A Step Towards Distributed Control of Massive-scale Networks

João Domingos
oliveira.domingos@ist.utl.pt

João Xavier
jxavier@isr.ist.utl.pt

***Abstract*—Stabilizing a dynamical system implies conducting it to an intended operating point, after a momentary disturbance. The system is stabilized by the controller: a device that observes the state through sensors and acts on it through actuators. In many centralized cases, the controller has access to the complete state. This hypothesis is reasonable for low complexity systems, but in current applications - power grid, urban traffic, epidemic control, robotic vehicles - this no longer holds true. Therefore, in the distributed case the controller must act taking into account partial information. Mathematically a sparsity pattern is imposed on the controller. In the centralized case the problem can be solved using convex optimization; however, to handle the sparsity constraints the problem becomes nonconvex and $\mathcal{NP}$ hard.**

In this thesis we study three problems: (1) centralized design of stabilizing sparse controllers, based on heuristics for the non convex problem; (2) design of sparsity patterns that lead to the most stable closed loop system; (3) decentralized design of stabilizing sparse controllers. For (1) our methods can stabilize $40\% - 77\%$ of systems with sparsities of 50%, i.e., with half the degrees of freedom. Relative to (2) we were able to draw sparsities that increase the stabilization percentage by 28%, comparing against a purely randomized design. Referring to (3), we have designed a distributed algorithm with theoretical guarantees of performance, at the expense of considering purely positive systems.

***Index Terms*—Sparse Stabilization, Decentralized Stabilization, Sparsity Design, Linear Matrix Inequalities, Bilinear Matrix Inequalities**

I. Introduction

Stabilization could be seen as the most simple control problem: given a momentary disturbance we design a controller that ensures that the dynamical system returns to the desired operating point. If the controller can observe the entire state, the problem can be solved optimally using standard LMI theory. If only partial observations are available, the problem becomes $\mathcal{NP}$ hard. In [1], stabilization with decentralized static controllers that receive output measurements, was mentioned as one of the three major open problems in systems and control theory. In this document, we study three related problems: (1) the centralized computation of sparse controllers (2) optimal constrained sparsity design, in a closed loop stability sense and (2) decentralized computation of sparse controllers.

Regarding the first problem, Alavian and Rotkowitz [2] present a randomized constructive algorithm to deal with the stabilization problem, under information constraints assuming a dynamic controller type. The major insight of [2] consists in constructing a dynamic controller that can reduce the number of unfixed unstable modes, while leaving all fixed modes stable. This result is impressive; however, the authors assume that the dynamics of the controller is centralized, i.e., the controller is a dynamical system which receives some available measurements and computes the control signal in a centralized manner: all state variables of the controller, are potentially needed for the computation of the control signal. In [3], Sadabadi and Peaucelle present a complete review on the most successful heuristics for the problem of structured stabilization considering static output feedback controllers, while highlighting the special cases where the problem becomes convex and hence computationally tractable. However this survey assumes a continuous time state space formulation: the literature is vast for continuous time but poor for discrete time models. This serves as a motivation for assuming a discrete time formalism. The main ideas of [3] include: iterative LMI heuristic that try to solve the underlying BMI; Lyapunov matrix methods to decouple the Lyapunov certificate P from the closed loop system; and non-Lyapunov-based methods that approach the non-convex, non smooth optimization problem. Classical sufficient conditions for particular sparsity patterns can be found in [4], [5].

Regarding the second problem, whenever one combines the words *engineering* and *sparsity* in the same sentence, the expected outcome is to minimize some domain specific penalty function, with an added l_1 sparsity promoting regularization factor. In [6] the authors refer to l_1 regularization as the "modern least squares" when pursuing sparsity. Hence it is not surprising that this approach is also embedded in controller synthesis [7]. Some standard analysis can be found in chapter 6 of [8] and in [7] (section 6). The regularization scheme was applied in [9] (section 5.1) and [10] (section 4.1), for stabilizing continuous-time systems. This idea has one downfall: the regularizer is promoting sparsity but in an uncontrollable way, i.e., the l_1 regularization will put some entries of controller K to zero. However we do not want any zero entries. We have a constrained sparsity pattern, regarding the maximum number of non-zero entries for each row of controller K. This is the main reason why sparsity promoting techniques are not generally valid for this problem: we need sparsity ensuring methods. When reviewing the literature, one actually finds that most of sparsity design methods use some type of sparsity promoting technique [11], [12].

For the third problem, we make use of an interesting result which states that decentralized controller synthesis actually becomes a convex problem when one restricts the closed loop system to be non-negative [13] (Proposition 2). Using this observation we propose a decentralized control strategy by combining the work of Aybat and Hamedani [14] with some standard consensus results from the field of network science [15]. We found no other truly decentralized control strategy for producing sparse stabilizing controllers, for discrete time systems.

The remainder of the document is organized as follows: in sections II, III and IV we define and study the three corresponding control challenges. For each section, we formulate the control problem and then proceed to describe our methods. In section V we present a numerical comparison of the developed heuristics and further illustrate the convergence properties of a provable algorithm from section IV-B: the Decentralized Primal Dual Synthesizer with Consensus - $\mathcal{DPDSC}$.

II. Centralized Computation of Decentralized Controllers

A. Problem Formulation

Consider a linear, time invariant system (LTI) described by the following matrix difference equations:

$$x(k+1) = Ax(k) + Bu(k) \tag{1}$$

where $A \in \mathbf{R}^{n\times n}$ and $B \in \mathbf{R}^{n\times m}$. This model comes from state space theory where $x(k) \in \mathbf{R}^n$ represents the state variables at instant k and $u(k) \in \mathbf{R}^m$ is the m dimensional control signal. The control signal $u(k)$ will be computed by a static linear controller of the following type

$$u(k) = Kx(k). \tag{2}$$

The controller receives input measurements $x(k)$ and computes the control signal $u(k)$. The controller is defined by matrix $K \in \mathbf{R}^{m\times n}$. From now on assume that (A, B) forms a stabilizable pair, i.e., for any initial condition $x(0)$ there exists a control sequence $\{u(k)\}_{k\geq 0}$ such that $x(k) \to 0$ where $x(k)$ is given by equation (2). Combining the controller and the plant in a single system (equations (1) and (2)) we conclude that

$$x(k+1) = (A + BK)x(k). \tag{3}$$

A decentralized controller K is a gain matrix that must respect a pre-defined assignment of which state variables are available to compute each control component. To formalize this idea define the indicator function $I(i,j)$ as:

$$I(i,j)\colon \{1,\dots,n\} \times \{1,\dots,m\} \to \{0,1\} \tag{4}$$
$$(i,j) \mapsto \begin{cases} 1 & \text{if } i \overset{c}{\sim} j \\ 0 & \text{otherwise} \end{cases}$$

where $i \overset{c}{\sim} j$ means that the control component $u_j(k) \in \mathbf{R}$ can be computed using $x_i(k) \in \mathbf{R}$ for any time instant $k \geq 1$. The set of distributed controllers $\mathcal{K}_I$ is given by:

$$\mathcal{K}_I = \{K \in \mathbf{R}^{m\times n} : K_{i,j} = 0 \text{ if } I(i,j) = 0\}, \tag{5}$$

since if $K \in \mathcal{K}_I$ then the computation of $u_j(k)$ involves only the j-th row of controller K and state variables $x_i(k)$ for $I(i,j) = 1$. A centralized controller is given by $I(i,j) = 1$ for any valid pair (i,j), i.e., the function I dictates how sparse the controller must be.

