Multiplicative Robust and Stochastic MPC with Application to Wind Turbine Control

Martin A. Evans

A thesis submitted for the degree of Doctor of Philosophy

Worcester College

16 August 2014

Abstract

A robust model predictive control algorithm is presented that explicitly handles multiplicative, or parametric, uncertainty in linear discrete models over a finite horizon. The uncertainty in the predicted future states and inputs is bounded by polytopes. The computational cost of running the controller is reduced by calculating matrices offline that provide a means to construct outer approximations to robust constraints to be applied online. The robust algorithm is extended to problems of uncertain models with an allowed probability of violation of constraints. The probabilistic degrees of satisfaction are approximated by one-step ahead sampling, with a greedy solution to the resulting mixed integer problem. An algorithm is given to enlarge a robustly invariant terminal set to exploit the probabilistic constraints.

Exponential basis functions are used to create a Robust MPC algorithm for which the predictions are defined over the infinite horizon. The control degrees of freedom are weights that define the bounds on the state and input uncertainty when multiplied by the basis functions. The controller handles multiplicative and additive uncertainty.

Robust MPC is applied to the problem of wind turbine control. Rotor speed and tower oscillations are controlled by a low sample rate robust predictive controller. The prediction model has multiplicative and additive uncertainty due to the uncertainty in short-term future wind speeds and in model linearisation. Robust MPC is compared to nominal MPC by means of a high-fidelity numerical simulation of a wind turbine under the two controllers in a wide range of simulated wind conditions.

Contents

Acknowledgements

This work was funded by EPSRC and Vestas Technology UK.

It would not have been possible without the tireless enthusiasm of my supervisors Mark Cannon and Basil Kouvaritakis, who always found time for me. Help and guidance on the wind turbine application came from Vestas, with special thanks due to Ian Couchman, Rob Bowyer and Gijs van der Veen. Chris Spruce was the keystone making the collaboration possible and believing in it months before work started.

I'd personally like to thank those who put me up and put up with me these last few years. Meg's infinite support has kept me positive throughout. Row & Tom, Kathryn & Cliff, Douglas, James and Frankie & Pete have all gone out of their way to help.

Finally, thanks are due to all those who kept me distracted and put things in perspective: College friends, Oxford Hub, Worcester College Boat Club and OUSU E&E.

For Mum and Dad

Chapter 1

Introduction

1.1 Model Predictive Control

Automatic control is found all around us in the modern world. A thermostat connected to a heater forms a simple control system wherein the thermostat turns the heater on when the temperature drops below a lower threshold and off when it rises above an upper threshold. More complex than such an on-off controller is a proportional controller, the output of which is a gain value multiplied by the difference between the measurement and a set-point. The behaviour of a system under proportional control will often be preferable to the behaviour of the system under on-off control because the controller does not wait for the measurement to reach a threshold and then slam the control action fully on but rather smoothly applies the control action depending on the deviation from the set-point.

An on-off controller typically does not cause the controlled system to settle at a constant steady state. Instead, the system will alternate between the two thresholds. A proportional controller may result in a constant steady-state but it may not be at the desired state. We can alter the steady-state of a system under proportional control by applying an offset to the control action, i.e. adding a constant value. However, in many applications, the exact offset required may not be known or may change over time. A proportional-plus-integral, or proportional-integral (PI) controller could be used to adapt the control offset over time. The integral action integrates the error, the difference between the measurement and

the set-point. The control action is a weighted sum of that integral and the proportional term. A similar form of controller adds in a differential, or derivative, term. Such PID controllers form the basis of the majority of industrial controllers.

The choice of proportional, integral and derivative gains are the subject of a wide and mature field of research. Further adaptations allow for multiple inputs and outputs, limits or saturations of the inputs, and gains that vary over time. These adaptations are typically very application specific. Additionally they are purely reactive – while the control parameters may be chosen specifically for the system to be controlled, they do not exploit any information the engineer has about the system dynamics at run-time.

Model predictive control (MPC) is a strategy for the control of a system by forming predictions of the behaviour of the system and then optimising those predictions. The predictions are variables that depend on the proposed controller actions. That dependence is defined by a 'model', a mathematical representation of the system to be controlled. The optimisation is a numerical problem constituting the minimisation of a cost function with respect to the proposed controller actions, or 'predicted inputs', over a finite time horizon into the future. The first predicted inputs of the cost minimising sequence are applied to the system, then a new measurement of the system is taken, which requires the optimisation to be solved again, thus providing a mechanism for incorporating feedback.

What sets MPC apart is its direct ability to account for constraints, not only on the inputs to the system but also on the system's states. The numerical solution of a minimisation of a convex cost function subject to convex constraints is a task for which efficient software methods exist. Nonetheless, when solving a constrained optimisation problem repeatedly like this, the computational time taken at each time step may be a limiting design factor. For this reason, the control model used for predictions is often taken to be a simplified description of the system dynamics. A detailed review of many aspects of MPC is given in [Mayne et al., 2000], a summary written at a time when linear MPC, in discrete time with no uncertainty, was quite mature. Predictive controllers wherein the model accounts for no uncertainty are termed 'nominal MPC'.

1.2 Robust & Stochastic MPC

An autonomous controller is one with no online degrees of freedom. A combination of autonomous controller, control model and a set of constraints defines a feasible region in the space of model states, or 'state space'. When the state evolves under a controller with online degrees of freedom, the feasible region is in the combined space of the model states and those degrees of freedom. However, in this non-autonomous case, not only must the solution of the optimisation problem satisfy the constraints, it must be possible to solve the same problem one step later, having applied the calculated input to the system. This requirement necessitates a tighter set of constraints on the variables defining the predicted sequence of future system states and inputs. Part of the task of designing a suitable MPC algorithm is finding such a constraint set. This aspect of the design process is more involved when uncertainty is explicitly accounted for in the model.

Consider someone walking along the edge of a cliff. Without taking uncertainties into account when deciding what steps to take, one strategy might be to take steps that are chosen to approach the very edge. When disturbances, for example wind gusts, act, it may become impossible to maintain stability because there has been left no margin for error. Allowing a large safety margin avoids this risk but can be overly conservative – how much is enough? By modelling the effect that the uncertainties have on the system, appropriate conservativeness can be employed, producing a strategy for walking close to the edge without the risk of falling. Incorporating bounded uncertainty into the constraint set is the role of Robust MPC (RMPC).

Uncertain prediction models do not produce a unique sequence of predicted states as deterministic models do. If a linear control problem has bounded uncertainty then there exists a finite set of possible predicted states. Applied to such a system, given a feasible initial state, RMPC will find a sequence of inputs for which all possible predicted states and inputs will satisfy the constraints. Furthermore RMPC guarantees that it will always be able to solve that problem in the future, for any uncertainty realisation within those given bounds, a property called 'recursive feasibility'.

If the uncertainty is additive, the same constraint set as used in nominal MPC

can be employed, with an adjustment made to some parameters to tighten the constraints enough to guarantee robustness. On the other hand, with multiplicative uncertainty, also known as model or process uncertainty, the required degree of tightening depends on the inputs, which turns the tightening values into optimisation variables. If the uncertainty is bounded by a convex set with finite known vertices, a brute force approach could be to constrain the predicted states and inputs for every possible sequence of extreme uncertainty realisations. Under linear dynamics, the uncertain predictions then lie in the set bounded by the predictions under the worst-case dynamics.

Such a brute-force approach leads to an exponential growth of constraints, thereby rendering the control problem intractable. This thesis instead develops an outer approximation to the worst case predictions, producing a problem to which the solution requires computational effort that scales linearly with the horizon length.

In an application where one or more constraints on a system need only be satisfied with a given probability, RMPC will result in overly conservative operation. Allowing a constraint to be violated with (up to) a given probability is the task of Stochastic MPC. Due to the complexity introduced by multiplicative uncertainty in prediction models, there are two main types of SMPC for such models: one-step ahead and scenario sampling.

If the uncertainty is infinitely supported, e.g. Gaussian noise, it is impossible to find robust bounds and therefore recursive feasibility can no longer be guaranteed. Scenario sampled SMPC foregoes recursive feasibility and instead introduces a probability of feasibility, for which constraints on the predictions are calculated to give a sufficiently high lower bound on the probability of feasibility. In practice, a probability of feasibility very close to unity can be achieved by careful controller design. The scenario approach considers probabilistic statements regarding the predicted state as it evolves over the entire prediction horizon, for which the probability distribution requires many convolutions and therefore cannot be used explicitly in the optimisation problem. Instead, in this approach, samples are taken of the distributions – each samples corresponds to one possible scenario for the evolution of the state over the prediction horizon.

If the uncertainty is finitely bounded, recursive feasibility may be retained

using techniques from RMPC, while adaptations allow individual constraints to be violated. This kind of SMPC, termed 'one-step ahead', forms probabilistic statements regarding the evolution of the state over each step in the prediction horizon separately, while bounds on the worst-case evolution are maintained.

Physical systems do not suffer infinitely supported uncertainty. However, a finitely supported probability distribution may have very long tails, with very little probability of an uncertainty realisation near its bounds. A hybrid of the two types of SMPC is possible, whereby feasibility is recursively guaranteed for a curtailed uncertainty distribution. This thesis focusses on one-step sampled SMPC, which is presented as an extension of RMPC, although most of the methods presented apply to either, or can be readily adapted.

1.3 Control of Wind Turbines

Wind turbines are expected to operate autonomously for twenty years with minimal manual intervention. This thesis considers upwind horizontal axis wind turbines, the type that generates all but an insignificant amount of all global wind power. Upwind turbines turn to face the wind automatically, using a simple controller that measures the wind direction relative to the rotor heading. Rotational speed and power output are typically controlled by PID.

This thesis applies novel RMPC techniques to the problem of wind turbine control. It models various uncertainties and uses RMPC to account for the uncertainties that propagate into the predictions. We test the controller in closed-loop with a realistic simulation of a wind turbine. The differences between the simple prediction model and the complex simulation pose a tough control challenge, which is met by the RMPC implementation in a novel way. This section now introduces the physics of wind turbines and the place the controller has in their operation.

The lift force on a wind turbine blade is the same force that acts on aircraft wings. Unlike aeroplanes however, wind turbines can change their angle of attack easily, and do so to regulate their rotational speed. This ability is called pitching and this thesis only considers pitch angles that do not cause the blades to stall.

Extracting electrical power decelerates the rotor due to the generator torque. The difference between the aerodynamic torque and the generator torque determ-

ines whether the rotor will accelerate or decelerate. In low wind speeds, the aim is to convert as much aerodynamic power as possible into electrical power. In high wind speeds, the aim is to prevent the rotor from gaining too much speed, with the electrical power held at its nominal, or nameplate value. Since the wind speed is always changing, this regulation is a constant process, which happens automatically, and for which the control system is responsible.

In addition to control of the rotor speed, current research is aimed at the control of tower oscillations. Tower oscillations can become more severe as wind turbines become larger. The economic trend in wind power is for ever increasing size of turbines so tower oscillations are expected to be an increasingly important issue in the industry. Figure 1.1 shows an upwind turbine with the degrees of freedom that are included in the control model of Chapter 6.

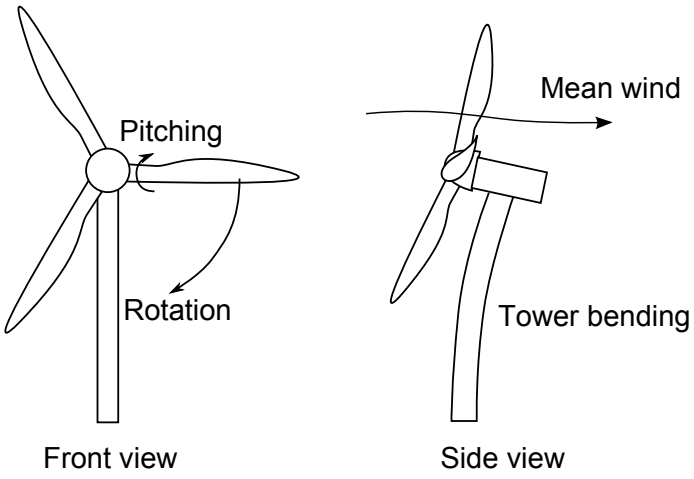

Figure 1.1: A three bladed upwind turbine illustrating some terminology. Note that the tower bending is exaggerated.

It is normally desirable to keep any controller as simple as possible. However, simple controllers can lack certain features that maybe useful or even essential. Classical control theory cannot handle constraints optimally, for example. A controller that explicitly considers all requirements will often perform better than one that gets around these issues in an indirect way. A good example of a design requirement that is only indirectly addressed by classical control is integrator windup, in which an integrated error term accumulates while an input is saturated.

Ignoring that problem can lead to instability, whereas applying feedback on the saturated signal, as in anti-windup control, leads to a suboptimal solution.

Equipped with novel theoretical extensions to model predictive control that allow it to handle the kinds of uncertainty introduced by wind speed variations, nonlinearities and low sample rates, this thesis presents synthesis of a controller that simultaneously regulates rotational speed and tower movement, while limiting pitch angle and generator power. The considered model is uncertain, multi-input-multi-output, with constraints, a class of model to which RMPC is suited. Robustness is used to guarantee recursive feasibility, which guarantees that the control scheme will not unexpectedly fail to produce a solution.

A large active research area is looking at how best to make control models that capture the salient features of wind turbine behaviour while leading to a tractable optimisation problem. Chapter 6 of this thesis models a large wind turbine in a three dimensional state, comprising rotor speed, tower displacement and tower velocity, and with just two inputs: pitch angle and generator power. The degrees of freedom that are excluded from this model are either subsumed as noise, e.g. blade oscillations, or are assumed to be controlled by auxiliary controllers that operate outside of the predictive controller at a much faster sample rate. An example of the latter is the pitch actuators. The wind turbine simulator used in this thesis contains a controller to drive the blades to a set pitch angle, overcoming the pitch system dynamics such as hydraulic pressure and various nonlinearities.

The techniques in this thesis can be extended to include any extra degrees of freedom that can be identified, provided the resulting optimisation can be solved online. Specialised solvers are available that have been shown in other applications, such as quadcopter control, to be capable of incorporating many states at high sample rates. This thesis employs an off-the-shelf solver because the focus is on the definition of the optimisation problem, not its solution.

1.4 Uncertainty in Wind Turbine Control

The wind that passes through the rotor disk (the swept area of the blades) is turbulent, varying spatially and temporally. To predict the aerodynamic forces with certainty, a very detailed wind speed measurement would be required and

the prediction model would have to be highly complex. In addition, future wind speeds that have not reached the rotor plane will affect future states. For these reasons, we accept that we cannot know what the wind speed is everywhere on the rotor disk and instead declare the uncertainty within the control problem. This makes the optimisation more complex, but by representing the uncertainty appropriately, the resulting optimisation problem is of the same order as that for the nominal problem, i.e. that arising from the assumption that the wind speed will not change from that most recently measured.

Additional uncertainty arises from the modelling process. A certain nonlinear system can be represented by an uncertain linear parameter varying (LPV) model, which may be conservative in order to include all possible dynamics around the linearisation points. Furthermore, coupling from any physical states that are ignored will contribute to differences between the predicted and subsequently realised modelled states. The scheduling parameter in model used in this work, around which linearisation are calculated, is the wind speed. Using an uncertain scheduling parameter introduces further challenges relating to the statistics of turbulence, which is studied in some detail in this thesis.

Techniques do exist to measure more of the wind field than wind turbines typically do. Light detection and ranging, or lidar, shines a laser into the oncoming wind and measures the Doppler shift of the reflection off dust particles or water droplets. Lidar systems that are specially designed for wind turbines can scan a volume of air in front of the rotor and provide a good estimation of the likely wind to reach the turbine in the short term future, as illustrated in Figure 1.2. However, there is a limit to the accuracy with which they can predict the evolution of turbulence over these distances. As such, uncertainty will still be present in the control problem.

This thesis does not use a lidar signal for two reasons. Firstly, any LPV representation of a wind turbine will be an uncertain model. Including lidar would not address that problem. Secondly, by exploiting the fewest possible signals that are specific to wind turbines, the resulting controller is more readily adapted to more diverse applications.

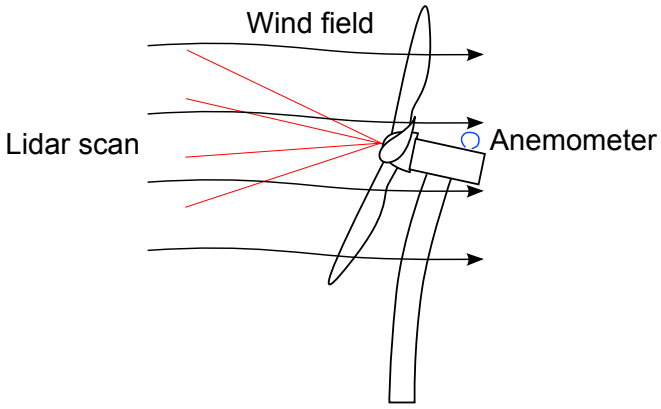

Figure 1.2: An illustration of two methods of measuring the wind field. A lidar system shines laser beams into the incoming wind. A nacelle anemometer measures the local wind speed at the tower top. In this thesis, only anemometer data is used.

1.5 Contributions

This thesis contributes in five ways. It takes a method for guaranteeing polytopic set invariance and develops it into a Robust MPC algorithm with recursive feasibility, suitable for models with multiplicative uncertainty, or both multiplicative and additive. The resulting constrained optimisation is a quadratic program with degrees of freedom that scale linearly with the horizon length.

It introduces a Stochastic MPC algorithm using one-step ahead probabilities in a field dominated by a radically different approach, that of scenario sampling. This departure allows for a controller with recursive feasibility despite individual constraint violations that occur with a probability up to a specified value.

It extends the development of the SMPC algorithm to produce a terminal set that is larger than anything that can be produced using only robust techniques. The algorithm grows the polytopic terminal set one vertex at a time, keeping track of the volume of the set, and guarantees the resulting set is positively invariant and satisfies the probabilistic constraints.

It extends the technique of using exponential basis functions in MPC to incorporate the novel RMPC algorithm. Optimisation variables directly affect all future predictions, as opposed to classical MPC where the basis functions are a sequence

of impulses. In this novel extension, the parameterisation of the sets that bound the uncertainty is also in terms of exponential basis functions, which produces a controller that gives promising results in closed-loop operation. It also has a more straightforward proof of recursive feasibility than the dual-mode RMPC controller, which may encourage adoption in industrial applications.

Finally, this thesis presents an RMPC algorithm that is used to control a complex nonlinear system by approximating it as an LPV system with only three states. The prediction uncertainty that arises as a result of model-reality mismatch and an uncertain scheduling parameter is bounded by polytopes. The work also contains a study into the statistics of turbulence, a brief discussion of data-driven model identification, and a thorough study of the benefit of RMPC over nominal MPC.

Contributions from this thesis have been published as follows. Conference papers on the contributions to RMPC and SMPC were presented at the American Control Conference 2012 and Conference on Decision and Control 2012. A brief paper on the application of RMPC to wind turbine control was published in IEEE Transactions on Control Systems Technology. A polytopic tube control scheme that simultaneously handles multiplicative uncertainty robustly and additive uncertainty probabilistically is to be presented at IFAC World Congress 2014.

1.6 Notation

The following notation is used throughout, though the meaning of specific characters varies between chapters.

Description	**Example/Abbrev.**
Sets are written in blackboard bold	$\mathbb{X}$
Real numbers, natural numbers, positive integers	$\mathbb{R}, \mathbb{Z}, \mathbb{Z}_+$
Matrices upper case, vectors lower, scalars either	M, v, N
Number of inputs, number of states	n_u, n_x
Transposed matrices and vectors	M^T, v^T
Such that, convex hull	s.t. , co
A model, a controller	$\mathcal{G}, \mathcal{K}$
Variable x at time step k	x_k
Predicted variable x at time step $k+i$ given measurements up to time step k	$x_{i\|k}$
Expectation of x at time k	$\mathrm{E}_k(x)$
Vector of ones	$\underline{1}$
Vector inequalities are element-wise	$v \leq \underline{1}$
Newly defined term	$\sigma := \sqrt{v}$
$P_1 - P_2$ is positive definite	$P_1 \succ P_2$
Superscript indexing in parethesis	$H^{(j)}$
Absolute value, 2-norm	$\|a-b\|, \|a-b\|$
Set cardinality, rounding down	$\|\mathbb{D}\|, \lfloor np \rfloor$
Candidate or optimal value	v^*
Convolution, complex conjugate	$v(t) * g(t), V^*(f)$
Elements of matrices implied by symmetry	$\begin{bmatrix} a & b \\ * & c \end{bmatrix}$

Chapter 2

Background

2.1 Chapter Outline

This chapter gives a brief overview of early work on MPC and continues through Robust MPC to Stochastic MPC. It starts by looking at how MPC naturally arises from discrete multi-input multi-output controllers as constraints are applied. As computational capability, not only processor speed but also algorithmic efficiency, has improved massively over the last few decades, what constitutes a 'solution' to a control problem has changed from a specific value to finding any algorithm that poses a numerical problem with a known solution. All MPC controllers in this chapter require the solution of a quadratic program (QP), that is the minimisation of a quadratic cost function subject to linear constraints.

Most MPC implementations require a terminal set. Section 2.2 demonstrates why, with references to important work that looked at stability of MPC at a time when much of its development lacked theoretical rigour. Section 2.3 returns to the subject of recursive feasibility, giving a formal definition and a proof that deterministic MPC is recursively feasible. Robustness in MPC with additive uncertainty is discussed in Section 2.4, followed by some background on the use of polytopes to bound state uncertainty. Sections 2.6 and 2.7 give two methods to find terminal sets by summarising two important set construction algorithms.

2.2 Deterministic MPC

A linear time invariant (LTI) model $\mathcal{G}_1$ is defined as follows:

$$\mathcal{G}_1 \ : \ x_{k+1} = Ax_k + Bu_k, \tag{2.1}$$

where $x \in \mathbb{R}^{n_x}$ and $u \in \mathbb{R}^{n_u}$.

The problem of minimising an infinite horizon quadratic cost on the state and inputs without constraints is defined as:

$$\min_{u_k} \sum_{k=0}^{\infty} x_k^T Q x_k + u_k^T R u_k \ \text{ s.t. } \ \mathcal{G}_1. \tag{2.2}$$

The solution to (2.2) is given by [Kalman, 1960]. It is the linear quadratic regulator (LQR), $\mathcal{K}_1$, defined as:

$$\mathcal{K}_1 \ : \ u_k = Kx_k, \quad K = -(R + B^T P B)^{-1} B^T P A, \tag{2.3}$$

where P is the solution to the discrete time algebraic Riccati equation. The feedback gain K is optimal for the unconstrained system.

Now we introduce linear state and input constraints to give a new problem:

$$\min_{u_k} \sum_{k=0}^{\infty} x_k^T Q x_k + u_k^T R u_k \tag{2.4a}$$

$$\text{s.t. } (2.1), \quad Fx_k + Gu_k \leq \underline{1}, \tag{2.4b}$$

where $\underline{1}$ is the vector of ones and where $F \in \mathbb{R}^{n_F \times n_x}$ and $G \in \mathbb{R}^{n_F \times n_u}$.

If we are given x_0 then (2.4) can be posed as a QP. Future predicted states are written as optimisation variables in terms of the inputs, which are degrees of freedom. The predicted states are $x_{i|k}$ and the predicted inputs are $u_{i|k}$. The subscript notation introduced here means the predicted value at time step $i + k$, given the known state at time step k, namely x_k. It is assumed the measurements provide perfect state information, and we denote $x_{0|k} = x_k$. The constrained

optimisation problem in terms of predicted states is thereby:

$$\min_{u_k} \sum_{i=0}^{\infty} x_{i|k}^T Q x_{i|k} + u_{i|k}^T R u_{i|k} \tag{2.5a}$$

$$\text{s.t.} \quad x_{i+1|k} = A x_{i|k} + B u_{i|k}, \quad F x_{i|k} + G u_{i|k} \leq \underline{1}. \tag{2.5b}$$

The quadratic program (2.5) is intractable because there are infinite degrees of freedom $u_{i|k}, i \to \infty$. The 'dual mode' paradigm provides a way to reduce the problem to one involving a finite number of degrees of freedom. Firstly we find a positively invariant region of state space where the controller $\mathcal{K}_1$ satisfies the constraints (2.4b). This is termed the terminal set $\mathbb{X}_f$, and methods for finding it are discussed in Sections 2.6 and 2.7. We use a polytopic terminal set here, so set membership can be checked by a set of linear inequalities.

The next stage in dual mode MPC design is to break the horizon into two modes. Mode 1 has degrees of freedom for the inputs $\mathbf{u}_k := [u_{0|k}^T \ \ldots \ u_{N-1|k}^T]^T$ and Mode 2 is under autonomous LQR control. The cost of the infinite horizon of Mode 2 is given by $x_{N|k}^T \bar{Q} x_{N|k}$, where $\bar{Q}$ is the solution of a Lyapunov equation. The dual mode control problem is as follows:

$$\min_{\mathbf{u}_k} \left(x_{N|k}^T \bar{Q} x_{N|k} + \sum_{i=0}^{N-1} x_{i|k}^T Q x_{i|k} + u_{i|k}^T R u_{i|k} \right) \tag{2.6a}$$

$$\text{s.t.} \quad x_{i+1|k} = A x_{i|k} + B u_{i|k} \tag{2.6b}$$

$$\text{and} \quad F x_{i|k} + G u_{i|k} \leq \underline{1}, \quad i < N. \tag{2.6c}$$

The horizon length N remains a decision variable. To solve (2.6), [Sznaier and Damborg, 1987] suggests choosing an initial value of N, solving (2.6), checking whether $x_{N|k} \in \mathbb{X}_f$ and increasing N if not. The authors note that if $x_{N|k}$ is in the terminal set, a lower cost cannot be achieved with a longer horizon. The work of [Scokaert and Rawlings, 1998] shows that successively increasing the horizon length will lead to a feasible solution in a finite number of steps, provided x_k lies in the region of attraction, the set of initial conditions from which the state can be driven to the origin subject to the constraints. The control strategy therefore includes an unbounded univariate search over N, evaluating a QP on every iteration. The

QP returns the first input in the optimised input sequence if and only if N is sufficiently large. The resulting controller is given by Algorithm 2.2.

Algorithm 1 Constrained LQR

Require: $N_0 \in \mathbb{Z}_+$
  $N \leftarrow N_0$
  repeat
    Solve QP (2.6) to get $u_{0|k}, x_{N|k}$
    Increase N
  until $x_{N|k} \in \mathbb{X}_f$
Implement: $\mathcal{K}_2 \ : \ u_k = u^*_{0|k}$

Subsequent predicted states contain terms in increasing powers of A so, to avoid numerical ill-conditioning, we re-state the prediction dynamics in terms of $\Phi := A+BK$, as in [Rossiter et al., 1998]. If (A, B) is stabilisable then it is possible to find such a K and in that case Φ is Hurwitz when K is the LQR feedback gain. The discrete-time autonomous system $x_{N+1|k} = \Phi x_{N|k}$ is asymptotically stable if all the eigenvalues of Φ are less than unity in magnitude.

Accordingly, the predicted inputs are decomposed as:

$$u_{i|k} = Kx_{i|k} + c_{i|k}, \tag{2.7}$$

where $c_{i|k} = 0$ for $i \geq N$.

The work of [Rawlings and Muske, 1993] shows that satisfaction of constraints (2.6c) at time at time k implies feasibility for all time after k, a property termed 'recursive feasibility', provided N is sufficiently long. The sequence of input degrees of freedom is concatenated and denoted $\mathbf{c}_k := [c^T_{0|k} \ \ldots \ c^T_{N-1|k}]^T$.

Controller $\mathcal{K}_2$ is optimal but should we desire a controller that executes only one QP per time step, we must pose a new problem whose solution might not be optimal. We now assume a fixed N and impose the terminal set constraint $x_{N|k} \in \mathbb{X}_f$ explicitly rather than checking for it a posteriori. The resulting constrained optimisation problem is as follows:

$$\min_{\mathbf{c}_k} \left(x^T_{N|k}\bar{Q}x_{N|k} + \sum_{i=0}^{N-1} x^T_{i|k}(Q + K^T RK)x_{i|k} + c^T_{i|k}Rc_{i|k} \right) \tag{2.8a}$$

$$\text{s.t.} \quad x_{i+1|k} = \Phi x_{i|k} + B c_{i|k}, \tag{2.8b}$$

$$(F + GK)x_{i|k} + G c_{i|k} \leq \underline{1} \tag{2.8c}$$

$$x_{N|k} \in \mathbb{X}_f. \tag{2.8d}$$

The corresponding controller evaluates the following algorithm, which requires that x_k is in the region of attraction for the constraints of (2.8), denoted $\mathbb{X}_\infty$. A fixed horizon length N restricts this region of attraction if state constraints are treated as hard (inviolate). An algorithm for computing $\mathbb{X}_\infty$ can be adapted from the maximal invariant set algorithm given in [Gilbert and Tan, 1991].

Algorithm 2 Nominal MPC

Require: $N \in \mathbb{Z}_+, \quad x_k \in \mathbb{X}_\infty$

Solve QP (2.8) to get $c_{0|k}$

Implement: $\mathcal{K}_3 \ : \ u_k = K x_k + c_{0|k}$

2.3 Recursive Feasibility

MPC solves a constrained optimisation problem at every time step. It seeks to minimise a cost function over the domain of some degrees of freedom, subject to a set of constraints. If the constrained space is non-empty then the problem is termed 'feasible'. If feasibility at time step k implies feasibility at time step $k+1$ then the control scheme is said to be recursively feasible.

The work of [Rawlings and Muske, 1993] considers linear deterministic systems subject to state and input constraints. The work shows that, for stable or stabilisable systems, a solution that satisfies constraints over a finite horizon, where the horizon is longer than the number of unstable modes, and assuming zero inputs beyond that horizon, implies satisfaction of all resulting future optimisation problems.

In [Scokaert et al., 1999], Lyapunov stability theory is extended for discontinuities in the control law for nonlinear systems. The proof that feasibility implies stability uses the concept of an invariant set. If the state is steered into such a set in a finite number of time steps, then a finite horizon controller is stable for infinite time.

Definition 1. An MPC controller is recursively feasible if and only if the feasibility of the constraints at a given time step implies feasibility at the successor time step.

For time invariant constraints, Definition 1 requires us to prove that there exist optimisation variables that satisfy the constraints, given optimisation variables that satisfied the constraints one time step prior.

Theorem 1. *Controller $\mathcal{K}_3$ is recursively feasible.*

Proof. Since the system being controlled is deterministic, $x_{i+1|k} = x_{i|k+1}$. Therefore, for every optimisation variable at time step $k+1$, the choice $c_{i|k+1} = c_{i+1|k}$ results in satisfaction of all constraints prior to the final time step in the horizon. For all i except $i = N-1$, this is obvious from the constraints of (2.8), whereas for $i = N-1$ this is demonstrated by the following argument. We previously enforced that $x_{N|k} \in \mathbb{X}_f$, therefore $x_{N-1|k+1} \in \mathbb{X}_f$, which is positively invariant, so $x_{N|k+1} \in \mathbb{X}_f$ under the Mode 2 dynamics, i.e. $c_{N-1|k+1} = c_{N|k} = 0$. Therefore all constraints are satisfied at the successor time step. □

2.4 Additive Robust MPC

Many real systems are subject to disturbances that can be modelled as polytopic bounded additive noise as in $\mathcal{G}_2$:

$$\mathcal{G}_2 \ : \ x_{k+1} = Ax_k + Bu_k + w_k \tag{2.9a}$$

$$w_k \in \mathbb{W} = \text{co}\{w^{(j)} : j = 1, \ldots, \rho\}. \tag{2.9b}$$

No knowledge of the uncertainty distribution is assumed except that it is bounded by the convex hull of the time invariant vertices $w^{(j)} : j = 1, \ldots, \rho$.

Assumption 1. $\mathbb{W}$ *contains the origin.*

To form linear constraints on the predicted states that are robust to all possible combinations of extreme values of the disturbances, as in [Campo and Morari, 1987], it may be helpful to see the first two terms in the state prediction sequence:

$$x_{1|k} = \Phi x_{0|k} + Bc_{0|k} + w_k \tag{2.10a}$$

$$x_{2|k} = \Phi(\Phi x_{0|k} + Bc_{0|k} + w_k) + Bc_{1|k} + w_{k+1}. \tag{2.10b}$$

Thus the predicted state $x_{i|k}$ depends on i independent uncertain terms $w_k, \ldots,$ w_{k+i-1}. Therefore, an exhaustive enumeration of all possible combinations of uncertainty requires a number of constraints that grows exponentially as ρ^i through the finite horizon. An alternative formulation in [Allwright and Papavasiliou, 1992] brought the growth down to linear in N for a restricted class of systems, namely linear systems with finite impulse response models.

