Convex Optimization Approaches to Long-Term Sensor Scheduling

Marco F. Huber

Variable Image Acquisition and Processing Research Group (VBV)
Fraunhofer Institute of Optronics, System Technologies and
Image Exploitation IOSB
marco.huber@ieee.org

Technical Report IES-2009-01

Abstract: The optimization over long time horizons in order to consider long-term effects is of paramount importance for effective sensor scheduling in multi-sensor systems like sensor arrays or sensor networks. Determining the optimal sensor schedule, however, is equivalent to solving a binary integer program, which is computationally demanding for long time horizons and many sensors. For linear Gaussian models, two efficient long-term sensor scheduling approaches are proposed in this report. The first approach determines approximate but close to optimal sensor schedules via convex optimization. The second approach combines convex optimization with a branch-and-bound search for efficiently determining the optimal sensor schedule. Both approaches are compared by means of numerical simulations.

Notation

$x, \underline{x}$	deterministic variable/vector
$\boldsymbol{x}, \underline{\boldsymbol{x}}$	random variable/vector
$\hat{x}, \underline{\hat{x}}$	mean value of random variable/vector
$\underline{x}_k, \underline{x}_{1:k}$	vector at time step k / sequence of vectors from time step 1 to k
$\mathcal{A}$	general set
$\mathbf{A}$	general matrix
$\lvert\mathbf{A}\rvert$	matrix determinant
$\mathcal{N}(\underline{x}; \underline{\hat{x}}, \mathbf{C})$	multivariate Gaussian density with mean $\underline{\hat{x}}$ and covariance $\mathbf{C}$
$\underline{u}_{1:k}, \underline{u}^*_{1:k}$	sensor schedule for time step 1 to k / optimal sensor schedule
$\underline{u}^{\mathrm{l}}_{1:k}, \underline{u}^{\mathrm{u}}_{1:k}$	sensor schedule obtained via: convex optimization (real-valued, provides lower bound) / conversion (binary-valued, provides upper bound)
$J(\underline{u}_{1:k})$	objective function value of sensor schedule $\underline{u}_{1:k}$

1 Introduction

Recent developments in wireless communication and sensor technology facilitate building up and deploying sensor systems for a smart and persistent surveillance. For instance, sensor networks consisting of numerous inexpensive sensor nodes are a popular subject in research and practice for monitoring physical phenomena including, e.g., temperature and humidity distributions, biochemical concentrations, or vibrations in buildings [ASSC02]. For many of such sensor systems it is necessary to balance between maximizing the information gain and minimizing the consumption of limited resources like energy, computing power, or communication bandwidth. *Sensor scheduling*, which is also referred to as sensor selection, allows trading off these conflicting goals and forms the basis for an efficient and intelligent processing of the sensor data.

A sensor schedule specifies a time sequence of sensors to be allocated for performing future measurements. The main objective is to allocate the sensors in a most informative way, which requires making decisions involving multiple time steps ahead. In this report, sensor scheduling for linear Gaussian dynamics and sensor models is studied, where one out of a set of sensors is selected at each time instant for performing a measurement. For such models, one of the first works on long-term sensor scheduling can be found in [MPD67]. It is shown that a separation principle holds, i.e., the sensor schedule can be determined independent of the control of the observed system and independent of the measurement values. The optimal sensor schedule then results from off-line traversing a decision tree consisting of all possible sensor sequences. In order to avoid enumerating all schedules in a brute force fashion, which is of exponential complexity, optimal or suboptimal pruning techniques are employed. Optimal techniques yield the optimal sensor schedule by all means without the need of examining all schedules (see for example [LI99, HH08]). Suboptimal methods as those in [GCHM04] allow more significant savings in computational demand by abdicating the guarantee of conserving the optimal schedule. Greedy, or myopic, scheduling algorithms represent an extreme case of suboptimal search, where a series of one-step ahead solutions is calculated [Osh94, QKS07].

Alternatively to traversing the decision tree, which corresponds to solving a binary integer program, convex optimization approaches have recently been proposed for solving sensor selection problems, i.e., problems of selecting the best n-element subset from a set of sensors (see [CMPS07, JB09]). These approaches can significantly improve the efficiency of determining informative sensor schedules, but they are so far not appropriate for *optimal long-term* sensor scheduling for *arbitrary* linear Gaussian dynamics and sensor models.

Both long-term sensor scheduling approaches proposed in this report overcome these restrictions. At first a general sensor scheduling problem for linear Gaussian models is formulated in Section 2. In Section 3 it is shown that this sensor scheduling problem is a convex optimization problem when employing continuous relaxation of the decision variables. The first approach directly solves the resulting convex program, which leads to suboptimal but valuable sensor schedules without demanding many computations and memory. In order to provide the optimal sensor sequence, the second approach described in Section 4 utilizes branch-and-bound search for traversing a decision tree. To exclude complete subtrees containing suboptimal sensor schedules as early as possible, the solution of the convex optimization is used for calculating tight lower and upper bounds to the subtrees' values. The performance of the proposed approaches is demonstrated by means of simulations in Section 5, while in Section 4 conclusions and an outlook to future work are given.

2 Problem Formulation

In this report, the sensor scheduling problem for discrete-time linear Gaussian models is examined. The dynamics model of the observed system is given by

$$\underline{\boldsymbol{x}}_{k+1} = \mathbf{A}_k \cdot \underline{\boldsymbol{x}}_k + \underline{\boldsymbol{w}}_k \, . \tag{2.1}$$

A finite set $\mathcal{S}$ of sensors is considered for performing measurements, where measurement $\underline{\boldsymbol{z}}_k^i$ from sensor $i \in \mathcal{S} = \{1, \ldots, S\}$ is related to the system state $\underline{\boldsymbol{x}}_k$ via the measurement model

$$\underline{\boldsymbol{z}}_k^i = \mathbf{H}_k^i \cdot \underline{\boldsymbol{x}}_k + \underline{\boldsymbol{v}}_k^i \, .$$

Both $\mathbf{A}_k$ and $\mathbf{H}_k^i$ are time-variant matrices. The noise terms $\underline{\boldsymbol{w}}_k$ and $\underline{\boldsymbol{v}}_k^i$ are zero-mean white Gaussian with covariance matrices $\mathbf{C}_k^w$ and $\mathbf{C}_k^{v,i}$, respectively. A measurement value $\hat{\underline{z}}_k^i$ of sensor $i \in \mathcal{S}$ is a realization of $\underline{\boldsymbol{z}}_k^i$. The initial system state $\underline{\boldsymbol{x}}_0 \sim \mathcal{N}(\underline{x}_0; \hat{\underline{x}}_0, \mathbf{C}_0^x)$ at time step $k = 0$ is Gaussian with mean $\hat{\underline{x}}_0$ and covariance $\mathbf{C}_0^x$.

