Approximating the Core via Iterative Coalition Sampling

Ian Gemp
Google DeepMind
London, United Kingdom
imgemp@google.com

Marc Lanctot
Google DeepMind
Montreal, Canada
lanctot@google.com

Luke Marris
Google DeepMind
London, United Kingdom
marris@google.com

Yiran Mao
Google DeepMind
London, United Kingdom
yiranm@google.com

Edgar Duéñez-Guzmán
Google DeepMind
London, United Kingdom
duenez@google.com

Sarah Perrin
Google DeepMind
Paris, France
sarahperrin@google.com

Andras Gyorgy
Google DeepMind
London, United Kingdom
agyorgy@google.com

Romuald Elie
Google DeepMind
Paris, France
relie@google.com

Georgios Piliouras
Google DeepMind
London, United Kingdom
gpil@google.com

Michael Kaisers
Google DeepMind
Paris, France
mkaisers@google.com

Daniel Hennes
Google DeepMind
Zürich, Switzerland
hennes@google.com

Kalesha Bullard
Google DeepMind
London, United Kingdom
ksbullard@google.com

Kate Larson
Google DeepMind
Montreal, Canada
katelarson@google.com

Yoram Bachrach
Google DeepMind
London, United Kingdom
yorambac@google.com

ABSTRACT

The core is a central solution concept in cooperative game theory, defined as the set of feasible allocations or payments such that no subset of agents has incentive to break away and form their own subgroup or coalition. However, it has long been known that the core (and approximations, such as the least-core) are hard to compute. This limits our ability to analyze cooperative games in general, and to fully embrace cooperative game theory contributions in domains such as explainable AI (XAI), where the core can complement the Shapley values to identify influential features or instances supporting predictions by black-box models. We propose novel iterative algorithms for computing variants of the core, which avoid the computational bottleneck of many other approaches; namely solving large linear programs. As such, they scale better to very large problems as we demonstrate across different classes of cooperative games, including weighted voting games, induced subgraph games, and marginal contribution networks. We also explore our algorithms in the context of XAI, providing further evidence of the power of the core for such applications.

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

KEYWORDS

Cooperative Game Theory; Core; Explainable AI

ACM Reference Format:
Ian Gemp, Marc Lanctot, Luke Marris, Yiran Mao, Edgar Duéñez-Guzmán, Sarah Perrin, Andras Gyorgy, Romuald Elie, Georgios Piliouras, Michael Kaisers, Daniel Hennes, Kalesha Bullard, Kate Larson, and Yoram Bachrach. 2024. Approximating the Core via Iterative Coalition Sampling. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 10 pages.

1 INTRODUCTION

Coalitional game theory studies games played by self-interested players where incentives are aligned and fully-binding contracts are supported [19]. The utility achieved depends on which group, *i.e.*, which *coalition*, of players is formed to solve the game. The key problem that researchers have focused on is how this total utility should be divided among the individual members of the team. Several solution concepts have been proposed to address this question, with the core [38] and the Shapley value [63] being among the best known. The Shapley value quantifies the contribution of each individual player via the differences in value obtained over all possible subgroups which include and exclude that player. In contrast, the core is a rough cooperative analogue of the Nash equilibrium: a utility division that is stable, *i.e.*, no subset of players has incentive to deviate from the chosen coalition.

Cooperative game theory has found many applications, ranging from analyzing power in decision making bodies and political settings [64], through predicting how players may share team payoffs or costs [29, 49] to risk attribution [66]. One prominent application of cooperative game theory is in analyzing what drives the predictions of machine learning models. As more of our decisions are now informed by learned models, it is crucial to understand *why* our models output what they do, which is the main goal of explainable AI (XAI) [12]. The Shapley value has been adopted in the machine learning community to explain model predictions [50]: the features are treated as players, and the payoff to a coalition is the output of the model when training exclusively on the subset of input features in the coalition. The Shapley value then computes the marginal contribution of each feature to the output, offering additive explanations for the effect of each feature on prediction.

However, there are some limitations regarding Shapley values: it may produce counter-intuitive explanations [65], behavioral studies have shown the core can be more predictive of human payment divisions, and accurate estimation requires many samples [6, 51, 52][1]. As such, the core has been recently motivated as a possible alternative to Shapley values for XAI [69]. Unlike Shapley values, the core focuses on stability [19]: a payment division is *stable* if no player has incentive to deviate from the coalition and form a different one on their own. Consequently, core payments may better reflect each player's "market value" [23, 61].

A key barrier to applying cooperative game theory in practice is the high computational cost associated with calculating the various solution concepts, which may be hard to compute even for very restricted forms of cooperative games [19]. For some solutions, such as the Shapley value and other power indices, there exist efficient approximation algorithms [14, 18], e.g., an approximate form of the Shapley value can be computed using Monte Carlo sampling [6, 36, 52], which might explain its wide adoption.

In contrast, solutions based on the core have proved to be more elusive; there exist multiple approaches that can approximate or exactly solve for the core, but can only be applied to very restricted classes of coalitions games [5, 32, 45, 46]. Computing the core exactly requires solving a linear program (LP) with an exponential number of constraints (one per each possible subset of the agents). Alternatively, assuming access to incrementally sampled coalition values, bounds can be given for likely stable payments that relate to the core [10]. A recent breakthrough in approximating the core is a Monte Carlo algorithm that samples coalitions in this way, yielding probabilistic bounds on approximation quality [69]. This approach significantly expands the set of games for which one can tractably approximate the core, albeit it still requires solving LPs, where the size of the LP grows linearly in the number of samples.

Our contribution: In this paper, we introduce novel iterative algorithms for the core. None of our algorithms require solving a linear program: as such, they enjoy greater scaling potential to very large problems. We demonstrate this efficiency in practice across several classes of coalition games including weighted voting games [32], induced subgraph games [26], marginal contribution networks [42], and apply them to feature-importance and data-valuation problems arising in XAI [12]. See extended arXiv version [37] for Appendix and OpenSpiel[2] [48] for code.

[1]See Appendix B for a more comprehensive discussion of the differences between the Shapley value and the core.

2 BACKGROUND

The central concept in cooperative game theory is that of a *coalition*. Given a set of n agents, $I = \{1, 2, \dots, n\}$, a coalition is a subset of agents, $C \subseteq I$, where the *grand coalition* is the coalition containing all agents in I. A coalitional game, G, is further coupled with a characteristic function, $v : 2^I \to \mathbb{R}$ that assigns a real value to each coalition of agents, representing the total utility that the coalition of agents achieve together. We study transferable utility games, where $v(C)$ is interpreted as value that is to be shared and transferred between members of the coalition C. As is standard, $v(\emptyset) = 0$.

The characteristic function only defines the gains a coalition can achieve; it does not define how these gains are distributed among the coalition members. An *imputation* (also called payoff vector) $p = (p_1, \dots, p_n)$ is a division of the gains of the grand coalition I among the agents, where $p_i \geq 0$, such that $\sum_{i=1}^{n} p_i = v(I)$. We call p_i the payoff of agent i, and denote the payoff of a coalition C as $p(C) = \sum_{i \in C} p_i$.

A basic requirement for a good imputation is *individual rationality*, which states that for all agents $i \in C$, we have $p_i \geq v(\{i\})$; otherwise, some agent has incentives to work alone instead. Similarly, we say a coalition B *blocks* imputation $(p_1, \dots, p_n)$ if $p(B) < v(B)$, since the members of B can split from the original coalition, derive the gains of $v(B)$ by working together and then reallocate this *excess*, $v(B) - p(B) > 0$, to agents in B. This incentive to break away and form new coalitions leads to instability and has long been a focus of cooperative game theory. The most prominent solution concept is that of the core [38].

