Solving Eikonal Equations using Djikstra's algorithm

Srivatsa Srinivas

1 The Hamilton-Jacobi-Bellman equation

The purpose of this section is to define the Hamilton-Jacobi-Bellman equation. The Hamilton-Jacobi-Bellman (HJB for short) equation describes the way to choose the next "direction" that an agent must go in order to minimize a "loss-function". We will introduce the data required to define the HJB equation. We will use conventions from the document [Cal24], mixed with our own.

Definition 1 (HJB triple)**.** *An HJB triple is a triple (L, U, g) where $U \subset \mathbb{R}^n$ is open and bounded, g is a continous function on ∂U and $L : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ is a continous function such that*

1. *(1-Positive Homogeneity): We have for all $\alpha > 0$ that $L(\alpha x, y) = \alpha L(x, y)$*
2. *(1-Positivity): We have that for all $p, x \in \mathbb{R}^n$ that if $p \neq 0$ then $L(p, x) > 0$.*

Definition 2 (Cost and Value functions)**.** *Let (L, U, g) be an HJB triple*

1. *(Cost Function). For any $x, y \in \mathbb{R}^n$ we define*
$$\mathrm{Cost}(x, y) := \inf_{w} \int_0^1 L(w'(t), w(t))\, dt$$
where $w : [0, 1] \to \mathbb{R}^n$ ranges over all continous functions from $[0, 1]$ to $\mathbb{R}^n$ such that $w(0) = x, w(1) = y$.
2. *(Value function). For any $x \in \mathbb{R}^n$ we define*
$$u(x) := \inf_{y \in \partial U} (g(y) + \mathrm{Cost}(y, x))$$
3. *(Compatibility). We say that (L, U, g) is compatible if for every $x, y \in \partial U$ we have that*
$$g(y) - g(x) \leq \mathrm{Cost}(y, x)$$

From this point of the document onwards, we **assume** that every HJB triple is compatible

Example 1 (Distance Function)**.** *If $L : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is defined by*
$$L(x, y) := \|\pi_L(x, y)\|_2 = \|x\|_2$$
and $U \subset \mathbb{R}^n$ is bounded, and $g : \partial U \to \mathbb{R}$ is just the 0 function i.e, $g(x) = 0(x) = 0$, then we have that $(L, U, 0)$ is an HJB triple with cost function
$$\mathrm{Cost}(x, y) = \|x - y\|_2$$
and value function
$$u(x) = d_{\partial U}(x)$$
where $d_A(x)$ is the minimum distance of x from the set A.

The following lemma is similar to Lemma 4.3 and 4.4 in [Cal24]

Lemma 1 (Locality and Locally Lipschitz properties of value function)**.** *Let (L, U, g) be an HJB triple*

Submitted as final project to MATH 273 B, Spring 2024

1. *(Locally Lipschitz). We have that the value function $u : \mathbb{R}^n \to \mathbb{R}$ is Locally Lipschitz.*
2. *(Locality). We have that for all $x \in \mathbb{R}^n, r > 0$ that*
$$u(x) = \inf_{y \in B(x,r) \setminus \{x\}} u(y) + \mathrm{Cost}(y, x)$$

The above lemma tells us that if the value function, u, of an HJB triple (L, U, g) is differentiable at a point x_0 then we have that
$$0 = \sup_{\|a\|_2 = 1} (-\nabla u \cdot a - L(a, x))$$

Definition 3 (HJB equation)**.** *Let (L, U, g) be an HJB triple*

1. *(Hamiltonian). Given a point $x_0 \in \mathbb{R}^n$ and a function ϕ that is differentiable at x_0 we define the Hamiltonian, H to be*
$$H(\phi, x_0) := \sup_{\|a\|_2 = 1} (-\nabla \phi \cdot a - L(a, x))$$
2. *(HJB equation). We say that a function ϕ which is continous and piecewise differentiable on $\mathbb{R}^n$ satisfies the HJB equation if for all points x at which it is differentiable we have that*
$$H(\phi, x) = 0$$
and for all $x_0 \in \partial U$ we have that
$$\phi(x_0) = g(x_0)$$

Thus, we can now ask the question: Is it possible to recover the value function by finding a solution to the HJB equation of a triple? Let us look at the example of the HJB triple $(L(x, y), (0, 1), 0)$, with $L(x, y) = 1$. The value function, u, of this triple should be the function $d_{\{0,1\}} : \mathbb{R} \to \mathbb{R}$, ie the minimum distance from the set $\{0, 1\}$. This satisifies the equation
$$|du/dx|^2 - 1 = 0$$
and $u(0) = u(1) = 0$. But there are infinitely many solutions to the above equation which are piecewise differentiable. Is there a way to recover the value function from the above equation? In order to answer this, we need the notion of a viscosity solution.

