Introducing INTSOSTOOLS: A SOSTOOLS plug-in for integral inequalities

Giorgio Valmorbida and Antonis Papachristodoulou

***Abstract*— INTSOSTOOLS is a plug-in for the MATLAB toolbox SOSTOOLS for the formulation of optimization problems with constraints of the form of one-dimensional integral inequalities. For polynomial problem data, semidefinite programming can be used to solve the underlying optimization problems. This paper details a method for checking integral inequalities and shows how to use INTSOSTOOLS to cast optimization problems with integral constraints and linear objective functions. Two examples illustrate the potential of the proposed method and of the developed software.**

I. Introduction

Sum-of-Squares (SOS) software assists in casting SOS programmes (SOSP), optimization problems of the form

$$\begin{array}{l}\text{maximize } \lambda \\ \text{subject to } f(x) - \lambda \in \Sigma[x],\end{array} \tag{1}$$

as semidefinite programmes (SDP)[3], [4], [5]. The existence of SOS software packages has promoted the use of SOSP to solve control-related problems. Among the available software packages that handle SOSPs, SOSTOOLS is a free software package running on MATLAB[1] released one decade ago [1], [2]. An update to SOSTOOLS [6], has been recently released and includes: 1) an interface to MATLAB symbolic engine MuPad [7]; 2) the formulation of SDPs for an extended set of SDP solvers; 3) matrix polynomials and matrix sum-of-squares constraints manipulation. Another package that allows for the formulation of SOS programs (SOSP) is Yalmip [8], [9]. The dual problem to the optimization problem (2) is defined by the generalized problem of moments. Software to solve the problem of moments is also available [10]. These software packages handle polynomial objects using MATLAB symbolic toolbox or efficient *ad-hoc* toolboxes for polynomial manipulation, therefore helping the user to define SOS constraints and extract solutions to the underlying SDPs.

Among the successful application of SOSP to control problems, we may cite the Lyapunov analysis for polynomial vector fields in both local and global contexts. In the local case, non-convex formulations [11] requiring iterative solutions, allowed for computation of stability region estimates defined by level sets of polynomial Lyapunov functions (LF). These studies, however, did not consider maximal Lyapunov functions characterizing the *exact* stability region. As defined in [12] these functions "blow up" at the boundary of the region of attraction, which is not possible for polynomial LFs. An approach based on invariant sets has been presented in [13]. In the global case, a convex formulation is available, however there exist stable polynomial vector fields without polynomial Lyapunov functions certificates [14]. Parametric uncertainty analysis with SOSP has been emphasized in the survey paper [15]. The design problem for polynomial vector fields with SOS polynomials has been studied in [16], [17], [18], [19], where, in contrast to the linear systems result in [20], the proposed approaches provide a convex formulation in the global case only for particular instances. A drawback of SOS techniques is the fact that the dimensions of the associated SDP (number of variables and number of constraints) have combinatorial grow defined by the number of variables and the degree of the constaints. In order to alleviate the poor scalability of SOSPs, promising research directions, where polynomial positivity query is obtained by solving Linear and Second Order Cone Programmes, have been proposed [21].

In this paper we are interested in obtaining a (numerically) tractable formulation for optimization problems with linear objective functions subject to one-dimensional integral inequalities as, for instance,

$$\begin{array}{l}\text{maximize } \lambda \\ \text{subject to } \int_0^1 \left[F(\theta, u(\theta)) - \lambda\right] \mathrm{d}\theta \geq 0.\end{array} \tag{2}$$

and to present the main features of INTSOSTOOLS, an SOSTOOLS plug-in.

Two examples illustrate the proposed method: one aiming to compute bounds for the Wirtinger inequality of periodic functions and a second one, which illustrates the $\mathcal{L}_2$ stability analysis of a Partial Differential Equation. We highlight important features of the INTSOSTOOLS and illustrate how integrands, which are polynomial on the dependent variables, are handled. The paper illustrates the use of INTSOSTOOLS by detailing how to set a problem with integral constraints paralleling the structure of [1] where SOSTOOLS was introduced.

The study of integral inequalities is related to Lyapunov function structures studied for delay differential equations [22]. For this class of systems SOS-based solutions to Lyapunov functional inequalities are available [23]. These results contrast with this paper since it allows to consider more general inequalities and more general boundary values.

Notation: Let $\mathbb{R}, \mathbb{R}_{\geq 0}, \mathbb{R}_{>0}$ and $\mathbb{R}^n$ denote the field of reals, non-negative reals, positive reals and the n-dimensional

The authors are with the Department of Engineering Science, University of Oxford, Oxford, OX1 3PJ, UK e-mail: ({giorgio.valmorbida, antonis}@eng.ox.ac.uk). G. Valmorbida is also affiliated to Somerville College, University of Oxford, Oxford, U.K. A. Papachristodoulou was supported in part by the Engineering and Physical Sciences Research Council projects EP/J012041/1, EP/I031944/1 and EP/J010537/1.

[1]A registered trademark of MathWorks Inc.

