A multiresolution approximation method for fast explicit model predictive control

Sean Summers, Colin N. Jones, John Lygeros, and Manfred Morari

***Abstract*—A model predictive control law is given by the solution to a parametric optimization problem that can be pre-computed offline and provides an explicit map from state to control input. In this paper, an algorithm is introduced based on wavelet multiresolution analysis that returns a low complexity explicit model predictive control law built on a hierarchy of second order interpolets. The resulting interpolation is shown to be everywhere feasible and continuous. Further, tests to confirm stability and to compute a bound on the performance loss are introduced. Since the controller approximation is built on a grid hierarchy, convergence to a stabilizing control law is guaranteed and the evaluation of the control law in real-time systems is naturally fast and runs in a bounded logarithmic time. Two examples are provided; A two-dimensional example with an evaluation speed of 31 ns and a four-dimensional example with an evaluation speed of 119 ns.**

***Index Terms*—Model Predictive Control; Multiscale Methods; Explicit MPC; Fast MPC; Receding-Horizon control.**

I. Introduction

The implementation of a model predictive control (MPC) law requires the solution of an optimization problem online at each sampling instant. This optimization problem can be posed parametrically, with the measured state x as the parameter

$$J^*(x) := \min \{h(x,u) : g(x,u) \leq 0\}.$$

The offline computation of this parametric problem results in an explicit optimal control law $u^*(x)$ mapping the current state x to the optimal system input [1]–[3]. The result is an online computation of the optimal control law which depends on the evaluation of the explicit function $u^*(x)$ at state x rather than the solution of the optimization problem. In many relevant cases, the result is a significant decrease in the required online computation. This approach is known as Explicit MPC.

There are two main limitations of Explicit MPC. First, if the optimal control law can be computed, the storage requirements of the explicit controller can grow quickly with problem size. Also, as the controller complexity grows, the worst case computation time may rise above a practical value, thereby eliminating it as a viable choice in a real-time system. Therefore, it is only natural to consider approximate controllers whenever a reduction in the storage requirements and/or online computation time is needed. For a general overview of the problem, the reader is referred to the recent surveys [4], [5].

Research supported by the Swiss National Science Foundation under grant 200021-122072 and by ETH Zurich under grant ETH-12 09-2

Sean Summers, Colin N. Jones, John Lygeros, and Manfred Morari are with the Automatic Control Laboratory, Department of Information Technology and Electrical Engineering, ETH Zurich, Switzerland summers,cjones,lygeros,morari@control.ee.ethz.ch

Several authors have proposed approximation algorithms that can produce simpler explicit control laws for linear MPC problems at the cost of optimality [4], [6]–[9]. In almost all cases, the authors initially approximate the epigraph of the optimal cost function J^* with a polyhedron $\tilde{J}$, usually designed to ensure specific stability or performance constraints. Then, a feasible control law $\tilde{u}(x)$ is computed such that $\tilde{J}$ is a Lyapunov function for the resulting closed-loop system. The computation of a feasible control law $\tilde{u}$ which preserves the stability and performance of the cost function approximation can be done using the techniques discussed in [4], [6], [7], [9], [10].

In the same spirit, we introduce an algorithm that constructs a low complexity approximate explicit control law using adaptive multiscale bases. Such bases provide a powerful means to detect local singularities and often lead to quite simple refinement strategies. Some examples of possible bases include orthogonal Daubechies wavelets [11]–[13], biorthogonal spline wavelets [14], and interpolets [15], [16]. Here, we use a hierarchy of basis functions to approximate the explicit control law, where the basis functions are constructed using a tensor product expansion of second order interpolets.

The resulting approximation by multiscale bases is shown to be everywhere continuous, low in storage requirements, and require limited online computation. Further, we show that the second order interpolets result in an interpolation by barycentric coordinates, which guarantees that the resulting interpolation lies within the convex hull of the points being interpolated. We emphasize that it is this property, i.e. the interpolation by barycentric coordinates, which motivates the use of second order interpolets as the basis function of choice rather than one of many alternatives, e.g. radial basis functions [17]. As a result of this property, we are able to accurately evaluate the stability and feasibility of the approximate model predictive control law, and even guarantee the controller to be within a specified performance threshold of optimal.

In Section II we construct a d-dimension multiscale basis function, introduce an adaptive thresholding approach for sparse function approximation, and show that the multiscale basis we have constructed is comprised of barycentric coordinates. In Section III we introduce the well known linear Model Predictive Control problem and the corresponding feasibility, stability, and performance guarantees for the approximate control law. In Section IV we introduce the numerical implementation of the adaptive algorithm for the approximate control law and discuss the computational complexity of the method. In Section V we provide numerical examples of the method.

Preprint submitted to IEEE Transactions on Automatic Control. Received: April 19, 2011 09:58:22 PST

II. Multiscale Function Approximation

Our approximation method effectively relies on coarsely gridding the state space, and then systematically regridding with increasing resolution the regions which have not been approximated sufficiently (i.e. not all grid points are evaluated) while only keeping the grid points which play a significant role in the function approximation. We start with a brief introduction of the sparse multiresolution methodology for function approximation. The reader is referred to [18]–[22] for more information on which subsections II-A, II-B, and II-C are based.

A. One-Dimensional Multiscale Basis

Define the standard one-dimensional scaling function (hat function) with support $[-1,1]$ by

$$\phi(x) := \begin{cases} 1-|x| \ , & \text{if } x \in [-1,1], \\ 0 \ , & \text{otherwise.} \end{cases} \tag{1}$$

In one dimension, we consider a dyadic discretization on the unit interval $\Omega = [0,1]$. The resulting grid Ω_l is characterized by the level of discretization l and the index i. At level l the distance between points is $h_l = 2^{-l}$ and the number of points is $N = 2^l + 1$. The index i determines the location of the grid points according to the equation

$$x_{l,i} := i \cdot h_l, \ \ 0 \le i \le 2^l.$$

Consider $\phi_{l,i}$ a family of basis functions defined on Ω with support $[x_{l,i} - h_l, x_{l,i} + h_l]$. The function $\phi_{l,i}$ is generated from function (1) via translation and dilation,

$$\phi_{l,i}(x) = \phi\left(\frac{x - i \cdot h_l}{h_l}\right). \tag{2}$$

The family of basis functions $\phi_{l,i}$ is commonly referred to as a nodal basis of level l, and can be used to construct a function space V_l of piecewise linear functions,

$$V_l := \text{span}\{\phi_{l,i} : 0 \le i \le 2^l\}.$$

That is, any function $u_l \in V_l$ can be uniquely represented in the nodal basis as a weighted sum of basis functions

$$u_l(x) = \sum_{i=0}^{2^l} v_{l,i} \cdot \phi_{l,i}(x) \ , \tag{3}$$

where the coefficients $v_{l,i}$ are equal to the function value $u_l(x_{l,i})$.

Equivalently, V_l can be constructed as the summation of hierarchical function spaces W_k, where the basis corresponding to W_k is referred to as a hierarchical basis. The hierarchical function space W_k, for $k \in \mathbb{N}_0$ and $k \ge l_0$, is defined as

$$W_k := \text{span}\left\{\phi_{k,i} : i \in I_k\right\} \ ,$$

with the hierarchical index set defined

$$I_k = \begin{cases} \{i \in \mathbb{N}_0 : 1 \le i \le 2^k - 1, i \text{ odd}\} & k > l_0 \ , \\ \{i \in \mathbb{N}_0 : 0 \le i \le 2^k\} & k = l_0 \ , \end{cases}$$

where $l_0 \in \mathbb{N}_0$ denotes the lowest level of discretization (typically $l_0 = 0$). The family of univariate multiscale functions $\psi_{k,i}$ which make up the hierarchical basis are hereafter defined as

$$\psi_{k,i} = \phi_{k,i}, \quad i \in I_k.$$

It can be shown that

$$V_l = \bigoplus_{k \le l} W_k,$$

where $\oplus$ symbolizes the direct sum. In other words, the same function $u_l \in V_l$ which was expressed in the nodal basis in (3) can be equivalently represented in the hierarchical basis by the expression

$$u_l(x) = \sum_{k=l_0}^{l} \sum_{i \in I_k} w_{k,i} \cdot \psi_{k,i}(x) \ ,$$

with coefficients $w_{k,i} \in \mathbb{R}$, commonly referred to as the hierarchical details, corresponding to the difference between the true function value $u_l(x_{k,i})$ and the value of the approximate function one level below. Since every basis function in the hierarchical function space can be defined as a weighted sum of basis functions in the nodal function space, i.e. for every $\psi \in \{\psi_{k,i} : l_0 \le k \le l, \ \ i \in I_k\}$

$$\psi = \sum_{i=0}^{2^l} a_i \cdot \phi_{l,i}$$

where $a_i \in \mathbb{R}$ for all $i \in \{0, \ldots, 2^l\}$, there exists a linear transformation from the nodal basis representation to the hierarchical basis representation. Considering the vector of the nodal basis functions $\vec{\phi}$ and the vector of the hierarchical basis functions $\vec{\psi}$, the transformation between basis' can be expressed

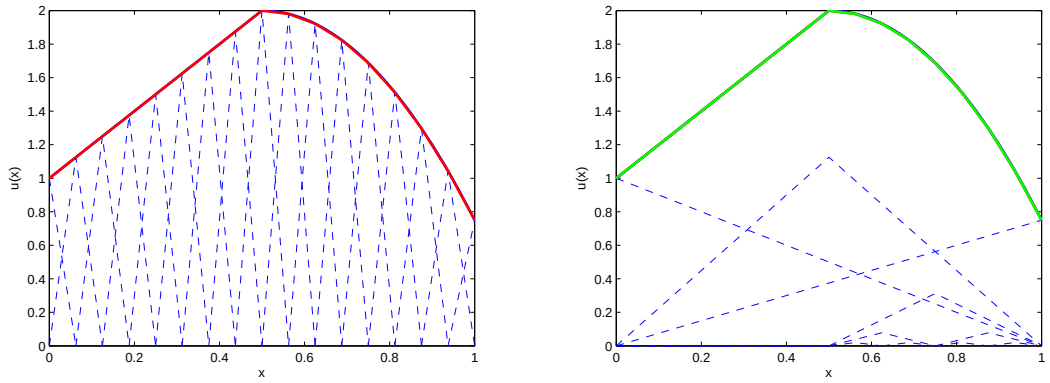
$$\vec{\psi} = A\vec{\phi},$$

where the matrix A is square and invertible (i.e. there also exists an inverse transformation from the hierarchical basis to the nodal basis). In the wavelet community, these transformations are known as forward and inverse wavelet transforms [20], [22].

