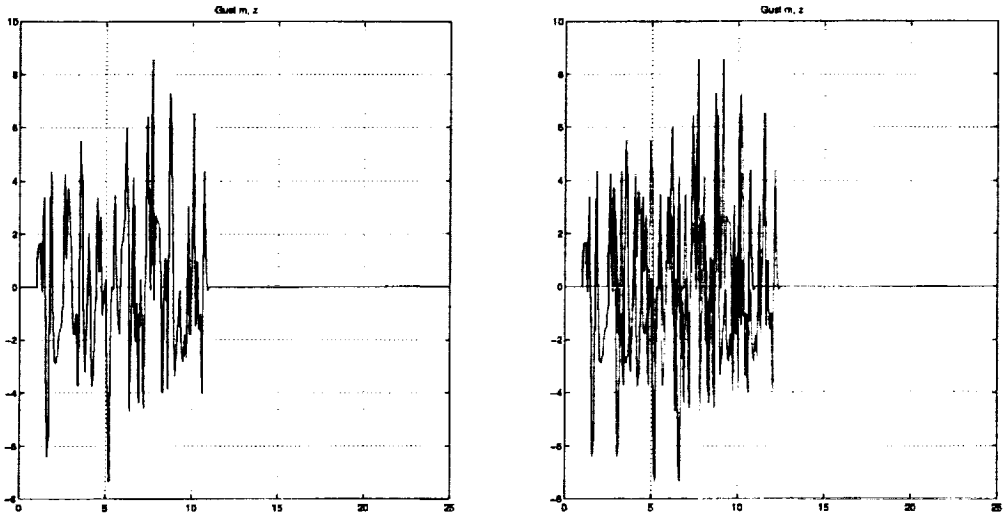

Figure 8.21: Measurement data and disturbance for n=1 and n=30

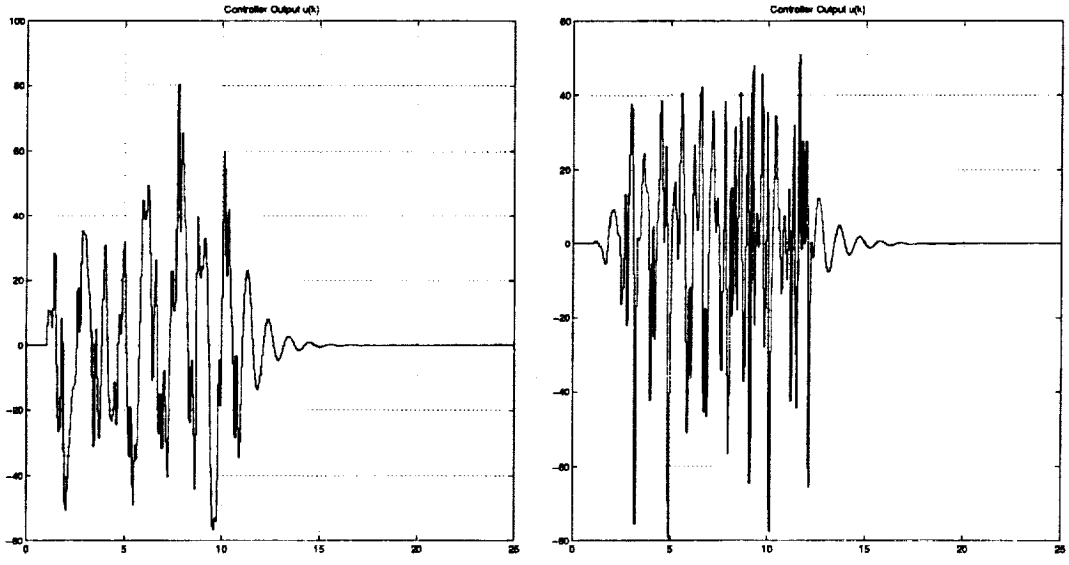

Figure 8.22: Controller output for n=1 and n=30

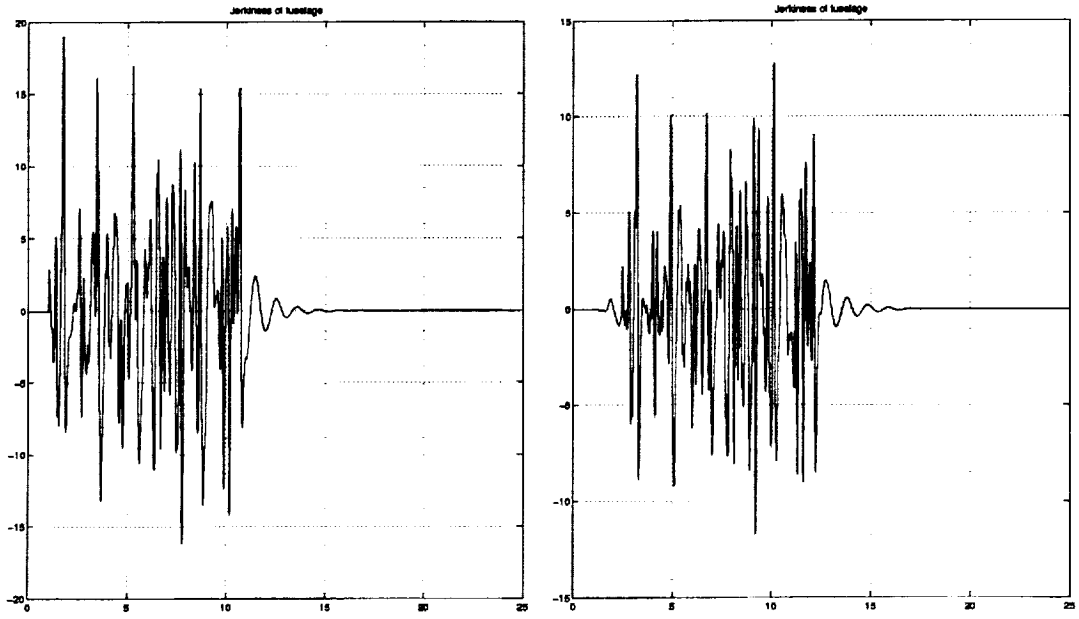

Figure 8.23: Jerkiness of the fuselage for n=1 and n=30

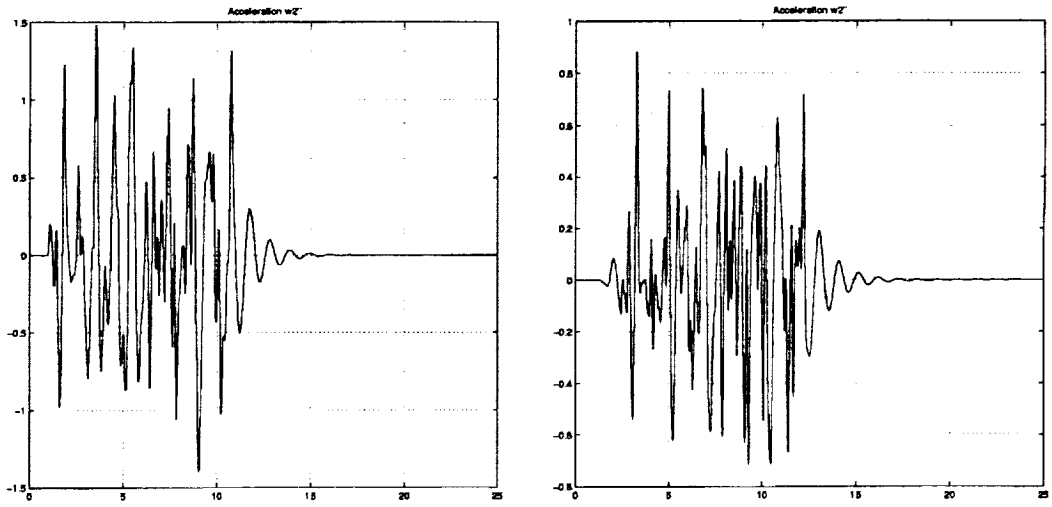

Figure 8.24: Acceleration of the fuselage for n=1 and n=30

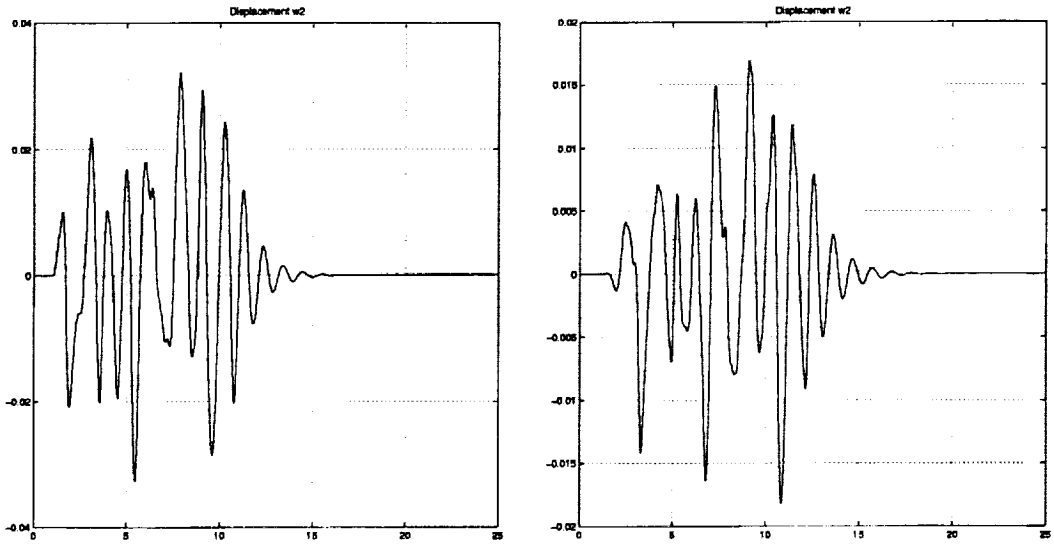

Figure 8.25: Displacement of the fuselage for n=1 and n=30