The aim of long-term sensor scheduling is to minimize the covariance $\mathbf{C}_k^x$ of the state $\underline{\boldsymbol{x}}_k$ and thus, to minimize the uncertainty of the state estimate under the consideration of the future behavior of the observed dynamical system and long-term sensing costs. For this purpose, the optimal sensor schedule $\underline{u}_{1:N}^* = \left[(\underline{u}_1^*)^{\mathrm{T}}, \ldots, (\underline{u}_N^*)^{\mathrm{T}} \right]^{\mathrm{T}} \in \{0,1\}^{S \cdot N}$ is determined over a finite N-step time horizon. Here, $\underline{u}_k^* = [u_{k,1}, \ldots, u_{k,S}]^{\mathrm{T}}$ encodes the index of the sensor scheduled for measurement at time step k, i.e., if sensor i is scheduled at time step k then $u_{k,i} = 1$ and $u_{k,j} = 0$ for all $j \neq i$.

For determining the optimal sensor schedule $\underline{u}^*_{1:N}$, the constraint optimization problem

$$\underline{u}^*_{1:N} = \arg\min_{\underline{u}_{1:N}} J(\underline{u}_{1:N}) \tag{2.2}$$

$$\text{subject to } \sum_{k=1}^{N} \underline{c}_k^{\mathrm{T}} \cdot \underline{u}_k \leq C \,, \tag{2.3}$$

$$\underline{1}^{\mathrm{T}} \cdot \underline{u}_k = 1 \,, \quad k = 1, \ldots, N \,, \tag{2.4}$$

$$\underline{u}_k \in \{0,1\}^S \,, \quad k = 1, \ldots, N \tag{2.5}$$

is formulated, where $J(\underline{u}_{1:N}) = \sum_{k=1}^{N} g_k(\underline{u}_{1:k})$ is the cumulative *objective function* to be minimized. In (2.3), $\underline{c}_k = [c_{k,1}, \ldots, c_{k,S}]^{\mathrm{T}}$ contains the sensor costs $c_{k,i}$, e.g., energy or communication, of selecting sensor i at time step k. With this constraint it is guaranteed that a feasible sensor schedule does not exceed a maximum cost C. The scalar functions $g_k(\,\cdot\,)$, i.e., the summands of $J(\underline{u}_{1:N})$, quantify the uncertainty subsumed in $\mathbf{C}^x_k(\underline{u}_{1:k})$. They can be

- the trace operator $\mathrm{trace}\,(\mathbf{C}^x_k(\underline{u}_{1:k}))$, whose minimization corresponds (graphically spoken) to minimizing the perimeter of the rectangular region enclosing the covariance ellipsoid,
- the root-determinant $\sqrt{|\mathbf{C}^x_k(\underline{u}_{1:k})|}$, which leads to the minimization of the volume of the covariance ellipsoid, or
- the maximum eigenvalue $\lambda_{\max}\,(\mathbf{C}^x_k(\underline{u}_{1:k}))$, whose minimization corresponds to minimizing the largest principal axis of the covariance ellipsoid.

The covariance itself is given by the information form of the Kalman filter covariance recursion (see for example [KSH00])

$$\mathbf{C}^x_k(\underline{u}_{1:k}) = \Big(\big(\mathbf{A}_{k-1} \cdot \mathbf{C}^x_{k-1}(\underline{u}_{1:k-1}) \cdot \mathbf{A}_{k-1}^{\mathrm{T}} + \mathbf{C}^w_{k-1} \big)^{-1} + \sum_{i=1}^{S} u_{k,i} \cdot \big(\mathbf{H}^i_k \big)^{\mathrm{T}} \cdot \big(\mathbf{C}^{v,i}_k \big)^{-1} \cdot \mathbf{H}^i_k \Big)^{-1} \,, \tag{2.6}$$

commencing from $\mathbf{C}^x_0$.

The constraints in (2.4) and (2.5) together ensure that one sensor per time step is selected for measurement. This restriction is made for brevity and clarity reasons. The extension to selecting multiple sensors per time step can be achieved by replacing the right hand side of (2.4) with the desired number of sensors. Alternatively, by modifying (2.2) and (2.3), is is also possible to minimize the sensor costs regarding a maximum allowed value of $J(\,\cdot\,)$, i.e., a maximum allowed uncertainty.

3 Convex Relaxation

The optimization problem in (2.2)–(2.5) is a so-called *binary integer program*. Problems of this type are known to be NP-hard (see [Kar72]) and thus, obtaining the optimal solution for large N and/or large S is computationally prohibitive in general. However, by replacing the binary non-convex constraints in (2.5) with the linear constraints $\underline{u}_k \in [0,1]^S$ for $k = 1,\ldots,N$, a convex relaxation of the original problem (2.2) is obtained. To see this, it is important to note that the constraints (2.3) and (2.4) are already convex. Furthermore, as shown in the following theorem, the sum to be minimized in (2.2) is now convex as well.