Note that one can project general continuous functions $u : [0,1] \to \mathbb{R}$ onto the subspace V_l. The resulting projection $u_l \in V_l$ is a piecewise linear approximation of u and can be represented uniquely in both the nodal basis and the hierarchical basis. Consider the continuous function

$$u(x) = \begin{cases} 2x+1 \ , & \text{if } x \le \frac{1}{2} \\ -5(x-\frac{1}{2})^2 + 2 \ , & \text{else} \end{cases} \tag{4}$$

defined on $x \in [0,1]$. Results for a piecewise linear approximation of (4) with level $l = 4$ and minimal hierarchical level $l_0 = 0$ are shown in Figure 1. It is important to note that the piecewise linear approximation $u_4(x)$ of (4) is equivalent in both the nodal and hierarchical representations. For the nodal basis functions, the projection of u onto u_l (Figure 1(a)) is easy since the weights are equivalent to the value of (4) at the nodes. As a result, the weights tend to be large and the removal of a single nodal basis function would be detrimental to the approximation. In contrast, the projection of u onto u_l using hierarchical basis functions (Figure 1(b))



(a) Nodal representation (b) Hierarchical representation

Fig. 1. Example of a piecewise linear approximation of the function $u(x)$ defined in (4) with a maximum resolution of level $l = 4$ given in the nodal basis 1(a) and the hierarchical basis 1(b). 1(a) The piecewise linear approximation $u_4(x)$ (solid line) with corresponding nodal basis functions (dashed lines). 1(b) The piecewise linear approximation $u_4(x)$ (solid line) with corresponding hierarchical basis functions (dashed lines). In both 1(a) and 1(b), the piecewise linear function (solid line) is the result of the summation of the basis functions (dashed lines).

is slightly more complex, but allows for adaptive thresholding (see Section II-C) since the weights (hierarchical details) tend to decrease significantly with resolution. That is, the removal of the hierarchical basis functions with very small weights would not greatly effect the approximation of (4).

B. d-Dimensional Multiscale Basis

Let $\mathbf{y} \in \mathbb{N}_0^d$ denote a d-dimensional multi-index (i.e. $\mathbf{y}$ is a d-vector of indices taking values in the non-negative integers), where operations (e.g. addition) and comparisons (e.g. $\leq$) hold component-wise and y_j is the jth component of $\mathbf{y}$. A multivariate multiscale basis on the unit cube $\Omega^d = [0,1]^d$, where d is the dimension, can be constructed by tensor product expansion of the one-dimensional multivariate functions $\psi_{k,i}$ (see Figure 2), i.e.

$$\psi_{k,\mathbf{i}} = \prod_{j=1}^{d} \psi_{k,i_j}$$

with the multi-index $\mathbf{i} \in I_k^d$ and

$$I_k^d = \begin{cases} \left\{\mathbf{i} \in \mathbb{N}_0^d : \mathbf{0} \leq \mathbf{i} \leq \mathbf{2}^k\right\} \setminus \left\{\mathbf{i} \in \mathbb{N}_0^d : \right. \\ \left. \mathbf{0} \leq \mathbf{i} \leq \mathbf{2}^k, i_j \text{ even } \forall j \in \{1,\ldots,d\}\right\} & k > l_0 \ , \\ \left\{\mathbf{i} \in \mathbb{N}_0^d : \mathbf{0} \leq \mathbf{i} \leq \mathbf{2}^k\right\} & k = l_0 \ . \end{cases}$$

Note that I_k^d is simply the full grid at level k minus those points seen at previous levels, as depicted in Figure 3 for the 2-dimensional case. We may now define the d-dimensional hierarchical function spaces of piecewise d-linear functions as

$$W_k^d = \text{span}\{\psi_{k,\mathbf{i}} : \mathbf{i} \in I_k^d\}.$$

Defining the family of d-dimensional nodal basis functions

$$\phi_{l,\mathbf{i}}(x) = \prod_{j=1}^{d} \phi_{l,i_j}(x_j)$$

and the d-dimensional nodal function space

$$V_l^d = \text{span}\{\phi_{l,\mathbf{i}} : \mathbf{0} \leq \mathbf{i} \leq \mathbf{2}^l\},$$

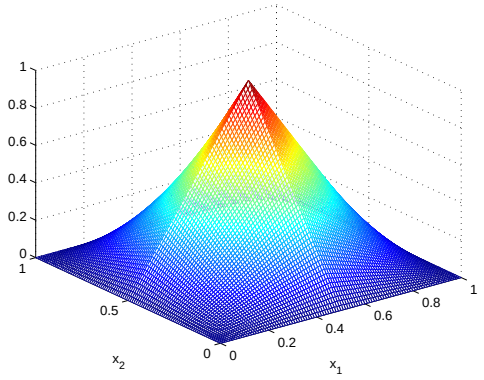


Fig. 2. Two dimensional hierarchical basis function by tensor product expansion of the one dimensional hat function.

it can be shown that

$$V_l^d = \bigoplus_{k \leq l} W_k^d.$$

$$u_l(x) = \sum_{\mathbf{i}=\mathbf{0}}^{\mathbf{2}^l} v_{l,\mathbf{i}} \cdot \phi_{l,\mathbf{i}}(x)$$

with coefficients $v_{l,\mathbf{i}}$. Likewise, the same function $u_l \in V_l^d$ can be uniquely represented in the hierarchical basis by

$$u_l(x) = \sum_{k=\mathbf{0}}^{l} \sum_{\mathbf{i} \in I_k^d} w_{k,\mathbf{i}} \cdot \psi_{k,\mathbf{i}}(x)$$

with hierarchical details $w_{k,\mathbf{i}} \in \mathbb{R}$.

C. Adaptive Thresholding

In the hierarchical basis methodology, the weight of the specific scaling function $w_{k,\mathbf{i}}$, and not the function value $u(x)$, is saved at each grid point $x \in \Omega_k^d$. The level of approximation is initialized at the coarsest level, usually $l_0 = 0$, and continues

Preprint submitted to IEEE Transactions on Automatic Control. Received: April 19, 2011 09:58:22 PST

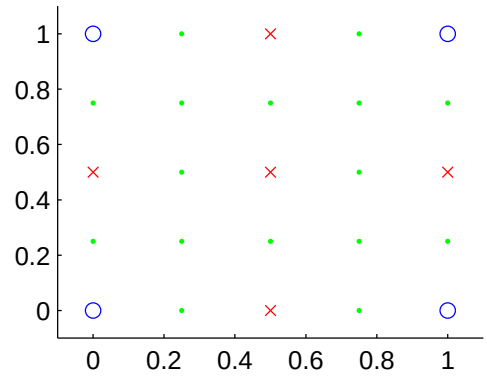


Fig. 3. Grid points for subspaces W_0^2 (circles), W_1^2 (x's), and W_2^2 (dots).

Algorithm 1 Adaptive Function Approximation

Require: A function $u \in V$ from which samples can be taken and a threshold $\delta \in \mathbb{R}$

Ensure: An approximate function given by Λ^δ and $\mathbf{w}$

1: Define the initial 'active' index set $\Lambda^\delta = \{(l_0, \mathbf{i}) : \mathbf{i} \in I_k\}$ and temporary index set $\hat{\Lambda}^\delta = \emptyset$
2: Compute initial details $\mathbf{w} = \{w_{k,\mathbf{i}} : (k, \mathbf{i}) \in \Lambda^\delta\}$
3: **while** $\Lambda^\delta \backslash \hat{\Lambda}^\delta \neq \emptyset$ **do**
4: Set $\hat{\Lambda}^\delta = \Lambda^\delta$
5: Compute $\Lambda_n^\delta = \bigcup_{(k,\mathbf{i}) \in \Lambda^\delta} \mathcal{M}(k, \mathbf{i})$
6: Compute $\mathbf{w}_n = \{w_{k,\mathbf{i}} : (k, \mathbf{i}) \in \Lambda_n^\delta, |w_{k,\mathbf{i}}| \geq \delta\}$
7: Evaluate $\tilde{\Lambda}_n^\delta = \{(k, \mathbf{i}) : (k, \mathbf{i}) \in \Lambda_n^\delta, w_{k,\mathbf{i}} \in \mathbf{w}_n\}$
8: Increase index set $\Lambda^\delta = \Lambda^\delta \cup \tilde{\Lambda}_n^\delta$
9: Increase details set $\mathbf{w} = \mathbf{w} \cup \mathbf{w}_n$
10: **end while**

in a direction of dyadic refinement. In many relevant cases the hierarchical details tend to zero as the grid space is refined [18]–[20]. To decrease the storage requirements for the approximation of a function, one may consider storing only those grid points whose weights are larger than a user specified threshold. Thus, we may significantly reduce the number of stored variables (hierarchical details) for the approximation of a function $u(x)$ without a significant loss in the accuracy of the approximation. The adaptive function approximation can be expressed as

$$\hat{u}(x) = \sum_{(k,\mathbf{i}) \in \Lambda^\delta} w_{k,\mathbf{i}} \cdot \psi_{k,\mathbf{i}}(x) \ ,$$

where the 'active' index set Λ^δ (i.e. the index set corresponding to non-zero detail coefficients) is a function of the user defined threshold δ as well as the function to be approximated u. Common thresholding, $|w_{k,\mathbf{i}}| \geq \delta$, simply reduces to taking the absolute value of the hierarchical detail greater than the threshold, and discarding the weight $w_{k,\mathbf{i}}$ if it is lower than the threshold.

