Decentralized Optimization in Remote Estimation

Sebin Mathew

Department of Electrical and Computer Engineering
McGill University
Montréal, Quebec

A thesis submitted to McGill University in partial fulllment of the requirements for the degree of Master of Engineering.

Abstract

In many emerging applications, multiple sensors transmit their measurements to a remote estimator over a shared medium. In such a system, the optimal sampling rates at each sensor depend on the nature of the stochastic process being observed as well as the available communication capacity. Our main contribution is to show that the problem of determining optimal sampling rates may be posed as a network utility maximization problem and solved using appropriate modifications of the standard dual decomposition algorithms for network utility maximization. We present two such algorithms —synchronous and asynchronous— and show that under mild technical conditions, both algorithms converge to the optimal rate allocation. We present a detailed simulation study to illustrate that the asynchronous algorithm is able to adapt the sampling rate to changes in the number of sensors and the available channel capacity and is robust to network delays and packet drops.

Sommaire

Dans de nombreuses applications émergentes, plusieurs capteurs transmettent leurs mesures à un estimateur distant, situé sur un support partagé. Dans un tel système, les taux d'échantillonnage optimaux à chaque capteur dépendent de la nature du processus stochastique observé ainsi que de la capacité de communication disponible. Notre contribution principale est de montrer que le problème de détermination des taux d'échantillonnage optimaux peut se présenter sous la forme d'un problème de maximisation de l'utilité du réseau et résolu en utilisant les modifications appropriées des algorithmes standard de décomposition double pour la maximisation de l'utilité du réseau. Nous présentons deux de ces algorithmes - synchrones et asynchrones - et montrons que sous des conditions techniques modérées, les deux algorithmes convergent vers l'allocation de débit optimale. Nous présentons une étude de simulation détaillée pour illustrer que l'algorithme asynchrone est capable d'adapter le taux d'échantillonnage aux changements dans le nombre de capteurs et la capacité de canal disponible. Il est résistant à la lenteur du réseau et aux diminutions de paquets.

Acknowledgements

Firstly, I would like to thank Prof. Aditya Mahajan for his help, support and guidance on both academic and personal level over the past two years. I am deeply grateful for his advice and help on my thesis and in making research an enriching experience for me. I also thank him for the financial support I was offered through a graduate internship opportunity at Interdigital Inc., Canada under MITACS Accelerate program.

Secondly, I am grateful to Dr. Benoît Pelletier at Interdigital for the discussions we had during my internship. I also thank Prof. Karl Johansson at KTH, Sweden for his ideas that contributed to my thesis and feel grateful to have had an opportunity to discuss research with him.

I would also like to thank my friends Aman, Ashish, Madhusudhan and Mitali for making my experience at Montréal a memorable one. I also want to thank Pradeepta and Chandini for their camaraderie over the years.

For all the sacrifices they have made, and for all the support they have shown without which none of life's achievements would have been possible, I thank my parents and my sister for being there for me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we provide an overview of the technical problem to be studied. Then the existing literature aimed at solving this problem is surveyed. Next, we summarize the contributions made by this thesis. Finally, the thesis organization along with certain mathematical notations are explained.

1.1 Problem Overview

Many systems like sensor networks consist of several network nodes that can observe their environment and communicate with other nodes in the network. The information collected is often some state variable of the environment or a certain process, which has to be communicated to a central unit for estimation. Different nodes communicate over capacity constrained networks. We consider a setup where a system of sensors observe stationary Gauss-Markov processes and communicate the state information to a remote estimator over a shared medium with limited capacity. The estimator needs to produce estimates of the state of the processes in real-time. Given the limited and time-varying availability of capacity of the communication channel, each sensor has to periodically transmit its sampled observations at optimal rates to minimize the mean-squared error at the remote estimator. Each sensor at its current sampling rate contributes to the estimation error and hence is associated with a utility function in terms of the estimation error it generates while observing a certain stochastic state process. The objective is to minimize the aggregate mean-squared error at the estimator, which may be viewed as the network utility of the sensor-remote estimator system.

In addition to the limited channel resource, the network is subject to changes over time with sensors operating between active and sleep modes. A non-ideal channel entails delays and packet-drops. A distributed approach to schedule sensors for optimal sampling provides an efficient method of resource allocation and also significantly lowers signalling overhead in the network, which facilitates the application of such a rate allocation scheme to large networks.

In this thesis, we seek to develop decentralized algorithms for optimal rate allocation to sensors communicating with a remote estimator over a shared medium, by treating the objective as a variant of the network utility maximization problem. We provide simulation details and numerical results on some practical concerns with the implementation of these algorithms in practical networks.

1.2 Literature Survey

Advances in wireless communication have enabled the development and deployment of wireless sensor networks with applications ranging from environment monitoring, healthcare, home automation, and so on. Sensors in a wireless sensor network have limited power, computation, and communication capabilities and operate over a time-varying network. In applications where remote estimation is critical, each sensor periodically samples its observation and transmits the sampled measurements to a remote estimator over the network. When these sensors share a medium, the available capacity becomes a limited resource that has to be allocated optimally to prevent congestion and minimize estimation error at the remote estimator.

Various variants of remote estimation and sensor scheduling under communication constraints have been investigated in the literature. Research on the general sensor scheduling problem dates back to the 1970s [1]. Traditional forms of problem formulation are such that only a subset of the sensor network can be selected at any instant of time to observe the output of a linear stochastic system. Each sensor is associated with a measurement cost while predicting the future state of the system using measurements over a finite time horizon. In [2], the authors proposed a stochastic sensor scheduling scheme and provided the optimal probability distribution over the sensors to be selected. Sensor scheduling for discrete-time state estimation using a Kalman filter was studied in [3]. Design trade-offs between estimation performance, processing delay and communication cost for a sensor scheduling problem over a heterogeneous network was discussed in [4]. A typical class of problems is to find optimal offline sensor schedule in terms of the estimation error covariance under different resource constraints [5], [6], [7]. Despite the advantage of low computation capacity requirement and simple implementation, offline methods work inefficiently. Offline sensor scheduling means the sensor is scheduled to take observation or conduct communication based on some a priori information about the system (e.g. statistics of random variables, system matrices). The online information (e.g. sensor observation, battery energy level) is not taken into account when making schedules [8]. With the advances in hardware devices, sensors are endowed with stronger computational capabilities. Consequently, the sensors are able to make schedules based on all the information they have (a priori information as well as online information), which motivates the formulation of online sensor scheduling problems.

Event-based schedules where a sensor communicates with the remote estimator only when a pre-defined event happens often has superior performance compared to using time-based schedules. In [9] it was shown that optimal observer policy for a scalar linear system stochastic process with limited measurements has a solution in an event-triggered form. The authors of [10] considered the design of an event-triggering rule to compute the minimum mean-squared error of the state estimate at the remote estimator subject to a constraint on how frequently the information can be transmitted. The work of [11], [12] focussed on sensor scheduling to achieve desired balance between sensor-to-estimator communication rate and the estimation quality for event-based sampling. In [13], the proposed schedule provides a tradeoff between time-based and event-based schedules for networks with limited communication resources. The traditional approach to monitor the system state is to sample and send the signals periodically. The problem of finding optimal time-periodic sensor schedules for estimating the state of discrete-time dynamical systems was discussed in [14]. In [15], [16] an event-triggered control (ETC) strategy is proposed by striking a balance between conventional periodic sampled-data control and ETC, leading to so-called periodic event-triggered control (PETC). In PETC, the event-triggering condition is verified periodically and at every sampling time it is decided whether or not to compute and to transmit new measurements and new control signals. A decentralized ETC scheme for multiple sensors sampling the system output was studied in [17] to select only those necessary sampled-data packets to be transmitted to save communication resources. Self-triggered control, which is proactive and computes the next sampling instance ahead of time is introduced in [18].

Resource allocation is a fundamental problem in shared communication networks. An efficient resource allocation strategy ensures successful sharing of the communication channel among sensors while maximizing system performance as a whole. On the other hand, stability, fairness and robustness of rate control algorithms is critical [19]. Resource allocation problems are typically framed as optimization problems where the objective is to maximize aggregate sensor utility over their transmission rates. The authors of [20] provide synchronous and asynchronous distributed algorithms for such a network utility maximization problem and prove their convergence in a static environment. They also illustrate the convergence of the algorithms in a slowly time-varying environment. Applying decomposition techniques to the global optimization problem allows us to identify critical information that needs to be communicated between nodes and across layers, and suggests how network elements should react to this information in order to attain the global optimum. Decentralized techniques for utility-maximizing protocols - primal, dual, and cross decomposition were studied in [19], [20], [21], [22].

1.3 Thesis Contributions

Sensor data scheduling has been a hot topic due to its practical importance as well as the technical challenges involved. A typical problem formulation considers an optimal schedule to minimize the terminal estimation error covariance within a finite time horizon T under the constraint that that only $n < T$ measurements could be taken and the sensor can communicate with the remote estimator only $d \ll T$ times. The goal is to study discrete-time sensor scheduling and remote estimation problems employing Kalman-like filtering to compute the estimate and associated error covariance [23], [24]. The two different types of constraints that arise in the works mentioned above are hard and soft constraints that specify the frequency of communication and cost of communication respectively. The traditional approach to monitor the system by sampling and sending the signals periodically allows for easy implementation.

In this thesis, we consider a sensor-remote estimator system where each sensor observes a continuous-time stochastic process, samples its observation and sends the sampled measurements to the remote estimator over a network with finite capacity. Our goal is to minimize the mean-squared error at the estimator across all sensors and provide the optimal sampling strategy. Our approach shows that the above objective can be posed as a variant of the network utility maximization problem. We describe two decentralized algorithms, namely the synchronous and the asynchronous dual decomposition algorithms, based on an optimization approach to allocate rates, and prove their convergence. While the synchronous algorithm requires that updates at the sensors and the remote estimator be synchronized, the asynchronous algorithm is an online algorithm and allows for feedback signals to be delayed and time-varying, without having to reset or establish a hand-shaking protocol among the sensors with changes in network conditions.

We rely on extensive simulation studies to demonstrate that the asynchronous algorithm is robust to slowly changing network conditions. The simulations study the impact of random sensor arrivals and departures, changing channel capacity, and packet drops on the asynchronous algorithm. Although convergence of the dual decomposition algorithms is proven for an ideal channel, simulations confirm that the asynchronous algorithm is not significantly impacted by non-ideal effects.

Dissemination of the Work

The results based on this thesis will be submitted to the 57th IEEE Conference on Decision and Control as
S. Mathew, K. H. Johannson, and A. Mahajan, *”Optimal sampling of multiple linear processes over a shared medium”*

1.4 Thesis Organization

The rest of the thesis is structured as follows. Chapter 2 reviews basic concepts of convex optimization and decomposition theory, highlighting the primal and dual decomposition techniques. Illustrating the idea behind decomposition with an example, a general framework for decomposition structures is discussed. Formulation of a network utility maximization problem and its features are elaborated in Chapter 3. Minimizing the mean-squared error under capacity constraint is shown to be similar to the above problem under mild technical conditions. The proof of convergence of the two dual decomposition algorithms is also provided. Simulation results are presented and discussed in Chapter 4 to validate the convergence of the dual decomposition algorithms discussed in the previous chapter and to demonstrate the robustness of the asynchronous algorithm to changes in network conditions in practical networks. Finally, Chapter 5 draws conclusions of the whole thesis and indicates possible research directions for the future.

Notation. Vectors and matrices are denoted by bold lower-case and upper-case letters, respectively. Superscript ^{T} represents transposition. $\mathbb{R}$ denotes the set of real numbers and $\mathbb{Z}$ denotes the set of integers. $[.]^+$ denotes projection onto the positive orthant, that is, $[x]^+ = \max\{0, x\}$. The $:=$ symbol means that the variable on the left is now defined to take the value of the expression on the right upon evaluation.

Chapter 2

Decomposition Theory

In this chapter, we will review the basics of convex optimization, describe the standard primal and dual decomposition techniques and provide a general framework for decomposition structures.

2.1 Introduction

Efficiently solving a utility maximization problem requires investigation at three important levels. The first one is to check if the objective satisfies conditions of convex optimization. If so, a local optimum is also a global optimum and the duality gap is zero under mild conditions. The second one is to analyze the computational properties so that the rate of convergence to the optimal solution can be deduced. Fast and scalable polynomial-time algorithms such as interior-point methods exist. These are basically centralized algorithms. In large scale networks, a third point of inspection is if the objective can be decomposed into sub-problems that allow for distributed algorithms that converge to the global optimum. Distributed solutions are particularly attractive in such networks where a centralized solution is infeasible, non-scalable, too costly or too fragile.

The importance of "decomposability" to distributed solutions is similar to that of "convexity" to efficient computation of global optimum. Often non-convex optimization can be transformed into a convex one. Similarly, alternative problem representations may reveal hidden decomposability structures without changing the optimal solution. The choice for a distributed algorithm depends on the specifics of the application and is mainly dependent on characteristics such as rate and robustness of convergence, tradeoff between local computation and global communication, and quantity and symmetry of message passing.

Decomposition theory provides a mathematical framework to build an analytical foundation for the design of modular and distributed control of networks. However, unlike the well-defined notion of convexity, decomposability is not strictly defined. It is often quantified by the least amount of global communications needed among decomposed nodes to solve a problem and ease of implementability. In this chapter, we will review the basic concepts of convex optimization and Lagrange duality, touching upon gradient/subgradient algorithms and convergence properties. We will then introduce the basic techniques of primal and dual decomposition and study the general framework for decomposition structures.

2.2 Basic Concepts

2.2.1 Convex Optimization and Lagrange Duality

The generic optimization problem is as follows:

$$\begin{array}{lll} \underset{\mathbf{x}}{\text{minimize}} & f_0(\mathbf{x}) & \\ \text{subject to} & f_i(\mathbf{x}) \le 0 & 1 \le i \le m, \\ & h_i(\mathbf{x}) = 0 & 1 \le i \le p \end{array} \tag{2.1}$$

where $\mathbf{x} \in \mathbb{R}^N$ is the variable we wish to optimize, f_0 is the cost or objective function, $f_1,\ldots, f_m$ are the m inequality constraint functions, and $h_1,\ldots,h_p$ are the p equality constraint functions. The problem is a *convex optimization* problem if the objective and inequality constraints are convex and the equality constraint functions are linear (more generally, affine). A point $\mathbf{x}$ in the domain of the problem is *feasible* if it satisfies all the constraints $f_i(\mathbf{x}) \le 0$, $i \in \{1, 2, \ldots, m\}$, and $h_i(\mathbf{x}) = 0$, $i \in \{1, 2, \ldots, p\}$. The problem (2.1) is *feasible* if there exists one such feasible point and *infeasible* otherwise.

The optimal solution that minimizes the objective is denoted by $\mathbf{x}^*$ and the optimal value is given by $f^* = f_0(\mathbf{x}^*)$.

The property of Lagrange duality of convex optimization problems leads to decomposability structures. The original minimization problem (2.1) termed primal problem is linked with a dual maximization problem, which sometimes directly gives rise to decomposition possibilities. Relaxing the original problem (2.1) by transferring the constraints to the objective in the form of a weighted sum, the Lagrangian is formulated as

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = f_0(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i f_i(\mathbf{x}) + \sum_{i=1}^{p} \nu_i h_i(\mathbf{x}) \tag{2.2}$$

where λ_i and ν_i are the *Lagrangian multipliers* associated with the ith inequality constraint $f_i(\mathbf{x}) \leq 0$ and with the ith equality constraint $h_i(\mathbf{x}) = 0$ respectively. The optimization variable is called the primal variable and the Lagrangian multipliers $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_m]$ and $\boldsymbol{\nu} = [\nu_1, \ldots, \nu_p]$ are termed the dual variables. The dual variables, from an economics point of view, are often termed as shadow prices. We assume that the objective pays this fictitious price to reach its optimum. The dual objective $g(\boldsymbol{\lambda}, \boldsymbol{\nu})$ is defined as the minimum value of the Lagrangian over $\mathbf{x}$

$$g(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \inf_{\mathbf{x}} \quad L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) \tag{2.3}$$

The dual objective is concave even if the original problem is not convex because it is the pointwise infimum of a family of affine functions of $(\boldsymbol{\lambda}, \boldsymbol{\nu})$. The dual variables are feasible if $\boldsymbol{\lambda} \geq 0$. An important result that holds for any feasible $\mathbf{x}$ and $(\boldsymbol{\lambda}, \boldsymbol{\nu})$ is that the primal and dual objectives satisfy $f_0(\mathbf{x}) \geq g(\boldsymbol{\lambda}, \boldsymbol{\nu})$. This allows for the dual function to be maximized therefore providing a lower bound on the optimal value f^* of the original problem (2.1)

$$\begin{aligned} \underset{\boldsymbol{\lambda}, \boldsymbol{\nu}}{\text{maximize}} \quad & g(\boldsymbol{\lambda}, \boldsymbol{\nu}) \\ \text{subject to} \quad & \boldsymbol{\lambda} \geq 0 \end{aligned} \tag{2.4}$$

which is always a convex optimization problem even if the original problem is not convex. The duality gap is defined as the difference between optimal primal objective f^* and the optimal dual objective g^*, which is always non-negative. As such, weak duality means $f^* - g^* > 0$. Under some mild conditions (Slater's condition), the duality gap reduces to zero (strong duality) at the optimal solution. Therefore, the primal problem (2.1) can be equivalently solved by solving the dual problem (2.4).