Euclidean space respectively. The set of symmetric $n \times n$ matrices is denoted by $\mathbb{S}^n$. The notation M^T denotes the transpose of matrix M. The function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be positive definite if $f(x) > 0$ for all non-zero $x \in \mathbb{R}^n$, similarly if $f(x) \geq 0$ for all $x \in \mathbb{R}^n$ then f is said to be positive semidefinite. For $x \in \mathbb{R}^m$ the ring of polynomials in m variables is denoted by $\mathcal{R}[x_1, \dots, x_m]$, when it is clear from context we will use $\mathcal{R}[x]$. A polynomial, $p(x)$, is said to be a sum of squares (a subset of the polynomial ring) if there exists a finite set of polynomials $g_1(x), \dots, g_p(x)$ such that $p(x) = \sum_{i=1}^{p} g_i^2(x)$. The set of sum-of-squares, (SOS), polynomials in x is denoted by $\Sigma[x_1, \dots, x_m]$ which can be abbreviated to $\Sigma[x]$. Equivalently $p(x)$ is SOS if there exists a positive semidefinite matrix Q such that $p(x) = Z^T(x) Q Z(x)$ where $Z(x)$ is an appropriately chosen vector of monomials. For a monomial vector $Z(x)$ containing all monomials up to degree d, its cardinality is denoted $\sigma(m, d) := \frac{m-d!}{(n-1)!d!}$ Note that the search for Q can be formulated as a semidefinite program and thus the search for a sum-of-squares decomposition is a convex optimization problem [4]. It may be necessary at times to specify the number of variables and a degree bound d on a (SOS) polynomial, in which case we use the notation ($\Sigma_{m,d}[x]$) $\mathcal{R}_{m,d}[x]$. The space of k-times continuously differentiable functions defined on Ω is denoted by $\mathcal{C}^k(\Omega)$. For a multi-variable function $f(x, y)$, we use the notation $f \in \mathcal{C}^k[x]$ to show k-times continuously differentiability of f with respect to variable x. If $p \in \mathcal{C}^1(\Omega)$, then $\partial_x p$ denotes the derivative of p with respect to variable $x \in \Omega$, i.e. $\partial_x := \frac{\partial}{\partial x}$. We also adopt the Schwartz's multi-index notation. For $u \in \mathcal{C}^\alpha(\Omega)$, $\alpha \in \mathbb{N}_0^n$, define

$$D^\alpha u := \left(u_1, \partial_x u_1, \dots, \partial_x^{\alpha_1} u_1, \dots, u_n, \partial_x u_n, \dots, \partial_x^{\alpha_n} u_n\right).$$

II. Motivating Examples

We present two problems leading to optimization problems subject to integral inequalities.

A. Wirtinger inequality

A well-studied inequality involving integrals is given by the Wirtinger inequality [24, p185]. For periodic functions with zero average, this inequality provides a bound on the integral of the square of the function in terms of the integral of the square of the derivative of the function. The problem of computing such a bound can be formulated as

$$\begin{array}{ll} \text{minimize} & \nu \\ \text{subject to} & \nu \int_0^1 (u'(\theta))^2 \mathrm{d}\theta \geq \int_0^1 (u(\theta))^2 \mathrm{d}\theta \\ & \int_0^1 u(\theta) \mathrm{d}\theta = 0 \\ & u(1) = u(0) = k. \end{array} \tag{3}$$

In order to transform the zero-average integral constraint a boundary constraint, we rewrite the above by problem in two steps: the first step consists of computing the value at the boundaries, k, of the minimizer function $u(\theta)$ and the second step consists of a change of variables.

Lemma 1: The minimizer $u(\theta)$ of (3) satisfies $u(1) = u(0) = 0$.

Proof: Take $\bar{u} = u - k$ so that $\bar{u}(1) = \bar{u}(0) = 0$. We have

$$\begin{aligned} \int_0^1 \bar{u}^2 \mathrm{d}\theta &= \int_0^1 (u - k)^2 \mathrm{d}\theta \\ &= \int_0^1 \left(u^2 - 2ku\right) \mathrm{d}\theta + \int_0^1 k^2 \mathrm{d}\theta = \int_0^1 u^2 \mathrm{d}\theta + k^2 \end{aligned} \tag{4}$$

we also have $\bar{u}' = u'$, which gives

$$\int_0^1 \bar{u}'^2 \mathrm{d}\theta = \int_0^1 u'^2 \mathrm{d}\theta. \tag{5}$$

Therefore, in variable $\bar{u}$, inequality in (3) is expressed as

$$\nu \int_0^1 \bar{u}'^2 \mathrm{d}\theta \geq \int_0^1 \bar{u}^2 \mathrm{d}\theta - \int_0^1 k^2 \mathrm{d}\theta = \int_0^1 \bar{u}^2 \mathrm{d}\theta - k^2. \tag{6}$$

Hence if it holds for $k = 0$ then it holds for $k \neq 0$, which shows that the inequality is tighter with $k = 0$, that is, when $u(0) = u(1) = 0$. ■

Now let us introduce the variable

$$v(x) := \int_0^x u(\theta) \mathrm{d}\theta,$$

yielding

$$v'(x) = u(x), \quad v''(x) = u'(x). \tag{7}$$

From its definition and from (3) we have

$$v(0) = v(1) = 0.$$

According to Lemma 1, problem (3) is solved with $u(1) = u(0) = 0$ which, with (7), gives

$$v'(1) = v'(0) = 0.$$

We can then write (3) in terms of variable v as follows

$$\begin{array}{ll} \text{minimize} & \nu \\ \text{subject to} & \int_0^1 \nu (v'')^2 - (v')^2 \mathrm{d}\theta \geq 0 \\ & v(1) = v(0) = 0 \\ & v'(1) = v'(0) = 0. \end{array} \tag{8}$$