Theorem 1 (Convex Objective Function) *The objective function $J(\underline{u}_{1:N})$ in* (2.2) *is convex in terms of $\underline{u}_{1:N} \in [0,1]^{S\cdot N}$.*

PROOF. To prove the convexity of $g_k(\underline{u}_{1:k})$ and thus of $J(\underline{u}_{1:N})$, it must be shown that (see for example [BV08])

$$g_k(\lambda\cdot\underline{u}_{1:k} + (1-\lambda)\cdot\tilde{\underline{u}}_{1:k}) \;\leq\; \lambda\cdot g_k(\underline{u}_{1:k}) + (1-\lambda)\cdot g_k(\tilde{\underline{u}}_{1:k}) \qquad (3.1)$$

for $k = 1,\ldots,N$, $\forall\,\underline{u}_{1:k},\tilde{\underline{u}}_{1:k} \in [0,1]^{k\cdot S}$, and $\forall\,\lambda\in[0,1]$.

At first, it is proven by induction that the covariance recursion (2.6) is a convex function of $\underline{u}_{1:k}$. The induction starts with $\mathbf{C}_1^x(\underline{u}_1)$. Defining $\mathbf{M}_k^i := \left(\mathbf{H}_k^i\right)^{\mathrm{T}}\cdot\left(\mathbf{C}_k^{v,i}\right)^{-1}\cdot\mathbf{H}_k^i$ and $\mathbf{P}_1(\underline{u}_1) := \left(\mathbf{A}_0\cdot\mathbf{C}_0^x\cdot\mathbf{A}_0^{\mathrm{T}} + \mathbf{C}_0^w\right)^{-1} + \sum_i u_{1,i}\cdot\mathbf{M}_1^i$ and utilizing the results in [Kra36] on matrix convex functions, it follows from the matrix convexity property of the matrix inversion that

$$\begin{aligned}\mathbf{C}_1^x(\lambda\cdot\underline{u}_1 + (1-\lambda)\cdot\tilde{\underline{u}}_1) &= (\lambda\cdot\mathbf{P}_1(\underline{u}_1) + (1-\lambda)\cdot\mathbf{P}_1(\tilde{\underline{u}}_1))^{-1}\\ &\leq \lambda\cdot\underbrace{\mathbf{P}_1^{-1}(\underline{u}_1)}_{=\mathbf{C}_1^x(\underline{u}_1)} + (1-\lambda)\cdot\underbrace{\mathbf{P}_1^{-1}(\tilde{\underline{u}}_1)}_{=\mathbf{C}_1^x(\tilde{\underline{u}}_1)}\end{aligned}$$

$\forall\,\underline{u}_1,\tilde{\underline{u}}_1 \in [0,1]^S$ and $\forall\,\lambda\in[0,1]$. Defining the predicted covariance $\mathbf{C}_k^p(\underline{u}_{1:k-1}) := \mathbf{A}_{k-1}\cdot\mathbf{C}_{k-1}^x(\underline{u}_{1:k-1})\cdot\mathbf{A}_{k-1}^{\mathrm{T}} + \mathbf{C}_{k-1}^w$, it generally holds that

$$\begin{aligned}&\mathbf{C}_k^x(\lambda\cdot\underline{u}_{1:k} + (1-\lambda)\cdot\tilde{\underline{u}}_{1:k})\\ &= \left(\mathbf{C}_k^p\big(\lambda\cdot\underline{u}_{1:k-1} + (1-\lambda)\cdot\tilde{\underline{u}}_{1:k-1}\big)^{-1} + \sum_{i=1}^{S}(\lambda\cdot u_{k,i} + (1-\lambda)\cdot\tilde{u}_{k,i})\cdot\mathbf{M}_k^i\right)^{-1}\\ &\overset{(a)}{\leq} \left(\lambda\cdot\left(\mathbf{C}_k^p\big(\underline{u}_{1:k-1}\big)^{-1} + \sum_{i=1}^{S} u_{k,i}\cdot\mathbf{M}_k^i\right)\right.\\ &\qquad\left. + (1-\lambda)\cdot\left(\mathbf{C}_k^p\big(\tilde{\underline{u}}_{1:k-1}\big)^{-1} + \sum_{i=1}^{S}\tilde{u}_{k,i}\cdot\mathbf{M}_k^i\right)\right)^{-1}\end{aligned}$$

$$\overset{(b)}{\leq} \lambda \cdot \mathbf{C}_k^x(\underline{u}_{1:k}) + (1-\lambda) \cdot \mathbf{C}_k^x(\tilde{\underline{u}}_{1:k}) \tag{3.2}$$

for $k = 2, \ldots, N$, $\forall\, \underline{u}_{1:k}, \tilde{\underline{u}}_{1:k} \in [0,1]^{k \cdot S}$, and $\forall\, \lambda \in [0,1]$. Here, (a) results from the induction hypothesis that $\mathbf{C}_{k-1}^x(\underline{u}_{1:k-1})$ is convex in $\underline{u}_{1:k-1}$, from the convexity of the matrix inversion, and from rearranging terms; (b) is the result of a repeated application of the convexity of the matrix inversion.

As the trace is a linear matrix function and the root-determinant as well as the maximum eigenvalue are convex matrix functions (see for example [BV08]), the inequality in (3.1) holds if these three functions are applied on (3.2). Thus, $g_k(\underline{u}_{1:k})$ is convex and the nonnegative sum $J(\underline{u}_{1:N}) = \sum_{k=1}^{N} g_k(\underline{u}_{1:k})$ is convex as well, which concludes the proof. □

It is important to note that the sensor scheduling problem formulated by (2.2)–(2.5) and its convex relaxation proven in Theorem 1 extends existing convex approaches [CMPS07, JB09] in many ways. Instead of one-step time horizons, i.e., myoptic/greedy scheduling, arbitrarily long time horizons are possible. Furthermore, the dynamics model in (2.1) need not to be restricted to regular system matrices $\mathbf{A}_k$ and to system noise covariances $\mathbf{C}_k^w = \mathbf{0}$. Especially the latter is of paramount importance for realistic sensor scheduling problems. Finally, there is no restriction to a specific scalar function $g_k(\,\cdot\,)$ as in [CMPS07]. Instead, various functions for evaluating the quality of a sensor schedule are considered here.