Given some sparsity pattern $\mathcal{K}_I$ the decentralized optimal stabilization problem consists in designing a controller which insures the decay of state variables $x(k)$ to zero, in a minimal amount of time regardless of initial condition $x(0)$. Formally, the problem is formulated as:

$$\underset{K\in\mathcal{K}_I}{\text{minimize}} \;\; \rho(A + BK) \tag{6}$$

where the quantity $1/\rho(A + BK)$ corresponds to the closed loop decay rate and

$$\rho(A+BK) := \max\Big\{|\lambda_i(A+BK)| : i = 1,\dots,n\Big\}, \tag{7}$$

with $\lambda_i(A + BK)$ denoting the i-th eigenvalue of $A + BK$, assuming some ordering. In particular, controller K should guarantee that $\rho(A + BK) < 1$, such that the closed loop system is stable:

Theorem 1. *Let $x(k) \in \mathbf{R}^n$ such that $x(k+1) = (A + BK)x(k)$ for some stabilizable pair $(A,B) \in \mathbf{R}^{n\times n}\times\mathbf{R}^{n\times m}$, $K \in \mathbf{R}^{m\times n}$ and for $k \geq 1$. Then $x(k) \to 0$ iff $\rho(A+BK) < 1$ where $\rho(.)$ denotes the spectral radius. In fact $x(k) = (A + BK)^k x(0)$ where $x(0)$ denotes the initial state condition, so the decay is exponential.*

In the centralized case, problem (6) can be solved optimally by combining theorem 2 with a bisection strategy. Theorem 2 provides a computable answer to the following question: given a fixed bound γ, is there a decentralized controller $K \in \mathcal{K}_I$ such that $\rho(A + BK) < \gamma$?

Theorem 2. *For any stabilizable pair $(A,B) \in \mathbf{R}^{n\times n} \times \mathbf{R}^{n\times m}$ and $\gamma > 0$ there exists a controller $K \in \mathbf{R}^{m\times n}$ such that $\rho(A + BK) < \gamma$ iff there exists a $Q \succ 0$ and $G \in \mathcal{R}^{m\times n}$ such that $\begin{bmatrix} \gamma^2 Q & (AQ+BG)^T \\ AQ+BG & Q \end{bmatrix} \succ 0$, where $G = KQ$.*

The bisection procedure is simply a strategy that constructs a sequence of γ values, for which several feasibility questions are asked. Given some bound $u_{init} \geq ||A||_2$ assume the following initialization:

$$\Big[l(0), u(0)\Big] := \Big[0, u_{init}\Big], \tag{8}$$

where $\Big\{l(k), u(k)\Big\}_k$ defines a sequence of intervals, $\Big[l(k), u(k)\Big]$ that contain the solution of problem (6). Denote

$\gamma(k)$ as the middle point between $l(k)$ and $u(k)$, i.e.,

$$\gamma(k) = \frac{u(k) + l(k)}{2}. \tag{9}$$

The bisection procedure updates $\Big\{l(k), u(k)\Big\}_k$ as follows

$$\begin{cases} u(k+1) = \gamma(k) & \text{if } \exists\ K: \ \rho(A+BK) < \gamma(k) \\ l(k+1) = \gamma(k) & \text{otherwise.} \end{cases} \tag{10}$$

Combining update (10) with theorem 2 we can solve problem (6) within any precision $\delta > 0$, i.e., we can find a controller K such that $|\rho(A+BK) - \rho(A+BK^*)| \leq \delta$, where K^* denotes the optimal solution of (6).

In the decentralized case, the problem becomes numerically challenging since theorem 2 does not ensure that $K \in \mathcal{K}_I$, i.e., there are no sparsity guarantees. In general, decentralized controller design is $\mathcal{NP}$ hard [16], [17] since problem (6) is equivalent to

$$\begin{aligned} \min_{P,K,\phi} \quad & \sqrt{\phi} \\ & \phi P \succ (A+BK)^T P (A+BK) \\ & P \succ 0, \ \ \phi > 0, \ \ K \in \mathcal{K}_I, \end{aligned} \tag{11}$$

which involves solving a Bilinear Matrix Inequality (BMI): a matrix inequality which is linear in each of the individuals variables (ϕ, P, K). This reformulation follows trivially from theorem 3 [18] (Theorem 3.5.4).

Theorem 3. *For any $\Sigma \in \mathbf{R}^{n\times n}$ and $\gamma > 0$ we have that $\rho(\Sigma) < \gamma$ iff there exists $P \succ 0$ such that $\Sigma^T P \Sigma \prec \gamma^2 P$, where $P \succ 0$ means that P is a symmetric positive definite matrix.*

B. Algorithms

In this section we present some heuristics for decentralized control synthesis, i.e., problem (6). There are other approaches, but the algorithms presented here allow to introduce several ideas for which good numerical results were encountered.

1) Incremental Inverse: The incremental inverse algorithm, is based on theorem 4 which gives an equivalent condition for the generalized inequality of theorem 3 in terms of P and a new variable Q:

Theorem 4. *For any $\Sigma \in \mathbf{R}^{n\times n}$ and $\gamma > 0$ we have that $\rho(\Sigma) < \gamma$ iff there exists a $Q, P \succ 0$ such that $\begin{bmatrix} \gamma^2 P & \Sigma^T \\ \Sigma & Q \end{bmatrix} \succ 0$, $\begin{bmatrix} P & I_n \\ I_n & Q \end{bmatrix} \succeq 0$ and $\mathrm{trace}(PQ) = n$.*

The feasible set, given in theorem 4, is non convex due to the constraint involving the bilinear term PQ. The value of this result is that it allows to test whether the pair (P, Q, Σ) belongs to this non-convex set, by evaluating the solution of an optimization problem:

Theorem 5. *For any $\Sigma \in \mathbf{R}^{n\times n}$ and $\gamma > 0$ we have that there exists a $Q, P \succ 0$ such that $\begin{bmatrix} \gamma^2 P & \Sigma^T \\ \Sigma & Q \end{bmatrix} \succ 0$, $\begin{bmatrix} P & I_n \\ I_n & Q \end{bmatrix} \succeq 0$ and $\mathrm{trace}(PQ) = n$ iff $f(\Sigma, \gamma) = n$ where the function f is defined as:*

$$\begin{aligned} f(\Sigma, \gamma) = \min_{P,Q} \quad & \mathrm{trace}(PQ) \\ & \begin{bmatrix} \gamma^2 P & \Sigma^T \\ \Sigma & Q \end{bmatrix} \succ 0 \\ & \begin{bmatrix} P & I_n \\ I_n & Q \end{bmatrix} \succeq 0, \ \ P, Q \succ 0. \end{aligned}$$

The incremental inverse heuristic is a path following method which does incremental updates in the variables P and Q, for a fixed bound $\gamma > 0$. Consider $P = \hat{P} + \Delta P \succ 0$ and $Q = \hat{Q} + \Delta Q \succ 0$ and note that, by theorem 5, the problem of finding a controller $K \in \mathcal{K}_I$ such that $\rho(A+BK) < \gamma$ can be transformed into optimizing the non convex function

$$\mathrm{trace}(PQ) = \mathrm{trace}(\hat{P}\hat{Q} + \hat{P}\Delta Q + \Delta P \hat{Q} + \Delta P \Delta Q), \tag{12}$$

in a convex set. Since the increments will correspond to the optimization variables, the objective function is non convex due to the cross terms $\Delta P \Delta Q$ but, intuitively, since ΔP and ΔQ represent incremental variables their product should have a "low" trace hence, by assuming $\mathrm{trace}(\Delta P \Delta Q) \approx 0$ we drop this term and lose necessity.

Algorithm 1 Incremental inverse heuristic with $\gamma > 0$

(i) Given some $\gamma > 0$ initialize $P_0, Q_0 \succ 0$ and further define N_{max} as the maximum number of iterations and ϵ as a stopping tolerance.