We employ a unidirectional nearest neighbor bisection approach for the grid resolution update. We denote a nearest neighbor as any point that is exactly one level up and at most one index removed in all dimensions (see (5)). The nearest neighbor function $\mathcal{M}(k, \mathbf{i})$ maps a single index $(k, \mathbf{i}) \in \Lambda^\delta$ to its set of nearest neighbors in level and space, i.e.

$$\begin{aligned} \mathcal{M}(k, \mathbf{i}) \quad = \quad & \{(k+1, \mathbf{q}) : \mathbf{q} \in I_{k+1}, \\ & |q_j - 2i_j| \leq 1 \ \ \forall j \in \{1, \ldots, d\}\} \, . \end{aligned} \tag{5}$$

We now proceed with Algorithm 1. The algorithm initializes the 'active' index at the coarsest level, i.e. l_0. Then, the nearest neighbors in level and space are determined and the corresponding hierarchical details are computed. Only the details with an absolute value greater than the user defined threshold are saved while the rest are discarded. This process is repeated until no more details can be added or a maximum resolution has been attained.

Remark 1: Ideally, the output from Algorithm 1 would result in an approximation containing all weights with an absolute value greater than the threshold. Yet, this will rarely be the case since only the neighbors of the saved grid points are explored, and thus significant areas of the space may be missed. A more sophisticated error indicator may alleviate this problem in specific cases, but modifications to the adaptive algorithm are usually advised against. Because, for any modification to the adaptive search method, there is always some function which will lead the adjusted procedure to fail. However, it will be seen in the sequel that when using the second order interpolets as the basis, an upper bound on the maximum error can be achieved if the function being approximated is convex.

D. Barycentric Coordinates

For any set $R \subset \mathbb{R}^d$, let $\text{conv}(R)$ be the convex hull of R and let $\text{extr}(R)$ be the set of extreme points.

Definition 1 (Barycentric Coordinates [23]): Let $S := \text{conv}(\{v_1, \ldots, v_n\}) \subset \mathbb{R}^d$ be a polytope. A set of functions $f_v(x)$ are called *barycentric coordinates* if for all $x \in S$ and $v \in \text{extr}(S)$

$$f_v(x) \geq 0 \ , \qquad \textit{positivity} \tag{6}$$

$$\sum_{v \in \text{extr}(S)} f_v(x) = 1 \ , \qquad \textit{partition of unity} \tag{7}$$

$$\sum_{v \in \text{extr}(S)} v f_v(x) = x \ . \qquad \textit{linear precision} \tag{8}$$

Barycentric coordinates provide a convenient way to interpolate a function on a mesh, requiring only the function values at the vertices for interpolation. In many cases, the interpolation is continuous and nonlinear. Additionally, the interpolation of the data at the vertices by barycentric coordinates is always guaranteed to be inside the convex hull of the points being interpolated [23]. As a result of this property, the authors in [10] were able to show that an approximate MPC law constructed via interpolation over polytopes by barycentric coordinates guaranteed feasibility, stability, and performance bounds.

In the same spirit as [10], we now show that the function approximation by adaptive hierarchical basis function expansion introduced in Section II-C generates a grid (of hypercubes) spanned by an interpolation by barycentric coordinates. Given

a nodal function approximation space V_l^d and a d-dimensional space Ω^d, consider the set of level l hypercubic regions $R_{l,\mathbf{j}}$ of width h_l whose centers are given by $x_{(l+1,\mathbf{j})} \in \Omega^d$, $\mathbf{j} \in I_{l,\mathbf{j}}$ where

$$I_{l,\mathbf{j}} = \{\mathbf{j} \in \mathbb{N}_0^d : \mathbf{0} \leq \mathbf{j} \leq \mathbf{2}^{(l+1)}, j_k \text{ odd } \forall k \in \{1,\dots,d\}\}. \tag{9}$$

A hypercubic region $R_{l,\mathbf{j}}$ is defined

$$R_{l,\mathbf{j}} \quad = \quad \left\{ x \in \Omega^d : x \in \bigcap_{\mathbf{i} \in I_{R_{l,\mathbf{j}}}} \text{supp}(\phi_{l,\mathbf{i}}) \right\} \quad , \tag{10}$$

where

$$I_{R_{l,\mathbf{j}}} := \{\mathbf{i} \in \mathbb{N}_0^d : \mathbf{0} \leq \mathbf{i} \leq \mathbf{2}^l, x_{(l+1,\mathbf{j})} \in \text{supp}(\phi_{l,\mathbf{i}})\}$$

is the set of indices corresponding to the vertices of the hypercube $R_{l,\mathbf{j}}$. Note that supp$(\cdot)$ denotes the support of a function and that $\bigcup_{\mathbf{j}} R_{l,\mathbf{j}} = \Omega^d$. Also, it holds that all $R_{l,\mathbf{j}}$ are hypercubes since the basis functions are axis aligned and have hypercubic support, hence, finite intersections result in either the empty set or a hypercube.

We now arrive at a critical lemma in which we show that the nodal basis functions are barycentric coordinates for all $R_{l,\mathbf{j}} \subseteq \Omega^d$, and thus will lead to provable arguments regarding the feasibility, stability, and performance of the approximate control law. An alternative proof for Lemma 1 can be found in [24].

Lemma 1 (Nodal Barycentric Coordinates): Given a d-dimensional space Ω^d and nodal function approximation space V_l^d, the multivariate basis of tensor product second order interpolets defined over any hypercube $R_{l,\mathbf{j}} \subseteq \Omega^d$ are barycentric coordinates for $R_{l,\mathbf{j}}$.

Proof: By definition, $R_{l,\mathbf{j}}$ is a hypercube with 2^d vertices $v \in \text{extr}(R_{l,\mathbf{j}}) = \{v \in \Omega_l^d : \mathbf{i} \in I_{R_{l,\mathbf{j}}}, v = [i_1 h_l, \dots, i_d h_l]\}$, where $h_l = 2^{-l}$ is the width of the hypercube $R_{l,\mathbf{j}}$. Note that because $R_{l,\mathbf{j}}$ is a hypercube, the index values i_k in the k^{th} dimension take only two values, an upper $\bar{i}_k$ and a lower $\underline{i}_k$, and are separated by a value of one, i.e. $\bar{i}_k - \underline{i}_k = 1$. We evaluate each of the three characteristics of barycentric coordinates. *Positivity* is straightforward by the definition of the hat function (1). For the *partition of unity* property, we first note that for all $j \in \{1,\dots,d\}$

$$\phi_{l,\bar{i}_j}(x_j) + \phi_{l,\underline{i}_j}(x_j) \quad = \quad 1.$$

It can then be shown that

$$\begin{aligned} \sum_{v \in \text{extr}(R_{l,\mathbf{j}})} f_v(x) \quad &= \quad \sum_{\mathbf{i} \in I_{R_{l,\mathbf{j}}}} \prod_{j=1}^{d} \phi_{l,i_j}(x_j), \\ &= \quad \prod_{j=1}^{d} (\phi_{l,\bar{i}_j}(x_j) + \phi_{l,\underline{i}_j}(x_j)), \\ &= \quad 1, \end{aligned}$$

which satisfies the *partition of unity*. Lastly, we prove that the multiscale basis satisfies the *linear precision* requirement. We evaluate *linear precision* dimension by dimension, i.e. for each $k \in \{1,\dots,d\}$ we show that $\sum_{v \in \text{extr}(R_{l,\mathbf{j}})} v_k f_v(x) = x_k$ where v_k and x_k are the kth coordinates of v and x. For $k \in \{1,\dots,d\}$ we have

$$\begin{aligned} \sum_{v \in \text{extr}(R_{l,\mathbf{j}})} v_k f_v(x) \quad &= \quad \sum_{\mathbf{i} \in I_{R_{l,\mathbf{j}}}} i_k h_l \prod_{j=1}^{d} \phi_{l,i_j}(x_j) \\ &= \quad (\bar{i}_k h_l \phi_{l,\bar{i}_k}(x_k) + \underline{i}_k h_l \phi_{l,\underline{i}_k}(x_k)) \cdot \\ &\quad \prod_{j \neq k} (\phi_{l,\bar{i}_j}(x_j) + \phi_{l,\underline{i}_j}(x_j)) \\ &= \quad \bar{i}_k h_l \phi_{l,\bar{i}_k}(x_k) + \underline{i}_k h_l \phi_{l,\underline{i}_k}(x_k). \end{aligned} \tag{11}$$

By the definition of the interpolets, (1) and (2), we can expand (11) such that

$$\begin{aligned} &\bar{i}_k h_l \phi_{l,\bar{i}_k}(x_k) + \underline{i}_k h_l \phi_{l,\underline{i}_k}(x_k) = \\ &\bar{i}_k h_l \left(1 - \left|\frac{x_k - \bar{i}_k h_l}{h_l}\right|\right) + \underline{i}_k h_l \left(1 - \left|\frac{x_k - \underline{i}_k h_l}{h_l}\right|\right). \end{aligned}$$

Taking into account the properties $\bar{i}_k - \underline{i}_k = 1$ and $\underline{i}_k h_l \leq x_k \leq \bar{i}_k h_l$, simple arithmetic leads us to the result

$$\bar{i}_k h_l \left(1 - \left|\frac{x_k - \bar{i}_k h_l}{h_l}\right|\right) + \underline{i}_k h_l \left(1 - \left|\frac{x_k - \underline{i}_k h_l}{h_l}\right|\right) \quad = \quad x_k.$$

Repeating this process for all dimensions $k \in \{1,\dots,d\}$ leads to the property of *linear precision*. ■