Instead, we decompose the predictions into nominal and uncertain parts as in [Kouvaritakis et al., 2010]. The decomposed predictions evolve separately as follows:

$$x_{i|k} = z_{i|k} + \epsilon_{i|k}, \tag{2.11}$$

which have their own dynamics as follows, firstly for the deterministic component:

$$z_{i+1|k} = \Phi z_{i|k} + Bc_{i|k}, \qquad z_{0|k} = x_k \tag{2.12}$$

and secondly for the uncertain component:

$$\epsilon_i = \sum_{j=0}^{i-1} \Phi^j w_j. \tag{2.13}$$

Restated in terms of (2.11), the constraints of (2.8c) are:

$$\tilde{F}(z_{i|k} + \epsilon_{i|k}) + Gc_{i|k} \leq \underline{1}, \qquad i < N, \tag{2.14}$$

where $\tilde{F} := F + GK$. In this case, the uncertain parts are independent of the optimisation variables so (2.14) can be more conveniently written as

$$\tilde{F}z_{i|k} + Gc_{i|k} \leq \underline{1} - \tilde{F}\epsilon_i, \quad i < N. \tag{2.15}$$

Since (2.15) contains uncertain terms, it cannot be used for a constrained optimisation. Instead, we find the smallest allowable constraint tightening values

for each prediction step ahead:

$$h_{i+1} = h_i + \max_{w \in \mathbb{W}} \tilde{F}\Phi^i w, \quad i \in \mathbb{Z}, \quad h_0 = 0. \tag{2.16}$$

The constraints on the predicted variables in the finite horizon are re-written in terms of the constraint tighteners, as:

$$\tilde{F} z_{i|k} + G c_{i|k} \leq \underline{1} - h_i, \quad i = 0, \ldots, N-1, \tag{2.17}$$

while for $\bar{n}$ steps beyond the finite horizon, where $\bar{n}$ is a sufficiently large constant, under autonomous control, the constraints on the predicted variables are written:

$$\tilde{F} z_{N+i|k} \leq \underline{1} - h_{N+i}, \quad i = 0, \ldots, \bar{n}-1. \tag{2.18}$$

Beyond that extended horizon, predicted variables are constrained by an outer approximation to the tightening values, as:

$$\tilde{F} z_{N+\bar{n}+i|k} \leq \underline{1} - \bar{h}, \quad i \in \mathbb{Z} \tag{2.19a}$$

$$\bar{h} \geq h_i, \quad i \in \mathbb{Z}. \tag{2.19b}$$

The technique of switching from a sequence of constraint tightening terms to a constant term, shown in (2.19), was introduced in [Kouvaritakis et al., 2010].

Theorem 2. *h_i is monotonically increasing.*

Proof. Assumption 1 states that $\mathbb{W}$ contains the origin. Since Φ is real, then $\Phi^i w$ also contains the origin for $i \in \mathbb{Z}, \;\; w \in \mathbb{W}$. Now for any non-zero $\tilde{F}$,

$$\max_{w \in \mathbb{W}} \tilde{F}\Phi^i w \geq 0, \quad i \in \mathbb{Z},$$

so $h_{i+1} \geq h_i$ for $i \in \mathbb{Z}$. □

Now a terminal constraint is imposed as follows:

$$z_{N|k} \in \mathbb{H}_{n^*}, \quad \mathbb{H}_n := \big\{ z : F\Phi^i z \leq \underline{1} - h_{N+i}, \;\; i = 0, \ldots, n-1 \big\}, \tag{2.20}$$

where n^* is the minimum number of steps required such that subsequent sets are equal, i.e.

$$n^* = \inf\{n : \mathbb{H}_n = \mathbb{H}_{n+1}\}. \tag{2.21}$$

The definition of (2.20) follows the work of [Gilbert and Tan, 1991], which is explained in Section 2.7.

The distant horizon constraint tightening term $\bar{h}$ is introduced to prove that n^* is finite, but in practice its value is not required to be calculated since $\bar{n}$ can be chosen to be sufficiently large.

Theorem 3. *n^* is finite.*

Proof. If there exists an $n < \bar{n}$ such that $\mathbb{H}_n = \mathbb{H}_{n+1}$, then $n^* \leq n$. Otherwise, since Φ is asymptotically stable and $\bar{h}$ is constant, there exists a finite n such that $\Phi^n z$ is a member of $\mathbb{H}_n$ for all $z \in \mathbb{H}_n$, which implies $\mathbb{H}_{n+1} = \mathbb{H}_n$. □

Calculating $\mathbb{H}_{n^*}$ makes it possible to pose a tractable control problem that can be solved online to reach a robust control action. We minimise a cost based on the nominal state predictions:

$$\min_{\mathbf{c}_k} \left(z_{N|k}^T \bar{Q} z_{N|k} + \sum_{i=0}^{N-1} z_{i|k}^T (Q + K^T R K) z_{i|k} + c_{i|k}^T R c_{i|k} \right) \tag{2.22a}$$

$$\text{s.t.} \quad (2.12), (2.17), (2.20), \tag{2.22b}$$

where (2.12) governs the dynamics of the certain part of the predictions, (2.17) represents the mixed constraints in terms of optimisation variables, and (2.20) is the terminal constraint.

Algorithm 3 Additive Robust MPC

Require: $N \in \mathbb{Z}_+, \quad x_k \in \mathbb{X}_\infty$

Solve QP (2.22) to get $c_{0|k}$

Implement: $\mathcal{K}_4 \ : \ u_k = K x_k + c_{0|k}$

Theorem 4. *Controller $\mathcal{K}_4$ is recursively feasible.*

Proof. Under the uncertain dynamics, $z_{i|k+1} = z_{i+1|k} + \Phi^i w_k$. For the purpose of this proof, choose $c_{i|k+1} = c_{i+1|k}$ for $i < N-1$ and $c_{N-1|k+1} = 0$. Therefore for $i < N$,

$$\tilde{F} z_{i|k+1} + G c_{i|k+1} + h_i = \tilde{F} z_{i+1|k} + G c_{i+1|k} + h_i + \tilde{F}\Phi^i w_k,$$

where $w_k \in \mathbb{W}$. By construction in (2.16), $\tilde{F}\Phi^i w_k + h_i \leq h_{i+1}$ so

$$\tilde{F} z_{i|k+1} + G c_{i|k+1} + h_i \leq \tilde{F} z_{i+1|k} + G c_{i+1|k} + h_{i+1},$$

which we know is no greater than unity because the problem was feasible at time step k.

For $i = N$, i.e. checking that the terminal constraint is recursively feasible, note that by definition of n^*,

$$\mathbb{H}_{n^*} = \left\{ z : F\Phi^i z \leq \underline{1} - h_{N+i}, \quad i = 0, \ldots, n^* - 1 \right\} = \mathbb{H}_{n^*+1}$$

and so

$$\begin{aligned} z_{N|k} \in \mathbb{H}_{n^*} \Rightarrow z_{N|k} &\in \left\{ z : F\Phi^{i+1} z \leq \underline{1} - h_{N+i+1}, \quad i = 0, \ldots, n^* - 1 \right\} \\ &= \left\{ z : F\Phi^i \Phi z \leq \underline{1} - h_{N+i} - \Phi^{N+i} w, \quad i = 0, \ldots, n^* - 1, \quad w \in \mathbb{W} \right\} \\ &= \left\{ \Phi z + \Phi^{N+i} w : F\Phi^i z \leq \underline{1} - h_{N+i}, \quad i = 0, \ldots, n^* - 1, \quad w \in \mathbb{W} \right\} \\ \Rightarrow \Phi z_{N|k} + \Phi^{N+i} w_k &\in \left\{ z : F\Phi^i z \leq \underline{1} - h_{N+i}, \quad i = 0, \ldots, n^* - 1 \right\} = \mathbb{H}_{n^*}, \end{aligned}$$

and $z_{N|k+1} = \Phi z_{N|k} + \Phi^{N+i} w_k$.

□

2.5 Polytopic Set Invariance

This section discusses a technique to find a linear constraint that, when applied to a polytopic set, is sufficient to guarantee that set is invariant under some given dynamics. This technique is applied to MPC in Section 3.2.

A polytopic set, or polytope, is a bounded intersection of half-spaces. It can be written as an inequality in the state space vector x. The polytope $\mathbb{P}$ is here defined as:

$$\mathbb{P} := \{x : Vx \leq \alpha\}, \tag{2.23}$$

where each row of $V \in \mathbb{R}^{n_V \times n_x}$ and corresponding element of $\alpha \in \mathbb{R}^{n_V}$ define one half-space.

We may wish to know whether this polytope is invariant under the dynamics $x_{k+1} = \Phi x_k$, i.e. that $x_k \in \mathbb{P}$ implies $x_{k+1} \in \mathbb{P}$. Necessary and sufficient conditions for this requirement are given in [Bitsoris, 1988], which are that there must exist a matrix H with non-negative elements such that:

$$HV = V\Phi, \tag{2.24}$$

and such that $H\alpha \leq \alpha$. Constraint (2.24) is linear in H so a linear program can be used to find each row of H.

The technique is applied to a control system in [Hennet, 1989], wherein such non-homogeneous matrix inequalities are applied in two ways. Firstly, conditions are constructed for a positively invariant set that contains the origin. Secondly, conditions are constructed that enforce that any state in that positively invariant set satisfies given state constraints and that the input $u_k = Kx_k$ satisfies given input constraints, where K is a gain matrix to be calculated online. By imposing all of these conditions upon the choice of K, the control system guarantees that all future states and all future control actions will be admissible. This thesis will extend the work of [Hennet, 1989] to linear models with multiplicative uncertainty.

State and input constraints can be more generally posed as mixed constraints $Fx + Gu \leq g$, which, for inputs defined by linear feedback $u = Kx$, can be written

$$\tilde{F}x = (F + GK)x \leq g. \tag{2.25}$$

The positively invariant set $\mathbb{P}$ requires, as before, that a matrix with non-negative elements H is found such that $HV = V(A + BK)$ and $H\alpha \leq \alpha$. For this set to include the origin, $\alpha \geq 0$. To impose that all states in $\mathbb{P}$ satisfy (2.25), a

matrix with non-negative elements H_m is found such that

$$H_m V = \tilde{F}, \quad H_m \alpha \leq g. \tag{2.26}$$

Now any choice of (K, α) that satisfies (2.25) and (2.26) gives a suitable linear controller.

2.6 Maximal Invariant Ellipsoid

In the design of some MPC controllers, an invariant ellipsoidal set $\mathbb{E}$ is required, elements $x \in \mathbb{E}$ of which satisfy the constraint $\tilde{F}x \leq \underline{1}$, where $\tilde{F} \in \mathbb{R}^{n_F \times n_x}$ consists of rows f_h^T, $h = 1, \ldots, n_F$. A convenient way to define such an ellipsoid is as follows:

$$\mathbb{E} := \left\{ x : x^T S x \leq 1 \right\}, \tag{2.27}$$

where S is a positive definite matrix such that $x \in \mathbb{E} \Rightarrow \Phi x \in \mathbb{E}$ and such that $x \in \mathbb{E} \Rightarrow \tilde{F}x \leq \underline{1}$. This section demonstrates a method to find S such that the volume of $\mathbb{E}$ is maximised. Maximum volume ellipsoids may be desired for example as a terminal set.

We proceed to give a method for finding the maximum invariant ellipsoid, the result of which is illustrated in Figure 2.1, under the following constrained dynamics:

$$\tilde{F} = f^T = \begin{bmatrix} 1 & 1 \end{bmatrix}, \quad \Phi = \begin{bmatrix} 0.8 & 0.1 \\ -0.5 & 0.8 \end{bmatrix}. \tag{2.28}$$

If $x^T \Phi^T S \Phi x < x^T S x$ for all $x \in \mathbb{E}$ then $\mathbb{E}$ is invariant under the dynamics $x_{k+1} = \Phi x_k$. This can be expressed, equivalently, as the linear matrix inequality (LMI)

$$\Phi^T S \Phi \prec S. \tag{2.29}$$

Each row of the constraint matrix, $f_h^T x \leq 1$ can be written as $\left| f_h^T S^{-\frac{1}{2}} S^{\frac{1}{2}} x \right|^2 \leq 1$. This is implied, via the Cauchy-Schwarz inequality, by

$$\left\| f_h^T S^{-\frac{1}{2}} \right\|^2 \left\| S^{\frac{1}{2}} x \right\|^2 \leq 1 \tag{2.30}$$

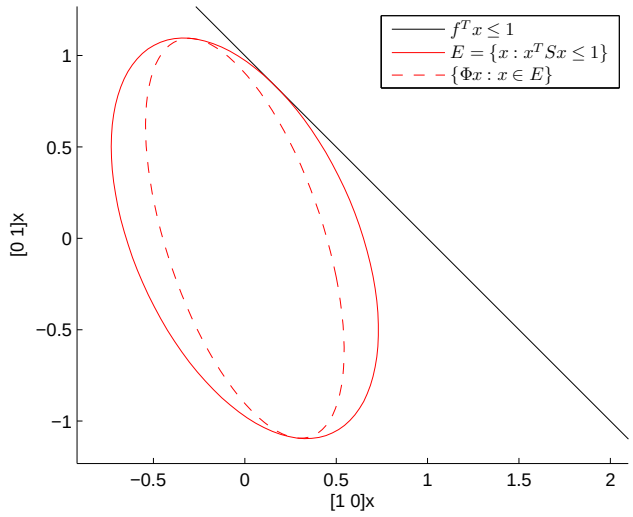

Figure 2.1: Linear constraint, maximal invariant ellipsoid, and set of successor states.

and equivalently,

$$f_h^T S^{-1} f_h \cdot x^T S x \leq 1. \tag{2.31}$$

The constraints $f_h^T x \leq 1,\ h = 1, \ldots, n_F$ need only hold when $x^T S x \leq 1$. Therefore, a condition that is necessary and sufficient is $f_h^T S^{-1} f_h \leq 1$. The volume of an ellipsoid defined as in (2.27) monotonically increases with the determinant of S^{-1} so maximising the volume of $\mathbb{E}$ subject to (2.29) is posed as the following problem, as in [Boyd et al., 1994]:

$$\begin{aligned} \max_{S^{-1}} \det S^{-1} \quad \text{s.t.} \quad & S - \Phi^T S \Phi \succ 0 \\ & f_h^T S^{-1} f_h \leq 1, \quad h = 1, \ldots, n_F \end{aligned} \tag{2.32}$$

Problem (2.32) is non-convex, but can be equivalently expressed in terms of S^{-1} by pre- and post-multiplying both sides of the LMI (2.29) by S^{-1} to give:

$$S^{-1} \Phi^T S \Phi S^{-1} \succ S^{-1}, \tag{2.33}$$

which can be incorporated using Schur complements.

Corollary 1. *The linear matrix inequality*

$$\begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \succ 0,$$

where $A = A^T$ and $C = C^T$, is equivalent to its Schur complement:

$$C \succ 0, \qquad A - BC^{-1}B^T \succ 0.$$

Now we have the following solvable SDP:

$$\begin{aligned} \max_{S^{-1}} \det S^{-1} \quad \text{s.t.} \quad & \begin{bmatrix} S^{-1} & S^{-1}\Phi^T \\ \Phi S^{-1} & S^{-1} \end{bmatrix} \succ 0 \\ & f_h^T S^{-1} f_h \leq 1, \qquad h = 1, \ldots, n_F. \end{aligned} \tag{2.34}$$

2.7 Maximal Invariant Set

The ellipsoidal invariant set is convenient because it can be calculated by solving an LMI problem, for which there are fast numerical methods. However it is in general conservative, especially if the constraints are asymmetric. A polytope defines a more flexible set but the method for finding the maximum polytopic invariant set is more involved. Important work in this field is given in [Gilbert and Tan, 1991], the first to give an algorithm to find the maximal output admissible set. When the constraint set is polyhedral, that algorithm constitutes a sequence of linear programs. This section begins with some definitions and proceeds to describe the method of [Gilbert and Tan, 1991] for finding the maximum invariant set for a discrete deterministic system.

A linear constraint $f^T x \leq 1$ is 'redundant' for set $\mathbb{X}_i$ if the intersection of $\mathbb{X}_i$ and $\mathbb{X}_{f,1}$ is equal to $\mathbb{X}_i$, where

$$\mathbb{X}_{f,1} := \{x : f^T x \leq 1\}. \tag{2.35}$$

Equivalently, $f^T x \leq 1$ is redundant if the relative complement $\mathbb{X}_i \setminus \mathbb{X}_{f,1}$ is the empty set, i.e.

$$\left\{x : x \in \mathbb{X}_i, \quad f^T x > 1\right\} = \emptyset. \tag{2.36}$$

A block of constraints $\tilde{F}x \leq \underline{1}$ is redundant if every row $f_h^T x \leq 1$ is redundant, for $h = 1, \ldots, n_F$, where n_F is the number of rows of $\tilde{F}$.

A polytope $\{x : H_n x \leq \underline{1}\}$ is invariant under the dynamics $x_{k+1} = \Phi x_k$ if

$$H_n x \leq \underline{1} \Rightarrow H_n \Phi^i x \leq \underline{1}, \quad i \in \mathbb{Z}_+. \tag{2.37}$$

The maximum invariant polytope shall be shown to be equal to $\mathbb{X}_{n^*}$ where n^* is finite and can be calculated by the following procedure from [Gilbert and Tan, 1991]. The procedure begins by initialising $H_0 = \tilde{F}$ and constructs successively highly facetted sets by concatenating block rows. The algorithm terminates when all constraints in the block are redundant. The algorithm can be stated formally as follows, where m is a binary 'flag' variable, indicating whether more constraints are required. Figure 2.2 is an illustration of this method applied to the constrained dynamics (2.28).

Algorithm 4 Maximal invariant set

Require: $\tilde{F} \in \mathbb{R}^{n_F \times n_x}$, $\Phi \in \mathbb{R}^{n_x \times n_x}$
  $i \leftarrow 0$, $H_0 \leftarrow \tilde{F}$, $m \leftarrow 1$
  while $m = 1$ **do**
    $m \leftarrow 0$
    for $h = 1, \ldots, n_F$ **do**
      if $\exists x$ s.t. $H_i x \leq \underline{1}$, $f_h^T \Phi^{i+1} x > 1$ **then**
        $m \leftarrow 1$
      end if
    end for
    if $m = 1$ **then**
      $H_{i+1} \leftarrow \left[H_i^T \ \tilde{F}\Phi^{i+1} \right]^T$
      $i \leftarrow i + 1$
    end if
  end while
Implement: $\mathbb{X}_{n^*} = \{x : H_i x \leq \underline{1}\}$

Theorem 5. *If there exists no x such that $H_n x \leq \underline{1}$ and $f_h^T \Phi^{n+1} x > 1$, for any row f_h^T in $\tilde{F}$, then $\mathbb{X}_n = \mathbb{X}_{n+i}$ for all positive i.*

Proof. If all constraints $f_h^T \Phi^{n+1} x \leq 1$, $h = 1, \ldots, n_F$ are redundant for $\mathbb{X}_n$, then $\tilde{F}\Phi^{n+1} x \leq \underline{1}$, so $\mathbb{X}_n = \mathbb{X}_{n+1}$, which is therefore implied by the satisfaction of a

sequence of constraints, namely:

$$\tilde{F}x \leq \underline{1},\ \tilde{F}\Phi x \leq \underline{1},\ \ldots,\ \tilde{F}\Phi^n x \leq \underline{1}.$$

Now define $y := \Phi^i x$ for any positive i. The same sequence of constraints, substituting y for x, now implies $\tilde{F}\Phi^{n+1}y \leq \underline{1}$, which means $\mathbb{X}_{n+i} = \mathbb{X}_{n+i+1}$. □

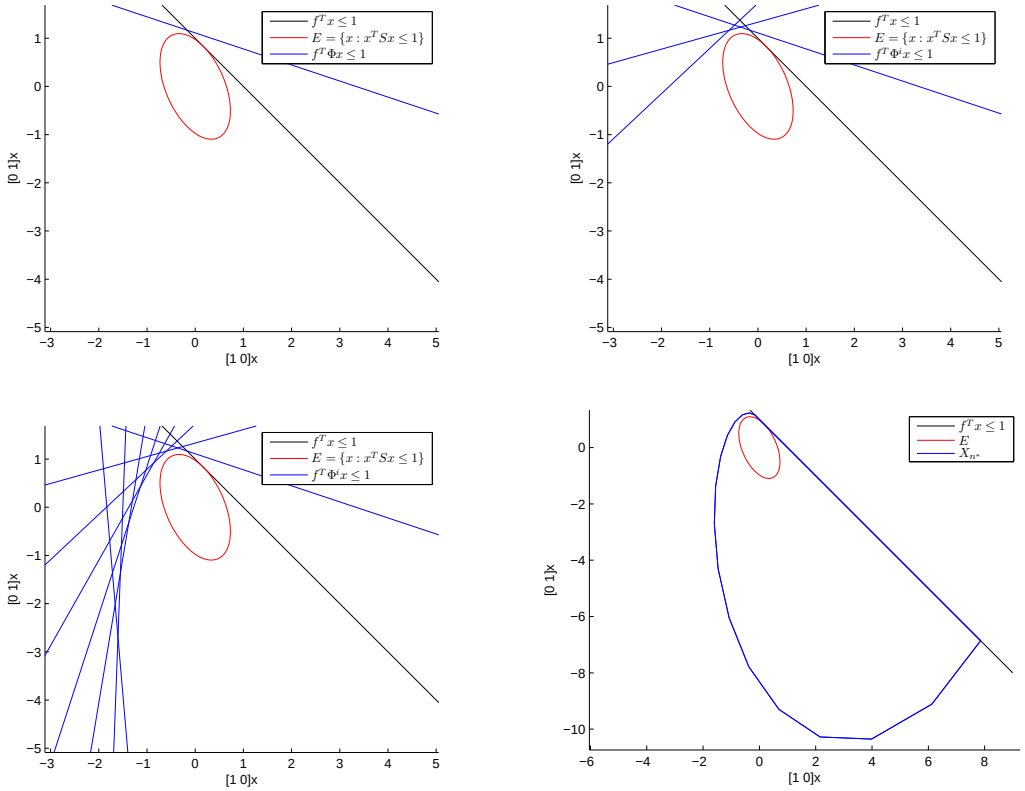

Figure 2.2: The construction of the maximum invariant set, showing successively added constraints. Each plot shows the maximum invariant ellipsoid and the linear constraint $f^T x \leq 1$. The first three plots also show $f^T \Phi^i x \leq 1$ for $i = 1$, $i = 1, 2, 3$ and $i = 1, \ldots, 8$ respectively. The final plot shows $\mathbb{X}_{n^*}$ where n^* was found to be 15 in this example.

Chapter 3

Robust and Stochastic Prediction Strategies

3.1 Chapter Outline

In Section 2.4, a Robust MPC algorithm was given consisting of the solution of a quadratic programming problem. That solution guarantees satisfaction of state and input constraints at all future time steps and is recursively feasible, meaning that the same problem at the next time step is guaranteed to have a solution. The system model class was restricted to additive uncertainty only.

A wider class of system can be modelled by also accounting for multiplicative uncertainty in the form:

$$x_{k+1} = (A + \Delta_k)x_k + (B + \Gamma_k)u_k + w_k.$$

The approach to handling the additive uncertainty was to calculate constraint tightening values as in (2.19) and then treat the remaining problem as deterministic. A similar process performed on a problem involving both multiplicative and additive uncertainty will remove the additive part of the problem by introducing appropriately tightened constraints. So to aid simplicity of presentation, in this chapter we set $w_k = 0$ without loss of generality. Chapter 5 tackles multiplicative and additive uncertainty by a decomposition of the predictions, hence it is unnecessary to complicate this chapter with the extra notation.

In Section 3.2, a Robust MPC algorithm is given to control systems modelled by multiplicative uncertainty. The technique presented uses a polytope to bound the state and input predictions in the finite horizon. This is motivated by the need to avoid the computational intractability of constraining a control problem by all combinations of extreme values of uncertainty in a prediction sequence. Constraints are then created in terms of the bounding polytopes, which ensure the predicted states and input satisfy constraints despite being uncertain.

Stochastic MPC is increasingly attractive due to its ability to reduce the conservativeness of the closed loop behaviour relative to RMPC. Typically, probabilistic constraints are non-convex [Prékopa, 1995], so lead to intractable control problems. Section 3.3 gives an overview of two methods of using samples taken from the probability distribution to create approximate probabilistic constraints. Criteria for choosing the number of such samples are discussed. An SMPC algorithm is presented that retains the recursive feasibility property of RMPC, in contrast to the majority of recent research in the area. The sampled probabilistic problem is harder to solve than the quadratic programs required in RMPC. An approximate 'greedy' solution to this specific type of sampled problem is presented, which has computational complexity only a constant times higher than a QP.

3.2 Multiplicative Robust MPC

An LPV model $\mathcal{G}_3$ is defined as follows:

$$\mathcal{G}_3 \ : \ x_{k+1} = (A + \Delta_k)x_k + (B + \Gamma_k)u_k \tag{3.1a}$$

$$(\Delta_k, \Gamma_k) \in \text{co}\left\{(\Delta^{(j)}, \Gamma^{(j)}) : j = 1, \ldots, \rho\right\} = \mathbb{D}_c \tag{3.1b}$$

As before, we find a stabilising feedback gain K and define $\Phi = A + BK$. We then decompose the prediction dynamics, introducing an uncertain prediction $e_{i|k}$:

$$x_k = z_{0|k} + e_{0|k} \tag{3.2a}$$

$$z_{i+1|k} = \Phi z_{i|k} + Bc_{i|k} \tag{3.2b}$$

$$e_{i+1|k} = \tilde{\Delta}_{k+i} z_{i|k} + (\Phi + \tilde{\Delta}_{k+i})e_{i|k} + \Gamma_{k+i} c_{i|k} \tag{3.2c}$$

$$u_{i|k} = Kz_{i|k} + c_{i|k} \tag{3.2d}$$

where $\tilde{\Delta}_{k+i} := \Delta_{k+i} + \Gamma_{k+1}K$. It will also be useful to define the following set:

$$\mathbb{D} := \{\Delta + \Gamma K : (\Delta, \Gamma) \in \mathbb{D}_c\}. \tag{3.3}$$

Unlike in Section 2.4, the effect of uncertainty on the constraints depends on the optimisation variables, so there is no offline computation that will allow us to pose an equivalent problem with certain dynamics. Instead, the number of possible extreme values of the uncertainty predictions grows exponentially through the horizon. This exponential growth, in general, leads to a computationally intractable problem. In [Barmish and Sankaran, 1979], it is suggested that the exponential growth can be alleviated by occasional 'rectangularisation', whereby an outer approximation to the set of possible extreme values of the uncertain predictions is constructed at stages along the prediction horizon, breaking it into several sub-horizons. This bounding outer approximation to the uncertain predictions is an early form of the polytopic Robust MPC algorithm given in this section. Indeed, preventing the exponential growth of constraint sets over the horizon by outer-bounding polytopes is mentioned in [Schuurmans and Rossiter, 2000] but not pursued.

Bounding sets by polytopes that are described by their vertices would mean that enforcing linear constraints for the whole set would require only enforcing them for the vertices. However, the way the uncertainty propagates at each time step would require constraints on products of optimisation variables. Such constraints would not in general be convex, so cannot be considered computationally tractable. Instead here we bound the uncertainty sets by polytopes defined by their facets, requiring only linear constraints and some offline calculation. We begin by stating the bounding condition:

$$Ve_{i|k} \leq \alpha_{i|k}, \qquad i < N \tag{3.4}$$

where each row of the matrix $V \in \mathbb{R}^{n_V \times n_x}$ is the transpose of a vector defining a facet normal of the bounding polytope and where $\alpha_{i|k} \in \mathbb{R}^{n_V}$ is a vector of optimisation variables that completes the definition of the hyperplane containing

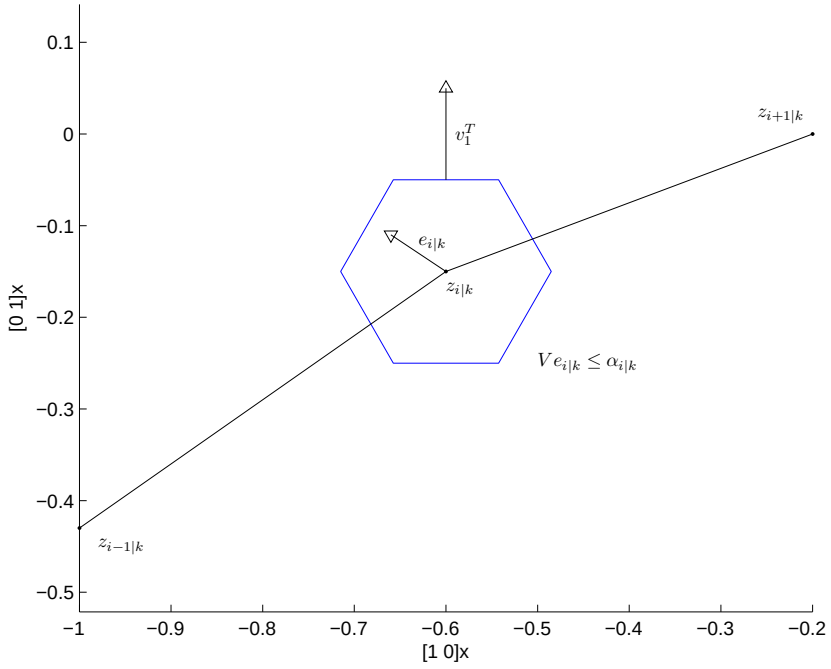

Figure 3.1: An example nominal prediction sequence in black; uncertainty set around the nominal prediction at step $k+i$ in blue; uncertain error in prediction at the same step show as a vector; first bounding polytope facet normal vector v_1^T, where v_1 is the first row of V and $n_V = 6$.

each facet. If the rows of V are normalised, then each element of $\alpha_{i|k}$ is the normal displacement of the corresponding hyperplane from the origin in e-space. Figure 3.1 shows an example polytopic set bounding the uncertain predictions. It is useful to write these polytopes alternatively in set notation, as follows:

$$\mathbb{S}_{i|k} := \{e : Ve \leq \alpha_{i|k}\}. \tag{3.5}$$

The uncertain prediction $e_{i+1|k}$ depends on all uncertainty that enters the system up to prediction step i. To avoid an exponential growth in the number of constraints on these uncertain predictions, we form constraints that depend on only one step of uncertainty, in addition to the parameterisation of the previous

bounding set. A necessary condition for $e_{i+1|k} \in \mathbb{S}_{i+1|k}$ is:

$$\tilde{\Delta}_{k+i} z_{i|k} + (\Phi + \tilde{\Delta}_{k+i}) e + \Gamma_{k+i} c_{i|k} \in \mathbb{S}_{i+1|k} \quad \forall e \in \mathbb{S}_{i|k}. \tag{3.6}$$

The sequence of bounding sets is termed a 'tube', more specifically here, a polytopic tube. Combining (3.2c), (3.4) and (3.6) gives:

$$V e_{i|k} \leq \alpha_{i|k} \Rightarrow V(\tilde{\Delta}_{k+i} z_{i|k} + (\Phi + \tilde{\Delta}_{k+i}) e_{i|k} + \Gamma_{k+i} c_{i|k}) \leq \alpha_{i+1|k}. \tag{3.7}$$

Finding linear constraints that guarantee (3.7) was solved in [Hennet, 1989] by extending the Farkas Lemma. More detail on this work and relevant prior work is given in Section 2.5. For the robust case we must repeat the treatment for each vertex in the uncertainty, which we treat separately. First we find matrices $H^{(j)}, j = 1, \ldots, \rho$ with non-negative elements such that:

$$H^{(j)} V = V \left(\Phi + \tilde{\Delta}^{(j)} \right), \tag{3.8}$$

where $\tilde{\Delta}^{(j)} := \Delta^{(j)} + \Gamma^{(j)} K$.

The degrees of freedom in matrices $H^{(j)}$ could be exploited online to minimise the cost function. However, this would add significant computational complexity, so instead those degrees of freedom are given up offline, by minimising the sum of the elements of $H^{(j)}$ subject to (3.8). Multiplying both sides of (3.4) by $H^{(j)}$, substituting (3.8) into (3.7) and rearranging gives:

$$H^{(j)} V e_{i|k} \leq H^{(j)} \alpha_{i|k} \Rightarrow H^{(j)} V e_{i|k} \leq \alpha_{i+1|k} - V \left(\tilde{\Delta}^{(j)} z_{i|k} + \Gamma^{(j)} c_{i|k} \right), \tag{3.9}$$

which can be imposed online as follows:

$$H^{(j)} \alpha_{i|k} \leq \alpha_{i+1|k} - V \left(\tilde{\Delta}^{(j)} z_{i|k} + \Gamma^{(j)} c_{i|k} \right), \quad i < N \tag{3.10a}$$

$$\alpha_{0|k} \geq V e_{0|k} \tag{3.10b}$$

The rationale behind minimising the sum of the elements of $H^{(j)}$ offline is to relax the condition in (3.10a) as much as possible.

With $H^{(j)}$ fixed offline, (3.10) creates an outer approximation to the optimal

solution. The conservativeness of this outer approximation can be decreased by increasing n_V. The number of constraints that are applied online scales only linearly with n_V, so the conservativeness can be reduced significantly with little extra computational effort. In addition, the facets need not be evenly distributed. Various applications may benefit from more facets in some directions than in others. The α-variables are optimisation variables, so it is important to use a QP solver that is appropriate for high-dimensional problems with relatively few constraints, known as large-scale, or sparse. Additional computational savings can be made by warm-starting the solver with the results from the previous time step, shifted by one time step according to $\alpha_{i|k+1} = \alpha_{i+1|k}$.

