Upper and lower bounds for saddle functions

Regan Baucke
Department of Engineering Science
University of University
New Zealand
r.baucke@auckland.ac.nz

Abstract

In this talk, we will present an intuitive construction of upper and lower bound functions for a given saddle function. We will present these bounding functions in the context of an algorithm which computes a saddle point of the given saddle function. Finally, we demonstrate how these upper and lower bounding functions play a part in a more complex algorithm used to solve a class of minimax stochastic dynamic programmes.

Examples of such problems are risk averse multistage stochastic programmes, stochastic programming problems where uncertainty appears in both the right-hand-side and the objective function, and stochastic sub-game perfect equilibria.

Key words: stochastic programming, minimax, dynamic programming, saddle point, zero-sum game.

1 Introduction

Dynamic programming and temporal decomposition techniques are popular methodologies for solving large optimisation problems. The philosophy behind these techniques is that solving a series of smaller optimisation problems and exploiting their structure should be faster than solving the optimisation problem in its entirety.

In dynamic programming, *cost-to-go functions* are computed which encode the value of future optimal decisions conditional on a given state. A stage problem is then solved which seeks to minimise the current cost plus the cost-to-go which is then used to define the cost-to-go function for the current stage and state.

Under certain assumptions (convexity of cost functions, convex and compact state and control sets) we can define a lower-bound on the cost-go-function as the maximum of a set of linear cuts. The stochastic dual dynamic programming algorithm (or SDDP see [Philpott and Guan, 2008]) is a popular algorithm which fits this description. Briefly, this algorithm constructs a lower approximation to the value-function and iteratively refines this lower approximation by sampling the value function and updating the function with a “cut” which uses first-order (gradient) and zeroth-order (point) information. These lower bound value function estimates are nested backwards by virtue of the recursive nature of the dynamic programming equations. Although the lower approximation isn't accurate everywhere on the cost-to-go function's domain, a particular sampling procedure ensures that the value functions can be made arbitrarily accurate near the eventual optimal solution.

The key to the convergence of these algorithms relies on two main properties of the bounding functions: that the functions are valid bounds over the domain of the cost-to-go function, and secondly that the bounding functions are tight.

In this paper we will introduce a pair of bounding functions which have the saddle function analogue of these two properties. We will show how these bounding functions work to compute a saddle point of a given saddle function. However, the foremost utility of these saddle functions is their ability to provide valid bounds on value functions when the lower and upper bounds are nested i.e. in dynamic programming. We will conclude the paper with a discussion about an algorithm which is able to solve a broad class of multistage stochastic minimax dynamic programmes.

2 Saddle functions

In this section, we discuss several preliminary concepts which are essential for the description of our algorithm. We introduce the concepts of a saddle function, saddle points, and discuss an extended Lipschitz continuity concept on the Cartesian product of two vector spaces.

We motivate this section by an algorithm which computes the saddle point of a saddle function. This problem can be viewed as the saddle function analogue of the classic Kelley Cutting Plane algorithm for computing the minimum of convex functions as in [Kelley, 1960].

2.1 Preliminaries

We begin by presenting our definitions and notation of various elementary concepts.

Definition 1. *Given sets $\mathcal{U} \subset \mathbb{R}^n$ and $\mathcal{V} \subset \mathbb{R}^m$, a function $g : \mathcal{U} \times \mathcal{V} \mapsto \mathbb{R}$ is a saddle function if $g(\cdot, v)$ is a convex function for all $v \in \mathcal{V}$ and $g(u, \cdot)$ is a concave function for all $u \in \mathcal{U}$.*

Definition 2. *A point $(\dot{u}, \dot{v}) \in \mathcal{U} \times \mathcal{V}$ is a saddle point of a saddle function $g : \mathcal{U} \times \mathcal{V} \mapsto \mathbb{R}$ if*

$$g(\dot{u}, v) \leq g(\dot{u}, \dot{v}) \leq g(u, \dot{v}), \ \forall (u, v) \in \mathcal{U} \times \mathcal{V}. \tag{2.1}$$

Let $\mathcal{U}$ and $\mathcal{V}$, now be compact and convex subsets of their respective Euclidean spaces. Consider the functions $g_v(u) = \max_{v \in \mathcal{V}} g(u, v)$ and $g_u(v) = \min_{u \in \mathcal{U}} g(u, v)$; their existence is guaranteed because $g(\cdot, \cdot)$ is real valued and $\mathcal{U}$ and $\mathcal{V}$ are both compact. It is easy to see that $g_v(u)$ is a convex function (and symmetrically, $g_u(v)$ is a concave function). These functions are useful in the next lemma.

Lemma 2.1. *If $g : \mathcal{U} \times \mathcal{V} \mapsto \mathbb{R}$ is a saddle function, as given in Definition 1 and $\mathcal{U}$ and $\mathcal{V}$ are both convex and compact, then the set of saddle points is given by*

$$\mathcal{G} = \big\{ \arg\min_{u} g_v(u) \times \mathcal{V} \cap \arg\max_{v} g_u(v) \times \mathcal{U} \big\} \subseteq \mathcal{U} \times \mathcal{V}. \tag{2.2}$$

Proof. From the minimax theorem of von Nuemann, we have $g_v(u^*) = g(u^*, v^*) = g_u(v^*)$. With this equality, it is easy to see that (u^*, v^*) directly satisfies Definition 2. This establishes the forward direction of the lemma. Note that for all $(\hat{u}, \hat{v}) \notin \mathcal{G}$ we have either:

$$\begin{aligned} g_v(u^*) \le g_v(\hat{u}) \quad &\text{and} \quad g_u(v^*) > g_u(\hat{v}), \text{ or} \\ g_v(u^*) < g_v(\hat{u}) \quad &\text{and} \quad g_u(v^*) \ge g_u(\hat{v}). \end{aligned}$$