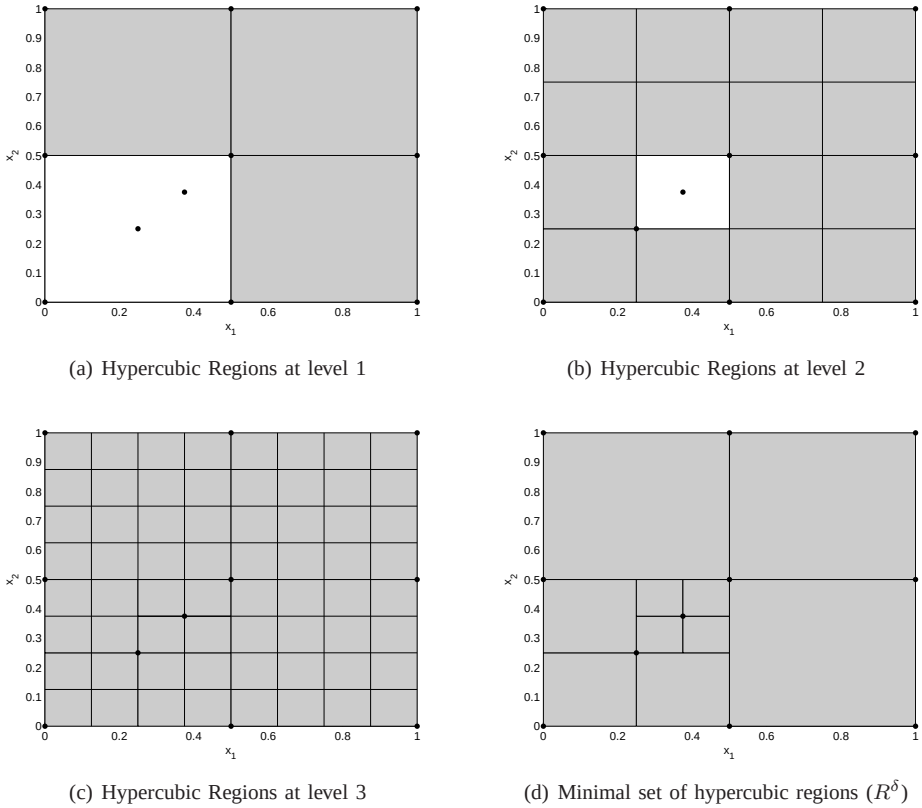
Next, we show that the above also holds for adaptive hierarchical function approximation spaces.

Corollary 1 (Hierarchical Barycentric Coordinates): Given an active multiscale index set $\Lambda^\delta(u)$ spanning multiple levels of resolution from l_0 to l (i.e. the indices corresponding to all active hierarchical details in $W_{l_0}^d, \dots, W_l^d$), the following properties hold

1) The function approximation space $\bigoplus_{k \leq l} W_k^{d,\delta}$ can be decomposed into hypercubes with barycentric coordinates at the vertices, where $W_k^{d,\delta} := \text{span}\{\psi_{k,\mathbf{i}} : (k,\mathbf{i}) \in (k, I_k^d) \cap \Lambda^\delta(u)\}$.
2) The set of hypercubic regions up to level l in which the basis functions form an interpolation by barycentric coordinates is
$$\overline{R} = \left\{ \begin{array}{c} R \subseteq \Omega^d : k \in \{l_0,\dots,l\},\ \ R = R_{k,\mathbf{j}}\ \ , \mathbf{j} \in I_{k,\mathbf{j}} \\ \text{and} \\ \forall (\overline{k},\mathbf{i}) \in \Lambda^\delta(u),\ \ \overline{k} > k,\ \ x_{\overline{k},\mathbf{i}} \notin R \end{array} \right\}$$
where $R_{k,\mathbf{j}}$ is defined in (10) and $I_{k,\mathbf{j}}$ is defined in (9).
3) The minimal set of hypercubic regions spanning Ω^d in which the basis functions form an interpolation by barycentric coordinates is
$$R^\delta = \{R \in \overline{R} : R \not\subset Q,\ \ \forall Q \in \overline{R} \setminus R\}. \tag{12}$$

Proof: The first item is straightforward. We fill the function approximation space with $w_{k,\mathbf{i}} = 0$ for all $(k,\mathbf{i}) \in \{(k,\mathbf{i}) \in (\mathbb{N},\mathbb{N}^d) : k \in \{l_0,\dots,l\},\ (k,\mathbf{i}) \notin \Lambda^\delta(u)\}$ such that $V_l^d = \bigoplus_{k \leq l} W_k^{d,\delta}$, and then apply Lemma 1. The second item is a consequence of (9) and (10). The third item results in Ω^d decomposed into a minimal set of hypercubes (possibly) varying in size depending on the set of active basis functions. ■

Preprint submitted to IEEE Transactions on Automatic Control. Received: April 19, 2011 09:58:22 PST



(a) Hypercubic Regions at level 1

(b) Hypercubic Regions at level 2

(c) Hypercubic Regions at level 3

(d) Minimal set of hypercubic regions (R^δ)

Fig. 4. An example of a $d = 2$ hierarchical function approximation in Ω^2 over levels $k \in \{1,\ 2,\ 3\}$. The active hierarchical indices are given by the corresponding centers (dots) on the grids. The hypercubic regions in which the basis functions form an interpolation by barycentric coordinates are marked gray.

Corollary 1 effectively states that any function approximation built adaptively in the hierarchical function spaces forms a function interpolation by barycentric coordinates. Properties 1 and 2 exploit the fact that, at the highest level of resolution, a transformation from the hierarchical function space to the nodal function space results in a full nodal grid. Further, while the total number of hypercubic regions satisfying the properties of barycentric coordinates are possibly overlapping (since all $R_{l,\mathbf{j}}$ at the maximum level of resolution satisfy this characteristic at the least), because the hierarchical approximation space may in some cases be sparse, the minimal set of hypercubes spanning Ω^d can be quite small (Property 3). An example is illustrated in Figure 4. We consider a $d = 2$ hierarchical function approximation in Ω^2 over levels $k \in \{1,\ 2,\ 3\}$ where the active hierarchical indices are given by the corresponding centers (dots on the grids). Subfigures 4(a), 4(b), and 4(c) illustrate the hypercubic regions at levels 1, 2, and 3 (marked gray) in which the basis functions form an interpolation by barycentric coordinates. Note that these regions are overlapping since any basis function at level k can be uniquely represented by a linear combination of nodal basis functions at any level $l \geq k$. Further, the set $\overline{R}$ is the union of the hypercubes marked gray in Subfigures 4(a), 4(b), and 4(c). The minimal set of hypercubes spanning the approximation space Ω^2 (i.e. R^δ) is represented in Subfigure 4(d). Note that the hierarchical function spaces at levels $k \in \{2,\ 3\}$ are sparsely populated (i.e. only one basis function at each level $k \in \{2,\ 3\}$ is active), and as a result the set R^δ also conveys this sparsity.

III. Applications to Model Predictive Control

The interest in parametric programming in the control community in recent years has arisen from the ability to pose certain optimal control problems as parametric programs, resulting in numerous methods to pre-compute the optimal control law offline. In this paper we are specifically interested in the following finite horizon optimal control problem:

$$\begin{aligned} J^*(x) = \min_{\{u_0,\ldots,u_{N-1}\}} & J(u_0,\ldots,u_{N-1},x_0,\ldots,x_N) \qquad (13)\\ \text{s.t.} \quad & x_{i+1} = Ax_i + Bu_i, \\ & \qquad \forall i = 0,\ldots,N-1 \\ & (x_i,u_i) \in \mathcal{X}\times\mathcal{U},\ \forall i = 0,\ldots,N-1 \\ & x_N \in \mathcal{X}_F, \\ & x_0 = x, \end{aligned}$$

where

$$J(u_0,\ldots,u_{N-1},x_0,\ldots,x_N) := V_N(x_N) + \sum_{i=0}^{N-1} l(x_i,u_i)\ , \qquad (14)$$

and $\mathcal{X}$, $\mathcal{U}$, and $\mathcal{X}_F$ are compact convex constraints on the states and inputs and the stage cost l is a strictly convex function. A function $\gamma(\cdot) : \mathbb{R} \to \mathbb{R}$ is assumed to exist that

is continuous, strictly increasing and has $\gamma(0) = 0$ such that $\gamma(||x||) \leq l(x, 0)$ for all x. Problem (13) can be re-written as a parametric optimization problem:

$$u^*(x) := \arg\min_u \{h(x, u) : g(x, u) \leq 0\}, \tag{15}$$

where u is a vector containing the sequence of inputs $u_0, \ldots, u_{N-1}$ and appropriate auxiliary variables and the functions h and g are convex [1]. The system input is then given in a receding horizon fashion [25] by $u_0^*(x)$, which is the first input in the optimal control sequence of (13).

The following sections will demonstrate that an approximate control law built from the multiresolution methodology presented above leads to a stabilizing feasible explicit control law for the MPC problem (13).

A. Stability Guarantees of Approximate Controllers

In Section II-D we showed that the approximate controller built from the adaptive multiscale basis functions can be separated into hypercubic regions spanned by an interpolation by barycentric coordinates. We now evaluate the feasibility, stability, and performance of the individual regions by constructing a Lyapunov function for the approximate closed-loop system $x^+ = Ax + B\hat{u}(x)$.

Consider a hierarchical approximate control law defined on $\Omega^d \subset \mathbb{R}^d$ ($\mathcal{X}_F \subseteq \mathcal{X} \subseteq \Omega^d$) with maximum level l

$$\hat{u}(x) = \sum_{(k,\mathbf{i})\in\Lambda^\delta(u^*)} w_{k,\mathbf{i}} \cdot \psi_{k,\mathbf{i}}(x). \tag{16}$$

where u^* is given by (15). By Corollary 1, we see that the approximate control law defined on $R \in R^\delta$, where R^δ is defined in (12), can be expressed as an interpolation by barycentric coordinates

$$\hat{u}(x) = \sum_{v\in\text{extr}(R)} \hat{u}(v) f_v(x), \qquad \text{if } x \in R. \tag{17}$$

The following Lemma states that the resulting interpolation (i.e. the approximate control law) defined across $R \in R^\delta$ lies within the convex hull of the points being interpolated.