To ensure that the state will be steered into the terminal set $\mathbb{X}_f$ by the end of the finite horizon, we need to impose a set constraint. In the case where there is no uncertainty, the terminal constraint is simply $x_{N|k} \in \mathbb{X}_f$. In the robust and stochastic cases, we require all possible states to reach the terminal set, so the set $\{z_{N|k} + e : Ve \leq \alpha_{N|k}\}$ must be a subset of the terminal set $\mathbb{X}_f = \{x : H_t x \leq \underline{1}\}$. This is illustrated in Figure 3.2. This robust terminal constraint can be invoked through the use of a matrix with non-negative elements H_f, which can be computed offline by LPs to satisfy:

$$H_f V = H_t. \tag{3.11}$$

Then online, the terminal constraint is:

$$H_f \alpha_{N|k} \leq \underline{1} - H_t z_{N|k}. \tag{3.12}$$

For the mixed state and input constraints the same process is required once more, stated here without further explanation as follows, where the subscript m signifies the use of the matrix is for enforcing mixed constraints:

$$H_m V = \tilde{F}, \qquad H_m \geq 0 \tag{3.13a}$$

$$\tilde{F} z_{i|k} + H_m \alpha_{i|k} + G c_{i|k} \leq \underline{1}, \qquad i < N. \tag{3.13b}$$

We now pose the quadratic programming problem and state the controller. As in the nominal MPC algorithm, the cost is defined in terms of the nominal state

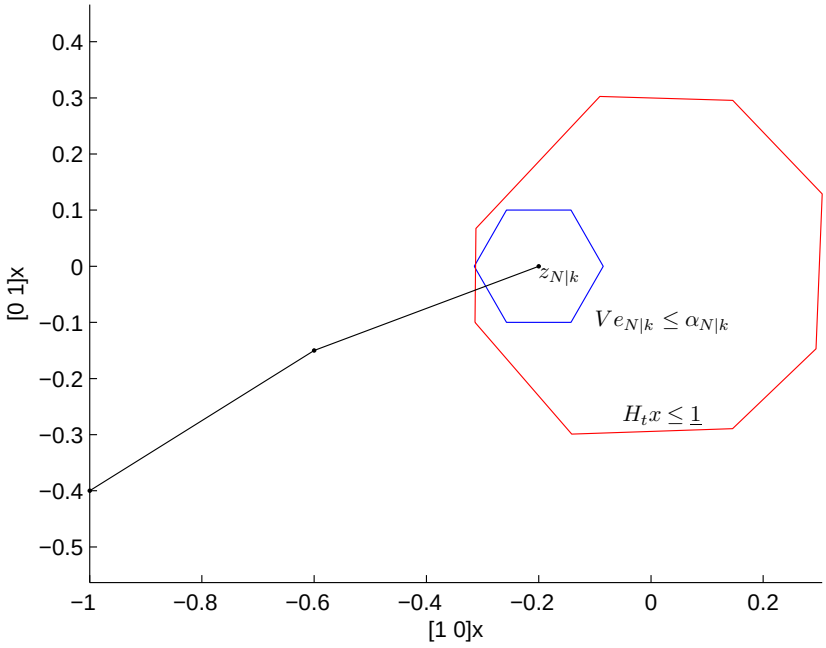

Figure 3.2: Uncertainty set $\{z_{N|k} + e : Ve \leq \alpha_{N|k}\}$ around nominal terminal prediction point $z_{N|k}$, shown to lie inside terminal set $\{x : H_t x \leq \underline{1}\}$.

predictions. The value that is actually applied to the plant is again the optimal $c^*_{0|k}$ but RMPC minimises over a larger space: the input degrees of freedom $\mathbf{c}_k$, the polytope definition vectors (stacked into a vector $\mathbf{a}_k$) and the initial uncertainty $e_{0|k}$, where

$$\mathbf{a}_k := [\alpha^T_{1|k}, \ldots, \alpha^T_{N|k}]^T. \tag{3.14}$$

The QP to be solved is a minimisation of a cost function that is quadratic in the input degrees of freedom, subject to constraints that are linear in x_k, $\mathbf{c}_k$, $\mathbf{a}_k$ and $e_{0|k}$.

$$\begin{aligned} \min_{\mathbf{c}_k, \mathbf{a}_k, e_{0|k}} & \left(z^T_{N|k} \bar{Q} z_{N|k} + \sum_{i=0}^{N-1} z^T_{i|k}(Q + K^T R K) z_{i|k} + c^T_{i|k} R c_{i|k} \right) \\ \text{s.t.} \quad & (3.2), (3.10), (3.12), (3.13\text{b}), \end{aligned} \tag{3.15}$$

where (3.2) governs the dynamics of the nominal predictions, (3.10) are constraints

on the optimisation variables that define the bounding polytope, (3.12) constrains the terminal bounding polytope, and (3.13b) constrains the polytopic tube to respect the mixed constraints.

Algorithm 5 Multiplicative Robust MPC

Require: $N \in \mathbb{Z}_+, \quad x_k \in \mathbb{X}_\infty$

Solve QP (3.15) to get $c_{0|k}$

Implement: $\mathcal{K}_5 \ : \ u_k = Kx_k + c_{0|k}$

Due to the conservativeness of the polytopic uncertainty sets, this controller is not recursively feasible. To illustrate this, propose values of $\mathbf{c}_{k+1}$ by shifting each element forward by one per time step and replace the final time step with zeros:

$$\mathbf{c}_{k+1} := M\mathbf{c}_k, \tag{3.16}$$

where M is the zero matrix with $n_u \times n_u$ identity matrices on its super-diagonal blocks, so that

$$M\mathbf{c}_k = \begin{bmatrix} c_{1|k}^T & \cdots & c_{N-1|k}^T & 0 \end{bmatrix}^T. \tag{3.17}$$

Doing the same for $\mathbf{a}_{k+1}$, i.e. $\alpha_{N|k+1} = 0$, is clearly not appropriate. Indeed there is no method to find a value for $\alpha_{N|k+1}$ that guarantees it will lead to a feasible control problem. An example of such an infeasibility is shown in Figure 3.3. The nominal predicted terminal state $z_{N|k}$ is shown. Also shown is the corresponding uncertainty set $\mathbb{S}_{N|k}$ and its vertices. The uncertainty set is a subset of the terminal set $\mathbb{X}_f$, which is positively invariant.

Under the autonomous controlled dynamics, the successor state Φx of any state $x \in \mathbb{X}_f$ is a member of $\mathbb{X}_f$. If the terminal set is robustly positively invariant, the same holds for successor states $(\Phi + \tilde{\Delta})x$ for $\tilde{\Delta} \in \mathbb{D}$ but this is omitted from the figure for clarity. Therefore, while selecting a candidate successor nominal predicted terminal state $z^*_{N|k+1} = \Phi z_{N|k}$ results in the successor vertices to $\mathbb{S}_{N|k}$ being members of the terminal set, there is no possible candidate uncertainty set bounding variable $\alpha^*_{N|k+1}$ such that the corresponding candidate uncertainty set $\mathbb{S}^*_{N|k+1}$ is a subset of the terminal set.

Three options are now presented as methods to handle the lack of recursive feasibility of Algorithm 3.2.

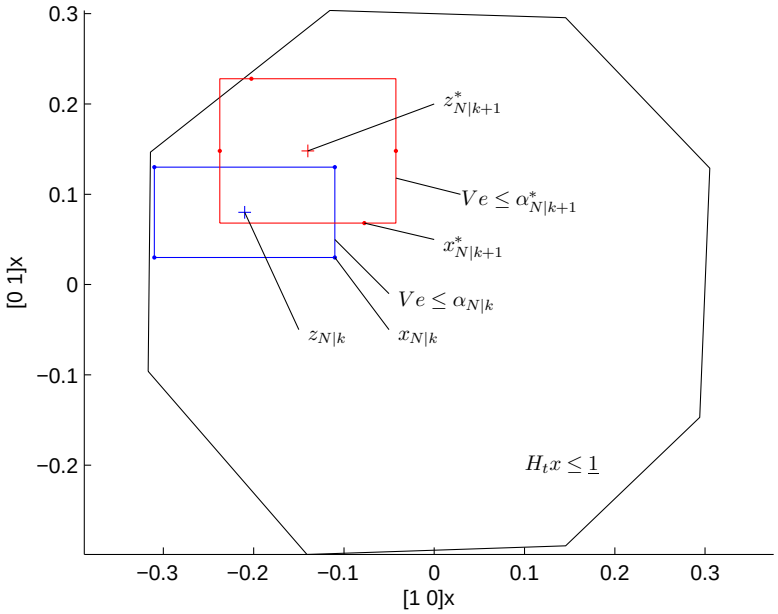

Figure 3.3: Terminal set $\mathbb{X}_f = \{x : H_t x \leq \underline{1}\}$ shown in black. Predicted nominal state $z_{N|k}$, points $x_{N|k} = z_{N|k} + e_{N|k}$ where $e_{N|k}$ are extreme vertices of uncertainty set $\mathbb{S}_{N|k}$, and set $\mathbb{S}_{N|k} = \{e : Ve \leq \alpha_{N|k}\}$ shown in blue. Candidate successor predicted nominal state $z^*_{N|k+1} = \Phi z_{N|k}$, candidate successor points $x^*_{N|k+1} = \Phi x_{N|k}$ and tightest set surrounding those points shown in red. The tightest set is defined by $\{z^*_{N|k+1} + e : Ve \leq \alpha^*_{N|k+1}\}$ where $\alpha^*_{N|k+1} = \max Ve^*_{N|k+1}$.

1. Reduce the horizon length where required

In the event of a terminal set infeasibility, a sub-optimal solution is to reduce the horizon length by one step. This will necessarily lead to a feasible optimisation problem. On the other hand, if it is possible to find a feasible $\alpha_{N|k+1}$, then the full horizon can be used, allowing the optimiser to reduce the cost of the solution. This is illustrated in Figure 3.4.

The least computationally expensive way of implementing such a solution is to solve the QP with a horizon of $N-1$ then use the optimisation variables to warm start another optimisation with a horizon of N. If that is infeasible, the shorter horizon solution is used. In that case, it is necessary to start the optimisation with a horizon of $N-2$ at the next step, with the option to build back up to N.

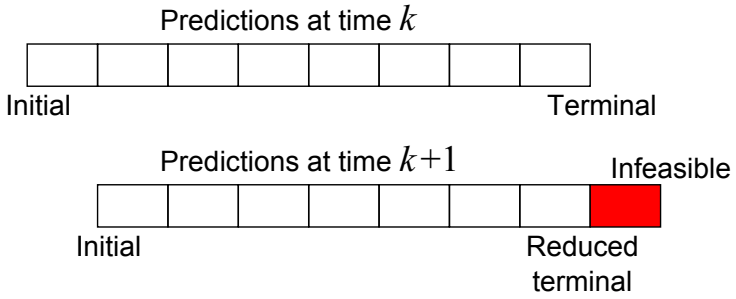

Figure 3.4: Schematic showing the technique of reducing the length of the prediction horizon to avoid infeasibility due to non-recursively feasible terminal constraint. The terminal constraint is applied one prediction step earlier, which is necessarily feasible because it was feasible on the previous time step.

This could in principle continue until the horizon vanishes and the autonomous controller takes over. However, due to the extra measurements that will have been taken in the meantime, the uncertainty in the horizon will be reduced compared to the original solution at time k. Thus in practical terms, the shortening of the horizon would only happen rarely.

2. Construct a terminal set in the appropriate space

This solution only requires one QP per time step. A robustly invariant terminal set is constructed in (z, α)-space. This set is defined in terms of z and α where, under the autonomous dynamics for z and some suitable dynamics for α, membership of the set at one time step implies membership for the subsequent time step. The dynamics for α dictated by this framework are:

$$\alpha_{N|k+1} = \max_j \left(H^{(j)} \alpha_{N|k} + V \tilde{\Delta}^{(j)} z_{N|k} \right). \tag{3.18}$$

The main challenge in finding such a set is that finding a terminal set in $n_x + n_V$ dimensions is very computationally demanding. Recent progress in this area is given in [Fleming et al., 2013], where a secondary horizon is employed with autonomous dynamics for α, which in turn has a terminal constraint that is recursively feasible. That terminal constraint is defined without the need for a terminal set in a high-dimensional space. However, there are many online optimisation

variables introduced.

3. An a posteriori test on the terminal set

This solution employs the original multiplicative RMPC algorithm (Algorithm 3.2) and tests whether it is always possible to find $\alpha_{N|k+1}$ for every feasible pair $(z_{N|k}, \alpha_{N|k})$.

A pair $(z_{N|k}, \alpha_{N|k})$ is feasible if:

$$\{z_{N|k} + e : Ve \leq \alpha_{N|k}\} \subseteq \mathbb{X}_f, \tag{3.19}$$

which is implied by (for fixed H_f) the following condition:

$$H_f \alpha_{N|k} + H_t z_{N|k} \leq \underline{1}, \quad \alpha_{N|k} \geq 0. \tag{3.20}$$

The set of all (z, α) that satisfty (3.20) is a polytope and it is possible (though computationally demanding) to find its vertices, here denoted $(z_\infty^{(i)}, \alpha_\infty^{(i)})$.

We now select the candidate successors $z^*_{N|k+1} = \Phi z_{N|k}$ and

$$\alpha^*_{N|k+1} = \max_j \{H^{(j)} \alpha_{N|k} + V\Delta^{(j)} z_{N|k}\}. \tag{3.21}$$

If it is possible to find any element of $H_t x^*_{N|k+1}$ greater than unity such that:

$$x^*_{N|k+1} = \Phi z_\infty^{(i)} + e \;\text{ s.t. }\; Ve \leq \max_j \{H^{(j)} \alpha_\infty^{(i)} + V\Delta^{(j)} z_\infty^{(i)}\} \tag{3.22}$$

for any vertex pair $(z_\infty^{(i)}, \alpha_\infty^{(i)})$, then the set $\mathbb{X}_f$ does not lead to a recursively feasible RMPC algorithm and the test returns *false*. If it is not possible to satisfy (3.22), then the test returns *true*.

Theorem 6. *Controller $\mathcal{K}_5$ is recursively feasible if the terminal set passes the a posteriori test.*

Proof. For this proof, we choose candidate optimisation variables $c^*_{i|k+1} = c_{i+1|k}$ and $\alpha^*_{i|k+1} = \alpha_{i+1|k}$ for $i < N-1$, $c^*_{N-1|k+1} = 0$ and

$$e^*_{0|k+1} = \Delta_k z_{0|k} + (\Phi + \Delta_k) e_{0|k}.$$

Constraints for prediction steps $0, \ldots, N-2$ are feasible by replacement of variables. Predicted state $x^*_{N-1|k+1}$ is in a subset of the terminal set. Since $c^*_{N-1|k+1} = 0$, the mixed constraints are satisfied for that prediction step by construction of the terminal set. The a posteriori test, if passed, ensures by convexity that any pair (z, α) for which

$$\{z + e : Ve \leq \alpha\} \subseteq \mathbb{X}_f$$

has a valid successor for α, with the successor for z being Φz. Since $(z_{N|k}, \alpha_{N|k})$ satisfies that condition, it is possible to find $\alpha^*_{N|k+1}$. This completes the proof as all constraints have been satisfied by candidate optimisation variables.

□

3.3 An Overview of Sampling Techniques

Controller $\mathcal{K}_5$ steers the state into the terminal set, while respecting the state and input constraints for any scenario constituting a possible sequence of uncertain parameters. The controller does not require a known uncertainty distribution, only that the distribution is finitely supported by known vertices. Some probability distributions have the majority of their mass in the interior away from the vertices, such as the triangle distribution. If we allow a probability of constraint violation less than unity, control of such systems may benefit from softer constraints. MPC schemes that exploit such probabilistic constraints are termed 'Stochastic MPC' (SMPC).

The challenge in SMPC is to calculate online how much the constraints can be relaxed while restricting the probability of constraint violation to be no greater than a specified value. Additionally, since the probability distribution is known, it is possible to define a cost based on the expectation of the cost rather than a cost based on the nominal dynamics. This form of cost function explicitly accounts for the variance of the predicted states in addition to the mean.

This thesis focuses on the handling of multiplicative uncertainty since the additive case has been studied extensively and it is straightforward to adapt the controller presented in this section to handle additive uncertainty, as in Chapter 5.

It is clear from (3.2c) that the uncertain prediction $e_{i|k}$ contains a product of the form $\left(\prod_{j=0}^{i-1} \Delta_{k+j}\right) z_{0|k}$. Calculating the probability distribution of such a term requires multidimensional integration, which would be unacceptably computationally costly to perform online. Performing this calculation offline is also not an option, as $z_{0|k}$ is not known offline.

For the above reasons, we approximate the uncertainty distribution by sampling as in [Batina, 2004], in which an MPC algorithm is developed that applies constraints on samples of the probability distribution. The confidence with which the solution to the optimisation problem satisfies the underlying probabilistic constraints is studied, and is shown to increase with the number of samples. The work is limited to additive uncertainty.

There are two principal options for using sampling in Stochastic MPC. These are introduced as follows:

- One-step sampling, where n_s samples are drawn from $\mathbb{D}$ for each time step independently. A given probabilistic statement applies only to one time step. There are Nn_s samples in total.

- Scenario sampling, where each sample represents one sequence of realisations of the uncertainty over the finite horizon. Probabilistic statements apply over the whole horizon. There are n'_s samples in total.

The two methods generate different forms of probabilistic statement. One-step sampling approximates the probability of an outcome of the evolution over a single time step of a state that is known with certainty to satisfy the constraints. It is similar to a particle filter, often used in signal processing and robotics, e.g. [Blackmore et al., 2010]. In scenario sampling on the other hand, constraint satisfaction for a subset of ν'_s samples implies a probability of constraint satisfaction of the evolution of a measured state over a finite horizon.

The accuracy of approximation of probabilities in both methods depends on the number of samples employed. Confidence levels in approximate solutions to robust convex programs are the focus of [Calafiore et al., 2003]. A more formal definition of the scenario approach to sampling is given in [Calafiore and Campi, 2006]. However, this work is very conservative with respect to the realised rate

of constraint satisfaction in a closed loop system. This is due to the method employed, by which constraints are formed from all the samples taken from the uncertainty distribution. If enough samples are created then a given confidence bound is achieved. An improved control scheme is given in [Campi and Garatti, 2011], being the first to discard samples in order to bring the conservatism down. Section 3.4 will show that the techniques given [Campi and Garatti, 2011] are applicable to the one-step method, which is employed throughout this thesis.

In summary, one-step sampling allows us to guarantee recursive feasibility if the probability distribution is finitely supported, while scenario sampling allows for a different kind of probabilistic statement, but does not guarantee recursive feasibility.

3.4 Multiplicative Stochastic MPC

3.4.1 Introduction

This section introduces some notation. A probability distribution function over an uncertain parameter $a \in \mathbb{A}$ is denoted $f(a)$. The probability that a single realisation of that uncertain parameter is in some subset $\mathbb{A}' \subseteq \mathbb{A}$ is defined:

$$\Pr(a \in \mathbb{A}') = \int_{\mathbb{A}'} p(a)\ da. \tag{3.23}$$

The model in Section 3.2 comprised multiplicative uncertainty in both the system gain $(A + \Delta_k)$ and the input gain $(B + \Gamma_k)$. It was shown that both these sources of uncertainty can be handled in the same manner. This section defines $\Gamma_k = 0$ for simplicity of presentation. We proceed to define a new model for which the probability distribution of the uncertain parameters is known and is finitely supported:

$$\mathcal{G}_4 \ : \ x_{k+1} = (A + \Delta_k)x_k + Bu_k \tag{3.24a}$$

$$\mathbb{D}_c := \text{co}\left\{\Delta^{(j)} : j = 1, \ldots, \rho\right\} \tag{3.24b}$$

$$\Pr(\Delta_k \in \mathbb{D}_c) = 1. \tag{3.24c}$$

It is convenient to have the expectation $\mathrm{E}_k(\Delta_k)$ of future uncertain parameters

equal to zero. This can be achieved without loss of generality since one can define a new model

$$x_{k+1} = \big[(A + \mathrm{E}_k(\Delta_k)) + (\Delta_k - \mathrm{E}_k(\Delta_k))\big]x_k + Bu_k$$

that has the same form as (3.24) and has zero mean uncertainty. Here, E_k is the expectation operator that confers the expectation given knowledge at time step k.The expectation of past uncertain parameters is not necessarily zero because while operating online, it may be possible to deduce some information about past disturbances from measurement data.

We denote the probability of satisfaction of constraints at time step $k+i$, given measurements up to time step k as:

$$p_{i|k} := \Pr\left(Fx_{i|k} + Gu_{i|k} \leq \underline{1}\right). \tag{3.25}$$

Here $p_{i|k}$ is consistent with the notation for predicted inputs $u_{i|k}$ and predicted states $x_{i|k}$ rather than with the notation for expectation $\mathrm{E}_k(\cdot)$ since $p_{i|k}$ refers to a prediction made at time k rather than a probability conditional on information at time k. Note that $p_{i|k}$ is not known precisely for $i > 0$. The remainder of this section discusses ways to approximate (3.25) and what impact those approximations have on the SMPC algorithm.

3.4.2 Probabilistic constraint enforcement

The purpose of SMPC is to attempt to enforce $p_{i|k} \geq p$ for a given constant $p \in \mathbb{R}^{n_F}$. For this condition to be recursively feasible, we must consider the worst-case disturbances for the previous time steps in the prediction horizon. In other words, any solution that satisfies the constraint $p_{i+1|k} \geq p$ must also satisfy the constraint $p_{i|k+1} \geq p$ regardless of Δ_k. For this reason, we maintain the robust polytopic tube from Section 3.2 and define one step ahead probabilities:

$$\begin{aligned} p_{i+1|k} &= \Pr\left(\tilde{F}(z_{i+1|k} + e_{i+1|k}) + Gc_{i+1|k} \leq \underline{1}\right) \\ &= \Pr\left(\tilde{F}z_{i+1|k} + \tilde{F}\left((\Phi + \Delta_{k+i})e_{i|k} + \Delta_{k+i}z_{i|k}\right) + Gc_{i+1|k} \leq \underline{1}\right) \end{aligned} \tag{3.26}$$

The probability distribution over $e_{i+1|k}$ is non-convex, which would lead to a computationally intractable problem. Sampling the uncertainty distribution avoids this problem, as shall be explained below. We sample Δ_{k+i} according to its probability distribution, producing the set:

$$\mathbb{D} = \left\{ \Delta^{\{j\}} : j = 1, \ldots, n_s \right\}, \tag{3.27}$$

noting that the samples are independent of time. To avoid the burden of extra notation to identify individual rows of the constraints matrices and elements of p, we apply only one probabilistic constraint, meaning p is a scalar. The techniques in this thesis do not require this however.

We replace (3.26) with n_s constraints in terms of the sampled parameters, not all of which are enforced online. A prescribed fraction of the sampled constraints are enforced, while the rest are discarded. Whether each sampled constraint is enforced is denoted by the binary variable $b_{i|k}^{\{j\}}$. For $b_{i|k}^{\{j\}} = 1$,

$$\tilde{F} z_{i+1|k} + \tilde{F} \left((\Phi + \Delta^{\{j\}}) e_{i|k} + \Delta^{\{j\}} z_{i|k} \right) + G c_{i+1|k} \leq 1, \tag{3.28}$$

while for $b_{i|k}^{\{j\}} = 0$, the corresponding sampled constraint is ignored.

To impose (3.28) on an online optimisation, the constraints require restating in terms of optimisation variables, a technique demonstrated in (3.13). For $j = 1, \ldots, n_s$, compute offline the matrix $H^{\{j\}}$ of non-negative elements with the lowest element sum such that:

$$H^{\{j\}} V = \tilde{F}(\Phi + \Delta^{\{j\}}) \tag{3.29}$$

and re-define the sampled constraint as follows:

$$\tilde{F} z_{i+1|k} + H^{\{j\}} \alpha_{i|k} + \tilde{F}(\Delta^{\{j\}} z_{i|k}) + G c_{i+1|k} \leq 1, \tag{3.30}$$

which is enforced online if $b_{i|k}^{\{j\}} = 1$.

3.4.3 Confidence and conservativeness

The SMPC algorithm now imposes a new constraint on the number of these sampled constraints that are active. This number must be large enough such that

the underlying probability of constraint satisfaction is more than p with a sufficient confidence. The number of sampled constraints that are enforced is expressed in terms of an 'assumed probability', $\hat{p}$, such that

$$\lceil \hat{p} n_s \rceil = \sum_{j=1}^{n_s} b_{i|k}^{\{j\}}, \qquad i = 0, \ldots, N-1, \tag{3.31}$$

where $\lceil a \rceil$ is the 'rounding up' or 'ceiling' operation, returning the smallest integer no less than a.

The converse, the number of constraints that are discarded per prediction step, is denoted κ as follows:

$$\kappa := \left| \{ j : b_{i|k}^{\{j\}} = 0 \} \right| = \lfloor (1 - \hat{p}) n_s \rfloor, \tag{3.32}$$

where $|\mathbb{A}|$ denotes the cardinality of (the number of elements in) set $\mathbb{A}$ and where $\lfloor a \rfloor$ is the 'rounding down' or 'floor' operation.

When (3.30) is imposed on the QP, with values of $b_{i|k}^{\{j\}}$ that satisfy (3.31), the resulting problem becomes a mixed integer quadratic program (MIQP). A discussion on the solution of such problems is found later in this chapter. Once the MIQP has been solved, there is a calculable risk, β, that the solution leads to a probability of constraint satisfaction of less than p. The term 'confidence' is defined as $1 - \beta$, where

$$1 - \beta = \Pr(p_{i+1|k} \geq p) \tag{3.33}$$

How large the number of samples needs to be for an acceptable confidence level, applied in a general optimisation context, is examined in [Calafiore and Campi, 2006]. Applied to the present control problem, this can be presented as follows. The probability of satisfaction of a constraint one step ahead is $p_{i+1|k}$ as in (3.26). This is imposed by constraining the QP by all the sampled constraints for which $b_{i|k}^{\{j\}} = 1$. We wish to know the probability that we have succeeded in finding a solution that satifies the mixed constraints with probability p. This confidence value depends on our choice of the number of samples and the assumed probability. The confidence bounds derived in [Campi and Garatti, 2011] are adapted for the

problem considered here to give:

$$1 - \beta \geq \sum_{i=0}^{\kappa+d-1} \left(\begin{array}{c} n_s \\ i \end{array} \right) (1-p)^i p^{n_s-1}, \tag{3.34}$$

where d is the number of independent decision variables. Since the probabilistic statements apply to only one time step in the present optimisation problem, $d = n_u$, the number of inputs to the model.

Confidence is increased by increasing the assumed probability, i.e. discarding fewer samples. The upper plot of Figure 3.5 shows how confidence varies with assumed probability $\hat{p}$ for three choices of n_s. Note that κ holding integer values.

Recall that confidence is $\Pr(p_{i+1|k} \geq p)$ for a given choice of n_s and $\hat{p}$. But how conservative is that choice? To prevent a controller becoming overly conservative, it may be helpful to calculate what the probability is that the probability of satisfaction exceeds some upper value $\bar{p} > p$. We define 'conservativeness' as:

$$\bar{\beta} := \Pr(p_{i+1|k} \geq \bar{p}) \tag{3.35}$$

for a given choice of n_s and $\hat{p}$. The purpose of calculating $\bar{\beta}$ is that the probability of satisfaction of constraints is uncertain, and is more likely to exceed $\bar{p}$ for a smaller number of samples. One reason to use SMPC instead of RMPC is to reduce conservativeness (RMPC has a conservativeness of unity), so (3.35) can be helpful to choose n_s and $\hat{p}$.

The lower plot of Figure 3.5 illustrates (for $p = 0.6$) the conservativeness for three values of n_s, each with a different assumed probability $\hat{p}$. The assumed probability for each choice of n_s is chosen such that the confidence (read from the upper plot) is 0.9.

For example, with $n_s = 1000$ and $\hat{p} = 0.62$, we have confidence 0.9 that the probability of satisfaction is no less than 0.6. Considering the upper bound, i.e. conservativeness, there is probability 0.1 that the probability of satisfaction will be greater than 0.64. With $n_s = 25$ in order to maintain confidence $\hat{p}$ must be higher at 0.68. Now there is probability 0.1 that the probability of satisfaction is as high as 0.84. This example illustrates the benefit of employing more samples to reduce conservativeness.

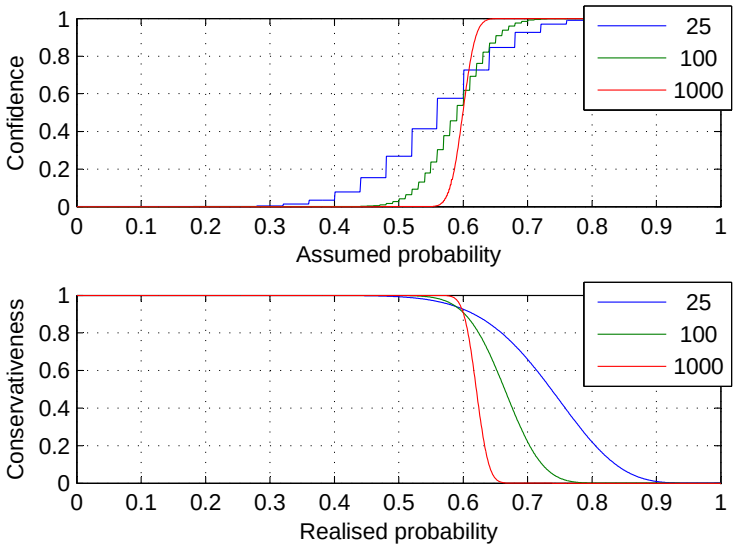

Figure 3.5: A comparison, for three choices of n_s, when $p = 0.6$. (Upper) Confidence in satisfaction of the probabilistic constraint versus assumed probability. (Lower) Conservativeness of solution for the specific choice of assumed probability that gives 0.9 confidence.

Remark 1. *To summarise, confidence is the probability that the probabilistic constraint will be satisfied by the chosen number of samples and proportion to discard. Conservativeness, for a chosen number of samples and proportion to discard, is the probability that the realised probability of violation is greater than some given value.*

3.4.4 Solution of mixed integer QPs

Exact solutions to the MIQP class of problem require a branch-and-bound solver, which are computationally demanding for large n_s. Instead we simply solve the robust QP, select binary values for $b_{i|k}^{\{j\}}$ according to a specified criterion, then solve a new QP with these constraints in place of (3.13b). This is a greedy algorithm approach. It has the disadvantage that it does not increase the region of attraction over that of RMPC but maintains the ability of SMPC to lower the cost relative to RMPC while retaining the latter's guarantee of recursive feasibility.

We impose as constraints the $(n_s - \kappa)$ samples with the highest 'degree of

satisfaction' after having solved the robust problem. Equivalently, the κ samples with the lowest degree of satisfaction are discarded. Each sample has a degree of satisfaction (DoS) defined as:

$$s_{i|k}^{\{j\}} := 1 - \left(\tilde{F} z_{i|k} + H^{\{j\}} \alpha_{i-1|k} + \tilde{F}(\Delta^{\{j\}} z_{i-1|k}) + G c_{i|k} \right). \tag{3.36}$$

To find the lowest κ samples to discard, ranked by DoS, there is no need to sort the set $\{s_{i|k}^{\{j\}} : j = 1, \ldots, n_s\}$, which would have computational complexity $O(n_s \log n_s)$. We need only to know which sample has κ samples with a lower DoS, i.e. find the (κ/n_s) percentile, and denote its DoS as $s_{i|k}^*$. For example, if $\hat{p} = 0.5$, we need to find the median sample. Finding the sample that separates the set of samples into those that have higher DoS and those having lower DoS can be achieved in $O(n_s)$ time. Then, we simply assign the rule:

$$b_{i|k}^{\{j\}} = \begin{cases} 1 & \text{if} s_{i|k}^{\{j\}} \geq s_{i|k}^* \\ 0 & \text{otherwise.} \end{cases} \tag{3.37}$$

Figure 3.6 illustrates the process of solving the robust problem, calculating the degree of satisfaction for each sampled constraint, selecting the critical sample such that κ sampled constraints have a lower degree of satisfaction than it, and ignoring those while enforcing those with a higher degree of satisfaction.

If there are many probabilistic constraints, it may be beneficial to re-evaluate every $s_{i|k}^{\{j\}}$ and $s_{i|k}^*$ and repeat the assignments in (3.37). This can lead to a more optimal solution at little extra computational effort if the solver accepts the sub-optimal decision variables as a starting point, known as a 'warm start'. The number of repetitions of these steps shall be denoted n_r.

3.4.5 Cost function

Since the probability distribution is known, the predicted cost can be written as an expectation, that is the stage cost is the expectation of a quadratic function rather than a quadratic function of an expected (nominal) predictions as in (3.15). The benefit of a cost in terms of an expectation of a quadratic function of uncertain predictions is that the variance of those predictions is penalised as well as the

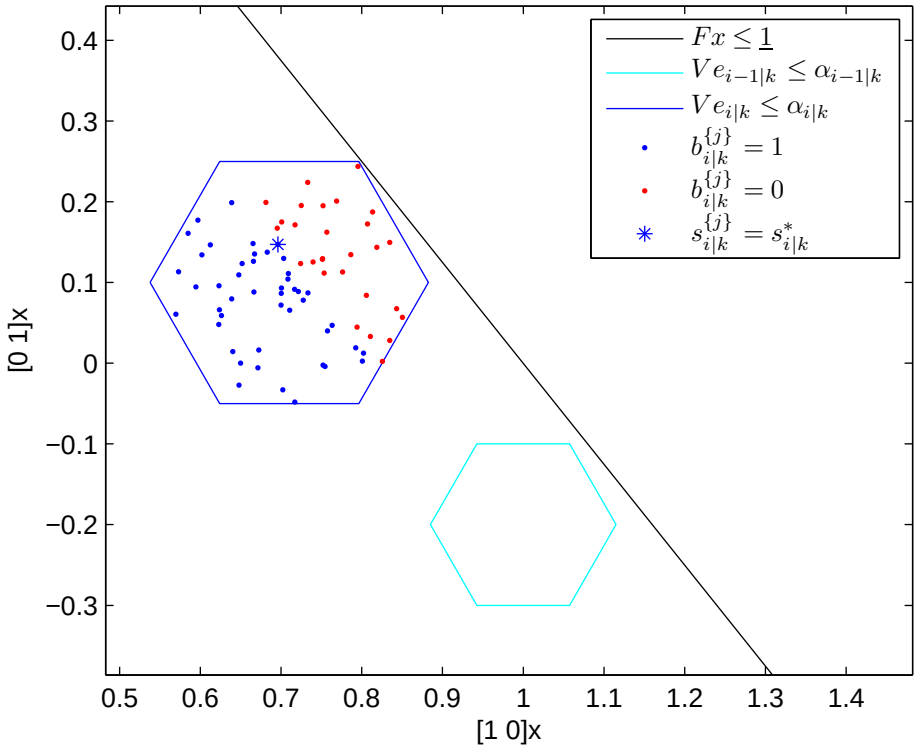

Figure 3.6: An illustration of the calculation of $s_{i|k}^{\{j\}}$, the selection of the critical sample (shown with an asterisk) and the assignment of $b_{i|k}^{\{j\}}$. In this illustration, $n_s = 70$ and $\hat{p} = 0.62$. Also shown is a constraint $Fx \leq \underline{1}$ and robust polytopic uncertainty bounding sets for time steps $k+i$ and $k+i-1$.

mean of the predictions. The state costs are summed over the infinite horizon to give the predicted cost at time step k as follows:

$$J_k = \sum_{i=0}^{\infty} \mathrm{E}_k(x_{i|k}^T Q x_{i|k} + u_{i|k}^T R u_{i|k}). \tag{3.38}$$

It will be proved in Theorem 7 that the cost (3.38) is quadratic in the optimisation variables, as given in [Cannon et al., 2009]. To show this, it is convenient to define a lifted state space representation of the predicted states and inputs:

$$\chi_{i+1|k} = \Psi_{k+i} \chi_{i|k}, \tag{3.39a}$$

$$\chi_{i|k} = \begin{bmatrix} z_{i|k} \\ \mathbf{c}_{k+i} \\ e_{i|k} \end{bmatrix}, \quad \Psi_{k+i} = \begin{bmatrix} \Phi & BE & 0 \\ 0 & M & 0 \\ \Delta_{k+i} & 0 & \Phi + \Delta_{k+i} \end{bmatrix} \tag{3.39b}$$

where M is defined as in (3.17) and E is a block row vector consisting of the identity matrix followed by zero blocks such that:

$$E\mathbf{c}_k = c_{0|k}. \tag{3.40}$$

Then, the stage cost of (3.38) can be written $\mathrm{E}_k(\chi_{i|k}^T \bar{Q} \chi_{i|k})$, with

$$\bar{Q} = \begin{bmatrix} Q + K^T R K & K^T R E & Q + K^T R K \\ * & E^T R E & E^T R K \\ * & * & Q + K^T R K \end{bmatrix}. \tag{3.41}$$

To evaluate the predicted cost, (3.38) is now stated as $J_k = \chi_{0|k}^T P \chi_{0|k}$, where positive symmetric P is calculated such that such that:

$$P - \mathrm{E}_k(\Psi_{k+i}^T P \Psi_{k+i}) = \bar{Q}, \qquad i \in \mathbb{Z}. \tag{3.42}$$

Theorem 7. *The cost* (3.38) *for the prediction dynamics of* (3.39) *is quadratic in* $(\mathbf{c}_k, e_{0|k})$ *and is given by* $J_k = \chi_{0|k}^T P \chi_{0|k}$.