It follows that $(\hat{u}, \hat{v})$ does not satisfy Definition 2. □

Following from Lemma 2.1, it is easy to see that $\mathcal{G}$ is a convex set (being the intersection of two convex sets), and that all elements of $\mathcal{G}$ attain the same value under g. So far we have established that the set of saddle points of our function must be convex, and all saddle points have the same function value. We will now introduce the concept of an ϵ-*saddle point*. These points extend the idea of a saddle point and will aid in describing the convergent properties of the proposed algorithm.

Definition 3. *A point $(\hat{u}, \hat{v})$ is an ϵ-saddle point of a function $g : \mathcal{U} \times \mathcal{V} \mapsto \mathbb{R}$ if $g_v(\hat{u}) - \epsilon \le g(\dot{u}, \dot{v}) \le g_u(\hat{v}) + \epsilon$, $\epsilon \ge 0$, where $(\dot{u}, \dot{v})$ is a saddle point. Denote the set of ϵ-saddle points as*

$$\mathcal{G}_g(\epsilon) = \{(\hat{u}, \hat{v}) \in \mathcal{U} \times \mathcal{V} \mid g_v(\hat{u}) - \epsilon \le g(\dot{u}, \dot{v}) \le g_u(\hat{v}) + \epsilon\}. \tag{2.3}$$

It immediately follows that $\mathcal{G}(0) = \mathcal{G}$; that is, the most strict set of ϵ-saddle points is the set of 'true' saddle points $\mathcal{G}$. As ϵ increases, the qualification of inclusion is relaxed so we obtain the property that $\epsilon_1 \le \epsilon_2 \iff \mathcal{G}(\epsilon_1) \subseteq \mathcal{G}(\epsilon_2)$.

We will now turn our attention to the gradient properties of the saddle function. For any (perhaps non-smooth) saddle function $g : \mathcal{U} \subset \mathbb{R}^n \times \mathcal{V} \subset \mathbb{R}^m \mapsto \mathbb{R}$, we want to make analogies to the concepts of subgradients from convex analysis. We remind the reader that $g(u, v)$ is convex in u, for a fixed v. We define

$$\partial_u g(\hat{u}, \hat{v}) := \big\{d_u \in \mathbb{R}^n \mid g(u, \hat{v}) \ge g(\hat{u}, \hat{v}) + \langle d_u, u - \hat{u}\rangle, \ \forall u \in \mathcal{U}\big\}. \tag{2.4}$$

This multifunction is a mapping from a point in $\mathcal{U} \times \mathcal{V}$ to the set of subgradients of the convex function $g(\cdot, \hat{v}) : \mathcal{U} \subset \mathbb{R}^n \mapsto \mathbb{R}$ at $\hat{u}$. Because $g(\cdot, v)$ is convex in u we retain all the usual properties of subgradients for wholly convex functions such as convexity and compactness of $\partial_u g(\hat{u}, \hat{v})$ over $\mathcal{U} \times \mathcal{V}$. Of course, the analogous mapping exists in the vector space where $g(u, \cdot)$ is concave as

$$\partial_v g(\hat{u}, \hat{v}) := \big\{d_v \in \mathbb{R}^m \mid g(\hat{u}, v) \le g(\hat{u}, \hat{v}) + \langle d_v, v - \hat{v}\rangle, \ \forall v \in \mathcal{V}\big\}. \tag{2.5}$$

Let us denote the elements of this set as $d_v(\hat{u}, \hat{v})$. If $g(\cdot, \cdot)$ is differentiable everywhere, then we have that $\partial_u g(\hat{u}, \hat{v})$ is a singleton and $\partial_u g(\hat{u}, \hat{v}) = D_u g(\hat{u}, \hat{v})$.

2.2 Lipschitz considerations

Here we present and extend the notion of Lipschitz continuity of saddle functions. The convergence of our proposed algorithm relies on certain Lipschitz continuity features. First we present the usual definition of Lipschitz continiuty.

Definition 4. *We define a Lipschitz saddle function as a saddle function $g(u, v)$ such that there exists some $\alpha \in \mathbb{R}$ for which*

$$|g(u_1, v_1) - g(u_2, v_2)| \le \alpha ||(u_1, v_1) - (u_2, v_2)||, \quad \forall (u_1, v_1), (u_2, v_2) \in \mathcal{U} \times \mathcal{V}. \tag{2.6}$$

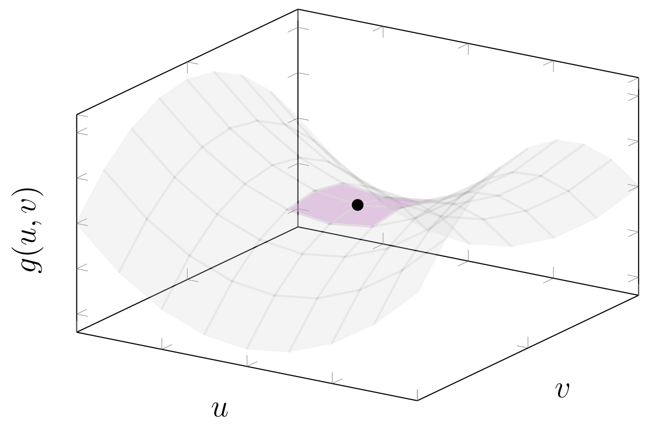


Figure 1: A saddle function $g(u,v)$ with $\mathcal{G}$ (a singleton in this instance) as the mark and $\mathcal{G}(\epsilon)$, $\epsilon > 0$ as the shaded region.

