Fast Action Elimination for Efficient Decision Making and Belief Space Planning Using Bounded Approximations

Khen Elimelech and Vadim Indelman

1 Introduction

Autonomous decision making is a fundamental problem in artificial intelligence and robotics, in which one wishes to select the action which maximizes some objective function. When solved iteratively, in a context of a specific problem with a goal definition, the problem turns into a planning problem. A solution to this problem must take into consideration many aspects such as goal setting, definition of candidate actions, accounting to different planning horizons and future developments, coordination of agents and so on. While these aspects can be optimized in the context of the discussed problem, basic decision making, the examination of a group of candidate actions under the objective measure, is still the common ground to all planning problems. Unfortunately, little work has been done on trying to optimize this elementary process, and even if the planning aspects are optimized, its solution is still usually done naively with an exhaustive calculation of the objective function for all candidates.

In our previous work [1–3], we raised the idea that the basic decision making process can be done more efficiently. There, we defined a set of fundamental notions to allow us to compare states in the context of the decision making. As far as we know, there were no prior attempts to do so beforehand. There we introduced the concept of *action consistent* state approximations, in order to to solely and directly reduce the complexity of the decision making, while maintaining the same action

K. Elimelech (✉)
Robotics and Autonomous Systems Program, Technion - Israel Institute of Technology, 32000 Haifa, Israel
e-mail: khen@campus.technion.ac.il
URL: https://www.khen.io

V. Indelman
Department of Aerospace Engineering, Technion - Israel Institute of Technology, 32000 Haifa, Israel
e-mail: vadim.indelman@technion.ac.il

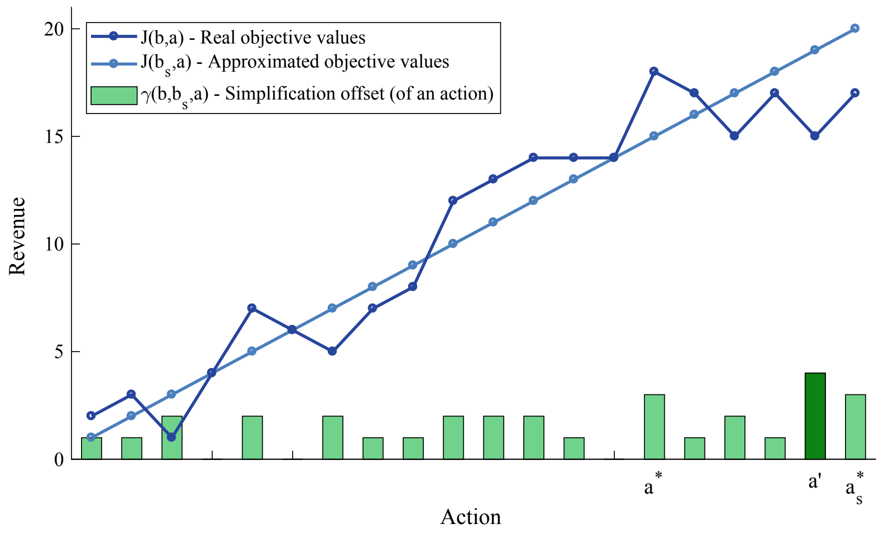

Fig. 1 This conceptual figure demonstrates a relation between two states b, b_s in the context of decision making. The dark blue graph shows the real objective values, based on the original state b, while the light blue graph shows the approximation of these values, based on the approximated state b_s. The states are not *action consistent*, since the graphs do not preserve the same trend. Hence, a^* is the real best candidate action, and a_s^* is the best candidate action based on the approximation. The green bars stand for the *simplification offset* between the states, for each action. The maximal offset, the one of action a', is the overall simplification offset between the two states. Obviously, without action elimination, in order to solve the problem, all the real objective values (the entire dark blue graph) have to be calculated

selection (Fig. 1). Despite involving state approximations, no dilution of information is done from the maintained state in the inference process, as the approximations are only used in candidate examination. In this work we wish to further demonstrate that this framework of definitions and qualitative analysis is worthwhile, by using it to create a new candidate-elimination paradigm to efficiently solve the decision making problem, in a general context. In contrast to our fundamental papers, where we solved an approximated problem, and then analyzed its consequences, this time we utilize state approximation to create a tool to be used in the solution of the original problem.