Definition 1. *The core of a coalitional game G is the set of all imputations that are not blocked by any coalition. That is it contains imputation $p = (p_1, \dots, p_n)$ if and only if for all $C \subseteq I$, $p(C) \geq v(C)$.*

Unfortunately, the core can be empty, meaning that for every imputation, there exists a blocking coalition. Thus, relaxations of the core are often studied by, for example, assuming that the gains made by forming a blocking coalition are small. The *ϵ-core* allows for such slight relaxations of the inequalities in the core definition.

Definition 2. *Given ϵ, the ϵ-core of coalitional game G is the set of all imputations such that for any $C \subseteq I$, $p(C) \geq v(C) - \epsilon$, i.e., ϵ-core $\overset{\text{def}}{=} \{(p_1, \dots, p_n) \mid p(C) \geq v(C) - \epsilon \ \forall \ C \subseteq I\}$.*

Clearly, for large enough values of ϵ, the ϵ-core is non-empty. Furthermore, if $\epsilon = 0$, the definition of the ϵ-core is equivalent to the core. A natural question is what is the smallest ϵ such that the ϵ-core is non-empty. Given a game G we consider the set $\{\epsilon |$ the ϵ-core of G is not empty$\}$. It is easy to see that this set is compact, and thus has a minimal element ϵ_{min}.

Definition 3. *Given coalitional game G, define $\epsilon_{\min} = \min\{\epsilon |$ ϵ-core of G is non-empty$\}$. The least core of G is the ϵ_{min}-core of G, and this value ϵ_{min} is called the Least-Core Value (LCV) of G.*

[2]github.com/google-deepmind/open_spiel/…/coalitional_games

3 APPROXIMATING THE CORE THROUGH ITERATIVE PAYOFF ADJUSTMENTS VIA COALITION SAMPLING

Earlier work has shown that given a game G, checking if the core is non-empty is NP-hard [26] and, furthermore, that computing the least-core exactly is impossible [11]. This opens up the question we address in this section; how to best approximate the least core.

The least-core can be formulated as a linear programming (LP) problem. Let c be a binary coalition-membership vector for coalition C such that $c_i = 1$ if $i \in C$ and 0 otherwise. When clear from the context we sometimes use c to represent coalition C. Furthermore, without loss of generality, assume that $v(I) = 1$, re-scaling v with a factor $\frac{1}{v(I)}$ if necessary. Then the least-core problem is to solve

$$\min_{p,\epsilon} \quad \epsilon \tag{1}$$

$$s.t. \quad v(c) - \epsilon - p^\top c \leq 0 \quad \forall\ c \tag{2}$$

$$\sum_{i=1}^{n} p_i = 1 \tag{3}$$

$$p_i \geq 0 \quad \forall\ i \in I. \tag{4}$$

The challenge is that the Constraint 2 relates to 2^n coalitions, making it infeasible to solve for games with many players. A recent method to approximating the least-core randomly samples some number of the constraints imposed on the coalitions and solves the resulting LP [69]. However, even when only using sub-sampled constraints, solving LPs is still time consuming.[3] We present alternative iterative algorithms that avoid the requirement of solving the LP, circumventing a significant computational bottleneck.

Our algorithms are presented in increasing complexity. Our first two algorithms, Iterative Projections (Section 3.1) and Stochastic Subgradient Descent (Section 3.2) take a given value of ϵ and seek an imputation in the ϵ-core (assuming the ϵ-core is non-empty). In order to compute the least-core, one may apply an outer loop that calls these methods so as to perform a binary search for the minimal value ϵ_{min} yielding a non-empty ϵ_{min}-core. Our Core Lagrangian method (Section 3.3) directly returns the least-core, including both the ϵ_{min} (least-core value), and an imputation in the ϵ_{min}-core. Our empirical results in Section 4 are reported for this algorithm alone.

3.1 The ϵ-Core via Iterative Projections

We observe that each Constraint 2 on p represents a half-space, namely a convex set. Assuming that there exists some p such that all constraints can be satisfied for a given ϵ, the corresponding constraint satisfaction problem is to find p such that Constraint 2 is satisfied. This is equivalent to solving a convex feasibility problem by finding a p on the $(n-1)$-dimensional simplex Δ that lies within the intersection of all these convex sets, a problem that can be solved via von Neumann-Halperin method of cyclic alternating projections [9, 13, 27]. We start with an initial guess for p and then cycle through the constraints (for each coalition, C), iteratively modifying the guess by projecting it onto the feasible set represented by each individual constraint. We observe that the projection, $\texttt{Proj}_{c,\epsilon}(p)$,

[3]Methods that guarantee polynomial runtime for solving LPs [44] have a runtime that is cubic in the number of parameters. An efficient implementation of the Simplex method [20] is much faster in practice, but is still time consuming for large LPs.

Algorithm 1 ϵ-Core via von Neumann-Halperin

```
Input: Number of iterations $T$
  $p_0 = \frac{1}{n}\mathbf{1}_n$
  for $t = 1 \leq T$ do
    Select coalition $C$ (cyclically) as binary vector $c$
    $d_c = \max(0, v(C) - \epsilon - p^\top c)$
    $p_t \leftarrow p_{t-1} + \frac{d_c}{|C|}c$
    $p_t \leftarrow \texttt{Proj}_\Delta(p_t)$
  end for
Output: $p_T$
```

of an imputation p onto a half-space $p^\top c \geq v(C) - \epsilon$ has a closed form solution that can be computed efficiently:

$$\texttt{Proj}_{c,\epsilon}(p) = \begin{cases} p & v(C) - \epsilon - p^\top c \leq 0 \\ p - (p^\top c + \epsilon - v(C))/||c||^2 c & \text{otherwise} \end{cases}$$
$$= p + \frac{d_c}{|c|}c \quad \text{where } |c| = \text{the size of the coalition} \tag{5}$$

and where $d_c = \max(0, v(C) - \epsilon - p^\top c)$ is the deficit. If $v(C) - \epsilon - p^\top c < 0$, players in coalition C are being paid *more* than they are actually contributing (where the payment is $\epsilon + p^\top c$). Intuitively, players are incentivized to remain in a coalition if $d_c = 0$, otherwise, the coalition is unstable. Algorithm 1 formalizes this method.

Theorem 1. *Let $v : 2^I \to \mathbb{R}$ be a characteristic function. Given an ϵ and assuming the ϵ-core exists, Algorithm 1 converges to an ϵ-core imputation asymptotically, i.e.: $\lim_{T\to\infty} p_T \to \epsilon$-core$(v)$.*

Proof. We can appeal directly to classical convergence results of the von Neumann-Halperin cyclic projections algorithm [9]. □

3.2 The ϵ-Core via Stochastic (Sub)Gradient Descent

While projecting an imputation p onto a hyperplane can be computed efficiently, cycling through all constraints is not, and it is also computationally expensive to update p on a batch of constraints as that involves projecting p into the intersection of a set of half-spaces. Instead, we propose a new approach that re-formulates the projection problem as an optimization problem and admits an efficient batch algorithm.