B. $\mathcal{L}_2$ stability of the heat equation with a reaction term

Consider the heat equation

$$\begin{array}{ll} \partial_t u(t, x) = \partial_x^2 u(t, x) + \mu u(t, x) & x \in \Omega = [0, 1] \\ u(t, 0) = u(t, 1) = 0. \end{array} \tag{9}$$

with $\mu > 0$. We want to prove the $\mathcal{L}_2$-norm stability of the solutions to (9) with an equivalent norm defined by the the square root of the weighted $\mathcal{L}_2$-norm as

$$V(u) = \frac{1}{2} \int_0^1 p(\theta) u^2(t, \theta)\ \mathrm{d}\theta. \tag{10}$$

By imposing the positivity condition $V(u) > 0$ and the Lyapunov derivative condition, $\frac{dV(u)}{dt} < 0$ we obtain the following feasibility problem

$$\begin{array}{ll} \text{find} & p(\theta) \\ \text{subject to} & \frac{1}{2} \int_0^1 p(\theta) u^2(t, \theta)\ \mathrm{d}\theta > 0, \\ & \int_0^1 p(\theta) (u(t, \theta) \partial_\theta^2 u(t, \theta) + \lambda u^2(t, \theta))\ \mathrm{d}\theta > 0 \\ & u(t, 0) = u(t, 1) = 0. \end{array} \tag{11}$$

III. Integral Inequalities

The results in this section were introduced in [25] and present a formulation to solve integral inequalities. Conditions for an integral inequality to hold are formulated using a quadratic-like representation of the integrand. The Fundamental Theorem of Calculus (FTC) is used to relate the dependent variables to their values at the boundaries. As a result, the positivity of the integral can be checked via the positivity of a quadratic-like form in the domain of integration and the positivity of a quadratic-like form on the boundary values of the dependent variables.

We present a formulation to verify integral inequalities of the form

$$\phi(D^{\alpha-1}u(1), D^{\alpha-1}u(0)) + \int_0^1 f(\theta, D^{\alpha}u(\theta))\mathrm{d}\theta \geq 0. \quad (12)$$

where $\phi(x,y) \in \mathcal{R}[x,y]$ and $f(\cdot,x) \in \mathcal{R}[x]$, that is ϕ is a polynomial with real coefficients on both arguments and $f(\cdot,x)$ is a polynomial on its second argument for any value of the first argument and the dependent variable $u : [0,1] \to \mathbb{R}$ satisfies

$$u(\theta) \in \mathcal{U}_s(Q) := \left\{ u(\theta) \mid Q \begin{bmatrix} D^{\alpha-1}u(1) \\ D^{\alpha-1}u(0) \end{bmatrix} = 0 \right\}. \quad (13)$$

The above inequality can then be written as

$$\eta\left(\begin{bmatrix} D^{\alpha-1}u(1) \\ D^{\alpha-1}u(0) \end{bmatrix}\right)^T F_c \eta\left(\begin{bmatrix} D^{\alpha-1}u(1) \\ D^{\alpha-1}u(0) \end{bmatrix}\right) + \int_0^1 \xi^T(D^{\alpha}u(\theta))F(\theta)\xi(D^{\alpha}u(\theta))\ \mathrm{d}\theta \geq 0. \quad (14)$$

with $F_c \in \mathbb{S}^{\sigma(n_{\alpha}-1,d_{\eta})}$, $n_{\alpha-1} = \sum_{i=1}^{n}(\alpha_i - 1)$, $i = 0,1$, $F : [0,1] \to \mathbb{S}^{\sigma(n_{\alpha},d_{\xi})}$, $n_{\alpha} = \sum_{i=1}^{n}\alpha_i$ and η, ξ are vectors of monomials of degree d_{η} and d_{ξ} on $[D^{\alpha-1}u(1)\ D^{\alpha-1}u(0)]$ and $D^{\alpha}u(\theta)$ respectively.

The lemma below is a straightforward application of the FTC and relates the values of u at the boundary ($u(1)$ and $u(0)$) to the integrand by introducing extra terms in the integral in (14). These terms allow the verification of the inequality in the subspace defined in terms of the boundary values (13).

Lemma 2: Consider a matrix function $H(\theta) \in \mathcal{C}^1[\theta]$, $H : [0,1] \to \mathbb{S}^{\sigma(n_{\alpha-1},d_{\zeta})}$ and a vector of monomials $\zeta(D^{\alpha-1}u(\theta)) : [0,1] \to \mathbb{R}^{\sigma(n_{\alpha-1},d_{\zeta})}$. We have

$$\begin{aligned} &\zeta^T(D^{\alpha-1}u(1))H(1)\zeta(D^{\alpha-1}u(1)) \\ &\qquad - \zeta^T(D^{\alpha-1}u(0))H(0)\zeta(D^{\alpha-1}u(0)) \\ = &\int_0^1 \tfrac{\mathrm{d}}{\mathrm{d}\theta}\left[\zeta^T(D^{\alpha-1}u(\theta))H(\theta)\zeta(D^{\alpha-1}u(\theta))\right]\ \mathrm{d}\theta \\ = &\int_0^1 \zeta^T(D^{\alpha-1}u(\theta))\tfrac{dH(\theta)}{d\theta}\zeta(D^{\alpha-1}u(\theta)) \\ &\qquad + 2\zeta^T(D^{\alpha-1}u(\theta))H(\theta)\tfrac{d\zeta(D^{\alpha}u(\theta))}{d\theta}\mathrm{d}\theta. \end{aligned} \quad (15)$$