We say that the saddle function is α-Lipschitz if the definition above holds for the particular value α. We extend the above definition naturally to allow for different α values for the convex and concave domains.

Definition 5. *A (α_u, α_v)-Lipschitz saddle function is a saddle function $g(u,v)$ for which there exists some $\alpha_u, \alpha_v \in \mathbb{R}$ where*

$$|g(u_1,v) - g(u_2,v)| \leq \alpha_u ||(u_1,v) - (u_2,v)||, \quad \forall u_1,u_2 \in \mathcal{U}, \forall v \in \mathcal{V}, \tag{2.7}$$

$$|g(u,v_1) - g(u,v_2)| \leq \alpha_v ||(u,v_1) - (u,v_2)||, \quad \forall u \in \mathcal{U}, \forall v_1,v_2 \in \mathcal{V}. \tag{2.8}$$

We say a function is (α_u, α_v)-Lipschitz if the above definition holds for the two values α_u and α_v in their respective spaces. Clearly, if a function is (α,α)-Lipschitz, then it is α-Lipschitz. We formalise this with the following lemma.

Lemma 2.2. *There exists two constants α_u, α_v for which $g(u,v)$ is (α_u,α_v)-Lipschitz if and only if there exists a constant α for which $g(u,v)$ is α-Lipschitz.*

Proof. The backwards direction of the proof is trivial: from (2.6), picking $v_1 = v_2$ clearly satisfies (2.7). Further, picking $u_1 = u_2$ satisfies (2.8). We omit the forward direction of the proof for the purposes of brevity. $\square$

2.3 Saddle function estimates

Using the constructions defined above, we will define two functions $\bar{G}(u,v)$ and $\underset{\sim}{G}(u,v)$ that bound a given saddle function $g(u,v)$. We will then show how these functions play a role in an algorithm for computing an ϵ-saddlepoint of the given saddle function. Formally, the problem for consideration in this section is the following: for a given saddle function $g(u,v)$ we wish to compute a point $(\dot{u},\dot{v})$ such that

$$(\dot{u},\dot{v}) \in \mathcal{G}_g(\epsilon) \subseteq \mathcal{U} \times \mathcal{V} \tag{2.9}$$

where $\mathcal{U} \subset \mathbb{R}^n$ and $\mathcal{V} \subset \mathbb{R}^m$ are both convex and compact. The set $\mathcal{G}(\epsilon)$ is the set of ϵ-saddle points for the saddle function $g(u,v)$. Further, we require that the saddle function is Lipschitz continuous on $\mathcal{U} \times \mathcal{V}$.

Let $[\mathcal{U}\times\mathcal{V}]$ denote the set of all finite subsets of $\mathcal{U}\times\mathcal{V}$. For a given set of sampled points $S \in [\mathcal{U}\times\mathcal{V}]$, we are interested in forming bounding functions, $\bar{G}^S : \mathcal{U}\times\mathcal{V} \mapsto \mathbb{R}$, which have the following properties:

$$S_1 \subseteq S_2 \implies \bar{G}^{S_1}(u,v) \geq \bar{G}^{S_2}(u,v) \quad \forall (u,v) \in \mathcal{U}\times\mathcal{V}, \tag{2.10}$$

$$\bar{G}^S(u,v) \geq g(u,v) \quad \forall (u,v) \in \mathcal{U}\times\mathcal{V},\ \forall S \in [\mathcal{U}\times\mathcal{V}], \tag{2.11}$$

$$\bar{G}^S(\bar{u},\bar{v}) = g(\bar{u},\bar{v}) \quad \forall (\bar{u},\bar{v}) \in S,\ \forall S \in [\mathcal{U}\times\mathcal{V}], \tag{2.12}$$

where $\bar{G}^S$ is the upper bound function using information from a set of sampled points S. Much like the cutting plane methods from convex optimisation, we will make use of first-order and zeroth-order information to form their bound.

For a given sample point $(\hat{u},\hat{v})$, let us define the function $\bar{h}^{\hat{u},\hat{v}} : \mathbb{R}^m \mapsto \mathbb{R}$ as

$$\bar{h}^{\hat{u},\hat{v}}(v) := g(\hat{u},\hat{v}) + \big\langle d_v(\hat{u},\hat{v}), (v-\hat{v})\big\rangle. \tag{2.13}$$

We note here that the uniqueness of $d_v(\hat{u},\hat{v})$ is not required in the subsequent proofs; without loss of generality we refer to an arbitrary element $d_v(\hat{u},\hat{v}) \in \partial_v g(\hat{u},\hat{v})$.

Lemma 2.3.

$$\bar{h}^{\hat{u},\hat{v}}(v) \geq g(\hat{u},v),\ \forall v \in \mathcal{V},\ \forall (\hat{u},\hat{v}) \in \mathcal{U}\times\mathcal{V}. \tag{2.14}$$

Proof. This follows directly from the definition of $d_v(\hat{u},\hat{v})$. □

Using these cutting planes defined by $\bar{h}$ on their respective spaces, we are now in a position to define upper and lower bounding functions for a given saddle function g.