Here, we depend on the intuitive conclusion that in most cases, we can easily realize when an action is clearly "bad". For example, in the context of navigation to a given goal, backtracking away from the goal is (usually) undesired, and there is no need to invest in the calculation of how "bad" it is, when there are clearly better options. Thus, we should thoroughly examine only the subset of candidates which seems to be "good". Our method suggests using easy-to-calculate approximation of the objective function to identify unfit candidates, which, using our previous notions, can be guaranteed not to be the desired one, and eliminate them. The problem can then be efficiently solved considering a smaller subset of candidates, while sparing

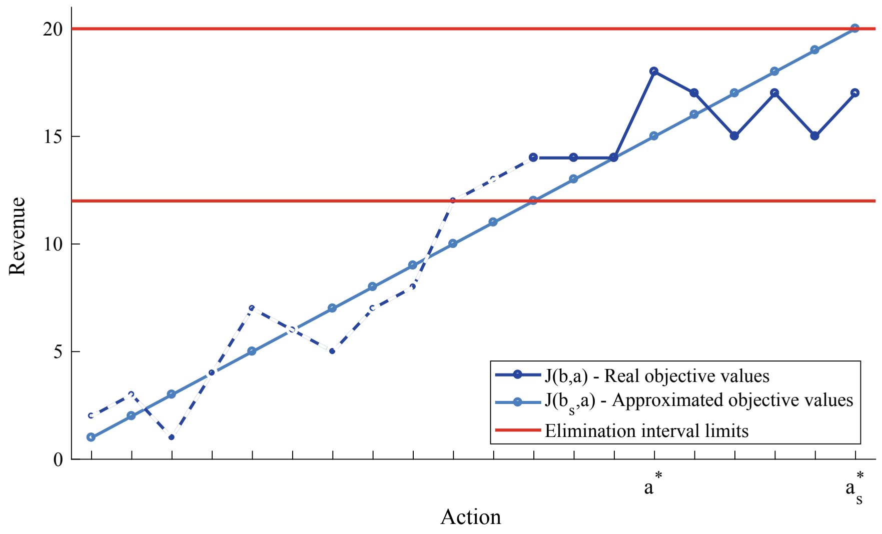

Fig. 2 This figure demonstrates the application of our action elimination method on the scenario described in Fig. 1. The light blue graph of the approximated objective values is easily calculated, and the approximated best action a_s^* is found. Using the simplification offset (known in this case, or bounded in the general case) and Eq. 8 from Theorem 1, the elimination interval, bounded by the two red lines, is calculated. $J(b_s, a^*)$, the approximated objective value of the real best candidate action, must be in this interval. Thus, all the actions for which the objective approximation $J(b_s, a)$ (in light blue) is below the lower red line, can be eliminated (the dashed section). Now, to solve the problem and find a^*, we should only calculate the real values $J(b, a)$ of the remaining actions (solid dark blue)

the need to calculate the exact objective values for each one (Fig. 2). By doing so, under real-time constraints, we are able to examine more complicated candidate actions, or use a higher resolution to discretize a continuous action domain, leading to an indirect improvement of solution quality as well. Moreover, since the method does not depend on any specific planning problem, it can be added "on top" of other methods, after the planning aspects were optimized, with no contradiction.

As a highly relevant case study, we extend the discussion to the sub-problem of decision making under uncertainty and belief space planning. Relevant planning problems include sensor deployment, for which we demonstrate the approach, and also robotic arm manipulation, autonomus navigation and SLAM. For the last one, improving the computational cost is especially important, as it is often required to solve this problem *online* and on limited hardware (such as quadcopters) [4]. Also, as the state size tends to grow over time in such problems, long-term autonomy is another challenge. In this context, several methods achieve indirect improvement in planning efficiency by using active sparsification of the maintained state [5, 6], but thus sacrifice the quality of solution of the state inference and by such also of the planning. Our method turns out superior in that matter, by being independent of the inference process, in addition to not affecting the quality of action selection.

Numerous studies have also attempted to directly optimize planning processes, although almost always in the context of specific problems, by adjusting and optimizing aspects in the representation of the problem; e.g., in the context of decision making under uncertainty and belief space planning [7–9], and specifically SLAM [10], path planning [11, 12], sensor deployment [13] and robotic arm manipulation [14]. Some papers have examined action elimination in the context of planning problems. In [15] actions are eliminated from a constructed plan, comprised of a sequence of actions, in order to reduce its execution complexity, and not the planning complexity. For learning problems, a few papers [16–18] have attempted to remove actions from consideration by transferring previous learned knowledge across learning tasks. Our method requires no previous knowledge, and not even for the decision making to be sequential. Several studies [19, 20] have utilized bounds for action elimination, in a similar way our approach, yet these bounds are tightly coupled with their formulation for planning under the limited MDP framework. In the context of motion planning in the belief space, pruning of the planning tree has been applied in order to remove unpromising candidate paths [21]. Overall, none of these approaches rely solely on the general decision problem, making these applications limited to the discussed scenario. Being applicable for different problems is the main advantage of our method, which also allows, as previously stated, to apply it with no contradiction alongside other optimizations or pruning methods.

2 Problem Formulation

2.1 Decision Making

Let us first formally define the general decision problem. Starting from an initial state b, we measure the value of an action a, using an objective function $J(b, a)$. This function can measure either the expected value of the posterior state, or the value of the action itself, or a combination of these values. Given a set of candidate actions $\mathscr{A}$, we look for the action which maximizes this function. Thus, the problem can be formulated as

$$a^* = \underset{a \in \mathscr{A}}{\operatorname{argmax}}\, J(b, a). \tag{1}$$

According to this basic problem definition, we are required to calculate the value of the objective function per action. Yet, in many real world problems, especially when discussing a discretization of a continuous action domain, as common to do in navigation, or object manipulation, the set of candidate actions can grow very large. Also, in some cases, such as decision making under uncertainty and belief space planning in high dimensional states, calculation of this objective function is computationally expensive, as it requires calculating information measures (e.g. entropy or information gain) and by such, handling high dimensional covariance or information matrices.