(ii) Find $(K_{k+1}, P_{k+1}, Q_{k+1})$ by solving the following optimization problem:

$$\begin{aligned} \underset{P,Q,\Delta P,\Delta Q,K}{\arg\min} \quad & \mathrm{trace}(P_k \Delta Q + Q_k \Delta P + P_k Q_k) \\ \text{subject to} \quad & \begin{bmatrix} \gamma^2 P & (A+BK)^T \\ A+BK & Q \end{bmatrix} \succ 0 \\ & \begin{bmatrix} P & I_n \\ I_n & Q \end{bmatrix} \succeq 0, \ \ P, Q \succ 0, \ \ K \in \mathcal{K}_I \\ & P = P_k + \Delta P, \ \ Q = Q_k + \Delta Q. \end{aligned} \tag{13}$$

(iii) Repeat (ii) until $\Big\{\rho(A+BK_k) < \gamma\Big\} \bigvee \Big\{k = N_{max}\Big\} \bigvee \Big\{|\rho(A+BK_{k-1}) - \rho(A+BK_k)| < \ \epsilon\Big\}$

Algorithm 1 tries to find a controller $K \in \mathcal{K}_I$ such that $\rho(A+BK) < \gamma$ for a fixed γ. However we can also employ a bisection framework where the feasibility strategy of equation (10) is combined with algorithm 1.

2) Decentralized Convex-Concave Approach: This section introduces the convex-concave heuristic applied to problem (6): the decentralized convex-concave approach. The standard convex concave procedure (CCP) is a well known heuristic that converges to the set of stationary points of difference of convex (DC) optimization problems [19].

Before introducing our approach we have to generalize the concept of scalar convexity to matrix convexity. A function $f_i : \mathbf{R}^n \mapsto \mathbf{S}^m$ is matrix convex iff:

$$\forall \ \theta \in [0,1] : f_i(\theta x + (1-\theta)y) \preceq \theta f_i(x) + (1-\theta) f_i(y) \tag{14}$$

for any $(x, y) \in \mathbf{R}^n \times \mathbf{R}^n$. The following theorem is the main reason why the decentralized control design task can be interpreted as a matrix value DC optimization problem in $\mathcal{S}_n^{++}$, i.e., the cone of positive semidefinite matrices:

Theorem 6. *The function $f(\Sigma, P) = \Sigma^T P^{-1} \Sigma$ is matrix convex where (Σ, P) is defined in $\mathbf{R}^{n\times n} \times \mathbf{S}_{++}^n$.*

Given some bound $\gamma > 0$ the controller design problem is the following:

$$\begin{aligned} \min_{K,P} \quad & 0 \\ & (A+BK)^T P^{-1}(A+BK) - \gamma^2 P^{-1} \prec 0 \\ & P \succ 0, \ \ K \in \mathcal{K}_I. \end{aligned} \tag{15}$$

Problem (15) is a matrix DC feasibility problem, since function P^{-1} is also matrix convex (set $\Sigma = I_n$ in theorem 6). Having recognized this structure, we can some insight from the classical CCP approach, namely the linearization of the matrix convex function P^{-1}:

Theorem 7. *Let $F(P) = P^{-1}$ be a matrix function defined in $\mathbf{S}_{++}^n$. If $\hat{P} \in \mathbf{S}_{++}^n$ then*

$$P^{-1} \succeq \hat{P}^{-1} - \hat{P}^{-1}(P - \hat{P})\hat{P}^{-1}$$

so $\hat{P}^{-1} - \hat{P}^{-1}(P - \hat{P})\hat{P}^{-1}$ denotes the first order approximation of function F at point $\hat{P}$.

It follows trivially that, by utilizing theorem 7 one can get sufficient conditions for decentralized stability, for a fixed bound $\gamma < 1$. To transform this sufficient condition in a algorithmic procedure, we make use of one fairly simple insight: start with a "big" bound r_k and, iteratively, reduce it by a factor $\beta < 1$ if $\rho(A + BK_k) < r_k$ (reduction strategy). This strategy could be thought as a path following method (homotopy) [10] where the design objective is iteratively improved, by at least β, leading to a "path" of LMIs that try to find a local solution of problem (6). This idea is also the basis for the incremental PK approaches seen in the numerical section V-A.

There is no need to provide an initial $\gamma > 0$, since the heuristic will continue running even if $\rho(A+BK_k) < \gamma$. The reduction strategy can be thought of as a "global" method, in the sense that it tries to find the optimal controller and not just a controller K_k that guarantees that $\rho(A + BK_k) < \gamma$.

Algorithm 2 Decentralized convex-concave heuristic

(i) Find some $P_0 \succ 0$ and $K_0 \in \mathcal{K}_I$ such that $(A + BK_0)^T P_0^{-1}(A + BK_0) \prec r_0^2 P_0^{-1}$ for some $r_0 > 0$. Further define N_{max} as the maximum number of iterations, ϵ as a stopping tolerance and $\beta > 0$ as the **shrinking factor**, i.e., $\beta < 1$. Set $r_0 = \beta r_0$.

(ii) Find $P_{k+1} = P \succ 0$ and $K_{k+1} = K \in \mathcal{K}_I$ such that

$$(A+BK)^T P^{-1}(A+BK) \prec r_k^2 \{P_k^{-1} - P_k^{-1}(P-P_k)P_k^{-1}\}. \tag{16}$$

(iii) If $\rho(A + BK_k) < r_k$ then $r_{k+1} = \beta r_k$; otherwise go to (v).

(iv) Repeat until $\Big\{ |\rho(A+BK_{k-1}) - \rho(A+BK_k)| < \ \epsilon \Big\}$
$\bigvee \Big\{ k = N_{max} \Big\}$

(v) Return $K^* \in \arg\min_{K\in\mathcal{K}_I} \big\{\rho(A+BK_0), \ldots, \rho(A+BK_{end})\big\}$, where K_{end} denotes the controller associated with the last iteration of the algorithm. At most, the heuristic runs for N_{max} iterations.

III. Optimal Sparsity Patterns

A. Problem Formulation

Define a budget $\phi_j \in \{0, \ldots, n\}$ as the maximum number of sensor readings, that can be used to compute control component u_j, where $j = 1, \ldots, m$. Clearly the centralized setting can be recovered with $\phi_j = n$, for any valid j. The budget vector $\Phi = (\phi_i)_{j=1}^m$ can be used to model the computational resources available to compute each control component: if $\phi_i = 1$ then the computational entity that computes u_i, can only process one measurement.

If controller K respects budget Φ, then the i-th row of K has a maximum of ϕ_i non-zero entries. Define $\mathcal{K}_\Phi \subseteq \mathbf{R}^{m\times n}$ as the set of controllers that respect a given budget Φ. The number of structural constraints associated with the i-th row of $K \in \mathcal{K}_\Phi$ is given by

$$N_i = \sum_{l=0}^{\phi_i} \binom{n}{l} \tag{17}$$

where scalar N_i represents all possible combinations of choosing a maximum of ϕ_i non-zero entries among n choices. Applying this reasoning for each row, the controller K can exhibit a total of $N = \prod_{i=1}^m N_i$ sparsity patterns. It follows trivially that

$$\mathcal{K}_\Phi = \bigcup_{i=1}^N \mathcal{K}_\Phi^i, \tag{18}$$

where $\{\mathcal{K}_\Phi^i\}_{i=1}^N$ denotes a partition of the feasible space. Under this framework we would like to find the controller that yields the stablest close loop system, i.e.,

$$\underset{K \in \mathcal{K}_\Phi}{\text{minimize}} \ \rho(A+BK). \tag{19}$$

By (18), the set $\mathcal{K}_\Phi$ is given by the union of N disjoint sets. This implies that problem (19) is actually combinatorial, i.e., in order to solve (19) we must solve N optimization problems imposing that $K_i \in \mathcal{K}_\Phi^i$ for $i = 1, \ldots, N$ and then pick the best found controller, in a spectral radius sense. This adds a new of degree difficult: $\mathcal{NP}$ hardness plus a combinatorial nature.