We define a loss function, $\ell_c(p, \epsilon)$ for each coalition C (represented by vector c):

$$\ell_c(p, \epsilon) = \frac{1}{2|c|}\big(\max(0, v(C) - \epsilon - p^\top c)\big)^2 = \frac{d_c^2}{2|c|}. \tag{6}$$

Observe that if $v(C) - \epsilon - p^\top c \leq 0$, $\ell_c = 0$, otherwise, we accrue some positive loss for not satisfying the constraint for coalition c. Furthermore, a valid negative (sub)gradient of ℓ_c with respect to p is exactly the update direction computed by the projection $\texttt{Proj}_{c,\epsilon}$ from equation (5):

$$-\nabla_p \ell_c = \frac{d_c}{|c|}c. \tag{7}$$

Therefore, if we run stochastic gradient descent (SGD) with learning rate equal to 1 on $\sum_c \ell_c(p)$, sampling one coalition at a time, we recover Algorithm 1. However, it is also possible to increase the minibatch size, sampling multiple coalitions at one time, as shown in Algorithm 2. Theorem 2 shows that if Algorithm 2 returns an imputation that is approximately in the ϵ-core, then it provably lies in the ϵ'-core, albeit with $\epsilon' > \epsilon$. The proof can be found in Appendix A.1.

Algorithm 2 ϵ-Core via SGD

```
Input: Number of iterations $T$
Input: Batch size $B$
Input: Step size schedule $\eta_t$
  $p = \frac{1}{n}\mathbf{1}_n$
  for $t = 1 \leq T$ do
    Sample batch $C_B$ containing $B$ coalitions
    Compute average gradient over batch: $\nabla_p = \frac{1}{B}\sum_{C \in C_B} \nabla_p \ell_c$
    $p \leftarrow p - \eta_t \nabla_p$
    $p \leftarrow \mathtt{Proj}_\Delta(p)$
  end for
  Sample $t^* \in \{0, \ldots, T\}$ according to $P(t^* = t) = \frac{\eta_t}{\sum_{t'} \eta_{t'}}$.
Output: $p_{t^*}$
```

Theorem 2. *If $\ell_c(\epsilon, p) \leq \gamma^2$ for all c, then p is in the $(\epsilon + \sqrt{2n}\gamma)$-core.*

Theorem 3. *Let $v : 2^I \to \mathbb{R}$ be a characteristic function. Given an ϵ and assuming the ϵ-core exists, Algorithm 2 converges to an ϵ-core imputation in expectation at a rate of $O(T^{-1/4})$.*

Proof. We can appeal directly to convergence rates of stochastic projected subgradient algorithms for convex optimization [22]. These rates prove Algorithm 2 converges to a stationary point with $O(1/\sqrt{T})$ expected squared gradient norm. Recall equation (7) to determine that the squared gradient is proportional to d_c^2. Therefore, the violation of the core constraints, d_c, decays as $O(T^{-1/4})$. □

3.3 The Least-Core as a Saddle Point Problem

Algorithms 1 and 2 assume an ϵ is provided such that all constraints represented by Constraint 2 are satisfied.[4] However, we are really interested in finding the smallest such ϵ for which the constraints still hold, i.e. the least-core value (LCV). To this end, we reformulate the original LP, making use of a single non-linear constraint:

$$\min_{p \in \Delta, \epsilon} \quad \epsilon \tag{8}$$

$$s.t. \quad \sum_{C \subseteq I} \ell_c(p, \epsilon) \leq \gamma^2 \tag{9}$$

for some constant $\gamma > 0$. Note that if $\gamma = 0$, we would recover the solution to the least-core LP in (1)-(4). For $\gamma > 0$, we can recover an approximate solution via Theorem 2. While this form is no longer an LP, it retains a crucial property: convexity. Each $\ell_c(p, \epsilon)$ is convex in p and ϵ, and hence the nonlinear constraint $\sum_{C \subseteq I} \ell_c(p, \epsilon) - \gamma^2$ remains convex in p and ϵ. Similarly Objective 8 is convex as it is linear in ϵ and p. This allows us to view the optimization problem as a saddle-point problem using Lagrange multipliers.[5] We first observe that via Karush-Kuhn-Tucker conditions, convexity in the objective and constraint is sufficient for optimality of the solution to the corresponding Lagrangian formulation (see Sec 5.5 of [16]):

$$\min_{p \in \Delta, \epsilon \in [\underline{\epsilon}, \overline{\epsilon}]} \max_{\mu \in [0, \overline{\mu}]} \mathcal{L}(p, \epsilon, \mu) = \epsilon + \mu\Big(\sum_{C \subseteq I} \ell_c(p, \epsilon) - \gamma^2\Big). \tag{10}$$

We can bound both ϵ and μ which ensures the function values and gradients are bounded as well. Let $v_{\max} = \max_{C \subseteq I} v(C)$. Then every coalitional constraint is trivially satisfied if $\epsilon \geq v_{\max}$, while the constraint associated with the grand coalition can not be satisfied if $\epsilon < 0$. Therefore, we can bound $\epsilon \in [\underline{\epsilon}, \overline{\epsilon}] = [0, v_{\max}]$. For μ, if only one constraint is violated, then we want μ to be large enough to force an increase in ϵ. Therefore, for any single violated coalitional constraint associated with C, we want $\nabla_\epsilon \mathcal{L} = 1 - \mu(d_c/|C|) < 1 - \mu(\gamma/|C|)$ to be strictly less than zero. This implies that if μ is at least $|C|/\gamma$, then ϵ will increase in response to any violated constraint. Hence, we set $\overline{\mu} = n/\gamma$ as an upper bound.

We observe that the Lagrangian formulation is convex in the primal variables (p, ϵ) and concave in the dual variable μ, hence equation (10) is typically referred to as a convex-concave saddle-point problem which can be equivalently formulated as a monotone variational inequality problem [35], for which stochastic algorithms exist including extra gradient [47] and Stochastic Mirror Prox [43]. Algorithm 4 provides pseudocode for the process. The key step is the Update function (Algorithm 3) which requires a $\mathtt{Prox}_x$ operator and a monotone map operator F.[6] In variational inequality formulations $VI(F, \mathcal{X})$, the problem is to find $x^* \in \mathcal{X}$ such that $\langle F(x^*), x - x^* \rangle \geq 0$ for all $x \in \mathcal{X}$, and the map $F : \mathcal{X} \to \mathbf{R}^{n+2}$ is typically written as a single vector valued map containing the "descent" directions of all variables. For our setting, we can define F as follows:

$$F(x) = \begin{bmatrix} \nabla_p \mathcal{L} \\ \nabla_\epsilon \mathcal{L} \\ -\nabla_\mu \mathcal{L} \end{bmatrix} = \begin{bmatrix} -\mu\Big(\sum_{C \subseteq I} \frac{d_c}{|C|} c\Big) \\ 1 - \mu\Big(\sum_{C \subseteq I} \frac{d_c}{|C|}\Big) \\ -\Big(\sum_{C \subseteq I} \ell_c(p, \epsilon) - \gamma^2\Big) \end{bmatrix} \tag{11}$$

where $x = [p, \epsilon, \mu] \in \mathcal{X} = \Delta \times [\underline{\epsilon}, \overline{\epsilon}] \times [0, \overline{\mu}]$. Observe that we can use sampling of minibatches of coalitions to Monte Carlo estimate the sums in $F(x)$, e.g., $\sum_{C \subseteq I} \ell_c(p, \epsilon) = \frac{2^n}{B} \mathbb{E}_{C_B \sim I}[\sum_{C \in C_B} \ell_c(p, \epsilon)]$.

Lemma 1. *The map F in equation (11) is monotone, i.e., $\langle F(x) - F(x'), x - x' \rangle \geq 0$ over all $x, x' \in \mathcal{X}$.*

The proof of Lemma 1 can be found in the Appendix.