When planning in the belief space, we assume the states, actions, and observations are stochastic. Hence, to describe a sequential planning process (POMDP) at time k, we use the posterior distribution over the state vector X_k

$$b_k \doteq \mathbb{P}(X_k \mid a_{0:k-1}, z_{0:k}) \sim \mathcal{N}(\hat{X}_k, \Lambda_k^{-1}), \tag{2}$$

where $a_{0:k-1} \doteq \{a_0, \dots, a_{k-1}\}$ and $z_{0:k} \doteq \{z_0, \dots, z_k\}$ are the actions and observations taken until time k, respectively. To describe such a distribution we use the term *belief*. The transition and observation models can be described probabilistically as $\mathbb{P}(x_{k+1} \mid x_k, a_k)$ and $\mathbb{P}(z_k \mid X_k)$, respectively. We assume that both models and the beliefs are *normally* distributed.

When applying a new action a_k over belief b_k (and then taking a new observation), its information matrix Λ_k is updated according to the corresponding transition and observation models. Due to the additive quality of the information form, such update can be compactly described as

$$\Lambda_{k+1} = \Lambda_k + A_k^T A_k, \tag{3}$$

where the collective Jacobian A_k encapsulates all the new information in a single term. This Jacobian can hold information (or new factors, in a factor graph representation) regarding the action, or the sequence of actions, for the nonmyopic scenario, and the following observations. Representation of an update in such form can be done for every candidate action (for more details see [1]). We will use this representation throughout the paper.

In this context, we wish to minimize the uncertainty in the future belief, or equivalently maximize the information gain in the posterior belief. Thus, we define the following objective function (although other objective definitions may also be applicable):

$$J(b_k, a_k) \doteq |\Lambda_{k+1}| = |\Lambda_k + A_k^T A_k|, \tag{4}$$

Calculation of the objective values requires computing the determinant of the posterior information for each candidate action. We recall that a single determinant evaluation of a matrix of size $n \times n$ is valued at $O(n^3)$, for the general case. Although this computation can be done efficiently for sparse matrices (a quality which is often utilized by state of the art code frameworks), it is still considerable when discussing high dimensional states.

The variables in the state vector are defined with relation to the discussed problem, e.g. for a sensor deployment problem, it can include the measured values from the placed sensors; for a SLAM problem it can include previous poses of the agent and the locations of observed landmarks. In that case, the dimension of the state vector and information matrix can rapidly grow such that revenue calculation becomes challenging to do in real time.

2.2 Goal

As demonstrated, solving the decision problem may encapsulates a high computational cost, caused by an expensive objective function and/or numerous candidate actions. We wish to select the best candidate action without the explicit and exhaustive calculation of all of the objective values. Of course, we want to minimize any effect over the action selection, in order to maintain the original quality of solution, as our focus is on reducing the computational cost of the problem.

3 Approach

3.1 Action Elimination

As stated, solution of the decision making problem requires calculation of the revenue function for each candidate action, given an initial state b. Alternatively, say we were to calculate the revenues of the same set of candidate actions, while considering a different initial state b_s (s is used to mark a simplified or sparse version of b). Considering these two initial states, if all the respective pairs of revenues were equal, then the selected candidate action, the one with the highest revenue, would be the same in both cases. To describe this tight correlation, we say that b and b_s are *action consistent*. When this situation is guaranteed, we could equivalently examine the actions using either of these states. Formally:

Definition 1 Consider a set of actions $\mathscr{A}$ and an objective function $J(state, action)$ (these notations will also be relevant for the definitions to follow). Two states b, b_s are ***action consistent***, in relation to J and $\mathscr{A}$, and marked $b \simeq b_s$, if the following applies $\forall a_i, a_j \in \mathscr{A}$:

$$J(b, a_i) < J(b, a_j) \iff J(b_s, a_i) < J(b_s, a_j) \tag{5}$$

In other cases, when there is a difference in the objective values of some actions, the action selection might be affected. To examine this situation, we suggest the *simplification offset* as a "metric" between such states, in the context of decision making. Naturally, when the offset between the states is zero, they are action consistent.

Definition 2 The ***simplification offset of an action*** $a \in \mathscr{A}$ is defined as:

$$\gamma(b, b_s, a) \doteq |J(b, a) - J(b_s, a)| \tag{6}$$

The ***simplification offset between the two states*** is defined as:

$$\gamma(b, b_s) \doteq \max_{a \in \mathscr{A}} \gamma(b, b_s, a) \tag{7}$$

Please note, Definitions 1 and 2, were initially definied in our previous work [1], but due to their extensive use throughout the paper, are brought here again, in order to allow fluent reading. These definitions are demonstrated in Fig. 1.