3.1 Solving the Relaxed Problem

The computational complexity of optimally solving the original binary integer program is in $\mathcal{O}(S^N)$. Various methods are available for efficiently solving the convex relaxation of the sensor scheduling problem, e.g., interior-point methods [BV08]. These methods typically require only a few tens of iterations for calculating the optimal solution even for large problem sizes, e.g., length of time horizon and number of sensors beyond 10. The computational complexity of one iteration is polynomial in the number of variables in $\underline{u}_{1:N}$, which is $S \cdot N$. The derivation of the gradient of $J(\underline{u}_{1:N})$ necessary for interior-point methods is shown in Appendix A.

The solution $\underline{u}_{1:N}^{\mathrm{l}}$ of the convex problem, however, only approximates the optimal solution $\underline{u}_{1:n}^{*}$ of the original scheduling problem. More specifically, $\underline{u}_{1:N}^{\mathrm{l}}$ is no longer binary and the objective function value $J^{\mathrm{l}} := J(\underline{u}_{1:N}^{\mathrm{l}})$ is a *lower bound* of the optimal value $J(\underline{u}_{1:N}^{*})$. The latter finding follows directly from the convexity of the relaxed problem and from the fact that the relaxed solution set $[0,1]^{S \cdot N}$ contains the binary set of the original problem.

3.2 Conversion into Binary Solution

In order to allow selecting sensors for measurement, $\underline{u}^{\mathrm{l}}_{1:N}$ has to be converted into a binary vector by employing an appropriate conversion or rounding method. The value $J^{\mathrm{u}} := J(\underline{u}^{\mathrm{u}}_{1:N})$ of the resulting (binary) sensor schedule $\underline{u}^{\mathrm{u}}_{1:N}$ has to be as close as possible to the optimal one in order to provide informative sensor measurements. In the following, two appropriate conversion methods are introduced. Independent of the chosen conversion method, the value J^{u} of the converted sensor schedule provides an *upper bound* to the optimal value $J(\underline{u}^{*}_{1:N})$.

3.2.1 Sampling

Each component $\underline{u}^{\mathrm{l}}_{k}$ of $\underline{u}^{\mathrm{l}}_{1:N}$ can be interpreted as a discrete probability distribution over the set of sensor indices $\mathcal{S}$. This is due to the constraint in (2.4), whereby the elements $u^{\mathrm{l}}_{k,i}, i = 1, \ldots, S$ of $\underline{u}^{\mathrm{l}}_{k}$ are within the interval $[0, 1]$ and sum up to one. Hence, a sensor i corresponding to an element $u^{\mathrm{l}}_{k,i}$ with a large value can be considered as being more likely in the optimal sensor schedule than sensors with small values.

To convert $\underline{u}^{\mathrm{l}}_{1:N}$ into a feasible binary vector, for each $k = 1, \ldots, N$ a (single) sensor is randomly selected according to the distribution $\underline{u}^{\mathrm{l}}_{k}$. For being *feasible*, the resulting converted schedule $\underline{u}^{\mathrm{u}}_{1:k}$ has to satisfy the cost constraint (2.3). Otherwise, the schedule is discarded. This procedure is repeated multiple times, where only the currently best feasible schedule, i.e., the schedule that satisfies (2.3) *and* provides the currently smallest objective function value J^{u} is stored. The sampling-based conversion method can be terminated for example after a predefined number of trials or when the currently best value J^{u} remains unchanged for a predefined number of trials.

3.2.2 Swapping

To improve a converted schedule $\underline{u}^{\mathrm{u}}_{1:N}$, the swapping method proposed in [JB09] can be adapted. A modified sensor schedule is derived from $\underline{u}^{\mathrm{u}}_{1:N}$ by swapping a scheduled sensor with one of the unselected sensors for each time step. The choice of an unselected sensor at time step k is *deterministically* guided according to the probabilities represented by $\underline{u}^{\mathrm{l}}_{k}$, i.e., the sensors are selected in descending order of the values in $\underline{u}^{\mathrm{l}}_{k}$. If the modified schedule is feasible and improves the objective function value J^{u}, it is used for initializing the next swapping trial.

In order to start the swapping method with a feasible schedule, the sensor schedule that selects at each time step k the sensor $i = \arg\min_j c_{k,j}$ with the smallest cost

is chosen initially. The method must terminate because there is only a finite but very large number of swapping possibilities. To bound the computational demand, the number of swapping trials is limited by means of a predefined value.

4 Optimal Scheduling

Determining the optimal sensor schedule and thus, directly solving the binary integer program given by (2.2)–(2.5) can be considered as searching a decision tree with depth N and branching factor S. The problem here is that the optimal solution often can be found at an early stage when employing appropriate search methods, while the proof of its optimality requires evaluating most of the suboptimal sensor schedules, which is infeasible for large problem sizes. In this section, the previously introduced convex optimization approach is combined with efficient search methods for decision trees for early eliminating (pruning) suboptimal schedules.

4.1 Branch-and-Bound

A search technique common for classical decision problems like traveling-salesman or knapsack is branch-and-bound (BB) search. The basic idea of BB is to assign lower and upper bounds of the achievable objective function value to any visited node. Based on these bounds, nodes and thus complete subtrees can be pruned under the guarantee that the pruned node is not part of the optimal sensor schedule.

For a particular node that was reached during the search by employing the sensor schedule $\underline{u}_{1:k} \in \{0,1\}^{k \cdot S}$, the objective function can be written according to

$$J(\underline{u}_{1:N}) = \underbrace{J(\underline{u}_{1:k})}_{\text{known}} + \underbrace{J(\underline{u}_{k+1:N})}_{\text{unkown}} , \tag{4.1}$$

where only the value of the first summand is already evaluated and thus known. While the value of the second summand is not calculated yet, a lower and upper bound can be easily assigned to it by exploiting the results of Section 3.1 and Section 3.2. The value of the optimal solution $\underline{u}^{\mathrm{l}}_{k+1:N}$ of the convex relaxation for minimizing $J(\underline{u}_{k+1:N})$ serves as lower bound and the conversion of $\underline{u}^{\mathrm{l}}_{k+1:N}$ into a binary-valued vector $\underline{u}^{\mathrm{u}}_{k+1:N}$ provides an upper bound. Hence, the inequality

$$J(\underline{u}_{1:k}) + J(\underline{u}^{\mathrm{l}}_{k+1:N}) \leq J(\underline{u}_{1:N}) \leq J(\underline{u}_{1:k}) + J(\underline{u}^{\mathrm{u}}_{k+1:N})$$

holds for the objective function value in (4.1).