Theorem 4. *Let $v : 2^I \to \mathbb{R}$ be a characteristic function. Algorithm 4 reduces the duality gap $\max_{\mu'} \mathcal{L}(p, \epsilon, \mu') - \min_{p', \epsilon'} \mathcal{L}(p', \epsilon', \mu)$ at a rate of $O(1/\sqrt{T})$.*

Proof. We can appeal directly to convergence rates of mirror prox algorithms for monotone variational inequalities [43] given we have already argued above that the norm of the map F and its variance are both finite and the map F is monotone (Lemma 1). □

In experiments, we use a tailored $\mathtt{Prox}$ operator, specifically,

$$\mathtt{Prox}_x(\eta F(y)) = \begin{bmatrix} \mathtt{softmax}(\log(p) - \eta F_p(y)), \\ \mathtt{clip}(\epsilon - \eta F_\epsilon(y), \underline{\epsilon}, \overline{\epsilon}), \\ \mathtt{clip}(\mu - \eta F_\mu(y), 0, \overline{\mu}) \end{bmatrix} \tag{12}$$

[4]Recall that such an ϵ is guaranteed to exist.

[5]Note that we introduce the constant $\gamma > 0$ so as to ensure Slater's condition holds so as to meet the necessary and sufficient conditions for optimality of the subsequent saddle point solution [16].

[6]See [53] for proximal point approaches to saddle point problems.

Algorithm 3 Update

```
Input: Initial iterate $x$, map evaluation iterate $y$, batch size $B$, step size $\eta$, Prox operator
  Sample batch $C_B$ containing $B$ coalitions
  Compute $F(y)$ using batch of coalitions
  $x' \leftarrow \text{Prox}_x(\eta F(y))$
Output: $x'$
```

Algorithm 4 Least-Core via Mirror Prox (Core Lagrangian)

```
Input: Number of iterations $T$, batch size $B$, step size schedule $\eta_t$
  $p_0 = \frac{1}{n}\mathbf{1}_n$
  $\epsilon_0 = \overline{\epsilon}$
  $\mu_0 = \overline{\mu}$
  $x_0 = [p_0, \epsilon_0, \mu_0]$
  for $t = 1 \leq T$ do
    $x' \leftarrow \text{Update}(x_{t-1}, x_{t-1}, B, \eta)$
    $x_t \leftarrow \text{Update}(x_{t-1}, x', B, \eta)$
  end for
  Compute weighted average of imputations: $p^* = \frac{\eta_t p_t}{\sum_{t'} \eta_{t'}}$.
Output: $p^*$
```

where $F_z(y)$ retrieves the part of the vector output $F(y)$ corresponding to the variable y and $\text{softmax}(s) = [\frac{e^{s_1}}{\sum_j e^{s_j}}, \dots, \frac{e^{s_n}}{\sum_j e^{s_j}}]$.

Instructions for accelerating our proposed algorithms on GPUs/TPUs are in Appendix A.2. Hyperparameters including, e.g., learning rate schedules, used in experiments are found in Appendix A.3.

4 EMPIRICAL ANALYSIS

We tested our algorithm, Core Lagrangian (CL) (Alg. 4), on a range of cooperative games and report our findings here. In our first set of experiments we compared the performance of CL to that of a recent state-of-the-art algorithm for approximating the least-core which relies on solving the LP using sampled coalitions [69] (Section 4.1). We then use our algorithm as a tool to study the stability properties of a number of prominent cooperative-game classes, illustrating the benefits of having algorithms for approximating the core for large games (Section 4.2). Finally, we examine applications of our algorithms for explainable AI (XAI) purposes (Section 5).

4.1 Timing Sampled LPs vs. Core Lagrangian

To test how efficient CL is in practice, we evaluate its performance across very large ($n = 100$) instances of weighted voting games.

Definition 4. *([19, Definition 4.1]) A weighted voting game G is a tuple $(I, \mathbf{w}, q)$ where $I = \{1, 2, \cdots, n\}$ is a set of agents, $\mathbf{w} = (w_1, w_2, \cdots, w_n) \in \Re^n$ is a vector of weights (one per player), and $q \in \Re$ is a quota. The characteristic function is*

$$v(C) = \begin{cases} 1 & \text{if } \sum_{c \in C} w_c \geq q; \\ 0 & \text{otherwise.} \end{cases}$$

Weighted voting games are well studied in the literature (*e.g.*, [3, 31, 59, 62, 67, 71]) and can model real-world scenarios such as the European Union voting system [15, 57], or multiagent resource allocation problems where an agent's weight correspond to resources available for that agent, and the threshold is the total amount of pooled resources required to accomplish some task of interest.

In our experiments, each agent i has a weight $w_i \in \{1, 2, \cdots, 100\}$ drawn uniformly at random and we set the quota $q = \xi n \mathbb{E}[w_i]$, where ξ is the proportional threshold (fraction of the expected total weight). We use proportional thresholds ξ sampled uniformly at random in the range $[0.1, 0.9]$. There are 2^{100} coalitions and thus 2^{100} constraints in the LP formulation of the least core (Constraint 2), so exactly solving for the least-core is computationally infeasible.

We compared our method to Yan and Procaccia's recent algorithm for approximating the least core [69]. Their algorithm samples a subset of coalitions and returns the LP solution based on constraints built only from the sampled subset. With high probability, this method is guaranteed to reduce the approximation error as the number of coalitions sampled increases [69, Theorem 1].

It is difficult to compare iterations of CL to the number of sampled coalitions used in LP; instead, we chose a number of sampled coalitions k and recorded the wall clock time taken by the LP method for each value of k, say t_k seconds. Then, we let CL run for t_k seconds and retrieved the solution after that amount of elapsed time. The LP method used CVXPY [28] while our CL code used JAX and optax [4]. For each t_k, both methods returned an imputation, p_{LP} and p_{CL}, respectively. For each imputation we computed its respective (average) ϵ value. Due to the size of the game the error of the solution cannot be computed exactly, so we approximated the value of ϵ by sampling a set of 50,000 coalitions, $\hat{C}$, and computing $\hat{\epsilon}(p, \hat{C}) = \max_{c \in \hat{C}}(v(c) - p^\top c)$. Figure 2 summarizes our findings. In particular, we observed a significant improvement in the core approximation when using our method (CL) as opposed to the LP-based method. We did observe, however, that the threshold affects the degree to which CL outperforms the LP-based method. In particular, when the threshold was close to 50% of the total expected weight, the LP-based method would sometimes outperform CL.

4.2 Analyzing Stability in Compact Game Representations

After validating that our algorithm can effectively solve large cooperative games, showing improvements over a strong baseline, we further illustrate its value by making a rigorous study across prominent classes of cooperative games to better understand how different game-features affect stability.

4.2.1 Weighted Voting Games. In our first set of experiments we studied how stability was impacted by different parameterizations of weighted voting games, defined in Section 4.1. Since we were running multiple experiments, we set the number of agents to be $n = 15$, but drew agents' weights from different distributions and varied the proportional quota, ξ. Figure 1 presents our results. We sample 10,000 games for each parameter configuration, and examine the average least-core value in these games. We first note that regions that have a low least-core value (blue) are stable or very close to being stable, whereas high ϵ-core indicate regions of instability. Second, we observe that the threshold greatly affected the stability of an instance. If the threshold was low, many coalitions form and achieve the maximum coalitional value of 1. If the threshold was very high, in expectation the only successful coalition was the grand coalition, reducing the likelihood of blocking coalitions.