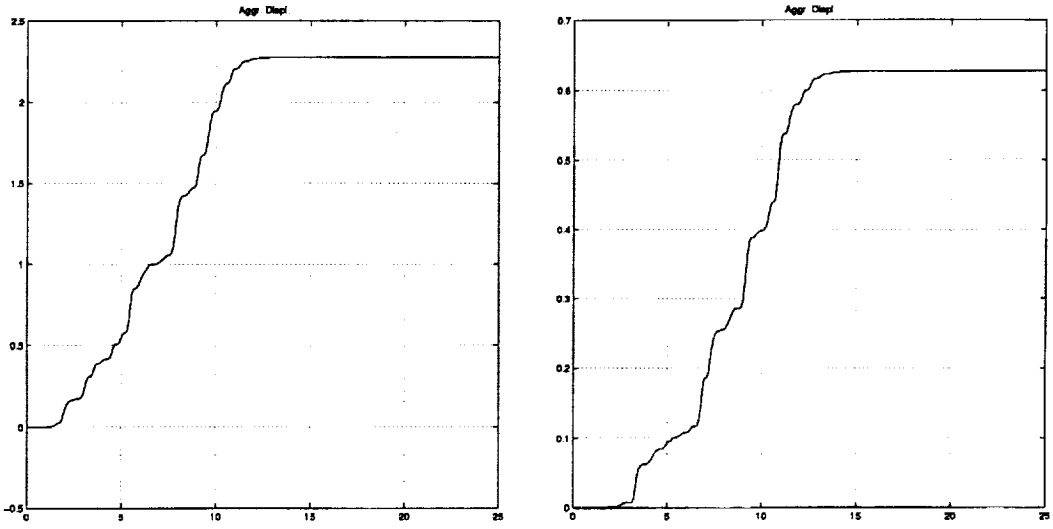

Figure 8.26: "Aggregated displacements" as the integral of squared displacements of the fuselage for n=1 and n=30

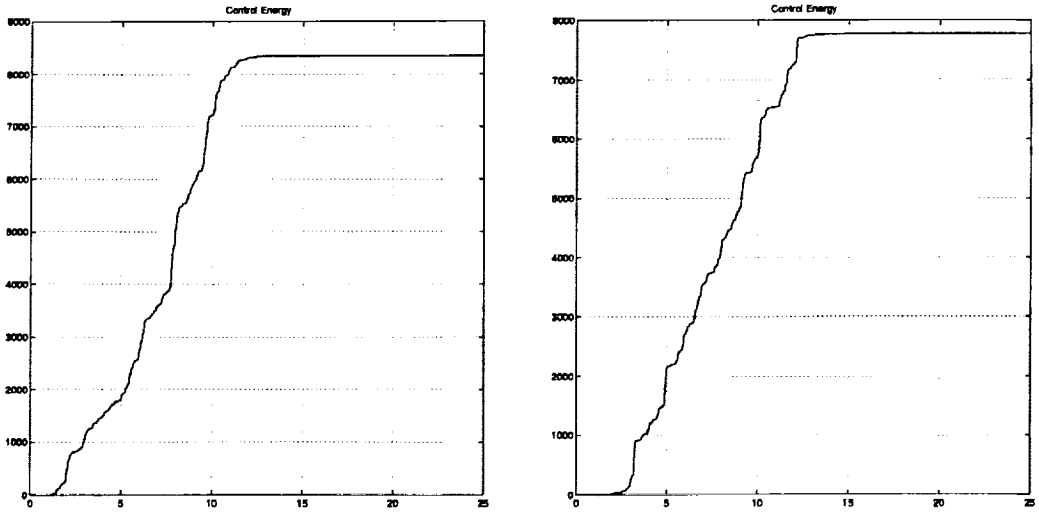

Figure 8.27: Control energy for n=1 and n=30

8.4 Discussion of appropriate sensor positions

I already gave an affirmative answer to the question whether moving the measurement device ahead of the wings has a remarkable beneficial effect. Indeed, that should be the case for almost every system and I showed the improvement for the particular case of a wing-fuselage system. However, the question remains how far ahead it is appropriate to place the sensors. There is no general answer to this question, and the possible improvement is highly dependent on the particular system as well as the actuator that is used. In principle, additional information should be of more benefit if the delays between the encounter of gust and the undesirable effects on the one hand and the delays between the start of countermeasures and their effect on the other are large.