B. Algorithms

Given some budget vector $\Phi \in \mathbf{R}^n$ assume the existence of an algorithm, $\mathcal{A}$, that can be used to "solve" problem (19) under some fixed sparsity pattern $\mathcal{K}_\Phi^i$ (consult section II-B). This section will develop oracles for optimal sparsity patterns: strategies that will try to find the optimal pattern for problem (19), by avoiding the exhaustive search over the partition $\{K_\Phi^i\}_1^N$. Our motivation will to use perturbation theory arguments for certain optimization problems: convex problems for which strong duality holds and the primal solution is finite [20], [8]. For these problems the optimal dual variables allow to quantity how much a given linear constraint locally influences the solution of the primal problem.

1) Static Incremental Inverse Oracle - $\mathcal{O}_{II}^s$: Given some budget vector $\Phi \in \{0, \ldots, n\}^m$ a feasible sparsity structure can always be found by imposing that $K = 0$, i.e., considering a full sparse controller. Consider the following experiment: assume we deploy the incremental inverse heuristic, algorithm 1, for a fixed bound $\gamma = 1$ and imposing a full sparse controller. The heuristic will finish at the second iteration, since one of the stopping criteria is immediately met:

$$|\rho(A+BK_1) - \rho(A+BK_2)| = 0 < \epsilon, \ \forall \ \epsilon > 0. \tag{20}$$

This might seem redundant since the underlying process served as a pass-all filter: given a full sparse controller it returned the same full sparse controller. However this is not entirely true, since the underlying method also returned valuable dual information, namely the dual variables $\lambda_{i,j}^*$ associated with the constraints:

$$\lambda_{i,j}^* : \ \ K^*(i,j) = 0, \ \ (i,j) \in \{1, \ldots, m\} \times \{1, \ldots, n\}. \tag{21}$$

The dual variable with the largest absolute value indicates which constraints has a greater influence on the optimal value of problem (13), assuming some technical properties are verified. The incremental inverse static oracle, $\mathcal{O}_{II}^s$, uses one very simple idea: construct the set $\mathcal{K}_{\mathcal{O}_{II}^s} \subseteq \mathcal{K}_\Phi$ using the most influential dual variables corresponding to the last iteration of the incremental inverse method. One can easily verify that the perturbation theory interpretation of problem (13) with $(P_k, Q_k) = (P_1, Q_1)$ is valid since (a) the problem is convex, (b) Slater's conditions holds and (c) the primal solution is finite. Algorithm 3 identifies the constraints which have the largest discriminative power to influence the locally optimal value of (13) with $(P_k, Q_k) = (P_1, Q_1)$. Our reasoning underlies one fundamental assumption: influential constraints for problem (13) are also influential for problem (19). If we have to choose between two constraints, we eliminate the one with the largest absolute dual variable, since variations on the optimal value of (13) may allow to infer on problem (19). In particular if (13) achieves the finite optimal value then the solution (19) is strictly upper bounded by $\gamma = 1$.

Algorithm 3 Incremental Inverse static oracle $\mathcal{O}_{II}^s$

(i) For any budget Φ, run the incremental inverse heuristic, algorithm 3, with $\gamma = 1$ and imposing a full sparse controller, i.e., $K = 0$.

(ii) Problem (13) will only be solved twice; hence, collect the corresponding matrix of dual variables Λ_{II}^s:

$$\Lambda_{II}^s \in \arg\max_{\Lambda} \quad L(\Lambda) \tag{22}$$

where L denotes the Lagrange Dual function of problem (13) with $(P_k, Q_k) = (P_1, Q_1)$. In general, there is no closed form expression for function L.

(iii) The dual information Λ_{II}^s allows to construct the corresponding feasible set $\mathcal{K}_{\mathcal{O}_{II}^s}$: the zero entries that defined the set $\mathcal{K}_{\mathcal{O}_{II}^s}$ correspond to the elements of Λ_{II}^s, with the lowest absolute value such that the entire budget Φ is spent.

2) Iterative Incremental Inverse Oracle - $\mathcal{O}_{II}^i$: Algorithm 3 has a static nature, in the sense that a single call to the underlying heuristic served as a surrogate for constructing the set $\mathcal{K}_{\mathcal{O}_{II}^s} \in \mathcal{K}_\Phi$. Another option is to use an iterative oracle that builds the same set using dual informations from several calls. We propose an iterative approach that instead of doing all assignments of $\mathcal{K}_{\mathcal{O}_{II}^s}$ in one, each assignment is associated with one run of the incremental inverse heuristic. Intuitively, this makes sense since doing all assignments at once can be to greedy.

The basic reasoning is to use the idea of algorithm 3 but in step (iii) we only choose the constraint associated with the largest absolute value of the underlying dual variable. We then impose this constraint and repeat the process, iteratively, until the entire budget vector is spent. Each call to algorithm 1 runs for N_{oracle} iterations.

IV. Decentralized Computation of Decentralized Controllers

A. Problem Formulation

Given a stabilizable system (A, B), let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ denote an undirected graph, where each node represents an agent that can measure a state variable. The set of nodes is partitioned into control, $\mathcal{N}_c$, and non-control nodes, $N \setminus \mathcal{N}_c$.

Control node $i \in \mathcal{N}_c$ has to compute some component of the control signal from local information, i.e., information from neighbouring nodes in graph $\mathcal{G}$. The concept of information will be made clear further on. $I_c(i)$ denotes the control component computed by node $i \in \mathcal{N}_c$. An edge $(i,j) \in \mathcal{E}$ indicates that nodes i and j can exchange bidirectional information between themselves, and $\mathcal{N}_i$ is the neighbouring set of node $i \in \mathcal{N}$. We will also assume that $\mathcal{G}$ is a connected graph, meaning that there exists a path between any two nodes, i.e., there is no isolated state variable.

As before, our goal is to compute a decentralized controller K that stabilizes system (A,B), where the sparsity of the controller is dictated by the topology of the network $\mathcal{G}$:

$$\forall\ i \in \mathcal{N}_c :\ \ K_{I_c(i),j} = 0,\ \text{ for } j \notin \mathcal{N}_i. \tag{23}$$

This section defines a new problem: the decentralized design of controller K.

Definition 1. *Given a stabilizable system (A,B) and graph $\mathcal{G}$ assume that each node $i \in \mathcal{N}$ is initialized with a non-trivial private variable $V_i(0) \in \mathbf{R}^{l_i \times p_i}$. If $(i,j) \in \mathcal{E}$ then node i can receive $V_j(k)$ and send $V_i(k)$ from/to node j, for any $k \geq 0$. $\mathcal{D}$ is a decentralized design strategy if and only if:*

(i) For every node $i \in \mathcal{N}$, the variable $V_i(k+1)$ is updated from neighbouring past information:

$$\Big\{V_i(m), V_j(m)\Big\}_{0 \leq m \leq k},\ \textit{for}\ j \in \mathcal{N}_i. \tag{24}$$

(ii) Asymptotically, control node $i \in \mathcal{N}_c$ computes the $I_c(i)$-th row of controller K, as a function of private information V_i:

$$V_i := \lim_{k\to\infty} V_i(k). \tag{25}$$

This automatically implies that the update strategy of (i), most converge.