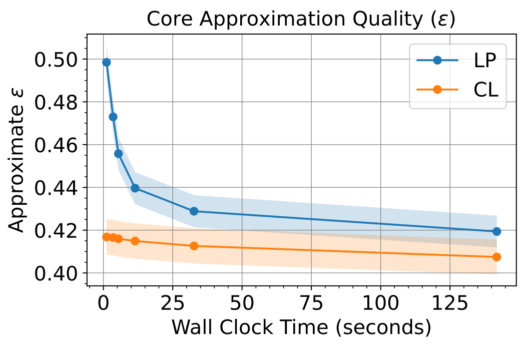


Figure 2: Approximate ϵ of the linear program (LP) versus the core Lagrangian (CL) method as a function of computation time (seconds). The x-axis corresponds to wall-clock time taken by both algorithms run side-by-side given $k \in \{500, 1000, 2000, 4000, 8000, 16000\}$ sampled coalitions for the LP method, t_k. A better core approximation quality is reflected in having a lower ϵ for the same runtime. The y-axis represents the approximate ϵ of the least-core solution found, $\hat{\epsilon}(p_{LP}, \hat{C})$ and $\hat{\epsilon}(p_{CL}, \hat{C})$, computed over the same set of $50,000$ coalitions, $\hat{C}$. Each data point $(t_k, \hat{\epsilon}(p_x, \hat{C}))$ represents an average over the same set of $2,500$ random weighted voting games with shading indicating standard error of the mean.

In Figure 1(a) we present results where the agents' weights were generated by a Gaussian distribution with $\mu = 1.0$ and $\sigma \in [0.01, 0.3]$. We observe that for a fixed quota, high weight-variance leads to less stable games. Figure 1(b) presents results when agents' weights were drawn from an exponential distribution with $\lambda \in [0.25, 2.50]$. We observe that as λ increases we find less stable games. We hypothesize that since agent weights are more concentrated around low values, possible successful coalitions often share agents, opening up the possibility of the formation of blocking coalitions. Finally, in Figure 1(c) we present results where we used a Beta distribution with parameters (α, β). We notice a difference in stability of games when the parameters are either less than 1.0 or greater than 1.0. If both α and β are high, games are less stable. However, there is a region of stability when $\alpha > 1.0$ and $\beta < 1.0$

4.2.2 *Graph Games.*

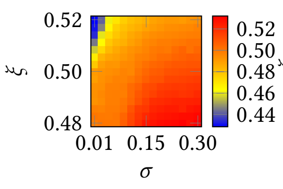
In many cooperative games, relationships between agents are modelled via a graph $G = \langle V, E \rangle$ (*e.g.*, [7, 8, 17, 25, 26, 54, 58, 60]). In induced subgraph games [26], the agents are represented by vertices of the graph and the edge weight, $w(e)$ for $e = (i, j)$, indicates the value that agents i and j accrue from being in the same coalition. An absence of an edge implies there is no interaction between a pair of agents and thus, no loss or benefit from being in the same coalition. Given graph G, the characteristic function for the induced cooperative game is $v(C) = \sum_{e \in \{(i,j) \in E | i,j \in C\}} w(e)$.

We conduct an empirical study to better understand how the underlying graph structure influences the stability of the game. To this end, we select a collection of well-known parameterized random graph models, each with different properties, and generate 80 cooperative graph games for each parameterization, measuring their stability by computing their least-core values via our algorithm CL. We set $n = 32$, and generate edge weights with a Gaussian distribution with variance $\sigma = 1.0$ (unless specified otherwise) and positive mean μ chosen such that 60% of edge weights are positive in expectation.[7] All graphs were generated using the NetworkX library [39]. We present results on two graph-classes, Erdős-Rényi [34] and Newman Watts Strogatz [55], and discuss four additional models in Appendix C.

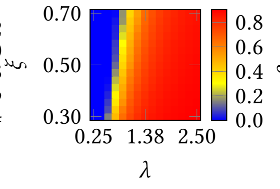
In the Erdős-Rényi graph model there is a single parameter p, which indicates the probability that for any pair of vertices, x, y, there is an edge connecting x and y. In our experiments we varied p from 0.0 to 1.0, and varied the variance σ of the weight distribution from 1.0 to 3.0. Our results are shown in Figure 1d. We observe that the stability of the games generated depends on both p and σ. In particular, games with high weight variance for edge weights, and high uncertainty as to whether an edge would form between any pair of vertices (*i.e.*, for $p \in [0.4, 0.7]$) are less stable.

The Newman Watts Strogatz model generates graphs with the small-world property and consists of two parameters. An instance of a graph is initialized as a ring and connected with its $s\lfloor \frac{k}{2} \rfloor$ nearest neighbours. Additional edges in the graph are added with probability p. Figure 1e presents our results as we varied k from 4 to 24, and $p \in [0.0, 1.0]$. The variance of the edge-weight distribution was fixed at $\sigma = 1.0$. In particular we observe that both parameters p and k positively correlate with stability in these graphs.

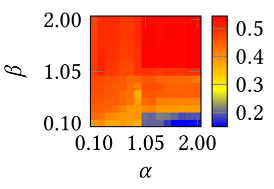
[7] If graph games have only positive edge weights, then the core is non-empty, i.e. the least-core value is 0.0 [26].



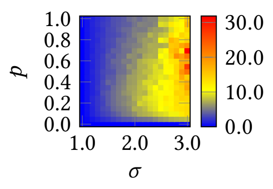
(a) Gaussian ($\mu = 1.0$)



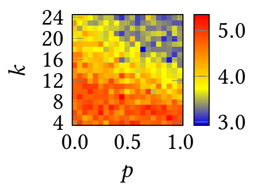
(b) Exponential



(c) Beta ($\xi = 0.25$)



(d) Erdős-Rényi



(e) Newman Watts Strogatz ($\sigma = 1$)

Figure 1: (a)-(c) Heatmaps illustrating the impact of parameter axes on the least-core value (in color) in three weighted voting games. (d)-(e) Mean least-core value for Erdős-Rényi and Newman Watts Strogatz when sweeping over two hyperparameters. Constant hyperparameters are shown in parenthesis.

More generally, these results (along with further results in Appendix C) show the interplay between the parameters of random-graph generators and the stability of the induced cooperative game. We believe that further research in this area is warranted, but also emphasize that such empirical analysis can only be carried out when using algorithms for tractably solving cooperative games at scale, such as the algorithms we have proposed.

5 EXPLAINABLE AI

Cooperative game theory has played an important role in the development of explainable AI (XAI) through the application of the Shapley value to feature-attribution and data-valuation problems [1, 2, 21, 50]. However researchers have also identified disadvantages of Shapley-based analysis, such as generating counter-intuitive explanations in various cases [65]. A recent paper argued that the core may provide an alternative to the Shapley value for XAI applications. In this section we explore this idea, applying our algorithm, CL, for computing the least-core on three different real-world datasets: Boston Housing dataset [40] (price regression), Diabetes dataset [30] (classification) and the Wisconsin Breast Cancer dataset [68] (classification). Our goal is to better understand how the Shapley value and the core are similar or different in their relative assessment of the impact of different features or data points have on the quality of trained models, specifically, scikit-learn's default linear or logistic regression models [56].

5.1 Global Explainability

In the global explainability problem, a full dataset is used to determine which features have high impact on the quality of the trained model. We formulate this as a cooperative game by defining a coalition, C, to be a subset of features and $v(C)$ to be the quality of the model trained only on features in C. Quality may be measured as the accuracy in classification tasks, or as the coefficient of determination R^2 for regression tasks. Computing the Shapley value or the least-core for this game returns individual values, one per each feature, which are can be interpreted as a measure of feature-importance.