One possibility to decide on the sensor position is to look at the feed forward part of the controller in the form of a transfer function, as described in chapter 6. This representation allows an insight in the control algorithm, as the weights of all available gust data are shown. Starting from the calculation for the sensor far ahead ($n = 30$ in this case), I can examine how many of the highest powers of z I could neglect since their coefficients are too small to have a major effect on the controller performance. The highest remaining power of z defines the upper bound of "reasonable" improvements.

To illustrate this strategy I would like to give the feed forward transfer function for the system of my preliminary simulation. The approach of minimizing the jerkiness using a PT2 actuator (section 7.3.2) provides the best example for that purpose:

$$\begin{aligned}\mathbf{U_{ff_z}}(z) = \mathbf{F3}\,\mathbf{GF_z}(z)\,\mathbf{M_z}(z) = \\ \Big[0.0016z^{30} - 0.0012z^{29} - 0.0041z^{28} - 0.0068z^{27} - \ldots - 0.0056z^{13} - \\ - 0.0007z^{12} + 0.0050z^{11} + 0.0112z^{10} + 0.0167z^{9} + 0.0167z^{8} - \\ - 0.0061z^{7} - 0.1148z^{6} - 0.5375z^{5} - 2.101z^{4} - 7.795z^{3} - \\ - 28.45z^{2} + 29.84z\Big]\; z^{-30}\mathbf{M_z}(z)\end{aligned}$$

It seems that the gust data of $\mathbf{g_k} \ldots \mathbf{g_{k+5}}$ have the most influence on the controller output, so one could try whether the performance for $n = 6$ is already sufficient.

For the wing-fuselage system, however, the coefficients of even the highest powers of z are significantly different from zero, what shows that the gust data in the distant future still help to improve ride comfort in the sense

of the particular approach.

The feed forward transfer function for the experiment of minimizing accelerations reads:

$$\mathbf{U_{ff_z}}(z) = \Big[0.4047z^{29} + 0.2655z^{28} + \ldots + 0.7933z^{15} + 1.176z^{14} + 1.476z^{13} + \\ + 1.652z^{12} + 1.676z^{11} + 1.534z^{10} + 1.215z^9 + 0.7121z^8 + 0.0505z^7 - \\ - 0.6693z^6 - 1.308z^5 - 1.854z^4 - 2.508z^3 - 3.341z^2 - \\ - 3.696z - 2.491\Big]\, z^{-29}\mathbf{M_z}(z)$$

The corresponding function for the approach of minimizing jerkiness:

$$\mathbf{U_{ff_z}}(z) = \Big[0.2141z^{30} + 0.1484z^{29} + 0.0572z^{28} - 0.0506z^{27} \pm \ldots - 0.2906z^6 + \\ + 0.6206z^5 + 0.5423z^4 - 3.459z^3 - 8.832z^2 - 0.8890z + 0.4897\Big] \cdot \\ \cdot z^{-30}\mathbf{M_z}(z)$$

Chapter 9

Final Remarks

9.1 Conclusions

The simulations proved that both controller designs are able to accomplish their individual design goal. It turned out that the main constraint for the application of gust measurement ahead of the wings is the availability of suitable measurement devices, their accuracy and the additional expenses for this technology rather than a limit of improvements close to the wings.

Future work in the field of direct gust measurement should deal with relaxed constraints for the accuracy of data to find an optimum between reliability of data and the prediction interval. The major task, however, is the further development of sensor technology. Aside from improvements regarding range and accuracy, the decoupling from the vibrations of the fuselage or the wings – wherever the device is mounted – is essential. Otherwise the measured data are also a function of system states and the feed forward property cannot be assumed anymore.

Even though this investigation equates the disturbance to the system with atmospheric turbulence, both control algorithms are applicable to any kind of ride comfort problem provided that the given set of constraints is satisfied.

9.2 Acknowledgments

Appreciation is expressed to Prof. A.V. Balakrishnan for giving me the opportunity of doing research within the Flight Systems Research Center at the University of California, Los Angeles.
I would like to thank Dr. Oscar S. Alvarez-Salazar for his helpful suggestions and the valuable discussions.
Also, I would like to express thanks to my team members, Luca Benassi, Jensen Lin, Ananth Subramanian and Gary Wang for the always pleasant working atmosphere and their support.