(iii) Asymptotically, the overall closed loop system is γ stable, i.e., $\rho(A+BK) < \gamma < 1$, where K is given by (ii).

The non-trivial initialization, in definition 1, prevents the usage of centralized design strategies. In order to stabilize system (A,B) one encounters two types of delay:

(i) The delay associated with the decentralized computation of K.

(ii) The delay associated with the natural decay of state variables, once controller K is applied.

B. Closed Loop Positive Systems

A closed loop system is called positive if $A+BK \in \mathbf{R}_+^{n\times n}$, i.e., each entry of matrix $A+BK$ is non-negative. The general interpretation of this condition is that the dynamics of the closed loop system are invariant to the nonnegative orthant, i.e.,

$$x(0) \in \mathbf{R}_+^n \ \Rightarrow\ x(k) = (A+BK)^k x(0) \in \mathbf{R}_+^n,\ \forall\ k \geq 0. \tag{26}$$

Note that the state space matrices, (A,B), are not required to be nonnegative. We have chosen to restrict ourselves to this class of system since the problem of synthesizing sparse stabilizing controllers, actually becomes convex when one restricts the closed loop system to be positive.

The next theorem summarizes the centralized results of designing sparse stabilizing controllers, for closed loop positive systems, with a cone program. For completeness, we also consider additional linear constraints on controller K:

$$L_1 K \geq L_2. \tag{27}$$

Theorem 8. *For any stabilizable pair $(A,B) \in \mathbf{R}^{n\times n} \times \mathbf{R}^{n\times m}$ and $\gamma > 0$ there exists a controller $K \in \mathcal{K}_I$ such that $\rho(A+BK) < \gamma$, $A+BK \geq 0$, $L_1K \geq L_2$ iff there exists exists a diagonal matrix $Q \succ 0$ and $G \in \mathcal{K}_I$ such that:*

$$\begin{bmatrix} \gamma^2 Q & (AQ+BG)^T \\ AQ+BG & Q \end{bmatrix} \succ 0 \tag{28}$$
$$AQ+BG \geq 0,\ \ L_1G \geq L_2Q$$

where controller $K \in \mathcal{K}_I$ is given by $K = GQ^{-1}$.

The novelty of this section consists in deriving a decentralized synthesis strategy, using the SDP of theorem 8 for a fixed γ. We denote our approach as $\mathcal{DPDSC}$ - Decentralized Primal Dual Synthesizer with Consensus.

1) Decentralized Primal Dual Synthesizer with Consensus: In the context of decentralized design strategies, the sparsity pattern of theorem 8 is interpreted as follows:

$$\forall i \in \mathcal{N}_c,\ \ k_i \in \mathcal{S}_i := \{x \in \mathbf{R}^n :\ \ x_j = 0,\ \text{ for } j \notin \mathcal{N}_i\} \tag{29}$$

where k_i denotes the i-th column of controller K transposed, i.e., $K^T = \Big[k_1 \dots k_m\Big]$. Assume the following decomposition of matrices (B, L_1),

$$B = \Big[b_1 \dots b_m\Big],\ \ L_1 = \Big[l_1^1 \dots l_m^1\Big]. \tag{30}$$

Given any $\epsilon_Q > 0$ and $\epsilon_\Sigma > 0$ the next feasibility problem is equivalent to the LMI condition given in theorem 8:

$$\begin{bmatrix} \gamma^2 Q_i & *^T \\ AQ_i + b_i g_i^T & Q_i \end{bmatrix} \succeq \Sigma_i,\ \ Q_i \succeq \epsilon_Q \tag{31}$$
$$AQ_i + b_i g_i^T \geq \Lambda_i,\ \ l_i^1 g_i^T - L_2 Q_i \geq \Gamma_i$$
$$\sum_{i=1}^m \Lambda_i \geq 0,\ \ \sum_{i=1}^m \Gamma_i \geq 0,\ \ \sum_{i=1}^m \Sigma_i \succeq \epsilon_\Sigma I_{2n}$$

where the optimization variables are $\Big\{Q_i, g_i, \Lambda_i, \Sigma_i, \Gamma_i\Big\}_{i=1}^m$, Q_i is diagonal and $g_i \in \mathcal{S}_i$. This reformulation simply uses a rank one sum decomposition of the products BK and L_1K, a homogeneity argument and the fact that positive definiteness is preserved trough conic combinations, i.e., sums.

Note that problem (31) has a very particular structure: we have m decoupled feasibility problems and three coupling inequalities that link the variables $\Big\{\Lambda_i, \Sigma_i, \Gamma_i\Big\}_{i=1}^m$. This type of structure can be used to derive a decentralized synthesis algorithm, i.e., a strategy that respects definition 1. The theoretical results mostly come from a recent paper of Aybat and Hamedani [14], where a decentralized algorithm is derived for certain decoupled optimization problems with coupling conic constraints. Define the following operators that given an input vector x or matrices $\{X_i\}$, construct a diagonal or block diagonal matrix:

$$\forall\ x \in \mathbf{R}^n:\ \ \mathrm{diag}(x) = \begin{bmatrix} x_1 & & \\ & \ddots & \\ & & x_n \end{bmatrix} \in \mathbf{R}^{n\times n} \tag{32}$$

$$\forall\ \Big\{X_i\Big\}_{i=1}^n:\ \ \mathrm{Diag}(X_1,\ldots,X_n) = \begin{bmatrix} X_1 & & \\ & \ddots & \\ & & X_n \end{bmatrix}. \tag{33}$$

We denote our approach as $\mathcal{DPDSC}$ - Decentralized Primal Dual Synthesizer with Consensus. In algorithm 4, $\mathcal{DPDS}$ will solve (31) in a decentralized way, i.e., asymptotically each control node has access to $\Big\{g_i, Q_i, \Lambda_i, \Sigma_i, \Gamma_i\Big\}$ such that (31) holds. To compute the controller row k_i:

$$k_i = g_i\Big\{\sum_{i=1}^m Q_i\Big\}^{-1} \tag{34}$$

each control node needs to know the inverse Lyapunov certificate after $N_{\mathcal{DPDS}}$ iterations:

$$\sum_{i\in\mathcal{N}_c} Q_i(N_{\mathcal{DPDS}}). \tag{35}$$

To compute this sum in a decentralized way, we consider n parallel consensus rounds [15]. Define $\Big\{Q_{j,j}(i,k)\Big\}_{j=1}^n$, where $Q_{j,j}(i,k)$ denotes the (j,j) entry of matrix (35) at consensus iteration k for node i. This consensus variables are initialized as follows:

$$\begin{aligned} i \in \mathcal{N}\setminus\mathcal{N}_c:&\ \ Q_{j,j}(i,0) = 0,\ \ \forall\ j\in\mathcal{N} \\ i \in \mathcal{N}_c:&\ \ Q_{j,j}(i,0) = n\Big\{Q_i(N_{\mathcal{DPDS}})\Big\}_{j,j},\ \ \forall\ j\in\mathcal{N}. \end{aligned} \tag{36}$$

Algorithm 4 Decentralized Primal Dual Synthesizer with Consensus - $\mathcal{DPDSC}$

(i) Given some stabilizable pair (A,B), connected graph $\mathcal{G}$ and bound $\gamma > 0$ run algorithm 5 with constants $(\epsilon_Q, \epsilon_\Sigma)$ and for $N_{\mathcal{DPDS}}$ iterations.

(ii) Run n parallel consensus rounds, each for N_c iterations. Using initializations (36) perform:

$$Q_{j,j}(k+1) = WQ_{j,j}(k),\quad Q_{j,j}(k) = \begin{bmatrix} Q_{j,j}(1,k) \\ \vdots \\ Q_{j,j}(n,k) \end{bmatrix} \tag{37}$$

for any $j\in\mathcal{N}$, W a consensus matrix [15] and $k\le N_c$.

