Notes on Parametrization vs Projection in Strategy Optimization

Connor Hughes

May 6^{th}, 2022

Single-Agent, Omniscient Intruder Strategy Optimization

Background:

See the paper "Stochastic Strategies for Robotic Surveillance as Stackelberg Games" [1] for the problem introduction. We have the following recursive formula for computing the capture probability matrix (F_{cap}):

$$\begin{aligned} F_{cap} &= \sum_{k=1}^{\tau} F_k \\ F_1 &= P \\ F_{k+1} &= P(F_k - diag(F_k)), \;\; 1 \leq k \leq \tau - 1 \end{aligned}$$

where P is the transition probability matrix that describes a Markov chain-based stochastic surveillance strategy, and τ is the intruder's attack duration.

We also have the following constraints, which ensure that the transition probabilities represent a valid probability distribution at each node (i and ii) and agree with the structure of the environment graph (iii):

$$P\,\mathbb{1}_n = \mathbb{1}_n \tag{i}$$
$$p_{ij} \geq 0, \;\; \forall (i,j) \in \mathcal{E} \tag{ii}$$
$$p_{ij} = 0, \;\; \forall (i,j) \notin \mathcal{E} \tag{iii}$$

where n is the number of nodes in the environment graph. Let us denote the space of matrices satisfying the above constraints by C.

We define the "minimum capture probability" (MCP) for a given strategy to be:

$$\mu = \min_{i,j} F_{cap}(i,j), \;\; \text{where } i,j \in \{1,...,n\}$$

For convenience, we denote the mapping from a given strategy $P \in C$ to the corresponding capture probability matrix by $f_{cap} : P \to F_{cap}$, and the mapping from P to the MCP by $f_{mcp} : P \to \mu$.

Projection vs Parametrization:

Maximizing μ via gradient-based algorithms has been attempted with some early success by two approaches. The first entails computing the gradient of f_{mcp}, updating P by a gradient-ascent step, then projecting the updated P matrix onto the nearest point (in an ℓ^2-sense) within the constraint set C.

The second is to parametrize the strategy P by defining an onto mapping $g : \mathbb{R}^{n \times n} \to C$. This enables us to then combine both the constraint enforcement and MCP computation within a new loss function $f \circ g : Q \to \mu$, where $Q \in \mathbb{R}^{n \times n}$. An example of such a parametrization and loss function is given below:

$$\begin{aligned} P &= g(Q) = diag(1/((ReLU(Q \odot A)\,\mathbb{1}_n))(ReLU(Q \odot A)) \\ \mu &= f \circ g(Q) \end{aligned}$$

where A is the binary adjacency matrix of the environment graph, $\odot$ represents the Hadamard product, and $ReLU$ is a Rectified Linear Unit which acts component-wise.

The mapping g in the example above is simply a composition of the following three operations performed on the parameter Q:

1) Hadamard product with the environment graph's binary adjacency matrix A, to enforce constraint (iii)

2) Component-wise ReLU, to enforce constraint (ii).

3) Scaling each row by its ℓ^1-norm, to enforce constraint (i).

Parametrization for Dimension Reduction:

The approach described above can be extended in order to leverage additional information about the surveillance problem and the graph structure to reduce the dimension of the parametrization, leading to more computationally-efficient optimization which generates more effective strategies. For example, the authors in [1] identified that setting the agent's probability of leaving a leaf node to 1 is a dominant strategy.

With the parametrization approach, we can enforce this dominant strategy by extending the mapping g, as follows. We identify the leaf nodes from the binary adjacency matrix A. Then, for each row i corresponding to a leaf node in the matrix $Q \in \mathbb{R}^{n \times n}$, we set the j^{th} entry of the row equal to 1 if $A(i,j) = 1$ and $i \neq j$, and set all other entries to zero. This row is now "fixed" with regard to the optimization process. Now, assuming Q contains m rows which do *not* correspond to a leaf node, we define the parameter $U \in \mathbb{R}^{m \times n}$. We add an additional step to the mapping g which precedes the 3 operations described above. This step entails setting the k^{th} non-leaf-node row of Q equal to the k^{th} row of U. Then, we apply the same 3 operations which previously comprised g, and our new mapping can be denoted $g' : \mathbb{R}^{m \times n} \to C$. The "leaf-node rows" in Q will be unchanged by the Hadamard product with A, the component-wise ReLU, and the ℓ^1-norm scaling. Thus, the matrix P will include the dominant strategy on leaf nodes as desired. As long as the new mapping g' is implemented in a manner that still permits autodifferentiation, we can run the optimizer just as before and have now reduced the dimension of the parameter space by $(n - m) \times n$. Note that we could also enforce this dominant strategy with a projection-based approach, though it does not reduce the dimension of the problem and we will not discuss the implementation details here.

For another example, we can see by symmetry arguments that, in the case of the star graph, the optimal strategy should have equal transition probabilities from the center node to each leaf node. Suppose these symmetry arguments hold more generally for any situation wherein multiple leaf nodes share the same neighbor. In that case, we can enforce the additional constraint that the transition probabilities from the shared neighbor to each of the leaves should be equal. This similarly enables a reduction in the dimension of the parameter space. We can define a new mapping h which takes a set of real numbers and maps each element of the set to an entry of U. If we are not reducing the dimension of the parameter space, this mapping simply arranges a set of $m \cdot n$ real numbers into an $m \times n$ matrix, and does not add value. However, composing this mapping with g' will prove useful for dimension reduction here, as follows. First, we can identify the leaf nodes from the binary adjacency matrix A, as well as the neighbors for each of those leaf nodes. Then, for each row i in U corresponding to a node which neighbors one or more leaf nodes, we identify the entries j within the row which correspond to the leaf nodes. The requirement that these entries are equal means that we can represent them with a single scalar parameter. Thus, we can modify the mapping h so that it maps a single real number to each of these entries within U. Thus, by defining $g'' = g' \circ h$, and using $f \circ g''$ as our loss function, we are able to reduce the dimension of the parameter space by [number of leaf nodes] - [number of leaf node neighbors].

References

[1] X. Duan, D. Paccagnan, and F. Bullo, "Stochastic strategies for robotic surveillance as stackelberg games," *IEEE Transactions on Control of Network Systems*, vol. 8, no. 2, pp. 769–780, 2021.