Bibliography

[1] S. Liguore and D. Followell: *Vibration Fatigue of Surface Mount Technology (SMT) Solder Joints*, Proceedings Annual Reliabilty and Maintainability Symposium, Washington, DC, 1995

[2] U.O. Lappe: *A Preliminary Evaluation of the F-100 Rough Rider Turbulence Measurement System*, Turbulence Consultants, Inc. / National Severe Storms Lab., Norman, Okla. Oct 1967.

[3] O.S. Alvarez-Salazar, G.M. Wang: *A Novel Gust Monitoring Device*, Flight Systems Research Center, UCLA, Los Angeles, CA, 1999

[4] D. Soreide, R.K. Bogue, L.J. Ehernberger, H. Bagley: *Coherent Lidar Turbulence Measurement for Gust Load Alleviation*, Optical instruments for weather forecasting, Proceedings of the Conference, Denver, CO, 1996, Bellingham, WA, Society of Photo-Optical Instrumentation Engineers (SPIE Proceedings. Vol. 2832), 1996

[5] A. Tewari: *Rate weighted Optimal Control of flexible Structures*, Structures, Structural Dynamics, and Materials Conference and Exhibit, 40th, St. Louis, Collection of Technical Papers, Vol. 4, Reston, VA, AIAA, 1999

[6] K. Ogata: *Discrete-Time Control Systems*, Prentice-Hall, Inc., New Jersey, 1987

[7] P. Dorato, C. Abdallah, V. Cerone: *Linear-Quadratic Control, an Introduction*, Prentice-Hall, Inc., New Jersey, 1995

Appendix A

Matlab M-Files

A.1 *const_def.m*

This is just an example. The m-file is adapted to each experiment.

```
% B1u=f1  -> [(M1w1''+)D1w1'+K1w1=f1]
%         -> w1 eigther first or second order diff. eqn. e.g. actuator
% B2w1=f2 -> [ M2w2''+ D2w2'+K2w2=f2]
%         -> w2 second order diff. eqn.

% x=[w2 w2' w1 w1']^T

% M,D,K all square matrices
% "actuator"
% M1=[];              % if applicable
D1=[0.2];
K1=[1];
B1=[0.5];             % f1=B1*u
% "system"  [wing fuselage]^T
M2=[1 0;
   0 100];
D2=[20 -20;
   -20 20.3];
K2=[700 -700;
   -700 4700];
B2=[1; 0];            % f2=B2*w1

G=[0;0; 10;0; 0];     % gust effect on x=[w2;w2';w1;w1']^T,
                      % x=[w2;w2';w1]^T repectively

S=[1000000000 0 0 0 0;
   0 100000 0 0 0;
   0 0 0 0 0;
   0 0 0 0 0;         % weight matrix for final value of x:
   0 0 0 0 0];        % x_(k+N)  must be symmetric
Q1=[0 0 0 0 0;
   0 0 0 0 0;
   0 0 1 0 0;         % weight matrix for x' must be symmetric
```

```
   0 0 0 10000000 0;  % and some terms pos.def.
   0 0 0 0 0];
Q2=1000;              % weight matrix for u  must me symmetric
                      % and pos. def.

RC=[0 0 0 0 0;
   0 0 0 0 0;
   0 0 0 0 0;
   0 0 0 1000 0;
   0 0 0 0 0];        % only lower right term unequal to zero
 AD=[0 0 0 0 0;
    0 1000 0 0 0;
    0 0 0 0 0;
    0 0 0 0 0;
    0 0 0 0 0];      % only upper left term unequal to zero
CE=1;                 % Evaluation of the control energy size as Q2

T=0.05;               % sample period
n=30;                 % number of available gust data
N=40;                 % length of the finite horizon, N>=n

w_gust=0.1;           % parameter for Dryden gust model
```