Algorithm 4.1 Initially $J_{\min} = \infty$. For a given sensor schedule $\underline{u}_{1:k}$ do:

if leaf node, i.e., $k = N$ **then**
  $J_{\min} \leftarrow J(\underline{u}_{1:N})$ // Global bound of currently best schedule $\underline{u}_{0:N-1}$
else
  $\mathcal{U} \leftarrow \emptyset$ // List of sensors to expand
  for all sensors $i \in \{1, \ldots, S\}$ **do** // $\underline{u}_{1:k}$ and $u_{k+1,i} = 1$ fixed
    if $\text{cost}_i \leq C$ and $J_i \leq J_{\min}$ **then**
      $J_i^{\text{l}} \leftarrow$ Solve convex optimization problem
      $J_i^{\text{u}} \leftarrow$ Calculate upper bound via conversion
      $\mathcal{U} \leftarrow \mathcal{U} \cup \{i\}$
    end if
  end for
  $\mathcal{U} \leftarrow \text{sort}(\mathcal{U})$ // Sort sensors based on J_i^{l}
  for all sensors $i \in \mathcal{U}$ **do**
    if $J_i^{\text{l}} \leq J_{\min}$ and $\forall\, j \in \mathcal{U} : J_i^{\text{l}} \leq J_j^{\text{u}}$ **then**
      Expand i // Set $u_{k+1,i} = 1$, call Algorithm 4.1
    end if
  end for
end if

It is worth mentioning that a valuable upper bound, i.e., a tight one, normally cannot be provided for branch-and-bound for sensor scheduling tasks (see for example [CMPS06, Hub09]). Here, the solution of the convex relaxation allows calculating tight upper bounds in a straightforward fashion. These further reduce the size of the search space.

4.2 Search Algorithm

The combination of BB search with convex optimization is illustrated in Algorithm 4.1, which basically employs a depth-first search. For a given sensor schedule $\underline{u}_{1:k}$ it is checked, which child nodes should be expanded, i.e., it is checked whether an element $u_{k+1,i}$, $i \in \mathcal{S}$ of $\underline{u}_{k+1}$ could be set to one or not. Therefore, for each child node $i \in \mathcal{S}$ the minimum cost possible is computed as

$$\text{cost}_i := \sum_{n=1}^{k} \underline{c}_n \cdot \underline{u}_n + c_{k+1,i} + \sum_{n=k+2}^{N} \min_j c_{n,j} \; .$$

Furthermore, the value $J_i := J(\underline{u}_{1:k+1})$ and the bounds $J_i^{\text{l}} := J_i + J(\underline{u}_{k+2:N}^{\text{l}})$, $J_i^{\text{u}} := J_i + J(\underline{u}_{k+2:N}^{\text{u}})$ are calculated, where $u_{k+1,i} = 1$ and $u_{k+1,j} = 0$ for

all $j \neq i$. Based on these values, a node i is expanded only if following four requirements are fulfilled:

1. The cost constraint can be met, i.e., a feasible solution exists (line 6).
2. The value J_i of the node is below the value $J_{\min}$ of the currently best sensor schedule (line 6).
3. The lower bound J_i^{l} is below $J_{\min}$ (line 14).
4. The lower bound is below the upper bounds of all neighboring nodes $j \neq i$ (line 14).

Obviously, the third requirement implies the second one. But in order to avoid an unnecessary calculation of the lower and upper bound, the second requirement is checked separately together with the first requirement (line 6–10). To further accelerate the search, the remaining sensors in $\mathcal{U}$ are sorted in descending order according of their lower bounds (line 12)[1]. In doing so, the search is continued with the most promising sensor first in order to force a stronger reduction of the currently best value $J_{\min}$. This value is automatically reduced once a leaf node is reached (line 1–2).

5 Simulation Results

The effectiveness of the proposed sensor scheduling methods is demonstrated in the following by means of a numerical simulation from the field of target tracking. The state $\underline{\boldsymbol{x}}_k = [\boldsymbol{x}_k, \dot{\boldsymbol{x}}_k, \boldsymbol{y}_k, \dot{\boldsymbol{y}}_k]^{\mathrm{T}}$ of the observed target comprises the two-dimensional position $[\boldsymbol{x}_k, \boldsymbol{y}_k]^{\mathrm{T}}$ and the velocities $[\dot{\boldsymbol{x}}_k, \dot{\boldsymbol{y}}_k]^{\mathrm{T}}$ in x and y direction. The system matrix and noise covariance matrix of $\underline{\boldsymbol{w}}_k$ of the dynamics model (2.1) are

$$\mathbf{A}_k = \mathbf{I}_2 \otimes \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{C}_k^w = q \cdot \mathbf{I}_2 \otimes \begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^2}{2} & T \end{bmatrix} , \tag{5.1}$$

respectively, where $\mathbf{I}_n$ indicates an $n \times n$ identity matrix and $\otimes$ is the Kronecker matrix product. In (5.1), $T = 1$ s is the sampling interval and $q = 0.2$ is the scalar diffusion strength. Mean and covariance of the initial state $\underline{\boldsymbol{x}}_0$ are $\underline{\hat{x}}_0 = [0, 1, 0, 1]^{\mathrm{T}}$ and $\mathbf{C}_0^x = 10 \cdot \mathbf{I}_4$, respectively.

[1] Alternatively, $\mathcal{U}$ can be sorted in ascending order with respect to the upper bounds.