Definition 4 (Viscosity Solution)**.** *Let (L, U, g) be an HJB triple with hamiltonian H. Let $V \subset \mathbb{R}^n$ be an open subset*

1. *(Viscosity Subsolution). We say that a function $v : V \to \mathbb{R}^n$ is a viscosity subsolution to H at a point $x_0 \in V$ if for all $\phi \in C^\infty(\mathbb{R}^n)$ such that $\phi(x_0) - v(x_0) = 0$ is a local maximum of $\phi - v$ we have that*
$$H(\phi, x_0) \leq 0$$
v is a viscosity subsolution to H on V if it is a viscosity subsolution to H for all $x_0 \in V$.
2. *(Viscosity Supersolution). We say that a function $v : V \to \mathbb{R}^n$ is a viscosity supersolution to H at a point $x_0 \in V$ if for all $\phi \in C^\infty(\mathbb{R}^n)$ such that $\phi(x_0) - v(x_0) = 0$ is a local minimum of $\phi - v$ we have that*
$$H(\phi, x_0) \geq 0$$
v is a viscosity supersolution to H on V if it is a viscosity supersolution to H for all $x_0 \in V$.
3. *(Viscosity Solution). A function $v : V \to \mathbb{R}$ is a viscosity solution to H if it is both a viscosity supersolution and subsolution to H on V*

The following theorem appears as Theorem 4.6 in [Cal24].

Theorem 1 (Recovering the value function)**.** *Given an HJB triple (L, U, g) with hamiltonian H, there is a unique function v such that v is a viscosity solution to H on $\mathbb{R}^n \setminus \partial U$ and is equal to g on ∂U. This function v is equal to the value function of the HJB triple, u. We say that u is the viscosity solution to the HJB equation of the triple (L, U, g)*

Thus we know that if we find the viscosity solution to an HJB equation, we find it's value function. But is there a way we can do this efficiently? The rest of this article is dedicated to using the "Generalized Djikstra's algorithm" to efficiently estimate the value function of special HJB triples on a box $[a_1, b_1] \times \cdots \times [a_n, b_n]$

2 The generalized Djikstra's algorithm

For this section we will define $\mathbb{K} := \mathbb{R} \cup \{\infty\}$. Also given, a graph $G = (V, E)$, we define the neigbor function $N : V \to 2^V$ by

$$N(v) := \{w \mid \{w, v\} \in E\}$$

The degree function $d : V \to \mathbb{N}$ is defined by

$$d(v) := |N(v)|$$

Definition 5. *Given a graph $G = (V, E)$ we say that a function $\Phi : (v : V) \to \mathbb{K} \times \mathbb{K}^{N(v)} \to \mathbb{K}$ is a **monotone local constraint** we have that following hold:*

- *Monotonocity: For all $v \in V$ we have that $\Phi(v)$ is continous, piecewise differentiable and*

  $$\partial\Phi(v)(t_0, r_1, \ldots, r_{d(v)})/\partial t \geq 0$$

  and for all $n_i \in N(v)$ we have that

  $$\partial\Phi(v)(t_0, r_1, \ldots, n_{d(v)})/\partial n_i \leq 0$$

  for every $(t_0, r_1, \ldots, r_{d(v)}) \in \mathbb{R} \times \mathbb{R}^{N(v)}$ at which the corresponding partial derivatives exist, where $(t, n_1, \ldots, n_{d(v)})$ are the corresponding co-ordinates in $\mathbb{K} \times \mathbb{K}^{N(v)}$.

- *Causality: For all $v \in V$ we have that if*

  $$\Phi(v)(t_0, r_1, \ldots, r_{d(v)}) = 0$$

  Then for all $1 \leq i \leq d(v)$ such that $r_i > t_0$ we have that

  $$\Phi(v)(t_0, \underbrace{r_1, \ldots, a}_{i}, \ldots, r_{d(v)}) = 0$$

  where $a \geq t_0$.