Algorithm 5 Decentralized Primal Dual Synthesizer - $\mathcal{DPDS}$

(i) Given a stabilizable pair (A,B), connected graph $\mathcal{G}$ and bound $\gamma > 0$ define an arbitrary scalar $\beta > 0$ and $N_{\mathcal{DPDS}}$ as the maximum number of iterations. Each node $i \in \mathcal{N}$ is initialized with an arbitrary $c_i > 0$, $S_i(0) = Y_i(0) = 0$ and weights $\tau_i = 1/c_i$, $\kappa_i = \frac{c_i}{2c_i\beta|\mathcal{N}_i|+1}$. Control nodes are also initialized with $\Big\{g_i(0), Q_i(0), \Lambda_i(0), \Sigma_i(0), \Gamma_i(0)\Big\}$ such that (31) holds **without** the last three coupling constraints.

(ii) Each control node $i\in\mathcal{N}_c$ updates the primal variables, $\Big\{Q_i(k+1), \Lambda_i(k+1), \Sigma_i(k+1), \Gamma_i(k+1), g_i(k+1)\Big\}$:

$$\begin{aligned} \underset{Q_i,\Lambda_i,\Sigma_i,g_i}{\text{minimize}}\quad & f(Q_i,\Lambda_i,\Sigma_i,\Gamma_i,g_i) \\ \text{subject to}\quad & \begin{bmatrix} \gamma^2 Q_i & *^T \\ AQ_i + b_i g_i^T & Q_i \end{bmatrix} \succeq \Sigma_i \\ & g_i \in \mathcal{S}_i,\ \ Q_i \succeq \epsilon_Q I_n\ \ \text{diagonal} \\ & AQ_i + b_i g_i^T \ge \Lambda_i,\ \ l_i^1 g_i^T - L_2 Q_i \ge \Gamma_i \end{aligned} \tag{38}$$

$$\begin{aligned} f(.) = \frac{1}{2\tau_i}\Big\{ & ||Q_i - Q_i(k)||_F^2 + ||\Lambda_i - \Lambda_i(k)||_F^2 + \\ & ||\Sigma_i - \Sigma_i(k)||_F^2 + ||g_i - g_i(k)||_F^2 + \\ & ||\Gamma_i - \Gamma_i(k)||_F^2\Big\} + \mathrm{trace}\Big(Y_i(k)^T\Omega\Big) \end{aligned}$$

$$\Omega = \mathrm{Diag}\Big\{\mathrm{diag}\big(\mathrm{vec}(\Lambda_i)\big), \mathrm{diag}\big(\mathrm{vec}(\Gamma_i)\big), \Sigma_i - \frac{\epsilon_\Sigma}{n}I_{2n}\Big\}.$$

(iii) Add/Update the following mismatch variable:

$$\forall\ i\in\mathcal{N}:\ \ E_i(k+1) = \sum_{j\in\mathcal{N}_i}\Big\{S_i(k) - S_j(k)\Big\}. \tag{39}$$

(iv) Each node $i\in\mathcal{N}$ updates the dual variable $Y_i(k)$:

$$\begin{aligned} \underset{Y_i}{\text{minimize}}\quad & g(Y_i) \\ \text{subject to}\quad & Y_i \preceq 0 \end{aligned} \tag{40}$$

$$\begin{aligned} g(.) =& \beta\mathrm{trace}\Big(Y_i(k)^T E_i(k+1)\Big) + \frac{1}{2\kappa_i}||Y_i - Y_i(k)||_F^2 \\ & - \mathrm{trace}\Big(Y_i^T\Psi\Big) \end{aligned}$$

$$\begin{aligned} \Psi = \mathrm{Diag}\Big\{ & \mathrm{diag}\Big(\mathrm{vec}\Big\{2\Lambda_i(k+1) - \Lambda_i(k)\Big\}\Big), \\ & \mathrm{diag}\Big(\mathrm{vec}\Big\{2\Gamma_i(k+1) - \Gamma_i(k)\Big\}\Big), \\ & 2\Sigma_i(k+1) - \Sigma_i(k) - \frac{\epsilon_\Sigma}{n}I_{2n}\Big\}. \end{aligned}$$

(v) Update the sum variable $S_i(k)$:

$$\forall\ i\in\mathcal{N}:\ \ S_i(k+1) = Y_i(k+1) + \sum_{l=0}^{k+1} Y_i(l). \tag{41}$$

(vi) Repeat until $k = N_{\mathcal{DPDS}}$.

Control agent $i \in \mathcal{N}_c$ only needs to know: (a) its neighbouring set $\mathcal{N}_i$ (b) the $I_c(i)$-th row of matrices (B, L_1) and (c) entire matrices (A, L_2). Non control nodes only need information regarding $\mathcal{N}_i$ since their single purpose is to update and exchange dual information i.e., variables $Y_i(k)$ and $S_i(k)$. The next theorem summarizes the convergence results of $\mathcal{DPDS}$:

Theorem 9. *For any stabilizable pair $(A, B) \in \mathbf{R}^{n\times n} \times \mathbf{R}^{n\times m}$, bound $\gamma > 0$, connected graph $\mathcal{G}$ and positive scalars $(\epsilon_P, \epsilon_\Sigma)$ assume problem (31) is feasible. As $N_{\mathcal{DPDS}} \to +\infty$ algorithm 5 converges to a controller K^* such that $k_i^* \in \mathcal{S}_i$, $\rho(A+BK^*) < \gamma$, $A+BK \geq 0$ and $L_1 K^* \geq L_2$. Controller $K^{*T} = \left(k_j^*\right)_{j=1}^{m}$ is given by:*

$$\forall\ i \in \mathcal{N}_c,\ \ j \in \{1, \ldots, m\}: \tag{42}$$
$$I_c(i) = j \ \ \Rightarrow\ \ k_j^{*T} = g_i(N_{\mathcal{DPDS}})^T \Big\{ \sum_{i \in \mathcal{N}_c} Q_i(N_{\mathcal{DPDS}}) \Big\}^{-1}.$$

V. Numerical Results

A. Centralized Computation of Decentralized Controllers

This section is devoted to the numerical comparison of the heuristics for centralized sparse stabilization. The experimental setup is fairly simple:

- For $n = 10$ and $m = 3$ we randomly generate $N = 30$ pairs (A, B) and define L as total number of zero entries for a controller $K \in \mathcal{K}_I$. Given L, we generate N random sparsity sets each with a total of L zero constraints. For each generation we run all six heuristics: the incremental inverse algorithm is deployed for a fixed bound $\gamma = 1$, in a bisection setup starting from γ and under a reduction strategy. We also consider the decentralized convex concave procedure and the two variants of the incremental PK approach.

In figure 1 we have considered the previous experiment for $L = 10$ respectively, and under a standard Gaussian Generation for the pair (A, B). Each circle represents an experiment, identified by the colour legend underneath each plot. We have decided to only plot successful trials, i.e., experiments for which we could produce a decentralized stabilizing controller. The colour legend also informs the reader about the number of successful experiments, for each heuristic, and some statistics regarding the original open loop systems.