Figure 3 presents a scatter plot showing the correlation between the feature-importance measures returned by applying the Shapley value and those produced by the least core (using our algorithm). Every point represents a single feature, with the x-axis reflecting its importance according to the Shapley value and the y-axis reflecting its importance according to the least core. The two importance measures are correlated, but reflect a different order of feature importance. Interestingly, the correlation is high when there are a few dominant features (Figures 3a,3b), but not as strong in the case where the predictions are driven by many features where none of the features is dominant (Figure 3c).

5.2 Local Explainability

In the local explainability problem, the focus is on the prediction a trained model makes on an individual instance. It seeks to determine how the values of individual features increase or decrease the model's prediction for some instance. We define a cooperative game where a coalition, C, is a set of features and the characteristic function is the model's prediction on a new instance, created by taking some original instance from the dataset, fixing the values of all features in C, and replacing the values of features not in C by randomly sampling other instances.[8] Given a dataset containing d instances, such an analysis results in f feature importance measures per each instance, resulting in $d \cdot f$ feature-importance measures, illustrating the importance of having scalable solutions for computing both the Shapley value and least-core.

Figure 3 presents results showing the correlation between Shapley feature-importance and least-core measures, over all the instances in each dataset. We present the data using contour plots due to the sheer number of data-points. Similar to our findings for the global explainability problem, we observe positive correlation between least-core and Shapley value. Note that the least core returns non-negative payoffs, which explains the bend in the trend at the origin.

5.3 Data Valuation

In the data valuation problem, the importance of each *data point* in the training set is measured. This problem can be formulated as a cooperative game by defining a coalition, C, to be a set of data points, with characteristic function, $v(C)$, being the quality of the model trained only on data points in C. The core (or least core) is a particularly compelling solution for the data valuation problem as imputations in the core can be interpreted as prices that must be paid to data providers so as to ensure the data is available (*e.g.*, in the same way that core-pricing is used in package-auctions [23, 24]).

We conducted a series of data-valuation studies across different data sets. We used a similar methodolgy as Yan and Procaccia [69]: we imposed a budget of 50, 000 calls to the characteristic function, limiting the number of permutations sampled for the Shapley value approximation and iterations for our least-core algorithm, CL. We then sorted the data by importance according to the Shapley value and least-core, and retrained the models by removing the most important data in blocks of 5% at a time.

Results are shown in Figure 4. In particular, these show data points deemed most important by the least-core are more critical to model performance than those selected according to Shapley value. These findings are consistent with previous literature [69]. However, we also applied this process to other data sets including an evaluation problem for large language models (LLMs), presented in Appendix D.2. There, the results are more nuanced in that there exist scenarios where a Shapley value approach is better at identifying key data points. These results open up new research questions around characterizations as to when Shapley-based or core-based data valuation is more appropriate.

5.3.1 Elo Ratings on Chatbot Arena data set. We now describe another data valuation experiment that differs from the others in that it is not a traditional regression nor classification task. The Chatbot Arena data set is composed of humans rating the quality of answers to questions to 20 different large language models (LLMs) [70]. Each data point consists of a question and two answers: each answer generated by two different LLMs. The human then picks which answer is the best one and this is recorded as a win for the LLM that

[8]In our experiments with the least core, we use the version where non-coalition features are taken from a random instance selected from the 10% of the instances with the lowest prediction.

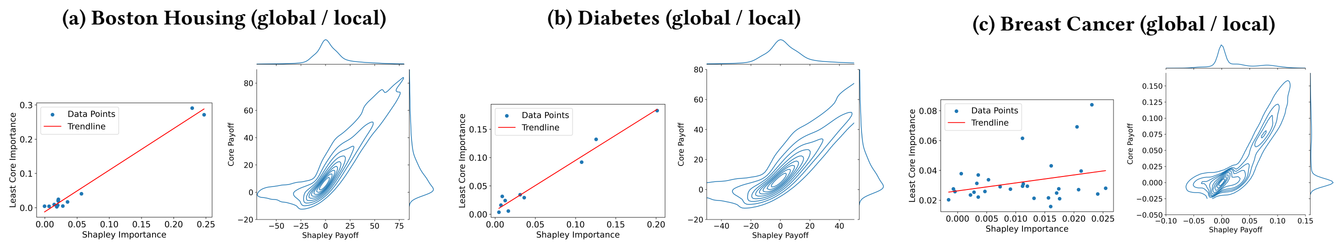


Figure 3: Correlation of Shapley and Core importance measures of features at both global dataset and individual instance levels.

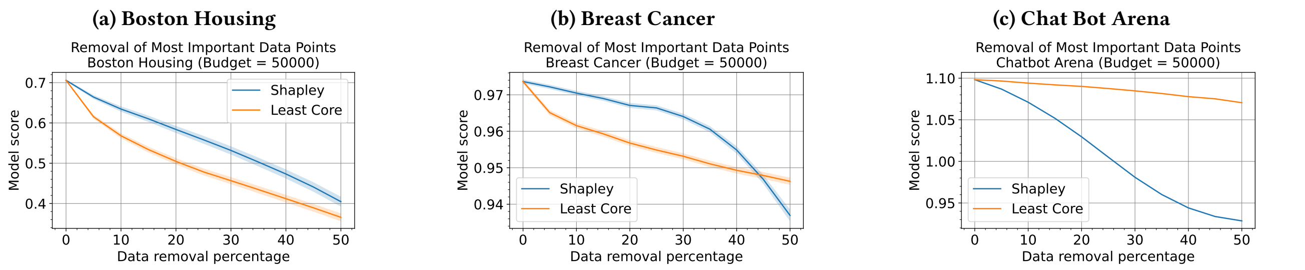


Figure 4: Data Valuation on the Boston Housing, Breast Cancer, and Chat Bot Arena. Error bars correspond to 95% confidence intervals (1000 repeats).

generated the better response and a loss for the one that generated the worse response. In total, the data set consists of 33000 data points.

The supervised learning problem is to learn an Elo rating [33] for each LLM that can be ranked to compare the “skill level” (in this context: propensity to generate the better answer) of each LLM. Elo is classic rating system that was proposed for ranking chess engines; each LLM is assigned a rating, say r_i and r_j, that models the probability of LLM i beating LLM j as a logistic function of there ratings, $\Pr(\text{i beats j}) = \sigma\left((r_i - r_j)/400\right) = \frac{1}{1+e^{(r_j - r_i)/400}}$, where σ is the sigmoid function $\sigma(x) = (1 + e^{-x})^{-1}$. The ratings can be learned using online updates, or more precisely given a batch of data using logistic regression or minorization-maximization (MM) algorithms [41].

We apply data valuation in a similar way as before. For each experiment, we first sample a training set and test set pair $(\mathcal{D}_R, \mathcal{D}_T)$, with $|\mathcal{D}_R| = 1000$ and $|\mathcal{D}_T| = 10000$. A coalition is then a subset of data points from this training set, $c \subseteq \mathcal{D}_R$, leading to a 1000-player coalitional game. As in the other settings, the characteristic function is a measure of how well the model learned on c performs on the test set $\mathcal{D}_T$. For each coalition, c, we run the MM method [41] for 20000 iterations to learn Elo ratings of the batch of data $c \subseteq \mathcal{D}_R$. Given the learned ratings $\vec{r}_c$, we compute the average cross entropy loss over the test set, $L_{CE}(\vec{r}_c, \mathcal{D}_T)$. Finally, we define the the characteristic function for as $v(c) = 2 - L_{CE}(\vec{r}_c, \mathcal{D}_T)$. As before we set a budget of 50000 calls to the characteristic function for both Shapley and the least core computations. Figure 4c shows that in contrast to the other data sets and experiments in [69], the Shapley values attribute importances that are more critical to model performance than the least core.