While it appears that calculating this offset essentially requires calculation of all objective values, and thus actually solving the problem, in many cases it can still be bounded without actually calculating them, but by considering the structure of the state, actions, and objective. An example of calculating such a bound is to appear ahead.

If b_s represents an approximation or simplification of the original state b, then for each action a, we can say that $J(b_s, a)$ represents an approximation of the exact value of this action. By using a simplified, or sparse b_s, the approximated values should be easier to calculate than the real ones. For intuition, when the offset between states is small enough, then the action selection is "similar" in both cases. Therefore, since our goal is to improve the computational complexity of the problem, with minimal effect on the action selection, utilizing such an approximation may be worthwhile.

When approximating the objective using b_s (which is not action consistent), although they are expected to be faster to calculate than the values, we might not achieve the highest potential value, due to selection of a sub optimal action. To make any valuable conclusion, we should seek a formal connection between the approximated objective values and the exact ones.

Theorem 1 (Elimination Interval) *For two states b, b_s, and an objective function $J(state, action)$:*

$$\boxed{J(b_s, a_s^*) - 2 \cdot \gamma(b, b_s) \quad \le \quad J(b_s, a^*) \quad \le \quad J(b_s, a_s^*)} \quad where \quad \begin{aligned} a^* &= \underset{a}{argmax}\, J(b, a) \\ a_s^* &= \underset{a}{argmax}\, J(b_s, a), \end{aligned} \tag{8}$$

Proof According to [2], given a state and its approximation, the error in the objective value can be bounded in the following manner:

$$\left| J(b, a^*) - J(b, a_s^*) \right| \le 2 \cdot \gamma(b, b_s). \tag{9}$$

Since this statement is true for every selection of b, b_s, it is also true when switching between the two beliefs, as if b is the approximation of b_s, giving the following equation:

$$\left| J(b_s, a_s^*) - J(b_s, a^*) \right| \le 2 \cdot \gamma(b, b_s). \tag{10}$$

By definition a_s^* yields higher value than any other action, considering b_s as the prior. Hence, we can omit the absolute value,

$$0 \leq J(b_s, a_s^*) - J(b_s, a^*) \leq 2 \cdot \gamma(b, b_s). \tag{11}$$

Finally, we can rearrange the equation into

$$J(b_s, a_s^*) - 2 \cdot \gamma(b, b_s) \leq J(b_s, a^*) \leq J(b_s, a_s^*). \tag{12}$$

□

Note that despite its short proof, Theorem 1 is not trivial. It transforms a statement over the exact values (Eq. 9), which are unknown to us, into a statement based solely on the approximation, and allows us to infer about the desired best candidate action.

This important conclusion allows us to learn about the desired best candidate action a^*, without having to know what it is, nor to exactly calculate the objective for the actions, but only using the easy to calculate approximated values, and the revenue offset of the approximation (which, as previously stated, can often be bounded without having to solve the problem, and such an example appears ahead). Eq. 8 defines an interval of size $2 \cdot \gamma(b, b_s)$, in which the approximated value of the real best candidate action $J(b_s, a^*)$ must lie. Thus, without having to know the exact values, we can eliminate actions which cannot possibly be the best candidate a^*, since the approximation is not in this range.

Of course, when using a bound of the offset, and not the real value, the interval grows larger in size. Hence, when bounding the offset, we should balance between a tight bound, and keeping it easy to calculate. We recall again that we aspire to use a simplified approximated state b_s, such that by using it, the approximated objective values are easy to calculate. Overall, when an approximation of the initial state is given, considering its offset from the original state can be calculated or bounded, we can easily eliminate unfit actions, resulting in a smaller subset of candidates, to be used for solving the problem. This action elimination method is summarized and demonstrated in Fig. 2.

Again, this discussion is not limited to a specific class of problems, not even decision making under uncertainty, and is general to any decision problem. Finding such b_s can be done in any way for which the offset from the real state $\gamma(b, b_s)$ can be calculated. Overall, an action elimination process can either be applied once before the action selection, or recursively, when the approximation is scalable, as described in the next section.

3.2 Recursive Elimination

As a rule of thumb, we consider that calculating a rough approximation of the objective (high offset) is faster than calculating a more refined one (smaller offset). For example, a factor graph can be approximated using node sparsification. Removal of more nodes from it is expected to increase the offset from the original graph, as the approximation becomes more "rough". Hence, given state approximations of different scale (with different offsets from the original), one can balance between investing in minimizing the offset and by such the size of the interval, potentially leading to elimination of more actions, and an easier calculation of the approximation, with a more lenient elimination.