After performing a decomposition, the original objective function decouples into two levels: a master problem, and a set of sub-problems, each of which interacts with the master problem. The objective of the resulting master problem may or may not be differentiable. For differentiable/non-differentiable functions a gradient/subgradient method is very convenient because of its simplicity, little requirements of memory usage, and amenability for parallel implementation. Consider the following general convex minimization over a convex set:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{X} \end{aligned} \tag{2.5}$$

Both the gradient and subgradient projection methods generate a sequence of feasible points $\{\mathbf{x}(k)\}$ as

$$\mathbf{x}(k+1) = \big[\mathbf{x}(k) + \alpha(k)\mathbf{s}(k)\big]_{\mathcal{X}}$$

where $\mathbf{s}(k)$ is a gradient of f_0 evaluated at $\mathbf{x}(k)$ if f_0 is differentiable and a subgradient otherwise, $[.]_{\mathcal{X}}$ denotes the projection onto the feasible set $\mathcal{X}$, and $\alpha(k)$ is a positive step-size. It must be noted that while each iteration of the gradient method improves the objective value, the same is not true for the subgradient method. However, what makes the subgradient method converge is that, for a sufficiently small step size, the distance of the current solution $\mathbf{x}(k)$ at iteration k to the optimal solution $\mathbf{x}^*$ decreases [25].

There are many results on convergence of the gradient/subgradient method with different choices of step-sizes. A diminishing step-size rule $\alpha(k) = (1+m)/(k+m)$, where m is a fixed nonnegative number, guarantees the convergence of the algorithm to the optimal value assuming bounded gradients/subgradients. For a constant step-size $\alpha(k) = \alpha$, more convenient for distributed algorithms, the gradient algorithm converges to the optimal value provided that the step-size is sufficiently small, whereas for the subgradient algorithm, the best value converges within some range of the optimal value.

Whenever a primal or dual decomposition is applied, the original problem is essentially decomposed into a master and subproblems with communication between the two levels. This communication is in the form of message passing which introduces control overhead in the network. In certain cases, however, this message passing can be implicit in the system, e.g., delay and packet error probability, as these quantities have physical meanings and can be implicitly measured without the need of explicit signalling. The decomposition techniques presented in this chapter have a provable convergence to the global optimum of the original problem which will be assumed convex. The rate of convergence is more difficult to quantify and depends on many factors such as the number of levels of decomposition, the amount of signalling, and the particular combination of numerical algorithms employed.

2.3 Decomposition Theory

Most of the existing decomposition techniques can be categorized into primal and dual decomposition methods. In primal decomposition, the original primal problem is decomposed into subsystems that coordinate to choose some values of primal variables. In dual decomposition, the Lagrangian dual problem is decomposed as dual variables (prices) are manipulated to solve the global problem. The solvable subproblems are then coordinated by a high-level master problem by means of some kind of signalling [26].

2.3.1 Primal Decomposition

Consider a simple case of unconstrained optimization of the form

$$\underset{\mathbf{x},\mathbf{y}}{\text{minimize}} \quad f_1(x_1, y) + f_2(x_2, y) \tag{2.6}$$

The objective in (2.6) is block separable in x_1 and x_2 save for the coupling variable y. To solve the subproblems independently, we fix the coupling variable y. We can think of $x_1(x_2)$ as the *private variable* or *local variable* associated with the first(second) subproblem, and y as the *public variable* or *interface variable* or *boundary variable* between the subproblems. Fixing y, let ϕ_1(y) and ϕ_2(y) be defined as below:

$$\begin{aligned} \phi_1(y) &= \underset{x_1}{\text{minimize}} \quad f_1(x_1, y) \\ \phi_2(y) &= \underset{x_2}{\text{minimize}} \quad f_2(x_2, y) \end{aligned} \tag{2.7}$$

The subproblems in (2.7) are convex if f_1 and f_2 are convex and the original problem (2.6) is now equivalent to the following problem

$$\underset{\mathbf{y}}{\text{minimize}} \quad \phi_1(y) + \phi_2(y) \tag{2.8}$$

If (2.6) is convex, so is the master problem (2.8). The variables of the master problem are the coupling variables of the original problem and the objective now is the sum of optimal values of the subproblems. The master problem can be solved using an iterative method such as the gradient method (assuming the function is differentiable). Each iteration requires solving (2.7) and their gradients. Let $\tilde{x_1}$ and $\tilde{x_2}$ solve (2.7) and let $g_1 \in \partial\phi_1(y)$ and $g_2 \in \partial\phi_2(y)$ be the gradients. Algorithm 2.1 summarizes the primal decomposition algorithm for unconstrained optimization

Algorithm 2.1 Primal Decomposition Unconstrained

```
repeat
    Solve the subproblems (preferably in parallel)
        Find $\tilde{x_1}$ that minimizes $f_1(x_1, y)$ and a subgradient $g_1 \in \partial\phi_1(y)$
        Find $\tilde{x_2}$ that minimizes $f_2(x_2, y)$ and a subgradient $g_2 \in \partial\phi_2(y)$
    Update complicating variable
        $y := y - \alpha_k\big(g_1 + g_2\big)$
```

Here α_k is the step length and can be chosen in any standard way. In each iteration of the algorithm, a subgradient for the master problem is constructed from the two local solutions of subproblems, and using this the coupling variable is updated. Each update moves the coupling variable in a direction of improvement of the overall objective. We can also consider the case where the two subproblems are coupled via constraints that involve both sets of variables. Suppose our problem has the form

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f_1(x_1) + f_2(x_2) \\ \text{subject to} \quad & x_1 \in \mathcal{C}_1, \quad x_2 \in \mathcal{C}_2 \\ & h_1(x_1) + h_2(x_2) \leq 0 \end{aligned} \tag{2.9}$$

Here $\mathcal{C}_1$ and $\mathcal{C}_2$ are feasible sets of the subproblems and the functions h_1 and h_2 are convex. To use primal decomposition, we can introduce a variable t that represents the amount of the resources allocated to the first subproblem. As a result, $-t$ is allocated to the second subproblem. The subproblems then become

$$\begin{aligned} \text{minimize} \quad & f_1(x_1) \\ \text{subject to} \quad & x_1 \in \mathcal{C}_1, \quad h_1(x_1) \leq t, \\ \\ \text{minimize} \quad & f_2(x_2) \\ \text{subject to} \quad & x_2 \in \mathcal{C}_2, \quad h_2(x_2) \leq -t \end{aligned} \tag{2.10}$$

Let $\phi_1(t)$ and $\phi_2(t)$ denote the optimal values of the subproblems in (2.10). The original problem (2.9) is now equivalent to the master problem of minimizing $\phi(t) = \phi_1(t) + \phi_2(t)$ over the allocation vector t. Keeping t fixed, the subproblems in (2.10) can be solved independently. The subgradient for the optimal value of each subproblem in (2.10) can be found out from an optimal dual variable associated with the coupling constraint while framing the Lagrangian and dual function corresponding to that subproblem. By this, we mean that to solve the subproblems in (2.10), we need to frame the Lagrangian $L_1(x_1, \lambda_1, t)$ and $L_2(x_2, \lambda_2, t)$ for the two subproblems like we solve a generic convex optimization problem. We thus have

$$\begin{aligned} \text{minimize} \quad & f_1(x_1) - \lambda_1(h_1(x_1) - t) \\ \text{subject to} \quad & x_1 \in \mathcal{C}_1 \\ \\ \text{minimize} \quad & f_2(x_2) - \lambda_2(h_2(x_2) + t) \\ \text{subject to} \quad & x_2 \in \mathcal{C}_2 \end{aligned} \tag{2.11}$$

Let λ_1^* and λ_2^* be the optimal dual variables. These can be found by setting $\frac{\partial L_i}{\partial x_i} = 0$ and $\frac{\partial L_i}{\partial \lambda_i} = 0$, $i = 1, 2$. Then, $\lambda_1^* - \lambda_2^* \in \partial\phi(t)$. The primal decomposition algorithm for the constrained case is given below as Algorithm 2.2.

Algorithm 2.2 Primal Decomposition Constrained

```
repeat
  Solve the subproblems (preferably in parallel)
    Solve (2.10), to find optimal $x_1^*$, $x_2^*$, $\lambda_1^*$ and $\lambda_2^*$
  Update resource allocation
    $t := t - \alpha_k\big(\lambda_2^* - \lambda_1^*\big)$
```

It should be noted that the iterates must be projected onto the feasible set for the solution to be feasible. Primal decomposition methods correspond to a direct resource allocation since the master problem allocates the existing resources by directly giving each subproblem the amount of resources it can use. This method works well when there are few coupling variables, and we have good or fast methods for solving the subproblems. At every step of a primal decomposition algorithm we have points that are feasible for the original problem. This is because for any resource t that the master problem allocates, $h_1(x_1) \le t$ and $h_2(x_2) \le -t$ imply that $h_1(x_1) + h_2(x_2) \le 0$. The intuition behind $\lambda_1^* - \lambda_2^* \in \partial\phi(t)$ is that at optimality, for the resources allocated by the master problem, the subproblems pay equal price.

2.3.2 Dual Decomposition

We can apply dual decomposition to (2.6) by first expressing the problem as

$$\begin{aligned} \underset{\mathbf{x},\mathbf{y}}{\text{minimize}} \quad & f(\mathbf{x},\mathbf{y}) = f_1(x_1, y_1) + f_2(x_2, y_2) \\ \text{subject to} \quad & y_1 = y_2 \end{aligned} \tag{2.12}$$

by introducing a new variable and an equality constraint. Along with the introduction of a *local version* of the complicating variable y, a *consistency constraint* that requires the two local versions to be equal has also been added. The objective is now separable and the Lagrangian is

$$L(x_1, y_1, x_2, y_2, \nu) = f_1(x_1, y_1) + f_2(x_2, y_2) + \nu y_1 - \nu y_2$$

which is separable and the dual function is

$$g(\nu) = g_1(\nu) + g_2(\nu)$$

where

$$\begin{aligned} g_1(\nu) &= \inf_{x_1,y_1} \quad (f_1(x_1,y_1) + \nu y_1) \\ g_2(\nu) &= \inf_{x_2,y_2} \quad (f_2(x_2,y_2) - \nu y_2) \end{aligned}$$

Both g_1 and g_2 can be evaluated independently. The master problem of dual decomposition therefore is

$$\text{maximize} \quad g_1(\nu) + g_2(\nu) \tag{2.13}$$

Algorithm 2.3 Dual Decomposition Unconstrained

```
repeat
  Solve the subproblems (preferably in parallel)
    Find $\tilde{x}_1$ and $\tilde{y}_1$ that minimize $f_1(x_1, y_1) + \nu y_1$
    Find $\tilde{x}_2$ and $\tilde{y}_2$ that minimize $f_2(x_2, y_2) - \nu y_2$
  Update dual variables (prices)
    $\nu := \nu - \alpha_k(\tilde{y}_2 - \tilde{y}_1)$
```

To evaluate a subgradient of $-g_1$ (or $-g_2$) is easy. We find $\tilde{x}_1$ and $\tilde{y}_1$ that minimize $f_1(x_1, y_1) + \nu y_1$ over x_1 and y_1. Then a subgradient of $-g_1$ at ν is given by $-\tilde{y}_1$. Similarly, a subgradient of $-g_2$ at ν is given by $\tilde{y}_2$. From these, one can calculate the subgradient of the negative of (2.13) as $\tilde{y}_2 - \tilde{y}_1$. Using the subgradient method to solve the master problem, the dual decomposition algorithm has the form as given above in Algorithm 2.3.

Dual decomposition under constraints as in (2.9) is straightforward. The Lagrangian is formed as

$$\begin{aligned} L(x_1, x_2, \lambda) &= f_1(x_1) + f_2(x_2) + \lambda(h_1(x_1) + h_2(x_2)) \\ &= \Big(f_1(x_1) + \lambda h_1(x_1)\Big) + \Big(f_2(x_2) + \lambda h_2(x_2)\Big) \\ &= L_1(x_1, \lambda) + L_2(x_2, \lambda) \end{aligned}$$

which is separable, so we can minimize over x_1 and x_2 separately, given the dual variable λ, to find g(λ) $= g_1(\lambda) + g_2(\lambda)$. Since minimizing the original objective is equivalent to maximizing the dual function, the problem reduces to

$$\begin{array}{ll} \text{maximize} & g(\lambda) \\ \text{subject to} & \lambda \geq 0 \end{array} \tag{2.14}$$

To find $g_1(\lambda)$ and $g_2(\lambda)$, we solve the following subproblems

$$\begin{array}{ll} \text{minimize} & f_1(x_1) + \lambda h_1(x_1) \\ \text{subject to} & x_1 \in \mathcal{C}_1, \\ & \\ \text{minimize} & f_2(x_2) + \lambda h_2(x_2) \\ \text{subject to} & x_2 \in \mathcal{C}_2 \end{array} \tag{2.15}$$

A subgradient of $-g(\lambda)$ is $h_1(\tilde{x}_1) + h_2(\tilde{x}_2)$ where $\tilde{x}_1$ and $\tilde{x}_2$ are solutions of (2.15). The iterates need not be feasible in the dual decomposition algorithm. This is because as can be seen above, the gradient for the dual objective is nothing but the constraint for the original problem. Since convergence of the algorithm implies the constraint is met, the iterates will be feasible only at convergence. The algorithm is summarized in Algorithm 2.4.

Algorithm 2.4 Dual Decomposition Constrained

```
repeat
  *Solve the subproblems (preferably in parallel)*
    Solve (2.15) to find optimal $\tilde{x}_1$ and $\tilde{x}_2$
  *Update dual variables (prices)*
    $\lambda := \Big[\lambda - \alpha_k\big(h_1(\tilde{x}_1) + h_2(\tilde{x}_2)\big)\Big]^+$
```

Dual decomposition methods correspond to resource allocation via pricing since the master problem sets the prices for the resources to each subproblem, which has to decide the amount of resources to be used depending on the price.

2.3.3 Illustrative Example

Let us now look at a simple example illustrating the above decomposition methods. Let $f(x_1, x_2) = 2x_1^2 + 3x_2^2$ for $\mathbf{x} \in \mathbb{R}^2$. Here $\mathbf{x} = \left[x_1, x_2\right]$ is the primal optimization variable. Consider the following problem:

$$\begin{aligned} \underset{x_1, x_2}{\text{minimize}} \quad & f(x_1, x_2) \\ \text{subject to} \quad & x_1 > 0, x_2 > 0 \\ & x_1 + 2x_2 = 10 \end{aligned}$$

Here, $f(x_1, x_2) = f_1(x_1) + f_2(x_2)$, where $f_1(x_1) = 2x_1^2$ and $f_2(x_2) = 3x_2^2$. The equality constraint is $g(\mathbf{x}) = x_1 + 2x_2 - 10$. Using the primal decomposition approach, we decompose the above problem into two sub-problems given by

$$\begin{aligned} \underset{x_1}{\text{minimize}} \quad & 2x_1^2 \\ \text{subject to} \quad & x_1 > 0 \\ & x_1 = t \end{aligned}$$

$$\begin{aligned} \underset{x_2}{\text{minimize}} \quad & 3x_2^2 \\ \text{subject to} \quad & x_2 > 0 \\ & 2x_2 - 10 = -t \end{aligned}$$

The master problem allocates resources t and $-t$ to both the sub-problems. To find the optimal value $\phi_1(t)$ of sub-problem 1 for a particular t is now straightforward. We form the Lagrangian as

$$L_1(x_1, \nu_1) = 2x_1^2 - \nu_1(x_1 - t)$$

Setting $\frac{\partial L_1}{\partial x_1} = 0$ we have

$$\nu_1 = 4x_1$$

Similarly, for the optimal value $\phi_2(t)$ of sub-problem 2,

$$\frac{\partial L_2(x_2, \nu_2)}{\partial x_2} = 0$$

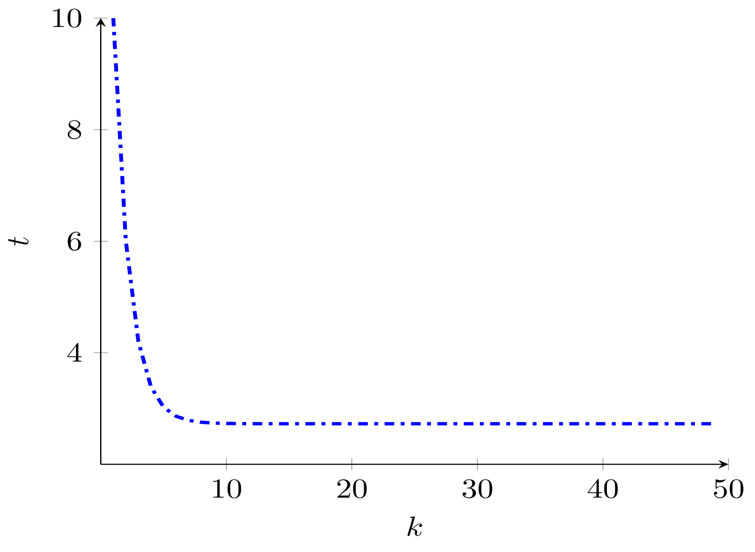


Fig. 2.1: Plot of resource t versus iteration under Algorithm 2.2 for the illustrative example in Section 2.3.3

$$\frac{\partial\left(3x_2^2 - \nu_2(2x_2 - 10 + t)\right)}{\partial x_2} = 0$$

$$\nu_2 = 3x_2$$

The update rule then becomes

$$t := t - \alpha(\nu_1 - \nu_2)$$

We begin with an initial guess of $t = 10$ and $\alpha = 0.1$. Iteration number is denoted by k. After about 50 iterations, the value of t converges to about 2.73 as shown in Fig. 2.1. The primal variables x_1 and x_2 converge to approximately 2.73 and 3.63 respectively. As shown in Fig. 2.2, in the primal decomposition approach, the primal variables satisfy the constraint at all times.