Proof. Let $V_{i|k} := \chi_{i|k}^T P \chi_{i|k}$, so

$$\mathrm{E}_k(V_{i|k}) - \mathrm{E}_k(V_{i+1|k}) = \mathrm{E}_k(\chi_{i|k}^T P \chi_{i|k}) - \mathrm{E}_k\left(\chi_{i|k}^T \Psi_{k+i}^T P \Psi_{k+i} \chi_{i|k}\right).$$

Expectation is a linear operator and $\chi_{i|k}$ is independent of Ψ_{k+i} so

$$\begin{aligned} \mathrm{E}_k(V_{i|k}) - \mathrm{E}_k(V_{i+1|k}) &= \mathrm{E}_k\left(\chi_{i|k}^T (P - \mathrm{E}_k(\Psi_{k+i}^T P \Psi_{k+i})) \chi_{i|k}\right) \\ &= \mathrm{E}_k(\chi_{i|k}^T \bar{Q} \chi_{i|k}) \\ &= \mathrm{E}_k(x_{i|k}^T Q x_{i|k} + u_{i|k}^T R u_{i|k}). \end{aligned}$$

Summing these differences over $i = 0, 1, \ldots$ yields

$$V_{0|k} = \mathrm{E}_k(V_{0|k}) = \sum_{i=0}^{\infty} \mathrm{E}_k(x_{i|k}^T Q x_{i|k} + u_{i|k}^T R u_{i|k}), \tag{3.43}$$

and it follows that $V_{0|k} = J_k$. □

Since $\Delta_k \in \mathbb{D}_c$, it can be written as a weighted sum of the vertices of $\mathbb{D}_c$ as follows:

$$\Delta_k = \sum_{j=1}^{\rho} \lambda_k^{(j)} \Delta^{(j)}, \qquad \sum_{j=1}^{\rho} \lambda_k^{(j)} = 1. \tag{3.44}$$

Likewise $\Psi_k = \mathrm{E}_k(\Psi_k) + \sum_{j=1}^{\rho} \lambda_k^{(j)} \Psi^{(j)}$, where

$$\Psi^{(j)} := \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \Delta^{(j)} & 0 & \Delta^{(j)} \end{bmatrix}. \tag{3.45}$$

This decomposition of Ψ_k leads to a semi-definite program (SDP) in P as follows. The expectation of the product of two weights can be found from the probability distribution of the uncertainty.

$$P - \mathrm{E}_k(\Psi)^T P \mathrm{E}_k(\Psi) - \sum_{j,l} \Psi^{(j)T} P \Psi^{(l)} \mathrm{E}_k(\lambda_j \lambda_l) = \bar{Q} \tag{3.46}$$

Solving (3.46) for P allows us to state the unconstrained objective function of Stochastic MPC as follows:

$$\min_{\mathbf{c}_k, e_{0|k}} \begin{bmatrix} x_k \\ \mathbf{c}_k \\ e_{0|k} \end{bmatrix}^T P \begin{bmatrix} x_k \\ \mathbf{c}_k \\ e_{0|k} \end{bmatrix}. \tag{3.47}$$

3.4.6 Algorithm

This section gives a summary of the SMPC algorithm, a proof that the algorithm is recursively feasible, and a numerical example. The algorithm first minimises the cost (3.47) subject to the robust constraints. It then repeatedly (n_r times) solves

the same cost subject to sampled constraints, where only $\hat{p}n_s$ of those sampled constraints are enforced. The selection of which sampled constraints to enforce is made by evaluating the degree of satisfaction of the mixed constraint for each sample, denoted $s_{i|k}^{\{j\}}$. A given sampled constraint is enforced by setting its binary variable $b_{i|k}^{\{j\}} = 1$ if $s_{i|k}^{\{j\}} \geq s_{i|k}^{*}$, where $s_{i|k}^{*}$ is chosen such that the number of sampled constraints that is enforced is always $\hat{p}n_s$. By repeatedly solving the QP with sampled constraints, an approximate solution to the MIQP problem is reached.

Algorithm 6 Multiplicative Stochastic MPC

Require: $N \in \mathbb{Z}_+, \quad x_k \in \mathbb{X}_\infty, \quad p \leq \underline{1}, \quad n_r \in \mathbb{Z}_+$
  Solve QP (3.47) s.t. (3.10), (3.12), (3.13b)
  for $m = 1, \ldots, n_r$ **do**
    Evaluate $s_{i|k}^{\{j\}}$ by (3.36) for $j = 1, \ldots, n_s$
    Find $s_{i|k}^{*}$, the $\hat{p}n_s$-th largest value in $\left\{ s_{i|k}^{\{j\}} : j = 1, \ldots, n_s \right\}$
    Evaluate $b_{i|k}^{\{j\}}$ by (3.37) for $j = 1, \ldots, n_s$
    Solve QP (3.47) s.t. (3.10), (3.12), (3.30)
  end for
Implement: $\mathcal{K}_6 \ : \ u_k = Kx_k + c_{0|k}$

Theorem 8. *Controller $\mathcal{K}_6$ in recursively feasible if the terminal set leads to a recursively feasible controller $\mathcal{K}_5$.*

Proof. At each time step, controller $\mathcal{K}_6$ initially solves the same problem as $\mathcal{K}_5$, namely Robust MPC. Therefore if $\mathcal{K}_5$ is recursively feasible, then the RMPC step in $\mathcal{K}_6$ will always be feasible. The candidate optimisation variables for the successor RMPC step are selected from the solution of the RMPC step, not the final value they hold after the MIQP iterations.

The set of optimisation variables constrained by the robust constraints is an improper subset of the set of variables constrained by the sampled constraints. That is to say the sampled constraints are more relaxed than the robust constraints. Therefore any solution to the robust problem satisfies the sampled constrained problem. □

3.4.7 Numerical Example

This subsection illustrates SMPC for a multiplicatively uncertain model, defined:

$$A = \begin{bmatrix} 1 & -0.25 \\ 0.25 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad \rho = 3$$
$$\Delta^{(1)} = \begin{bmatrix} 0 & 0.02 \\ -0.05 & 0 \end{bmatrix} \quad \Delta^{(2)} = \begin{bmatrix} -0.01 & 0 \\ 0.05 & -0.01 \end{bmatrix} \quad \Delta^{(3)} = \begin{bmatrix} 0.01 & -0.02 \\ 0 & 0.01 \end{bmatrix} \tag{3.48}$$

The probability distribution for the uncertain dynamics is uniform in the convex hull of the vertices $\Delta^{(1)}$, $\Delta^{(2)}$ and $\Delta^{(3)}$. The constraints and cost are defined by:

$$G = \begin{bmatrix} 0 \\ 7 \\ -7 \end{bmatrix}, \quad F = \begin{bmatrix} -1 & 5 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 4 \tag{3.49}$$

The state constraint is treated probabilistically and the two input constraints are treated as 'hard', i.e. robustly. The corresponding probability bound is $p = [0.6 \quad 1 \quad 1]^T$. The choice $n_s = 64$ and $\hat{p} = 0.63$ gives a confidence of 0.8 that the realised probability of satisfaction is more than 0.6 and conservativeness probability of 0.25 that the realised probability of satisfaction is more than 0.7. Figure 4.2 shows fifty realisations over four time steps of the controlled system from one initial condition. Another trial of 5000 realisations resulted in an average constraint violation for time step 2 of 0.404.

3.5 Conclusions

Modelling a system with uncertainty for linear MPC brings benefits in closed-loop but requires specialised techniques to handle the uncertain predictions. Most prior work restricts this uncertainty to additive disturbances because multiplicative uncertainty introduces terms into the predictions that depend on previous instances of the uncertainty and also on the predicted input sequence. If multiplicative uncertainty is bounded by a polytope, then the uncertain predicted states lie in a polytope with a complexity that grows exponentially with the horizon length.

Drawing on prior work where uncertainty in the initial conditions propagates

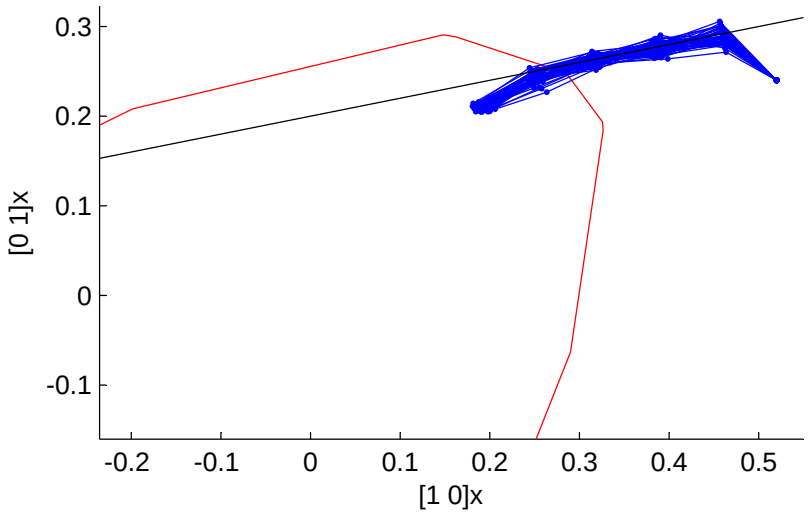

Figure 3.7: State constraint (black), terminal set (red), and fifty realisations of the state under SMPC over four steps (blue). The horizon length $N = 5$.

via a deterministic model through the prediction horizon and is bounded by polytopes, this chapter uses an extension to Farkas's lemma to define a polytopic tube that bounds the state and input uncertainty. Any number of facets can be defined to bound the uncertainty polytopes – more may reduce conservativeness in some applications, while fewer may speed up the online solution to the resulting quadratic program.

The complication in ensuring recursive feasibility of the terminal constraint is a shortcoming of this work. The solution to the QP at time k ensures that all predicted states N steps ahead lie in the terminal set. Indeed, if the whole sequence of predicted inputs were applied to the system rather than just the first one, the state at time $k+N$ would lie in the terminal set. But recursive feasibility requires that the *problem* can be solved one time step later, which requires a modification to the algorithm. Three such modifications are suggested in this chapter. This issue motivates the work of Chapter 5, where there is no finite horizon. There, the control degrees of freedom act over the infinite horizon and the terminal set is replaced by a feasible set in an augmented state space.

A stochastic extension to RMPC is presented that uses one-step-ahead sampling to allow constraints to be violated with a given probability. A predefined number

of those samples is discarded according to a greedy algorithm, which is refined over a few iterations to approximate the solution of the mixed integer problem that arises.

Since representing the probability with samples is an approximation, the confidence with which the probabilistic statements hold is given. It is shown that using more samples in total and discarding fewer samples both increase the confidence level. The benefit of using more samples is shown by deriving a measure of conservativeness of the solution. This analysis allows the SMPC algorithm to be tailored to an application, with significant considerations such as confidence, conservativeness and computational complexity all addressed.

Chapter 4

Stochastic Terminal Set Construction

4.1 Introduction

Section 2.6 showed how to find the maximum volume ellipsoidal set such that every element in the set satisfies the mixed constraints and such that the set is positively invariant under deterministic dynamics. Section 2.7 showed the same for the maximal invariant polytopic set, which, for a discrete system, is the maximal invariant set of any form.

This chapter considers models with multiplicative uncertainty. This section shows how to find the maximum volume ellipsoid and polytope such that every element satisfies the mixed constraints and such that the set is positively invariant under uncertain dynamics.

Section 4.2 presents a novel algorithm for finding a polytopic set that is robustly positively invariant and satisfies the probabilistic constraints in a one-step-ahead manner. The algorithm grows the set one vertex at a time, while at each step ensuring invariance and satisfaction of the probabilistic constraints. We prove that the probability of satisfaction of those constraints over one time step, with the initial state located anywhere inside the terminal set, is no less than the probability of satisfaction with the initial state located at one of the set's vertices.

Firstly, let us consider an ellipsoidal set $\mathbb{E}$ such that:

$$x \in \mathbb{E} \Rightarrow (\Phi + \Delta^{(j)})x \in \mathbb{E}, \quad j = 1, \ldots, \rho. \tag{4.1}$$

The maximum robustly invariant ellipsoid can be found similarly to the deterministic case but with more constraints. This technique is given in [Boyd et al., 1994] and results in the following semi-definite program:

$$\begin{aligned} \max_{S^{-1}} \det S^{-1} \quad \text{s.t.} \quad & \begin{bmatrix} S^{-1} & S^{-1}(\Phi + \Delta^{(j)})^T \\ (\Phi + \Delta^{(j)})S^{-1} & S^{-1} \end{bmatrix} \succ 0, \quad j = 1, \ldots, \rho \\ & f_h^T S^{-1} f_h \leq 1, \quad h = 1, \ldots, n_F, \end{aligned} \tag{4.2}$$

where n_F is the number of rows in the mixed constraint matrices F and G.

The polytopic case is not so straightforward. At each time step, the number of constraints that must be added is ρ-times greater than in the previous step, leading to an exponential growth in the number of constraints in the algorithm. This is avoided in [Pluymers et al., 2005], where a tree structure is formed, exploring only branches that result from non-redundant newly added constraints. It is proven that this finds the maximal robustly invariant set, providing $\max_j \|\Phi + \Delta^{(j)}\| < 1$.

4.2 Probabilistic Algorithm

The Stochastic MPC algorithm in Section 3.4 requires a robustly invariant terminal set. Methods such as the tree structure algorithm given in [Pluymers et al., 2005] are suitable to produce such a set. However, SMPC has constraints that need only be satisfied with a given probability. This motivates finding a set that satisfies the state and input constraints in a one-step-ahead probabilistic manner. Formally we wish to find the largest possible set $\mathbb{X}_f \subseteq \mathbb{X}_f^*$, where:

$$\mathbb{X}_f^* := \left\{ x : (\Phi + \Delta^{(j)})x \in \mathbb{X}_f, \quad j = 1, \ldots, \rho, \quad \Pr(\tilde{F}(\Phi + \Delta)x \leq \underline{1}) \geq p \right\} \tag{4.3}$$

To find such a set, it is convenient to use a vertex representation of the polytope.

The set is defined as the convex hull of its n_v vertices:

$$\mathbb{X}_f = \text{co}\{v_j, \quad j = 1, \ldots, n_v\}. \tag{4.4}$$

For the algorithm in this section, it is required to maintain a dual representation of the set, whereby each facet has an associated set of indices that correspond to vertices in (4.4) that are on that facet. A polytope, with n_f facets is defined by the following inequalities:

$$\mathbb{X}_f = \left\{x : h_k^T x \leq 1, \quad k = 1, \ldots, n_f, \right\}, \tag{4.5}$$

where h_k are column vectors. $\mathbb{X}_f$ has one set of vertex indices per facet k, denoted $\mathbb{V}_k$. The equation that links the facets to the vertices is:

$$\mathbb{V}_k = \{j : h_k^T v_j = 1\}, \quad k = 1, \ldots, n_f. \tag{4.6}$$

An illustration of this dual representation is given in Figure 4.1.

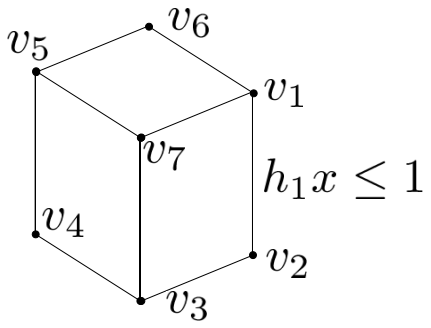

Figure 4.1: A simple polytope consisting of six facets and eight vertices. Facet 1 is defined by the inequality $h_1 x \leq 1$. Vertices 1, 2, 3 and 7 are on Facet 1. Therefore, $\mathbb{V}_1 = \{j : h_1 v_j = 1\} = \{1, 2, 3, 7\}$.

Definition 2. A non-negative function $f(x)$ is log-concave if, for all x and y in a convex set,

$$f(\lambda x + (1 - \lambda)y) \geq f(x)^{\lambda} f(y)^{1-\lambda}$$

for $0 < \lambda < 1$.

Definition 3. A real valued function $f(x)$ is quasiconvex if, for all x and y in a convex set,

$$f(\lambda x + (1-\lambda)y) \leq \max(f(x), f(y))$$

for $0 \leq \lambda \leq 1$. Another way to state this condition is that the maximum value of $f(z)$ over all z in the convex hull of $\{x, y\}$ is either equal to $f(x)$ or equal to $f(y)$. A log-concave function is quasiconcave, which is defined analogously to quasiconvexity.

Theorem 9. *Assume Δ has a continuous log-concave probability density function. Taken over states in $\mathbb{X}_f$, the minimum probability of constraint satisfaction is found at one of its vertices as defined in* (4.4).

Proof. This follows from a proof from [Prékopa, 1995], which establishes the following for state $x \in \mathbb{R}^{n_x}$, variable $y \in \mathbb{R}^{n_q}$ and random vector $\xi \in \mathbb{R}^{n_q}$. If real-valued functions $g_1(x, y), \ldots, g_r(x, y)$ are quasi-convex and ξ has a continuous log-concave probability density function, then $G(x)$ is log-concave, where

$$G(x) := \Pr(g_1(x, \xi) \geq 0, \ldots, g_r(x, \xi) \geq 0).$$

To apply this to the present theorem, consider the vectorised form of the uncertainty $\xi := \text{vec}(\Delta)$, and individual constraint functions $g_h := 1 - \tilde{f}_h^T(\Phi + \Delta)x$, where $\tilde{f}_h^T$ is row h of $\tilde{F}$. Now $g_1, \ldots, g_{n_F}$ are quasi-convex. If Δ has a continuous log-concave distribution then so does ξ. Therefore, $G(x)$ is log-concave and so quasi-concave. Therefore, the minimum value of $f(x)$ over all x in the convex hull of the vertices of $\mathbb{X}_f$ is obtained at one of those vertices. □

Let us assume that the probability distribution $f(\Delta_k)$ is independent of time k and can be sampled, meaning it is possible to draw a sample Δ^* such that:

$$\Pr(\Delta^* \in \mathbb{D}_s) = \int_{\mathbb{D}_s} f(\Delta) d\Delta \tag{4.7}$$

for any subset $\mathbb{D}_s \subseteq \mathbb{D}$ and $\mathbb{D}$ such that $\Pr(\Delta \in \mathbb{D}) = 1$.

The number of samples used in the process of finding a stochastic terminal set can be greater than the number of samples used online – computational time

is not critical provided the problem is tractable. The method only approximates the largest volume invariant set subject to probabilistic constraints. The method exploits randomness to find regions of state space that can be included in the set $\mathbb{X}_f$. An outline of the method follows:

1. Initialise $\mathbb{X}_f$ by defining it as a polytope with a set of vertices selected to lie just inside the surface of the maximal robustly invariant ellipsoid $\mathbb{E}$.

2. Find the facets of this set and the vertex index set $\mathbb{V}_k$ for each facet. Arrange the facets in a cyclic queue: a structure that allows the facets to be addressed in turn.

3. For the facet at the front of the queue: randomly choose a candidate vertex $v^* \in \mathbb{R}^{n_x}$ according to a probability distribution centred on the mean location of the vertices associated with the facet. Ensure the candidate vertex is outside $\mathbb{X}_f$.

4. Define the candidate set $\mathbb{X}_f^*$ as the convex hull of $\mathbb{X}_f \cup \{v^*\}$.

5. If $(\Phi + \Delta^{(j)})v^* \in \mathbb{X}_f^*, \;\; j = 1, \ldots, \rho$ and $\Pr(\tilde{F}(\Phi + \Delta)v^* \leq \underline{1}) \geq p$ then replace $\mathbb{X}_f$ with $\mathbb{X}_f^*$, delete redundant facets, vertices and vertex index sets, and add newly created facets to the back of the cyclic queue.

6. If the rate of growth is below some threshold and all the initial vertices have been deleted then stop, otherwise go to 3.

The following subsections give a more detailed description of the steps above.

4.2.1 Initial vertices

A polytope inscribed in an invariant ellipsoid is not necessarily invariant. This is why the termination condition in step 6 is imposed, namely that all the initial vertices must be subsumed into the set. If initial vertices were selected on the surface of the invariant ellipsoid then there would be an initial vertex near to a constraint plane, and if p is large or unity, then it is likely to take an unreasonably long to subsume that vertex. This is why the vertices are initialised to lie just inside the surface of the invariant ellipsoid.

In a state space with more than two dimensions, there is no closed-form way to equally space vertices on the surface of an ellipsoid. Instead, initial vertices are drawn from the normal distribution, normalised to have a distance from the origin of just less than unity, e.g. $r = 0.99$, and transformed into the shape of the invariant ellipsoid $\mathbb{E} = x : x^T S x \leq 1$. Therefore, the initial vertices are defined as:

$$\mathbb{V}^{(0)} := \left\{ v_j = S^{-\frac{1}{2}} p_j, \quad j = 1, \ldots, n_{v,0} \right\} \tag{4.8}$$

where $n_{v,0}$ defines how many initial vertices there are and where p_j are random points normalised such that $p_j^T p_j = r^2$.

4.2.2 Facets and vertex index sets

In general, finding the facets corresponding to a large set of vertices is computationally demanding but in this algorithm, it is an operation that is only executed once. All future facet-vertex operations will be performed only on small sub-sets of the vertices and facets.

The MATLAB function `convhulln` finds the vertex index sets defined in (4.6). Each of these sets corresponds to a facet. To find the normal vector that describes the half-space for that facet, we use the following property of co-planar points. This is an extension to the vector cross-product to any number of dimensions.

$$v_\perp \cdot v_j = v_\perp \cdot v_k, \qquad j, k = 1, \ldots, n_x \tag{4.9a}$$

$$v_\perp = \det \begin{bmatrix} \mathbf{e}_1 & \ldots & \mathbf{e}_{n_x} \\ v_2^T - v_1^T & & \\ & \vdots & \\ v_{n_x}^T - v_1^T & & \end{bmatrix}, \tag{4.9b}$$

where $\mathbf{e}$ are the unit basis vectors.

For a set of co-planar points $\mathbb{V}_k = \{v_1, \ldots, v_{n_x}\}$, the corresponding half-space $\{x : h_k^T x \leq 1\}$ is found by scaling $v_\perp$ as follows:

$$h_k = \frac{v_\perp}{v_\perp \cdot v_1}. \tag{4.10}$$

4.2.3 Candidate vertex

The process of growing the set involves repeatedly choosing a candidate vertex, finding the set formed by including the candidate vertex in the growth set, and testing whether a state that starts at that point has all its successor states inside this larger candidate set and also satisfies the probabilistic constraint. Each facet is selected in turn – the vertices that lie on that facet are known. We identify the selected facet by the index k. Its vertices are given by the members of the set $\mathbb{V}_k$. The candidate vertex is first assigned the mean position $\bar{v}$ of the facet's vertices. It is then disturbed by some Gaussian noise with zero mean and a standard deviation equal to the maximum distance of any facet vertex from the mean point:

$$v' = \mathcal{N}\left(0, \max_j\{\|v_j - \bar{v}\|, \quad v_j \in \mathbb{V}_k\}\right) \tag{4.11}$$

Finally, if the candidate vertex is inside $\mathbb{X}_f$, it is mirrored about the plane $h_k^T x = 1$. This operation can be written symbolically as follows:

$$v^* = \bar{v} + v'\text{sgn}\big(h_k^T(\bar{v} + v' - 1)\big), \tag{4.12}$$

where sgn is the 'sign function' defined, for any real number x, such that $x = \text{sgn}(x) \cdot |x|$.

4.2.4 Candidate set

The set $\mathbb{H}_-^* = \{m : h_m^T v^* \leq 1\}$ contains the indices of the facets that are unaffected by the addition of the candiate vertex v^*. All the other facets are redundant in the candidate polytope. We wish to remove them and replace them with new facets that are found by a convex hull operation on only the vertices associated with the affected facets and the candidate vertex. The set of affected vertices is:

$$\mathbb{V}^* = \{v_j : j \in \mathbb{V}_m, \quad h_m^T v^* > 1\} \tag{4.13}$$

Finding $\text{co}\,(\mathbb{V}^* \cup \{v^*\})$ gives the sets of vertex indices that lie on each new facet. By selecting only facets whose vertex index sets contain the vertex index

for the candidate vertex, we find the facets that must be added to complete the candidate set, namely $\mathbb{H}_+^*$. The candidate set is then defined as:

$$\mathbb{X}_f^* := \{x : h_m^T x \leq 1, \quad m \in (\mathbb{H}_-^* \cup \mathbb{H}_+^*)\} \tag{4.14}$$

By this process, some vertices may be in the growth set from the previous iteration but not included in the candidate set. If this candidate set is accepted, these vertices will be lost into the interior.

4.2.5 Conditional set growth

Checking whether all possible successor states of the candidate vertex lie inside the candidate set is straightforward: $(\Phi + \Delta^{(j)})v^* \in \mathbb{X}_f^*$ as defined in (4.14). If this condition is met, we proceed to check the probabilistic constraint. Otherwise, the candidate vertex is rejected in the manner described below.

Checking the probabilistic contraints is achieved by sampling the uncertainty distribution. A large number n_s of samples of the uncertain matrix are created:

$$\mathbb{D}^* = \left\{\Delta^{\{j\}} : j = 1, \ldots, n_s\right\}. \tag{4.15}$$

A vector of binary indicator variables $b^{\{j\}} \in \mathbb{R}^{n_F}$ is created for each sample, each element of which is set to unity if the corresponding constraint is satisfied under the dynamics $(\Phi + \Delta^{\{j\}})v^*$. We estimate the probability that the successor state to the candidate vertex will satisfy the mixed constraints using the empirical mean of the indicator variables over all samples:

$$\tilde{p} = \frac{1}{n_s} \sum_{j=1}^{n_s} \beta^{\{j\}}. \tag{4.16}$$

If $\tilde{p} \geq \hat{p}$, where $\hat{p}$ is the assumed probability, then the candidate vertex is accepted and the terminal set grows by being replaced by the candidate set. Otherwise, the candidate set is discarded. A discussion on how to select values for the assumed probability and the number of samples is given in Section 3.4.

4.2.6 Termination criteria

The growth set is invariant only if all possible successor states of all its vertices lie inside the set. This was not verified for the initial vertices $\mathbb{V}^{(0)}$. Hence the first termination criterion is that all of the initial vertices have been removed by the growth.

As the set grows, it will become less likely that a suitable candidate vertex will be found for each active facet in the queue. Iterations where the candidate vertex is rejected contribute nothing to the volume of the set. The growth algorithm terminates when a low-pass filtered rate of growth is lower than a given threshold. The volume contribution of iteration i is defined as the difference in volume from the previous growth set $J_i := \text{vol}(\mathbb{V}^{(i)}) - \text{vol}(\mathbb{V}^{(i-1)})$. The filtered rate of growth is defined as follows:

$$J'_i := \mu J_i + (1 - \mu) J'_{i-1}, \tag{4.17}$$

where $0 < \mu \ll 1$ is a filter constant. The purpose of the low pass filter on the rate of growth is to prevent an early termination of the algorithm due to a random selection of an individual candidate vertex that results in a very small volume increase. The second termination condition is then $J'_i < J_c$, where J_c is a given threshold.

It remains to find the volume of the growth set at each iteration. If a polytope in $\mathbb{R}^{n_x}$ is defined as the convex hull of vertices, and for each vertex, a plane can be constructed such that the vertex is in that plane and all the other vertices are strictly one side of the plane, then every facet of the polytope is in-plane with exactly $\mathbb{R}^{n_x}$ such vertices. The vertices in-plane with each facet therefore form a simplex with the origin.

The polytope can be equivalently defined as the union of these simplexes. The volume of a polytope is the sum of the volume of the simplexes. The volume of a simplex defined by the n_x vectors in $\mathbb{V}_k$ and the origin is:

$$\text{vol}(\mathbb{V}_k \cup \{0\}) = \frac{1}{n_x!} \det \begin{bmatrix} 1 & v_1^T \\ \vdots & \vdots \\ 1 & v_{n_x}^T \end{bmatrix} \tag{4.18}$$

It is straightforward to find the volume of the initial set $\mathrm{vol}(\mathbb{V}^{(0)})$ because the vertex sets are known for all the facets. Now, whenever a facet is removed, its corresponding simplex volume is subtracted from the set volume. Whenever a facet is added, its simplex volume is added to the set volume.

4.3 Numerical example

This subsection illustrates the method of contructing and growing a terminal set with probabilistic constraints. The plant model is defined by the same parameters as in Section 3.4.7, namely:

$$\begin{gathered} A = \begin{bmatrix} 1 & -0.25 \\ 0.25 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad \rho = 3 \\ \Delta^{(1)} = \begin{bmatrix} 0 & 0.02 \\ -0.05 & 0 \end{bmatrix} \quad \Delta^{(2)} = \begin{bmatrix} -0.01 & 0 \\ 0.05 & -0.01 \end{bmatrix} \quad \Delta^{(3)} = \begin{bmatrix} 0.01 & -0.02 \\ 0 & 0.01 \end{bmatrix} \end{gathered} \tag{4.19}$$

The probability distribution for the uncertain dynamics is again uniform in the convex hull of the vertices $\Delta^{(1)}$, $\Delta^{(2)}$ and $\Delta^{(3)}$. The constraints and cost are defined by:

$$G = \begin{bmatrix} 0 \\ 7 \\ -7 \end{bmatrix}, \quad F = \begin{bmatrix} -1 & 5 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 4 \tag{4.20}$$

As in Section 3.4.7, the state constraint is treated probabilistically and the two input constraints are treated as 'hard', i.e. robustly. The corresponding probability bound is $p = [0.6 \quad 1 \quad 1]^T$. The number of samples of the uncertain parameters was $n_s = 1000$. Figure 4.2 shows the first fifty successful inclusions of candidate vertices into the terminal set, which grows randomly outwards from an initial ellipsoid. Note that a region of state space that violates the state constraint is included in the final terminal set. Recall that any element in that set evolves under the autonomous uncertain dynamics to a state that satisfies the state constraint with probability at least 0.6.

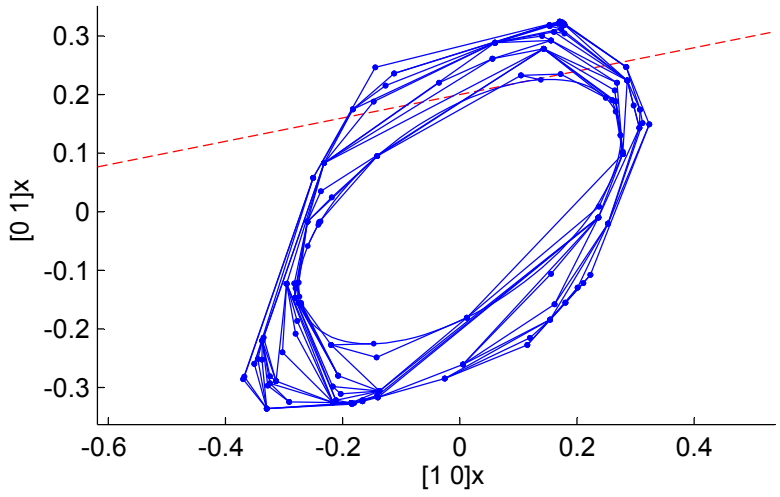

Figure 4.2: State constraint (red dashed), initial vertices inscribed inside the maximal robustly invariant ellipsoid (blue), and fifty steps of the growth algorithm that create successively larger polytopes. Any polytope that contains the initial vertices in its interior is robustly invariant.

4.4 Conclusions

Under dynamics that have multiplicative uncertainty, finding a terminal set to exploit constraints that may be violated with a given probability is non-trivial. This chapter demonstrates how sampling can be used to assess the suitability of individual points in the state space. Any suitable point implies that the region defined by the convex hull of that new point and previously found suitable points is also suitable. An iterative process is defined to find suitable points and add the implied suitable region to a convex set. By this process, the convex set is grown from a given initial set to increase its volume. Once all the vertices of the initial set are subsumed into the growing set, it is proven that the growing set is robustly positively invariant. The process of growing the set allows the volume to be known at all stages. The algorithm terminates when the rate of growth falls below a threshold.