Definition 6.

$$\begin{aligned} \bar{G}^S(u,v) = \max_{\mu,\lambda} \quad & \mu + \langle \lambda, u\rangle \\ \text{s.t.} \quad & \mu + \langle \lambda, \bar{u}\rangle \leq \bar{h}^{\bar{u},\bar{v}}(v) \quad \forall (\bar{u},\bar{v}) \in S, \qquad (*) \\ & ||\lambda||_q \leq \alpha_u. \end{aligned} \tag{2.15}$$

This upper bound function is similar to that introduced in [Baucke et al., 2017], but rather than the right-hand side of constraint $(*)$ being a constant, it is now an affine function of v. Immediately we can see that $\bar{G}^S(u,v)$ is a saddle function: being a linear programme (which is maximising) with v in its right-hand side and u in the objective function. The following lemmas establish that $\bar{G}^S(u,v)$ satisfies the required properties, given in (2.10)-(2.12), as well a Lipschitz-continuity result.

Lemma 2.4. *If $g(u,v)$ is a saddle-function and (α_u,α_v)-Lipschitz on $\mathcal{U}\times\mathcal{V}$ under the $||\cdot||_p$ norm, then $\bar{G}^S(u,v)$ is (α_u,α_v)-Lipschitz.*

Proof. For all $v \in \mathcal{V}$, the function $\bar{G}^S(u,v)$ is clearly α_u-Lipschitz. For all u, the value of $|\bar{G}^S(u,v_1) - \bar{G}^S(u,v_2)|$ is bounded by

$$\max_{(\hat{u},\hat{v})\in S} \big\langle d_v(\hat{u},\hat{v}), (v_1 - v_2)\big\rangle. \tag{2.16}$$

The function $g(u,\cdot)$ is α_v-Lipschitz so it has uniformly bounded sub-gradients (under the $||\cdot||_q$-norm, where $\frac{1}{p}+\frac{1}{q} = 1,\ \forall p,q \in (1,\infty)$). So $||d_v(\hat{u},\hat{v})||_q \leq \alpha_v,\ \forall (\hat{u},\hat{v}) \in \mathcal{U}\times\mathcal{V}$. This gives the result. □

Lemma 2.5. *If $g(u,v)$ is a saddle-function and (α_u,α_v)-Lipschitz on $\mathcal{U}\times\mathcal{V}$ under the $||\cdot||_p$ norm, and $(\hat{u},\hat{v})\in S$, then $\bar{G}^S(\hat{u},\hat{v})=g(\hat{u},\hat{v})$.*

Proof. For a given S, the pair (μ,λ) in the feasible region of (2.15) represent the coefficients of a supporting hyperplane of the points $(\bar{u},\bar{h}^{\bar{u},\bar{v}}(\hat{v}))$, $\forall(\bar{u},\bar{v})\in S$ where μ is the intercept and λ is the gradient of the hyperplane. We note here that the feasible region is always non-empty since

$$\mu=\min_{(\bar{u},\bar{v})\in S}\bar{h}^{\bar{u},\bar{v}}(\hat{v}),\ \text{and}\ \lambda=\mathbf{0}, \tag{2.17}$$

is always feasible. Because the function $g(u,v)$ is convex in u and (α_u,α_v)-Lipschitz, there exists a supporting hyperplane

$$g(\hat{u},\hat{v})+\langle d_u,u-\hat{u}\rangle\le g(u,\hat{v}),\ \forall u\in\mathcal{U}, \tag{2.18}$$

where $||d_u||_q\le\alpha_u$ and d_u is in the set of sub-gradients of $g(u,v)$ at $(\hat{u},\hat{v})$. So a candidate solution for $\bar{G}^S(\hat{u},\hat{v})$ is $\tilde{\mu}=g(\hat{u},\hat{v})-\langle d_u,\hat{u}\rangle$ and $\tilde{\lambda}=d_u$ which gives $\bar{G}^S(\hat{u},\hat{v})=g(\hat{u},\hat{v})$. This solution is feasible because if the plane supports $(\bar{u},g(\bar{u},\hat{v}))$, $\forall(\bar{u},\bar{v})\in S$, then by the concavity of $g(u,v)$ in v and Lemma 2.3, it also supports $(\bar{u},\bar{h}^{\bar{u},\bar{v}}(\hat{v}))$, $\forall(\bar{u},\bar{v})\in S$. If another candidate (μ,λ) gives $\mu+\langle\lambda,\hat{u}\rangle<g(\hat{u},\hat{v})$, then the pair is not optimal because of the existence of $(\tilde{\mu},\tilde{\lambda})$. If a further candidate (μ,λ) gives $\mu+\langle\lambda,\hat{u}\rangle>g(\hat{u},\hat{v})$, then the pair does not support the point $(\hat{u},\bar{h}^{\hat{u},\hat{v}}(\hat{v}))$, making it infeasible. This completes the proof. □

Lemma 2.6. *If $S^1\subseteq S^2$, then $\bar{G}^{S^1}(u,v)\ge\bar{G}^{S^2}(u,v)$.*

Proof. The function $\bar{G}^S(u,v)$ is defined by a maximisation problem which is feasible and bounded. $S^1\subseteq S^2$ implies that the feasibility set of S^2 is contained within S^1, giving the result. □

Lemma 2.7. *If $g(u,v)$ is a saddle function and α-Lipschitz on $\mathcal{U}\times\mathcal{V}$ under the $||\cdot||_p$ norm, then $\bar{G}^S(u,v)\ge g(u,v)$ for all $(u,v)\in\mathcal{U}\times\mathcal{V}$.*

Proof. For the sake of contradiction, suppose there exists a $(\tilde{u},\tilde{v})\in\mathcal{U}\times\mathcal{V}$ for which $\bar{G}^S(\tilde{u},\tilde{v})<g(\tilde{u},\tilde{v})$. By defining $S^*=S\cup(\tilde{u},\tilde{v})$, and by Lemma 2.5, we have $g(\tilde{u},\tilde{v})=\bar{G}^{S^*}(\tilde{u},\tilde{v})$. So $\bar{G}^S(\tilde{u},\tilde{v})<\bar{G}^{S^*}(\tilde{u},\tilde{v})$, which is precluded by Lemma 2.6, completing the proof. □

The following lemma establishes a monotonicity result with respect to the bounding functions' saddle point.