Denoting the value of the objective at the kth iteration as f_k, the distance between f_k and the optimum value f^* is plotted in Fig. 2.3. As stated previously, $\phi_1(t)$, $\phi_2(t)$ and their sum $\phi(t)$ are all convex if the original problem is convex. The minimum value of $\phi(t)$ is f^*. From Fig. 2.4, $f^* = 54.54$ which is attained at $t = 2.73$

Let us now take the dual decomposition route for the same problem. We first

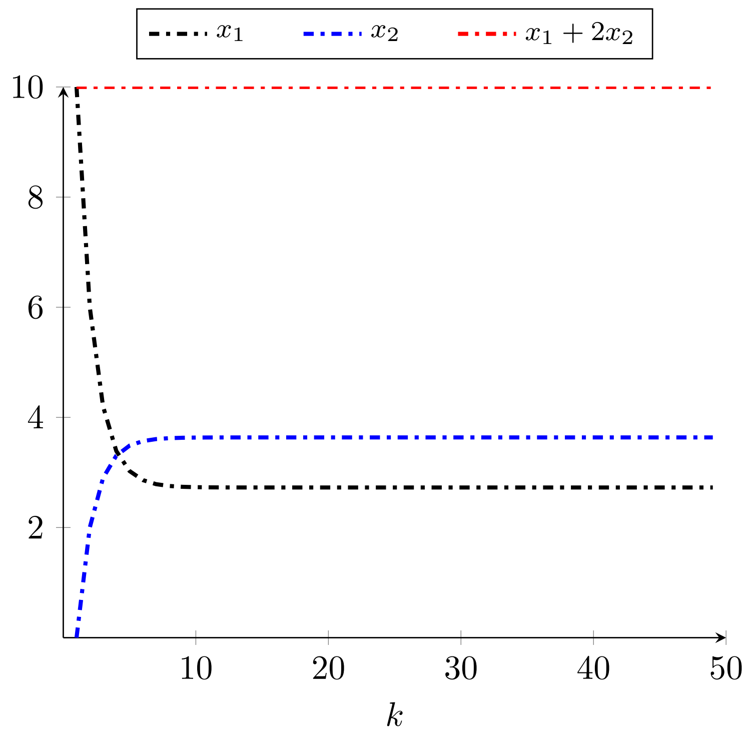


Fig. 2.2: Evolution of primal variables x_1 and x_2 for the primal decomposition algorithm

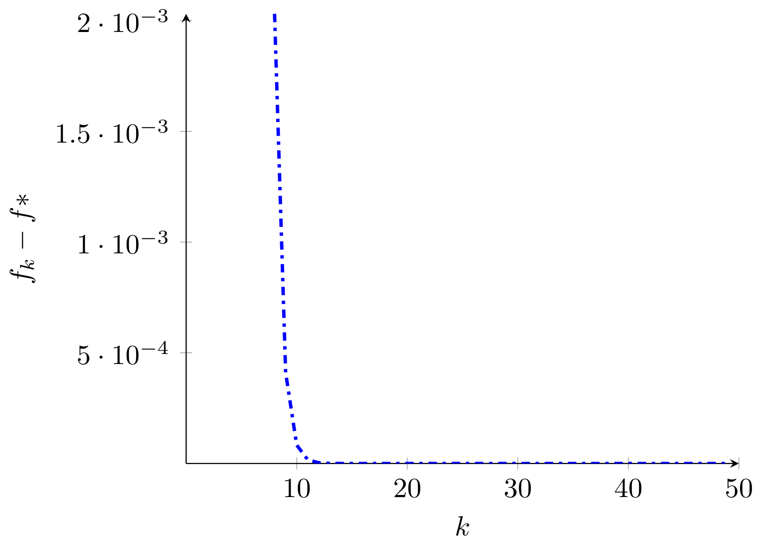


Fig. 2.3: Distance between f_k and f^* versus iteration for the primal decomposition algorithm

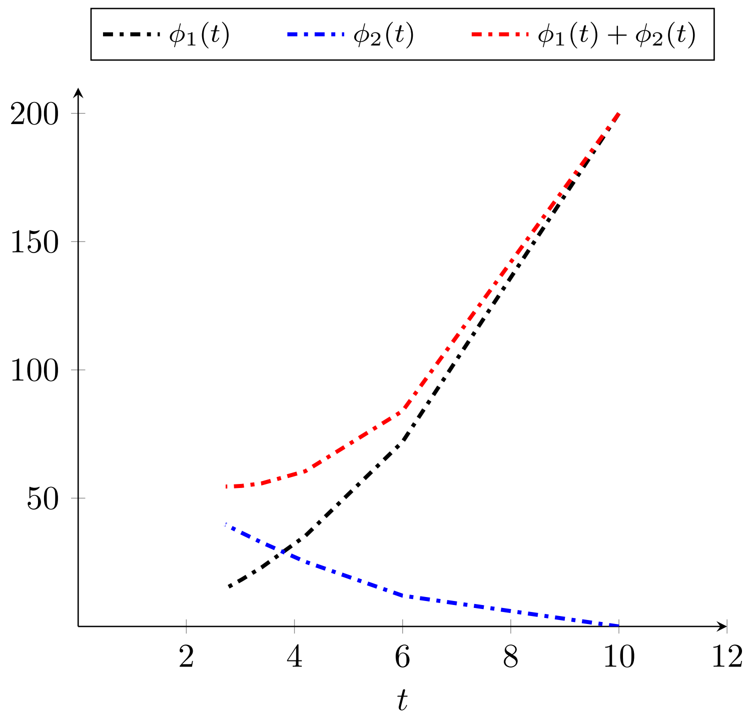


Fig. 2.4: Optimal values of master problem and subproblems versus resource t for the illustrative example under Algorithm 2.2

form the Lagrangian as

$$L(x_1, x_2, \nu) = 2x_1^2 + 3x_2^2 - \nu(x_1 + 2x_2 - 10)$$

This can be rewritten as

$$L(x_1, x_2, \nu) = \left(2x_1^2 - \nu x_1\right) + \left(3x_2^2 - 2\nu x_2 + 10\nu\right)$$

This separates the original problem into two sub-problems given by

$$\begin{aligned} \underset{x_1}{\text{minimize}} \quad & 2x_1^2 - \nu x_1 \\ \text{subject to} \quad & x_1 > 0 \end{aligned}$$

$$\begin{aligned} \underset{x_2}{\text{minimize}} \quad & 3x_2^2 - 2\nu x_2 + 10\nu \\ \text{subject to} \quad & x_2 > 0 \end{aligned}$$

The optimal value for sub-problem 1 is

$$\begin{aligned} g_1(\nu) &= \inf_{x_1} L_1(x_1, \nu) \\ &= \inf_{x_1} \left(2x_1^2 - \nu x_1\right) \end{aligned}$$

Setting $\frac{\partial L_1}{\partial x_1} = 0$ we have

$$4x_1 = \nu$$

Similarly,

$$\begin{aligned} g_2(\nu) &= \inf_{x_2} L_2(x_2, \nu) \\ &= \inf_{x_2} \left(3x_2^2 - 2\nu x_2 + 10\nu\right) \end{aligned}$$

Setting $\frac{\partial L_2}{\partial x_2} = 0$ we have

$$3x_2 = \nu$$

The update rule then becomes

$$\nu := \Big[\nu - \alpha(x_1 + 2x_2 - 10)\Big]^+$$

We begin with an initial guess of $\nu = 1$ and take $\alpha = 0.1$. After about 150 iterations, ν settles at around 10.09 as shown in Fig. 2.5. x_1 and x_2 converge to 2.73 and 3.63 as in the primal decomposition case. However, as shown in Fig. 2.6, the dual iterates satisfy the equality constraint only at the optimal point. The distance between f_k and f^* is shown in Fig. 2.7. Note that the dual functions are concave as shown in Fig. 2.8 and the master dual function $g(\nu)$ reaches a maximum value of 54.54 at $\nu = 10.09$. This maximum is the required f^*.

2.3.4 Framework for decomposition structures

In this section, we describe decomposition with a general structure in more detail. A primal decomposition is appropriate when the problem has a coupling variable such that, when fixed to some value, the rest of the optimization problem decouples into several subproblems. A dual decomposition is appropriate when the problem has a coupling constraint such that, when relaxed, the optimization problem decouples into several independent subproblems.

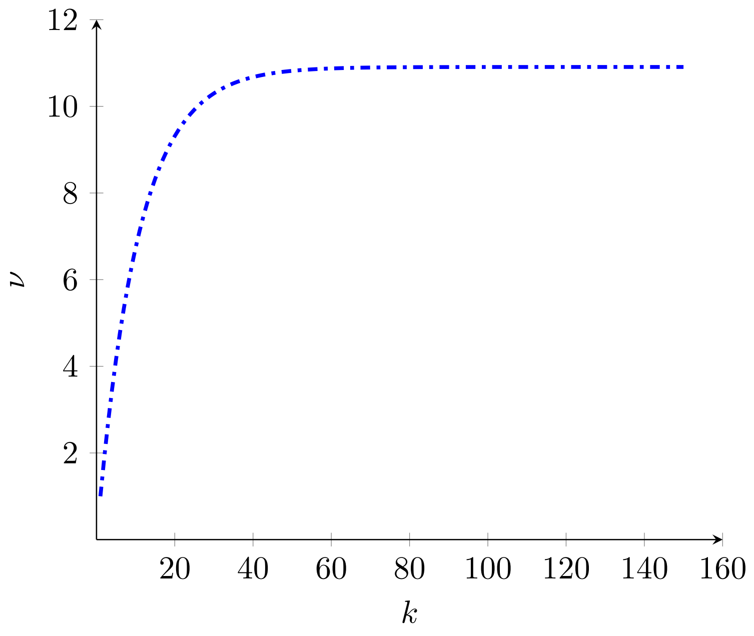


Fig. 2.5: Plot of price ν versus iteration under Algorithm 2.4 for the illustrative example in Section 2.3.3

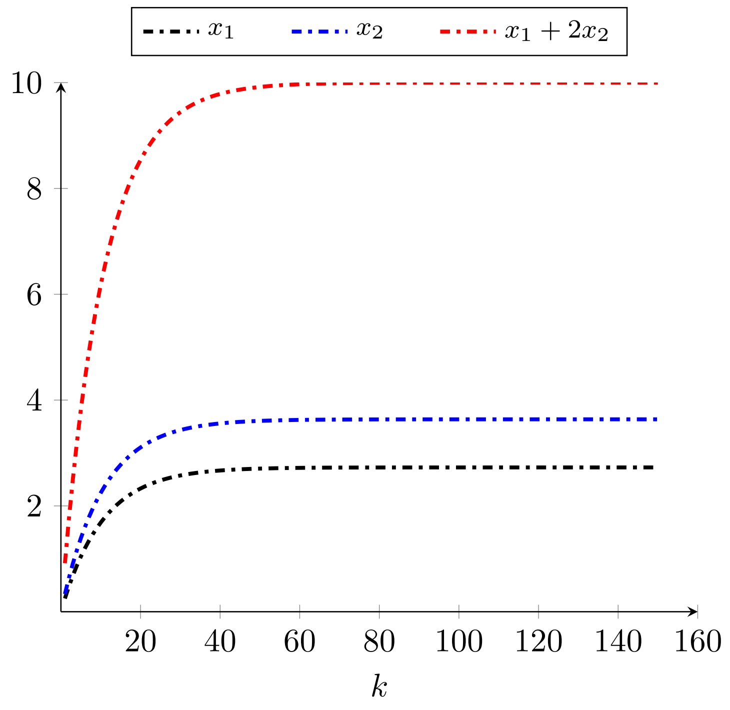


Fig. 2.6: Evolution of primal variables x_1 and x_2 for the dual decomposition algorithm

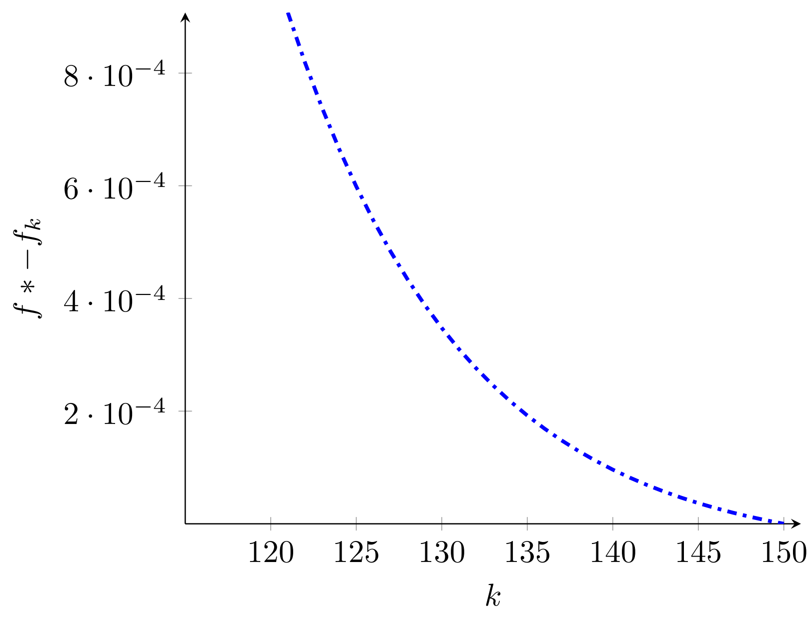


Fig. 2.7: Distance between f^* and f_k vs iteration for the dual decomposition algorithm

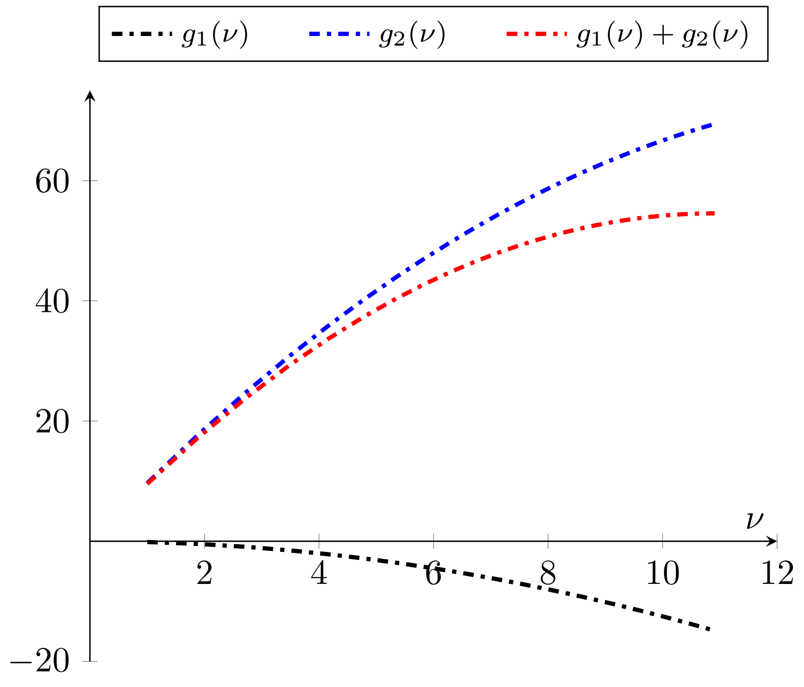


Fig. 2.8: Optimal values of master problem and subproblems versus price ν for the illustrative example under Algorithm 2.4

Primal Decomposition

$$\begin{aligned} \underset{\mathbf{y},\{\mathbf{x}_i\}}{\text{maximize}} \quad & \sum_i f_i(\mathbf{x}_i) \\ \text{subject to} \quad & \mathbf{x}_i \in \mathcal{X}_i \quad \forall i \\ & \mathbf{A}_i\mathbf{x}_i \leq \mathbf{y} \\ & \mathbf{y} \in Y \end{aligned} \tag{2.16}$$

where $\mathbf{x} = \left[\mathbf{x}_1^T, \ldots, \mathbf{x}_n^T\right]^T$, the feasible set is the Cartesian product of closed convex sets $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_n$ and Y is the feasible set for $\mathbf{y}$. Clearly, if variable $\mathbf{y}$ was fixed, then the problem would decouple. At the lower level we have the subproblems, one for each i, in which (2.16) decouples when $\mathbf{y}$ is fixed

$$\begin{aligned} \underset{\{\mathbf{x}_i\}}{\text{maximize}} \quad & f_i(\mathbf{x}_i) \\ \text{subject to} \quad & \mathbf{x}_i \in \mathcal{X}_i \\ & \mathbf{A}_i\mathbf{x}_i \leq \mathbf{y} \end{aligned} \tag{2.17}$$

At the higher level, we have the master problem in charge of updating the coupling variable $\mathbf{y}$ by solving

$$\begin{aligned} \underset{\mathbf{y}}{\text{maximize}} \quad & \sum_i f_i^*(\mathbf{y}) \\ \text{subject to} \quad & \mathbf{y} \in Y \end{aligned} \tag{2.18}$$

where $f_i^*(\mathbf{y})$ is the optimal objective value of problem (2.17) for a given $\mathbf{y}$. The subproblems as well as the master problem are convex if the original problem is convex. In general, the objective of master problem (2.18) may be non-differentiable, and the subgradient method is a convenient approach which only requires the knowledge of a subgradient for each $f_i^*(\mathbf{y})$ given by

$$\mathbf{s}_i(\mathbf{y}) = \lambda_i^*(\mathbf{y})$$

where $\lambda_i^*(\mathbf{y})$ is the optimal Lagrange multiplier corresponding to the constraint $\mathbf{A}_i\mathbf{x}_i \leq \mathbf{y}$ in (2.17). The global subgradient is $\mathbf{s}(\mathbf{y}) = \sum_i \mathbf{s}_i(\mathbf{y})$. The subproblems in (2.17) can be locally and independently solved with the knowledge of $\mathbf{y}$.