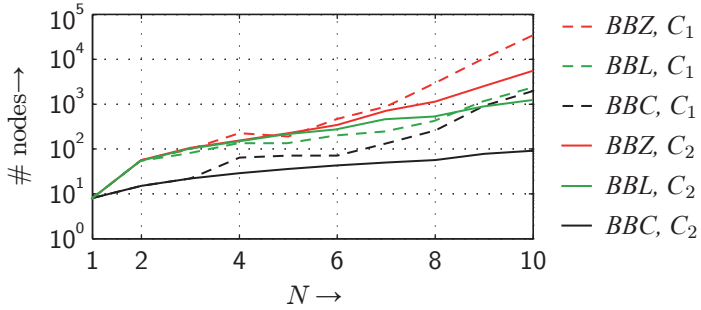

Figure 5.1: Number of nodes in the decision tree when applying the branch-and-bound-based scheduling methods *BBC* (black lines), *BBL* (green), and *BBZ* (red) for different time horizons lengths N and for two different maximum cost functions C_1 (dashed) and C_2 (solid) in log-scale.

A sensor network observes the target. It consists of six sensors with measurement matrices

$$\mathbf{H}_k^1 = \mathbf{H}_k^3 = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \ , \ \mathbf{H}_k^2 = \mathbf{H}_k^5 = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \ ,$$
$$\mathbf{H}_k^4 = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \ , \ \mathbf{H}_k^6 = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix} \ ,$$

noise variances $C_k^{v,1} = 0.2$, $C_k^{v,2} = C_k^{v,3} = C_k^{v,4} = 0.1$, $C_k^{v,5} = C_k^{v,6} = 0.05$, and costs $\underline{c}_k = [1, 1, 2, 1, 2, 2]^{\mathrm{T}}$ for each k. Furthermore, it is also possible to omit a measurement. This option can be considered as having a seventh sensor with infinite noise variance. Performing no measurement is free of cost, i.e., $c_{k,7} = 0$. Altogether, the set $\mathcal{S}$ comprises $S = 7$ sensors. The scalar functions $g_k(\,\cdot\,)$ are set to the root-determinant for each k.

For comparison, five different scheduling methods are considered: (1, denoted in the following by *CONVEX*) The approach described in Section 3, which directly solves the convex optimization problem and employs the swapping method for conversion. (2, *BBC*) The BB approach described in Section 4. For determining the upper bounds via conversion, the swapping method is employed. (3, *BBL*) Like *BBC* but without utilizing upper bounds for pruning. (4, *BBZ*) BB search that employs no upper bounds and bounds the second summand in (4.1) from below with zero. (5, *GREEDY*) The sensors are scheduled in a greedy (one-step lookahead) fashion (see for example [KG05]). For *CONVEX* and *BBC*, the number of swapping trials is set to $S \cdot N$.

In Figure 5.1, the search performance of the three BB methods is compared. For this purpose, two different maximum costs $C_1(N) = \mathrm{round}\left(\frac{3}{4} \cdot N\right)$ and $C_2(N) =$

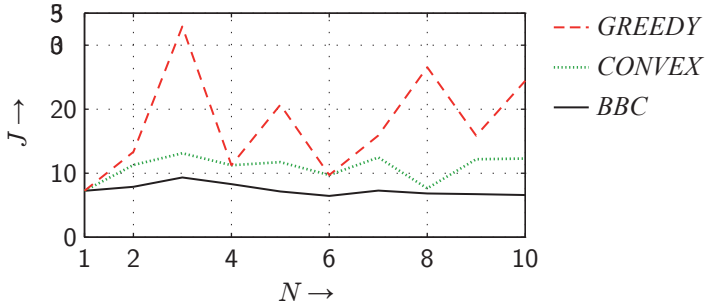

Figure 5.2: Objective function values J of the scheduling methods *BBC* (black, solid), *CONVEX* (green, dotted), and *GREEDY* (red, dashed) for maximum cost function C_1.

$2 \cdot N$ are considered, which depend on the change of the time horizon length $N = 1, \ldots, 10$. The maximum cost function $C_2(N)$ allows sensor scheduling without omitting a measurement. With the proposed optimal scheduling method *BBC*, the number of nodes in the decision tree can be kept on a low level. Here, the search performance clearly benefits from the tight lower and upper bounds provided by the convex optimization and the conversion, respectively. This can be seen in particular for C_2, where *BBC* only visits at most 92 nodes, while the complete decision tree contains $\sum_{k=1}^{N} 7^k < 3.3 \cdot 10^8$ nodes. The higher number of visited nodes for cost function C_1 compared to C_2 results from the effect that the more restrictive cost constraint provided by C_1 leads to looser bounds.

Without considering upper bounds for pruning as it is the case for *BBL* , the number of visited nodes increases significantly. But still, the search performance of *BBL* is much better than *BBZ* as the lower bound provided by the solution of the convex optimization is closer to the true values of the subtrees.

Since calculating lower and upper bounds by means of convex relaxation is computationally more demanding than calculating the simple bound used for *BBZ* , the runtime of *BBZ* is lower for short time horizons even if *BBZ* leads to larger decisions trees. But with increasing length of the time horizon, the difference in runtime between *BBZ* and the other BB methods becomes smaller and at some point, both methods outperform *BBZ*. For example, with the current, barely optimized implementation based on MATLAB version 7.9, *BBC* outperforms *BBZ* from horizon length $N = 9$ on for cost function C_1. It is expected that employing an optimized implementation, outperforming *BBZ* occurs for significantly shorter time horizons.

In Figure 5.2, the objective function values of *BBC* are compared with *GREEDY* and *CONVEX* for the costs $C_1(N)$. The *GREEDY* method is the computationally cheapest one, but provides highly suboptimal results. Due to the myopic planning, *GREEDY* is not able to anticipate the long-term effect of early selecting costly sensors. In this simulation example, *GREEDY* omits measurements at the last time steps of the horizon and not in between in order to meet the maximum cost constraint. The proposed suboptimal *CONVEX* method provides sensor schedules close to the optimal ones, whereas the computational demand is significantly smaller compared to *BBC*, especially for very long time horizons. *CONVEX* trades scheduling quality off against scheduling complexity, which is desirable for computationally constrained sensor systems.