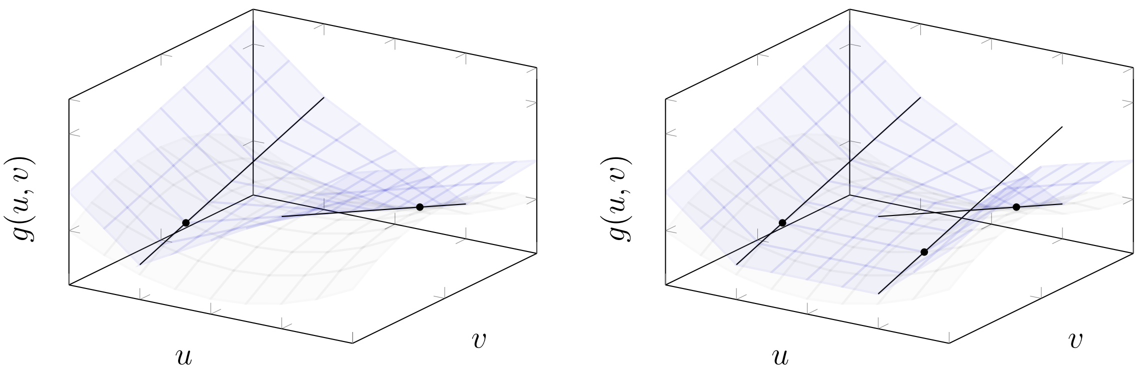
Lemma 2.8.

$$\min_{u\in\mathcal{U}}\bar{G}^S(u,v)\ge\min_{u\in\mathcal{U}}g(u,v),\quad\forall v\in\mathcal{V}. \tag{2.19}$$

Proof. Suppose the statement of this lemma is false, and therefore there exists a $\tilde{v}\in\mathcal{V}$ for which $\min_{u\in\mathcal{U}}\bar{G}^S(u,\tilde{v})<\min_{u\in\mathcal{U}}g(u,\tilde{v})$. Now define $\bar{u}$ as an arbitrary minimiser of $\bar{G}^S(u,\tilde{v})$; this gives the following inequality:

$$\bar{G}^S(\bar{u},\tilde{v})<\min_{u\in\mathcal{U}}g(u,\tilde{v})\le g(\bar{u},\tilde{v}). \tag{2.20}$$

By defining $S^*=S\cup(\bar{u},\tilde{v})$, and by Lemma 2.5, we have $g(\bar{u},\tilde{v})=\bar{G}^{S^*}(\bar{u},\tilde{v})$. So $\bar{G}^S(\bar{u},\tilde{v})<\bar{G}^{S^*}(\bar{u},\tilde{v})$, which is precluded by Lemma 2.6, completing the proof. □



(a) $|\bar{S}_1| = 2$ (b) $|\bar{S}_2| = 3$ and $S_1 \subset S_2$

Figure 2: For the saddle function $g(u, v)$ in grey, the blue surfaces are the upper bounding function $\bar{G}(u, v)$.

Corollary 2.1.

$$\max_{v\in\mathcal{V}} \bar{G}^S(u,v) \geq \max_{v\in\mathcal{V}} g(u,v), \quad \forall u \in \mathcal{U}. \tag{2.21}$$

Corollary 2.2.

$$\max_{v\in\mathcal{V}} \min_{u\in\mathcal{U}} \bar{G}^S(u,v) \geq \max_{v\in\mathcal{V}} \min_{u\in\mathcal{U}} g(u,v). \tag{2.22}$$

We note here that the lemmas established above hold symmetrically for the lower bounding function $\underline{G}^S(u, v)$.

We now proceed to outline an algorithm that utilises the bounding functions described above. The algorithm can be considered as the saddlepoint analogy of Kelly's Cutting Plane algorithm. We require the conditions set out in (2.9) on $g(u, v)$ and $\mathcal{U}$ and $\mathcal{V}$ for the following lemma.

Theorem 2.1. *Consider the sequences*

$$u^k = \arg\min_{u\in\mathcal{U}} \max_{v\in\mathcal{V}} \underline{G}^{S_{k-1}}(u,v), \tag{2.23}$$

$$v^k = \arg\max_{v\in\mathcal{V}} \min_{u\in\mathcal{U}} \bar{G}^{S_{k-1}}(u,v), \tag{2.24}$$

$$S_k = S_{k-1} \cup (u^k, v^k),$$

with

$$\bar{g}^k = \min_{u\in\mathcal{U}} \max_{v\in\mathcal{V}} \bar{G}^{S_{k-1}}(u,v), \ \underline{g}^k = \min_{u\in\mathcal{U}} \max_{v\in\mathcal{V}} \underline{G}^{S_{k-1}}(u,v). \tag{2.25}$$

We have that

$$\lim_{k\to\infty} \left(\bar{g}^k - \underline{g}^k\right) = 0.$$

We will use $\bar{G}^k(u, v)$ as a notational shorthand for $\bar{G}^{S_k}(u, v)$. We proceed with the proof of Theorem 2.1.