In order to write terms in (15) in a compact form, assume the degree of vector of monomials ζ satisfies $d_{\zeta} \leq d_{\xi}$ and define the matrix function $\bar{H}(\theta) \in \mathcal{C}^1[\theta]$, $\bar{H} : [0,1] \to \mathbb{S}^{\sigma(n_{\alpha-1},d_{\zeta})}$ to be the matrix satisfying

$$\begin{aligned} &\xi^T(D^{\alpha}u(\theta))\bar{H}(\theta)\xi(D^{\alpha}u(\theta)) := \\ &\qquad \zeta^T(D^{\alpha-1}u(\theta))\frac{dH(\theta)}{d\theta}\zeta(D^{\alpha-1}u(\theta)) \\ &\qquad + 2\zeta^T(D^{\alpha-1}u(\theta))H(\theta)\frac{d\zeta(D^{\alpha}u(\theta))}{d\theta}. \end{aligned} \quad (16)$$

Therefore (15) gives

$$\begin{aligned} 0 = &\int_0^1 \xi^T(D^{\alpha}u(\theta))\bar{H}(\theta)\xi(D^{\alpha}u(\theta))\ \mathrm{d}\theta \\ &- \left[\zeta^T(D^{\alpha-1}u(1))H(1)\zeta(D^{\alpha-1}u(1)) \right. \\ &\left. -\zeta^T(D^{\alpha-1}u(0))H(0)\zeta(D^{\alpha-1}u(0))\right], \end{aligned} \quad (17)$$

which, added to (14), gives

$$\begin{aligned} &\int_0^1 \xi^T(D^{\alpha}u(\theta))\left[F(\theta) + \bar{H}(\theta)\right]\xi(D^{\alpha}u(\theta))\ \mathrm{d}\theta \\ &+ \eta^T\left(\begin{bmatrix} D^{\alpha-1}u(1) \\ D^{\alpha-1}u(0) \end{bmatrix}\right) F_c \eta\left(\begin{bmatrix} D^{\alpha-1}u(1) \\ D^{\alpha-1}u(0) \end{bmatrix}\right) \\ &- \left[\zeta^T(D^{\alpha-1}u(1))H(1)\zeta(D^{\alpha-1}u(1)) \right. \\ &\left. -\zeta^T(D^{\alpha-1}u(0))H(0)\zeta(D^{\alpha-1}u(0))\right] \geq 0. \end{aligned} \quad (18)$$

With the above expression we can now formulate conditions to verify inequality (14) for u satisfying (13) as follows.

Proposition 1: If there exists a matrix $H(\theta) \in \mathcal{C}^1[\theta]$, $H : [0,1] \to \mathbb{S}^{\sigma(n_{\alpha-1},d_{\zeta})}$ satisfying

$$F(\theta) + \bar{H}(\theta) \geq 0,\ \forall\theta \in [0,1], \quad (19a)$$

and

$$\begin{aligned} &\eta^T\left(\begin{bmatrix} D^{\alpha-1}u(1) \\ D^{\alpha-1}u(0) \end{bmatrix}\right) F_c \eta\left(\begin{bmatrix} D^{\alpha-1}u(1) \\ D^{\alpha-1}u(0) \end{bmatrix}\right) \\ &- \left[\zeta^T(D^{\alpha-1}u(1))H(1)\zeta(D^{\alpha-1}u(1)) \right. \\ &\left. -\zeta^T(D^{\alpha-1}u(0))H(0)\zeta(D^{\alpha-1}u(0))\right] \geq 0 \\ &\qquad \forall u \in \mathcal{U}_s(Q) \end{aligned} \quad (19b)$$

then (14) holds for all $u \in \mathcal{U}_s(Q)$.

Remark 1: The above proposition formulates the test of (14) as the test of positivity of the matrix $F(\theta)+\bar{H}(\theta)$ over the domain $\theta \in [0,1]$ and a polynomial inequality involving the values at the boundary of the domain which is performed for the set of dependent variables belonging to a subspace defined by $\mathcal{U}_s(Q)$ as in (13). Notice that (19a) and (19b) are related via matrix H introduced in Lemma 2 (which defines the entries of $\bar{H}$).

Proposition 1 defines a test for the integral expression by relating the integral term and the boundary terms via the entries of matrix H and testing separately the boundary inequality and the integrand term inequality. The main functions of the INTSOSTOOLS plug-in that we introduce next automate the computation of the matrix $H(\theta)$, given the terms of (14) and formulate the two inequalities (19) in the subspace (13).