The stochastic terminal set algorithm is a 'Monte Carlo' algorithm, meaning it exploits randomness. This class of algorithm is used where a deterministic solution is computationally intractable. The curse of dimensionality, i.e. searching in high-

dimensional space, would strike when trying to find the optimal location for a candidate vertex. Randomly selecting locations for candidate vertices constitutes a sparse search, similar to the way genetic algorithms work.

Since the growth algorithm is executed offline, the number of samples can be very large. In the numerical example of Section 4.3, the number of samples was chosen to be 1000. Recalling the definition of conservativeness from Section 3.4.3, it is clear that such a large number of samples gives high confidence that the growth algorithm finds candidate vertices that satisfy the probabilistic constraints, while a low probability can be assigned to the chance that the vertices have a probability of satisfaction that is ineffectively high.

The probability of satisfaction of the constraints is only enforced at the vertices. Theorem 9 shows that the probability of satisfaction is lowest at one of the vertices. Therefore verification of the whole set requires only verification at the vertices, which is the manner in which the algorithm grows the set.

Although the successor states of any state that lies in the probabilistic terminal set only satisfies the mixed constraints with a given probability between p and unity, it lies inside the terminal set with certainty. That is, the probabilistic terminal set is robustly positively invariant. The important result of this property for the purposes of the operation of any SMPC algorithm that uses the probabilistic terminal set is that a violation of the mixed constraints at one time step cannot result in a probability of satisfaction at the subsequent time step that is less than p. This is a benefit of one-step sampling over scenario sampling, where the same does not hold – a state that violates the mixed constraint might have any probability of satisfaction at the subsequent time step.

A drawback, in computational terms, of the algorithm presented in this chapter is that it can produce polytopes with many facets and vertices. Each facet of the terminal set adds one constraint to be enforced online. While QP solvers can be optimised to handle many linear constraints computationally efficiently, the complexity of the terminal set increases the computational complexity of the recursive feasibility test, if used. Further research is recommended into the adaptation to SMPC of recent developments in terminal set construction that result in recursively feasible polytopic tube RMPC algorithms.

Chapter 5

Exponential Basis Function MPC

5.1 Motivation

A terminal set, a set in state space that is robustly positively invariant, can be computationally expensive to find. The process is offline but for applications where rapid prototyping of a controller is desired, even offline processes must execute in a reasonable time on a standard desktop computer. Additionally, the terminal constraint (that the furthest predicted set of possible states lies inside the terminal set) restricts the feasible region for fixed length horizons. Schemes where the horizon can change online typically introduce an undesirable online computational burden. It is therefore in the engineer's interest to have a long Mode 1 horizon. However, in most MPC schemes, every time step in Mode 1 contributes one degree of freedom per input. Such a scheme, combined with the polytopic uncertainty tube scheme given in Chapter 3, gives a QP in $(n_u + n_V)N$ variables. Here we seek to find a scheme that uses fewer input degrees of freedom while guaranteeing recursive feasibility for systems with both multiplicative and additive uncertainty.

Additionally, recursive feasibility guarantees given in Section 3.2 for polytopic tubes are computationally expensive and somewhat unnatural in the context of RMPC. This chapter proposes a scheme with arbitrarily small conservativeness and automatic guarantees of recursive feasibility.

5.2 Background

One potential technique to reduce the number of optimisation variables is 'input blocking' where each time step either has its own independent input degree of freedom, or is equal to its predecessor. Shortcomings of some implementations of this are detailed in [Cagienard et al., 2007], along with a suggested work-around. A recursive feasibility problem can arise because after one time step, if the input blocks march forward, they overlap with the blocks from the previous time step. This means one cannot shift the previous optimisation variables by one time step to prove that there exists a feasible solution at the subsequent step. The work-around is for the blocks to not march forward, but for the front block to reduce in length by one at each step, thus requiring a time varying input blocking scheme. These two arrangements are illustrated in Figure 5.1.

The benefits of Exponential Basis Function MPC over Moving Window Blocking MPC are twofold. Firstly, the optimisation variables have an effect over long (theoretically infinite) time scales, while a careful choice of basis function can ensure the optimisation variables also have enough control authority where it matters more, in the shorter term. Secondly, there is no terminal constraint, which can be difficult to enforce with recursive feasibility. Instead, the optimisation variables are constrained to lie inside an invariant feasible set.

Nominal MPC (where there is no model uncertainty or disturbance) can be represented in lifted state space form as follows:

$$\begin{bmatrix} x_{k+1} \\ \mathbf{c}_{k+1} \end{bmatrix} = \begin{bmatrix} \Phi & BE \\ 0 & M \end{bmatrix} \begin{bmatrix} x_k \\ \mathbf{c}_k \end{bmatrix}, \tag{5.1}$$

where $\Phi := A + BK$, M is the zero matrix with $n_u \times n_u$ identity matrices on its super-diagonal blocks and $E := \begin{bmatrix} I & 0 & \cdots & 0 \end{bmatrix}$, so that

$$M\mathbf{c}_k = \begin{bmatrix} c_{1|k}^T & \cdots & c_{N-1|k}^T & 0 \end{bmatrix}^T, \qquad E\mathbf{c}_k = c_{0|k}. \tag{5.2}$$

The region of attraction for MPC with horizon length N, where the Nth state prediction lies in the maximal terminal set, is equal to the projection of the maximal invariant set for the autonomous dynamics in the lifted space of states and

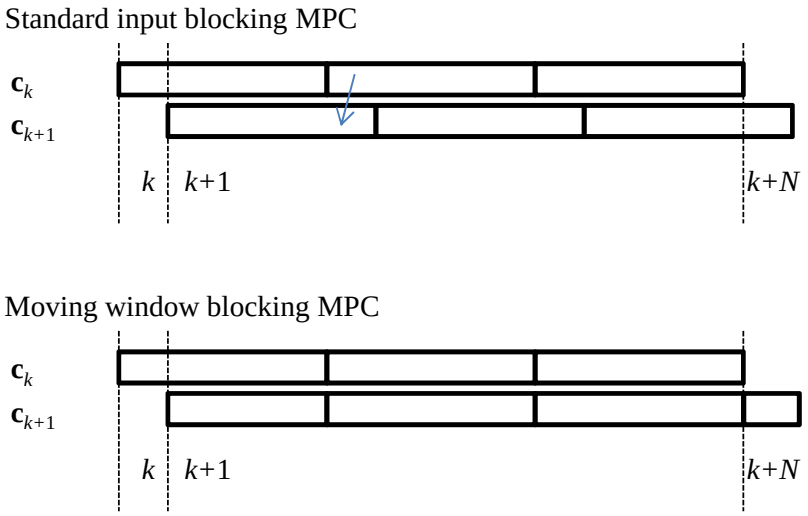

Figure 5.1: Two input blocking methods: standard receding blocks and moving window blocks. In the illustration of the standard blocking method, the arrow highlights why there is no recursive feasibility. In moving window input blocking, the block at the start of the horizon is shortened by one every time step and the block at the end grows by one every step.

input degrees of freedom. This allows one to remove the requirement for a terminal constraint in nominal MPC, replacing it with an invariant feasible set in the lifted space. Such a set can be found in the lifted state space using the technique in [Gilbert and Tan, 1991].

However, it is possible to obtain a larger maximal invariant set with a different choice of autonomous lifted dynamics. It is shown in [Cannon and Kouvaritakis, 2005] that it is possible to optimise these lifted dynamics to increase the size of the projection of the invariant set onto the state space. The optimisation is only tractable for ellipsoidal approximations to the set, but one could use controller dynamics that are optimal for the ellipsoidal set to define the lifted state space dynamics in order to calculate the maximal invariant polytopic set.

The technique of [Cannon and Kouvaritakis, 2005] could be used here to optimise control performance over controller dynamics but this not the focus of this chapter. Instead we show that for a given choice of exponential dynamics, a Robust MPC control scheme can be formed with attractive theoretical and computational

properties. A simple exponential basis function set that has been extensively studied is that of Laguerre functions as in [Wang, 2004; Rossiter and Wang, 2008; Rossiter et al., 2010], work that follows from [Cannon and Kouvaritakis, 2000]. A discussion of different choices for the dynamics of an exponential basis function implementation of MPC for a system with no uncertainty is given in [Khan and Rossiter, 2013].

The maximal robustly positively invariant set is in general hard to find for systems with multiplicative uncertainty because the number of possible extremes for the uncertainty grows exponentially with the number of time steps considered. If the algorithm does not converge quickly, the complexity may grow prohibitively. One possible way round this is given in [Pluymers et al., 2005], where the tree structure formed by successive growth in complexity is pruned at every growth stage to limit the computational demands for the next step of the algorithm. This however, does not prevent the worst-case computational demand from being exponential. The present work uses polytopic tube invariance to maintain a constant complexity in the terminal set algorithm regardless of the number of time steps required to converge.

5.3 Laguerre Functions

Laguerre functions are a sequence of discrete orthonormal basis functions that approach zero asymptotically. This chapter uses a Laguerre parameterisation of input predictions and polytopic tube facets. Each Laguerre function has an initial value and a matrix defining its dynamics with respect to the previous values of itself and lower-order Laguerre functions. A fixed number n_L of these functions is used, which can be increased to improve control performance at increased computational expense. The rate at which Laguerre functions decay is defined by a parameter $0 < \gamma < 1$.

The initial values of the first n_L Laguerre functions is hereby represented by a vector $\ell_0 \in \mathbb{R}^{n_L}$. In the application presented in this thesis, the initial values of the Laguerre functions is not important (providing they are non-zero), since they will be each multiplied by an optimisation variable. Therefore in the sequel, $\ell_0 = \underline{1}$.

The dynamic equation for the Laguerre functions, denoted by the sequence of

vectors ℓ_i, is given by:

$$\ell_{i+1} = A_L \ell_i, \quad i \in \mathbb{Z}, \quad A_L = \begin{bmatrix} \gamma & 0 & 0 & \dots \\ \eta & \gamma & 0 & \dots \\ -\gamma\eta & \eta & \gamma & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \quad \eta = 1 - \gamma^2. \tag{5.3}$$

An example sequence of Laguerre functions is shown in Figure 5.2.

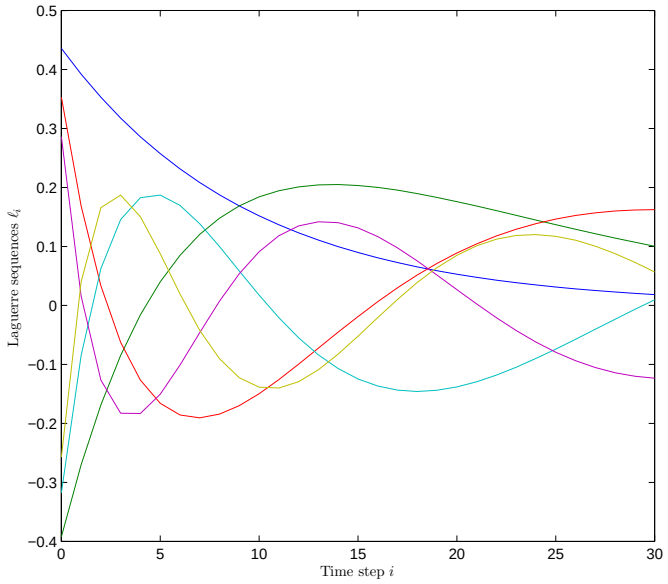

Figure 5.2: The first six Laguerre functions with $\gamma = 0.9$ shown for 30 time steps. The functions have each been multiplied by a scalar for clarity of presentation.

5.4 Problem Definition

The uncertain model $\mathcal{G}_5$ is defined as follows:

$$\mathcal{G}_5 \ : \ x_{k+1} = (A + \Delta_k)x_k + (B + \Gamma_k)u_k + w_k \tag{5.4a}$$

$$\Delta_k \in \mathbb{D} = \text{co}\left\{\Delta^{(j)}, \quad j = 1, \dots, \rho\right\} \tag{5.4b}$$

$$w_k \in \mathbb{W} = \text{co}\left\{w^{(j)}, \quad j = 1, \ldots, q\right\} \tag{5.4c}$$

As in Section 3.4.1, we set $\Gamma_k = 0$ for simplicity of presentation, without loss of generality. The system is subject to mixed constraints:

$$Fx_k + Gu_k \leq g. \tag{5.5}$$

This chapter develops a controller that allows for a time varying target input $\hat{u}_k$ and state $\hat{x}_k$ in any form that can be represented by an exponential basis function. Since these always converge to zero, the cost function to be minimised is finite. It is not necessary for the target functions to evolve according to the same dynamics as the inputs and tube facet definitions. The matrix $U_k \in \mathbb{R}^{n_u \times n_L}$ constitutes weights by which the target input Laguerre functions ℓ'_i are multiplied to give the target input vector for time step $k+i$. The target input Laguerre functions evolve according to the matrix A'_L:

$$\begin{array}{rcl} \hat{u}_k & = & U_k \ell'_0 \\ \hat{u}_{k+i} & = & U_k A'^i_L \ell'_0 = U_k \ell'_i \end{array} \tag{5.6}$$

Equally, the target state can evolve in any manner described by an exponential basis function expansion. However, to simplify presentation, we select target states that form steady-state pairs with the target inputs.

$$\hat{x}_k = \hat{B}\hat{u}_k, \qquad \hat{B} := (I - A)^{-1}B. \tag{5.7}$$

In previous chapters of this thesis, the uncertain predicted states and inputs were decomposed into a certain part and the remaining uncertain part of the prediction. The uncertain part lies in an uncertainty set that contains the origin. When the cost is calculated in expectation, the certain predictions are no longer required. Instead, the uncertainty set can simply contain the uncertain predicted states. However, with both additive and multiplicative uncertainty, it is useful to decompose the predictions once more, but in a different way. In this chapter, the uncertain predictions are decomposed into those that depend on the input degrees of freedom and multiplicative uncertainty, and those that depend on the additive disturbances.

We hereby introduce the decomposed predictions:

$$u_{i|k} = Kz_{i|k} + Le_{i|k} + c_{i|k} \tag{5.8a}$$

$$z_{i+1|k} = (A + BK + \Delta_{k+i})z_{i|k} + Bc_{i|k} \tag{5.8b}$$

$$e_{i+1|k} = (A + BL + \Delta_{k+i})e_{i|k} + w_k, \tag{5.8c}$$

where K and L are feedback gains. The independent inputs $c_{i|k}$ are defined by Laguerre functions, weighted by the optimisation variable matrix $C_k \in \mathbb{R}^{n_u \times n_L}$, as follows:

$$\begin{aligned} c_{0|k} &= C_k \ell_0 \\ c_{i+1|k} &= C_k A_L \ell_i = C_k \ell_{i+1}. \end{aligned} \tag{5.9}$$

The controlled state predictions $z_{i|k}$ are bounded by polytopes with fixed facet normals and variable scaling. These scaling vectors are also expressed in terms of Laguerre functions, in this case weighted by the opimisation variable matrix $M_k \in \mathbb{R}^{n_V \times n_L}$, giving:

$$Vz_{i|k} \leq \alpha_{i|k} = M_k \ell_i. \tag{5.10}$$

The uncontrolled state predictions $e_{i|k}$ are handled offline in the next section.

5.5 Offline Constraint Tightening

The uncontrolled part of the decomposed state predictions evolves according to (5.8c). Finding the effect that these uncertain variables have on the controller is a similar problem to that encountered in additive Robust MPC, for which a solution is given in Section 2.4.

In this case the problem is more complex, due to the multiplicative term $\Delta_{k+i}e_{i|k}$. The impact that these uncertain terms have on the constraints (5.5) is independent of the control degrees of freedom:

$$F(z_{i|k} + e_{i|k}) + G(Kz_{i|k} + Le_{i|k} + c_{i|k}) \leq g \tag{5.11}$$

It is therefore possible to find a constraint tightening vector h_i for each step

ahead, so the constraints can be re-written as follows:

$$Fz_{i|k} + G(Kz_{i|k} + c_{i|k}) \leq g - h_i \tag{5.12a}$$

$$h_i = \max_{e \in \mathbb{H}_i} (F + GL)e \tag{5.12b}$$

$$\mathbb{H}_{i+1} = (\{A + BL\} \oplus \mathbb{D}) \otimes \mathbb{H}_i \oplus \mathbb{W}, \quad \mathbb{H}_0 = \{0\}. \tag{5.12c}$$

The sets $\mathbb{H}_i$ are found sequentially. Initially, $\mathbb{H}_1 = \mathbb{W}$; thereafter,

$$\mathbb{H}_{i+1} = \text{co}\left\{(A + BL + \Delta)e + w \quad : \quad e \in \mathbb{H}_i, \quad \Delta \in \mathbb{D}, \quad w \in \mathbb{W}\right\}. \tag{5.13}$$

As proved in the following section, the number of sets that are required to be calculated offline is finite.

5.6 Feasible Set

This section gives a method for handling the uncertainty in the controlled predictions $z_{i|k}$. From (5.8b) and (5.10), we have that:

$$Vz_{i+1|k} = V(A + BK + \Delta_{k+i})z_{i|k} + VBc_{i|k}. \tag{5.14}$$

To bound the successor uncertain state robustly, we require

$$Vz_{i+1|k} \leq M_k \ell_{i+1}, \quad \Delta_{k+i} \in \mathbb{D}, \tag{5.15}$$

which requires the inequality of (5.15) to hold for Δ_{k+i} equal to any vertex of $\mathbb{D}$, requiring:

$$V(A + BK + \Delta^{(j)})z_{i|k} + VBc_{i|k} \leq M_k \ell_{i+1}, \quad \Delta_{k+i} \in \mathbb{D}, \quad j = 1, \ldots, \rho. \tag{5.16}$$

As in Section 3, we proceed to select offline matrices $H^{(j)}$ with non-negative elements such that $H^{(j)}V = V(A + BK + \Delta^{(j)})$. Now a sufficient condition for

(5.16) is the following, noting that $\ell_{i+1} = A_L \ell_i$,

$$H^{(j)} M_k \ell_i + V B C_k \ell_i \leq M_k A_L \ell_i. \tag{5.17}$$

Similarly, a matrix H_m with non-negative elements is found such that

$$H_m V = F + GK, \tag{5.18}$$

which allows the constraints (5.5) to be expressed as:

$$H_m M_k \ell_i + G C_k \ell_i \leq g - h_i. \tag{5.19}$$

Constraints (5.17) and (5.19) must hold for all future time steps, but control synthesis requires a finite number of constraints. We thus require a set

$$\mathbb{L}_{n^*} \subset \mathbb{R}^{n_V \times n_L} \times \mathbb{R}^{n_u \times n_L}, \tag{5.20}$$

where any element (M_k, C_k) gives state and input predictions that satisfy (5.19) for all $i \in \mathbb{Z}$ and guarantee a feasible selection (M_{k+1}, C_{k+1}) is possible regardless of the values of Δ_k and w_k.

The feasible set can be constructed using a similar procedure as [Gilbert and Tan, 1991], described in Section 2.7. Constraints (5.17) and (5.19) are applied for increasing values of i, consecutively defining sets $\mathbb{L}_m$ as follows:

$$\begin{aligned} \mathbb{L}_m := \big\{ (M, C) \quad : \quad & H^{(j)} M \ell_i + V B C \ell_i \leq M A_L \ell_i, \\ & H_m M \ell_i + G C \ell_i \leq g - h_i, \\ & i = 0, \ldots, m-1, \quad j = 0, \ldots, \rho \big\} \end{aligned} \tag{5.21}$$

Whenever adding a row of (5.19) leaves the set unchanged, that row is ignored for all following steps. Once all rows are redundant for increasing values of i, the algorithm terminates. The resulting constraint set defines the robustly invariant feasible set. To give an indication of the amount of offline computation required by this procedure, in the numerical example of Section 5.8, 75 steps were required for the algorithm to terminate, with the first row becoming redundant after 16 steps.

Theorem 10. *If row r of constraint* (5.19) *is redundant in $\mathbb{L}_{m+1}$ then it will be redundant in $\mathbb{L}_{m+\mu}$ for positive μ.*

Proof. The inequalities in the definition of the set are element-wise, so each row can be treated separately. Therefore

$$\mathbb{L}_m = \bigcap_i^{n_F} \mathbb{L}_m^{(i)},$$

where n_F is the number of rows in F and where

$$\begin{aligned}\mathbb{L}_m^{(r)} := \big\{(M,C) \quad : \quad & H^{(j)}M\ell_i + VBC\ell_i \leq MA_L\ell_i, \\ & H_m^r M\ell_i + G^r C\ell_i \leq g^r - h_i^r, \\ & i = 0,\ldots,m-1, \quad j = 0,\ldots,\rho\big\},\end{aligned}$$

where the superscript notation denotes row selection. If row r of the constraint is redundant in $\mathbb{L}_{m+1}$ then $\mathbb{L}_m^{(r)} = \mathbb{L}_{m+1}^{(r)}$. Therefore, for any $(M',C') \in \mathbb{L}_m^{(r)}$, the same pair is in $\mathbb{L}_{m+1}^{(r)}$.

Now we define $\tilde{M} := M'A_L^\mu$ and $\tilde{C} := C'A_L^\mu$. By substitution of variables,

$$(\tilde{M},\tilde{C}) \in \mathbb{L}_m^{(r)} \Rightarrow (\tilde{M},\tilde{C}) \in \mathbb{L}_{m+1}^{(r)}$$

$$\begin{aligned}\Rightarrow (M',C') \in \big\{(M,C) \quad : \quad & MA_L^\mu\ell_{i+1} \geq H^{(j)}MA_L^\mu\ell_i + VBCA_L^\mu\ell_i, \\ & H_m^r MA_L^\mu\ell_i + G^r CA_L^\mu\ell_i \leq g^r - h_i^r, \\ & i = 0,\ldots,m-1, \quad j = 0,\ldots,\rho\big\}.\end{aligned}$$

Now, $A_L^\mu\ell_i = \ell_{i+\mu}$ and $h_{i+\mu} > h_i$, so $(M',C') \in \mathbb{L}_{m+\mu}$.

□

Thus, $\mathbb{L}_{n*}$ is positively invariant and feasible where n^* is the minimum n such that $\mathbb{L}_n = \mathbb{L}_{n+1}$.

5.7 Cost Function and Online Algorithm

This section derives a cost function for the tracking problem that is quadratic in the optimisation variables. We wish to minimise the following expected cost at

time k:

$$J_k = \sum_{i=0}^{\infty} \mathrm{E}_k \left[(z_{i|k} - \hat{x}_{k+i})^T Q (z_{i|k} - \hat{x}_{k+i}) + (u_{i|k} - \hat{u}_{k+i})^T R (u_{i|k} - \hat{u}_{k+i}) \right]. \quad (5.22)$$

For ease of presentation, the sequel is presented for a scalar input. The results work equally well for any number of inputs. We define a new lifted state:

$$\chi_{i+1|k} = \Psi_{k+i} \chi_{i|k} \quad (5.23a)$$

$$\chi_{i|k} = \begin{bmatrix} z_{i|k} \\ C_k^T \\ U_k^T \end{bmatrix}, \quad \Psi_{k+i} = \begin{bmatrix} A + BK + \Delta_{k+i} & B\ell_0^T & 0 \\ 0 & A_L^T & 0 \\ 0 & 0 & A_L'^T \end{bmatrix} \quad (5.23b)$$

The stage cost of (5.22) is equal to $\chi_{i|k}^T \bar{Q} \chi_{i|k}$, where:

$$\bar{Q} = \Lambda^T \begin{bmatrix} Q & 0 \\ 0 & R \end{bmatrix} \Lambda, \quad \Lambda := \begin{bmatrix} I & 0 & -\hat{B}\ell_0'^T \\ K & \ell_0^T & -\ell_0'^T \end{bmatrix}. \quad (5.24)$$

As proved in Theorem 7, the sum of the stage costs to infinity is equal to $\chi_{0|k}^T P \chi_{0|k}$, where positive symmetric P is calculated such that such that:

$$P - \mathrm{E}_k(\Psi_{k+i}^T P \Psi_{k+i}) = \bar{Q}, \quad i \in \mathbb{Z}, \quad (5.25)$$

a semi-definite program that can be easily solved offline [Boyd et al., 1994].

We now present the Exponential MPC algorithm, which is to solve the following QP for the measured x_k and the desired target input definition U_k:

$$\min_{M_k, C_k} \begin{bmatrix} x_k \\ C_k^T \\ U_k^T \end{bmatrix}^T P \begin{bmatrix} x_k \\ C_k^T \\ U_k^T \end{bmatrix} \quad \text{s.t.} \quad (M_k, C_k) \in \mathbb{L}_{n^*} \quad (5.26)$$

The control input is then simply

$$\mathcal{K}_7 \ : \ u_k = K x_k + C_k^* \ell_0, \quad (5.27)$$

where C_k^* is the minimising value of C_k in the QP of (5.26).

Theorem 11. *Controller $\mathcal{K}_7$ in recursively feasible.*

Proof. By construction, $\mathbb{L}_{n^*}$ is positively invariant. Now $M_k\ell_{i+1} = M_k A_L \ell_i = M_{k+1}\ell_i$, and similarly for C_k, therefore there exists a $(M_{k+1}, C_{k+1}) \in \mathbb{L}_{n^*}$. This is the only constraint applied online, so the proof is complete. □

5.8 Numerical Example

The system $\mathcal{G}_5$ from (5.4) is defined as follows for this example:

$$\begin{gathered} A = \begin{bmatrix} 0.9 & 0.1 \\ -0.1 & 0.9 \end{bmatrix} \qquad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad \rho = 2 \\ \Delta^{(1)} = 0.015 \cdot \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix} \qquad \Delta^{(2)} = -\Delta^{(1)} \qquad \bar{w} = 0.005 \\ w^{(1)} = \begin{bmatrix} \bar{w} \\ \bar{w} \end{bmatrix} \quad w^{(2)} = \begin{bmatrix} \bar{w} \\ -\bar{w} \end{bmatrix} \quad w^{(3)} = \begin{bmatrix} -\bar{w} \\ \bar{w} \end{bmatrix} \quad w^{(4)} = \begin{bmatrix} -\bar{w} \\ -\bar{w} \end{bmatrix} \qquad q = 4 \end{gathered} \tag{5.28}$$

The constraints are defined as:

$$F = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad G = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ -1 \end{bmatrix} \quad g = \begin{bmatrix} 0.3 \\ 1 \\ 1 \\ 1 \\ 0.05 \\ 0.05 \end{bmatrix} \tag{5.29}$$

The cost matrices and corresponding LQR gains are:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad R = 20, \quad K = L = \begin{bmatrix} -0.1396 & 0.0048 \end{bmatrix} \tag{5.30}$$

The uncertainty tube cross-section is chosen to be simply an orthotope, so $V = [I \ -I]^T$. Six Laguerre sequences are chosen, with a decay rate of $\gamma = 0.9$,

chosen to be approximately equal to the magnitude of the eigenvalues of Φ. The first 30 time steps of these are shown in Figure 5.2.

The target input function was chosen to evolve according to the same dynamics as the controller, i.e. $A_L = A'_L$ and $\ell_0 = \ell'_0$. The coordinates U_k were chosen to approximate an arbitrary function chosen to demonstrate the capabilities of the controller. Specifically,

$$U_k = \arg\min_U \sum_{i=0}^{120} \left(U\ell'_i - 0.04\sin(i\pi/60)\right)^2. \tag{5.31}$$

Figure 5.3 shows the target state and input functions evolving over 120 time steps with $n_L = 6$. Bounds on the polytopic tubes are shown. In the state space plots, the bounds at time step i are $\{x : Vx \leq M_k\ell_i\}$ and in the input space plot, the bounds are $\{Kx + C_k\ell_i : Vx \leq M_k\ell_i\}$.

Figure 5.4 shows the same but for $n_L = 10$. Compared to when $n_L = 6$, the target input function is closer to the ideal target input defined in (5.31). The size of the polytopic tube is smaller and its centre is nearer to the target. This is a result of adding extra degrees of freedom. The input degrees of freedom required in RMPC, i.e. single time step impulse basis functions, would be 120. Input blocking by blocks sufficiently large to reduce this number to 10 would require the same input to be applied for the first 12 time steps.

5.9 Conclusions

This chapter has brought together two successful RMPC techniques: polytopic tubes and exponential basis functions (EBFs). Chapter 3 gives an RMPC algorithm that requires one optimisation variable per input per time step, plus one per facet of the polytopic uncertainty sets per time step. Since we require recursive feasibility to extend into the infinite horizon, which implies stability of the closed-loop system, and since computational tractability requires the number of optimisation variables to be finite, a terminal set is required. Chapter 4 gives an overview of the challenges of building a terminal set when there is multiplicative uncertainty in the dynamics. The separation of the algorithm into a finite horizon

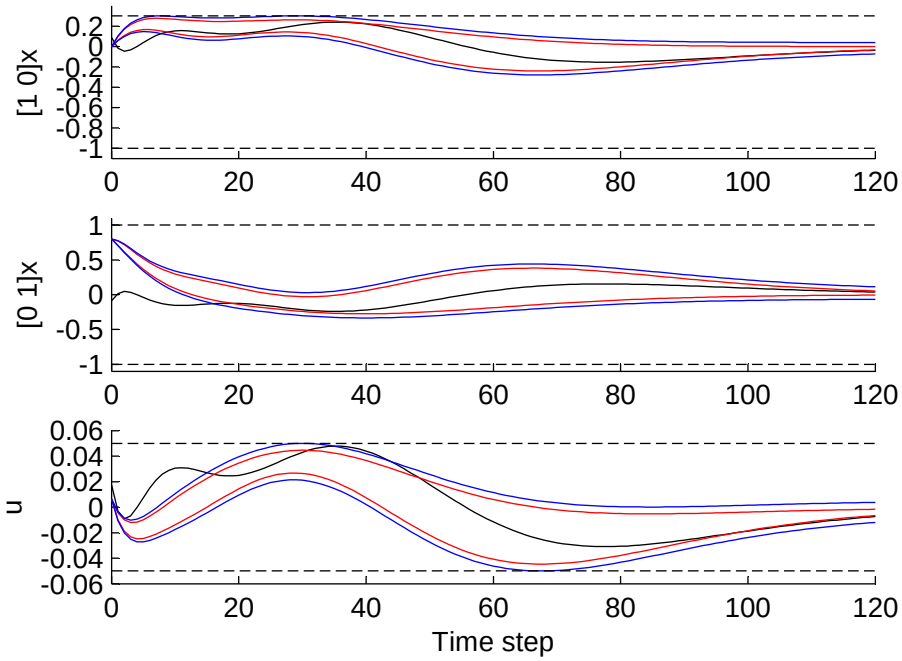

Figure 5.3: Numerical example of exponential basis function MPC with $n_L = 6$. Black curves are target states and inputs. Red curves are bounds of uncertainty polytopic tube. Blue curves are tube bounds plus additive constraint tightening. Dashed lines are constraints.

and a terminal set was termed 'dual mode'.

The use of exponential basis functions obviates dual mode because they are defined over the infinite horizon. This chapter redefines the polytopic tube in terms of EBFs and constrains it to be positively invariant under dynamics involving input degrees of freedom that are also present in the EBF space. Finally, the target input and state are in the EBF space, which can approximate a desired target input and state with an accuracy that depends on the number of EBFs employed.

This chapter gives a complete solution to a control problem involving both additive and multiplicative uncertainty. The constraint tightening values that are calculated offline cover all the uncertainty introduced by the additive part, which is subject to uncertain dynamics due to the multiplicative part. The remaining uncertainty is dependent on initial conditions and optimisation variables, which are not known offline. It is this 'online' uncertainty that is bounded by the EBF

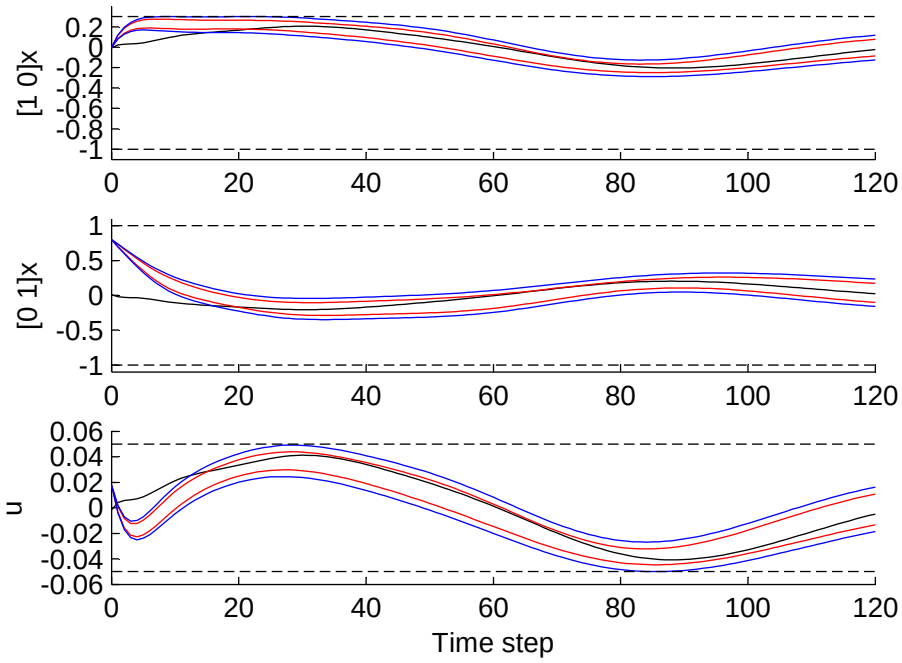

Figure 5.4: Numerical example of exponential basis function MPC with $n_L = 10$.

polytopic tubes. A numerical example illustrates the benefit of using more EBFs in the controller, while the computational complexity stays well below that of classical (single time step impulse basis function) RMPC for a prediction horizon long enough to cover the length of the target input sequence, in this case 200 time steps.