A.2 *const_calc_1.m*

```
D1size=size(D1);
D2size=size(D2);
B1size=size(B1);

if exist('M1','var')==1
   A=[zeros(D2size) eye(D2size(1)) zeros(D2size(1),D1size(2)) zeros(D2size(1),D1size(2));
      -inv(M2)*K2 -inv(M2)*D2 inv(M2)*B2 zeros(D2size(1),D1size(2));
      zeros(D1size(1),D2size(2)) zeros(D1size(1),D2size(2)) zeros(D1size) eye(D1size(1));
      zeros(D1size(1),D2size(2)) zeros(D1size(1),D2size(2)) -inv(M1)*K1 -inv(M1)*D1];
   B=[zeros(2*D2size(1)+D1size(1),B1size(2)) ; inv(M1)*B1];
   % case of w1: 2nd order diff. eqn.
else
   A=[zeros(D2size) eye(D2size(1)) zeros(D2size(1),D1size(2));
      -inv(M2)*K2 -inv(M2)*D2 inv(M2)*B2;
      zeros(D1size(1),D2size(2)) zeros(D1size(1),D2size(2)) -inv(D1)*K1];
   B=[zeros(2*D2size(1),B1size(2)) ; inv(D1)*B1];
   % case of w1: 1st order diff. eqn.
end

% G already defined in const_def
Asize=size(A);
Bsize=size(B);
Gsize=size(G);
Phi=expm(A*T);
Gamma=inv(A)*(Phi-eye(Asize(1)))*B;
Theta=inv(A)*(Phi-eye(Asize(1)))*G;

S11=A'*Q1*A;
S12=A'*Q1*B;
S13=A'*Q1*G;
```

```
S21=B'*Q1*A;
S22=B'*Q1*B+Q2;
S23=B'*Q1*G;
S31=G'*Q1*A;
S32=G'*Q1*B;
S33=G'*Q1*G;

Minv=inv(Gamma'*inv(Phi')*S12-S22);

% now calculating P_i for i=k+N...k+1, here k set to 0
% P_i is of the same size as A
P=zeros(Asize(1), Asize(2), N);
P(:,:,N)=S;
for l=N-1:-1:1
   P(:,:,l)=inv(eye(Asize(1))-(Phi'*P(:,:,l+1)*Gamma+S12)*Minv*Gamma'*inv(Phi'))
            *(Phi'*P(:,:,l+1)*(Phi-Gamma*Minv*(Gamma'*inv(Phi')*S11-S21))+S11-S12
            *Minv*(Gamma'*inv(Phi')*S11-S21));
end

% now calculating F1_i and F2_i for i=k+n-1...k+1,
% sufficient as only p_(k+n-1)...p_(k+1) calculated,
% here k set to 0
% F1_i is of the same size as A, F2_i is of the same size as G
if n>1
   F1=zeros(Asize(1), Asize(2), n-1);
   F2=zeros(Gsize(1), Gsize(2), n-1);
   for l=n-1:-1:1
      F1(:,:,l)=inv(eye(Asize(1))-(Phi'*P(:,:,l+1)*Gamma+S12)*Minv*Gamma'*inv(Phi'));
      F2(:,:,l)=Phi'*P(:,:,l+1)*Theta+S13-(Phi'*P(:,:,l+1)*Gamma+S12)
                *Minv*(Gamma'*inv(Phi')*S13-S23);
   end
end

% matrices required for controller
F3=-inv(S22+Gamma'*P(:,:,1)*Gamma);
F4=S21+Gamma'*P(:,:,1)*Phi;
F5=S23+Gamma'*P(:,:,1)*Theta;

GF_num=zeros(Bsize(2),n);     % #rows like u_k, n columns for z^(n-1)...z^0
GF_den=zeros(1,n);
GF_den(1,1)=1;                % denominator is z^(n-1)
for l=1:n-1                   % coefficient for z^l, for n=1 the loop is skipped !
   FProd=eye(Asize(1));
   for r=1:l
      FProd=FProd*F1(:,:,r)*Phi';
   end
   GF_num(:,n-l)=Gamma'*FProd*inv(Phi')*F2(:,:,l);
   % coefficient for z^l is located at column (n-l)
end
GF_num(:,n)=F5;
```