The following example illustrates a particular case of the integral inequalities presented above.

a) Example (Quadratic Functions): We now clarify how to formulate the inequalities (19) for given F_c and $F(\theta)$, in the case of quadratic expressions, i.e. expressions in which both vector of monomials η and ξ in (14) are of degree one, $n = 1$ and $\alpha = 2$. In this case we have

$$\eta = \begin{bmatrix} u(1) \\ \dfrac{d}{dx}u(1) \\ u(0) \\ \dfrac{d}{dx}u(0) \end{bmatrix}; \xi(D^2u(x)) = \begin{bmatrix} u(x) \\ \dfrac{d}{dx}u(x) \\ \dfrac{d^2}{dx^2}u(x) \end{bmatrix}. \qquad (20)$$

The formulation of (19a) is given in terms of the matrix $\bar{H}(\theta)$ which is obtained from the identity

$$\int_0^1 \frac{\mathrm{d}}{\mathrm{d}\theta}\left[\xi^T(D^{\alpha-1}u(\theta))H(\theta)\xi(D^{\alpha-1}u(\theta))\right] \,\mathrm{d}\theta$$
$$= \int_0^1 \xi^T(D^\alpha u(\theta))\bar{H}(\theta)\xi(D^\alpha u(\theta))\,\mathrm{d}\theta$$

in (15). For quadratic expressions we have

$$\bar{H}(\theta) =$$
$$\begin{bmatrix} \frac{\mathrm{d}}{\mathrm{d}\theta}H_{11}(\theta) & \frac{\mathrm{d}}{\mathrm{d}\theta}H_{12}(\theta) + H_{11}(\theta) & H_{12}(\theta) \\ \frac{\mathrm{d}}{\mathrm{d}\theta}H_{12}(\theta) + H_{11}(\theta) & \frac{\mathrm{d}}{\mathrm{d}\theta}H_{22}(\theta) + 2H_{12}(\theta) & H_{22}(\theta) \\ H_{12}(\theta) & H_{22}(\theta) & 0 \end{bmatrix}.$$

And, for (19b), we have

$$H(1) = \begin{bmatrix} H_{11}(1) & H_{12}(1) \\ H_{12}(1) & H_{22}(1) \end{bmatrix},$$

$$H(0) = \begin{bmatrix} H_{11}(0) & H_{12}(0) \\ H_{12}(0) & H_{22}(0) \end{bmatrix}.$$

A. Polynomial inequalities

In the above formulation we have only assumed polynomial dependence of f in (12) on the dependent variable u and its derivatives. For the case of polynomial dependence of f (and therefore of F in (14)) on variable θ we can impose a polynomial dependence of matrix $\bar{H}$ on θ to obtain a test for (19a) in terms of SOS polynomial matrices. The Positivstellensatz [26], defines a necessary and sufficient condition for the (19a) to hold in terms of the existence of a sum-of-squares matrix multiplier $M(\theta)$ such that

$$F(\theta) + \bar{H}(\theta) + (\theta^2 - \theta)M(\theta) \in \Sigma[\theta],\ M(\theta) \in \Sigma[\theta]. \quad (21)$$

In the remainder of the paper we assume that the integrands are polynomials in both the dependent and the independent variables.

IV. INTSOSTOOLS

A systematic procedure to solve (12) is to formulate SOSPs with constraints (21) and (19b). INTSOSTOOLS is a plug-in to SOSTOOLS assisting in the formulation of (19). To define and solve a feasibility or optimization problem the user of INTSOSTOOLS must:

1) Initialize a SOS program, declare the dependent and the independent variables
2) Set the integrand function and define integral-SOS constraints
3) Set objective function
4) Call solver
5) Extract solutions.

While the construction of matrices $H(\theta)$ in (19), and therefore the conversion from integral inequalities to SOS constraints can be performed manually for small dimensional instances, such a construction is difficult in general and an automated process to formulate the constraints given an integral inequality is desirable. This automation is achieved by INTSOSTOOLS functions. The solution to the SOS program is then performed by calling one of the SDP solvers supported by SOSTOOLS (see [6] for the list of solvers).

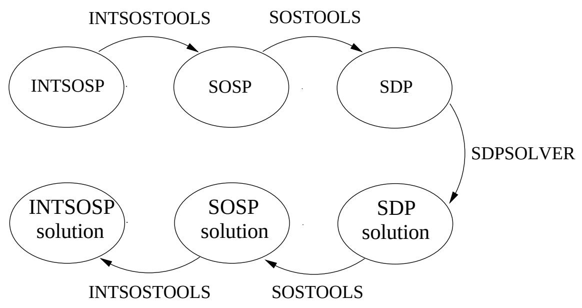

Fig. 1: Diagram depicting relations between INTSOSTOOLS, SOSTOOLS and SDP SOLVERS.

The INTSOSTOOLS functions, a set of examples and its user's guide can be downloaded from `http://sysos.eng.ox.ac.uk/control/sysos/index.php/User:Giorgio`

The plug-in INTSOSTOOLS is free, open source, and is released under a GNU licence.

V. Solving Integral Inequalities with INTSOSTOOLS

INTSOSTOOLS provides a set of functions to define a SOSP with constraints (19b), (21) and to extract solutions to integral inequalities. As in a standard SOSP in SOSTOOLS, the solution is obtained with the command `sossolve`. This section shows how INTSOSTOOLS functions are used to formulate the problems presented in Section II.