Often the lowest cost solution to a predictive control problem is one where the predicted states evolve in a similar way to the model dynamics. Classical impulse basis function MPC does not create such predictions naturally, whereas exponential basis functions can be chosen to match the dynamics of the model. More work is required to formalise the selection of basis functions such that they result in a computationally efficient algorithm. An extension that also deserves investigation is the development of an exponential basis function Stochastic MPC algorithm.

Chapter 6

Application to Wind Turbine Control

6.1 Chapter Outline

Recent research into wind turbine controllers has explored the use of MPC for its ability to explicitly handle state and input constraints. Robust MPC additionally accounts for uncertainty in the future evolution of the model states. This chapter demonstrates how the uncertainty in measured and predicted wind speeds propagates into the predicted system states and how RMPC can impose constraints that bound the likely extent of that uncertainty. The RMPC controller presented in this chapter is an application of the methods of Chapter 3 to a linear parameter varying (LPV) model that is designed specifically for this challenging practical problem.

State and input constraints are applied to a control model that is identified by data-driven methods. The robust controller bounds the uncertain predictions with a sequence of polytopes, which must fully satisfy the system constraints. The result is a controller that has improved closed-loop performance while retaining the computational complexity of a quadratic program. Modelling the wind turbine dynamics over a horizon of a few seconds requires a model of how the wind speed varies, both in time, as turbulence in the air moves past the rotor, and in space, since the wind speed is only measured at one location behind the rotor. Section 6.7

investigates the statistics of turbulent wind. The probability distributions are approximate; as such it is impossible to know how much risk there is that the bounds assumed on those distributions may be exceeded. For the purposes of demonstrating an application of RMPC, that risk is assumed to be zero.

To investigate the practical benefit RMPC can bring over nominal MPC, the two controllers are employed to regulate the rotor speed, electrical power and tower oscillations of a large wind turbine in a state-of-the-art simulation environment, and the respective performances are compared. The same comparison is given in [Evans et al., 2014, in print].

6.2 Background to Pitch Control

This section gives a short overview and orientation of the physical process of generating torque from the wind. Figure 6.1, showing two views of a three-bladed wind turbine, illustrates much of the terminology employed throughout this chapter. For the purposes of discussing control of the rotor speed and thrust force, the most important feature of the wind turbine is the blade pitch angle, whereby each blade (though usually controlled collectively) rotates along its length-wise axis to a pitch angle defined by the control system. Changing the pitch angle of a blade varies the angle of attack of its aerofoil, which alters the magnitude and direction of its lift force.

The side view of Figure 6.1 shows a blade end-on, and demonstrates the 'apparent wind' that the blade experiences. The apparent wind vector is composed of a downwind 'free stream' component and a component in the rotor plane due to the movement of the blade through the air as it rotates around the hub. The difference between the angle of the apparent wind and the pitch angle is the angle of attack. The angle of attack and the magnitude (speed) of the apparent wind determine the lift that the blade aerofoil produces.

The aerodynamic lift can also be decomposed into an in-plane component that generates rotor torque, and an out-of-plane component termed 'thrust'. This cross-sectional view of the rotor blade over-simplifies the physics somewhat, as the apparent wind vector depends on the distance of the cross-section from the centre of the hub, and the blade is designed with a lengthwise twist.

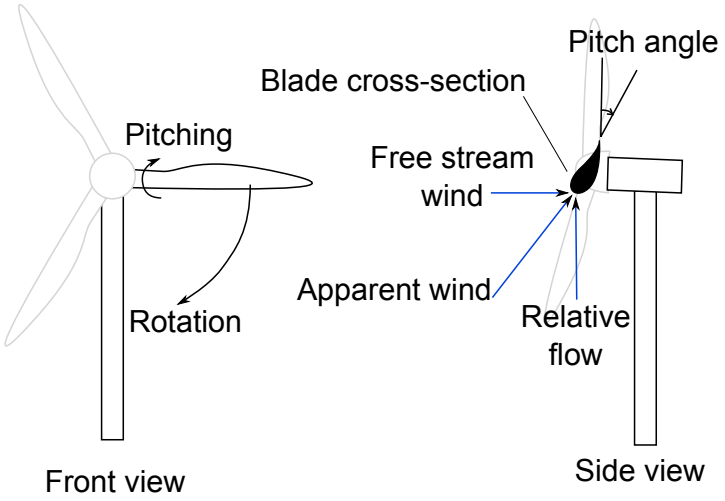

Figure 6.1: Front view (downwind) of a three-bladed wind turbine, with one blade and the tower highlighted, the other two blades and the hub greyed out. Side view (wind direction left-to-right) of the same wind turbine, with highlighted blade shown in cross-section, in front of greyed out hub and two other blades. Also visible in this view is the nacelle on top of the tower. Pitch angle of highlighted blade is measured clockwise end-on, with zero pitch signifying the blade is orientated such that the low pressure side faces downwind. Apparent wind is vector sum of free stream wind and relative flow due to blade motion through the air.

The blade is designed such that the maximum torque is generated at low pitch angles for low wind speeds. The wind speed for which the optimal pitch angle results in the maximum allowed torque is the 'rated wind speed'. Above that wind speed, it is necessary to reduce the lift that the blades produce. This can be achieved in two ways, 'active stall' and 'active pitch'. Increasing the angle of attack, by either increasing the wind speed or decreasing the pitch angle, can result in air flow over the aerofoil detaching from the low pressure surface, causing the blade to stall, as illustrated in Figure 6.2. Stall produces a sudden reduction in the lift force so it can be employed in a wind turbine control scheme to regulate the rotor speed and torque. Active stall controllers benefit from the passive reduction in aerodynamic power caused by stalling due to gusts of wind but they suffer from unpredictable forces on the blades during stall conditions [Burton et al., 2011].

An alternative control scheme from active stall is active pitch, whereby the angle of attack is reduced by increasing the pitch angle (also known as feathering),

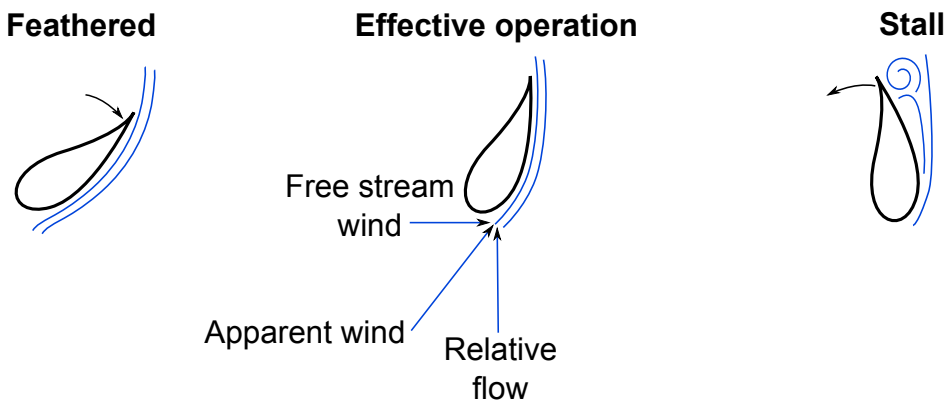

Figure 6.2: During effective operation of the turbine blade, air flows over the low pressure side, staying attached and laminar. If the angle of attack is increased sufficiently, either by decreasing the pitch angle, increasing the free stream wind speed or decreasing the rotor speed, the flow detaches and the blade stalls.

resulting in a controlled reduction in lift. The vast majority of modern large wind turbines use active pitch because it results in lower extreme loads on the blades. Pitch actuation systems for active pitch controllers are expensive and suffer fatigue damage of a different form from the load cycling experienced in the tower. Reduction of pitch system fatigue is outside the scope of this thesis but it is noted that less pitch 'travel', the sum of the magnitude of the pitch angle, is beneficial.

The sum of the thrust forces of the three blades is transferred to the nacelle at the tower top. Thrust and the tower's stiffness result in lightly damped tower oscillations. The apparent wind speed changes as the tower oscillates. At the tower top, the movement of the tower in the fore-aft direction can peak at a few metres per second, which changes the thrust and therefore the damping ratio – an uncertain process termed 'aerodynamic damping'.

Large wind turbines suffer fatigue damage to the tower due to its fore-aft motion. Structural mitigation of oscillation adds significantly to the material cost of the machine. Passive dampers have been used [Colwell and Basu, 2009] to reduce tower fatigue, such as pendula in oil, but these are expensive and must be tuned to a single frequency – a somewhat restrictive property since that frequency

is sensitive to ambient temperature and the tower oscillation frequency is sensitive to wind speed.

Active control of the rotor thrust is an attractive alternative approach. A thorough review of control methods for variable speed turbines is given in [Bossanyi, 2003], including active damping of the first fore-aft tower mode. Rotor thrust control can be achieved by varying the blade pitch angle, which is normally used only to control the rotor speed. Gain scheduling is shown to be important due to the wide variation of pitch sensitivity with wind speed. Coupling between pitch control for tower damping and pitch control for rotor speed above rated wind speed motivates a multivariable control approach. An LPV approach that covers the entire operating wind speed range is found in [Østergaard et al., 2009].

As QP solvers become more advanced, MPC is becoming an attractive method of control of wind turbines, which have constraints such as the stall pitch angle (where pitch angles too low for the rotor speed and wind speed cause the rotor blades to stall), maximum rotor speed and maximum power.

Early work that aimed at controlling a wind turbine with predictive control, e.g. [Nanayakkara et al., 1997], did so without constraints. The predictive element of MPC has often required the estimation of future wind speeds. In [Kani and Ardehali, 2011], past wind speeds were used to predict wind speeds in the very short term future. However, turbulence makes any statistical predictions difficult, so much uncertainty remains in the wind speed predictions using this method.

In modern turbines there is the potential for wind speed feed-forward with light detection and ranging, or lidar, a technology for measuring the wind speed upstream of the rotor [Hardesty and Weber, 1987]. With today's technology, lidar systems do not perfectly predict the wind speeds that the rotor will encounter, meaning that even a control scheme that does exploit such a preview signal would have uncertain predictions. Since there is a great deal of work already on the subject of incorporating lidar into MPC, this thesis does not consider the use of any preview signal. This decision also makes the presented algorithm more widely applicable, as most wind turbines still do not have lidar installed.

Much of the earlier work has applied linear MPC to wind turbines by restricting the control model to a single operating point. However, linear models of a wind turbine depend greatly on wind speed and this identifies a need for a control

scheme that covers the whole range of wind speeds, as in [Kumar and Stol, 2009; Soliman et al., 2011]. Both studies handle uncertainty in the future wind speed by using multiple models, each with an MPC controller, providing multiple inputs that are switched or mixed to arrive at the control action. The mixing operation is performed on the basis of a filtered wind speed scheduling signal. In [Henriksen et al., 2012], a hybrid MPC structure is proposed to handle switching between operating modes. Results therein show, using a high-fidelity aeroelastic code, that MPC can handle constraints more favourably than PI control with anti-windup.

Multiple model MPC such as [Kumar and Stol, 2009; Soliman et al., 2011] has a drawback that while each model can be optimised for a different wind speed, the predictions for that model assume constant wind speed throughout the horizon. In reality, the wind speed will vary significantly during the prediction horizon. The RMPC controller in this chapter allows for a range of possible wind speeds in the prediction horizon by bounding the set of uncertain state predictions by polytopes for the purposes of enforcing state and input constraints. This allows for uncertainty in both the state transition matrix and the input gain matrix, whereas previous work on RMPC in the context of wind turbine control such as [Koerber and King, 2013; Mirzaei et al., 2012] has restricted this uncertainty to the latter for ease of derivation of an optimisation method.

RMPC requires a model of the controlled plant, which must contain reasonably accurate bounds on the distributions of uncertain parameters and exogenous inputs. We derive a simple piecewise linear model whose matrix elements are functions of wind speed. These functions are estimated in a straightforward manner by offline data driven methods. There is a great deal of research into model identification for wind turbines, for example [Kusiak et al., 2009; van der Veen, 2013] and [van der Veen et al., 2013], which demonstrates identification of control models in data where turbulence is present. The focus of this chapter is on the handling of uncertainty in a given model; therefore a simple model identification technique is presented in place of advanced specialised methods to find such a model.

The work of [Scokaert and Rawlings, 1999] shows why it is not desirable to circumvent recursive feasibility problems with soft constraints applied to nominal MPC. In such a control scheme, the solutions to the QP at each time respect all the constraints whenever that is feasible. But due to unreckoned uncertainty, the

state deviates from these predictions. Then at some time step, the problem may become infeasible, at which stage the controller switches off the violated constraint and instead applies a cost to its violation. This creates a feasible QP but with a different objective. The switch between these controllers is unmodelled and can produce undesirable transients. In contrast, RMPC prevents the constraint from being violated by accounting for all possible instances of uncertainty in the prediction horizon.

RMPC requires that the uncertainty distribution is finitely supported, which is the case in all physical systems. The support may be very wide, with long unlikely tails. This chapter reduces conservativeness by approximating probability distributions with long tails by similar distributions without such tails. This retains the benefit of RMPC over nominal MPC, while introducing a small manageable risk that an uncertainty sequence may arise that is outside the imposed bounds. The study of these risks, which motivate Stochastic MPC, is the subject of future research.

6.3 Background to Fatigue Estimation

Materials such as steel fatigue as they are subjected to stress and strain cycles. For a repeated cycle of a constant amplitude s, the number of such cycles before material failure is approximated to be:

$$N(s) = Ks^{-m}, \tag{6.1}$$

where K is a material constant representing the total fatigue that the material can withstand and where m is a material constant called the Wöhler coefficient, which is around 4 for steel [Hammerum, 2006]. The relationship of (6.1) is called the S-N curve.

In real operation, the material will be subjected to a more complex loading pattern than a simple repeated cycle. To combine many cycles of different amplitudes, we use the Palmgren-Miner damage rule to calculate D, the fraction of

the fatigue life used up:

$$D = \sum_i \frac{1}{K} s_i^m. \tag{6.2}$$

The challenge is to extract out the equivalent stress cycles s_i from a stress time series. The most common method for calculating this is the rainflow count. Developed in [Matsuishi and Endo, 1968], the process originally required an entire time series, requiring a large amount of data storage and a dedicated offline process.

In [Downing and Socie, 1982], a rainflow counting algorithm is given to calculate the cycles online. However, the sequences required internally could still grow large over time. More recently, [Amzallag et al., 1994] reduced the computation further to a four-point method. This process identifies half-cycles and either removes pairs that form full cycles or discards excess half-cycles into a 'residual' pile to be processed at the end of the run. The technique given in Section 6.11 removes the requirement for a residual by counting the half-cycles separately, to be combined with the full cycles using an approximation to (6.2). More detail on the relationship between rainflow counts and fatigue can be found in [Hammerum, 2006]. The remainder of this section explains the computational steps required to find the rainflow cycles.

The first stage of rainflow counting is to reduce the stress time series to a series of alternating maxima and minima, as illustrated in Figure 6.3.

The second stage is to find stress cycles within this series of extrema. This can be visualised as rain flowing with time over the slopes. (It helps to rotate the page clockwise.) When the water falls off an extremum and lands on another slope, it defines a cycle provided that it does not extend beyond the end of an existing cycle that starts prior. That is, cycles must be nested, not overlapping. The valid rainflows are shown in Figure 6.4.

In the example given in Figure 6.4, the cycles have stress ranges of 2.6, 0.48, 5.4 and 2.3 (at 21 s, 35 s, 44 s and 72 s respectively). When raised to the Wöhler coeffecient 4, these ranges sum to the accumulated damage $KD = 906$. It is often more useful to express the accumulated fatigue damage in terms of an equivalent loading – the stress range required to produce the same accumulated damage when repeated a given number of times. This is proportional to the m-norm of the series of cycle amplitudes. The equivalent load $s_e(N)$ for $N = 100$ cycles for the above

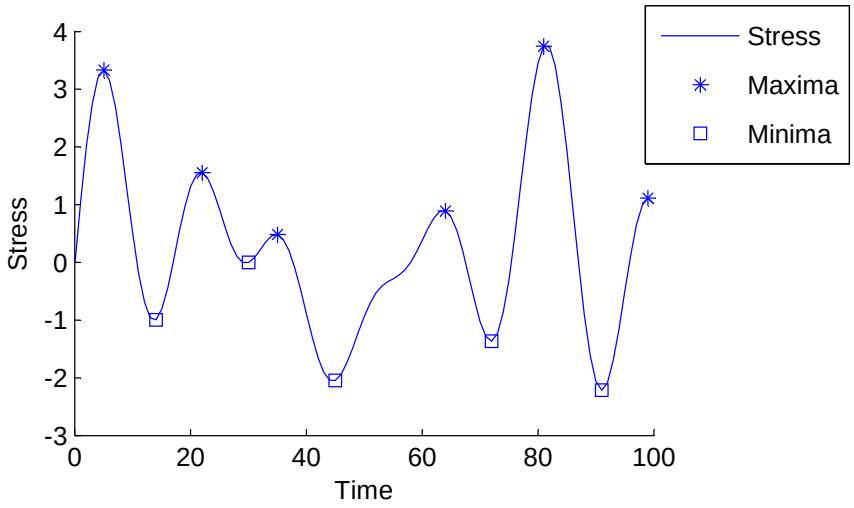

Figure 6.3: An example stress time series showing maxima and minima.

example, the calculation would be:

$$s_e(N) = \left(\frac{KD}{N}\right)^{1/m} = 1.73 \tag{6.3}$$

6.4 Simulation Environment

In the wind turbine industry, controllers are tested in a numerical simulation environment rather than a physical scale model or a real wind turbine. The simulation comprises an aeroelastic code, embedded into a Simulink block. The aeroelastic code used in this study is based on FLEX 5 [Øye, 1999], has 23 degrees of freedom and is sampled at 100 Hz. As shown in Figure 6.5, the simulation requires inputs for the pitch actuation valves and generator torque. Existing controllers and the MPC controllers presented here output pitch angle and generator power references instead. Therefore, existing regulation controllers have been employed in this study, to ensure the plant behaves as realistically as possible. Additionally, the optimal set-points block is unmodified from the existing simulation environment, as it is adept at calculating set-points for this particular turbine model.

The newly added blocks in the simulation environment are the proportional-integral square error calculation, the MPC controller and the wind speed and tower

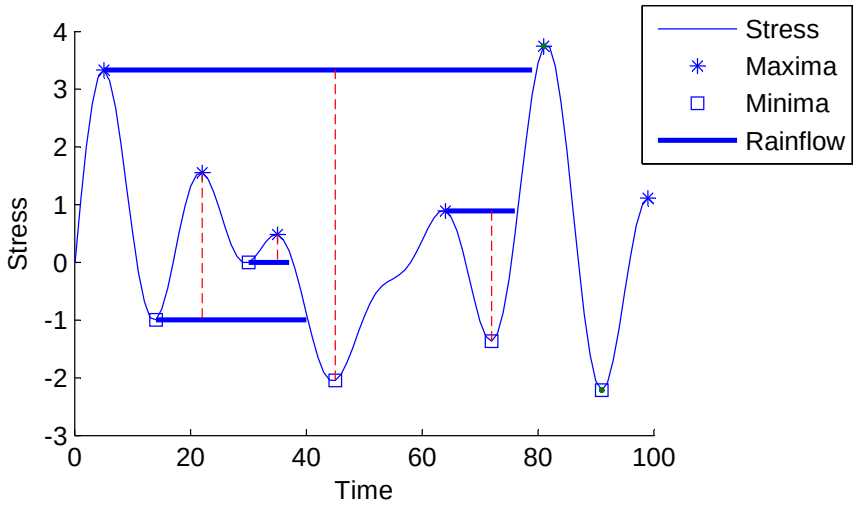

Figure 6.4: An example stress time series showing valid rainflow and cycles

filter data processing blocks. The block diagram also shows the sample rates at which all of these systems operate. The optimal set-point block runs at a slower sample rate than the simulator for the purpose of matching with a control block from the existing controller, now removed. The PI block is to remove steady-state offset in the squared generator speed. Numerical examples are computed with [Gurobi Optimization, 2014]. Constraints are built in Yalmip [Löfberg, 2004].

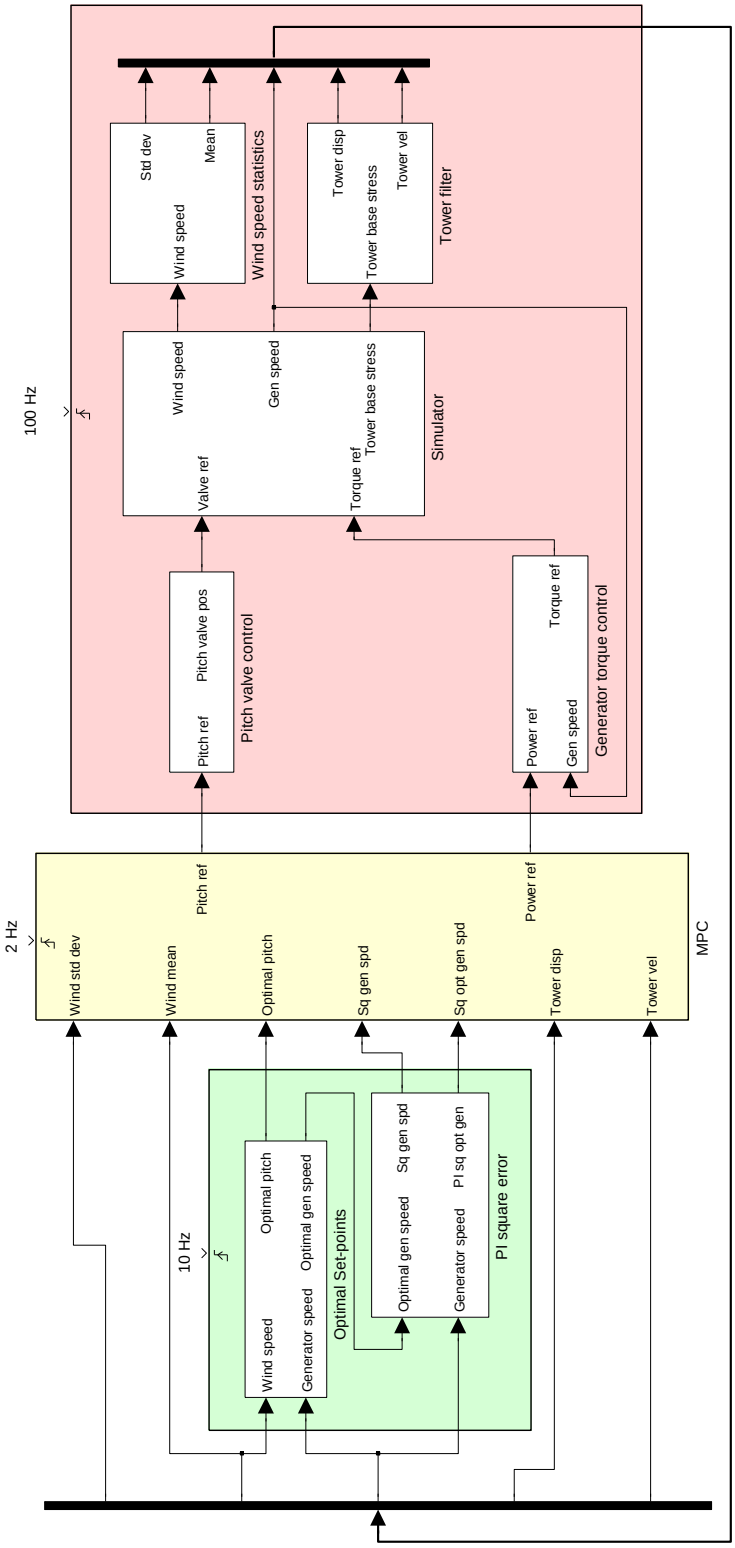

Figure 6.5: Simulation environment comprising MPC controller, regulation controllers, aero-elastic code and auxiliary systems.

6.5 Nonlinear Dynamics

This section extends the introduction of Section 6.2 to provide a context into which the control model of Section 6.6 is introduced. When assessing the performance of the controllers presented in this thesis, the simulation environment described in Section 6.4 is used, which does not have the simplifications of this section applied. The control model takes these simplifications and applies further simplifications in order to obtain a model that is sufficiently low-order to accommodate RMPC.

The simplified model assumes rigid rotor blades, rigid drivetrain, negligible generator and pitch system dynamics and no pitch-induced dynamic lift. A standard wind turbine model [e.g. Østergaard et al., 2009] incorporating the out-of-plane thrust on the rotor, mechanical power in the rotor and electrical power in the generator, and a lumped-parameter model of tower bending is summarised. The symbols and parameters are explained in Table 6.1.

P_a	Mechanical rotor power	F_T	Thrust acting on tower downwind
a	Rotor area	c_T	Coefficient of thrust
v	Wind speed (averaged over the rotor disk)	k, γ, m	Lumped parameter tower stiffness, damping, mass
ρ	Air density	p	Tower top displacement fore-aft
c_P	Coefficient of power	J	Rotor+drivetrain+generator inertia
β	Pitch angle	Q_a	Rotational torque at rotor
λ	Tip speed ratio $\propto \Omega/v$	Q_e, P_e	Torque at generator, generator power
Ω	Rotor speed		

Table 6.1: Nonlinear model parameters

A wind turbine derives its rotational mechanical power from the in-plane component of lift as air flows over the surfaces of the rotor blades. The lift force depends on the pitch angle of the blades, the rotor speed and the wind speed. Generator speed is taken to be proportional to rotor speed due to a rigid drive train assumption. The mechanical power at the rotor hub is proportional to the cube of the wind speed for a given pitch angle and tip speed ratio. That power is equal to the product of the rotor torque and speed:

$$P_a = \frac{1}{2} a \rho v^3 c_P(\beta, \lambda) = Q_a \Omega. \tag{6.4}$$

The out-of-plane lift component is thrust, a force that is transferred to the tower top in the downwind direction. The thrust force is proportional to the square of the wind speed for a given pitch angle and tip speed ratio. The tower can be considered a lightly damped flexible beam. The motion of the rotor disk into the wind changes the thrust, which introduces damping that varies significantly with wind speed. Thus thrust is related to the tower dynamics as follows:

$$F_T = \frac{1}{2} a \rho v^2 c_T(\beta, \lambda) = kp + \gamma(v)\dot{p} + m\ddot{p}. \tag{6.5}$$

Power is extracted from the turbine by a generator, connected via the drive train to the rotor. The difference between the aerodynamic torque and the generator torque (measured on the rotor end of the drive train) accelerates the rotor according to Newton's second law for rotation. Multiplying through by rotor speed gives:

$$P_a - P_e = \Omega(Q_a - Q_e) = \Omega\dot{\Omega}J. \tag{6.6}$$

Now (6.6) can be written as a linear first order equation in the square of the rotor speed:

$$P_a - P_e = \frac{1}{2}\frac{d\Omega^2}{dt}J. \tag{6.7}$$

In the absence of constraints on the generator power, controlling the rotor speed would be a trivial case of matching generator power to aerodynamic power. This is indeed the case when the wind speed is below rated, a situation termed 'partial load'. However, above rated wind speed, aerodynamic power must be reduced by pitching into a less efficient angle of attack, termed 'full load'. The present work presents a controller that respects the underlying constraints rather than treating these two operating modes separately.

6.6 Control Model

For the purposes of controller design, we adapt (6.4)-(6.6) to construct a state-space model with state $x_k \in \mathbb{R}^{n_x}$ at time step k comprising fore-aft tower displacement p, fore-aft velocity $\dot{p}$ and rotor speed squared Ω^2. The square of the rotor speed is used to provide a linear relationship with electrical power. The vector

$u_k \in \mathbb{R}^{n_u}$ of input variables to be specified by the controller at time step k comprises pitch angle β and generator power P_e. The dynamics are approximated so as to evolve the state according to the LPV difference equation

$$x_{k+1} = A(v_k)x_k + B(v_k)u_k + w(v_k), \tag{6.8}$$

where matrices $A(v), B(v), w(v)$, which depend explicitly on wind speed v, are obtained by data driven methods based on time series taken from closed-loop operation under an existing controller in the aeroelastic simulation environment and are treated as being piecewise linear in v.

The constraints considered are: the blades must not stall or exceed 90° in pitch angle; the squared rotor speed, as a ratio of the squared nominal rotor speed for a given wind speed $\bar{\Omega}^2(v)$, must lie within a fixed interval, e.g. $[m^-, m^+] = [0.75, 1.25]$; and the generator power must lie between zero and nominal, $\bar{P}_e$. For a given wind speed and rotor speed, there is a known stall angle $\beta_s(v, \Omega)$, which the pitch angle must exceed. Symbolically these constraints are written as follows.

$$\begin{bmatrix} \beta_s(v,\Omega) \\ 0 \end{bmatrix} \leq u \leq \begin{bmatrix} 90 \\ \bar{P}_e \end{bmatrix}, \quad \bar{\Omega}^2(v)m^- \leq \Omega^2 \leq \bar{\Omega}^2(v)m^+ \tag{6.9}$$

For any given wind speed there is a known optimum rotor speed. The nominal rotor speed is the smaller of this optimum and a limit imposed by the turbine design. The predicted states and inputs in discrete time are variables in the optimisation problem so the rate of change of predicted states and inputs are linear combinations of these. Rate constraints are therefore easy to apply but are not incorporated in the controllers of this chapter for simplicity of presentation.

The constraints of (6.9) are more usefully written as mixed constraints in a general form:

$$Fx + Gu \leq g(v). \tag{6.10}$$

Although (6.10) is time varying, mild assumptions allow us to treat $g(v)$ as constant over the prediction horizon. Predictions are made N steps ahead of the present state, with an assumption of perfect measurements, although measurement uncertainty can be easily incorporated into RMPC.

Control model identification is performed around wind speeds in the discrete set $\mathbb{V}$ to give the model set $\left\{\hat{A}(v) \ : \ v \in \mathbb{V}\right\}$. Values of $A(v)$ between these wind speeds are linearly interpolated. Functions $B(v)$ and $w(v)$ are defined analogously in terms of $\mathbb{V}$. Section 6.10 discusses model identification where derivations of the values for these functions are given.

6.7 Wind Speed Uncertainty

The temporal variation of the downwind component of the wind speed at a fixed point in space, $v(t, r)$, where r is a spatial variable, can be approximated by the Kaimal distribution [Burton et al., 2011], which specifies [IEC, 2005, 2009] the power spectral density (PSD) as a function of temporal frequency:

$$S(f) = \sigma^2 \frac{4L_x/U}{(1 + 6fL_x/U)^{5/3}}, \tag{6.11}$$

where U, σ are the mean and standard deviation of the wind speed and where L_x is the turbulence length scale, a characteristic of the wind that depends on the terrain. These three variables are assumed to vary so slowly that they are treated as constant. All simulations in this chapter use a value of $L_x = 340\,\mathrm{m}$ from [Kristensen and Jensen, 1979]. It is useful to introduce a non-dimensional variable to define the standard deviation of the wind termed 'turbulence intensity' T_i. The standard [IEC, 2009] defines the standard deviation as:

$$\sigma = T_i(\tfrac{3}{4}U + 5.6). \tag{6.12}$$

If the turbulence intensity is small, Taylor's hypothesis [Powell and Elderkin, 1974] allows us to treat the advection of a large volume of air to be the mean velocity of that air. This approximation means that although the wind speed at one point in front of the rotor may be higher than the mean, that gust will arrive at the rotor approximately simultaneously with the surrounding air, i.e. the turbulence is 'frozen'. The turbulence is also assumed to be approximately ergodic, allowing us to use Parseval's Theorem to relate the variance of the signal with its

PSD. This relationship is:

$$\sigma^2 = \int_{-\infty}^{\infty} S(f)df. \tag{6.13}$$

For the purposes of RMPC, the scheduling parameter $\tilde{v}$ must vary slowly compared to the controller sample time. The only spatial location where measurements are taken is the nacelle anemometer, at $r = 0$. A low-pass filter removes most of the rapid changes in the anemometer signal. The filter is defined by a convolution with an impulse response as follows:

$$\tilde{v}(t) = v(t, 0) * e^{-t/\tau}u(t), \tag{6.14}$$

where τ is the exponential time constant of the filter and where $u(\cdot)$ is the Heaviside step function. The Fourier transform of the low pass filter is:

$$H_t(f) = \frac{1/\tau}{1/\tau + j2\pi f}, \quad j^2 = -1. \tag{6.15}$$

If we assume the wind speed at the anemometer will remain in the range $U \pm n_c\sigma$, where n_c is a constant specifying a confidence level, then the scheduling parameter is bounded throughout the prediction horizon according to:

$$\tilde{v}(t + i\Delta t) \in \left\{(U + z\sigma - \tilde{v}(t))(1 - e^{-i\Delta t/\tau}) + \tilde{v}(t) \ : \ -n_c < z < n_c\right\}. \tag{6.16}$$

We can additionally impose bounds (with the same level of confidence) on the actual wind speed at the anemometer by reconstruction:

$$v(t, 0) \in \left\{\tilde{v}(t) + z\sigma_e \ : \ -n_c < z < n_c\right\}, \tag{6.17}$$

where σ_e is the standard deviation of the wind signal after the subtraction of the low-pass filtered signal. This can be calculated from Parseval's Theorem applied to the filtered signal:

$$\sigma_e^2 = \mathrm{Var}(v(t, 0) - \tilde{v}(t)) = \sigma^2 - \int_{-\infty}^{\infty} H_t(f)H_t^*(f)S(f)df. \tag{6.18}$$

So far, the analysis has focussed on the wind speed at the single location in

space denoted $r = 0$. The rotor actually has a spatial averaging effect on lift forces over the rotor area a. There are no sensors to measure this average wind speed, although a wind speed estimator could be used, such as in [Mirzaei et al., 2012]. Instead, we account for the averaging effect probabilistically, by convolving the spatial PSD with a mask representing the rotor. This mask can be chosen to have higher value towards the last third of the blade length to account for the uneven distribution of lift on the blades, but for simplicity, it is here represented by a square of side length $\sqrt{a}$. The box mask in one spatial dimension is defined as the product of two step functions:

$$\Pi(r) = u(-r - \sqrt{a}/2)u(r + \sqrt{a}/2), \tag{6.19}$$

which has the Fourier transform, in terms of a spatial frequency variable ϕ,

$$H_s(\phi) = \text{sinc}(\sqrt{a}\phi). \tag{6.20}$$

The PSD of the downwind component of the wind speed in the rotor plane, $S(\phi)$, is very difficult to study due to the sensor array that would be required. Assuming Taylor's frozen turbulence, and ignoring the wind shear due to ground friction, allows us to approximate $S(\phi)$ similarly to the temporal PSD of (6.11). This gives:

$$S(\phi) = \sigma^2 \frac{4L_x}{(1 + 6\phi L_x)^{5/3}}. \tag{6.21}$$

More realistic design of this function is outside the scope of this thesis but the principles given here apply to any spatial PSD. Useful work in this field includes [Petersen et al., 1998] and [Beaupuits et al., 2004].