Proof. From (2.23) and (2.25), we have

$$\underline{g}^k \geq \underline{G}^k(u^k, v), \quad \forall k, \forall v \in \mathcal{V}.$$

For $\bar{g}^k$, from (2.25) we have

$$\bar{g}^k \leq \bar{G}^k(u, v^k), \quad \forall k, \forall u \in \mathcal{U}.$$

It follows then that,

$$\bar{g}^k - \underline{g}^k \leq \bar{G}^k(u^k, v^k) - \underline{G}^k(u^k, v^k), \ \forall k. \tag{2.26}$$

We will now show that the RHS of (2.26) converges to 0 as k tends to infinity – this will complete the proof. Suppose there exist some $\epsilon > 0$ for which

$$\epsilon \leq \bar{G}^k(u^k, v^k) - \underline{G}^k(u^k, v^k), \ \forall k.$$

Subtracting $\bar{G}^k(u^{\hat{k}}, v^{\hat{k}}) - \underline{G}^k(u^{\hat{k}}, v^{\hat{k}})$ from both sides gives

$$\epsilon - \bar{G}^k(u^{\hat{k}}, v^{\hat{k}}) + \underline{G}^k(u^{\hat{k}}, v^{\hat{k}}) \leq$$
$$\bar{G}^k(u^k, v^k) - \underline{G}^k(u^k, v^k) - \bar{G}^k(u^{\hat{k}}, v^{\hat{k}}) + \underline{G}^k(u^{\hat{k}}, v^{\hat{k}}), \ \forall \hat{k}, k.$$

From Lemma 2.2, there exists a constant α for which both $\underline{G}^k$ and $\bar{G}^k$ are α-Lipschitz. So

$$\epsilon - \bar{G}^k(u^{\hat{k}}, v^{\hat{k}}) + \underline{G}^k(u^{\hat{k}}, v^{\hat{k}}) \leq 2\alpha||(u^k, v^k) - (u^{\hat{k}}, v^{\hat{k}})||, \ \forall \hat{k}, k.$$

From Lemma 2.5, $\bar{G}^k(u^{\hat{k}}, v^{\hat{k}}) - \underline{G}^k(u^{\hat{k}}, v^{\hat{k}}) = 0, \ \forall \hat{k} < k$ so,

$$\frac{\epsilon}{2\alpha} \leq ||(u^k, v^k) - (u^{\hat{k}}, v^{\hat{k}})||, \ \forall \hat{k} < k, \forall k,$$

which implies there exists no convergent subsequence of iterates (u^k, v^k). This is a contradiction of the compactness of $\mathcal{U} \times \mathcal{V}$. It must follow then, that no $\epsilon > 0$ exists and $\bar{g}^k - \underline{g}^k \to 0$ as $k \to \infty$. □

Unlike traditional optimisation, where simply a minimum or maximum is sought, in saddlepoint optimisation, the function $g(u, v)$ can take the value of the saddlepoint in several places without satisfying the saddlepoint conditions i.e. Definition 2. Convergence of value functions as their saddle-points is not sufficient to say that a saddlepoint has been located. We need to show that the algorithm converges to the saddlepoint value, and that iterates converge toward a saddlepoint as in Definition 3. The following two lemmas give the desired result.

Lemma 2.9.

$$\epsilon \geq g_v(\bar{u}) - g_u(\bar{v}) \implies (\bar{u}, \bar{v}) \in \mathcal{G}_g(\epsilon), \quad \forall (\bar{u}, \bar{v}) \in \mathcal{U} \times \mathcal{V}.$$

Proof. Restating Definition 3, we have

$$\mathcal{G}(\epsilon) = \{(\hat{u}, \hat{v}) \in \mathcal{U} \times \mathcal{V} \mid g_v(\hat{u}) - \epsilon \leq g(\dot{u}, \dot{v}) \leq g_u(\hat{v}) + \epsilon\}.$$

Substituting $\epsilon \geq g_v(\bar{u}) - g_u(\bar{v})$ into the definition yields

$$g_u(\bar{v}) \leq g(\dot{u}, \dot{v}) \leq g_v(\bar{u}),$$

which is true for all $(\bar{u}, \bar{v}) \in \mathcal{U} \times \mathcal{V}$, completing the proof. □

Lemma 2.10. *Consider the sequence of functions $\bar{G}^k$ and $\underline{G}^k$ generated by the algorithm. The set*

$$\mathcal{G}_g^k := \left\{ \{ \underset{u \in \mathcal{U}}{\arg\min} \max_{v \in \mathcal{V}} \bar{G}^{k-1}(u, v) \times \mathcal{V} \} \cap \{ \underset{v \in \mathcal{V}}{\arg\max} \min_{u \in \mathcal{U}} \underline{G}^{k-1}(u, v) \times \mathcal{U} \} \right\} \subseteq \mathcal{G}_g(\bar{g}^k - \underline{g}^k). \tag{2.27}$$