Lemma 2: If $R = \text{conv}(v_1, \ldots, v_{2^d}) \in \Omega^d$, $\hat{u}(v_j)$ is the approximate control law for the state v_j and $\hat{u}(x)$ is defined as in (17), then

$$\begin{pmatrix} x \\ \hat{u}(x) \end{pmatrix} \in \text{conv}\left(\begin{pmatrix} v_1 \\ \hat{u}(v_1) \end{pmatrix} \cdots \begin{pmatrix} v_{2^d} \\ \hat{u}(v_{2^d}) \end{pmatrix}\right),$$

for all $x \in R$.

Proof: Follows as consequence of $\hat{u}(x)$ expressed as an interpolation by barycentric coordinates [10], [26]. ■

Lemma 2 leads us to the following result.

Lemma 3 (Barycentric Feasibility): The approximate control law $\hat{u}(x)$ is a feasible solution of (13), i.e. $g(x, \hat{u}(x)) \leq 0, \ \forall x \in R$, if and only if $\hat{u}(v)$ is feasible for all $v \in \text{extr}(R)$.

Proof: Follows directly from Lemma 2 and the convexity of g. ■

We now show that the cost function (14) evaluated for the approximate control law (17) over $R \in R^\delta$ can be upper bounded by an approximate cost defined by the interpolation by barycentric coordinates of the approximate cost at the extremal points. In doing so, we show that the cost generated by the approximate control law (which is non-convex) is no more sub-optimal than the cost generated by the interpolation by barycentric coordinates, for which stability and performance can be computed. This enables us to prove stability and performance of the approximate control law.

Lemma 4: If $\hat{u}(x)$ is the barycentric control law defined in (17) and $\hat{u}(v)$ is feasible for all $v \in \text{extr}(R)$, then the following bounds hold

$$J^*(x) \leq J(\hat{u}(x)) \leq \tilde{J}(x), \quad \forall x \in R,$$

where

$$\tilde{J}(x) = \sum_{v\in\text{extr}(R)} J(\hat{u}(v)) f_v(x), \quad \forall x \in R.$$

Proof: The control law $\hat{u}(x)$ is feasible, yet approximate, for all $x \in R$ (Lemma 3), thus the cost function $J(\hat{u}(x))$ must be sub-optimal, which gives us the lower bound $J^*(x) \leq J(\hat{u}(x))$. The second portion of the proof can be found in [10], [26] and is briefly outlined as follows. Note that due to the barycentric interpolation it holds that $\hat{x}_i(x_0) = \sum_{v\in\text{extr}(R)} \hat{x}_i(v) f_v(x_0)$ and $\hat{u}_i(x_0) = \sum_{v\in\text{extr}(R)} \hat{u}_i(v) f_v(x_0)$ for all $i = 0, \ldots, N$, where $\hat{x}_i(x_0)$ is the solution of the state dynamics at the ith step in the sequence when initialized at x_0 and the approximate control law is applied at each step.

$$\begin{aligned} J(\hat{u}(x_0)) &= V_N(\hat{x}_N(x_0)) + \sum_{i=0}^{N-1} l(\hat{x}_i(x_0), \hat{u}_i(x_0)) \\ &= V_N\left(\sum_{v\in\text{extr}(R)} \hat{x}_N(v) f_v(x_0)\right) + \\ &\quad \sum_{i=0}^{N-1} l\left(\sum_{v\in\text{extr}(R)} \hat{x}_i(v) f_v(x_0), \sum_{v\in\text{extr}(R)} \hat{u}_i(v) f_v(x_0)\right) \end{aligned}$$

Convexity of the stage cost l and the terminal cost V_N gives the following relation:

$$\begin{aligned} &J(\hat{u}(x_0)) \\ &\leq \sum_{v\in\text{extr}(R)} f_v(x_0)\left(V_N(\hat{x}_N(x_0)) + \sum_{i=0}^{N-1} l(\hat{x}_i(x_0), \hat{u}_i(x_0))\right) \\ &= \sum_{v\in\text{extr}(R)} J(\hat{u}(v)) f_v(x_0) \end{aligned}$$

■

Lemma 4 proves that for each $x \in R$, the true approximate cost function $J(\hat{u}(x))$ will lie within the convex hull of the extreme points $\{(v, J(\hat{u}(v))) : v \in \text{extr}(R)\}$. With this key result in place, we can make use of an approximate stability theorem given in [26], which is motivated by the work [27].

Theorem 1: Let $J^* : \mathbb{R}^d \to \mathbb{R}$ be the cost function of the optimal control problem (13) and a Lyapunov function for the system $x^+ = Ax + Bu_0^*(x)$. The optimal cost function $J^*(x)$ is a Lyapunov function for the system $x^+ = Ax + B\hat{u}_0(x)$ if for all x in the feasible set $\mathcal{R}$ the control law $\hat{u}(x)$ is feasible and the condition $J^*(x) \leq J(\hat{u}(x)) \leq J^*(x) + \gamma(||x||)$ holds.

Proof: Assume that $J^*(x_0) \leq J(\hat{u}(x_0))$ for some x_0. It follows by assumption that $\hat{u}(x_0) = (\hat{u}_0(x_0), \ldots, \hat{u}_{N-1}(x_0))$

is a feasible input sequence for the state x_0, where $(x_0, \hat{x}_1(x_0), \ldots, \hat{x}_N(x_0))$ is the resulting feasible state sequence. Consider the time-shifted input and state sequences $(\hat{u}_1(x_0), \ldots, \hat{u}_{N-1}(x_0), \mu(\hat{x}_N(x_0)))$ and $(\hat{x}_1(x_0), \ldots, \hat{x}_N(x_0), A\hat{x}_N(x_0) + B\mu(\hat{x}_N(x_0)))$, which are feasible for (13) by the assumption that $\mathcal{X}_F$ is an invariant set under the control law $\mu(x)$. At the next time step we are at the state $x_1 = Ax_0 + B\hat{u}(x_0) = \hat{x}_1(x_0)$, and the shifted sequence is a feasible, sub-optimal solution to (13) at the state x_1. The cost J (13) evaluated at this shifted sequence is then bounded as follows

$$\begin{aligned}
J^*(x_1) \leq & J(\hat{u}_1(x_0), \ldots, \hat{u}_{N-1}(x_0), \mu(\hat{x}_N(x_0)), x_1, \ldots, \\
& \hat{x}_N(x_0), A\hat{x}_N(x_0) + B\mu(\hat{x}_N(x_0)) \\
= & V_N(A\hat{x}_N(x_0) + B\mu(\hat{x}_N(x_0))) \\
& + \sum_{i=1}^{N} l(\hat{x}_i(x_0), \hat{u}_i(x_0)) \\
= & V_N(\hat{x}_N(x_0)) + \sum_{i=0}^{N-1} l(\hat{x}_i(x_0), \hat{u}_i(x_0)) \qquad (18) \\
& + l(\hat{x}_N(x_0), \hat{u}_N(x_0)) - V_N(\hat{x}_N(x_0)) \qquad (19) \\
& + V_N(A\hat{x}_N(x_0) + B\mu(\hat{x}_N(x_0))) \qquad (20) \\
& - l(x_0, \hat{u}_0(x_0)).
\end{aligned}$$

Expression (18) is equal to the sub-optimal cost $J(\hat{u}(x_0))$ at x_0 and therefore less than $J^*(x_0) + \gamma(||x_0||)$ by assumption and the sum of (19) and (20) is non-positive by the assumption that the decay rate of V_N is greater than the stage cost within the set $\mathcal{X}_F$. Therefore, we get the inequality (by the convexity of l, $l(0,0) = 0$, and $\gamma(||x||) \leq l(x, 0)$ for all x by assumption)

$$\begin{aligned}
J^*(x_1) \leq & J^*(x_0) + \gamma(||x_0||) - l(x_0, \hat{u}_0(x_0) \\
\leq & J^*(x_0) + \gamma(||x_0||) - l(x_0, 0) \\
\leq & J^*(x_0)
\end{aligned}$$

which demonstrates that the optimal cost function J^* is a Lyapunov function for the approximate closed loop system $x^+ = Ax + B\hat{u}_0(x)$. ■

Remark 2: We point out that a similar result to Theorem 1 can be found in the work [28]. Under a slight modification to the assumptions, i.e. that the gamma function used in Theorem 1 be strictly less than the gamma function that lower bounds the stage cost, Theorem 16 in [28] can be used to guarantee asymptotic stability for the approximate closed-loop system. With respect to this result, it is important to note that this additional assumption for asymptotic stability is always satisfied in Theorem 2 and Algorithm 2 when the specified threshold is $\epsilon \in [0, 1)$. Further, in the event that the linear system of interest has noise, Theorem 9 in [28] provides Input-to-state stability under certain mild assumptions.

Theorem 1 and Lemma 4 lead us to our main stability and performance guarantee. Specifically, the following theorem states that the sub-optimality of a hypercubic region, in terms of a specified threshold ϵ, can be evaluated by solving a convex optimization problem.