6 Conclusions and Future Work

Employing convex optimization for determining long-term sensor schedules is a promising approach. In this report, a general sensor scheduling problem for linear Gaussian models was formulated and the convexity of its relaxation was proven. Based on this result, two scheduling methods utilizing convex optimization have been proposed. The first approach directly solves the relaxed sensor scheduling problem in order to provide suboptimal but computationally cheap solutions. The second approach provides the optimal sensor schedule, where convex optimization is utilized for eliminating suboptimal sensor schedules at an early stage of branch-and-bound search. Compared to existing approaches on sensor scheduling via convex optimization, general linear Gaussian sensor scheduling problems are covered. Furthermore, both proposed scheduling methods are appropriate for long time horizons and many sensors, where choosing the better suited approach for a given scheduling problem depends on the requirements on estimation quality and computational capabilities.

Future work is mainly devoted to three aspects: improving search speed for branch-and-bound search, incorporation of nonlinear dynamics and sensor models, and applying model-predictive/moving horizon control. Further improving search speed can for example be achieved by incorporating so-called *Gomory's cuts*, i.e., additional inequality constraints that reduce the size of the search space [SM99]. By this means the proposed branch-and-bound search would be extended into a branch-and-cut search. A completely different way of solving the sensor scheduling problem would be to employ so-called *outer approximation* [FL94]. Here, the convex relaxation of the sensor scheduling problem is transformed into a series of linear programs that finally leads to the optimal (binary) sensor schedule.

A comparison of branch-and-cut search and outer approximation with respect to computational demand would by of particular interest.

The incorporation of nonlinear models can be achieved by employing a conversion of the nonlinear models into linear ones via linearization, e.g., first-order Taylor series expansion or statistical linearization as in [Hub09]. This linearization can be combined with model-predictive control in order to facilitate sensor scheduling for very long or even infinite time horizons.

Appendix

A Analytical Expression of the Gradient

Since interior-point methods employ Newton's method, the computation time of solving the relaxed sensor scheduling problem can be significantly reduced by providing an analytical expression of the gradient of the objective function $J(\underline{u}_{1:N})$. The gradient is given by

$$\nabla J(\underline{u}_{1:n}) = \frac{\partial J(\underline{u}_{1:N})}{\partial \underline{u}_{1:N}} = \sum_{k=1}^{N} \frac{\partial g(\underline{u}_{1:k})}{\partial \underline{u}_{1:N}} \,,$$

which boils down to calculating the derivatives

$$\frac{\partial g(\underline{u}_{1:k})}{\partial \underline{u}_{1:N}} = \Bigg[\underbrace{\left(\frac{\partial g(\underline{u}_{1:k})}{\partial \underline{u}_{1:k}} \right)^{\mathrm{T}}}_{1:k}, \underbrace{\underline{0}^{\mathrm{T}}}_{k+1:N} \Bigg]^{\mathrm{T}} \tag{A.1}$$

for $k = 1, \ldots, N$.

The partial derivative $\frac{\partial g(\underline{u}_{1:k})}{\partial \underline{u}_{1:k}}$ in (A.1) requires determining the derivative $\frac{\partial \mathbf{C}_k^x}{\partial \underline{u}_{1:k}}$ with $\mathbf{C}_k^x$ according to (2.6).[2] With the differential identity (see [PP08])

$$\partial \mathbf{X}^{-1} = -\mathbf{X}^{-1} \cdot \partial \mathbf{X} \cdot \mathbf{X}^{-1} \,, \quad \mathbf{X} \in \mathbb{R}^{n \times n} \tag{A.2}$$

and defining

$$\begin{aligned} \mathbf{C}_k^p &:= \mathbf{A}_{k-1} \cdot \mathbf{C}_{k-1}^x \cdot \mathbf{A}_{k-1}^{\mathrm{T}} + \mathbf{C}_{k-1}^w \,, \\ \mathbf{M}_k^i &:= \left(\mathbf{H}_k^i \right)^{\mathrm{T}} \cdot \left(\mathbf{C}_k^{v,i} \right)^{-1} \cdot \mathbf{H}_k^i \,, \\ \mathbf{P}_k &:= \left(\mathbf{C}_k^x \right)^{-1} = \left(\mathbf{C}_k^p \right)^{-1} + \sum_i u_{k,i} \cdot \mathbf{M}_k^i \,, \end{aligned}$$

[2] The argument $\underline{u}_{1:k}$ is omitted in the following for clarity.

the desired derivative of $\mathbf{C}_k^x$ accords to

$$\frac{\partial \mathbf{C}_k^x}{\partial \underline{u}_{1:k}} = \frac{\partial \mathbf{P}_k^{-1}}{\partial \underline{u}_{1:k}} \overset{(A.2)}{=} -\mathbf{C}_k^x \cdot \frac{\partial \mathbf{P}_k}{\partial \underline{u}_{1:k}} \cdot \mathbf{C}_k^x \overset{(A.2)}{=} \begin{bmatrix} \mathbf{C}_k^x \cdot (\mathbf{C}_k^p)^{-1} \cdot \frac{\partial \mathbf{C}_k^p}{\partial \underline{u}_{1:k-1}} \cdot (\mathbf{C}_k^p)^{-1} \cdot \mathbf{C}_k^x \\ \mathbf{C}_k^x \cdot \left(\frac{\partial}{\partial \underline{u}_k} \sum_i u_{k,i} \cdot \mathbf{M}_k^i \right) \cdot \mathbf{C}_k^x \end{bmatrix} . \tag{A.3}$$

The first row in (A.3) can be written in recursive form as

$$\mathbf{C}_k^x \cdot (\mathbf{C}_k^p)^{-1} \cdot \frac{\partial \mathbf{C}_k^p}{\partial \underline{u}_{1:k-1}} \cdot (\mathbf{C}_k^p)^{-1} \cdot \mathbf{C}_k^x = \mathbf{C}_k^x \cdot (\mathbf{C}_k^p)^{-1} \cdot \mathbf{A}_{k-1} \cdot \frac{\partial \mathbf{C}_{k-1}^x}{\partial \underline{u}_{1:k-1}} \cdot \mathbf{A}_{k-1}^{\mathrm{T}} \cdot (\mathbf{C}_k^p)^{-1} \cdot \mathbf{C}_k^x$$

commencing from $\frac{\partial \mathbf{C}_1^x}{\partial u_{1,i}} = -\mathbf{C}_1^x \cdot \mathbf{M}_1^i \cdot \mathbf{C}_1^x$ for $i = 1, \ldots, S$. For each element $i = 1, \ldots, S$ of the second row in (A.3) holds $\frac{\partial}{\partial u_{k,i}} \sum_i u_{k,i} \cdot \mathbf{M}_k^i = \mathbf{M}_k^i$.