A. Wirtinger Inequality

To solve the optimization problem (8) using INTSOSTOOLS we start by defining the independent variables, the dependent variables and its values at the boundaries as symbolic (polynomial variables, `pvar`) objects

```
>>ord = 2;
>>nu = pvar('nu');
>>x = pvar('x');
>>Du = mpvar('Du',[1,ord+1]);
>>u = Du(1,1);
>>u_x = Du(1,2);
>>u_xx = Du(1,3);
```

```
>>Du1 = mpvar('Du1',[1,ord]);
>>Du0 = mpvar('Du0',[1,ord]);
```

where `ord` defines the number of spatial derivatives on the integrand, `x` is the independent variable (replacing θ in (8)), `u, u_x, u_xx` the dependent variable and its derivatives, taken from the columns of variable `Du`, and `Du1`, `Du0`, the values of the dependent variable at the boundaries of the domain $[0,1]$. At this point an SOS program is ready to be constructed with the created list of variables

```
>>dvartable = [Du(1:(ord+1)) Du1' Du0' ];
>>ivartable = [x];
>>prog = sosprogram([dvartable ivartable],...
nu);
```

and we can define the integrand of the integral constraint in (8) (`exprf` below) and call the INTSOSTOOLS function `SOSINTCONSTR` which builds the monomials related to the integration by parts, that is, the entries of matrices $\bar{H}$, $H(1)$, $H(0)$ in (19)

```
>>exprf = -nu*u_x^2+u_xx^2;
>>[exprh,Zd,IPMON] = SOSINTCONSTR(exprf,Du);
```

the output `exprh` is a structure containing fields (we keep the notation from the equation, in the code lines x replaces θ)

- `h`: symbolic variable associated to entries of matrix H in (15),
- `hdiff`: expression in (15) containing
$$\zeta^T(D^{\alpha-1}u(\theta))\frac{dH(\theta)}{d\theta}\zeta(D^{\alpha-1}u(\theta))$$
with entries $\frac{dH(\theta)}{d\theta}$ represented by symbolic variable `h`,
- `hvar`: expression in (15) containing
$$2\zeta^T(D^{\alpha-1}u(\theta))H(\theta)\frac{d\zeta(D^{\alpha}u(\theta))}{d\theta}$$
with entries $H(\theta)$ represented by symbolic variable `h`.

The output `Zd` contains the list of monomials $\xi(D^{\alpha}u(\theta))$ in (15) and the output `IPMON` is a structure containing the degree and the orders of monomials appearing in $\zeta(D^{\alpha-1}u(0))$ in (15). Then we can create a polynomial variable `hx` of degree `degh` with `degh` elements (the size of `h`) and compute its derivative

```
>>[prog,hx] = sospolymatrixvar(prog,...
monomials(x,degh), [neh,1]);
>>hxdiff = diff(hx,x);
```

and, with the above expressions, it is possible to substitute the symbolic variable `exprh.hvar` in the expressions `exprh.h` and `exprh.hdiff` as follows

```
>>exprhv = subs(exprh.h,exprh.hvar,hx);
>>exprhdiffv = subs(exprh.hdiff,...
exprh.hvar,hxdiff);
```

Moreover, we are able to set expression in the integral term of (18)

```
>>exprx = exprf+(exprhv+exprhdiffv);
```

We then formulate the test for the positivity in the interval $[0,1]$ as in (21) using the SOSTOOLS function `sosineq`. Here we write constraint equivalent expressions to (21) by first defining a quadratic expression $\xi^T(D^{\alpha}u)M(x)\xi(D^{\alpha}u)$ as the quadratic expression in the integrand of (15) and choosing the degree on x, `degreg`, of matrix $M(x)$

```
>>[prog,Mx] = sospolymatrixvar(prog,...
monomials(x,degreg),[1,size(Zd)]);
>>MxDu = Mx*Zd;
>>prog = sosineq(prog,MxDu,...
'sparsemultipartite',x,vec(Du));
```

the vector of monomials `Zd` is an output of `SOSINTCONSTR` we can then impose the first SOS constraint in (21)

```
>>Locx = exprx+(x^2-x)*(MxDu);
>>prog = sosineq(prog,Locx,...
'sparsemultipartite',x,vec(Du));
```

We then need to set the conditions on the boundaries (19b) with the help of the INTSOSTOOLS. First, define matrix Q setting the boundary values in (8) in the form (13) then set (19b) with INTSOSTOOLS function `BCCONSTRAINTS` which takes as arguments the polynomial `hx`, which has to be evaluated at the boundaries, `IPMON`, the output of function `SOSINTCONSTR` which defines the monomials in vector ζ in (19b), and the matrix Q which defines the set $\mathcal{U}_s(Q)$

```
>>Q = [1 0 0 0; 0 0 1 0;0 1 0 0; 0 0 0 1];
>>[prog,hb] = BCCONSTRAINTS(prog,x,...
[Du1;Du0],hx,IPMON.Lf,IPMON.Lof,Q,'linearBC');
```

Notice that for the integral constraint in this problem we have $F_c = 0$ in (19b). We are able to solve the SOS program by calling the SOSTOOLS function `sossolve`

```
>>prog = sossetobj(prog,nu);
>>prog = sossolve(prog);
```

The solution to the above SOSP setting different degrees `degh`, approach the exact value of $(2\pi)^{-1}$ as summarized in the table below.