Theorem 2 (Hierarchical Stability): The optimal cost function $J^*(x)$ satisfies the Lyapunov criteria of Theorem 1 with $\epsilon \in [0, 1]$ performance loss over a region $R \in R^\delta$ if $err(R) \geq 0$, where $err(R)$ can be computed by solving the convex optimization problem:

$$\begin{aligned}
err(R) = \min_{\lambda, u} \; & V_N(x_N) + \sum_{i=0}^{N-1} l(x_i, u_i) + \epsilon l(x_0, 0) \\
& - \sum_{v \in \text{extr}(R)} J(\hat{u}(v))\lambda_v \\
s.t. \; & \lambda_v \geq 0, \sum_{v \in \text{extr}(R)} \lambda_v = 1, x_0 = \sum_{v \in \text{extr}(R)} v\lambda_v, \\
& x_{i+1} = Ax_i + Bu_i, \; \forall i = 0, \ldots, N-1, \\
& x_i \in \mathcal{X} \; , u_i \in \mathcal{U} \; , x_N \in \mathcal{X}_F.
\end{aligned}$$

Proof: The optimal cost function $J^*(x)$ satisfies the Lyapunov criteria of Theorem 1 if for all $x_0 \in R$

$$J^*(x_0) + \epsilon l(x_0, 0) - J(\hat{u}(x_0)) \geq 0.$$

By Lemma 4, for all $x_0 \in R$

$$\begin{aligned}
J^*(x_0) + \epsilon l(x_0, 0) - J(\hat{u}(x_0)) \geq & J^*(x_0) + \epsilon l(x_0, 0) \\
& - \sum_{v \in \text{extr}(R)} J(\hat{u}(v)) f_v(x_0).
\end{aligned}$$

Further, note that $err(R)$ can be written as

$$\begin{aligned}
err(R) = \min_{\lambda} \; & J^*(x_0) + \epsilon l(x_0, 0) - \sum_{v \in \text{extr}(R)} J(\hat{u}(v))\lambda_v \\
s.t. \; & \lambda_v \geq 0, \sum_{v \in \text{extr}(R)} \lambda_v = 1, x_0 = \sum_{v \in \text{extr}(R)} v\lambda_v.
\end{aligned}$$

By the properties of barycentric coordinates (6), (7), and (8) and because $\sum_{v \in \text{extr}(R)} J(\hat{u}(v))\lambda_v$ is defined on the convex hull $\text{conv}(\{(v, J(\hat{u}(v))) : v \in \text{extr}(R)\})$, for all $x_0 \in R$

$$J^*(x_0) + \epsilon l(x_0, 0) - \sum_{v \in \text{extr}(R)} J(\hat{u}(v)) f_v(x_0) \geq err(R)$$

Therefore, if the worst case error between the optimal cost function and the convex hull of the approximate cost function is less than $\epsilon l(x_0, 0)$ (i.e. $err(R) \geq 0$), then the entire region spanned by the interpolation by barycentric coordinates satisfies the Lyapunov criteria of Theorem 1. ■

Corollary 2: The optimal cost function $J^*(x)$ defined over the set $R^* := \{x \in R : R \in R^\delta, \; err(R) \geq 0\}$ is a Lyapunov function over R^*.

Since the system must also be invariant or feasible for all time, a Lyapunov function alone is insufficient to prove stability for a constrained system. As discussed in [4], since level sets of Lyapunov functions are invariant [29], it is possible to determine an invariant subset of $\mathcal{X} \subseteq \Omega^d$. Let ∂R^* denote the boundary of R^*.

Corollary 3: If $J_{\min} := \min\{J^*(x) : x \in \partial R^*\}$ and R^* contains the origin, then the set

$$I := \{x \in R^* : J^*(x) \leq J_{\min}\}$$

is invariant under the control law $\hat{u}(x)$.

IV. Approximate Explicit Control Law

A. Offline Computation

We now develop a recursive algorithm for the multiresolution approximation of the model predictive control law. The algorithm initializes at a user defined coarse uniform grid, and then proceeds with a dyadic refinement strategy, saving only the points which violate a user defined thresholding law and/or feasibility condition.

Pseudocode for the approximation process is given in Algorithm 2. For state $x \in \mathbb{R}^d$ and control $u \in \mathbb{R}^m$, the model predictive control problem is defined as in (13) and (14) through the choice of system matrices A and B, convex state and control constraints $\mathcal{X}$ and $\mathcal{U}$, terminal set $\mathcal{X}_F$, time horizon N, and cost function defined by the running cost $l(\cdot,\cdot) : \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^m$ and the terminal cost $V_N(\cdot) : \mathbb{R}^d \to \mathbb{R}$. The overall target performance defined by the user is given by the threshold coefficient ϵ. Additionally, a heuristic threshold $\epsilon_J \in [0, \epsilon]$ may also be defined such that only coefficients which result in a cost function error larger than $\epsilon_J l(x, 0)$ are kept. It is important to point out that while ϵ_J can be used by the user to enforce a tighter pointwise approximation error criteria in hopes of speeding up the convergence of Algorithm 2, a decrease in the complexity of the solution cannot be guaranteed in general.

The index set Λ is initialized at level l_0 along with all indices and details. As the grid is refined, Λ stores the levels of resolution k and indices corresponding to the set of hierarchical details which are not discarded. The maximum level of resolution is given as $l_{\max}$, and the maximum level of resolution at the boundary is given by l_{bdry}. Note that as $l_{\max} \to \infty$, $l_{\text{bdry}} \to \infty$, and $\epsilon \to 0$, the approximate cost function and approximate control law converge to the true solutions. Given that in general the true cost function is continuous and nonsmooth, the convergence properties of Algorithm 2 with respect to ϵ, $l_{\max}$, and l_{bdry} cannot be well characterised.

It is necessary to store the hypercubic regions of guaranteed stability R_s, guaranteed infeasibility R_f (i.e. no feasible point exists within any hypercube in R_f), and regions at level l_{bdry} which intersect the boundary of the feasible set R_b, all of which are initialized as the null set. Also, at each $k \in \{l_0, \ldots, l_{\max}\}$, it is necessary to store the set of hypercubic regions R_k which are to be evaluated for feasibility, stability, and performance, and refined if these conditions are not satisfied. The set of stored detail coefficients is given by the set $\mathbf{w}$.

Algorithm 2 Adaptive Hierarchical Approximate MPC

Require: MPC problem (13), Cost (14), performance threshold ϵ, cost threshold ϵ_J, and minimum, maximum, and boundary level arguments l_0, $l_{\max}$, and l_{bdry}.

Ensure: detail coefficients $\mathbf{w}$ and index set Λ such that the approximate control law (16) $\hat{u}(x)$ has guaranteed feasibility, stability, and performance bound ϵ.

1: Initialize the 'active' set $\Lambda = \{(k, i) : i \in I_k, k = l_0\}$
2: Initialize the evaluation set R_k for $k = l_0$, the set of feasible regions $R_s = \emptyset$, the set of infeasible regions $R_f = \emptyset$, and the set of regions intersecting with the boundary $R_b = \emptyset$ at $k = l_{\text{bdry}}$
3: Compute the initial details $\mathbf{w} = \{w_{k,i} : (k, i) \in \Lambda\}$
4: **while** $R_k \neq \emptyset$ and $k \leq l_{\max}$ **do**
5: $R_{\text{update}} = \emptyset$
6: **for all** $R \in R_k$ **do**
7: Check Stability by Theorem (2) and Feasibility by Lemma (3) for current region R
8: Assign $R \in R_k$ to either R_s, R_f, R_b, or R_{update}
9: **end for**
10: Refine the hypercube regions in the update set R_{update} and assign to R_{k+1}
11: Define the set of new vertices in R_{k+1} as Λ_n
12: Compute

$$\mathbf{w}_n = \left\{ \begin{array}{ll} w_{k,\mathbf{i}} : & (k, \mathbf{i}) \in \Lambda_n \ \text{, and for } y = x_{k,\mathbf{i}} \\ & \hat{u}(y) \text{ infeasible, or} \\ & J(\hat{u}(y)) - J^*(y) > \epsilon_J l(y, 0) \end{array} \right\}$$

13: Let $\Lambda_n^* = \{(k, \mathbf{i}) : (k, \mathbf{i}) \in \Lambda_n, w_{k,\mathbf{i}} \in \mathbf{w}_n\}$
14: Update the 'active' index $\Lambda = \Lambda \cup \Lambda_n^*$
15: Update the detail set $\mathbf{w} = \mathbf{w} \cup \mathbf{w}_n$
16: $k = k + 1$
17: **end while**

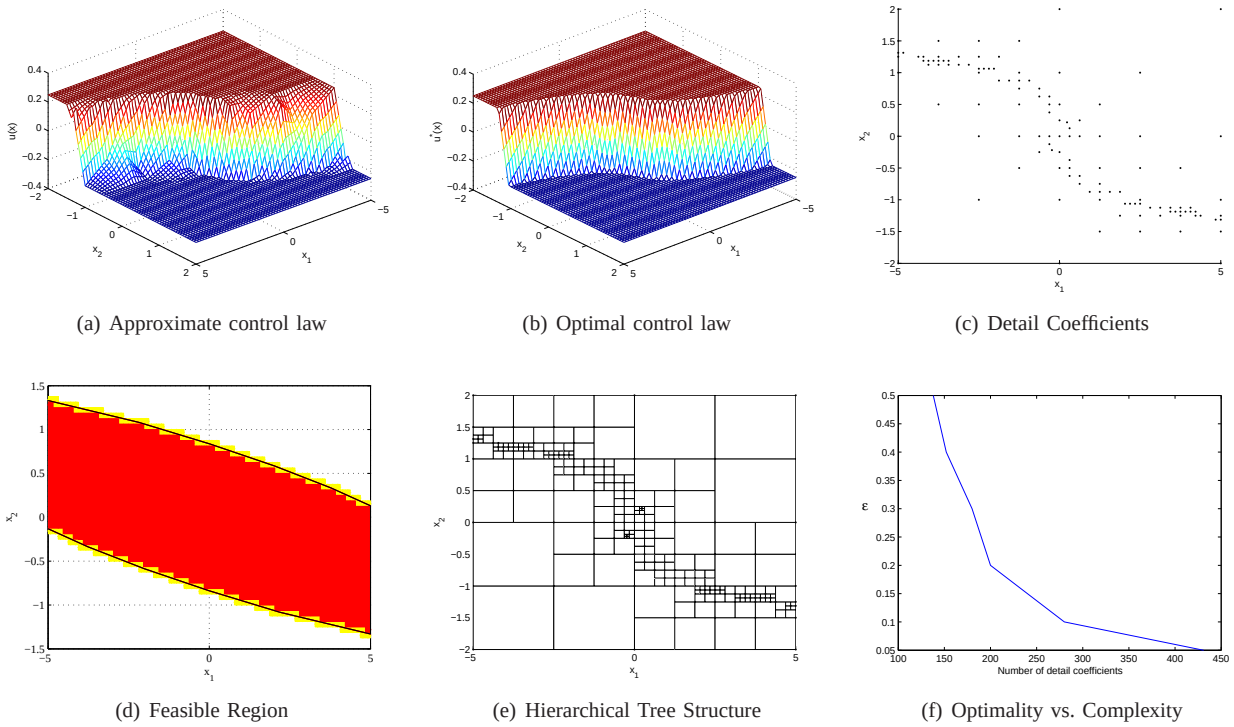
B. Online Computation

The online implementation of the approximate control law consists of evaluating (16) at each step. In the current approach, because of the adaptive nature of Algorithm 2, it is necessary to create a smart data structure for the hierarchical details. With this in mind, we introduce two data structures for the computation of (16) which trade-off speed and storage requirements.