6 CONCLUSION

We examined the core, a distribution of payoff over members in a coalition that is a central solution concept in cooperative game theory. We proposed a scalable solver for the least core that can handle the exponential number of possible coalitions that define the core constraints. We provided convergence rates and guarantees for this solver and showed empirically that it is faster than previous core solvers [69].

Our empirical analysis shows several applications of our core solver, including studying stability of prominent forms of coalitional games and explainable AI (XAI) problems. For XAI, we highlighted that analysis based on the core differs from the current de-facto standard based on the Shapley value. Further, for the purpose of data evaluation, our results indeed show that in certain cases the core outperforms the Shapley value as a way of selecting the data instances for training machine learning models.

Several problems remain open for future research. First, could one derive even faster algorithms for approximating the core (in terms of the worst-case performance, or in terms of empirical performance on problems of interest)? Second, could better approximation algorithms for the core be tailored to specific classes of games? Third, could one extend our analysis to other known forms of cooperative games to determine the key features that affect coalitional stability in them? Fourth, our analysis has identified ways of selecting data for training models. Could one leverage such results to speed up the training or runtime performance of machine learning models? Finally, when is it better to use the core and when is it better to use the Shapley value for feature importance measurements or data selection?

REFERENCES

[1] Liat Antwarg, Ronnie Mindlin Miller, Bracha Shapira, and Lior Rokach. 2019. Explaining anomalies detected by autoencoders using SHAP. *arXiv preprint arXiv:1903.02407* (2019).

[2] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion* 58 (2020), 82–115.

[3] Haris Aziz, Yoram Bachrach, Edith Elkind, and Mike Paterson. 2011. False-name manipulations in weighted voting games. *Journal of Artificial Intelligence Research* 40 (2011), 57–93.

[4] Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Antoine Dedieu, Claudio Fantacci, Jonathan Godwin, Chris Jones, Ross Hemsley, Tom Hennigan, Matteo Hessel, Shaobo Hou, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Markus Kunesch, Lena Martens, Hamza Merzic, Vladimir Mikulik, Tamara Norman, George Papamakarios, John Quan, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Laurent Sartran, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Miloš Stanojević, Wojciech Stokowiec, Luyu Wang, Guangyao Zhou, and Fabio Viola. 2020. *The DeepMind JAX Ecosystem*. http://github.com/deepmind

[5] Yoram Bachrach. 2011. The least-core of threshold network flow games. In *International Symposium on Mathematical Foundations of Computer Science*. Springer, 36–47.

[6] Yoram Bachrach, Evangelos Markakis, Ezra Resnick, Ariel D Procaccia, Jeffrey S Rosenschein, and Amin Saberi. 2010. Approximating power indices: Theoretical and empirical analysis. *Autonomous Agents and Multi-Agent Systems* 20 (2010), 105–122.

[7] Yoram Bachrach and Ely Porat. 2010. Path disruption games. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*. 1123–1130.

[8] Yoram Bachrach and Jeffrey S Rosenschein. 2007. Computing the Banzhaf power index in network flow games. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*. 1–7.

[9] Catalin Badea and David Seifert. 2016. Ritt operators and convergence in the method of alternating projections. *Journal of Approximation Theory* 205 (2016), 133–148.

[10] Maria-Florina Balcan, Ariel Procaccia, and Yair Zick. 2015. Learning Cooperative Games. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)*. 475–481.

[11] U.; Balkanski, E.; Syed and S. Vassilvitskii. 2017. Statistical Cost Sharing. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NeurIPS)*. 6221–6230.

[12] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion* 58 (2020), 82–115. https://doi.org/10.1016/j.inffus.2019.12.012

[13] Heinz H Bauschke and Jonathan M Borwein. 1996. On projection algorithms for solving convex feasibility problems. *SIAM Rev.* 38, 3 (1996), 367–426.

[14] JM Bilbao, JR Fernandez, A Jiménez Losada, and JJ Lopez. 2000. Generating functions for computing power indices efficiently. *Top* 8, 2 (2000), 191–213.

[15] JM Bilbao, N Jiminéz, and JJ López. 2002. Voting power in the European Union enlargement. *European Journal of Operations Research* 143 (2002), 181–196.

[16] Stephen P Boyd and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

[17] Simina Branzei and Kate Larson. 2011. Social Distance Games. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI 2011)*.

[18] Javier Castro, Daniel Gómez, and Juan Tejada. 2009. Polynomial calculation of the Shapley value based on sampling. *Computers & Operations Research* 36, 5 (2009), 1726–1730.

[19] Georgios Chalkiadakis, Edith Elkind, and Michael Wooldridge. 2012. *Computational Aspects of Cooperative Game Theory*. Morgan & Claypool Publishers.

[20] George Dantzig. 1963. *Linear programming and extensions*. Princeton University Press.

[21] Anupam Datta, Shayak Sen, and Yair Zick. 2016. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 598–617.

[22] Damek Davis and Dmitriy Drusvyatskiy. 2018. Stochastic subgradient method converges at the rate $O(k^{-1/4})$ on weakly convex functions. *arXiv preprint arXiv:1802.02988* (2018).

[23] Robert Day and Paul Milgrom. 2008. Core-selecting package auctions. *international Journal of Game Theory* 36 (2008), 393–407.

[24] Robert W. Day and Peter Cramton. 2012. Quadratic Core-Selecting Payment Rules for Combinatorial Auctions. *Operations Research* 60, 3 (2012), 588–603.

[25] Gabrielle Demange. 2004. On Group Stability in Hierarchies and Networks. *Journal of Political Economy* 112, 4 (2004), 754–778.

[26] Xiaotie Deng and Christos H. Papadimitriou. 1994. On the complexity of cooperative solution concepts. *Math. Oper. Res.* 19, 2 (1994), 257–266.

[27] Frank Deutsch. 1995. Dykstra's cyclic projections algorithm: the rate of convergence. *Approximation Theory, Wavelets and Applications* (1995), 87–94.

[28] Steven Diamond and Stephen Boyd. 2016. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research* 17, 83 (2016), 1–5.

[29] Pradeep Dubey. 1982. The Shapley value as aircraft landing fees–revisited. *Management Science* 28, 8 (1982), 869–874.

[30] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. 2004. Least angle regression. (2004).

[31] Edith Elkind, Leslie Ann Goldberg, Paul W Goldberg, and Michael Wooldridge. 2009. On the computational complexity of weighted voting games. *Annals of Mathematics and Artificial Intelligence* 56 (2009), 109–131.

[32] E. Elkind and D. Pasechnik. 2009. Computing the nucleolus of weighted voting games. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics Philadelphia, PA, USA, 327–335.

[33] Arpad E. Elo. 1978. *The Ratings of Chess Players, Past and Present* (2nd ed.). Arco Publishing, Inc.

[34] László Erdős, Antti Knowles, Horng-Tzer Yau, and Jun Yin. 2013. Spectral statistics of Erdős–Rényi graphs I: Local semicircle law. (2013).

[35] Francisco Facchinei and Jong-Shi Pang. 2003. *Finite-Dimensional Variational Inequalities and Complementarity Problems*. Springer.