Proof. From Lemma 2.9, for a point $(\tilde{u}, \tilde{v})$ to be a member of $\mathcal{G}_g(\epsilon)$, we require that ϵ satisfies

$$\epsilon \geq g_v(\tilde{u}) - g_u(\tilde{v}). \tag{2.28}$$

From Corollary 2.1, we have that $\bar{g}^k \geq g_v(u),\ \forall u \in \arg\min_{u\in\mathcal{U}} \max_{v\in\mathcal{V}} \bar{G}^{k-1}(u,v)$ and $\underline{g}^k \leq g_u(v),\ \forall v \in \arg\max_{v\in\mathcal{V}} \min_{u\in\mathcal{U}} \underline{G}^{k-1}(u,v)$. So clearly,

$$\bar{g}^k - \underline{g}^k \geq g_v(\tilde{u}) - g_u(\tilde{v}), \quad \forall (\tilde{u}, \tilde{v}) \in \mathcal{G}_g^k, \tag{2.29}$$

completing the proof. $\square$

Remark 2.1. *From Theorem 2.1, we have that $\bar{g}^k - \underline{g}^k$ approaches 0, so $\mathcal{G}_g^k$ approaches the set of true saddle points.*

In review of this section, we have developed a pair of bounding functions which when improved under our algorithm, converge to the saddle function value – and iterates converge to a point in $\mathcal{G}_g(0)$.

3 Discussion

In this section, we extend the ideas in the previous section to the more complex setting of multistage minimax optimisation. As with SDDP, we decompose the problem into stages and construct upper and lower approximations to the value functions. We show by induction that nested value function approximations (both upper and lower) converge at the state trajectories generated by the algorithm.

Consider the set of dynamic programming equations whose elements are indexed by $t \in \{0, \dots, T-1\}$ and are given by:

$$\begin{aligned} G_t(x_t, y_t) = \min_{x_{t+1}, u_t} \max_{y_{t+1}, v_t} \quad & C_t(x_t, y_t, u_t, v_t) + G_{t+1}(x_{t+1}, y_{t+1}) \\ \text{s.t.} \quad & x_{t+1} = f_t^x(x_t, u_t) \\ & y_{t+1} = f_t^y(y_t, v_t) \\ & (x_{t+1}, y_{t+1}) \in \mathcal{X}_{t+1} \times \mathcal{Y}_{t+1} \\ & (u_t, v_t) \in \mathcal{U}_t(x_t) \times \mathcal{V}_t(y_t) \end{aligned} \tag{3.1}$$

with the final value function $G_T(x_T, y_T)$ is given. We are concerned with evaluating $G_0(x_0, y_0)$ for a given *initial state* (x_0, y_0). We require several technical conditions of the functions and multifunctions that make up the dynamic programming equations. Briefly, these conditions ensure that each sub-problem is convex and admits subgradients.

The algorithm works by progressively refining both the upper and lower bound functions by a series of *samples* and *updates*. We will give a high level demonstration of an iteration (indexed by k) of our algorithm. Because the final value function is given, we set $\underline{G}_T^k = G_T, \bar{G}_T^k = G_T,\ \forall k \in \mathbb{N}$.

Starting at $t = 0$, solve

$$\begin{aligned} \underline{\theta}_t^k = \min_{x_{t+1}, u_t} \max_{y_{t+1}, v_t} \quad & C_t(x_t^k, y_t^k, u_t, v_t) + \underline{G}_{t+1}^{k-1}(x_{t+1}, y_{t+1}) \\ \text{s.t.} \quad & x_{t+1} = f_t^x(x_t^k, u_t) \\ & y_{t+1} = f_t^y(y_t^k, v_t) \\ & (x_{t+1}, y_{t+1}) \in \mathcal{X}_{t+1} \times \mathcal{Y}_{t+1} \\ & (u_t, v_t) \in \mathcal{U}_t(x_t^k) \times \mathcal{V}_t(y_t^k), \end{aligned} \tag{3.2}$$

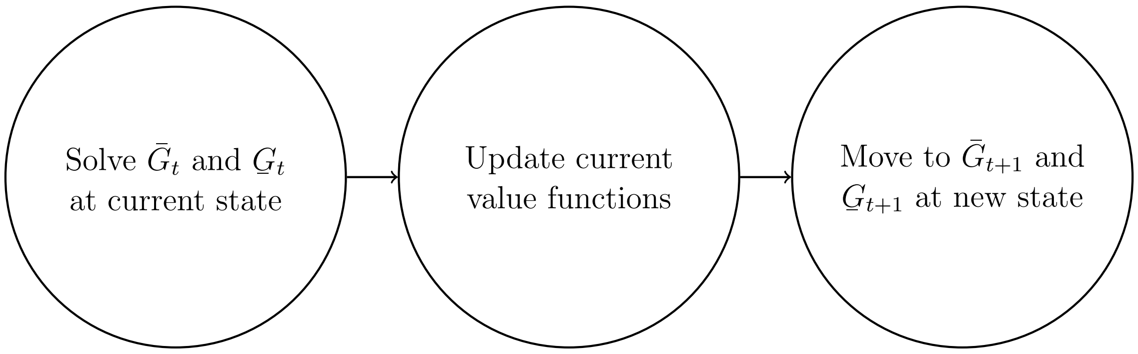

Solve $\bar{G}_t$ and $\underline{G}_t$ at current state
Update current value functions
Move to $\bar{G}_{t+1}$ and $\underline{G}_{t+1}$ at new state