Dual Decomposition

Consider the following problem:

$$\begin{aligned} \underset{\{\mathbf{x}_i\}}{\text{maximize}} \quad & \sum_i f_i(\mathbf{x}_i) \\ \text{subject to} \quad & \mathbf{x}_i \in \mathcal{X}_i \quad \forall i \\ & \sum_i \mathbf{h}_i(\mathbf{x}_i) \leq \mathbf{c} \end{aligned} \tag{2.19}$$

The Lagrangian is formed by relaxing the coupling constraint in (2.19) as

$$\begin{aligned} \underset{\{\mathbf{x}_i\}}{\text{maximize}} \quad & \sum_i f_i(\mathbf{x}_i) - \boldsymbol{\lambda}^T \Big(\sum_i \mathbf{h}_i(\mathbf{x}_i) \ - \ \mathbf{c} \Big) \\ \text{subject to} \quad & \mathbf{x}_i \in \mathcal{X}_i \quad \forall i \end{aligned} \tag{2.20}$$

At the lower level, we have the subproblems, one for each i, in which (2.20) decouples

$$\begin{aligned} \underset{\{\mathbf{x}_i\}}{\text{maximize}} \quad & \sum_i f_i(\mathbf{x}_i) - \boldsymbol{\lambda}^T \mathbf{h}_i(\mathbf{x}_i) \\ \text{subject to} \quad & \mathbf{x}_i \in \mathcal{X}_i \end{aligned} \tag{2.21}$$

At the higher level, we have the master dual problem in charge of updating the dual variable $\boldsymbol{\lambda}$ by solving the dual problem:

$$\begin{aligned} \underset{\lambda}{\text{minimize}} \quad & \sum_i g_i(\lambda) + \boldsymbol{\lambda}^T \mathbf{c} \\ \text{subject to} \quad & \boldsymbol{\lambda} \geq \mathbf{0} \end{aligned} \tag{2.22}$$

where $g_i(\boldsymbol{\lambda})$ is the dual function obtained as the maximum value of the Lagrangian solved in (2.21) for a given $\boldsymbol{\lambda}$. Since this approach solves the dual problem, it will give appropriate results only if strong duality holds. The subgradient for each $g_i(\boldsymbol{\lambda})$ is given by

$$\mathbf{s}_i(\mathbf{y}) = -\mathbf{h}_i(\mathbf{x}_i^*(\boldsymbol{\lambda}))$$

where $\mathbf{x}_i^*(\boldsymbol{\lambda})$ is the optimal solution of (2.21) for a given $\boldsymbol{\lambda}$. The global subgradient is then $\boldsymbol{\lambda} = \sum_i \mathbf{s}_i(\mathbf{y}) + \mathbf{c}$. The subproblems in (2.21) can be locally and independently solved with knowledge of $\boldsymbol{\lambda}$.

2.4 Chapter Summary

In this chapter, a general overview of the importance of decomposition in the context of formulating distributed algorithms was discussed and the standard decomposition techniques, namely the primal and dual decomposition were studied. We observed that the master problem of the dual decomposition technique is always concave, although the duality gap still depends on the convexity of the objective function (in particular, Slater's conditions need to be satisfied). Dual decomposition also requires more iterations to converge which makes the choice of step size α_k crucial. We now have the mathematical tools to approach a network utility maximization problem which will serve as a motivation for the system model and problem formulation that we will discuss in the next chapter.

Chapter 3

Network Utility Maximization

In the previous chapter, we discussed decomposition theory and how it naturally provides the mathematical language to build an analytic foundation for the design of modular and distributed control of networks. In this chapter, we will discuss an optimization approach to flow control where the objective is to maximize the aggregate source utility over their transmission rates. We will in particular, look at dual decomposition applied to flow control in communication networks and use that as a motivation to formulate our problem of resource allocation for a sensor network connected to a remote estimator. We will show that the above problem can be treated as a variant of the network utility maximization problem

3.1 Introduction

Consider a communication network with $L = \{1, \dots, |L|\}$ unidirectional links, each with a fixed capacity of c_l, and $S = \{1, \dots, |S|\}$ sources or nodes, each transmitting at a source rate of x_s. Each source s emits one flow, using a fixed set of links $L(s)$ in its path, and has a utility function $U_s(x_s)$. The path $L(s) \subset L$ is a set of links that source s uses. The network is as shown in Fig. 3.1. We assume $m_s \geq 0$ and $M_s < \infty$ are the minimum and maximum transmission rates, respectively, required by source s. The network utility maximization (NUM) problem is that of maximizing the total utility $\sum_s U_s(x_s)$, over the source rates $\mathbf{x}$, subject to linear flow constraints

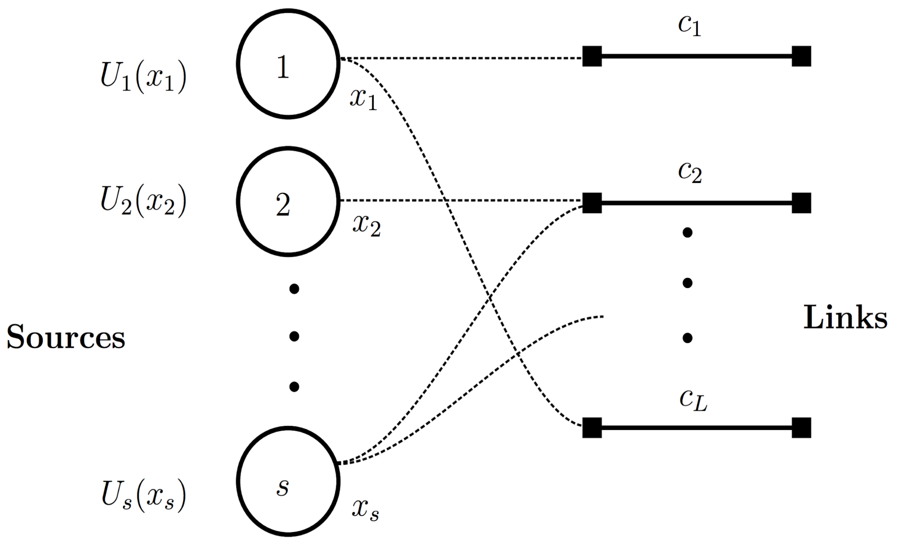


Fig. 3.1: Communication network with sources and links.

$\sum_{s:l\in L(s)} x_s \le c_l$ for all links l

$$\begin{aligned} \underset{\mathbf{x}\geq 0}{\text{maximize}} \quad & \sum_s U_s(x_s) \\ \text{subject to} \quad & \sum_{s:l\in L(s)} x_s \le c_l \quad \forall l \end{aligned} \tag{3.1}$$

where the utilities U_s are twice-differentiable, increasing, and strictly concave functions. A unique maximizer, called the primal optimal solution, exists since the objective function is strictly concave, and hence continuous, and the feasible solution set is compact.

Though the objective function is separable in x_s, the source rates x_s are coupled by the capacity constraint. Solving the above primal problem directly requires coordination among possibly all sources and is impractical in real networks. The key to a distributed and decentralized solution is to look at its dual. One of the standard distributed algorithms to solve (3.1) is based on a dual decomposition. We first form the Lagrangian of (3.1) as

$$\begin{aligned} L(\mathbf{x}, \boldsymbol{\lambda}) &= \sum_s U_s(x_s) + \sum_l \lambda_l \bigg(c_l - \sum_{s:l \in L(s)} x_s \bigg) \\ &= \sum_s L_s(x_s, \lambda^s) + \sum_l c_l \lambda_l \end{aligned} \tag{3.2}$$

where $\lambda_l \geq 0$ is the Lagrange multiplier (link price) associated with the linear flow constraint on link l, $\lambda^s = \sum_{l \in L(s)} \lambda_l$ is the aggregate path congestion price of those links used by source s, and $L_s(x_s, \lambda^s) = U_s(x_s) - \lambda^s x_s$ is the sth Lagrangian to be maximized by the sth source. The dual decomposition results in each source s solving, for the given λ^s as

$$x_s^*(\lambda^s) = \underset{x_s \geq 0}{\operatorname{argmax}} \quad [U_s(x_s) - \lambda^s x_s] \quad \forall s \tag{3.3}$$

which is unique due to the strict concavity of U_s.

The master dual problem is

$$\begin{aligned} &\underset{\boldsymbol{\lambda}}{\text{minimize}} \quad g(\boldsymbol{\lambda}) = \sum_s g_s(\boldsymbol{\lambda}) + \boldsymbol{\lambda}^T \boldsymbol{c} \\ &\text{subject to} \quad \boldsymbol{\lambda} \geq 0 \end{aligned} \tag{3.4}$$

where $g_s(\boldsymbol{\lambda}) = L_s(x_s^*(\lambda^s), \lambda^s)$. Since the solution to (3.2) is unique, it follows that the dual function $g(\boldsymbol{\lambda})$ is differentiable and the following gradient method can be used:

$$\lambda_l(t+1) = \left[\lambda_l(t) - \alpha \bigg(c_l - \sum_{s:l \in L(s)} x_s^*(\lambda^s(t)) \bigg) \right]^+ \quad \forall l \tag{3.5}$$

where t is the iteration index and $\alpha > 0$ is a sufficiently small positive step size.

The dual variable $\boldsymbol{\lambda}(t)$ will converge to the dual optimal $\boldsymbol{\lambda}^*$ as $t \to \infty$ and, since the duality gap for (3.1) is zero and the solution to (3.3) is unique, the primal variable $x^*(\boldsymbol{\lambda}(t))$ will also converge to the primal optimal variable $\mathbf{x}^*$.

There is no need for explicit message passing since λ^s can be measured by each source s as queing delay and $\sum_{s:l \in L(s)} x_s$ can be measured by each link l as the total traffic load.

The network utility maximization problem is a prime example of the application of dual decomposition where network links and sources are viewed as processors of a

Algorithm 3.1 Dual Algorithm to solve (3.1)

```
• Parameters: each source needs its utility $U_s$ and each link its capacity $c_l$.
• Initialization: set $t = 0$ and $\lambda_l(0)$ equal to some nonnegative value for all $l$.
1) Each source locally solves its problem by computing (3.3) and then broadcasts
   the solution $x_s^*(\lambda^s(t))$.
2) Each link updates its prices with the gradient iterate (3.5) and broadcasts
   the new price $\lambda_l(t+1)$.
3) Set $t \leftarrow t+1$ and go to step 1 (until satisfying termination criterion).
```

distributed computation system to solve the dual problem using gradient projection algorithm. In this system, sources select transmission rates that maximize their own benefits, utility minus bandwidth cost, and network links adjust bandwidth prices to coordinate the sources' decisions.

We will now proceed to describe the system model for sensors communicating with a remote estimator over a capacity limited channel where the objective is to determine optimal sampling rates to minimize mean-squared error at the estimator.

3.2 Problem Model and Formulation

3.2.1 Model

Consider a remote estimation system in which n sensors are connected to a remote estimator over a network. The sensors are indexed by the set $N := \{1, \dots, n\}$. Sensor i, $i \in N$, observes a continuous-time stochastic process $\{X_i(t)\}_{t\geq 0}$, $X_i(t) \in \mathbb{R}$, where $X_i(0) \sim \mathcal{N}(0, \theta)$ and for $t \geq 0$

$$dX_i(t) = a_i X_i(t)dt + \sigma_i dW_i(t)$$

where $a_i \in \mathbb{R}$ is a known constant and $\{W_i(t)\}_{t\geq 0}$, $W_i(t) \in \mathbb{R}$, is a stationary stochastic process with finite variance and satisfies a technical assumption (A1) that we state later. We assume that the initial states $\{X_i(0)\}$, $i \in N$, and the processes

$\{W_i(t)\}_{t\geq 0}$, $i \in N$, are independent. This implies that the stochastic processes $\{X_i(t)\}_{t\geq 0}$, $i \in N$, are independent across sensors.

Sensor i, $i \in N$, samples its observation at a rate of $R_i = 1/T_i$ and sends the sampled measurements to the remote estimator over the network. It is assumed that the network is ideal and does not cause any delays or packet drops.[1] However, the network has a finite capacity C and the rates $\mathbf{R} = (R_1, \ldots, R_n)$ must lie in the rate region

$$\mathcal{R} = \left\{(R_1, \ldots, R_n) \in \mathbb{R}^n_{\geq 0} : \sum_{i=1}^{n} R_i = C\right\}.$$

In between the sampling instances, the estimator generates estimates $\{\hat{X}_i(t)\}_{t\geq 0}$ to minimize the mean-squared error. In particular, it can be shown that the optimal estimation strategy is as follows: at the sampling time of sensor i, $\hat{X}_i(t) = X_i(t)$; at other times

$$d\hat{X}_i(t) = a_i \hat{X}_i(t) dt.$$

Define the error process as $E_i(t) = X_i(t) - \hat{X}_i(t)$. Then, for all sampling times $t = kT_i$, $k \in \mathbb{Z}_{\geq 0}$, $E_i(t) = 0$, and for $t \in (kT_i, (k+1)T_i)$, it follows the dynamics

$$dE_i(t) = a_i E_i(t) dt + \sigma_i dW_i(t).$$

Since $E_i(t)$ is a periodic process, the average mean-squared error (MSE) $M_i(T_i)$ for sensor i when the sampling period is T_i is given by

$$\begin{aligned} M_i(T_i) &= \frac{1}{T_i}\mathbb{E}\left[\int_0^{T_i} (X_i(t) - \hat{X}_i(t))^2 dt\right] \\ &= \frac{1}{T_i}\mathbb{E}\left[\int_0^{T_i} (E_i(t))^2 dt\right] \\ &= \frac{1}{T_i}\int_0^{T_i} \operatorname{Var}(E_i(t)) dt. \end{aligned} \tag{3.6}$$

The total mean-squared error across all sensors is given by

$$\sum_{i=1}^{n} M_i(T_i). \tag{3.7}$$

[1]Delays and packet drops will increase the mean-squared error but not change the nature of the problem in any fundamental way.

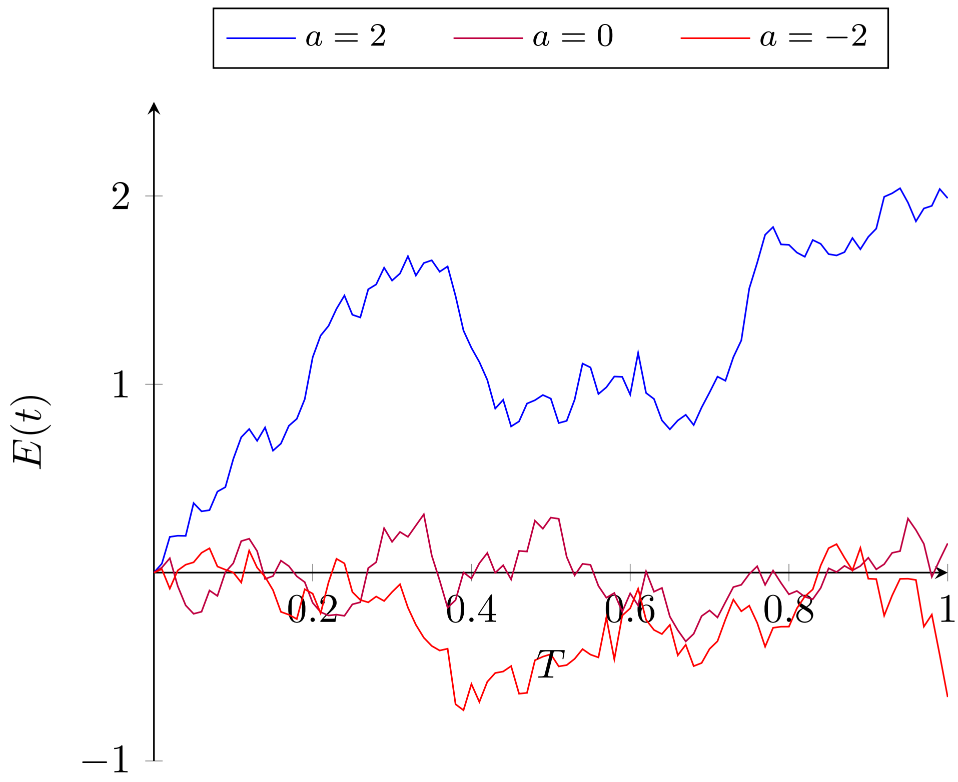


Fig. 3.2: Error process when $\sigma = 1$ for (a) $a = 2$ (unstable) , (b) $a = 0$ and (c) $a = -2$ (stable) .