Recursively refining the approximation (decreasing offset) can be the most beneficial. After each iteration of eliminating actions, we can try to reduce the size of the interval, using a more refined approximation, in order to perform another elimination iteration. Each iteration leaves us with a smaller subset of candidate actions to select from. This way actions which are clearly non beneficial are eliminated first, with minimal computational investment, and the refined approximation is only used for the top candidates. Ultimately, all the candidates but one would be eliminated, with convergence of the size of the interval into zero, and the approximated best candidate into the real one. Alternatively, considering the time constraints, if more than one candidate still remains, we can either solve the problem exactly for those remaining actions, or return the best choice based on the current approximation. This recursive method is summarized in the form of "anytime" Algorithm 1.

Note that whenever breaking on this anytime algorithm, the loss in potential value, induced by selecting the approximated best action, is still ensured to be bounded (according to Eq. 9).

Also, every iteration requires recalculation of the values according to the new scale of approximation. This can be optimized if the approximation method allows to update and refine them, instead of calculating the values all over again. We still assume that applying the elimination can be done fast enough, such that the saved time in objective calculation is greater than the run time of the elimination itself. In our experiments we show that this claim is indeed feasible. We note, however, that if the convergence of the elimination interval is very slow, then the overall computational cost may increase, due to the recalculation of the objective. Therefore, when applying sequential elimination, the refinement of the approximations should be considered in accordance to the problem, and the magnitude of the objective cost.

Algorithm 1: Recursive action elimination—an “anytime” algorithm

```
Inputs:
    Initial state $b$
    A set of candidate actions $\mathscr{A}$
Output:
    Best candidate action $a^*$
1  Calculate an approximation of the initial state $b_s$
2  Calculate an approximation of the objective values using $b_s$ for each candidate action in $\mathscr{A}$
3  Calculate or bound the simplification offset $\gamma(b, b_s)$
4  Eliminate actions for which the approximated value is not in the interval from Theorem 1.
   For the remaining subset of candidate actions:
5  if only a single action $a_s^*$ is in the interval then
6      return $a_s^*$ (the real best candidate action)
7  else if timeout then
8      return $a_s^*$ (the best current choice)
9  else
10     Refine the approximation $b_s$ and go to 3
11     or
12     Solve the problem exactly, for the remaining candidate actions
```

4 Practical Application

4.1 Decision Making Under Uncertainty

As stated earlier, the elimination process is general and independent of the problem structure. Here we extend the discussion to the highly relevant case of decision making under uncertainty and belief space planning, and specify practical tools for applying the method.

When discussing decision making under uncertainty, the states, known as beliefs, are stochastic, and described using a state vector and an information (or covariance) matrix. Thus, state approximation here translates to sparsification of the the underlying information matrices, as sparse matrices can be efficiently processed. In [2] we describe a scalable sparsification algorithm (brought here as Algorithm 2), which yields a sparse approximation of an information matrix, for which the offset can be bounded. Since the algorithm is scalable, and allows creating approximations of different degrees, it is an appropriate match to the recursive action elimination process.

According to the algorithm, we are able to select which variables, or rows, to sparsify from the matrix (the indices of the sparsified rows are collected to a set of indices $\mathscr{S}$). Sparsification of each row contributes somewhat to the offset from the original belief, with variables which are uninvolved in the current decision (see [1]) having zero contribution. One can freely select which and how many rows to sparsify, and by such control the scale of the sparsification, and its offset from the original belief.

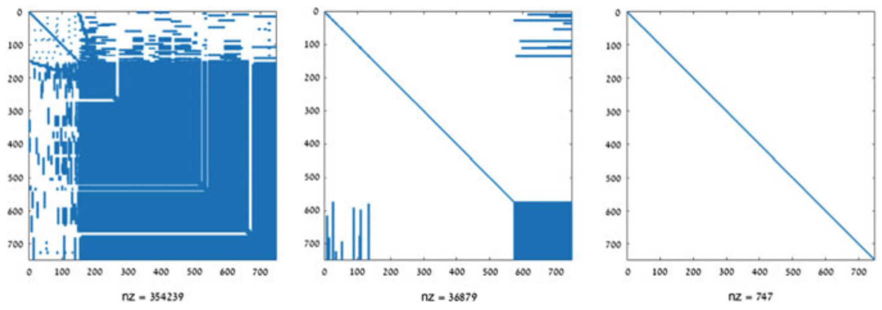

Fig. 3 Sparsification example using Algorithm 2. On the left—an information matrix taken from a SLAM process; In the middle—a sparse version, after sparsifying all the uninvolved variables; On the right—a diagonal approximation, after sparsifying all variables. Note the significant difference in the number of non zero elements

Recall the objective function for this problem includes an evaluation of a determinant (Eq. 4). The order of magnitude of calculation of a determinant of a general matrix of size $n \times n$ is $O(n^3)$, while for a fully sparsified matrix, a diagonal one, it is linear. Meaning, the calculation of the approximated objective can be done much faster, making the elimination process here very cost-effective. This progression also matches the intuition we previously got while discussing a recursive elimination. A sparser approximation leads to a faster calculation, yet a bigger offset and a more lenient elimination (Fig. 3).