In the first approach, we store the non-zero hierarchical details in a perfect hash function [30] and compute (16) using only the basis functions which are non-zero at state x. Specifically, we are interested in the perfect hash functions introduced in [31] and [32], which result in (minimal) perfect hash tables with constant worst case lookup time. With this approach, the required storage is minimal and the online computation of the control value at x becomes independent of the number of details stored. The overall storage requirement for the approximate control law depends on the hash function, the stored detail values, and the integer index associated with each detail. It is therefore necessary to store approximately n floating point numbers, n integers, and $2n$ binary variables where n is the total number of stored detail values (note that in this approach the storage (i.e. n) is implicitly dependent on the dimension). Further, it can be shown that the worst case number of floating point operations (flops) necessary for each evaluation of the optimal control law is given by the function

$$\begin{aligned} \text{flops}_{hash} \quad = \quad & d \cdot (4 + l_{\max} - l_0) + \text{flops}_{hash} \cdot (l_{\max} - l_0) \\ & \cdot (2^d - 1) + \sum_{i=1}^{d} 2^i \frac{d!}{i!(d-i)!} \\ & \cdot (l_{\max} - l_0) + (d + 1) \cdot 2^d - 1 \end{aligned}$$

where $l_{\max}$ is the finest level of detail, l_0 is the coarsest level



(a) Approximate control law (b) Optimal control law (c) Detail Coefficients

(d) Feasible Region (e) Hierarchical Tree Structure (f) Optimality vs. Complexity

Fig. 5. The approximate control law and optimal control law are shown in (a) and (b) respectively. Notice the sparsity of the required hierarchical details as shown in (c). In (d), the (inner) set denoting the feasible and stable region with bounded performance and the (outer) set denoting the regions which intersect with the boundary are given with respect to the the solid line representing the boundary of the optimal feasible set. In (e), we see the resulting hierarchical tree structure if speed is chosen over storage space. In (f) we illustrate optimality (ϵ) versus approximation complexity.

of detail, d is the dimension, and flops_{hash} is the required flops for the hash function of interest. For example, at a depth of $l_{\max} = 7$ with $l_0 = 0$ and using a hash function requiring 21 flops, we can evaluate the control law in approximately 0.5 μs, 1 μs, and 2 μs for $d = 2$, $d = 3$, and $d = 4$ assuming a processor speed of 1 Gflops/s .

In the second approach, similar to [7] and enabled by Corollary 1, we create a minimal search tree comprising hypercubes aligned to the axis, where each hypercube in the tree has 2^d values associated to it representing (16) at the vertices. With this approach, while there is a slight increase in the necessary storage, the online computation time of (16) is minimal and independent of the number of details stored. The overall storage requirement for the approximate control law is given by the minimum search tree and the stored nodal values at the vertices of each hypercube (approximately $m2^d$ floating point numbers, where the number of hypercubes m is implicitly a function of the saved hierarchical details n and dimension d). Further, it can be shown that the worst case number of floating point operations necessary for each evaluation of the optimal control law is given by the function

$$\text{flops}_{tree} = (d+1)\cdot 2^d + d\cdot(3 + l_{\max} - l_0) - 1$$

where $l_{\max}$ is the finest level of detail, l_0 is the coarsest level of detail, and d is the dimension. For example, at a tree depth of $l_{\max} = 7$ with $l_0 = 0$, we can evaluate the control law in 31 ns, 61 ns, and 119 ns for $d = 2$, $d = 3$, and $d = 4$ assuming a processor speed of 1 Gflops/s.

V. Numerical Examples

A. 2D MPC Example

Consider the simple two-dimensional example:

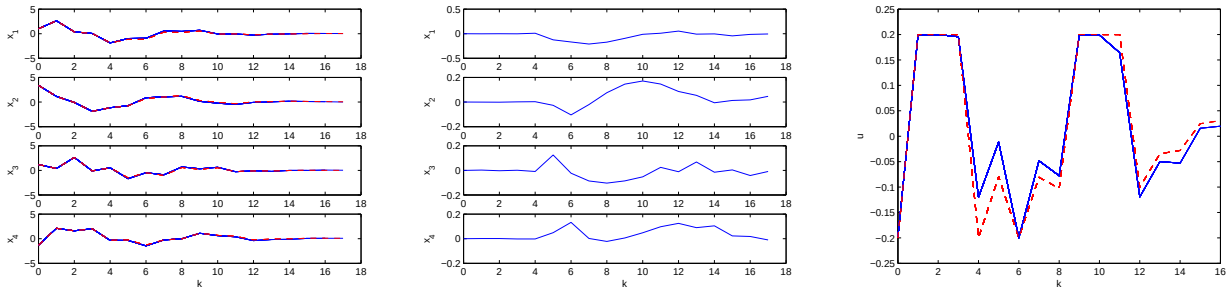
$$x^+ = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0.5 \end{bmatrix} u,$$

with input and state constraints $|u| \leq 0.25$, $||x||_\infty \leq 5$, and a horizon of $N = 10$. The stage cost is taken as $l(x,u) := x'x + 0.01u'u$. The explicit optimal control law, computed using the Multi-Parametric Toolbox [33], requires 221 regions and can be seen in Figure 5. With $\epsilon = 0.5$, $\epsilon_J = 0.25$, $l_0 = 1$, $l_{\text{bdry}} = 5$, and $l_{\max} = 7$, we compute a stabilizing control law using Algorithm 2 that consists of 138 hierarchical details spanning 7 levels. In Figure 5, the resulting control law and feasible regions are shown as well as the performance threshold ϵ versus the required number of non-zero hierarchical details, the hierarchical tree structure, and the map of hierarchical details. Assuming a processor speed of 1 Gflops/s, the explicit multiscale control law for this two-dimensional example can be evaluated in 31 ns or 0.5 μs with the search tree and perfect hash data structures respectively, and even faster if implemented on a parallel processor.

B. 4D MPC Example

Consider the following four-dimensional example:

$$x^+ = Ax + Bu$$



(a) Approximate vs. Optimal Solution (b) Error Trajectory (c) Approximate vs. Optimal Control Trajectory

Fig. 6. (a) Sample four-dimensional system trajectory with the optimal online receding horizon control (red dashed line) and the multiscale approximate receding horizon control (blue solid line). (b) Error trajectory for the sample four-dimensional system trajectory given by the optimal solution minus the approximate solution (Note the difference in scale compared to Figure 6(a)). (c) Sample control trajectory for the four-dimensional system with the optimal online receding horizon control value (red dashed line) and the multiscale approximate receding horizon control value (blue solid line).

with matrices

$$A = \begin{bmatrix} 0.4035 & 0.3704 & 0.2935 & -0.7258 \\ -0.2114 & 0.6405 & -0.6717 & -0.0420 \\ 0.8368 & 0.0175 & -0.2806 & 0.3808 \\ -0.0724 & 0.6001 & 0.5552 & 0.4919 \end{bmatrix},$$
$$B = \begin{bmatrix} 1.6124 & 0.4086 & -1.4512 & -0.6761 \end{bmatrix}^T.$$

The input and state constraints are defined $|u| \leq 0.2$ and $||x||_\infty \leq 5$, and the horizon is set to $N = 17$. The stage cost is taken as $l(x,u) := x'x + 0.2u'u$. Note that the system matrices, optimal control weights, and horizon length were chosen randomly and that the nominal system is stable. In this case, the Multi-Parametric Toolbox [33] computation was terminated (at ≈ 50000 regions) before it was able to obtain an explicit optimal control law with which to compare the approximate solution, therefore we only make comparisons with the online implementation of the MPC solution. With $\epsilon = 1$, $\epsilon_J = 0.25$, $l_0 = 1$, $l_{\text{bdry}} = 3$, and $l_{\max} = 7$, we compute a stabilizing control law using Algorithm 2 that consists of 3633 hierarchical details spanning 7 levels. In Figure 6(a), an example trajectory of the system is shown comparing the optimal online receding horizon solution with the multiscale explicit approximate control law. Figure 6(b) shows the error between the two solutions and Figure 6(c) compares the optimal control input and the sub-optimal control input. Additionally, 10000 Monte Carlo simulations were completed with the initial states drawn randomly from the feasible region according to a uniform distribution. The results included an average open loop performance loss of $\epsilon_{\text{average}} < 0.12$ and an average closed loop cost increase of less than 3 percent. As evident from the Monte Carlo analysis and Figures 6(a), 6(b), and 6(c), the implementation of the approximate control law performs very well. Yet, the most influential benefit of the explicit approximate control law in this case is the online evaluation speed that can be achieved. Assuming a processor speed of 1 Gflops/s, the explicit multiscale control law for this four-dimensional example can be evaluated in 119 ns or 2.5 μs with the search tree and perfect hash data structures respectively, and even faster if implemented on a parallel processor.