- *Solvability: For any $v \in V$ if one of $r_1, \ldots, r_{d(v)}$ is not equal to ∞ then we have that there exists a unique $t_0 \in \mathbb{R}$ such that*

  $$\Phi(v)(t_0, r_1, \ldots, r_{d(v)}) = 0$$

Definition 6 (Update function)**.** *Given a graph $G = (V, E)$ and a monotne local constraint Φ we define the update function to be a function $\mathrm{Upd}_\Phi : (V \to \mathbb{K}) \to (V \to \mathbb{K})$ defined by*

$$\mathrm{Upd}_\Phi(f)(v) := \begin{cases} \min\{f(v), t_0\} & \exists 1 \leq i \leq d(v).\, f(n_i) < \infty \\ \infty & \textit{else} \end{cases}$$

where $N(v) = \{n_1, \ldots, n_{d(v)}\}$ and $\Phi(v)(t_0, f(n_1), \ldots, f(n_{d(v)})) = 0$

Definition 7 (Fixed Points)**.** *Given a graph $G = (V, E)$, monotone local update Φ and a function $f : V \to \mathbb{K}$ we define*

$$\mathrm{Fix}(f) = \{v \in V \mid \mathrm{Upd}_\Phi^{(n)}(f)(v) = f(v) \textit{for all } n \in \mathbb{N}\}$$

where for any function $g : A \to A$ we define $g^{(0)} = id_A$, $g^{(n)} = g \circ g^{(n-1)}$.

Lemma 2 (Adding fixed points)**.** *Given a graph $G = (V, E)$, a monotone local update Φ, and a function $f : V \to \mathbb{K}$, let $A = \mathrm{Fix}(f)$ and $B = V \setminus \mathrm{Fix}(f)$. Define $g_k = \mathrm{Upd}_\Phi^{(k+1)}(f)$. Let $b^* \in B$ be such that*

$$s = g_0(b^*) = \min\{g_0(b) \mid b \in B\}$$

Then we have that $v \in \mathrm{Fix}(g_0)$.

Proof. We first prove the following claim

Claim. *For all $k \in \mathbb{N}, b \in B$ we have that $g_k(b) \geq s$*

Proof. The claim is definitionally true when $k = 0$. Thus we may assume that $k > 0$ and that the result holds for $0 \le k' < k$. Suppose that there exists a $b_0 \in B$ such that $g_k(b_0) < s$. Then since $g_{k-1}(b_0) \ge s$ we must have that $g_k(b_0) = \min\{g_{k-1}(b_0), t_0\} = t_0$ where

$$\Phi(b_0)(t_0, g_{k-1}(a_1), \dots, g_{k-1}(a_m), g_{k-1}(b_1), \dots, g_{k-1}(b_n)) = 0$$

where the a_i, b_j are the neighbors of b_0 that are in A and B respectively. Since $t_0 < s \le g_0(b_1), g_{k-1}(b_1) \dots, g_0(b_n), g_{k-1}(b_n)$, we have by causality and the fact that $a_i \in \mathrm{Fix}(f)$

$$\Phi(b_0)(t_0, f(a_1), \dots, f(a_m), g_0(b_1), \dots, g_0(b_n)) = 0$$

Since $\mathrm{Upd}_\Phi(h)(w) \le h(w)$ for all Φ, h, w we know that $t_0 < g_0(b_i) \le f(b_i)$. Applying causailty again we get that

$$\Phi(b_0)(t_0, f(a_1), \dots, f(a_m), f(b_1), \dots, f(b_n)) = 0$$

Thus we have that $g_0(b_0) = \min\{f(b_0), t_0\} = t_0 < s$, a contradiction by the definition of s. This proves the claim. □

By the claim we have that for all $k \in \mathbb{N}, b \in B$, we have that $\mathrm{Upd}_\Phi^{(k)}(g_0)(b) \ge s$. Thus we have that $s \le \mathrm{Upd}_\Phi^{(k)}(g_0)(b^*) \le s$ and we have finished the proof. □

Definition 8 (Priority Queue)**.** *A priority queue is a data structure that stores n points of the form (v, r) where r belongs to a type that is orderable. A priority queue is required to have a time complexity of $O(1)$ to return a point of the form (v_0, r_0) where r_0 is minimal amongst all the r such that there exists a (v, r) in the priority queue. A priority queue is also required to have a time complexity of $O(\log n)$ for inserting a new element (v, r) into the priority queue.*