Example 1 Suppose the noise process at sensor i, $i \in N$, is a Weiner process with variance σ_i^2. Then, the state process is a stationary Gauss-Markov (or a Ornstein-Uhlenbeck) process, which we denote by $\text{GaussMarkov}(a_i, \sigma_i)$. For any $t \in (0, T_i)$, we have that

$$\mathbb{E}[E_i(t)] = 0, \quad \text{and} \quad \text{Var}(E_i(t)) = \frac{\sigma_i^2}{2a_i}(e^{2a_i t} - 1).$$

Substituting this in (3.6), we get that

$$\begin{aligned} M_i(T_i) &= \frac{1}{T_i}\int_0^{T_i} \text{Var}(E_i(t))dt \\ &= \frac{1}{T_i}\int_0^{T_i} \frac{\sigma_i^2(e^{2a_i t} - 1)}{2a_i}dt \\ &= \frac{\sigma_i^2}{2a_i}\left[\frac{e^{2a_i T_i} - 1}{2a_i T_i} - 1\right]. \end{aligned}$$

3.2.2 Problem formulation

We are interested in the following optimization problem.

Problem 1 *Find a rate vector* $(R_1, \ldots, R_n)$ *in the rate region* $\mathcal{R}$ *that minimizes the mean-squared error* (3.7)*. Or formally,*

$$\min_{\mathbf{R}\in\mathbb{R}^n_{\geq 0}} M_i\left(\frac{1}{R_i}\right) \textit{ such that } \sum_{i=1}^{n} R_i \leq C. \tag{3.8}$$

We assume that the model satisfies the following assumptions:

(A1) For all sensors i, $i \in N$, and any sampling period $T_i \in \mathbb{R}_{\geq 0}$, the mean-squared error in (3.6) is strictly increasing and convex in T_i.

(A2) $M_i(T_i)$ is twice differentiable and the curvature $M_i''(T_i)$ on $\mathcal{R}$ is uniformly bounded away from zero, i.e., there exists a positive constant $\bar{\gamma}$ such that $M_i''(T_i) \geq \bar{\gamma}$ for all $T_i \in \mathbb{R}_{\geq 0}$ and $i \in N$.

Since $R_i = 1/T_i$, we can equivalently write (A1) and (A2) as follows:

(A1') For all sensors i, $i \in N$, and any sampling rate $R_i \in \mathbb{R}_{\geq 0}$, the mean-squared error in (3.6) is strictly decreasing and convex in R_i.

(A2') $M_i(1/R_i)$ is twice differentiable and the curvature $M_i''(1/R_i)$ on $\mathcal{R}$ is uniformly bounded away from zero, i.e., there exists a positive constant $\bar{\gamma}$ such that $M_i''(1/R_i) \geq \bar{\gamma}$ for all $R_i \in \mathbb{R}_{\geq 0}$ and $i \in N$.

These assumptions are mild and will be satisfied in most sensor networks. In particular, if $\mathrm{Var}(E_i(t)) > ct^{\alpha}$ where c and α are positive constants, then $M_i(T_i)$ is strictly increasing in T_i; if $\mathrm{Var}(E_i(t))$ is differentiable, then, $M_i(T_i)$ is twice differentiable.

The above assumptions are satisfied in Example 1 when $a_i > 0$. To see that note that,

$$M_i^{'}(T_i) = \frac{\sigma_i^2}{4a_i^2 T_i^2}\left[e^{2a_iT_i}(2a_iT_i - 1) + 1\right]$$

and

$$M_i^{''}(T_i) = \frac{\sigma_i^2 e^{2a_iT_i} - 2M_i^{'}(T_i)}{T_i}$$

If can be shown that for all $x \in \mathbb{R}_{>0}$, $e^x(x-1)+1 > 0$. Substituting $x = 2a_iT_i$ with $a_i > 0$, we get that $M_i^{'}(T_i) > 0$. Also we have,

$$\begin{aligned}
M_i^{''}(T_i) &= \frac{\sigma_i^2 e^{2a_iT_i}}{T_i} - \frac{2M_i^{'}}{T_i} \\
&= \frac{2a_i\sigma_i^2 e^x}{x} - \frac{4a_i\sigma_i^2\Big(e^x(x-1)+1\Big)}{x^3} \\
&= 4a_i\sigma_i^2\left(\frac{e^x}{2x} - \frac{e^x(x-1)+1}{x^3}\right) \\
&= 4a_i\sigma_i^2 f(x)
\end{aligned}$$

- $f(x)$ is increasing in x for $x > 0$
- $\lim_{x\to 0} f(x) = \frac{1}{6}$

Therefore, $M_i^{''}(T_i) > \frac{2a_i\sigma_i^2}{3}$, thus satisfying (A1) and (A2)

Under assumption (A1'), the objective function in Problem 1 is strictly convex in $(R_1, \ldots, R_n)$ and the constraint is convex in $(R_1, \ldots, R_n)$. If the coupling constraint is relaxed, the optimization problem decomposes into independent sub-problems. Therefore, Problem 1 can be solved efficiently using dual decomposition [22].

The basic idea of dual decomposition is as follows. Instead of the constrained problem (3.8) (which is the primal problem), we consider its Lagrangian dual:

$$D(\lambda) = \min_{\mathbf{R}\in\mathbb{R}^n_{\geq 0}} L(\mathbf{R}, \lambda)$$

where

$$\begin{aligned}
L(\mathbf{R}, \lambda) &= \sum_{i=1}^{n} M_i\left(\frac{1}{R_i}\right) - \lambda\Big(C - \sum_{i=1}^{n} R_i\Big) \\
&= \sum_{i=1}^{n}\Big(M_i\left(\frac{1}{R_i}\right) + \lambda R_i\Big) - \lambda C.
\end{aligned} \tag{3.9}$$

The key to a distributed solution lies in decomposing the dual objective (3.9) into two levels of optimization problems. In the context of the above model, at the lower level, it is assumed that the Lagrange multiplier λ is known and each sensor i, $i \in N$, chooses a sampling rate as a solution to the following optimization problem:

$$R_i^*(\lambda) = \arg\min_{R_i\in\mathbb{R}_{\geq 0}} M_i\left(\frac{1}{R_i}\right) + \lambda R_i. \tag{3.10}$$

At the higher level, the network assumes that the sensors use rates according to the solution of (3.10) and chooses the Lagrange multiplier by solving the dual problem

$$\min_{\lambda \in \mathbb{R}_{\geq 0}} L(\mathbf{R}^*(\lambda), \lambda), \tag{3.11}$$

where $\mathbf{R}^*(\lambda) = (R_1^*(\lambda), \ldots, R_n^*(\lambda))$.

In the sequel, we consider two different algorithms for simultaneously solving (3.10) and (3.11); we call these *synchronous* and *asynchronous* algorithms. Both algorithms are iterative algorithms where the remote estimator updates the value of the Lagrange multiplier (or the shadow price) λ and sensor i updates the value of the rate R_i. In both algorithms it is assumed that the remote estimator can broadcast the Lagrange multiplier λ to all sensors. The synchronous and asynchronous algorithms differ in how the update of the rates R_i is communicated back to the remote estimator.

The synchronous algorithm is performed as part of the initial handshaking protocol during which the sensors can directly communicate their updated rates to the remote estimator. Thus, at each iteration, the remote estimator synchronously updates the rates of all sensors. The algorithm is described in detail in Section 3.3.1.

The synchronous algorithm requires a control channel and additional bandwidth for the signalling overhead. On the other hand, the asynchronous algorithm is an on line algorithm where no additional bandwidth is required for signalling. The sensors do not explicitly communicate the updated rates to the remote estimator. Rather they simply transmit data at the updated rates and the remote estimator infers the rate through the inter-arrival time between successive transmissions. The algorithm is described in detail in Section 3.3.2.

3.3 The Two Dual Decomposition Algorithms

3.3.1 The Synchronous Dual Decomposition Algorithm

The synchronous dual decomposition algorithm may be viewed as an iterative gradient descent algorithm used by the remote estimator to solve (3.11). The remote estimator starts with an initial guess λ_0 of the optimal Lagrange multiplier. At each iteration k,

Algorithm 3.2 Synchronous update

```
Input
  $\mathcal{S}$ : set of sensors
  $C$ : network capacity
  $K$ : number of iterations
  $\alpha$ : gradient descent step size
  $\lambda_0$ : Lagrange multiplier

  procedure SYNC($\mathcal{S}$, $C$, $K$, $\alpha$, $\lambda_0$)
      for $k = 0$ upto $K$ do
          for each $i$ in $\mathcal{S}$ do
              solve $R_{i,k} : M_i'(1/R_{i,k}) = R_{i,k}^2 \lambda_k$
          end for
          $\lambda_{k+1} = \left[\lambda_k - \alpha(C - \sum_{i=1}^N R_{i,k})\right]^+$
      end for
      return $(R_{1,K}, \ldots, R_{n,K})$
  end procedure
```

the following steps are repeated (See Algorithm 3.2 for a formal description):

- User i, $i \in N$, chooses a rate $R_{i,k} = R_{i,k}^*(\lambda_k)$ by solving the optimization problem (3.10), which is a convex optimization problem. One possible solution is to identify a rate $R_{i,k}$ that satisfies

$$M_i'(1/R_{i,k}) - \lambda_k R_{i,k}^2 = 0 \tag{3.12}$$

  where M_i' denotes the derivative of M_i. User i then communicates $R_{i,k}$ to the remote estimator.

- Upon receiving the updated rate vectors $(R_{1,k}, \ldots, R_{n,k})$, the remote estimator updates the Lagrange multiplier in the direction of the gradient[2] as follows:

$$\lambda_{k+1} = \left[\lambda_k - \alpha_k\Big(C - \sum_{i\in N} R_i^*(\lambda_k)\Big)\right]^+, \tag{3.13}$$

where $\alpha_k > 0$ is an appropriately chosen learning rate and $[x]^+ = \max\{x, 0\}$.

Theorem 1 *Under assumptions* (A1') *and* (A2') *there exists a unique solution* $\mathbf{R}^* = (R_1^*, \ldots, R_n^*)$ *of Problem 1. Moreover, for any initial guess* $\lambda_0 > 0$ *and sufficiently small step size* α*, the rates* $\mathbf{R}_k = (R_{1,k}, \ldots, R_{n,k})$ *chosen in the synchronous dual decomposition algorithm converge to the optimal rates* $\mathbf{R}^*$ *as* $k \to \infty$*.*

Proof : Conditions (A1') and (A2') are equivalent to the conditions (C1) and (C2) of Theorem 1 in [20]. The proof follows from [20, Appendix I].

3.3.2 The Asynchronous Dual Decomposition Algorithm

In the asynchronous dual decomposition algorithm, each user $i, i \in N$, keeps track of the time T_i^S at which it will take the next sample. The remote estimator keeps track of the times $T_{i,k}^R$ when each sensor last transmitted and its estimate of their transmission rates $\hat{R}_i$. The remote estimator initializes $T_{i,-1}^R = 0$ and chooses an initial guess λ_0 of the Lagrange multiplier and broadcasts it to all sensors. Each sensor $i, i \in N$, then computes $R_{i,0}$ by solving (3.12) and initializes $T_i^S = 1/R_{i,0}$. Then the following steps are performed at every iteration k (See Algorithm 3.3 for formal description):

- Let j_k denote the user with the lowest sampling time T_i^S. At time $T_{j_k}^S$, user j_k takes a sample and sends its measurements to the remote estimator over the network.

- Upon receiving the message from user j_k, the remote estimator sets

$$T_{i,k}^R = \begin{cases} T_{j_k}^S & \text{if } i = j_k \\ T_{i,k-1}^R & \text{otherwise} \end{cases} \tag{3.14}$$

[2]Note that for a given choice of rates $\mathbf{R}$, the derivative of $L(\mathbf{R}, \lambda)$ with respect to λ is given by $C - \sum_{i=1}^n R_i$.

and

$$\hat{R}_{i,k} = \begin{cases} \dfrac{1}{T^R_{i,k} - T^R_{i,k-1}} & \text{if } i = j_k \\ \hat{R}_{i,k-1} & \text{otherwise} \end{cases} \tag{3.15}$$

- The remote estimator then chooses λ_{k+1} by updating the Lagrange multiplier by taking a step in the direction of the gradient as follows:

$$\lambda_{k+1} = \left[\lambda_k - \alpha_k\Big(C - \sum_{i\in N} \hat{R}_{i,k}(\lambda_k)\Big)\right]^+, \tag{3.16}$$

  where $\alpha_k > 0$ is an appropriately chosen learning rate and $[x]^+ = \max\{x, 0\}$.

- The Lagrange multiplier λ_{k+1} is broadcast and user j_k updates the sampling rate R_{j_k} according to (3.10) or (3.12) and sets $T^S_{j_k} = T^S_{j_k} + \frac{1}{R_{j_k}}$.

Let $\mathcal{T}^R = \{T^R_{j_k,k}\}_{k\geq 0}$ denote the set of time instances at which the remote estimator updates the Lagrange multiplier based on the current estimate of the sensor rates. Also, let $\mathcal{T}^S_i$, $i \in N$, denote the set of time instances at which the sensor i updates its rate based on the Lagrange multiplier broadcast to it. It is assumed that the following is satisfied:

(A3) The time between consecutive updates in $\mathcal{T}^R$ (i.e., at the remote estimator) and $\mathcal{T}^S_i$, $i \in N$, (i.e., at every sensor) are bounded.

Note that (A3) is satisfied if for all $\lambda \in \mathbb{R}_{>0}$, the optimal rate $R^*_i(\lambda)$ obtained in (5) is bounded. This is the case in Example 1 if $a_i > 0$.

Theorem 2 *Under assumptions* (A1'), (A2') *and* (A3)*, for any initial guess* $\lambda_0 > 0$ *and sufficiently small step size* α_k*, the rates* $\mathbf{R}_k = (R_{1,k}, \dots, R_{n,k})$ *chosen in the asynchronous dual decomposition algorithm converges to the unique solution* $\mathbf{R}^*$ *of Problem 1. Moreover, if the synchronous and asynchronous algorithms use the same learning rates* $\{\alpha_k\}_{k\geq 0}$*, then the corresponding Lagrange multiplies converge to the same value.*

Proof : Assumption (A3) being equivalent to (C3) in [20], the proof follows from [20, Theorem 2].

Algorithm 3.3 Asynchronous allocation of sampling rates

```
Input
  $\mathcal{S}$ : set of sensors
  $C$ : network capacity
  $\alpha$ : gradient descent step size
  $\lambda_0$ : Lagrange multiplier

  for each $i$ in $\mathcal{S}$ do
      solve $R_i$ : $M_i'(1/R_i) - R_i^2\lambda_0 = 0$
      initialize $T_i^S = 1/R_i$ and $T_{i,-1}^R = 0$.
  end for

  procedure ASYNC-SENSOR-$i$
      upon event $\langle$Current time $t = T_i^S\rangle$ do
          sample the state of the process and transmit
          observe updated $\lambda$
          solve $R_i$ : $M_i'(1/R_i) - R_i^2\lambda_0 = 0$
          set $T_i^S := T_i^S + (1/R_i)$
      end event
  end procedure

  procedure ASYNC-ESTIMATOR($C$)
      upon event $\langle$Packet received for sensor $j_k\rangle$ do
          update $T_{i,k}^R$ as given in (3.14)
          update $\hat{R}_{i,k}$ as given in (3.15)
          $\lambda_{k+1} = \left[\lambda_k - \alpha(C - \sum_{i=1}^{N} \hat{R}_{i,k})\right]^+$
          $k = k + 1$
      end event
  end procedure
```

Remark 1 The dual decomposition algorithm does not ensure that the dual iterates are feasible (i.e., at the intermediate steps of the algorithm, it is not guaranteed that $\sum_{i \in N} R_{i,k} \leq C$). To keep the iterates feasible, one could start with a large value of λ_0, which will ensure that the users pick small values of the initial rates $\{R_{i,0}\}_{i \in N}$.

3.4 Chapter Summary

In this chapter, a brief summary of the network utility maximization was provided. The remote estimation problem was formulated along the lines of network utility maximization under dual decomposition. Details of two decentralized dual decomposition algorithms, namely a synchronous and an asynchronous algorithm were discussed. We proved that they converge under certain assumptions and conditions. In order to validate our claims and results, we conduct a simulation study, the details of which will be discussed in the next chapter.

Chapter 4

Simulation Results and Discussions

In this chapter, we illustrate the convergence of the synchronous and asynchronous dual decomposition algorithms as proved in Theorems 1 and 2. We present simulation results demonstrating robustness of the asynchronous algorithm to slowly changing network conditions. The effect of packet drops on convergence is discussed. We then look into the behaviour of asynchronous algorithm in large networks. On the basis of the results so obtained, we can characterize the performance of the dual decomposition technique studied so far.

4.1 Convergence of the Dual Decomposition Algorithms

To illustrate how the algorithms work, consider a system with a total capacity of $C = 1$ and two sensors, GaussMarkov$(1, 1)$ and GaussMarkov$(1, 2)$. For the synchronous algorithm, suppose the remote estimator starts with an initial guess $\lambda_0 = 10$ and both sensors use a constant learning rate[1] of $\alpha_k = 10$. Then, the rates converge to $(R_1, R_2) = (0.4355, 0.5645)$ and $\lambda = 88.9136$. After 200 iterations, the value of λ is 88.357, which is within 0.625% of the optimal value. The values of λ,

[1]In practice, convergence speeds up considerably if the learning rate is adapted according the gradient (e.g., using ADAM or ADAGrad [27]). However, in this example, we choose a constant learning rate to simplify exposition.

Table 4.1: The values of the Lagrange multiplier λ and the rates R_1 and R_2 for a few iterations of the synchronous the example of Section 4.1.

Iteration	λ	R_1	R_2
1	10.000	0.6711	0.9519
2	16.230	0.6029	0.8360
3	20.619	0.5733	0.7865
4	24.217	0.5547	0.7558
5	27.322	0.5414	0.7339
$\vdots$	$\vdots$	$\vdots$	$\vdots$
196	88.307	0.4360	0.5653
197	88.320	0.4360	0.5653
198	88.333	0.4360	0.5653
199	88.345	0.4360	0.5652
200	88.357	0.4360	0.5652

R_1, and R_2 for a few iterations at the beginning and the end are shown in Table 4.1.