VI. Conclusion

The approximate explicit MPC method we have presented consists of a simple hierarchical gridding scheme which is easy to implement. The approach approximates the optimal control law directly, and because of the basis functions used to build the function approximation, can provide guaranteed stability, feasibility, and bounds on the performance. The ability to guarantee a level of accuracy between grid points (the hypercubes) enables an adaptive approach based on thresholding which can lead to a sparse representation of the explicit control law that is fast to implement. Future considerations include reducing the conservative nature of the verification procedure and extending the results to systems with nonlinear dynamics.

References

[1] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, pp. 3–20, January 2002.

[2] T. A. Johansen and I. Petersen, "On explicit suboptimal LQR with state and input constraints," in *in Proc. IEEE Conf. Decision and Control*, pp. 05–6, 2000.

[3] M. M. Seron, G. C. Goodwin, and J. A. D. Dona, "Geometry of model predictive control including bounded and stochastic disturbances under state and input constraints," tech. rep., University of Newcastle, 2000.

[4] C. N. Jones, M. Barić, and M. Morari, "Multiparametric linear programming with applications to control," *European Journal of Control*, vol. 13, no. 2-3, pp. 152–170, 2007.

[5] A. Alessio and A. Bemporad, "A survey on explicit model predictive control," in *Nonlinear Model Predictive Control* (L. Magni, D. Raimondo, and F. Allgwer, eds.), vol. 384 of *Lecture Notes in Control and Information Sciences*, pp. 345–369, Springer Berlin Heidelberg, 2009.

[6] A. Bemporad and C. Filippi, "An algorithm for approximate multiparametric convex programming," *Comput. Optim. Appl.*, vol. 35, no. 1, pp. 87–108, 2006.

[7] T. A. Johansen and R. Grancharova, "Approximate explicit constrained linear model predictive control via orthogonal search tree," *IEEE Trans. Automatic Control*, vol. 48, pp. 810–815, 2003.

[8] B. Lincoln and A. Rantzer, "Relaxing dynamic programming," *Automatic Control, IEEE Transactions on*, vol. 51, no. 8, pp. 1249–1260, 2006.

[9] A. Bemporad and C. Filippi, "Suboptimal explicit receding horizon control via approximate multiparametric quadratic programming," *Journal of Optimization Theory and Applications*, vol. 117, pp. 9–38(30), April 2003.

[10] C. Jones and M. Morari, "The Double Description Method for the Approximation of Explicit MPC Control Laws," in *Conference on Decision and Control, CDC*, 2008.

[11] I. Daubechies, "Orthonormal bases of compactly supported bases," *Communications On Pure and Applied Mathematics*, vol. 41, pp. 909–996, 1988.

[12] I. Daubechies, "Orthonormal bases of compactly supported wavelets ii: variations on a theme," *SIAM J. Math. Anal.*, vol. 24, no. 2, pp. 499–519, 1993.
[13] I. Daubechies, *Ten lectures on wavelets*, vol. 61 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM), 1992.
[14] A. Cohen, I. Daubechies, and J.-C. Feauveau, "Biorthogonal bases of compactly supported wavelets," *Comm. pure and Appl. Math.*, vol. 45, pp. 485–560, 1992.
[15] G. Deslauriers and S. Dubuc, "Symmetric iterative interpolation processes," *Constructive Approximation*, vol. 5, pp. 49–68, 1989.
[16] D. L. Donoho, "Interpolating wavelet transforms," tech. rep., Stanford, 1992.
[17] S. Ferrari, M. Maggioni, and N. Borghese, "Multiscale approximation with hierarchical radial basis functions networks," *Neural Networks, IEEE Transactions on*, vol. 15, pp. 178 –188, jan. 2004.
[18] R. A. DeVore, "Nonlinear approximation," *Acta Numerica*, vol. 7, no. -1, pp. 51–150, 1998.
[19] H.-J. Bungartz and M. Griebel, "Sparse grids," *Acta Numerica*, vol. 13, no. 1, pp. 147–269, 2004.
[20] S. G. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 674–693, 1989.
[21] M. Griebel and F. Koster, "Multiscale methods for the simulation of turbulent flows," in *Numerical Flow Simulation III* (E. Hirschel, ed.), vol. 82 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, pp. 203–214, Springer-Verlag, 2003.
[22] M. Griebel and F. Koster, "Adaptive wavelet solvers for the unsteady incompressible navier-stokes equations," in *Advanced Mathematical Theories in Fluid Mechanics*, pp. 67–118, Springer, 2000.
[23] M. Floater, K. Hormann, and G. Kós, "A general construction of barycentric coordinates over convex polygons," *Advances in Computational Mathematics*, vol. 24, pp. 311–331, January 2006.
[24] S. Summers, C. Jones, J. Lygeros, and M. Morari, "A Multiscale Approximation Scheme for Explicit Model Predictive Control with Stability, Feasibility, and Performance Guarantees," in *IEEE Conference on Decision and Control*, (Shanghai, China), Dec. 2009.
[25] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, pp. 789–814, June 2000.
[26] C. N. Jones and M. Morari, "Polytopic approximation of explicit model predictive controllers," *IEEE Trans. Automatic Control*, 2010. To Appear.
[27] P. O. M. Scokaert, D. Q. Mayne, and J. B. Rawlings, "Suboptimal model predictive control (feasibility implies stability)," *Automatic Control, IEEE Transactions on*, vol. 44, no. 3, pp. 648–654, 1999.
[28] M. Lazar and W. Heemels, "Predictive control of hybrid systems: Input-to-state stability results for sub-optimal solutions," *Automatica*, vol. 45, no. 1, pp. 180 – 185, 2009.
[29] F. Blanchini, "Set invariance in control - a survey," *Automatica*, vol. 35, no. 11, pp. 1747–1768, 1999.
[30] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Second Edition*. The MIT Press, September 2001.
[31] Z. J. Czech, G. Havas, and B. S. Majewski, "An optimal algorithm for generating minimal perfect hash functions," *Information Processing Letters*, vol. 43, pp. 257–264, 1992.
[32] F. C. Botelho, R. Pagh, and N. Ziviani, "Simple and space-efficient minimal perfect hash functions," in *In Proc. of the 10th Intl. Workshop on Data Structures and Algorithms*, pp. 139–150, Springer LNCS, 2007.
[33] M. Kvasnica, P. Grieder, and M. Baotić, "Multi-Parametric Toolbox (MPT)," 2004.

Sean Summers received the B.S. degree in Aerospace Engineering in 2004 and the M.S. degree in Mechanical Engineering in 2007, both from the department of Mechanical and Aerospace Engineering at the University of California, San Diego. He is currently a Ph.D. candidate in the Department of Information Technology and Electrical Engineering, Swiss Federal Institute of Technology (ETH), Zurich. His research interests include modeling and control of stochastic hybrid systems, reachability analysis, model predictive control, and applications of control theory in systems biology.

Colin N. Jones is a senior researcher (Oberassistent) at the Automatic Control Laboratory at the ETH in Zurich. He obtained a PhD in 2005 from the University of Cambridge for his work on polyhedral computational methods for constrained control. Prior to that, he was at the University of British Columbia in Canada, where he took a BASc and MASc in Electrical Engineering and Mathematics. Colin has worked in a variety of industrial roles, ranging from control of heating, ventilation and air conditioning to hydrometallurgical processes. He is co-founder of Apex Optimization; a custom optimization house that focuses on human resource scheduling. His current research interests are in the areas of high-speed predictive control, optimization, energy management and optimal scheduling.

John Lygeros received a B.Eng. degree in Electrical Engineering and an M.Sc. degree in Automatic Control from Imperial College, London, U.K., in 1990 and 1991, respectively. He then received a Ph.D. degree from the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, California, U.S.A. in 1996. He held postdoctoral research appointments with the National Automated Highway Systems Consortium, the Massachusetts Institute of Technology, and the University of California, Berkeley. In parallel, he was also a part-time Research Engineer with SRI International, Menlo Park, California, U.S.A. and a Visiting Professor with the Department of Mathematics, Universite de Bretagne Occidentale, Brest, France. Between 2000 and 2003, he was a university lecturer with the Department of Engineering, University of Cambridge, Cambridge, U.K. and a fellow of Churchill College. Between 2003 and 2006, he was an assistant professor with the Department of Electrical and Computer Engineering, University of Patras, Patras, Greece. In July 2006, he joined the Automatic Control Laboratory, ETH Zurich, Switzerland, where he is currently serving as the Head of the laboratory. His research interests include modeling, analysis, and control of hierarchical, hybrid and stochastic systems with applications to biochemical networks and large-scale engineering systems such as power networks and air traffic management.

Manfred Morari was appointed head of the Department of Information Technology and Electrical Engineering at ETH Zurich in 2009. He was head of the Automatic Control Laboratory from 1994 to 2008. Before that he was the McCollum-Corcoran Professor of Chemical Engineering and Executive Officer for Control and Dynamical Systems at the California Institute of Technology. He obtained the diploma from ETH Zurich and the Ph.D. from the University of Minnesota, both in chemical engineering. His interests are in hybrid systems and the control of biomedical systems. In recognition of his research contributions he received numerous awards, among them the Donald P. Eckman Award and the John R. Ragazzini Award of the Automatic Control Council, the Allan P. Colburn Award and the Professional Progress Award of the AIChE, the Curtis W. McGraw Research Award of the ASEE, Doctor Honoris Causa from Babes-Bolyai University, Fellow of IEEE, the IEEE Control Systems Field Award, and was elected to the National Academy of Engineering (U.S.). Manfred Morari has held appointments with Exxon and ICI plc and serves on the technical advisory boards of several major corporations.

Preprint submitted to IEEE Transactions on Automatic Control. Received: April 19, 2011 09:58:22 PST