A.3 *const_calc_2.m*

```
D1size=size(D1);
D2size=size(D2);
B1size=size(B1);

if exist('M1','var')==1
   A=[zeros(D2size) eye(D2size(1)) zeros(D2size(1),D1size(2)) zeros(D2size(1),D1size(2));
      -inv(M2)*K2 -inv(M2)*D2 inv(M2)*B2 zeros(D2size(1),D1size(2));
      zeros(D1size(1),D2size(2)) zeros(D1size(1),D2size(2)) zeros(D1size) eye(D1size(1));
      zeros(D1size(1),D2size(2)) zeros(D1size(1),D2size(2)) -inv(M1)*K1 -inv(M1)*D1];
   B=[zeros(2*D2size(1)+D1size(1),B1size(2)) ; inv(M1)*B1];
   % case of w1: 2nd order diff. eqn.
else
   A=[zeros(D2size) eye(D2size(1)) zeros(D2size(1),D1size(2));
      -inv(M2)*K2 -inv(M2)*D2 inv(M2)*B2;
      zeros(D1size(1),D2size(2)) zeros(D1size(1),D2size(2)) -inv(D1)*K1];
   B=[zeros(2*D2size(1),B1size(2)) ; inv(D1)*B1];
   % case of w1: 1st order diff. eqn.
end

% G already defined in const_def
Asize=size(A);
Bsize=size(B);
Gsize=size(G);
Phi=expm(A*T);
Gamma=inv(A)*(Phi-eye(Asize(1)))*B;
Theta=inv(A)*(Phi-eye(Asize(1)))*G;

R=A(D2size(1)+1:2*D2size(1),:);   % the rows of A refering to w2''
Thetat=[Theta zeros(Gsize)];
Gt=R*[G zeros(Gsize)]+G(D2size(1)+1:2*D2size(1),:)*[eye(Gsize(2)) -eye(Gsize(2))]/T;
Thetatsize=size(Thetat);

S11=A'*R'*Q1*R*A;
S12=A'*R'*Q1*R*B;
S13=A'*R'*Q1*Gt;
S21=B'*R'*Q1*R*A;
S22=B'*R'*Q1*R*B+Q2;
S23=B'*R'*Q1*Gt;
S31=Gt'*Q1*R*A;
S32=Gt'*Q1*R*B;
S33=Gt'*Q1*Gt;

Minv=inv(Gamma'*inv(Phi')*S12-S22);

% now calculating P_i for i=k+N...k+1, here k set to 0
% P_i is of the same size as A
P=zeros(Asize(1), Asize(2), N);
P(:,:,N)=S;
for l=N-1:-1:1
   P(:,:,l)=inv(eye(Asize(1))-(Phi'*P(:,:,l+1)*Gamma+S12)*Minv*Gamma'*inv(Phi'))
            *(Phi'*P(:,:,l+1)*(Phi-Gamma*Minv*(Gamma'*inv(Phi')*S11-S21))+S11-S12
            *Minv*(Gamma'*inv(Phi')*S11-S21));
end
```

```
% now calculating F1_i and F2_i for i=k+n...k+1,
% sufficient as only p_(k+n)...p_(k+1) considered,
% here k set to 0
% F1_i is of the same size as A, F2_i is of the same size as Thetat
F1=zeros(Asize(1), Asize(2), n);
F2=zeros(Thetatsize(1),Thetatsize(2), n);
for l=n:-1:1
   F1(:,:,l)=inv(eye(Asize(1))-(Phi'*P(:,:,l+1)*Gamma+S12)*Minv*Gamma'*inv(Phi'));
   F2(:,:,l)=Phi'*P(:,:,l+1)*Thetat+S13-(Phi'*P(:,:,l+1)*Gamma+S12)
             *Minv*(Gamma'*inv(Phi')*S13-S23);
end

% matrices required for controller
F3=-inv(S22+Gamma'*P(:,:,1)*Gamma);
F4=S21+Gamma'*P(:,:,1)*Phi;
F5=S23+Gamma'*P(:,:,1)*Thetat;

GF_num=zeros(Bsize(2),n+1);   % #rows like u_k, n+1 columns for z^n...z^0
GF_den=zeros(1,n+1);
GF_den(1,1)=1;                % denominator is z^n
FProd=eye(Asize(1));
for l=1:n-1                   % coefficient for z^l
   FProd=FProd*F1(:,:,l)*Phi';
   GF_num(:,n-l)=GF_num(:,n-l)+Gamma'*FProd*inv(Phi')*F2(:,:,l)
                 *[eye(Gsize(2));zeros(Gsize(2),Gsize(2))];
   % coefficient for z^l is located at column (n+1-l),
   % additional shift left because of {}z !
   GF_num(:,n-(l-1))=GF_num(:,n-(l-1))+Gamma'*FProd*inv(Phi')*F2(:,:,l)
                     *[zeros(Gsize(2),Gsize(2));eye(Gsize(2))];
   % coefficient for z^(l-1) is located at column (n+1-(l-1)),
   % additional shift left because of {}z !
end
GF_num(:,1)=GF_num(:,1)+Gamma'*FProd*F1(:,:,n)*F2(:,:,n)
            *[zeros(Gsize(2),Gsize(2));eye(Gsize(2))];
% since z_(k+n)=0
GF_num(:,n)=GF_num(:,n)+F5*[eye(Gsize(2));zeros(Gsize(2),Gsize(2))];
GF_num(:,n+1)=F5*[zeros(Gsize(2),Gsize(2));eye(Gsize(2))];
```