For the asynchronous algorithm, we again assume that the remote estimator starts with an initial guess $\lambda_0 = 10$ and both sensors use a constant learning rate of $\alpha_k = 10$. After 200 iterations, the value of λ is 88.399, which is within 0.579% of the optimal value. The values of λ, R_1, and R_2 for a few iterations at the beginning and the end are shown in Table 4.2. It should be noted that the sensors do not always update alternatively; sensor 2 was updated for two consecutive iterations at iterations 196 and 197.

For comparison, we plot the value of the Lagrange multiplier λ and rates R_1 and R_2 vs iteration for both the synchronous and the asynchronous algorithms in Fig. 4.1. As can been seen from the figure, at each iteration, the Lagrange multiplier and the rates as determined by the synchronous and the asynchronous algorithms are fairly close. The key difference is that the synchronous algorithm is implemented as part of the initial handshaking protocol while the asynchronous algorithm is on line where the sensors adapt their transmission rates while transmitting data. The 200

Table 4.2: The values of the Lagrange multiplier λ and the rates R_1 and R_2 for a few iterations of the asynchronous the example of Section 4.1.

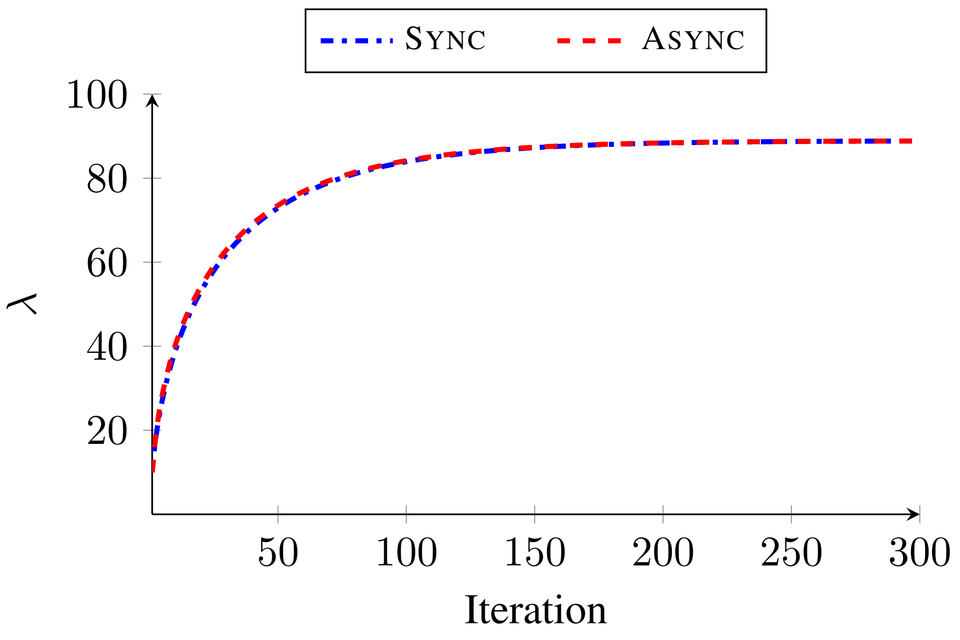
Iteration	λ	Updated Sensor	R_1	R_2	Time (in sec)
1	10.000	2	0.6757	0.9611	1.04
2	16.368	1	0.6757	0.8342	1.48
3	21.467	2	0.5685	0.8342	2.24
4	25.494	1	0.5685	0.7463	3.24
5	28.643	2	0.5364	0.7463	3.58
⋮	⋮	⋮	⋮	⋮	⋮
196	88.352	2	0.4360	0.5652	189.24
197	88.364	2	0.4360	0.5652	191.01
198	88.376	1	0.4360	0.5652	191.30
199	88.388	2	0.4359	0.5652	192.78
200	88.399	1	0.4359	0.5652	193.60

iterations of asynchronous algorithm takes about 194 sec.

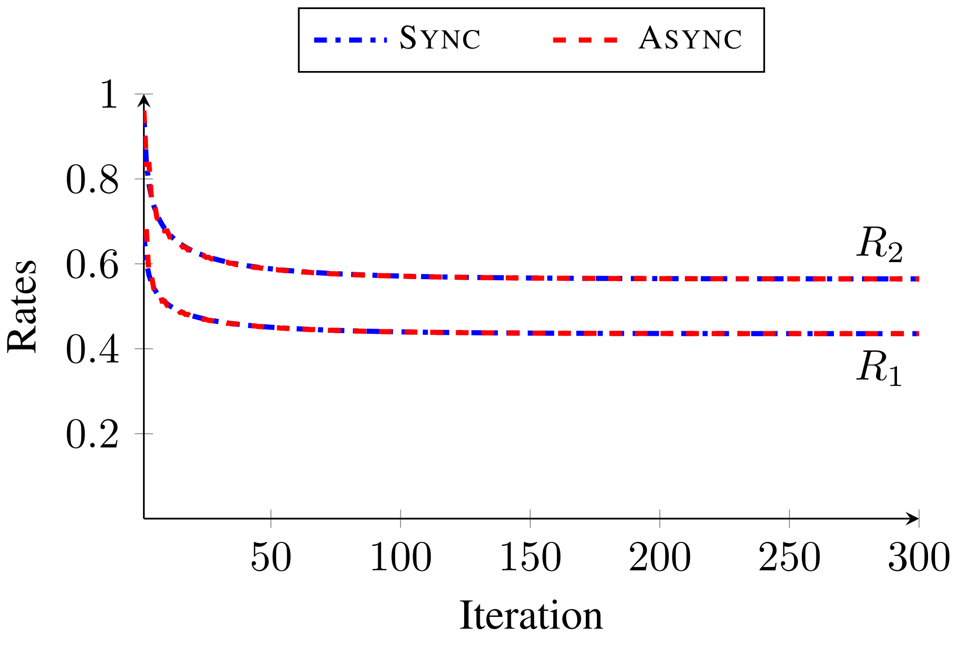
In the analysis shown in the previous section, the system is assumed to be static. However, the dual decomposition algorithm is robust to slow changes in the network conditions. We consider the scenarios below that illustrate the impact of the change in network conditions: (i) a new sensor comes aboard but the capacity remains constant; (ii) an existing sensor leaves but the capacity remains constant; (iii) the capacity changes but the number of sensors remain constant. In all three scenarios, we assume that the system consists of sensors that observe Gauss Markov processes. The details of each experiment and the result are shown in Figs. 4.2–4.4.

4.1.1 Robustness to Changing Network Conditions

In the first scenario, shown in Fig. 4.2, we start with a system consisting of two sensors: GaussMarkov$(1, 1)$ and GaussMarkov$(1, 2)$ in a system with capacity $C = 1$. For both the synchronous and the asynchronous algorithms, we start with the initial



(a)



(b)

Fig. 4.1: Plot of (a) Lagrange multiplier λ and (b) rates R_1 and R_2 versus iteration for the illustrative example of Sec. 4.1.

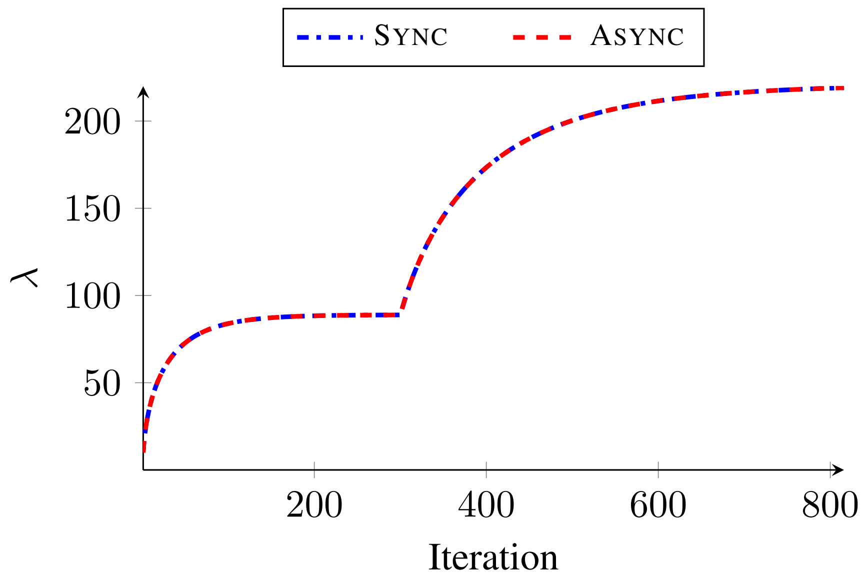
$\lambda_0 = 10$ and use a constant learning rate of $\alpha_k = 10$. After 300 iterations, the value of λ is 88.8674 and the rates allocated to the two sensors are 0.4355 and 0.5645, respectively. We assume that at that time a new GaussMarkov$(0.4, 0.4)$ sensor comes aboard. Since there are now three sensors competing for the same resource, the Lagrange multiplier increases and is around 219 after a total of 800 iterations. The rates of the three sensors are 0.3764, 0.4743, and 0.1505, respectively.

Similarly, in the second scenario, shown in Fig. 4.3, we start with three sensors: GaussMarkov$(1, 1)$, GaussMarkov$(1, 2)$ and GaussMarkov$(0.3, 0.3)$ in a system with capacity $C = 1$. With $\lambda_0 = 10$ and $\alpha_k = 10$, the value of λ after 450 iterations is around 175 for both the synchronous and asynchronous algorithms. The allocated rates are 0.3897, 0.4943 and 0.1169 respectively. We assume that at that time GaussMarkov$(1, 2)$ leaves the system. With only two sensors now competing for the same resource, the Lagrange multiplier decreases and settles at 5.6345 after 800 iterations. The rates of the other two sensors are 0.7692 and 0.2308.

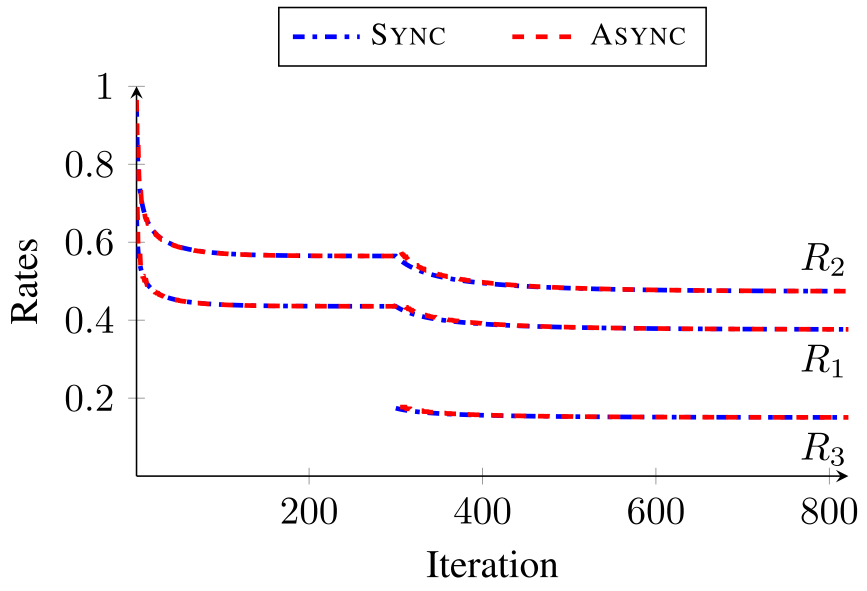
In the third scenario, shown in Fig. 4.4, we have two sensors: GaussMarkov$(1, 1)$ and GaussMarkov$(1.5, 1.5)$ in a system with capacity $C = 1$. With $\lambda_0 = 10$ and $\alpha_k = 10$, for both the synchronous and asynchronous algorithms, the value of λ is around 148 and the sensor rates are 0.4 and 0.6 after 450 iterations, at which point we assume that the system capacity becomes $C = 1.5$. With an increased capacity for the same set of two sensors, the Lagrange multiplier decreases and settles at 16.6 after 800 iterations. The allocated rates are 0.6 and 0.9, respectively.

4.1.2 Robustness of Asynchronous Algorithm to Packet Drops

For the asynchronous algorithm we assumed an ideal communication channel. The algorithm is robust to delays and packet drops introduced by the channel as long as assumption (A3) continues to hold. To illustrate this point, we reconsider the example of Sec. 4.1 but assume that packets are dropped with probability p. The plot of Lagrange multiplier versus number of iterations for different values of packet drop probability p is shown in Fig. 4.5. As can be seen from the figure, there is very little impact of packet drops on the convergence of the algorithm.

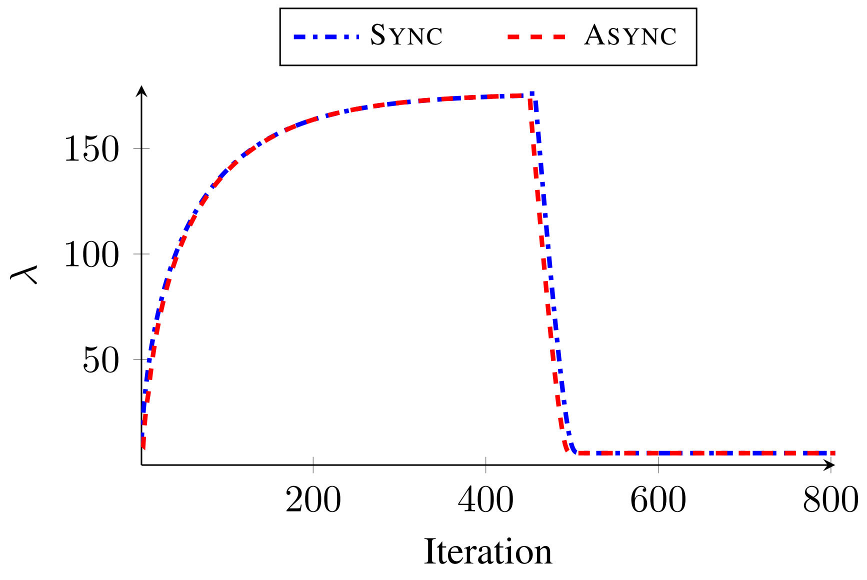


(a)

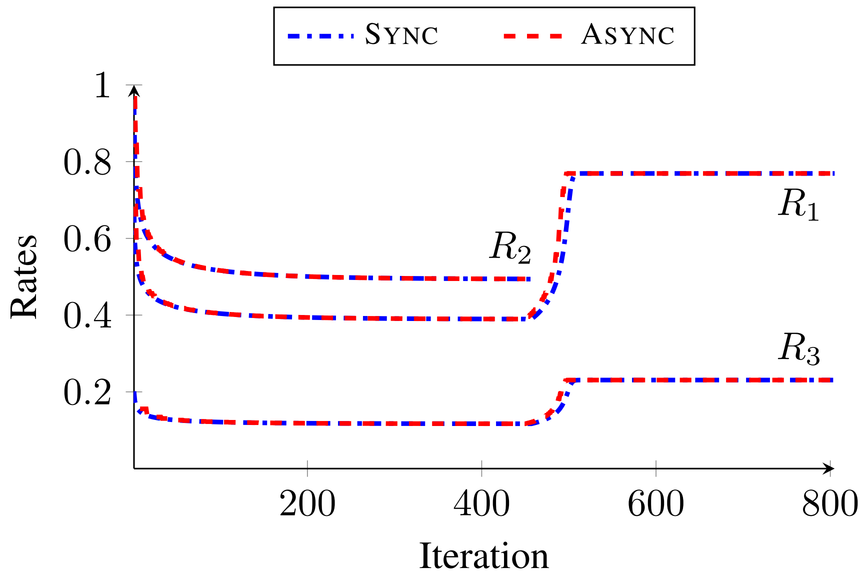


(b)

Fig. 4.2: Plot of (a) Lagrange multiplier λ and (b) rates versus iteration for a case illustrating a new sensor coming aboard. We start with a system consisting of two sensors: GaussMarkov$(1, 1)$ and GaussMarkov$(1, 2)$ with a total capacity $C = 1$. At iteration 300, a new sensor given by GaussMarkov$(0.4, 0.4)$ comes onboard.