Lemma 3 (Djikstra's algorithm)**.** *Let $G = (V, E)$ be a finite graph with monotone local constraint Φ. The following algorithm has a time complexity of $O(|V| \log |V|)$ that takes in a function $f : V \to \mathbb{K}$ and returns a function $f_\infty : V \to \mathbb{K}$ such that $\mathrm{Upd}_\Phi(f_\infty) = f_\infty = \mathrm{Upd}_\Phi^{(|V|+1)}(f)$*

```
#vals is a hash map from vertices to values supplied by the user
#that represents a function on vertices. upd is the update function
def djikstra(vals, upd):
        #pq is a priority queue that stores pairs of the form (vertex, value)
        pq = empty;
        pair_min = (argmin vals, min vals);
        pq.insert(pair_min);
        #solved_points is a set that contains the points which have
        #been popped from the priority queue
        solved_points = empty;
        while (not(pq.is_empty()))
                min_vertex, min_val = pq.pop();
                vals.update(min_vertex,min_val);
                solved_points.insert(min_vertex);
                for v in neighbors(min_vertex):
                        if not(solved_points.contains(x)):
                                new_val = upd(vals)(v);
                                vals.update(v,new_val);
                                pq.insert(v,new_val);
        return vals;
```

3 Using Djikstra's algorithm to numerically solve Eikonal equations

Definition 9. *Given an HJB triple (L, U, g) we say that it is an Eikonal triple if $L(x, y) = F(y)\|x\|_2$, where $\forall\, y \in \mathbb{R}^n.\, F(y) > 0$ and F is continous on $\mathbb{R}^n$.*

Thus if (L, U, g) is an Eikonal triple we have that the Hamiltonian of this triple is

$$H(\phi, x_0) = \|\nabla \phi\|_2^2 - F(x_0)$$

We will now define a graph and a local monotone update which will help us numerically solve an eikonal equation.

Definition 10 (Discretization). *Given a $K \in \mathbb{N}$ and a box $B = [a_1, b_1] \times [a_n, b_n] \subset \mathbb{R}^n$ we define the K-discretization of B as the graph $G_{K,B} = (V_{K,B}, E_{K,B})$*

$$V_{K,B} = \{(a_1 + (i_1 + 1/2)\delta_i, \ldots, a_n + (i_n + 1/2)\delta_n) \mid 0 \leq i_j \leq K - 1\}$$

$$\begin{aligned} E_{K,B} = &\{\{(a_1 + (i_i + 1/2)\delta_i, \ldots, a_n + (i_n + 1/2)\delta_{i_n}), \\ &(a_1 + (i_1 + 1/2 \pm 1)\delta_1, \ldots, a_n + (i_n + 1/2 \pm 1)\delta_n)\} \\ &|0 \leq i_j \leq K - 1\} \cap (V_{K,B} \times V_{K,B}) \end{aligned}$$

where

$$\delta_i = (b_i - a_i)/K$$

Basically, it is the grid of points obtained by dividing B into K^n boxes, taking the vertices to be the centers of the boxes and taking the edges to be those which connect a center to another adjacent center.

Definition 11 (Eikonal constraint). *Given an Eikonal triple (L, U, g) with $L(x, y) = \|x\|_2 F(y)$ and a discretization $G_{K,B}$ we define the eikonal constraint to be*

$$\Phi_{F,K,B} : (v : V_{K,B}) \to \mathbb{K} \times \mathbb{K}^{N(v)} \to \mathbb{K}$$

$$\Phi(v)(t, n_1, \ldots, n_{d(v)}) = (e_1/\delta_1)^2 + \cdots + (e_n/\delta_n)^2 - F(v)^2$$

where

$$e_i = \max\{\{0\} \cup \{t - v' \mid v' \in (\{v \pm \delta_i(\underbrace{0, \ldots, 1}_{i}, \ldots, 0)\} \cap V_{K,B})\}\}$$

So for example, if $n = 2$, and the Eikonal triple is $(L, U, 0)$ with $L(x, y) = \|x\|_2$ and $v \in V_K$ is not on the "boundary" of the grid then

$$\begin{aligned} \Phi(v)(t, r_{x,+}, r_{x,-}, r_{y,+}, r_{y,-}) = &\delta_x^{-2} \max\{t - r_{x,+}, t - r_{x,-}, 0\}^2 \\ &+ \delta_y^{-2} \max\{t - r_{y,+}, t - r_{y,-}, 0\}^2 - 1 \end{aligned}$$

where $r_{x,\pm}$ correspond to the horizontal neighbors $v \pm (\delta_x, 0)$ and $r_{y,\pm}$ correspond to the vertical neighbors $v \pm (0, \delta_y)$. Proving the following lemma would cause the document to exceed the page limit, but it is not too bad.