The usage of the sparsification Algorithm 2 for the recursive elimination Algorithm 1, is done by selecting a growing number of rows to sparsify in each iteration. Otherwise, for a single elimination process, one can use any row selection $\mathscr{S}$ for the sparsification.

```
Algorithm 2: Scalable sparsification of a belief
    Inputs:
        A belief $b \sim \mathcal{N}(x, \Lambda^{-1})$
        A set $\mathscr{S}$ of row indexes to sparsify
    Output:
        A sparse approximated belief $b_s$
1 Use Cholesky decomposition to find $R$ such that $\Lambda = R^T R$
2 Calculate $M = R^{-1}$
3 Generate a sparse $M_s$ according to:
      $(M_s)_{ij} = \begin{cases} 0 & i \in \mathscr{S} \text{ and } i \neq j \\ M_{ij} & \text{else} \end{cases}$
4 Calculate $R_s = M_s^{-1}$
5 Calculate $\Lambda_s = R_s^T R_s$
6 return $b_s \sim \mathcal{N}(x, \Lambda_s^{-1})$
```

According to Definition 2 and Eq. 3, the offset of an action a in this case is

$$\gamma(b, b_s, a) \doteq \left| |\Lambda + AA^T| - |\Lambda_s + AA^T| \right| . \tag{13}$$

For an approximation generated using Algorithm 2, given a specific selection of rows $\mathscr{S}$, and considering a *single-row collective Jacobian* A of an action a, the offset can be developed into (see [2])

$$\gamma(b, b_s, a) = \frac{1}{|\Sigma|} \cdot \left| \sum_{i=1}^{n} \sum_{j=1}^{n} A_i \cdot (\Sigma - \Sigma_s)_{ij} \cdot A_j \right| \quad \text{where} \quad \Lambda = \Sigma^{-1},\ \Lambda_s = \Sigma_s^{-1} \tag{14}$$

Again, the overall offset of the approximation is the maximal offset among all actions. Instead of explicitly calculating the offset of all the actions, we reduce the cost of calculating $\gamma(b, b_s)$ by settling on a bound over this value. This can be done by using an upper bound over the elements of A:

$$\gamma(b, b_s, a) = \frac{1}{|\Sigma|} \cdot \sum_{i,j} \left|A_i A_j\right| \cdot \left|(\Sigma - \Sigma_s)_{ij}\right| \leq \frac{\alpha}{|\Sigma|} \cdot \sum_{i,j} \left|(\Sigma - \Sigma_s)_{ij}\right| , \tag{15}$$

where the scalar α is an upper bound over the coefficients $\left|A_i A_j\right|$, for $1 \leq i, j \leq n$. When α bounds the coefficients $\forall a \in \mathscr{A}$, this bound over the offset becomes independent of a specific action, and only a single value needs to be calculated (instead of calculating the offset per action, and then choosing the maximal value). Another "middle solution" could be finding similar upper bounds for subsets of similar actions, and then taking the maximal value among the bounds of each of these subsets. We denote that despite what is implied from its definition, we were able to bound the offset without actually having to calculate the objective values of the actions, as previously promised.

In the more general scenario, where A is a multi-row matrix, representing a non-myopic action sequence, the offset can be bounded using various determinant bounds. This scenario is left to be investigated as a part of our future research.

4.2 Simulation

In this simulation, we demonstrate the application of the method on a sensor deployment problem, based on the previously presented analysis for decision making under uncertainty. We wish to place 10 sensors on the junctions of a 5×5 grid, in order to measure its temperature in the most informative way. This is a demonstration of planning in the belief space, hence the grid states are stochastic, and the information matrix (sized 25×25) represents the uncertainty of the current temperature measurement in the 25 junctions of the grid. At each step we greedily choose a position

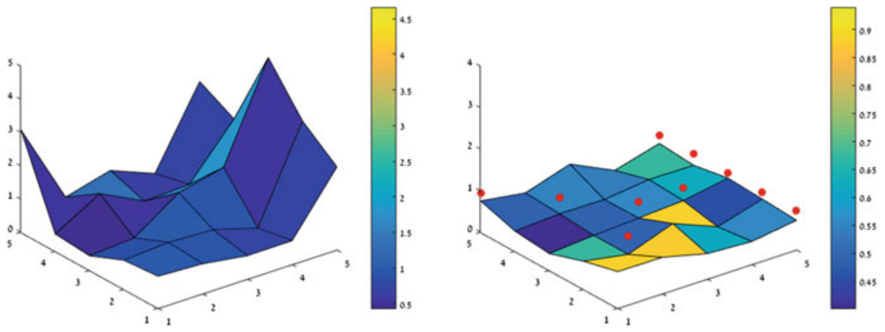

Fig. 4 Uncertainty levels in the temperature grid. Left—initial state, right—after sensor deployment. Higher values indicated higher uncertainty levels in that junction

from those available, and place a sensor on it (meaning, taking a measurement of the temperature from that position), as shown in Fig. 4. The problem is initialized with random uncertainty over the grid, leading to a dense prior information matrix.