TABLE I: default

`degh`	2	4	6	8	10	12
ν_{min}^{-1}	4.772	5.999	6.088	6.261	6.270	6.279

B. $\mathcal{L}_2$ stability of the heat equation with reaction term

In this example we use the MATLAB symbolic variables instead of the polynomial symbolic variables `pvar`. We follow the steps described in Section IV,

1) Initialize a SOS program, declare the dependent and the independent variables

```
>>ord = 2;
>>x = sym('x');
>>Du = sym('Du',[1,ord+1]);
```

```
>>u = Du(1,1);
>>u_xx = Du(1,3);
>>Du1 = sym('Du1',[1,ord]);
>>Du0 = sym('Du0',[1,ord]);
>>dvartable = [Du(1:(ord+1)) Du1' Du0' ];
>>ivartable = [x];
>>prog = sosprogram([dvartable ivartable]);
```

2) Set the integrand function and define integral-SOS constraints

Define a symbolic variable for polynomial $p(x)$

```
>>Zp = monomials(Du(:,1),2);
>>nZp = size(Zp,1);
>>p = sym('p',[1 nZp]);p = sym(p,'real');
>>Vp = p*Zp;
```

We can now write the derivative of the LF and calls INTSOSTOOLS function `SOSINTCONSTR`

```
>>lb = 1;
>>exprf = -(jacobian(Vp,Du(:,1))*(u_xx+lb*u));
>>[exprh,Zd,IPMON] = SOSINTCONSTR(exprf,Du)
```

Then we can create polynomial variables for symbolic objects p and h

```
>>neh = size(IPMON.Lf,1);
>>[ prog,px] = sospolymatrixvar(prog,...
monomials(x,degp), [nZp,1]);
>>[prog,hx] = sospolymatrixvar(prog,...
monomials(x,degh), [neh,1]);
>>hxdiff = diff(hx,x);
>>exprfv = subs(exprf,p,px);
>>exprhv = subs(exprh.h,exprh.hvar,hx);
>>exprhdiffv = subs(exprh.hdiff,exprh.hvar,...
hxdiff);
>>exprx = exprfv+(exprhv+exprhdiffv);
```

and formulate constraints (21)

```
>>[prog,Mx] = sospolymatrixvar(prog,...
monomials(x,degreg),[1,size(Zd)]);
>>MxDu = Mx*Zd;
>>prog = sosineq(prog,MxDu,...
'sparsemultipartite',x,vec(Du));
>>Locx = exprx+(x^2-x)*(MxDu);
>>prog = sosineq(prog,Locx,...
'sparsemultipartite',x,vec(Du));
```

and the positivity of weighting $p(x)$ in the interval $[0, 1]$

```
>>eps = 0.001;
>>epsx = eps*Du(:,1).'*Du(:,1);
>>Vpx = subs(Vp,p,px);
>>[prog,NPx] = sospolymatrixvar(prog,...
monomials(x,degreg),[1,nZp]);
>>NV = NPx*Zp;
>>prog = sosineq(prog,NV,...
'sparsemultipartite',x,vec(Du));
>>LocVx = (Vpx-epsx) +(x^2-x)*NV;
>>prog = sosineq(prog,LocVx,...
'sparsemultipartite',x,vec(Du));
```

We can now call INTSOSTOOLS function `SOSINTCONSTR` to implement (19b)

```
Q = [1 0 0 0; 0 0 1 0;0 1 0 0; 0 0 0 1];
[prog,hb] = BCCONSTRAINTS(prog,x,[DULIM;
DLLIM],hx,...
IPMON.Lf,IPMON.Lof,Q,'linearBC');
```

Notice that integral inequalities in (11) do not contain terms with matrix F_c as in (19b).

3) Set objective function

Problem (11) is a feasibility problem hence no objective function needs to be defined.

4) Call solver.

```
>>prog = sossolve(prog);
```

5) Extract solutions.

```
>>SOL_coeffsP = sosgetsol(prog,px);
>>SOL_coeffsH = sosgetsol(prog,hx);
>>HSOL = subs(exprh.h,exprh.hvar,SOL_coeffsH);
```

VI. Conclusions

This paper introduced INTSOSTOOLS, an SOSTOOLS plug-in for solving optimization problems with integral constraints. We have illustrated the use of INTSOSTOOLS via two examples: how to compute bounds for integral inequalities and how to perform Lyapunov analysis of a partial differential equation, leading to a set of integral inequalities. Preliminary analytical steps may be required to cast in the presented framework, as illustrated in Section II-A.

It is worth mentioning that INTSOSTOOLS can also be used in the context of control problems which are formulated in terms of integral constraints for bounded time intervals as, for instance, the optimal control problem with finite horizon, output feedback for switching systems and state-feedback design for periodic systems.

We are currently developing an extra set of functions to assist in the test for positivity of linear operators of the form

$$\mathcal{P}(\cdot) = p(x)(\cdot) + \int_0^1 N(x, \eta)(\cdot)\mathrm{d}\eta$$

generalizing the conditions for the operator $\mathcal{P}(\cdot) = p(x)(\cdot)$ used to generate (10). For integral inequalities involving integrals on two-dimensional domains we are generalising the results presented in this paper using Green's Theorem for simple geometries in 2D, such as rectangles and circles.

In [25] it is demonstrated that, for a transport equation with constant coefficients, the Lyapunov inequalities, similar to (11), are solved with exponential functions. That is, the solution to the inequalities, the weighting $p(\theta)$ and $H(\theta)$ in (19) involve exponentials. This highlights the fact that, polynomials in a bounded interval as considered here and implemented in INTSOSTOOLS, are *approximating* solutions to the differential inequalities (19a).