[36] Shaheen S Fatima, Michael Wooldridge, and Nicholas R Jennings. 2008. A linear approximation method for the Shapley value. *Artificial Intelligence* 172, 14 (2008), 1673–1699.

[37] Ian Gemp, Marc Lanctot, Luke Marris, Yiran Mao, Edgar A. Duéñez-Guzmán, Sarah Perrin, Andras Gyorgy, Romuald Elie, Georgios Piliouras, Michael Kaisers, Daniel Hennes, Kalesha Bullard, Kate Larson, and Yoram Bachrach. 2024. Approximating the Core via Iterative Coalition Sampling. *arXiv preprint arXiv:2402.03928* (2024).

[38] Donald Bruce Gillies. 1953. *Some theorems on n-person games*. Ph.D. Dissertation. Princeton University.

[39] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. 2008. Exploring Network Structure, Dynamics, and Function using NetworkX. In *Proceedings of the 7th Python in Science Conference*, Gaël Varoquaux, Travis Vaught, and Jarrod Millman (Eds.). Pasadena, CA USA, 11 – 15.

[40] David Harrison Jr and Daniel L Rubinfeld. 1978. Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management* 5, 1 (1978), 81–102.

[41] David R. Hunter. 2004. MM Algorithms for Generalized Bradley-Terry Models. *The Annals of Statistics* 32, 1 (2004), 384–406.

[42] Samuel Ieong and Yoav Shoham. 2005. Marginal contribution nets: a compact representation scheme for coalitional games. In *Proceedings of the 6th ACM Conference on Electronic Commerce*. 193–202.

[43] Anatoli Juditsky, Arkadi Nemirovski, and Claire Tauvel. 2011. Solving variational inequalities with stochastic mirror-prox algorithm. *Stochastic Systems* 1, 1 (2011), 17–58.

[44] Narendra Karmarkar. 1984. A new polynomial-time algorithm for linear programming. In *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*. 302–311.

[45] Walter Kern and Daniël Paulusma. 2003. Matching games: the least core and the nucleolus. *Mathematics of Operations Research* 28, 2 (2003), 294–308.

[46] Alf Kimms and Igor Kozeletskyi. 2016. Core-based cost allocation in the cooperative traveling salesman problem. *European Journal of Operational Research* 248, 3 (2016), 910–916.

[47] Galina M Korpelevich. 1976. The extragradient method for finding saddle points and other problems. *Matecon* 12 (1976), 747–756.

[48] Marc Lanctot, Edward Lockhart, Jean-Baptiste Lespiau, Vinicius Zambaldi, Satyaki Upadhyay, Julien Pérolat, Sriram Srinivasan, Finbarr Timbers, Karl Tuyls, Shayegan Omidshafiei, Daniel Hennes, Dustin Morrill, Paul Muller, Timo Ewalds, Ryan Faulkner, János Kramár, Bart De Vylder, Brennan Saeta, James Bradbury, David Ding, Sebastian Borgeaud, Matthew Lai, Julian Schrittwieser, Thomas Anthony, Edward Hughes, Ivo Danihelka, and Jonah Ryan-Davis. 2019. OpenSpiel: A Framework for Reinforcement Learning in Games. *CoRR* abs/1908.09453 (2019). arXiv:1908.09453 [cs.LG] http://arxiv.org/abs/1908.09453

[49] Stephen C Littlechild and GF Thompson. 1977. Aircraft landing fees: a game theory approach. *The Bell Journal of Economics* (1977), 186–204.

[50] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems* 30 (2017).

[51] Sasan Maleki, Long Tran-Thanh, Greg Hines, Talal Rahwan, and Alex Rogers. 2013. Bounding the estimation error of sampling-based Shapley value approximation. *arXiv preprint arXiv:1306.4265* (2013).

[52] Rory Mitchell, Joshua Cooper, Eibe Frank, and Geoffrey Holmes. 2022. Sampling permutations for shapley value estimation. *The Journal of Machine Learning Research* 23, 1 (2022), 2082–2127.

[53] Aryan Mokhtari, Asuman Ozdaglar, and Sarath Pattathil. 2020. A unified analysis of extra-gradient and optimistic gradient methods for saddle point problems: Proximal point approach. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 1497–1507.

[54] Roger B. Myerson. 1977. Graphs and Cooperation in Games. *Mathematics of Operations Research* 2, 3 (1977), 225–229.

[55] M.E.J. Newman and D.J. Watts. 1999. Renormalization group analysis of the small-world network model. *Physics Letters A* 263, 4 (1999), 341–346. https://doi.org/10.1016/S0375-9601(99)00757-4

[56] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[57] Tapio Raunio and Matti Wiberg. 1998. Winners and Losers in the Council: Voting Power Consequences of EU Enlargements. *Journal of Common Market Studies (JCMS)* 38 (1998), 549–562. Issue 4.

[58] Ezra Resnick, Yoram Bachrach, Reshef Meir, and Jeffrey S Rosenschein. 2009. The cost of stability in network flow games. In *Mathematical Foundations of Computer Science 2009: 34th International Symposium, MFCS 2009, Novy Smokovec, High Tatras, Slovakia, August 24-28, 2009. Proceedings 34*. Springer, 636–650.

[59] Anja Rey and Jörg Rothe. 2010. Complexity of merging and splitting for the probabilistic Banzhaf power index in weighted voting games. In *ECAI 2010*. IOS Press, 1021–1022.

[60] Anja Rey and Jörg Rothe. 2011. Bribery in path-disruption games. In *International Conference on Algorithmic Decision Theory*. Springer, 247–261.

[61] Dov Samet and Eitan Zemel. 1984. On the core and dual set of linear programming games. *Mathematics of Operations Research* 9, 2 (1984), 309–316.

[62] Abigail See, Yoram Bachrach, and Pushmeet Kohli. 2014. The cost of principles: analyzing power in compatibility weighted voting games. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems*. Citeseer, 37–44.

[63] Lloyd S. Shapley. 1951. Notes on the n-Person Game – II: The Value of an n-Person Game. RAND Corporation.

[64] Lloyd S Shapley and Martin Shubik. 1954. A method for evaluating the distribution of power in a committee system. *American Political Science Review* 48, 3 (1954), 787–792.

[65] Mukund Sundararajan and Amir Najmi. 2020. The many Shapley values for model explanation. In *International Conference on Machine Learning*. PMLR, 9269–9278.

[66] Nikola Tarashev, Kostas Tsatsaronis, and Claudio Borio. 2016. Risk attribution using the Shapley value: Methodology and policy applications. *Review of Finance* 20, 3 (2016), 1189–1213.

[67] Alan Taylor and William Zwicker. 1992. A characterization of weighted voting. *Proceedings of the American mathematical society* 115, 4 (1992), 1089–1094.

[68] William Wolberg, Olvi Mangasarian, Nick Street, and W. Street. 1995. Breast Cancer Wisconsin (Diagnostic). UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5DW2B.

[69] Tom Yan and Ariel D. Procaccia. 2021. If You Like Shapley Then You'll Love the Core. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 6 (May 2021), 5751–5759. https://doi.org/10.1609/aaai.v35i6.16721

[70] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. arXiv:2306.05685 [cs.CL] See also https://lmsys.org/blog/2023-05-03-arena/ and https://huggingface.co/spaces/lmsys/chatbot-arena-leaderboard. Accessed August 13th, 2023.

[71] Michael Zuckerman, Piotr Faliszewski, Yoram Bachrach, and Edith Elkind. 2012. Manipulating the quota in weighted voting games. *Artificial Intelligence* 180 (2012), 1–19.