At each iteration, we performed an elimination of unfit candidates, according to the previously described approach. The approximated objective values were calculated using a sparse approximation of the information matrix, generated using Algorithm 2, with $\mathscr{S}$ containing all rows, leaving only a diagonal matrix. Here the collective Jacobians of the actions contain only 0's and 1's, indicating placement of a sensor,

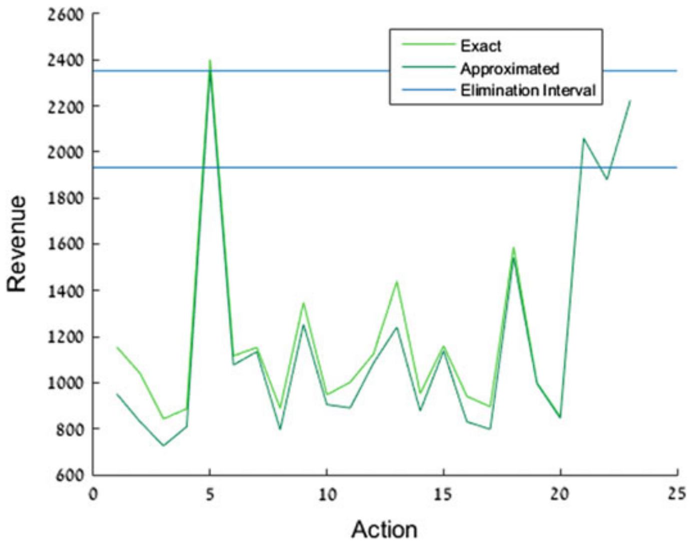

Fig. 5 A single elimination process. The approximated objective values (revenues) and the (unknown) exact values are shown in turquoise and green, respectively. Actions for which the approximated value is not between the blue lines can be eliminated

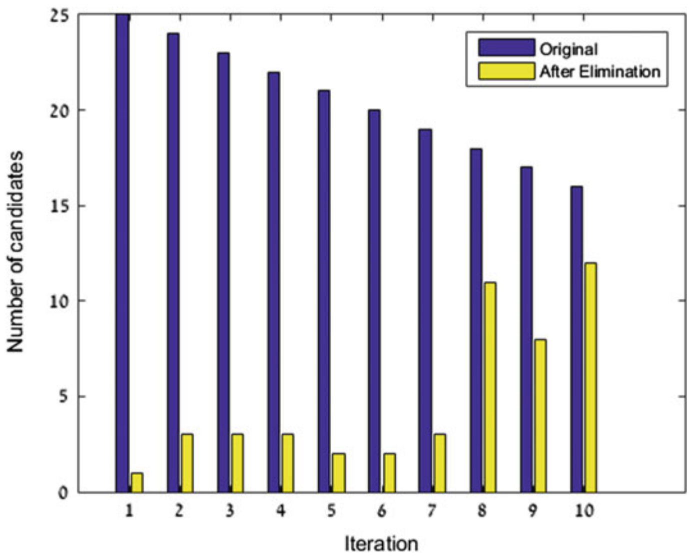

Fig. 6 Number of candidates action to examine, before and after the elimination process

and therefore in order to calculate the elimination interval (Eq. 8), we were able to calculate a single bound of the overall offset using Eq. 15.

Figure 5 describes a single elimination process from one of the iterations. The approximated objective values (revenues) and the (unknown) exact ones are shown in turquoise and green, respectively. The horizontal blue lines stand for the limits of the elimination interval—actions for which the approximated value is not between these lines, can be eliminated. In that iteration, only 3 actions passed the elimination. We could then continue and solve the decision problem to this minimal subset of candidates, without having to calculate the exact value of the eliminated ones.

Figure 6 shows the number of remaining candidate actions after the elimination, in each iteration, in comparison to the original number of candidates. In a few cases, all candidates besides one were eliminated, hence leaving us with the best candidate, without even calculating any value exactly. In the final iterations, the most informative sensors were already placed, and the remaining candidates yield similar value (no dominant actions), therefore more candidates passed the elimination.

5 Discussion and Conclusions

This paper introduced an approach to reduce the computational complexity of the basic decision problem. Instead of exhaustively calculating the exact objective values of the candidate actions, in order to select the best one, we can use a rough approximation of these values to quickly eliminate unfit candidates.

The main concept of this paper is not tied to a specific problem. Yet, since this approach is highly relevant to problems in which the action domain is large, and the objective is hard to calculate, we extended the discussion to the challenging case of decision making under uncertainty and belief space planning. There we used a dedicated scalable approximation algorithm, to form a practical set of tools alongside the elimination algorithm. We finally demonstrated the benefits of using the method while solving a sensor deployment problem.