References

[1] S. Prajna, A. Papachristodoulou, and P. Parrilo, “Introducing SOSTOOLS: a general purpose sum of squares programming solver,” in *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, vol. 1, Dec 2002, pp. 741–746 vol.1.

[2] S. Prajna, A. Papachristodoulou, P. Seiler, and P. Parrilo, “SOSTOOLS: control applications and new developments,” in *Computer Aided Control Systems Design, 2004 IEEE International Symposium on*, Sept 2004, pp. 315–320.

[3] M. Choi, T. Lam, and B. Reznick, “Sums of squares of real polynomials,” in *Symposia in Pure Mathematics*, vol. 58, no. 2, 1995, pp. 103–126.

[4] P. Parrilo, “Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization,” Ph.D. dissertation, California Institute of Technology, 2000.

[5] G. Chesi, A. Tesi, A. Vicino, and R. Genesio, “On convexification of some minimum distance problems,” in *5th European Control Conference*, Karlsruhe, Germany, 1999.

[6] A. Papachristodoulou, J. Anderson, G. Valmorbida, S. Prajna, P. Seiler, and P. Parrilo, “SOSTOOLS: Sum of squares optimization toolbox for MATLAB V3.00,” 2013.

[7] [Online]. Available: http://www.mathworks.co.uk/discovery/mupad.html

[8] J. Löfberg, “Yalmip : A toolbox for modeling and optimization in MATLAB,” in *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. [Online]. Available: http://users.isy.liu.se/johanl/yalmip

[9] ——, “Pre- and post-processing sum-of-squares programs in practice,” *IEEE Transactions on Automatic Control*, vol. 54, no. 5, pp. 1007–1011, 2009.

[10] D. Henrion, J. B. Lasserre, and L. J., “Gloptipoly 3: moments, optimization and semidefinite programming.” *Optimization Methods and Software*, vol. 24, no. 4-5, pp. 761–779, 2009.

[11] Z. Jarvis-Wloszek, R. Feeley, W. Tan, K. Sun, and A. Packard, “Control applications of sum of squares programming,” in *Positive Polynomials in Control*. Springer, 2005, pp. 3–22.

[12] A. Vannelli and M. Vidyasagar, “Maximal Lyapunov functions and domains of attraction for autonomous nonlinear systems,” *Automatica*, vol. 21, no. 1, pp. 69 – 80, 1985.

[13] G. Valmorbida and J. Anderson, “Region of attraction analysis via invariant sets,” in *American Control Conference (ACC), 2014*, June 2014, pp. 3591–3596.

[14] A. Ahmadi, M. Krstic, and P. Parrilo, “A globally asymptotically stable polynomial vector field with no polynomial Lyapunov function,” in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, Dec 2011, pp. 7579–7580.

[15] I. R. Petersen and R. Tempo, “Robust control of uncertain systems: Classical results and recent developments,” *Automatica*, vol. 50, no. 5, pp. 1315 – 1335, 2014.

[16] S. Prajna, P. Parrilo, and A. Rantzer, “Nonlinear control synthesis by convex optimization,” *Automatic Control, IEEE Transactions on*, vol. 49, no. 2, pp. 310–314, Feb 2004.

[17] S. Prajna, A. Papachristodoulou, and F. Wu, “Nonlinear control synthesis by sum of squares optimization: a Lyapunov-based approach,” in *Control Conference, 2004. 5th Asian*, vol. 1, July 2004, pp. 157–165 Vol.1.

[18] S. Yang and W. F., “Control of polynomial nonlinear systems using higher degree Lyapunov functions,” *J. Dyn. Sys., Meas., Control*, vol. 136, no. 3, pp. 1–12, Feb 2014.

[19] G. Valmorbida, S. Tarbouriech, and G. Garcia, “Design of polynomial control laws for polynomial systems subject to actuator saturation,” *Automatic Control, IEEE Transactions on*, vol. 58, no. 7, pp. 1758–1770, July 2013.

[20] J. Geromel, P. Peres, and J. Bernussou, “On a convex parameter space method for linear control design of uncertain systems,” *SIAM Journal on Control and Optimization*, vol. 29, no. 2, pp. 381–402, 1991.

[21] A. Ahmadi and P. A. Parrilo, “Towards scalable algorithms with formal guarantees for Lyapunov analysis of control systems via algebraic optimization,” in *Proceedings of the 53rd IEEE Conference on Decision and Control*, Los Angeles, California,USA, 1014.

[22] M. M. Peet, A. Papachristodoulou, and S. Lall, “Positive forms and stability of linear time-delay systems,” *SIAM J. Control and Optimization*, vol. 47, no. 6, pp. 3237—3258, 2007.

[23] M. Peet, “LMI parametrization of Lyapunov functions for infinite-dimensional systems: A framework,” in *American Control Conference (ACC), 2014*, June 2014, pp. 359–366.

[24] G. H. Hardy, J. E. Littlewood, and G. Pólya, *INEQUALITIES*. Cambridge, UK: Cambridge University Press, 1952.

[25] G. Valmorbida, M. Ahmadi, and A. Papachristodoulou, “Semi-definite programming and functional inequalities for distributed parameter systems,” in *53rd Conference on Decision and Control*, Los Angeles, CA, 2014.

[26] M. Putinar, “Positive polynomials on compact semi-algebraic sets.” *Indiana Univ. Math. J.*, vol. 42, no. 3, pp. 969–984, 1993.