Figure 3: High-level algorithm

storing (x_{t+1}^k, u_t^k) as the minimisers. Compute the sub gradients (β_t^k) with respect to x_t^k, then define the lower bound value function as

$$\begin{aligned}\underline{G}_t^k(x,y) = \min_{\mu,\lambda} \quad & \mu + \langle \lambda, y\rangle \\ \text{s.t.} \quad & \mu + \langle \lambda, y_t^k\rangle \geq \underline{\theta}_t^k + \langle \beta_t^k, x - x_t^k\rangle \\ & \mu + \langle \lambda, y_t^{\hat{k}}\rangle \geq \underline{\theta}_t^{\hat{k}} + \langle \beta_t^{\hat{k}}, x - x_t^{\hat{k}}\rangle, \quad \forall \hat{k} < k \\ & ||\lambda|| \leq \alpha_y.\end{aligned} \tag{3.3}$$

Next solve

$$\begin{aligned}\bar{\theta}_t^k = \max_{y_{t+1},v_t} \min_{x_{t+1},u_t} \quad & C_t(x_t, y_t, u_t, v_t) + \bar{G}_{t+1}^{k-1}(x_{t+1}, y_{t+1}) \\ \text{s.t.} \quad & x_{t+1} = f_t^x(x_t^k, u_t) \\ & y_{t+1} = f_t^y(y_t^k, v_t) \\ & (x_{t+1}, y_{t+1}) \in \mathcal{X}_{t+1} \times \mathcal{Y}_{t+1} \\ & (u_t, v_t) \in \mathcal{U}_t(x_t^k) \times \mathcal{V}_t(y_t^k),\end{aligned} \tag{3.4}$$

storing (y_{t+1}^k, v_t^k) as the maximisers. Compute the sub gradients (γ_t^k) with respect to y_t^k, then define the upper bound value function as

$$\begin{aligned}\bar{G}_t^k(x,y) = \max_{\mu,\lambda} \quad & \mu + \langle \lambda, x\rangle \\ \text{s.t.} \quad & \mu + \langle \lambda, x_t^k\rangle \leq \bar{\theta}_t^k + \langle \gamma_t^k, y - y_t^k\rangle \\ & \mu + \langle \lambda, x_t^{\hat{k}}\rangle \leq \bar{\theta}_t^{\hat{k}} + \langle \gamma_t^{\hat{k}}, y - y_t^{\hat{k}}\rangle, \quad \forall \hat{k} < k \\ & ||\lambda|| \leq \alpha_x.\end{aligned} \tag{3.5}$$

Repeat the above with $t \leftarrow t + 1$ until $t = T - 1$. That concludes an iteration of the algorithm. Figure 3 gives a diagrammatic representation of an iteration.

The proof of convergence of our algorithm seeks to construct a similar contradiction to Theorem 2.1. However, simply applying Theorem 2.1 at every stage is not sufficient for convergence as the theorem requires that the proceeding value functions to have converged; this is only true at stage T. Our proof relies on backward induction with convergence of the value functions at stage T forming the *base case*. Each preceding pair of bounds will converge in the limit as long as the future bounds converge in the limit.

Figure 4 shows how nested representations of value functions compound the *bound gap*. Notice the bound gap between the bounding functions is necessarily

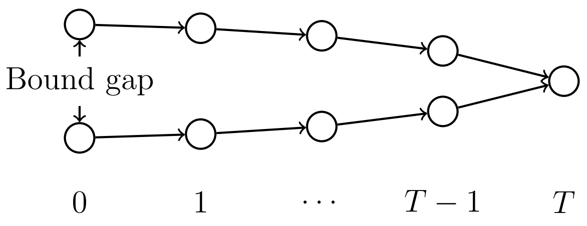


Figure 4: Nested bounding functions

zero at the final stage; this is because both the upper and lower estimations are equal to the true value function at this point. After each iteration of the algorithm, we are able to close the bound gap at all vertices until the gap is sufficiently small at $t = 0$ as in Figure 5.

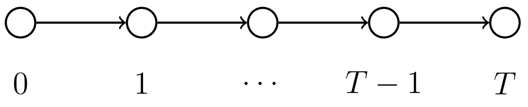


Figure 5: Converged bound gap

As a final note, we can further adapt this algorithm for multistage *stochastic* minimax problems whose dynamic programming equations are given by

$$\begin{aligned} G_n(x_n, y_n) = \min_{x_m, u_n} \max_{y_m, v_n} \quad & C_n(x_t, y_t, u_t, v_t) + \mathbb{E}[G_m(x_m, y_m)] \\ \text{s.t.} \quad & x_m = f_m^x(x_n, u_n), \quad \forall m \in R(n), \\ & y_m = f_m^y(y_n, v_n), \quad \forall m \in R(n), \\ & (x_m, y_m) \in \mathcal{X}_m \times \mathcal{Y}_m, \quad \forall m \in R(n), \\ & (u_n, v_n) \in \mathcal{U}_n(x_n) \times \mathcal{V}_n(y_n). \end{aligned} \tag{3.6}$$

Various sampling techniques exists to solve these problems; with an SDDP-like random sampling technique, we can obtain convergence *w.p.*1, while with the sampling technique used in [Baucke et al., 2017], we can achieve deterministic convergence.

References

Regan Baucke, Anthony Downward, and Golbon Zakeri. A deterministic algorithm for solving multistage stochastic programming problems. 2017. URL `http://www.optimization-online.org/DB_FILE/2017/07/6138.pdf`.

J.E. Kelley. The cutting-plane method. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

A. B. Philpott and Z. Guan. On the convergence of stochastic dual dynamic programming and related methods. *Operations Research Letters*, 36(4):450–455, 2008. ISSN 01676377. doi: 10.1016/j.orl.2008.01.013.