Removing Redundancies for Faster Inference

Arne Decadt ARNEDECADT@UGENT.BE
Foundations Lab for imprecise probabilities, Ghent University, Belgium

When a decision-making model is complex, it can also be slow and clunky. That's why we will look at a new technique for removing redundancies from a model, making it faster and easier to use. The particular model here is sets desirable gambles, which is a general framework to model uncertainty [4]. Suppose we have a finite set $\mathcal{X}$ containing all possible outcomes of an experiment, then gambles are maps on $\mathcal{X}$ that map these possible outcomes to a real-valued utility scale. They form a real vector space $\mathbb{R}^{\mathcal{X}}$.

Given assessment $A \subseteq \mathbb{R}^{\mathcal{X}}$ containing gambles that we assume to be desirable, its natural extension is the least committal coherent extension, if it exists. Let $\mathbb{R}^{\mathcal{X}}_{\geq 0}$ denote gambles with non-negative components, 0 the gamble that is zero for every outcome and $\mathrm{posi}(A) \coloneqq \left\{\sum_{k=1}^{n} \lambda_k f_k \colon \lambda_k > 0 \wedge f_k \in A \wedge n \in \mathbb{N}\right\}$ the positive hull operator. Then the natural extension $\mathcal{E}(A)$ of A is found (or in [4] defined) as a positive hull $\mathcal{E}(A) = \mathrm{posi}(A \cup (\mathbb{R}^{\mathcal{X}}_{\geq 0} \setminus 0))$ and exists if this doesn't contain 0. A common way of decision-making and the one that I will use here is checking if a given gamble is in the natural extension.

We look at the case where A is finite. The most direct approach to do this is checking whether a given gamble is greater than or equal to a positive linear combination of a finite number of gambles and is a linear feasibility problem that can be solved by linear programming [3]. Let $\mathbf{A}$ be the matrix composed by the concatenation of the gambles of A as column vectors, λ be an unknown vector with $|A|$ components and g a gamble for which we want to check if it is in the natural extension. Then we need to check whether the following program has a solution: $\text{find} \quad \lambda \quad \text{subject to} \quad \mathbf{A}\lambda \leq g, \lambda \geq 1.$ A less direct approach that works for low dimensions $d = |\mathcal{X}|$ is to compute the credal set, which amounts to finding the dual polytope, and check whether lower expectation is non-negative. This approach becomes more computationally expensive for higher dimensions as the number of extreme probability mass functions given k desirable gambles can be as many as $\binom{k - \lceil d/2 \rceil}{\lfloor d/2 \rfloor} + \binom{k - \lceil d/2 \rceil - 1}{\lfloor d/2 \rfloor - 1}$, which grows exponentially in d. Computer evidence with random polytopes suggests that for high dimension one attains this bound with high probability [2, p. 394-395].

So suppose we use the direct approach and have to check for multiple gambles if they are in the natural extension. Then it can be advantageous to remove redundant gambles from A and find another smaller gamble set A' such that $\mathcal{E}(A') = \mathcal{E}(A)$. This problem is essentially finding the extreme rays of $\mathcal{E}(A)$, for which algorithms are usually grouped with the convex hull problem. Here the same division of approaches can be made. Some approaches, e.g. QHULL, use the dual polytope but have problems in high dimensions because of the high number of facets [1]. Therefore many libraries, such as CDDLIB and POLYHEDRA.JL simply check for every $a \in A$ if $a \in \mathcal{E}(A \setminus \{a\})$ which amounts to $|A|$ linear programs.

For consistent sets of desirable gambles, i.e. for which $0 \notin \mathcal{E}(A)$, I have found that we can improve this method by first using linear programming to find a hyperplane that intersects all rays of set of desirable gambles. Such hyperplane can be found as all gambles g for which the inner product $\lambda^T g = 1$, where λ is the solution to the linear program $\text{find} \quad \lambda \quad \text{subject to} \quad \mathbf{A}^T \lambda \geq 1, \lambda \geq 1.$ Then on this hyperplane we can remove one of the coordinates and find the convex hull by $|A|$ linear programs on this lower-dimensional space to find the extreme rays. The last coordinate can easily be retrieved using the defining relation of the hyperplane $\lambda^T g = 1$. The price that we have to pay is that we have to add one equality constraint for the convex hull: that the the sum of the coefficients equals one. In computer experiments I have found that in practice this approach speeds up the calculations for high dimensions.

References

[1] David Avis and David Bremner. How good are convex hull algorithms? In *Proceedings of the eleventh annual symposium on Computational geometry*, pages 20–28, 1995.

[2] Martin Henk, Jürgen Richter-Gebert, and Günter M Ziegler. Basic properties of convex polytopes. In *Handbook of discrete and computational geometry*, pages 383–413. Chapman and Hall/CRC, 2017.

[3] Erik Quaeghebeur. The conestrip algorithm. In *Synergies of Soft Computing and Statistics for Intelligent Data Analysis*, pages 45–54. Springer, 2013.

[4] Erik Quaeghebeur. Desirability. *Introduction to imprecise probabilities*, pages 1–27, 2014.