In table I we present a comparison of heuristics for $L = 10$ and $L = 15$. The equivalent of figure 1, for $L = 15$, is not presented due to lack of space. Table I compares all heuristics in terms of (a) mean spectral radius of stable experiments (b) mean computational time and (c) percentage of successful experiments. The two values, for each entry of table I, correspond to the information regarding the experiments with $L = 10, 15$ respectively. For low sparsity counts, $L = 10$ the bisection incremental inverse method was able to stabilize all 30 systems while exhibiting the better overall spectral

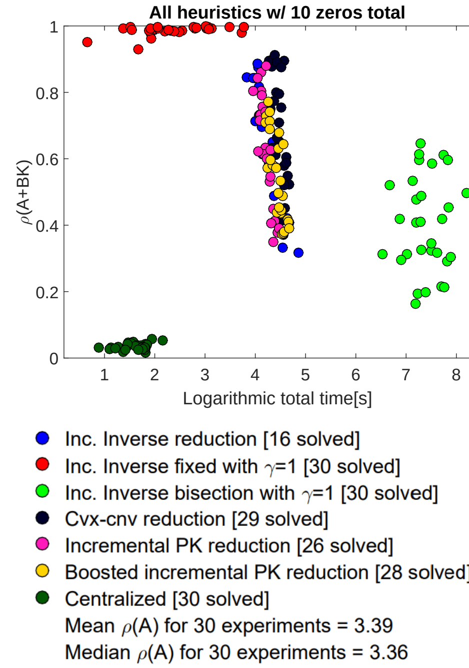

Fig. 1: Standard Gaussian Generation and $L = 10$.

results, as seen in figure 1 and table I. The drawback is that this method is 20-30 times slower than any other reduction mode heuristic, hence there is a clear compromise. If we only want to impose closed loop stability, the same method with a constant bound $\gamma = 1$ is a fast alternative that has the drawback of providing a spectral radius close to unit. For high sparsity levels, $L = 15$ the bisection incremental inverse was able to achieve the largest stability percentage (77%) but the boosted incremental PK algorithm is a suitable alternative that, although with only 40 % of stabilizations, was able to achieve considerable better spectral results while being 40 times faster.

In terms of all reduction mode algorithms, we elect the second and last methods of table I as computational efficient alternatives for the bisection approach: although they decrease, 7-37 % the stability percentages they present the better spectral results, among all reduction mode approaches, while exhibiting a low computational cost. In figure 1, note the low horizontal spread of reduction mode methods while compared with the bisection incremental inverse heuristic.

B. Optimal Sparsity Patterns

The goal of this section is to provide a numerical comparison of all previous oracles in terms of computational time and resulting spectral radius. Heuristic 2 is selected as algorithm $\mathcal{A}$. Consider the following experimental setup:

- Given a budget vector Φ we generate 100 stabilizable pairs (A, B) with $n = 10$, $m = 3$ under a standard Gaussian distribution. For each generation, the two oracles of section III-B are executed, generating two corresponding feasible sparsity sets. In the incremental inverse iterative oracle, each call to the incremental inverse heuristic

Heuristic	Mean $\rho(A+BK)$	Mean Comput. Time [min]	% Stable Exp.
Incremental PK	$0.64 \rightarrow 0.76$	$1.03 \rightarrow 0.67$	$87 \rightarrow 43$
Boosted Incremental PK	$0.55 \rightarrow 0.68$	$1.36 \rightarrow 0.82$	$93 \rightarrow 40$
Incr. Inverse fixed mode w/ $\gamma = 1$	$0.99 \rightarrow 0.99$	$0.23 \rightarrow 1.58$	$100 \rightarrow 73$
Incr. Inverse bisection w/ $\gamma = 1$	$0.40 \rightarrow 0.78$	$28.9 \rightarrow 33.2$	$100 \rightarrow 77$
Incr. Inverse reduction	$0.65 \rightarrow$ —-	$0.92 \rightarrow 0.39$	$53 \rightarrow 00$
Decentralized convex concave	$0.69 \rightarrow 0.86$	$1.40 \rightarrow 1.12$	$97 \rightarrow 43$

TABLE I: Comparison of heuristics: Standard Gaussian Generation.

runs for a maximum of $N_{oracle} = 20$ iterations. As an baseline, we also consider a purely randomized approach: given a budget vector Φ we randomly generate a feasible assignment of measurements to control components. For each oracle we infer on the optimality of the previously generated sparsity sets, by executing algorithm $\mathcal{A}$.

Figure 2 and table II present some results when one considers a control budget of $[8, 6, 3]^T$. In table II, $t_{\mathcal{O}}$ denotes the amount of time associated with each oracle and t_T is the total time needed to generate the underlying decentralized controller: oracle time + time of algorithm $\mathcal{A}$. The abscissa in figure 2 corresponds to t_T. The intuition of section III-B2

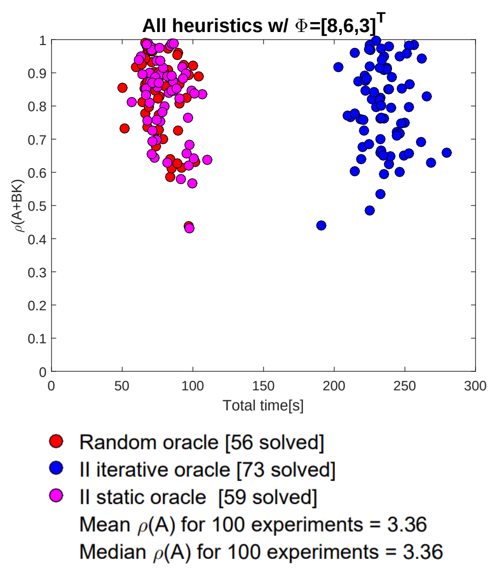

Fig. 2: Standard Gaussian Generation with budget = $[8, 6, 3]^T$.

is experimentally verified: the oracle that uses incremental information tends to outperform the non incremental approaches. Note the stabilization percentages of $\mathcal{O}^i_{II}$. Oracle $\mathcal{O}^i_{II}$ is approximately 120 times slower then any other method.

Again we find numerical evidences similar to that of section V-A: the slowest method provides better numerical results. The iterative method also provides slightly better mean spectral results, as observed in the point clouds of figure 2 and the first column of table II. The previous conclusions indicate that oracle $\mathcal{O}^i_{II}$ provides better numerical results while being considerable slower. The time performance of algorithm 3 can be easily improved: simply lower N_{oracle} which is the maximum number of iterations for each incremental inverse call. By re-executing oracle $\mathcal{O}^i_{II}$ for the same budget Φ and pairs (A, B), the stability percentages improved from 73% to 84% while oracle $\mathcal{O}^i_{II}$, with $N_{oracle} = 3$, is approximately seven times faster then with $N_{oracle} = 20$. Although a reduction of N_{oracle} is beneficial, the decrease should not be too drastic. For $N_{oracle} = 1$, the computational time gained ($\approx$ 8s) is not worth it since stability margins dropped 18%.

C. Decentralized Computation of Decentralized Controllers

In this section we provide numerical confirmation of the convergence properties, stated in theorem 9. To this end, consider the following experimental setup:

- Given some pair (n, m) we sample state space matrices (A, B) from an uniform distribution in the interval $[a, b]$. The m control nodes and graph $\mathcal{G}$ are also chosen at random. We are imposing a non-positive control law, i.e., $(L_1, L_2) = (-I_m, 0_{m\times n})$. Let K_c denote a sparse centralized controller computed from theorem 8 setting $\gamma = 1$. For the same bound, algorithm 4 is deployed for $(N_{\mathcal{DPDS}}, N_c)$ iterations, imposing $\epsilon_\Sigma = 10^{-7}$ and $\epsilon_P = 1$. We have selected W as the fastest Laplacian weight consensus matrix [15] (section 4.2, equation 23).

Figure 3 display some results where K_d denotes the final controller produced by $\mathcal{DPDSC}$. For this experiment, note that the pair (A, B) is entry wise non-positive, hence the controller is enforcing positiveness in the closed loop system.