This work is a conceptual introduction to the general case, and acts as a good basis to many future research directions. The simple simulation is brought in order to demonstrate how, in practice, the approach can be used, and to indicate its possible benefits. Applications to advanced scenarios require a thorough examination. Future extensions include, for example, in the context of belief space planning: setting bounds over multi-row Jacobians, to support non myopic cases; upgrading the sparsification algorithm, such that it becomes updatable—sparing the recalculation of the objective values in each iteration, which would dramatically improve the performance; or creating an entirely new sparsification method. Similar analysis can be developed to other types of problems as well.

A subsequent goal of this paper is to break the current nativity in approaching decision problems. To the best of our knowledge, our continuous work on the subject is the first attempt to analyze and directly improve the computational complexity of the fundamental problem. With this work we also wish to encourage the robotics and AI community to extend its focus on the algorithmic basis of autonomous systems, and specifically towards efficient decision making and planning. Despite the growth in academic research in this field, a great portion of recent work has been on adjusting specific properties in order to improve or upgrade existing methods. This paper proves that tackling the basics can be fruitful, and have a wider impact.

References

1. Elimelech, K., Indelman, V.: Consistent sparsification for efficient decision making under uncertainty in high dimensional state spaces. In: IEEE International Conference on Robotics and Automation (ICRA) (2017)
2. Elimelech, K., Indelman, V.: Scalable sparsification for efficient decision making under uncertainty in high dimensional state spaces. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2017)
3. Indelman, V.: No correlations involved: decision making under uncertainty in a conservative sparse information space. IEEE Robot. Autom. Lett. (RA-L) **1**(1), 407–414 (2016)
4. He, R., Prentice, S., Roy, N.: Planning in information space for a quadrotor helicopter in a gps-denied environment. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 1814–1820 (2008)
5. Huang, G., Kaess, M., Leonard, J.J.: Consistent sparsification for graph optimization. In: Proceedings of the European Conference on Mobile Robots (ECMR), pp. 150–157 (2012)
6. Carlevaris-Bianco, N., Kaess, M., Eustice, R.M.: Generic node removal for factor-graph SLAM. IEEE Trans. Robot. **30**(6), 1371–1385 (2014)
7. Roy, N., Gordon, G., Thrun, S.: Finding approximate pomdp solutions through belief compression. J. Artif. Intell. Res. **23**, 1–40 (2005)

8. Patil, S., Kahn, G., Laskey, M., Schulman, J., Goldberg, K., Abbeel, P.: Scaling up gaussian belief space planning through covariance-free trajectory optimization and automatic differentiation. In International Workshop on the Algorithmic Foundations of Robotics (2014)
9. Indelman, V., Carlone, L., Dellaert, F.: Planning in the continuous domain: a generalized belief space approach for autonomous navigation in unknown environments. Int. J. Robot. Res. **34**(7), 849–882 (2015)
10. Eustice, R., Walter, M., Leonard, J.: Sparse extended information filters: insights into sparsification. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3281–3288, Aug 2005
11. Prentice, S., Roy, N.: The belief roadmap: efficient planning in belief space by factoring the covariance. Int. J. Robot. Res. (2009)
12. Van Den Berg, J., Abbeel, P., Goldberg, K.: Lqg-mp: optimized path planning for robots with motion uncertainty and imperfect state information. Intl. J. Robot. Res. **30**(7), 895–913 (2011)
13. Krause, A., Singh, A., Guestrin, C.: Near-optimal sensor placements in gaussian processes: theory, efficient algorithms and empirical studies. J. Mach. Learn. Res. **9**, 235–284 (2008)
14. Platt, R., Tedrake, R., Kaelbling, L.P., Lozano-Pérez, T.: Belief space planning assuming maximum likelihood observations. In: Robotics: Science and Systems (RSS), Zaragoza, Spain, pp. 587–593 (2010)
15. Nakhost, H., Müller, M.: Action elimination and plan neighborhood graph search: two algorithms for plan improvement. In: Twentieth International Conference on Automated Planning and Scheduling (ICAPS), pp. 121–128 (2010)
16. Rosman, B., Ramamoorthy, S.: What good are actions? Accelerating learning using learned action priors. In: 2012 IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL), pp. 1–6. IEEE (2012)
17. Sherstov, A., Stone, P.: Action-space knowledge transfer in mdps: Formalism, suboptimality bounds, and algorithms. In: Conference on Learning Theory (2005)
18. Abel, D., Hershkowitz, D.E., Barth-Maron, G., Brawner, S., O'Farrell, K., MacGlashan, J., Tellex, S.: Goal-based action priors. In: Twenty-Fifth International Conference on Automated Planning and Scheduling(ICAPS), pp. 306–314 (2015)
19. Kuter, U., Hu, J.: Computing and using lower and upper bounds for action elimination in mdp planning. In: International Symposium on Abstraction, Reformulation, and Approximation, pp. 243–257. Springer, Heidelberg (2007)
20. Even-Dar, E., Mannor, S., Mansour, Y.: Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. J. Mach. Learn. Res. **7**(Jun), 1079–1105 (2006)
21. Bry, A., Roy, N.: Rapidly-exploring random belief trees for motion planning under uncertainty. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 723–730 (2011)