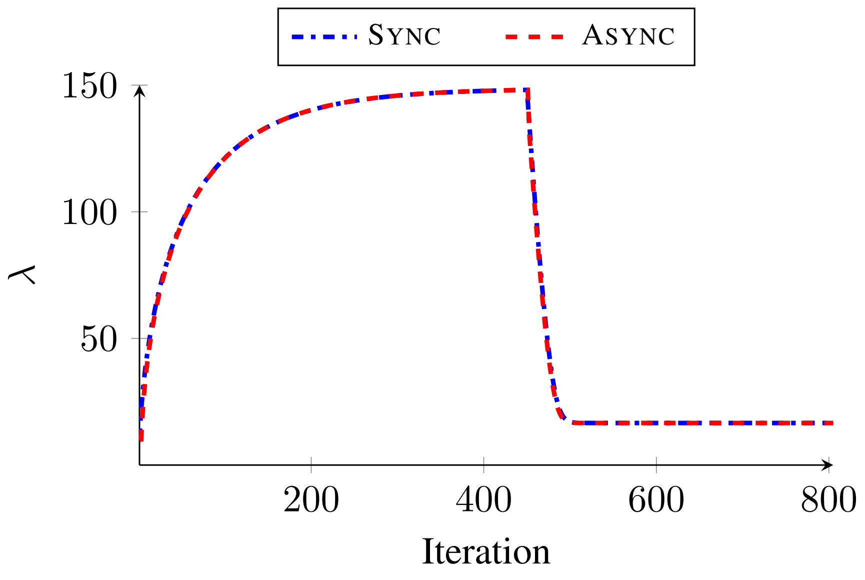


(a)

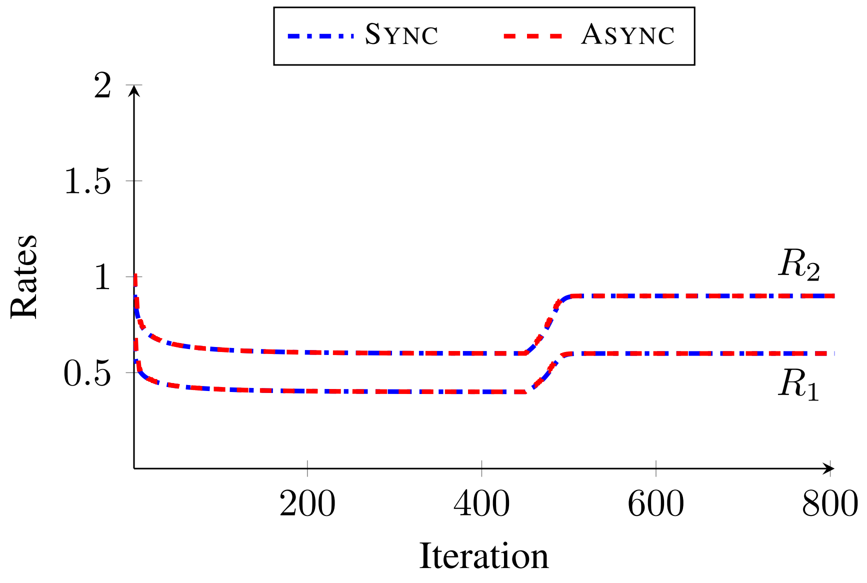


(b)

Fig. 4.3: Plot of (a) Lagrange multiplier λ and (b) rates versus iteration for a case illustrating a sensor leaving. We start with a system consisting of three sensors: GaussMarkov$(1, 1)$, GaussMarkov$(1, 2)$ and GaussMarkov$(0.3, 0.3)$ with a total capacity of $C = 1$. At iteration 450, GaussMarkov$(1, 2)$ leaves.



(a)



(b)

Fig. 4.4: Plot of (a) Lagrange multiplier λ and (b) rates versus iteration for a case where the capacity of the network changes. We start with a system consisting of two sensors: GaussMarkov$(1, 1)$ and GaussMarkov$(1.5, 1.5)$ with a total capacity $C = 1$. At iteration 450, the capacity changes to $C = 1.5$.

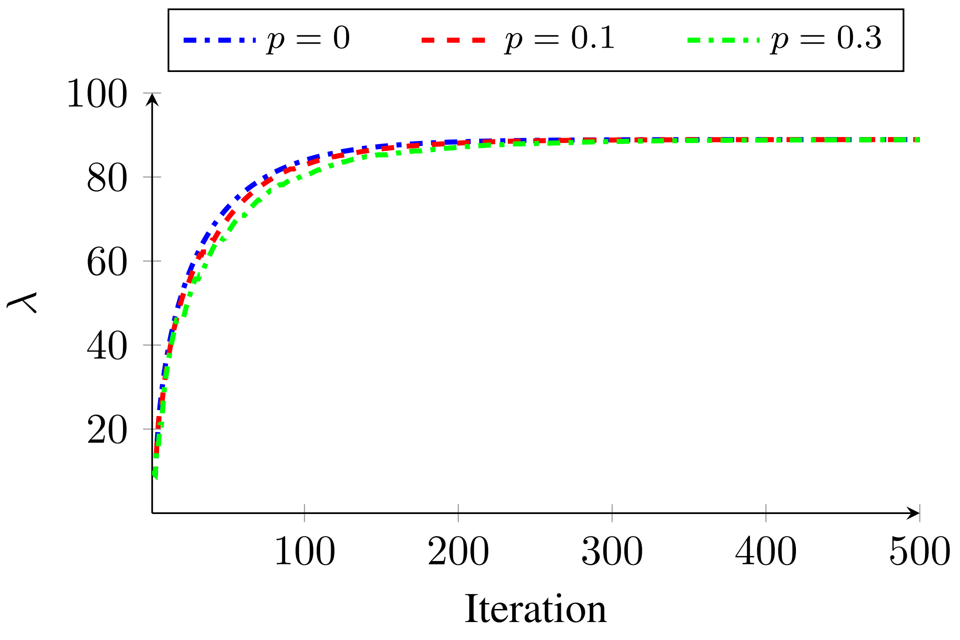


Fig. 4.5: Plot of Lagrange multiplier λ versus iteration for the asynchronous algorithm under packet drop for the illustrative example of Sec. 4.1.

4.2 Asynchronous Algorithm in a Large Dynamic System

We consider an experimental setup where the number of sensors $N(t)$ changes according to a stochastic process. We assume that new sensors arive according to a Poisson process with rate ρ and stays in the system for an exponentially distributed amount of time with rate ρ, after which the sensor leaves the system. Each new sensor is GaussMarkov(a_i, σ_i) where a_i and σ_i are chosen randomly. We assume that the remote estimator broadcasts the value of the Lagrange multiplier λ at all times. When a new sensor arrives, its initial sampling rate is determined based on the current value of λ. The remote estimator continues to adapt λ according to Algorithm 3.3, without being explicitly aware that a new sensor has arrived. Similarly, the remote estimator is not explicitly aware when a sensor leaves the system.

We consider a scenario of 450 seconds where we start with $N(0) = 25$ sensors and sensors arrive and leave at a rate of $\rho = 2$ per minute. Each new sensor is

GaussMarkov(a_i, σ_i), where $a_i \sim$ Unif$[0.1, 2]$ and $\sigma_i = 1$. At $T = 0$, the system capacity is 25; at $T = 200$, the capacity changes to $C = 20$; and at $T = 400$, it changes to $C = 30$. We run the asynchronous algorithm with a constant learning rate of $\alpha_k = 0.01$ throughout. The plot of $N(t)$ and λ versus time as well as $\sum_{i \in N} R_i$ versus time are shown in Fig. 4.6. These plots illustrate the robustness of the asynchronous algorithm to changing network conditions.

In Fig. 4.7(a), we zoom into Fig. 4.6(b) at $T = 180$ when the system has 25 sensors, $C = 25$, and $\lambda = 2.58$. At this time, a new sensor comes aboard sees the current value of λ and chooses a transmission rate using (3.10). When the new sensor transmits, the sum rate exceeds the channel capacity[2] The remote estimator adjusts Lagrange multiplier λ according to (3.11). Since there is one more sensor competing for the same resource, the Lagrange multiplier increases and converges to 3.09 at $T = 189$.

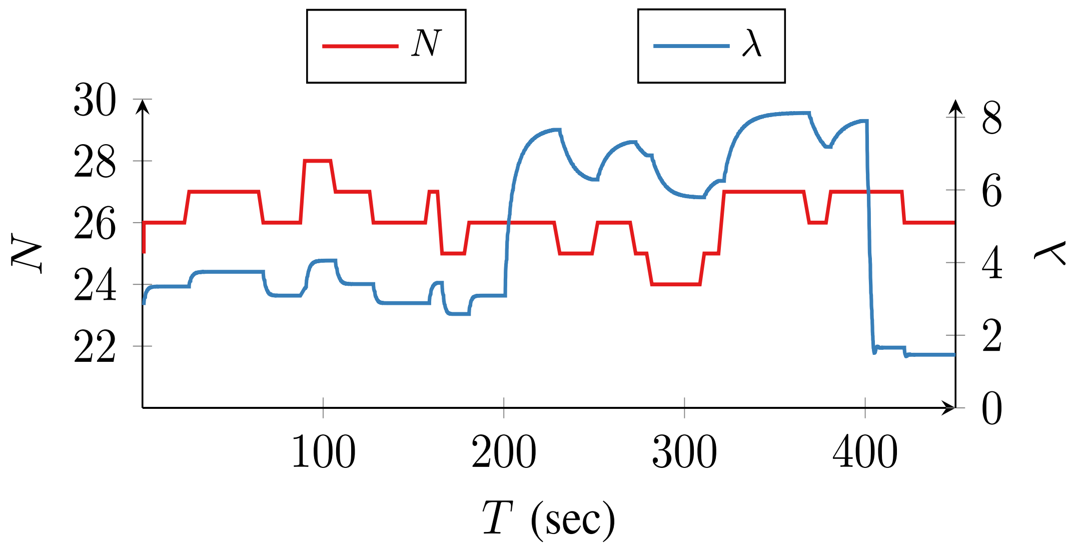
In Fig. 4.7(b), we zoom into Fig. 4.6(b) at $T = 127$ when the system has 27 sensors, $C = 25$, and $\lambda = 3.41$. At this time, one of the sensors leaves and the sum rate falls below the network capacity. The remote estimator adjusts the Lagrange multiplier λ according to (3.11). Since there is one less sensor competing for the same resource, the Lagrange multiplier decreases and converges to 2.88 at $T = 135$.

In Fig. 4.7(c), we zoom into Fig. 4.6(b) at $T = 200$ when the system has 26 sensors, $C = 25$, and $\lambda = 3.09$. At this time, the system capacity reduces to $C = 20$. The remote estimator adjusts the Lagrange multiplier using (3.11). Since there are the same number of sensors competing for less resources, the Lagrange multiplier increases and converges to 7.65 at $T = 227$.

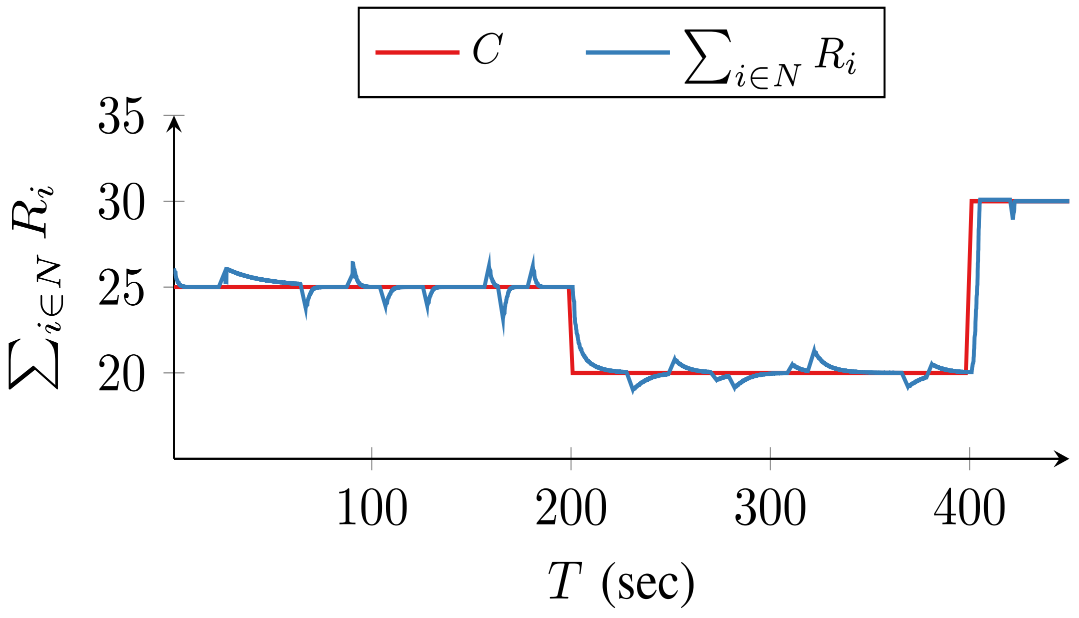
To observe how individual sensor rates vary with changes in network conditions, we pick two sensors, GaussMarkov$(1.3, 1)$ and GaussMarkov$(0.4, 1)$ that stay active throughout the experiment. The rate allocation by the asynchronous algorithm for these sensors is shown in Fig. 4.8(a) and the empirical MSE is shown in Fig. 4.8(b).

We compare the optimal rate allocation described in the asynchronous dual

[2]In practice, the sum rate exceeding the channel capacity will result in delay or packet drops but such effects are not taken into account in our model.

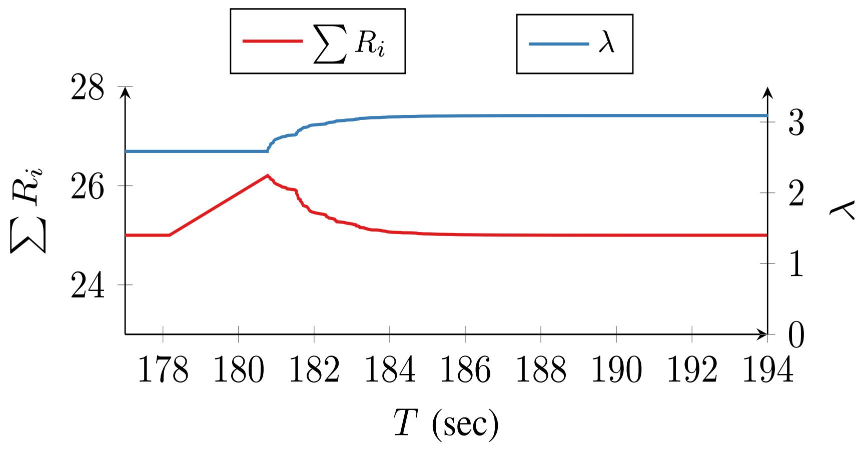


(a)

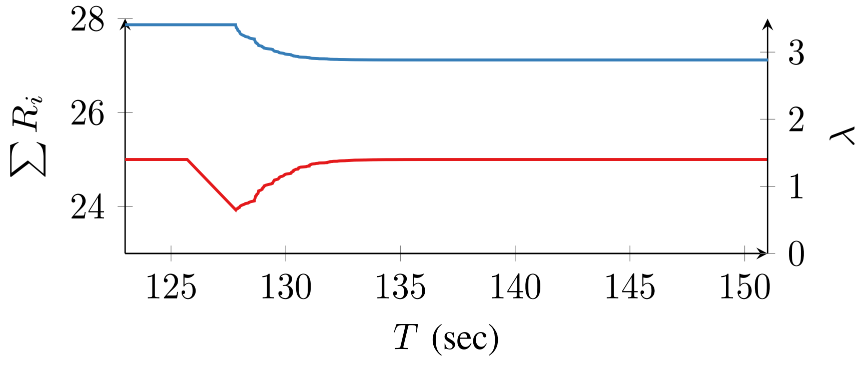


(b)

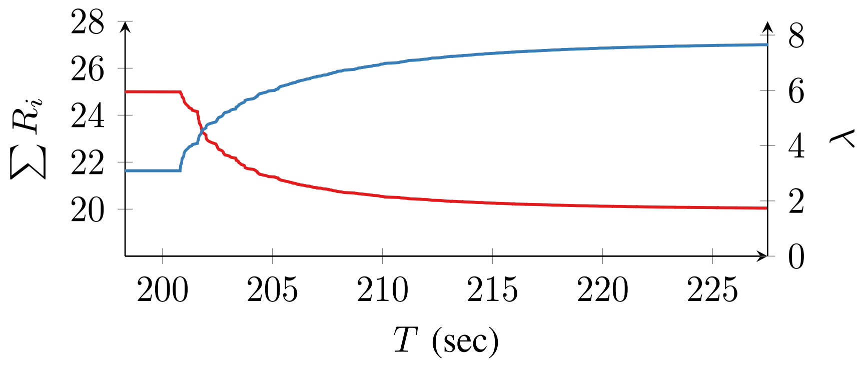
Fig. 4.6: Plot of (a) number of sensors N and λ, and (b) sum rate $\sum R_i, i \in N$ versus time for the asynchronous algorithm for the system described in Sec. 4.2.