In table III we compare the centralized and decentralized approaches for the generations of figure 3. The computational time associated with the decentralized method is given by the sum of two terms: the time of $\mathcal{DPDS}$ plus the time of n consensus rounds, respectively.

Controller	K_c	K_d
$\rho(A+BK)$	0.7993	0.5450
$\min_{(i,j)}\left\{A+BK\right\}_{i,j}$	0.0255	5.12011×10^{-4}
$\min_{(i,j)}\left\{L_1K - L_2\right\}_{i,j}$	0	0
Computational Time [min]	$\frac{0.3930}{60}$	$91.84 + \frac{0.0058}{60}$

TABLE III: Centralized vs Decentralized controller Synthesizes: data of figure 3.

Regarding the consensus plot of figure 3 we have chosen the third diagonal entry of matrix (35) to display the convergence properties, i.e., legend $Q_{3,3}$ corresponds to the $(3, 3)$ entry of

Heuristic	Mean $\rho(A+BK)$	Mean $t_{\mathcal{O}}$ [s]	Mean t_T [s]	% Stable Exp.
II static oracle	0.82	1.24	72.88	59
II iterative oracle	0.79	152.10	231.23	73
Random oracle	0.83	10^{-8}	74.51	56

TABLE II: Comparison of Oracles: Standard Gaussian Generation with budget $= [8,6,3]^T$.

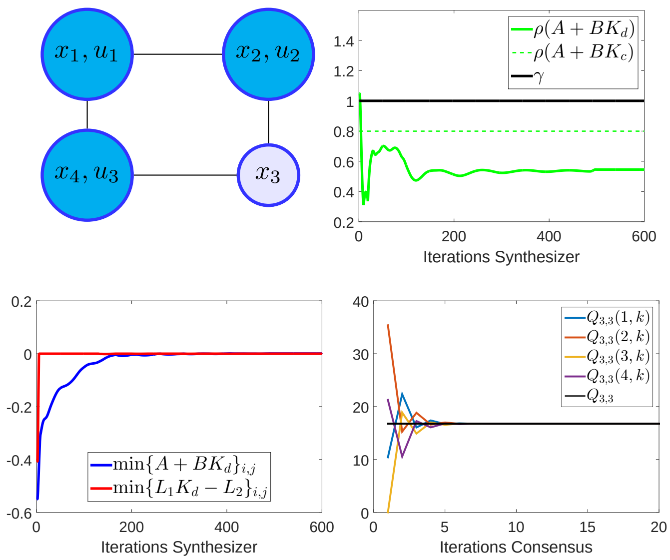

Fig. 3: Achieved results with $(N_{\mathcal{DPDS}}, N_c, n, m, a, b) = (600, 20, 4, 3, -0.5, 0)$ and a random connected graph $\mathcal{G}$.

(35). Note that $\mathcal{DPDSC}$ was able to compute a much stable controller, comparing with the centralized version. There is no contradiction since we simply require a stabilizing controller, i.e., the bound γ was set to one. An interesting observation: $\mathcal{DPDS}$ will guarantee closed loop stability long before ensuring positiveness.

VI. Conclusion

Section II presented several methods for controller synthesizes, given a pre-defined sparsity pattern. We assessed their performance through extensive numerical experiments. The decentralized convex concave method and the boosted incremental-PK approach were elected as the most suitable algorithms, achieving acceptable stabilization margins while being computational fast. In section III, our main insight was to use perturbation theory arguments to develop sparsity design strategies. Both our oracles were shown to be effective, since they outperformed the randomized design option. Section IV approaches an even more difficult problem: the decentralized computation of sparse stabilizing controllers. Having rigorously defined our understanding of the problem, we presented a truly decentralized design strategy, for the particular instance of closed loop positive systems. Although the positiveness assumption can be restrictive, $\mathcal{DPDSC}$ is based on necessary and sufficient conditions; hence, it is highly attractive for realistic applications that demand some kind of performance guarantee. Our experiments motivate further investigation of $\mathcal{DPDSC}$ for general closed loop system.

References

[1] V. Blondel and J. N. Tsitsiklis, "Np-hardness of some linear control design problems," in *Proceedings of 1995 34th IEEE Conference on Decision and Control*, vol. 3, pp. 2910–2915 vol.3, Dec 1995.

[2] A. Alavian and M. C. Rotkowitz, "Constructive Stabilization and Pole Placement by Arbitrary Decentralized Architectures," *ArXiv e-prints*, Apr. 2017.

[3] M. S. Sadabadi and D. Peaucelle, "From static output feedback to structured robust static output feedback: A survey," *Annual Reviews in Control*, vol. 42, pp. 11 – 26, 2016.

[4] J. Geromel, J. Bernussou, and P. Peres, "Decentralized control through parameter space optimization," *Automatica*, vol. 30, no. 10, pp. 1565 – 1578, 1994.

[5] C. A. R. Crusius and A. Trofino, "Sufficient lmi conditions for output feedback control problems," *IEEE Transactions on Automatic Control*, vol. 44, pp. 1053–1057, May 1999.

[6] E. J. Cands, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted l1 minimization," vol. 14, pp. 877–905, 11 2007.

[7] A. Hassibi, J. How, and S. Boyd, "Low-authority controller design via convex optimization," in *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No.98CH36171)*, vol. 1, pp. 140–145 vol.1, 1998.

[8] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[9] Q. T. Dinh, S. Gumussoy, W. Michiels, and M. Diehl, "Combining convex-concave decompositions and linearization approaches for solving bmis, with application to static output feedback," *IEEE Transactions on Automatic Control*, vol. 57, pp. 1377–1390, June 2012.

[10] A. Hassibi, J. How, and S. Boyd, "A path-following method for solving bmi problems in control," in *Proceedings of the 1999 American Control Conference (Cat. No. 99CH36251)*, vol. 2, pp. 1385–1389 vol.2, Jun 1999.

[11] B. T. Polyak, M. V. Khlebnikov, and P. S. Shcherbakov, "Sparse feedback in linear control systems," *Autom. Remote Control*, vol. 75, pp. 2099–2111, Dec. 2014.

[12] F. Lin, M. Fardad, and M. R. Jovanovi, "Sparse feedback synthesis via the alternating direction method of multipliers," in *2012 American Control Conference (ACC)*, pp. 4765–4770, June 2012.

[13] A. Rantzer, "Distributed Control of Positive Systems," *ArXiv e-prints*, Feb. 2012.

[14] N. S. Aybat and E. Y. Hamedani, "Distributed primal-dual method for multi-agent sharing problem with conic constraints," in *2016 50th Asilomar Conference on Signals, Systems and Computers*, pp. 777–782, Nov 2016.

[15] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, vol. 5, pp. 4997–5002 Vol.5, Dec 2003.

[16] O. Toker and H. Ozbay, "On the np-hardness of solving bilinear matrix inequalities and simultaneous stabilization with static output feedback," in *American Control Conference, Proceedings of the 1995*, vol. 4, pp. 2525–2526 vol.4, Jun 1995.

[17] M. Fu and Z.-Q. Luo, "Computational complexity of a problem arising in fixed order output feedback design," *Systems & Control Letters*, vol. 30, no. 5, pp. 209 – 215, 1997.

[18] R. Skelton, T. Iwasaki, and K. Grigoriadis, "A unified algebraic approach to control design," 05 2018.

[19] T. Lipp and S. Boyd, "Variations and extension of the convex–concave procedure," *Optimization and Engineering*, vol. 17, pp. 263–287, Jun 2016.

[20] D. P. Bertsekas, A. Nedić, and A. E. Ozdaglar, *Convex Analysis and Optimization*. Athena Scientific, 2003.