The variance of the spatial PSD, when convolved with the spatial filter (6.19) in the two dimensions of the rotor plane, is σ_s^2, calculated as:

$$\sigma_s^2 = \int_{-\infty}^{\infty} (H_s(\phi)H_s^*(\phi))^2 S(\phi) d\phi. \tag{6.22}$$

The ratio σ_s^2/σ^2 is the reduction in the variance of the wind speeds at an instance in time due to the spatial averaging effect of the rotor. This reduction applies to temporally filtered signals too, so we can write the variance of the

uncertainty in the scheduling parameter $\hat{\sigma}^2$ as the reduction ratio multiplied by the variance due to the temporal filter from (6.18), i.e.

$$\hat{\sigma}^2 = \sigma_e^2 \frac{\sigma_s^2}{\sigma^2}. \tag{6.23}$$

Suitable values for the parameters in this section can be found by analysing recorded wind time series or by simulation such as in [Koerber and King, 2013]. A numerical example of these filters is presented in Figures 6.6 and 6.7. An example anemometer measurement of 600 s length taken from the simulation environment is Fourier transformed and smoothed to illustrate the similarity of its frequency content to the theoretical Kaimal spectrum. Low frequencies are not shown due to the limited time series length. The product of the Kaimal spectrum and the filter spectrum is the expected spectrum of the filtered measurements.

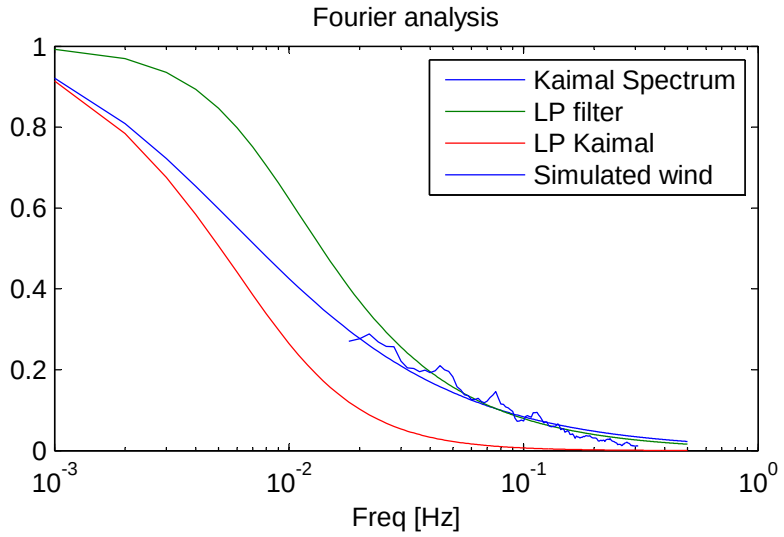

Figure 6.6: Kaimal spectrum for mean wind speed 10 m/s and turbulence intensity 21%, shown with smoothed finite Fourier transform (FFT) of an example wind speed time series with the same mean speed and turbulence intensity. A low-pass filter with time constant 20 s is shown in green. The product of the Kaimal spectrum and the filter is shown in red.

Figure 6.7 shows both filters: the temporal filter in the Fourier domain and in the time domain, and the spatial filter in the Fourier domain and the space domain. The time and space series (lower two plots) were created by taking the

amplitude of the Kaimal spectrum, applying a random phase for each frequency, uniformly distributed in $[0, 2\pi]$, and performing an inverse Fourier transform.

The wind parameters for the examples shown in Figures 6.6 and 6.7 are: mean wind speed 10 m/s and turbulence intensity 21%. The temporal filter has a time constant of 20 s. The spatial filter is designed to have the same area as a rotor of diameter 112 m.

6.8 Robust Control Design

The control model is uncertain because the future wind speeds are uncertain and due to model-plant mismatch. The approximation error in the control model for any given wind speed, i.e. the difference between the dynamics of (6.4)-(6.6) and (6.8) is subsumed into the uncertainty on the additive term $w(\tilde{v})$. The predicted wind speed at time step $k+i$, denoted $v_{i|k}$, is modelled as the slowly moving part $\tilde{v}_{k+i}$ that lies in the range (6.16), plus the stationary uncertainty given in (6.23).

To form constraints from these uncertain predicted wind speeds, two sets of constraints are applied: one for a range of uncertainty around the upper bound on $\tilde{v}_k$ and one for a range around the lower bound. The two time varying centres of these uncertainty sets are:

$$v^{\pm}_{i|k} := (U \pm n_c\sigma - \tilde{v}_k)(1 - e^{-i\Delta t/\tau}) + \tilde{v}_k. \tag{6.24}$$

The half-width of each set is $n_c\hat{\sigma}$. An illustration of this relationship is given in Figure 6.8.

We proceed to approximate the probability distribution of the wind speeds through the horizon. Further study into the statistics of the wind turbulence could be used to produce an ideal distribution for this purpose but for simplicity we instead use the triangular distribution, the density of which is given by

$$f(v_{i|k}) = \begin{cases} (r_{i|k} - |v_{i|k} - v^0_{i|k}|)/r^2_{i|k} & : \quad |v_{i|k} - v^0_{i|k}| < r_{i|k} \\ 0 & : \quad \text{otherwise} \end{cases}, \tag{6.25}$$

where $r_i := (v^+_{i|k} - v^-_{i|k})/2 + n_c\hat{\sigma}$ and where $v^0_{i|k} := (v^+_{i|k} + v^-_{i|k})/2$.

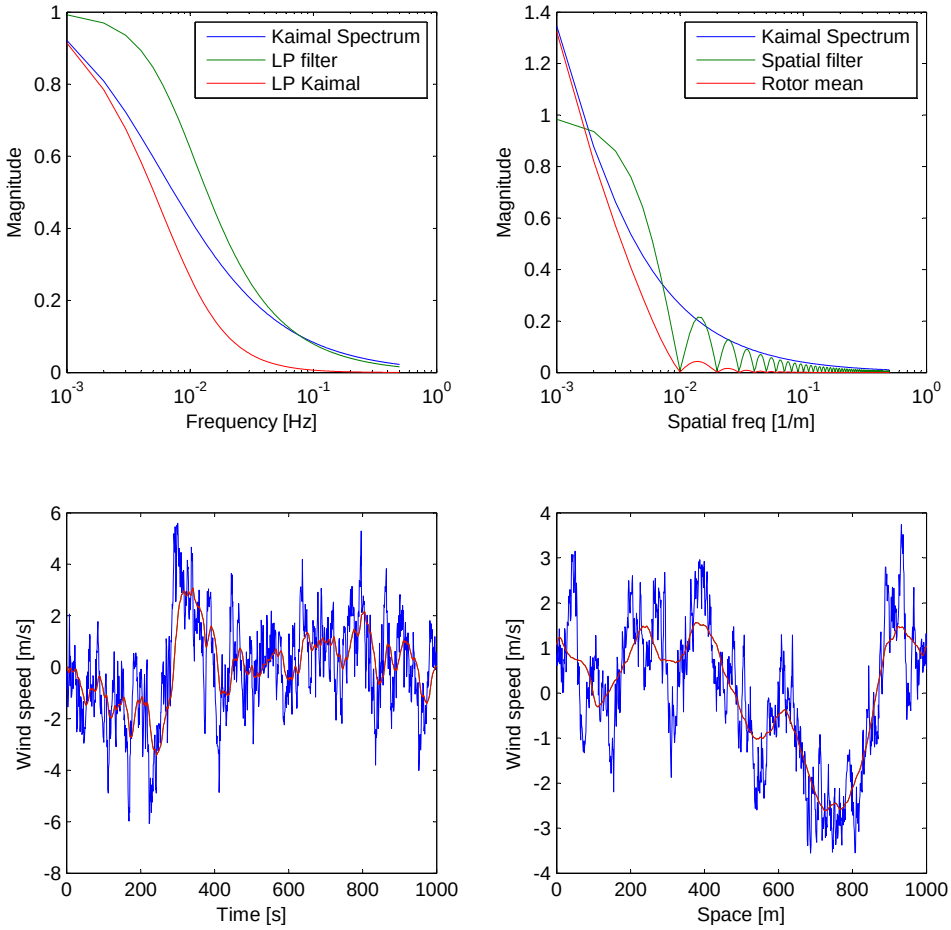

Figure 6.7: (Upper left) Temporal Kaimal spectrum, low-pass filter and filtered Kaimal spectrum in the Fourier domain. (Upper right) Spatial Kaimal spectrum, box filter and filtered Kaimal spectrum in the Fourier domain. The box filtered spatial spectrum represents the rotor mean wind speed. (Lower left) Example time series with frequency content prescribed by the temporal Kaimal spectrum, and low-pass filtered time series. (Lower right) Example space series with frequency content prescribed by the spatial Kaimal spectrum, and box filtered space series.

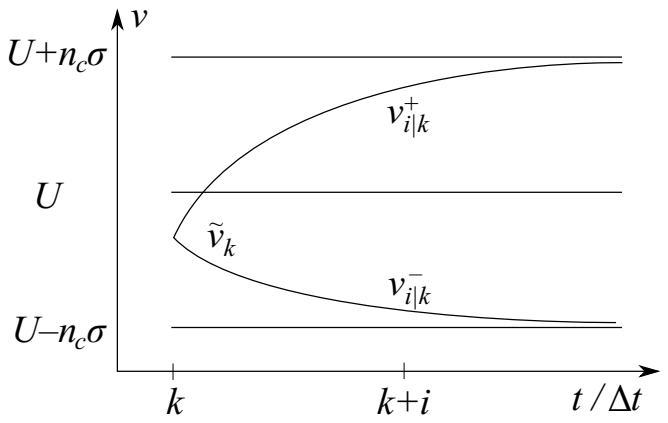

Figure 6.8: The range of possible future low-pass filtered wind speeds, assuming n_c standard deviations from the mean wind speed U and exploiting a measurement at time step k.

The uncertain dynamics are:

$$x_{i+1|k} = (A^{\pm}_{i|k} + A^{\delta}_{i|k})x_{i|k} + (B^{\pm}_{i|k} + B^{\delta}_{i|k})u_{i|k} + (w^{\pm}_{i|k} + w^{\delta}_{i|k}) \tag{6.26}$$

where[1] $A^{+}_{i|k} := A(v^{+}_{i|k})$ and where the superscript δ denotes stationary uncertainty, with $A^{\delta}_{i|k}$ belonging to a polytopic set, defined as:

$$A^{+}_{k} + A^{\delta}_{i|k} \in \left[A(v^{+}_{i|k} + n_c\hat{\sigma}), A(v^{+}_{i|k} - n_c\hat{\sigma})\right]. \tag{6.27}$$

State uncertainty over the prediction horizon is handled by a decomposition:

$$x_{i|k} = z_{i|k} + e_{i|k}, \quad z_{i|k} = \mathrm{E}_k(x_{i|k}), \tag{6.28}$$

where $z_{i|k}$ is the nominal prediction and $e_{i|k}$ is the uncertain component of the predicted state, which is contained within a polytopic prediction uncertainty set denoted by $\mathbb{P}_{i|k}$. The purpose of this set is to facilitate the construction of constraints that account for the uncertainty in the predictions.

The size of $\mathbb{P}_{i+1|k}$ can be reduced by defining linear feedback $Le_{i|k}$ on the preceding uncertain part of the state prediction. This makes the input predictions

[1] and similarly for A^{-}, B^{+}, B^{-}, w^{+} and w^{-}

uncertain. The input predictions

$$u_{i|k} = c_{i|k} + Le_{i|k}, \quad i < N \tag{6.29}$$

are the sum of this feedback and the input degrees of freedom $c_{i|k}$, which are optimisation variables that introduce feed-forward into the predicted evolution of the state.

The purpose of feedback gain L is to attenuate the uncertainty in the prediction horizon. We calculate it as the LQR optimal feedback for the nominal dynamics. Initial predicted states $z_{0|k}$ and $e_{0|k}$ are free variables in the online optimisation; their sum is constrained to equal the measured state x_k. When decomposed, the uncertain dynamics (6.26) can be written:

$$z_{i+1|k} = A^0_k z_{i|k} + B^0_k c_{i|k} + w^0_k \tag{6.30a}$$

$$e_{i+1|k} = A^\delta_{i|k} z_{i|k} + (A^0_k + A^\delta_{i|k} + B^0_k L + B^\delta_{i|k} L) e_{i|k} + B^\delta_{i|k} c_{i|k} + w^\delta_{i|k}. \tag{6.30b}$$

This decomposition is also applied to the constraints of (6.10), giving:

$$F(z_{i|k} + e_{i|k}) + G(c_{i|k} + Le_{i|k}) \leq g(\tilde{v}_k). \tag{6.31}$$

We define the tube cross-section, $\mathbb{P}_{i|k}$, at each prediction step i, as a bounding polytope of the uncertain prediction

$$e_{i|k} \in \mathbb{P}_{i|k} = \{e : Ve \leq \alpha_{i|k}\}, \quad \alpha_{i|k} \geq 0 \tag{6.32}$$

where $V \in \mathbb{R}^{n_V \times n_x}$ is constant, each row of which is normalised and defines the direction of a facet of $\mathbb{P}_{i|k}$, while $\alpha_{i|k} \in \mathbb{R}^{n_V}$ is an optimisation variable defining the distance of each facet from the origin. For simplicity, the simulations in this chapter select V to define the faces of a cube in $\mathbb{R}^{n_x}$.

Theorem 12. *If* $Ve_{i|k} \leq \alpha_{i|k}$*, then under the dynamics of* (6.30)*,* $Ve_{i+1|k} \leq \alpha_{i+1|k}$

if and only if there exist matrices $H^+_{i|k}, H^-_{i|k}$ with non-negative elements such that

$$H^{\pm}_{i|k}V = V(A^{\pm}_{i|k} + B^{\pm}_{i|k}L) \tag{6.33a}$$

$$H^{\pm}_{i|k}\alpha_{i|k} + V(A^{\pm}_{i|k}z_{i|k} + B^{\pm}_{i|k}c_{i|k} + w^{\pm}_{i|k} - z_{i+1|k}) \leq \alpha_{i+1|k}. \tag{6.33b}$$

Proof. This is a consequence of Farkas' Lemma [Bitsoris, 1988; Hennet, 1989]. By convexity, applying this to both extremes of the uncertainty, establishes the claim of the theorem. A detailed discussion is found in Section 2.5. □

Clearly, $H^+_{i|k}$ and $H^-_{i|k}$ depend on wind speed, so we hereby write them as $H(v^{\pm}_{i|k})$. Rather than calculating (6.33a) online, we find each matrix $H(v)$ corresponding to wind speeds taken from a discrete set, and introduce small conservativeness by rounding up $v^+_{i|k}$ and rounding down $v^-_{i|k}$ to select $H^+_{i|k}$ and $H^-_{i|k}$ respectively online. The same treatment can be applied to determine necessary and sufficient conditions for (6.31). If $Ve_{i|k} \leq \alpha_{i|k}$ for $i = 1, \ldots, N$, then (6.31) is satisfied if and only if there exists matrix H_m with non-negative elements that satisfies

$$H_m V = F + GL \tag{6.34a}$$

$$H_m\alpha_{i|k} + Fz_{i|k} + Gc_{i|k} \leq g(\tilde{v}_k). \tag{6.34b}$$

Since the matrices $H(v)$ and H_m are designed offline so as to satisfy (6.33a) and (6.34a), the conditions are sufficient but not necessary, so there is a degree of conservativeness. Therefore, to relax the constraints (6.33b) and (6.34b), which are invoked online, the degrees of freedom available in the offline design of $H(v)$ and H_m are given up so as to minimise the sum of their elements.

6.9 Cost Function

We define $\beta_0(\tilde{v}_k)$ as the equilibrium pitch angle for the wind speed $\tilde{v}_k$. Its value is found numerically in Section 6.10. The target state $\bar{x}_k = [0\ 0\ \bar{\Omega}(\tilde{v}_k)^2]^T$ and target input $\bar{u}_k = [\beta_0(\tilde{v}_k)\ \bar{P}_e]^T$ are treated as time invariant, given that due to the low-pass filtering, $\tilde{v}_k$ varies sufficiently slower than the physical dynamics. We

select zero cost on tower position (and hence the first element of $\bar{x}$) because the tower movement is cyclic around a time varying offset that cannot be controlled. As such, penalising tower velocity is sufficient for damping, for which a zero offset is appropriate. The offset element $\bar{P}_e$ is used to drive the turbine to produce as much power as possible.

We minimise the expectation of the uncertain cost of the finite horizon given by (6.35), where $\tilde{x}_{i|k} = x_{i|k} - \bar{x}_k$, $\tilde{u}_{i|k} = u_{i|k} - \bar{u}_k$ and $\hat{Q}$ is a terminal weighting matrix, which extends the cost to the infinite horizon. By defining a cost function that applies in both partial load and full load, and applying input constraints, the resulting controller can operate across all wind speeds for which the turbine is designed. The cost to be minimised at time k is:

$$J_k = \mathrm{E}_k \left(\tilde{x}_{N|k}^T \hat{Q} \tilde{x}_{N|k} + \sum_{i=0}^{N-1} \tilde{x}_{i|k}^T Q \tilde{x}_{i|k} + \tilde{u}_{i|k}^T R \tilde{u}_{i|k} \right) \tag{6.35}$$

Writing this cost in terms of the decision variables is facilitated by a lifted representation $\chi_{i+1|k} = \Psi_{k+i} \chi_{i|k}$, where

$$\chi_{i|k} = \begin{bmatrix} z_{i|k} \\ \mathbf{c}_{i|k} \\ e_{i|k} \\ 1 \end{bmatrix}, \quad \Psi_{k+i} = \begin{bmatrix} A^0 & B^0 E & 0 & w^0 \\ 0 & M & 0 & 0 \\ A^\delta_{i|k} & B^\delta_{i|k} E & A_{i|k} + B_{i|k} L & w^\delta_{i|k} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.36}$$

and where $\mathbf{c}_{0|k} := \begin{bmatrix} c_{0|k}^T & \cdots & c_{N-1|k}^T \end{bmatrix}^T$, M is the zero matrix with identity matrices on its super-diagonal blocks and $E = \begin{bmatrix} I & 0 & \cdots & 0 \end{bmatrix}$, so that

$$M \mathbf{c}_{0|k} = \begin{bmatrix} c_{1|k}^T & \cdots & c_{N-1|k}^T & 0 \end{bmatrix}^T, \quad E \mathbf{c}_{i|k} = c_{i|k}.$$

The stage cost can now be written $\mathrm{E}_k(\chi_{i|k}^T \bar{Q} \chi_{i|k})$, with

$$\bar{Q} = \begin{bmatrix} Q & 0 & Q & -Q\bar{x} \\ * & E^T R E & E^T R L & -E^T R \bar{u} \\ * & * & Q + L^T R L & -L^T R \bar{u} - Q \bar{x} \\ * & * & * & \bar{x}^T Q \bar{x} + \bar{u}^T R \bar{u} \end{bmatrix}. \tag{6.37}$$

As proven in Section 3.4.5, the cost of (6.35) is equal to $\chi_{0|k}^T P \chi_{0|k}$, for a positive symmetric P that satisfies:

$$P - \mathrm{E}_k(\Psi_{k+i}^T P \Psi_{k+i}) = \bar{Q}, \qquad i \in \mathbb{Z}. \tag{6.38}$$

6.10 Control Model Identification

Numerical values for control model parameters are found in the aeroelastic simulation environment by analysing the time series in a similar manner to [Friis et al., 2011], i.e. step response analysis and linear regression performed on the output of a high-fidelity aeroelastic wind turbine simulation environment. The difference in this chapter is that the model identification is performed at several mean wind speeds to give an LPV model. The identification methods employed in this chapter do not require knowledge of turbine properties such as mass, stiffness or shape. Only data that it is possible to measure on a real turbine is included in the regression, which aids the applicability of the techniques presented.

The turbine definition used in the aeroelastic code in this chapter has a 112 m diameter rotor and has a multi-megawatt generator. At the time of producing the results in this chapter, it was the most realistic model available that represents a specific wind turbine currently under commercial development. From (6.8), with appropriate simplifications, we require the following sub-parameters, denoted FA, β, P for fore-aft, pitch and power.

$$x_{k+1} = \left[\begin{array}{cc|c} & & 0 \\ & A_{FA}(v_{i|k}) & 0 \\ \hline 0 & 0 & 1 \end{array}\right] x_k + \left[\begin{array}{c|c} B_{FA}(v_k) & 0 \\ \hline B_\beta(v_k) & B_P \end{array}\right] u_k + \left[\begin{array}{c} w_{FA}(v_k) \\ \\ w_\beta(v_k) \end{array}\right] \tag{6.39}$$

Closed loop model identification is used, with a PI controller defining the control inputs (β, P_e) on the basis of measurements of Ω^2. Inputs to the plant can be perturbed or switched to force certain behaviours or explore larger areas of the input-output space. We use the term 'freezing' to denote holding a control input constant to study the resulting behaviour.

Throughout this section we make the simplifying assumption that the identific-

ation methods will use appropriately filtered variables without separate notation. A reliable measurement of tower displacement requires a high-order low-pass filter to remove effects from blade vibration and higher order tower modes. The cut-off frequency, 4 Hz, is chosen to lie between the natural frequencies of the tower and the blades.

Step response identification

As shown in (6.6), we approximate the rate of change of rotor speed squared with respect to electrical power as independent of wind speed. We further assume that the rotor speed error is small during operation, so we need only identify B_P, which is straightforward. Set the turbulence intensity in the simulation environment to 0%. Set a switch time t_s, long enough for the turbine to reach steady state, e.g. 60 s. Cause the pitch to freeze at time t_s. Cause the electrical power to step up by Δ_P from its pre-step value of P_s at time t_s. Measure Δ_{Ω^2}, the rate of change of Ω^2 after the power step. Set $B_P = \Delta_{\Omega^2}/\Delta_P$.

Finding the change in Ω^2 due to pitch is more involved as it depends on wind speed. A separate identification for each wind speed is required to find $\hat{B}_\beta(v), \hat{w}_\beta(v)$ for each $v \in \mathbb{V}$. The functions $B_\beta(v), w_\beta(v)$ for $v \notin \mathbb{V}$ are linear interpolations of $\hat{B}_\beta(v), \hat{w}_\beta(v)$.

The step response identification method requires the turbulence intensity to be set to 0%, then the following calculations are performed for each wind speed $v \in \mathbb{V}$. Set switch time t_s. Freeze the power at t_s. Freeze the pitch at t_s and record its value as $\beta_0(v)$. Increase the pitch by a small amount. Measure $\hat{B}_\beta(v)$, defined as the rate of change of Ω^2. Then $\hat{w}_\beta(v) = -\beta_0 \hat{B}_\beta(v) - P_s B_P$.

Regression identification

The model parameters of the tower dynamics are found by running simulations of 600 s and fitting a white box affine model using least squares regression. The independent values of the regression are vector time series of tower position $\mathbf{p}$, velocity $\dot{\mathbf{p}}$ and pitch angle $\mathbf{b}$. The dependent values are vector time series of tower position and velocity shifted by the MPC sample time to give two more

sequences, $\mathbf{p}^+, \dot{\mathbf{p}}^+$. This identification is required for each $v \in \mathbb{V}$, interpolated to give $A_{FA}(v), B_{FA}(v), w_{FA}(v)$.

The regression identification method requires the turbulence intensity to be set to a relatively low value above zero – we have used 10%. Then the following calculations are performed for each wind speed $v \in \mathbb{V}$. Simulate 600 s of normal operation with a given PI controller. Set $X = [\mathbf{p} \quad \dot{\mathbf{p}} \quad \mathbf{b} \quad \mathbf{1}]$. Calculate $Y = [X^\dagger \mathbf{p}^+ \quad X^\dagger \dot{\mathbf{p}}^+]$, where $X^\dagger$ is the pseudo-inverse of X corresponding to least squares regression, resulting in Y, the best fit one-step-ahead prediction model for the purpose of predicting tower dynamics. Finally, set:

$$[\hat{A}_{FA}(v) \quad \hat{B}_{FA}(v) \quad \hat{w}_{FA}(v)] = Y^T. \tag{6.40}$$

To measure how well the data points fit the model that has been identified, the coefficient of determination $\mathcal{R}^2$ has been calculated for the fits of tower position and velocity. The definition of the coefficient of determination for the tower position is:

$$\mathcal{R}^2 = 1 - \frac{\left\| XX^\dagger \mathbf{p}^+ - \mathbf{p}^+ \right\|}{\|\mathbf{p}^+\|}. \tag{6.41}$$

The coefficients of determination for the regressions at 12 m/s mean wind speed (slightly above rated wind speed, so the most difficult to fit) were 0.99 for tower displacement and 0.91 for tower velocity. This indicates that there is more unmodelled behaviour in the tower velocity than in the tower displacement.

6.11 Online Rainflow Counting

The principles of rainflow counting are described in Section 6.3 but it is desirable to have a rainflow count that can operate online. In [Downing and Socie, 1982], as extrema in the stress signal are measured, which alternate between maxima and minima, they are added to a stack. Whenever this stack has more than one value stored after a new one is added, a series of conditions are checked to ascertain whether the stack contains any cycles. If valid cycles are found, the values are removed from the stack and added to a cycles stack. Large amounts of memory are required for such an online algorithm when run over long periods of time.

An alternative online rainflow counting algorithm is given in [Amzallag et al., 1994], called the four-point method. This operates similarly to the previous algorithm, except the extremum stack builds up to four values, s_1, s_2, s_3, s_4, whence a reduction process occurs. The absolute difference between each of these is denoted $\Delta S_1 := |s_2 - s_1|$ and so on, as shown in Figure 6.9. Now, if $\Delta S_1 > \Delta S_2 \leq \Delta S_3$ then it is an inner cycle and is removed, saving its magnitude on the cycles stack. Otherwise, the first element of the extremum stack is moved to a 'residue' stack to be dealt with offline, freeing up one more value in the four-point window. The downside of this process is the residue can build up considerably over time.

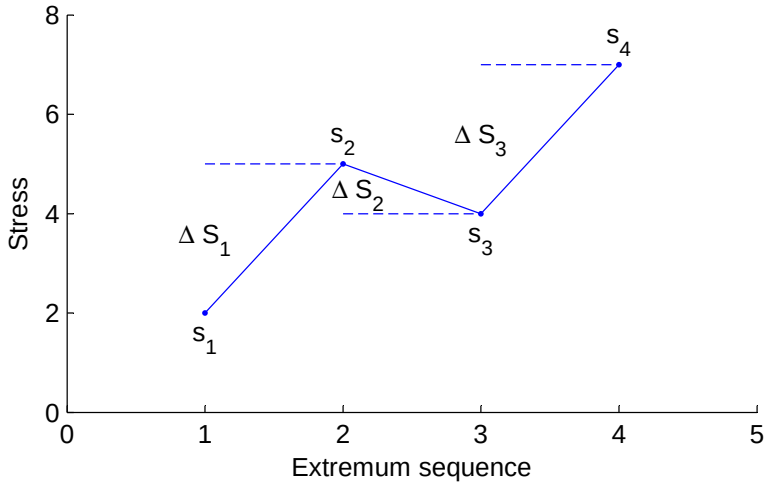

Figure 6.9: The four-point method of online rainflow counting

The simulations in this chapter apply the four-point method with an approximation to the Palmgen-Miner rule to avoid the need for a residue stack and hence require a fixed amount of memory to operate online. When the first value is dropped off the front of the extrema stack, the range ΔS_1 is counted as a half-cycle c_j. The accumulated fatigue is then approximated as:

$$D \approx \frac{1}{K} \sum_i s_i^m + \frac{1}{2K} \sum_j c_j^m. \tag{6.42}$$

6.12 Example Results

The numerical example of this section comprises 84 aeroelastic simulations controlled with each of MPC and RMPC. The MPC controllers operate sample-and-hold at 2 Hz and output reference pitch and power set-points to dedicated pitch and power controllers that lie inside the aeroelastic code. We use a prediction horizon of $N = 8$, which at 2 Hz is longer than one tower cycle. This leads to a QP of 16 variables and 40 inequality constraints in the nominal controller and 64 variables and 226 inequality constraints in the robust controller. All inputs and outputs are scaled to operate around unity for numerical conditioning purposes. All simulations use the same parameters $N, F, G, Q, \hat{Q}, R$ and all control model parameters. All RMPC simulations use the same parameters $V, L, \tau = 0.84, n_c = 2.5$.

Twelve random wind fields are generated per mean wind speed, six at a turbulence intensity of 16% and six at 14%, as per the IEC Standard [IEC, 2005] higher and medium turbulence classes. Figure 6.10 shows what effect robustness has on tower fatigue, mean power, pitch travel and rotor speed error.

The results show that RMPC reduces tower fatigue, pitch travel and rotor speed error, while sacrificing a small amount of power. The effect is more pronounced at lower mean wind speeds, where the pitch constraint is active more often and the mean thrust is higher. To estimate the total reduction in fatigue that a wind turbine would receive over a typical year, we weight the reductions by a typical annual wind speed distribution. In [Burton et al., 2011], the Weibull distribution with shape factor $k = 2$ is suggested. The IEC standard Class 1b defines the scale factor (mean wind speed) as $U = 10\,\mathrm{m/s}$. The Weibull distribution is the probability density function over wind speed v defined as:

$$f(v; U, k) = \frac{k}{U} \left(\frac{v}{U}\right)^{k-1} e^{-(v/U)^k}. \tag{6.43}$$

The Weibull analysis suggests that over a typical year, RMPC would save 3.2% tower fatigue at the expense of 0.4% of power. The benefit of RMPC is shown to be more significant at mean wind speeds where the pitch constraints are active often. The uncertainty feedback term L makes predicted inputs uncertain, which allows RMPC to anticipate input constraint activity due to potential fu-

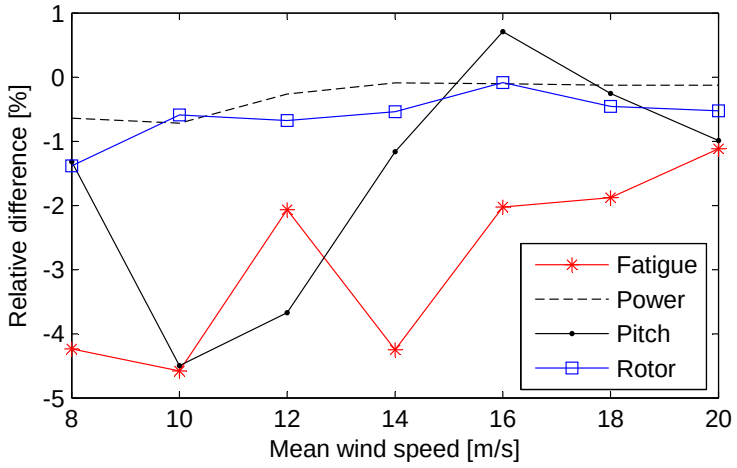

Figure 6.10: Relative difference of RMPC compared to MPC for four measures across a range of mean wind speeds. Each mean wind speed has 12 runs of 600 s with different random turbulence fields. Relative difference is the measure for RMPC divided by the measure for MPC, minus one, expressed as a percentage. Fatigue is the accumulated tower fatigue with Wöhler exponent 4. Power is the mean power output. Pitch is the total pitch travel, i.e. the sum of the absolute differences in pitch in successive time steps. Rotor is the RMS rotor speed error.

ture wind speed variations. We illustrate this effect with a deterministic load case. Figure 6.11 shows the wind speed undergoing a sudden drop (an artificially constructed deterministic wind case). Under nominal MPC, the pitch constraint saturates because $\tilde{v}$ lags v. Then there is no pitch actuation available to dampen the tower oscillations. Under Robust MPC, v lies within the uncertainty bounds of future wind speeds and predicted pitch angles hit tightened constraints earlier in time. The result is superior damping.

It may be useful to see examples of the controllers running with realistic wind cases. Caution should be taken here not to draw firm conclusions from a single simulation, but the comparison is given to illustrated some differences.