A.4 *Hinf_calc.m*

```
% This m-file calculates the H-infinity norms of various transfer functions with the
% gust measurement data as input "u" and the particular variable as the output "y".
% For a discrete-time state space representation I can make use of the Matlab
% function normhinf. Like for the rest of the simulation, I have to restrict to
% scalar gust data. The dynamic matrix Aall refers to the controlled system, Atall
% to the case of no controller in use

L=RC/norm(RC,'fro');

Bi_mat=zeros(Bsize(2),n+1);
if exist('Thetat','var')==1
   for l=1:n+1
      Bi_mat(:,1)=F3*GF_num(:,n+2-l);  % for approach II
   end
else
```

```
   for l=2:n+1
      Bi_mat(:,l)=F3*GF_num(:,n+2-l);  % for approach I
   end
end
K=F3*F4;
A11=Phi+Gamma*K;              % Phi_c
A12=Gamma*Bi_mat+[zeros(Gsize) Theta zeros(Gsize(1),(n-1)*Gsize(2))];
A21=zeros(n+1,Asize(1));
A22=zeros(n+1,n+1);
for l=1:n
   A22(l,l+1)=1;
end
Aall=[[A11 A12] ; [A21 A22]];
At11=Phi;
At12=[zeros(Gsize) Theta zeros(Gsize(1),(n-1)*Gsize(2))];
At21=A21;
At22=A22;
Atall=[[At11 At12] ; [At21 At22]];
Ball=zeros(Asize(1)+n+1,1);
Ball(Asize(1)+n+1,1)=1;        % last element of column vector is 1

Callx=[eye(Asize(1)) zeros(Asize(1),n+1)];  % for the "system" m->system->x
Callu=[K Bi_mat];                           % for the "system" m->system->u
if exist('Thetat','var')==1                 % approach II ?
   if exist('M1','var')==1                  % PT2 actuator ?
      Calljn=-L*inv(M2)*[K2 D2 zeros(D2size(1),2*D1size(2))]
            *[A zeros(Gsize) G zeros(Gsize(1),(n-1)*Gsize(2))];
   else
      Calljn=-L*inv(M2)*[K2 D2 zeros(D2size(1),D1size(2))]
            *[A zeros(Gsize) G zeros(Gsize(1),(n-1)*Gsize(2))];
   end
   Calljc=L*([R*(A+B*K) R*B*Bi_mat]
         +[zeros(D2size(1),Asize(2)) Gt*[zeros(Gsize(2),Gsize(2));eye(Gsize(2))]
         Gt*[eye(Gsize(2));zeros(Gsize(2),Gsize(2))] zeros(D2size(1),(n-1)*Gsize(2))]);
else                                        % approach I
   Callan=[L*A zeros(Gsize) L*G zeros(Gsize(1),(n-1)*Gsize(2))];
   Callac=L*([(A+B*K) B*Bi_mat]
         +[zeros(Asize) zeros(Gsize) G zeros(Gsize(1),(n-1)*Gsize(2))]);
end

Dallx=zeros(Asize(2),1);
Dallu=zeros(Bsize(2),1);
Dallan=zeros(Asize(2),1);
Dallac=Dallan;
Dalljn=zeros(D2size(2),1);
Dalljc=Dalljn;

sysx=ss(Aall,Ball,Callx,Dallx,T);
sysu=ss(Aall,Ball,Callu,Dallu,T);
if exist('Thetat','var')==1                 % approach II ?
   sysjn=ss(Atall,Ball,Calljn,Dalljn,T);
   sysjc=ss(Aall,Ball,Calljc,Dalljc,T);
else                                        % approach I
   sysan=ss(Atall,Ball,Callan,Dallan,T);
   sysac=ss(Aall,Ball,Callac,Dallac,T);
end
```

```
normx=normhinf(sysx)
normu=normhinf(sysu)
if exist('Thetat','var')==1                % approach II ?
   normjn=normhinf(sysjn)
   normjc=normhinf(sysjc)
   improve_j=(normjn-normjc)/normjc
else                                       % approach I
   norman=normhinf(sysan)
   normac=normhinf(sysac)
   improve_a=(norman-normac)/normac
end
```

A.5 *pzmap_calc.m*

```
% This m-file calculates the Pole-zero map of the system with and without
% the controller

A=Phi;
Ac=Phi+Gamma*F3*F4;
if exist('M1','var')==1
   C=[eye(D2size), zeros(D2size), zeros(D2size(1),2*D1size(2))];
else
   C=[eye(D2size), zeros(D2size), zeros(D2size(1),D1size(2))];
end
B=zeros(Bsize);  % B, D do not matter for this purpose
D=zeros(D2size(1),Bsize(2));

sys=ss(A,B,C,D,T);
sysc=ss(Ac,B,C,D,T);

[P,Z]=pzmap(sys);
[Pc,Zc]=pzmap(sysc);

dummy=tf(1,1);
pzmap(dummy);    % just to get the lables etc.
zgrid;
hold on
plot(P, 'gx');
plot(Pc, 'bx');
SZ=size(Z);
SZc=size(Zc);
if SZ(1)~=0
   plot(Z, 'go');
end
if SZc(1)~=0
   plot(Zc, 'bo');
end
```