As the function $g(\cdot)$ can be the trace, root-determinant, or the maximum eigenvalue (see Section 2), the identities

$$\partial \operatorname{trace}(\mathbf{X}) = \operatorname{trace}(\partial \mathbf{X}) , \tag{A.4}$$

$$\partial \sqrt{|\mathbf{X}|} = \tfrac{1}{2} \sqrt{|\mathbf{X}|} \cdot \operatorname{trace}\left(\mathbf{X}^{-1} \cdot \partial \mathbf{X} \right) , \tag{A.5}$$

$$\partial \lambda_i(\mathbf{X}) = \underline{v}_i^{\mathrm{T}} \cdot \partial \mathbf{X} \cdot \underline{v}_i \tag{A.6}$$

from matrix calculus in differential form have to be applied *to each* partial derivative $\frac{\partial g(\underline{u}_{1:k})}{\partial u_{k,i}}$ for concluding the derivation of $\frac{\partial g(\underline{u}_{1:k})}{\partial \underline{u}_{1:k}}$. In (A.4)–(A.6), $\mathbf{X} \in \mathbb{R}^{n \times n}$ has to be replaced by $\mathbf{C}_k^x$. (A.4) and (A.5) can be found in [PP08]. In (A.6), λ_i is the i-the eigenvalue of $\mathbf{X}$ and $\underline{v}_i$ is the corresponding i-th normalized eigenvector, with $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$ (see [OW95]).

Bibliography

[ASSC02] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38:393–422, January 2002.

[BV08] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2008.

[CMPS06] Amit S. Chhetri, Darryl Morrell, and Antonia Papandreou-Suppappola. Nonmyopic Sensor Scheduling and its Efficient Implementation for Target Tracking Applications. *EURASIP Journal on Applied Signal Processing*, 2006:1–18, 2006.

[CMPS07] Amit S. Chhetri, Darryl Morrell, and Antonia Papandreou-Suppappola. On the Use of Binary Programming for Sensor Scheduling. *IEEE Transactions on Signal Processing*, 55(6):2826–2839, June 2007.

[FL94] Roger Fletcher and Sven Leyffer. Solving mixed integer nonlinear programs by outer approximation. *Mathematical Programming*, 66(1):327–349, August 1994.

[GCHM04] Vijay Gupta, Timothy H. Chung, Babak Hassibi, and Richard M. Murray. Sensor Scheduling Algorithms Requiring Limited Computations. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 825–828, Montreal, Quebec, Canada, 2004.

[HH08] Marco F. Huber and Uwe D. Hanebeck. Priority List Sensor Scheduling using Optimal Pruning. In *Proceedings of the 11th International Conference on Information Fusion (Fusion)*, Cologne, Germany, July 2008.

[Hub09] Marco Huber. *Probabilistic Framework for Sensor Management*. PhD thesis, Universität Karlsruhe (TH), 2009.

[JB09] Siddharth Joshi and Stephen Boyd. Sensor Selection via Convex Optimization. *IEEE Transactions on Signal Processing*, 57(2):451–462, February 2009.

[Kar72] Richard M. Karp. Reducibility Among Combinatorial Problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–104. New York: Plenum, 1972.

[KG05] Andreas Krause and Carlos Guestrin. Near-optimal Nonmyopic Value of Information in Graphical Models. In *Proceedings of the Twenty-First Annual Conference on Uncertainty in Artificial Intelligence*, pages 324–33, Edinburgh, Scotland, July 2005.

[Kra36] Fritz Kraus. Über konvexe Matrixfunktionen. *Mathematische Zeitschrift*, 41(1):18–42, December 1936.

[KSH00] Thomas Kailath, Ali H. Sayed, and Babak Hassibi. *Linear Estimation*. Prentice Hall, 2nd edition, 2000.

[LI99] Andrew Logothetis and Alf Isaksson. On sensor scheduling via information theoretic criteria. In *Proceedings of the 1999 American Control Conference (ACC)*, volume 4, pages 2402–2406, San Diego, CA, 1999.

[MPD67] Lewis Meier, John Peschon, and Robert M. Dressler. Optimal Control of Measurement Subsystems. *IEEE Transactions on Automatic Control*, AC-12(5):528–536, October 1967.

[Osh94] Yaakov Oshman. Optimal Sensor Selection Strategy for Discrete-Time State Estimators. *IEEE Transactions on Aerospace and Electronic Systems*, 30(2):307–314, April 1994.

[OW95] Michael L. Overton and Robert S. Womersley. Second Derivatives for Optimizing Eigenvalues of Symmetric Matrices. *SIAM Journal on Matrix Analysis and Applications*, 16(3):697–718, July 1995.

[PP08] Kaare Brandt Petersen and Michael Syskind Pedersen. The Matrix Cookbook. Online: http://www.imm.dtu.dk/pubdb/views/edoc_download.php/3274/pdf/imm3274.pdf, November 2008.

[QKS07] Zhi Quan, William J. Kaiser, and Ali H. Sayed. A Spatial Sampling Scheme Based on Innovations Diffusion in Sensor Networks. In *Proceedings of the Sixth International Conference on Information Processing in Sensor Networks (IPSN)*, pages 323–330, Cambridge, MA, April 2007.

[SM99] Robert A. Stubbs and Sanjay Mehrotra. A branch–and–cut method for 0–1 mixed convex programming. *Mathmatical Programming*, 86(3):515–532, December 1999.