Figure 6.12 shows the tower base moment in the fore-aft direction, as experienced under three controllers: a gain scheduled proportional-plus-integral (PI) controller and the two controllers presented in this chapter, namely MPC and RMPC. The PI controller has no tower damping – it optimises the power capture

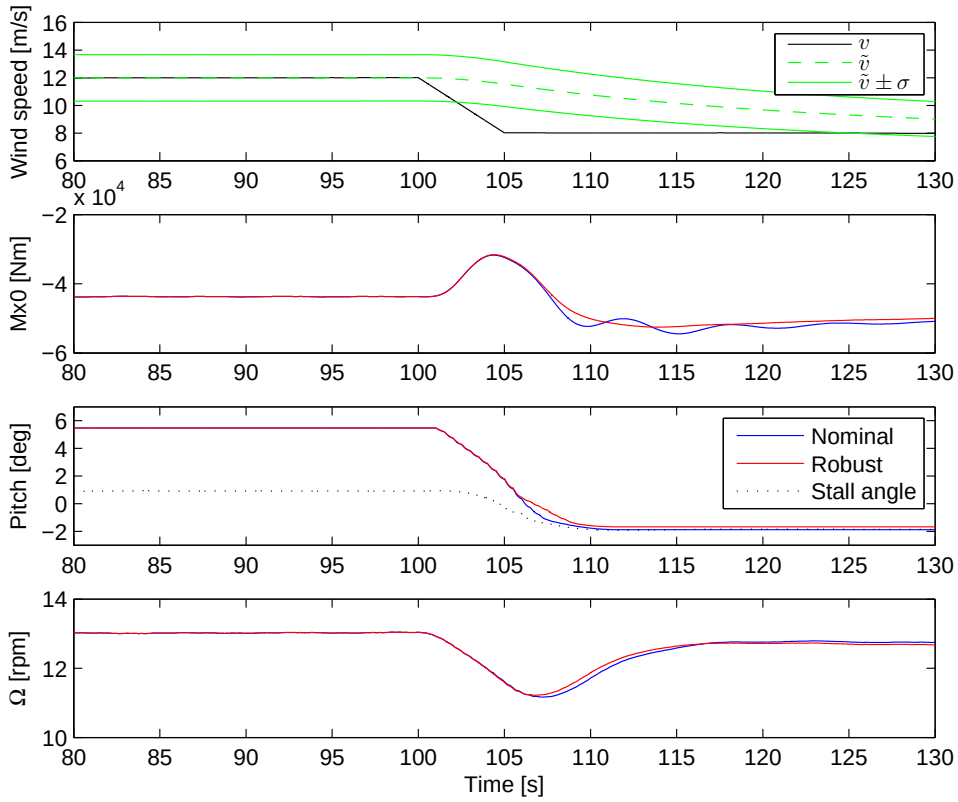

Figure 6.11: A deterministic load case. Wind speed v drops from 12 m/s to 8 m/s in 5 s. The tower base bending moment (Mx0) is more damped under RMPC than under MPC. Action to avoid pitch saturation is taken earlier under RMPC, visible at 106 s. The rotor speed Ω recovers more quickly under RMPC.

within limits by varying the pitch angle and generator power demand. It has switch logic to handle saturations imposed by the constraints. Broadly, the largest tower oscillations are seen after sudden changes in wind speed. The lower plot shows the frequency content of the tower base bending moment. MPC clearly performs better than PI at tower damping across all frequencies. RMPC performs noticeably better than MPC at frequencies above 0.2 Hz. Notably, MPC and RMPC perform equally at the tower natural frequency of 0.15 Hz.

Figure 6.13 shows the pitch activity experienced in the same simulations. The PI controller is much more intensive in pitch activity than MPC and RMPC. It aggressively seeks the optimal power capture when the wind speed is around rated wind speed. This is most noticeable between 350 s and 500 s. RMPC is more conservative than MPC. Examples to illustrate this difference are found between 300 s and 400 s, where RMPC does not pitch as aggressively (Figure 6.14 shows the implications this has for power capture). The lower graph shows the frequency content of the pitch activity. MPC clearly has less pitch activity than PI across all frequencies. RMPC has slightly less pitch activity than MPC across frequencies 0.15 Hz to 0.3 Hz.

Figure 6.14 shows the generator power produced in the same simulations. MPC clearly captures noticeably less power than PI for much of the time. RMPC captures slightly less power than MPC for some of the time, e.g. around 350 s. The difference in power capture illustrated in this simulation only occurs around rated wind speed. When the pitch system is saturated, the generator torque has full control over the rotor speed, which results in all controllers producing the same average power. Similarly when the power is saturated, clearly all controllers would also produce the same average power.

6.13 Conclusions

Wind turbines have many degrees of freedom, or states: rotor speed and azimuth, the pitch angle, flap-wise deflection and twist of each blade, fore-aft and side-side tower bending moments, and states associated with higher bending modes. Many of these states interact with other states, for example the coupling between flap-wise blade moments and fore-aft tower bending. Identifying a numerical model to

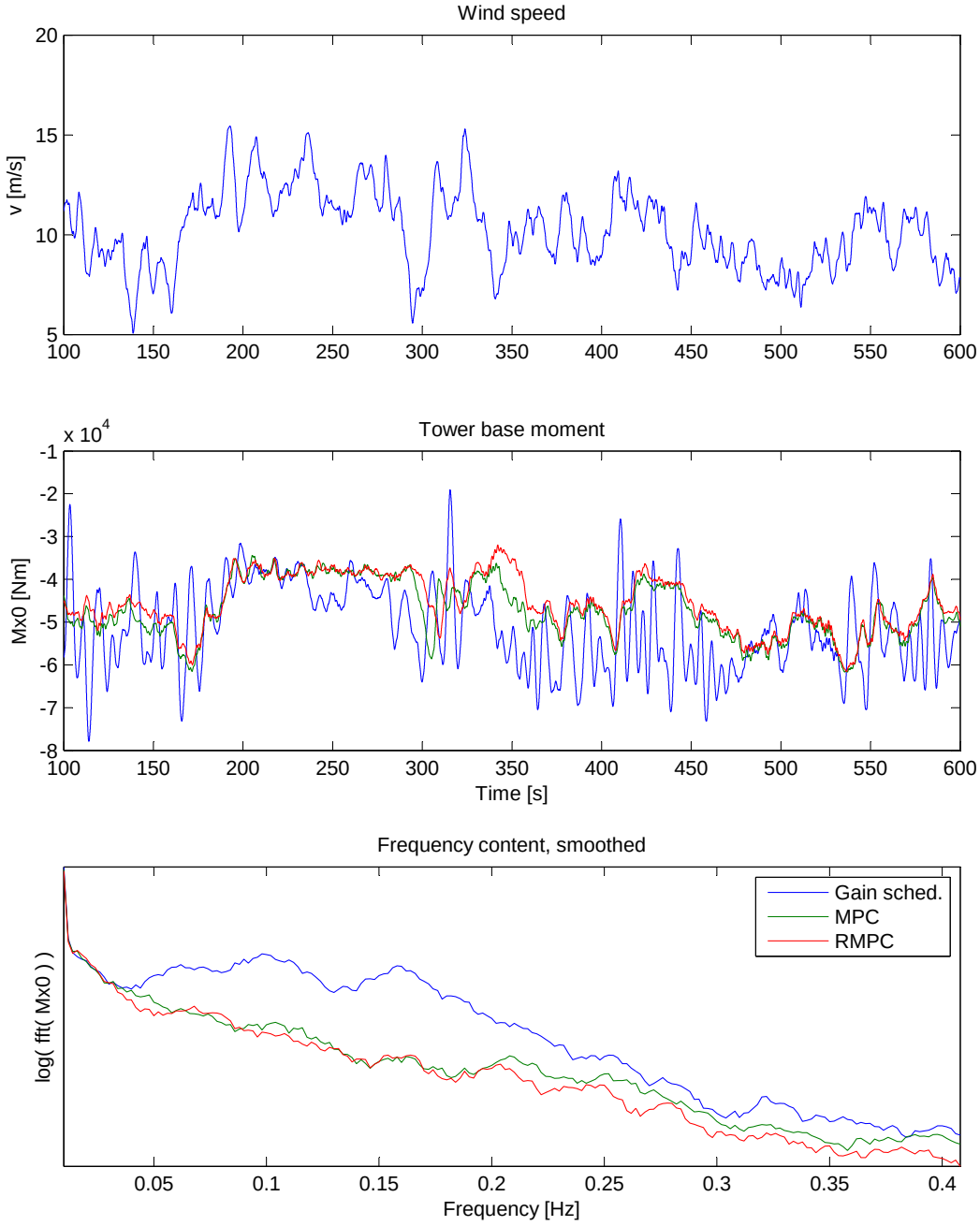

Figure 6.12: Comparison of the tower damping performance of three controllers: a gain scheduled PI controller, nominal MPC and RMPC. The simulation was run for 600 seconds for each controller; the first 100 s are not included as they are affected by initial conditions. The wind conditions were identical for each simulation. The top plot shows nacelle anemometer measured wind speed. The bottom plot shows the Fourier transform of the time series of the middle plot, smoothed with a box filter of bandwidth 0.016 Hz.

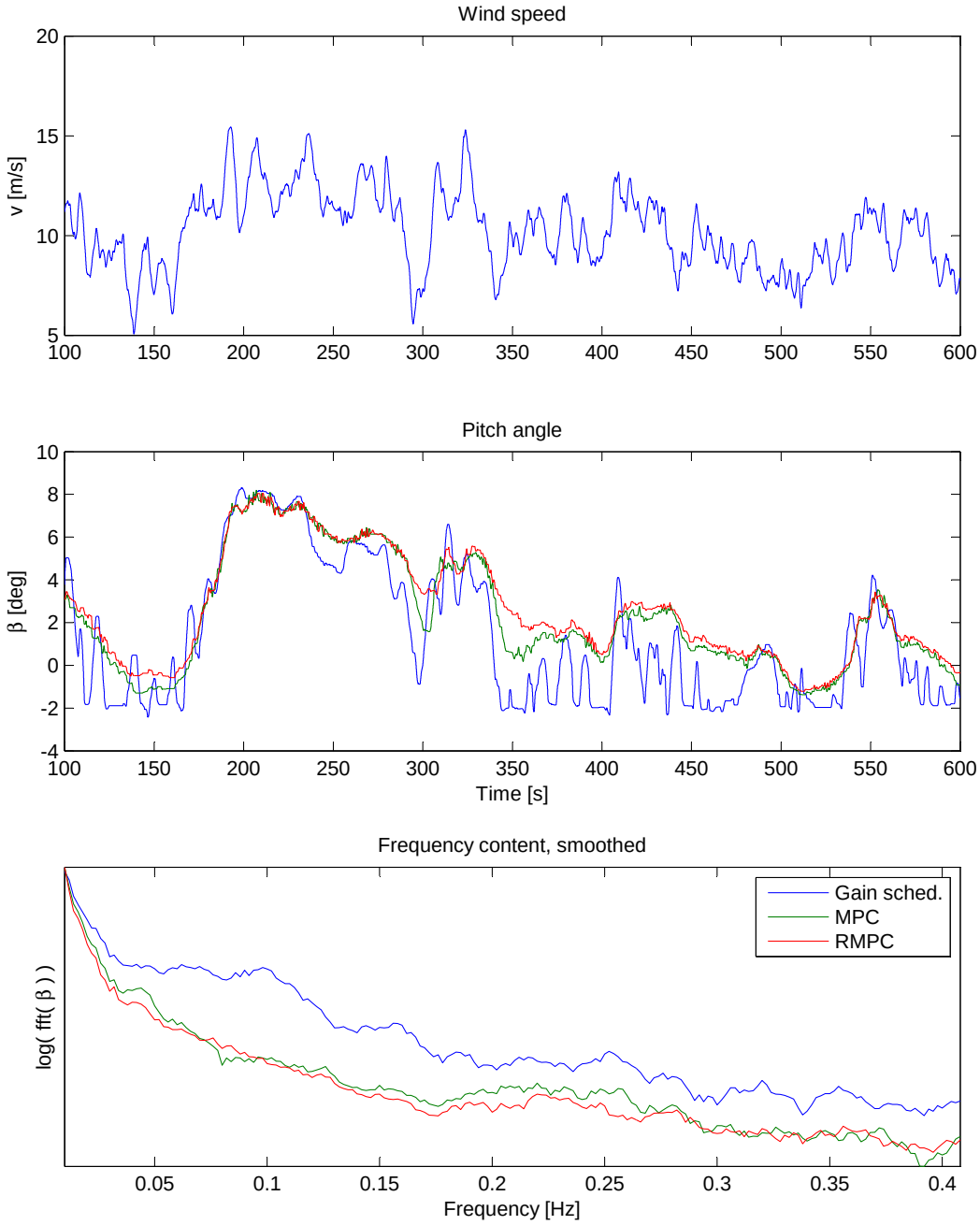

Figure 6.13: Comparison of the pitch activity (middle plot) of the same three controllers in the same simulations as in Figure 6.12. The bottom plot shows the Fourier transform of the time series of the middle plot, smoothed with a box filter of bandwidth 0.016 Hz.

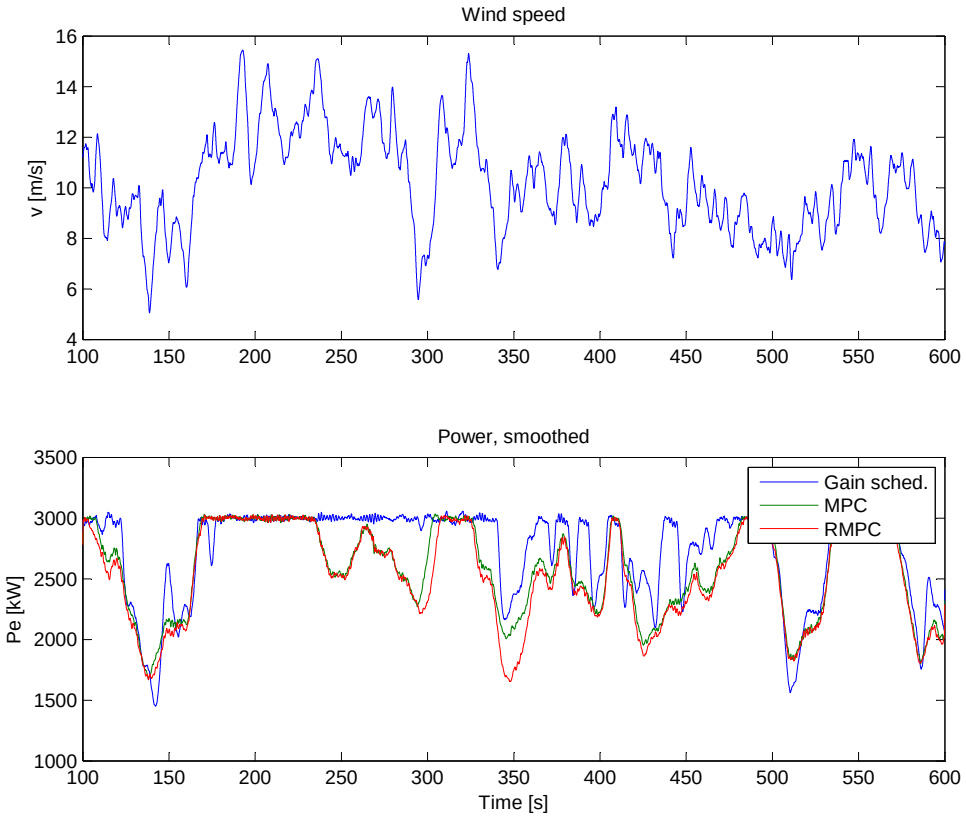

Figure 6.14: Comparison of the generator power produced under the same three controllers in the same simulations as in Figure 6.12.

represent all of this complex nonlinear behaviour is a huge task. The approach taken in this chapter has instead been to model the wind turbine by just three states, where the remaining behaviour is embodied in the model uncertainty. As far as the author knows, this is the first work to apply RMPC with multiplicative uncertainty to a wind turbine.

The main benefit that RMPC brings to this application over nominal MPC is in transient responses to constraint activity. The 'tube' of possible future states and inputs hits the constraints many time steps earlier than the nominal sequence it surrounds. This changes the solution to the constrained optimisation with enough time to actuate the pitch system to avoid the constraint being violated. Without considering the uncertainty in the model, when that uncertainty is realised, the

controller has to take more severe action to avoid a more imminent constraint.

The estimated tower fatigue saving of 3.2% under RMPC compared to nominal MPC is significant, though not huge. This is in part due to the closed-loop effect in MPC: it does not matter much that state predictions far into the horizon may be inaccurate because only the first predicted input is used. This chapter shows that if it's worth replacing PID control with MPC then it's worth taking the extra step to implement Robust MPC.

Furthermore, the trade-off between lower tower bending moments, pitch activity and total power capture is a decision for the wind turbine developer to make. The formulation of the MPC and RMPC controllers in this chapter makes this trade-off simple to investigate and manage. The cost function and prediction horizon length are easy to alter and the results from simulations such as this are easy to interpret using simple data visualisation methods or industrial bespoke software.

The decision not to use a terminal set in this chapter was taken because this particular application has a unique challenge with respect to autonomous control, solutions to which lie far outside the scope of this thesis. An autonomous control law would be required that, when applied to states inside the terminal set, produces successor states also inside that same set. A control law that operates in all wind speeds, both above and below rated, would need to be nonlinear.

Another challenge for the application of discrete-time RMPC to a continuous-time system is a consequence of shorter sample times. A motivation to increase the sample frequency might be to reduce the extent to which uncertainty builds up between measurements. A higher sample rate allows the controller to react to these measurements after less uncertainty has accrued because less can change in an unpredictable way in a shorter length of time. However, short sample times create a problem in the calculation of the terminal set. The eigenvalues of the A matrix tend to unity, which requires a polytope of many facets to be invariant.

Extensions to this chapter's contributions could be: increasing the state space to include more of the turbine degrees of freedom; and reducing conservativeness by introducing probabilistic constraints. Suggested new states are tower side-side bending and blade flap-wise bending. Extending the input space to control each pitch angle separately could also prove beneficial, providing the rotor azimuth

is also incorporated into the state. Applying SMPC will require more advanced model identification, whereby probability distributions are identified alongside the expected dynamic model.

Chapter 7

Conclusions and Further Work

Conclusions

Bridging the gap between theory and application in engineering can be a significant challenge. Specialised tools and techniques must be developed in order to produce useful results. Applying a control scheme without adapting it for the given application may violate its assumptions, which can result in unexpected closed-loop behaviour. Making controller adaptations that are too specific amounts to a complex tuning exercise, from which future researchers are unlikely to benefit. This thesis has developed the theory of Robust and Stochastic MPC in such a way that it can be applied more widely than before, and has demonstrated the steps required to make the controller work in a realistic environment.

Multiplicative (or process) uncertainty is more difficult to handle in MPC than additive (or disturbance) uncertainty. In all MPC algorithms, constraints are imposed on optimisation variables at every time step. In nominal MPC, in which the model assumes no uncertainty, constructing those constraints is trivially easy and the number of them applied to the optimisation problem is relatively small. When we account for additive uncertainty, parameters can be calculated offline, before the controller is run in closed-loop with the system. Those parameters represent the amount by which the constraints must be tightened. The number of constraints is then equal to that of the nominal case. No such offline calculation is possible when we account for multiplicative uncertainty.

Accounting for every possible combination of extreme values of uncertain model

parameters produces a set of constraints that scales exponentially with the length of the horizon. This may be acceptable for short horizons but commonly there are design considerations that favour longer horizons. In this thesis, polytopes were constructed with facets that have a fixed orientation but a variable distance from the origin. Online, constraints are applied to the polytope facet variables to ensure that any predicted state inside such a polytope will lie inside the subsequent polytope at the next prediction step. A sequence of such polytopes with this property forms an invariant polytopic 'tube' around the nominal predicted state sequence. Now the number of constraints scales linearly with the horizon length, at the expense of a small amount of conservativeness.

Conservativeness can be reduced if a constraint may be violated with a prescribed probability. Calculating the probability of violation of a constraint several prediction steps subsequent to the measured time step is an intractable problem, so sampling techniques are employed. Predictions formed from selected samples of the uncertainty distribution are allowed to violate the constraints, while the proportion of all samples that do so is constrained.

Unlike the majority of research into Stochastic MPC, this thesis has focussed on one-step ahead sampling, whereby, at each prediction time step on a horizon, which are constrained robustly, samples are used to represent a range of possible evolutions of the state over one additional time step. The resulting probabilistic statement is that no matter by how much a constraint is violated when uncertainty is realised, a solution exists to the same probabilistic problem as applied in the previous time step, i.e. the probability of violation is no more than the given constant. This guarantee is a form of stochastic recursive feasibility.

In MPC with a finite horizon, a terminal set provides a means to ensure stability. The predicted states in the finite horizon depend on optimisation variables, with the final predicted state constrained to lie in a positively invariant set, which entirely satisfies the state and input constraints. In the case of Robust MPC, the terminal set is robustly positively invariant. This thesis has contributed an algorithm to construct a terminal set tailored for a Stochastic MPC controller that exploits the probabilistic constraints. The set is robustly positively invariant, but an element therein may violate a probabilistic constraint provided its successor state satisfies the constraints with the required probability.

Dual-mode MPC, that is the combination of a finite prediction horizon with control degrees of freedom and an infinite horizon with no degrees of freedom, constitutes a controller that considers predictions over the infinite horizon with finite degrees of freedom. An alternative to dual-mode MPC is Exponential Basis Function MPC (EBF-MPC). In this approach, a small number of control degrees of freedom act over the infinite horizon by weighting a finite set of basis functions that all tend to zero as time tends to infinity. This thesis has combined EBF-MPC with the polytopic tube paradigm to construct a feasible region in the combined space of the variables that define the tube facets and the variables that define the input sequence. The resulting controller handles additive and multiplicative uncertainty, is recursively feasible and has computational complexity that scales linearly in the number of basis functions.

Armed with novel theoretical extensions, this thesis has tackled the challenge of controlling a wind turbine with Robust MPC. The RMPC algorithm controls the fore-aft tower-top displacement and velocity and the rotor speed, with demand pitch angle and demand generator power as inputs to the model. To allay concerns that the computational demand would be too great to be applicable, the sample rate was chosen to be just two hertz. Between the predictive controller time steps, specialised simple feedback controllers regulate the actual pitch position and generator torque. Uncertainty in future wind speeds and model-reality mismatches are represented as multiplicative and additive uncertainty, which are handled by the polytopic tube constraints.

Due to the expense of large wind turbines, all data is simulated. However, the simulation is state-of-the-art and represents as realistically as is practically possible all the interactions of turbulence, wind shear, lift, drag, bending, twisting and shearing in a multi-body model. The tower damping capability of the new RMPC controller was compared to that of the nominal MPC controller. RMPC is shown to reduce tower base fatigue and pitch activity at the expense of a small reduction in energy capture. Ultimately, the techniques employed in this thesis to ensure successful controller synthesis may serve as a demonstration to industry of the range of expertise required to apply MPC to wind turbine control.

Further Work

This thesis creates several, cheifly applied, opportunities for further research.

A Robust MPC algorithm is only recursively feasible if the assumptions are valid. The uncertainty distributions in a wind turbine model at low sample rate have long tails, which in this thesis are curtailed to a fixed number of standard deviations. It is possible that with a higher sample rate, appropriate bounds could be used for the uncertainty such that the probability of the realised uncertainty exceeding those bounds is negligibly small without causing the controller to act overly conservatively. A higher sample rate will require more prediction steps and consequently more optimisation variables. Creating a suitable algorithm to solve the more complex problem within a shorter sampling interval will be a significant challenge.

The applied RMPC algorithm of this thesis did not employ a terminal set. The primary reason for this was that there was no constant-gain autonomous control law that could stabilise the model across the entire range of the scheduling parameter. The probability distribution of the linearised dynamics depends on the scheduling parameter – the linearised model at low wind speeds is very different from that at high wind speeds. Research is needed into terminal sets that are robustly positively invariant under a nonlinear autonomous control and ways to guarantee recursive feasibility when Mode 1 employs linear models and Mode 2 employs nonlinear feedback.

Further theoretical research is also needed, including a detailed comparison of the relative merits of dual-mode versus EBF-MPC. The field would benefit from a study into the closed-loop cost of controlling a system under each, while applying a range of computational constraints. For example which technique can demonstrate, when carefully tuned, to control a given three-state system at the lower closed-loop cost, if both must apply fewer than two hundred linear constraints online? Does the same technique perform the better if the model has eight states and a much higher number of constraints are allowed?

Finally, this research would be enriched by the development of Exponential Basis Function Stochastic MPC. A controller that could handle multiplicative and additive uncertainty in the infinite horizon while satisfying constraints with a given

probability, in a form that is readily applicable without excessive tuning parameters, would be of great interest in industry. Combined with a study into the trade-off between closed-loop cost and computational cost, such work would give SMPC 'off the shelf' status, as nominal MPC achieved at the start of the century.

Bibliography

Allwright, J. and Papavasiliou, G. (1992). On linear programming and robust model-predictive control using impulse-responses. *Systems & Control Letters*, 18(2):159–164.

Amzallag, C., Gerey, J., Robert, J., and Bahuaud, J. (1994). Standardization of the rainflow counting method for fatigue analysis. *International Journal of Fatigue*, 16(4):287–293.

Barmish, B. and Sankaran, J. (1979). The propagation of parametric uncertainty via polytopes. *IEEE Transactions on Automatic Control*, 24(2):346–349.

Batina, I. (2004). *Model predictive control for stochastic systems by randomized algorithms*. PhD thesis, Technische Universiteit Eindhoven.

Beaupuits, J., Otárola, A., Rantakyrö, F., Rivera, R., Radford, S., and Nyman, L.-Å. (2004). ALMA Memo #497 – Analysis of wind data gathered at Chajnantor. Technical report, NRAO.

Bitsoris, G. (1988). On the positive invariance of polyhedral sets for discrete-time systems. *Systems & Control Letters*, 11(3):243–248.

Blackmore, L., Ono, M., Bektassov, A., and Williams, B. C. (2010). A probabilistic particle-control approximation of chance-constrained stochastic predictive control. *IEEE Transactions on Robotics*, 26(3):502–517.

Bossanyi, E. (2003). Wind turbine control for load reduction. *Wind Energy*, 6(3):229–244.

Boyd, S., Ghaoui, L. E., Feron, E., and Balakrishnan, V. (1994). *Linear Matrix Inequalities in System and Control Theory*. Society for Industrial and Applied Mathematics.

Burton, T., Jenkins, N., Sharpe, D., and Bossanyi, E. (2011). *Wind Energy Handbook*. John Wiley & Sons.

Cagienard, R., Grieder, P., Kerrigan, E. C., and Morari, M. (2007). Move blocking strategies in receding horizon control. *Journal of Process Control*, 17(6):563–570.

Calafiore, G. and Campi, M. (2006). The scenario approach to robust control design. *IEEE Transactions on Automatic Control*, 51(5):742–753.

Calafiore, G., Dabbene, F., and Tempo, R. (2003). Randomized algorithms in robust control. In *IEEE Conference on Decision and Control*, volume 2, pages 1908–1913. IEEE.

Campi, M. and Garatti, S. (2011). A sampling-and-discarding approach to chance-constrained optimization: Feasibility and optimality. *Journal of Optimization Theory and Applications*, 148:257–280.

Campo, P. and Morari, M. (1987). Robust model predictive control. In *American Control Conference*, pages 1021–1026. IEEE.

Cannon, M. and Kouvaritakis, B. (2000). Infinite horizon predictive control of constrained continuous-time linear systems. *Automatica*, 36(7):943–955.

Cannon, M. and Kouvaritakis, B. (2005). Optimizing prediction dynamics for robust MPC. *IEEE Transactions on Automatic Control*, 50(11):1892–1897.

Cannon, M., Kouvaritakis, B., and Wu, X. (2009). Probabilistic constrained MPC for multiplicative and additive stochastic uncertainty. *IEEE Transactions on Automatic Control*, 54(7):1626–1632.

Colwell, S. and Basu, B. (2009). Tuned liquid column dampers in offshore wind turbines for structural control. *Engineering Structures*, 31(2):358–368.

Downing, S. and Socie, D. (1982). Simple rainflow counting algorithms. *International Journal of Fatigue*, 4(1):31–40.

Evans, M., Cannon, M., and Kouvaritakis, B. (2014). Robust MPC tower damping for variable speed wind turbines. *IEEE Transactions on Control Systems Technology*, PP(99).

Fleming, J., Kouvaritakis, B., and Cannon, M. (2013). Regions of attraction and recursive feasibility in robust MPC. In *Mediterranean Conference on Control & Automation*, pages 801–806. IEEE.

Friis, J., Nielsen, E., Bonding, J., Adegas, F. D., Stoustrup, J., and Odgaard, P. F. (2011). Repetitive model predictive approach to individual pitch control of wind turbines. In *IEEE Conference on Decision and Control*, pages 3664–3670. IEEE.

Gilbert, E. and Tan, K. (1991). Linear systems with state and control constraints: The theory and application of maximal output admissible sets. *IEEE Transactions on Automatic Control*, 36(9):1008–1020.

Gurobi Optimization, I. (2014). Gurobi optimizer reference manual.

Hammerum, K. (2006). A fatigue approach to wind turbine control. Master's thesis, Technical University of Denmark.

Hardesty, R. and Weber, B. (1987). Lidar measurement of turbulence encountered by horizontal-axis wind turbine. *Journal of Atmospheric and Oceanic Technology*, 4:191–203.

Hennet, J.-C. (1989). Une extension du lemme de farkas et son application au problème de régulation linéaire sous contraintes. *CR Acad. Sci. Paris*, 308(I).

Henriksen, L., Hansen, M., and Poulsen, N. (2012). Wind turbine control with constraint handling: a model predictive control approach. *Control Theory Applications*, 6(11):1722–1734.

IEC (2005). IEC 61400-1. *International Electrotechnical Committee: Design requirements*.

IEC (2009). IEC 61400-3. *International Electrotechnical Committee: Design requirements for offshore wind turbines*.

Kalman, R. (1960). Contributions to the theory of optimal control. *Bol. Soc. Mat. Mexicana*, 5(2):102–119.

Kani, S. P. and Ardehali, M. (2011). Very short-term wind speed prediction: A new artificial neural network–Markov chain model. *Energy Conversion and Management*, 52(1):738–745.

Khan, B. and Rossiter, J. (2013). Alternative parameterisation within predictive control: a systematic selection. *International Journal of Control*, 86(8):1397–1409.

Koerber, A. and King, R. (2013). Combined feedback–feedforward control of wind turbines using state-constrained model predictive control. *IEEE Transactions on Control Systems Technology*, 21(4):1117–1128.

Kouvaritakis, B., Cannon, M., Raković, S. V., and Cheng, Q. (2010). Explicit use of probabilistic distributions in linear predictive control. *Automatica*, 46(10):1719–1724.

Kristensen, L. and Jensen, N. (1979). Lateral coherence in isotropic turbulence and in the natural wind. *Boundary-Layer Meteorology*, 17(3):353–373.

Kumar, A. and Stol, K. (2009). Scheduled model predictive control of a wind turbine. In *47th AIAA Aerospace Sciences Meeting*.

Kusiak, A., Song, Z., and Zheng, H. (2009). Anticipatory control of wind turbines with data-driven predictive models. *IEEE Transactions on Energy Conversion*, 24(3):766–774.

Löfberg, J. (2004). YALMIP: A toolbox for modeling and optimization in MATLAB. In *CACSD Conference*, Taipei, Taiwan.

Matsuishi, M. and Endo, T. (1968). Fatigue of metals subjected to varying stress. *Japan Society of Mechanical Engineers*, pages 37–40.

Mayne, D., Rawlings, J., Rao, C., and Scokaert, P. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814.

Mirzaei, M., Poulsen, N., and Niemann, H. (2012). Robust model predictive control of a wind turbine. In *American Control Conference*, pages 4393–4398.

Nanayakkara, N., Nakamura, M., and Hatazaki, H. (1997). Predictive control of wind turbines in small power systems at high turbulent wind speeds. *Control Engineering Practice*, 5(8):1063–1069.

Østergaard, K., Stoustrup, J., and Brath, P. (2009). Linear parameter varying control of wind turbines covering both partial load and full load conditions. *International Journal of Robust and Nonlinear Control*, 19(1):92–116.

Øye, S. (1999). FLEX 5 user manual. *Danske Techniske Hogskole*.

Petersen, E., Mortensen, N., Landberg, L., Højstrup, J., and Frank, H. (1998). Wind power meteorology. Part I: Climate and turbulence. *Wind Energy*, 1(s 1):25–45.

Pluymers, B., Rossiter, J., Suykens, J., and De Moor, B. (2005). The efficient computation of polyhedral invariant sets for linear systems with polytopic uncertainty. In *American Control Conference*, pages 804–809. IEEE.

Powell, D. and Elderkin, C. (1974). An investigation of the application of taylor's hypothesis to atmospheric boundary layer turbulence. *Journal of the Atmospheric Sciences*, 31(4):990–1002.

Prékopa, A. (1995). *Stochastic Programming*. Kluwer Academic.

Rawlings, J. and Muske, K. (1993). The stability of constrained receding horizon control. *IEEE Transactions on Automatic Control*, 38(10):1512–1516.

Rossiter, J., Kouvaritakis, B., and Rice, M. (1998). A numerically robust state-space approach to stable-predictive control strategies. *Automatica*, 34(1):65–73.

Rossiter, J. and Wang, L. (2008). Exploiting laguerre functions to improve the feasibility/performance compromise in MPC. In *IEEE Conference on Decision and Control*, pages 4737–4742. IEEE.

Rossiter, J., Wang, L., and Valencia-Palomo, G. (2010). Efficient algorithms for trading off feasibility and performance in predictive control. *International Journal of Control*, 83(4):789–797.

Schuurmans, J. and Rossiter, J. (2000). Robust predictive control using tight sets of predicted states. *Control Theory & Applications*, 147(1):13–18.

Scokaert, P., Mayne, D., and Rawlings, J. (1999). Suboptimal model predictive control (feasibility implies stability). *IEEE Transactions on Automatic Control*, 44(3):648–654.

Scokaert, P. and Rawlings, J. (1998). Constrained linear quadratic regulation. *IEEE Transactions on Automatic Control*, 43(8):1163–1169.

Scokaert, P. and Rawlings, J. (1999). Feasibility issues in linear model predictive control. *AIChE Journal*, 45(8):1649–1659.

Soliman, M., Malik, O., and Westwick, D. (2011). Multiple model predictive control for wind turbines with doubly fed induction generators. *IEEE Transactions on Sustainable Energy*, 2(3):215–225.

Sznaier, M. and Damborg, M. (1987). Suboptimal control of linear systems with state and control inequality constraints. In *IEEE Conference on Decision and Control*, volume 26, pages 761–762. IEEE.

van der Veen, G. (2013). *Identification of wind energy systems*. PhD thesis, Delft University of Technology.

van der Veen, G., van Wingerden, J., Fleming, P., Scholbrock, A., and Verhaegen, M. (2013). Global data-driven modeling of wind turbines in the presence of turbulence. *Control Engineering Practice*, 21(4):441–454.

Wang, L. (2004). Discrete model predictive controller design using Laguerre functions. *Journal of Process Control*, 14(2):131–142.