(a)

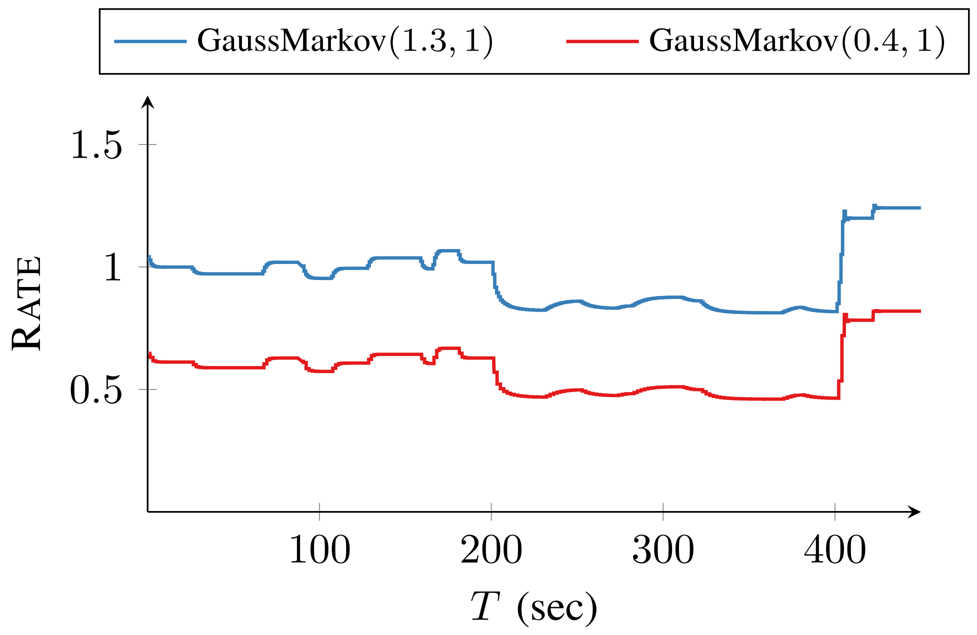


(b)

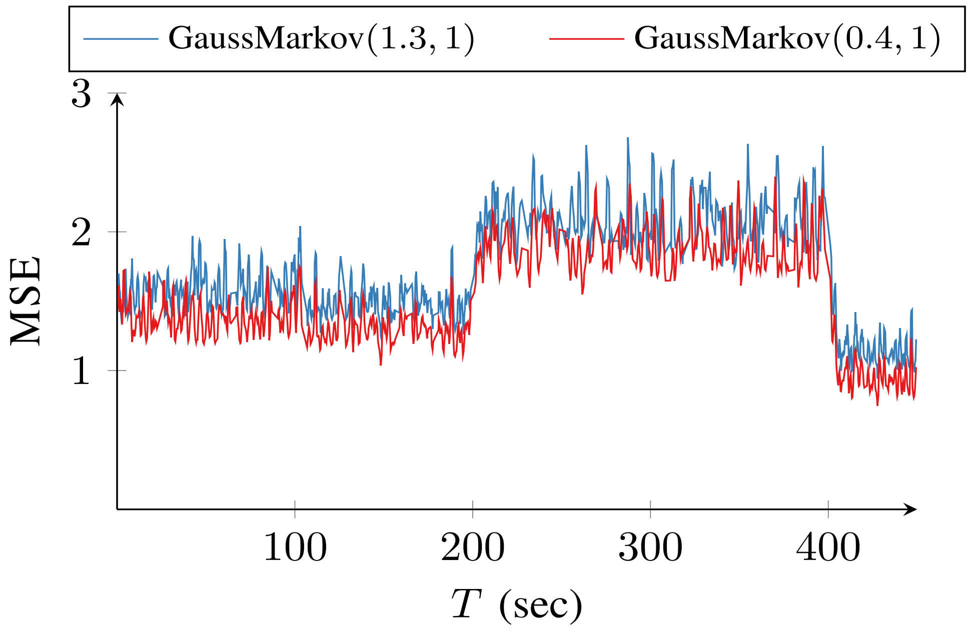


(c)

Fig. 4.7: Plot of sum rate $\sum R_i$ and λ versus time for the asynchronous algorithm, illustrating (a) sensor coming aboard, (b) sensor leaving and (c) capacity change for the system described in Sec. 4.2.



(a)



(b)

Fig. 4.8: Plot of (a) sampling rate and (b) empirical MSE versus time for GaussMarkov$(1.3, 1)$ and GaussMarkov$(0.4, 1)$ for the system described in Sec. 4.2.

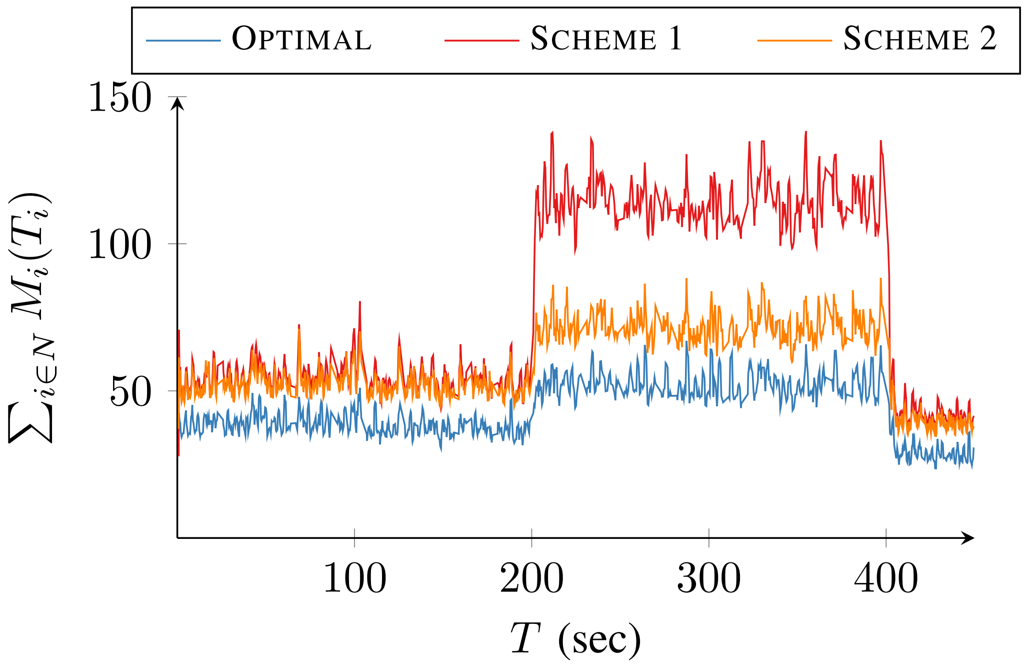


Fig. 4.9: Plot of aggregate empirical MSE illustrating optimality of the asynchronous algorithm against baseline rate allocation schemes 1 and 2, for the system described in Sec. 4.2.

decomposition algorithm with two baseline equal rate allocation schemes. In Scheme 1, we assume that $N(t) \leq 30$ and allocate a constant sampling rate $R_i = C/30$ to all active sensors $i \in N$; in Scheme 2, we assume that the remote estimator keeps track of $N(t)$ and allocates a rate of $R_i = C/N(t)$ to all active sensors $i \in N$. We plot the aggregate empirical MSE for the system in Fig. 4.9. The performance of the optimal scheme is especially higher when available capacity is low.

4.3 Discussion

In the above simulations, we assumed ideal conditions. In this section, we will discuss few non-ideal situations and their impact on the rate allocation algorithms.

4.3.1 Jitter

In the asynchronous algorithm described in Algorithm. 3.3, the sampling time $T_{j_k}^S$ of user j_k with the lowest sampling time, is updated as

$$T_{j_k}^S := T_{j_k}^S + \frac{1}{R_{j_k}}$$

Here, we assume that the remote estimator receives the measurements from user j_k with perfect timing. In practice, the sampling instances are subject to jitter due to which the update happens as

$$T_{j_k}^S := T_{j_k}^S + \frac{1}{R_{j_k}} + \Delta(t)$$

where $\Delta(t)$ is a random time varying jitter component to the ideal sampling instance. Interestingly, $\Delta(t)$ prevents the remote estimator from having measurements from two or more sensors arrive at the same time which otherwise is possible in an ideal system, especially when the sensors have settled to optimal sampling rates for a long time in a static network. While $\Delta(t)$ may affect the order in which sensors update rates, the asynchronous algorithm still converges as each λ update pushes the gradient to zero. However, the time taken to converge increases.

4.3.2 Channel Utilization

A key factor is the percentage channel utilization. The percentage deviation of the sum rate from the available channel capacity, $\frac{\sum R_i - C}{C} \times 100, i \in N$, is shown in Fig. 4.10. We observe that the channel is near capacity for about 80 % of the time. Due to dual iterates not being feasible, it can be seen that the channel is severely under-utilized when the available capacity increases as the asynchronous algorithm takes a finite time to converge under the new capacity. Similarly, the network experiences congestion when a new sensor comes aboard and when the available capacity decreases.

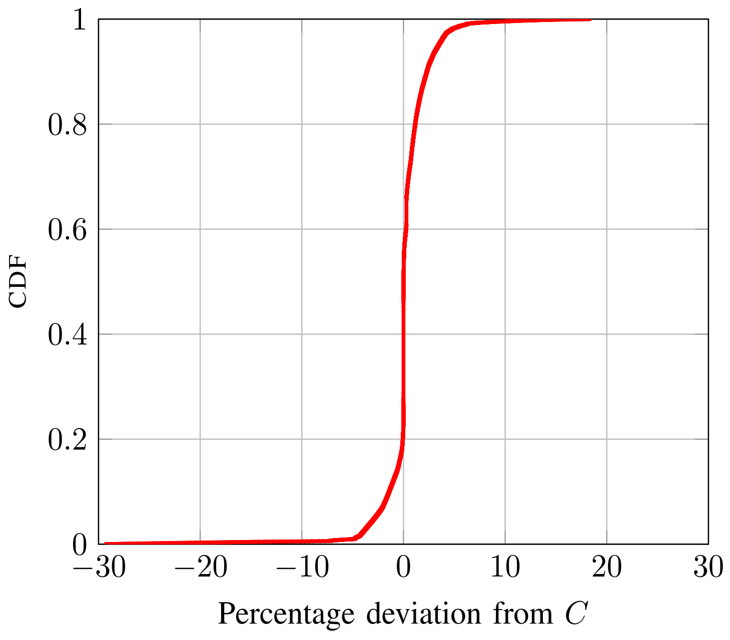


Fig. 4.10: Cumulative distribution of percentage deviation of sum rate from available channel capacity for the system described in Sec. 4.2.

4.4 Chapter Summary

In this chapter, we provided simulation methodology and results demonstrating the convergence and robustness of the dual decomposition algorithms. We also discussed the presence of jitter and looked into channel utilization under the asynchronous algorithm.

Chapter 5

Conclusion and Future Work

This thesis proposed a dual decomposition approach to minimizing mean-squared error in a remote estimation system subject to capacity constraint, by posing the objective as a variant of the network utility maximization problem. The mean-squared error was derived given that each sensor observes a continuous-time stationary Gauss-Markov process. The two dual decomposition algorithms provide a decentralized approach to fulfilling the objective and are shown to converge under mild technical conditions. The robustness of the asynchronous algorithm is also discussed. In this chapter, we provide a chapter-wise summary of the thesis and discuss the main results and provide directives for future work.

5.1 Summary of Work

In Chapter 2, we introduced the fundamentals of primal and dual decomposition and discussed the applicability of decomposition to obtain decentralized control. With the help of an illustrative example, we observed the interaction between the master problem and sub-problems. While dual decomposition decouples easily in the presence of a coupling constraint, there is a caveat in that it is sometimes non-trivial to recover the optimal primal assignment and the duality gap exists even for an optimal λ^*.

In Chapter 3, we first framed a general network utility maximization problem and studied the application of dual decomposition for allocation of optimal rates. The network links and nodes in such a system are treated as distributed processors updating link prices and rates independently to achieve global welfare. A model for the remote estimation system was developed and the objective to minimize the estimation error under finite channel capacity was framed similar to a network utility maximization problem. The dual decomposition approach allowed for two decentralized algorithms, namely a synchronous and an asynchronous rate allocation algorithm, the convergence of which was proved in Theorems 1 and 2.

In Chapter 4, we performed a series of simulations which demonstrated that:

- the two dual decomposition rate allocation algorithms converge
- the λ updates at the remote estimator for both the above algorithms are fairly close
- the asynchronous algorithm is robust to slowly changing network conditions
- the asynchronous algorithm is robust to packet drops
- the asynchronous algorithm is not affected by scale of the network

Some methods discussed in the thesis have limitations and could be investigated further. As previously stated, the dual decomposition algorithm does not ensure that the dual iterates are feasible. This results in the sum rate either overshooting available capacity or the channel being under-utilized. In our work, we assumed the network can accomodate excess rates without getting congested, which is not true in real networks. For small deviations, one could argue the excess rates would cause packet drops to which the asynchronous algorithm is robust. However, higher deviations cause congestion which has to be mitigated by queue management techniques and congestion control. Another possible line of investigation is to study the rate of convergence of the asynchronous algorithm and adapt step-size α_k for faster convergence and greater channel utilization.

While the rate allocation by the two algorithms is unique and optimal, it would also be interesting to look into fairness of the allocation. In a practical network, there might arise a situation where a particular sensor has a low rate and therefore samples less frequently than another sensor with higher rate. In such a case, the former faces resource starvation. One could look into an event-triggered scheduling scheme where the sensor with low rate is allowed to sample after a certain threshold wait time and study the convergence and performance of such a scheme against the asynchronous algorithm. It would also be worthwhile to look into a data driven approach in situations where a closed form expression for the mean-squared error is not available either because the distribution of the noise process is not known or because there is no closed form expression for $\mathrm{Var}(E_i(t))$.

Bibliography

[1] M. Athans, “On the determination of optimal costly measurement strategies for linear stochastic systems,” *Automatica*, vol. 8, no. 4, pp. 397–412, 1972.

[2] V. Gupta, T. H. Chung, B. Hassibi, and R. M. Murray, “On a stochastic sensor selection algorithm with applications in sensor scheduling and sensor coverage,” *Automatica*, vol. 42, no. 2, pp. 251–260, 2006.

[3] A. Tiwari, M. Jun, D. E. Jeffcoat, and R. M. Murray, “Analysis of dynamic sensor coverage problem using Kalman filters for estimation,” *IFAC Proceedings Volumes*, vol. 38, no. 1, pp. 53–58, 2005.

[4] H. Sandberg, M. Rabi, M. Skoglund, and K. H. Johansson, “Estimation over heterogeneous sensor networks,” *Decision and Control, 47th IEEE Conference on*, pp. 4898–4903, 2008.

[5] C. Yang and L. Shi, “Deterministic sensor data scheduling under limited communication resource,” *IEEE Trans. on Signal Processing*, vol. 59, no. 10, pp. 5050–5056, 2011.

[6] L. Shi and H. Zhang, “Scheduling two gauss–markov systems: An optimal solution for remote state estimation under bandwidth constraint,” *IEEE Trans. on Signal Processing*, vol. 60, no. 4, pp. 2038–2042, 2012.

[7] C. O. Savage and B. F. La Scala, “Optimal scheduling of scalar gauss-markov systems with a terminal cost function,” *IEEE Trans. on Automatic Control*, vol. 54, no. 5, pp. 1100–1105, 2009.

[8] X. Gao, E. Akyol, and T. Başar, “Optimal communication scheduling and remote estimation over an additive noise channel,” *Automatica*, vol. 88, pp. 57–69, 2018.

[9] O. C. Imer and T. Basar, “Optimal estimation with limited measurements,” *Decision and Control, European Control Conference. CDC-ECC. 44th IEEE Conference on*, pp. 1029–1034, 2005.

[10] L. Li, M. Lemmon, and X. Wang, “Event-triggered state estimation in vector linear processes,” *American Control Conference (ACC)*, pp. 2138–2143, 2010.

[11] R. Cogill, S. Lall, and J. P. Hespanha, “A constant factor approximation algorithm for event-based sampling,” *American Control Conference*, pp. 305–311, 2007.

[12] J. Wu, Q.-S. Jia, K. H. Johansson, and L. Shi, “Event-based sensor data scheduling:trade-off between communication rate and estimation quality,” *IEEE Trans. on automatic control*, vol. 58, no. 4, pp. 1041–1046, 2013.

[13] L. Shi, K. H. Johansson, and L. Qiu, “Time and event-based sensor scheduling for networks with limited communication resources,” *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 13 263–13 268, 2011.

[14] S. Liu, M. Fardad, E. Masazade, and P. K. Varshney, “Optimal periodic sensor scheduling in networks of dynamical systems,” *IEEE Trans. on Signal Processing*, vol. 62, no. 12, pp. 3055–3068, 2014.

[15] W. H. Heemels, M. Donkers, and A. R. Teel, “Periodic event-triggered control for linear systems,” *IEEE Trans. on Automatic Control*, vol. 58, no. 4, pp. 847–861, 2013.

[16] W. Heemels and M. Donkers, “Model-based periodic event-triggered control for linear systems,” *Automatica*, vol. 49, no. 3, pp. 698–711, 2013.

[17] X.-M. Zhang and Q.-L. Han, “A decentralized event-triggered dissipative control scheme for systems with multiple sensors to sample the system outputs,” *IEEE Trans. on cybernetics*, vol. 46, no. 12, pp. 2745–2757, 2016.

[18] W. Heemels, K. H. Johansson, and P. Tabuada, “An introduction to event-triggered and self-triggered control,” pp. 3270–3285, 2012.

[19] F. P. Kelly, A. K. Maulloo, and D. K. Tan, “Rate control for communication networks: shadow prices, proportional fairness and stability,” *Journal of the Operational Research society*, vol. 49, no. 3, pp. 237–252, 1998.

[20] S. H. Low and D. E. Lapsley, “Optimization flow control. I. Basic algorithm and convergence,” *IEEE/ACM Trans. on networking*, vol. 7, no. 6, pp. 861–874, 1999.

[21] B. Johansson, P. Soldati, and M. Johansson, “Mathematical decomposition techniques for distributed cross-layer optimization of data networks,” *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1535–1547, 2006.

[22] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, “Layering as optimization decomposition: A mathematical theory of network architectures,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 255–312, 2007.

[23] D. Han, Y. Mo, J. Wu, S. Weerakkody, B. Sinopoli, and L. Shi, “Stochastic event-triggered sensor schedule for remote state estimation,” *IEEE Trans. on Automatic Control*, vol. 60, no. 10, pp. 2661–2675, 2015.

[24] A. Nayyar, T. Başar, D. Teneketzis, and V. V. Veeravalli, “Optimal strategies for communication and remote estimation with an energy harvesting sensor,” *IEEE Trans. on Automatic Control*, vol. 58, no. 9, pp. 2246–2260, 2013.

[25] S. Boyd and L. Vandenberghe, “Convex optimization,” 2004.

[26] S. Boyd, L. Xiao, A. Mutapcic, and J. Mattingley, “Notes on decomposition methods, stanford university,” 2008.

[27] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.