Lemma 4 (Eikonal Update). *Given an Eikonal triple (L, U, g) and a discretization $G_{K,B}$ the Eikonal constraint $\Phi_{F,K,B}$ is a monotone local constraint. The function $\mathrm{Upd}_{\Phi_{F,K,B}}$ is called the Eikonal update.*

We now note how discretization can help us approximate the viscosity solutions to Eikonal triples. This corresponds to Theorem 9.17 in [Cal24].

Lemma 5. *Given an Eikonal triple (L, U, g) and a box $B \subset \mathbb{R}^n$, define $g_{B,K} : V_{K,B} \to \mathbb{K}$ by*

$$g_{B,K}(v) := \begin{cases} \min_{x \in B_v} g(x) & \text{if } \partial U \cap B_v \neq \varnothing \\ \infty & \text{else} \end{cases}$$

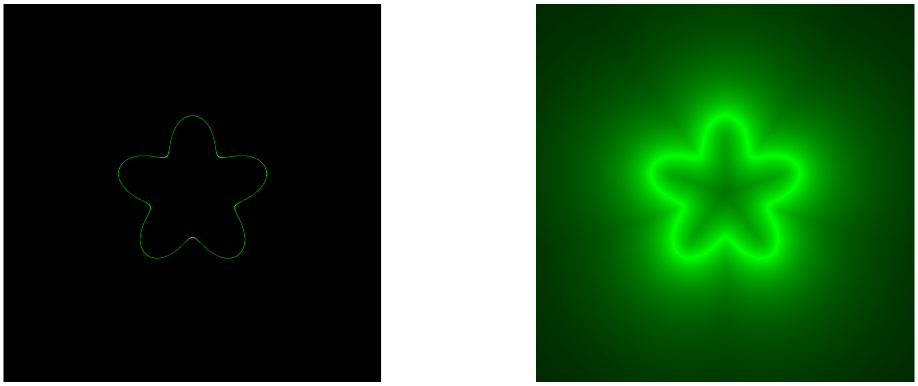
where B_v is the box $[v_1 - \delta_1/2, v_1 + \delta_1/2] \times \cdots \times [v_n - \delta_n/2, v_n + \delta_n/2]$ and v_i is the i^{th} component of v.

Then there exists a constant depending on the Eikonal triple and the box such that

$$|g_{B,K,\infty}(v) - u(v)| \leq C\sqrt{1/K}$$

for all $v \in V_{K,B}$, where u is the viscosity solution to the HJB equation of the triple (L, U, g) and $g_{B,K,\infty} = \mathrm{Upd}_{\Phi_{L,K,B}}^{(|V_{K,B}|+1)}(g_{B,K})$

Thus we have connected Djikstra's algorithm to numerically approximating the viscosity solution of an HJB equation corresponding to an Eikonal triple (L, U, g). This results in an algorithm known as the "Fast Marching Method" introduced by James Sethian; [KS98] provides a good explanation and history of this technique.



(a) (b)

Figure 1: The input and output of generalized Djikstra's algorithm

4 Implementation

We implement Djikstra's algorithm to solve eikonal equations in the programming language Rust. As an exercise, we implement the special case where we try to approximate the distance from a set given by the form $G(x, y) = 0$ where $G : \mathbb{R}^2 \to \mathbb{R}$. We do so by using Djikstra's algorithm on the Eikonal triple $(L, U, 0)$, with $L(x, y) = \|x\|_2$ and $U = G^{-1}(-\infty, 0)$. Figure 1 visually depicts approximating the distance from $G(x, y) = 0$, where

$$G(x, y) = x^2 + y^2 - 0.5\,(2 + \sin(5 \arctan(y/x)))$$

Let $B = [-3, 3] \times [3, 3]$. Each plot in Figure 1 represents a function on $f : V_{1024,B} \to \mathbb{R}_{\geq 0}$; At a vertex v we plot the color green with intensity $1/(1 + 0.01f(v))$. Figure 1a represents $g_{1024,B}$, the input into the generalized Djikstra's algorithm. Figure 1b represents $\mathrm{Upd}^{(1025)}_{\Phi_{L,1024,B}}(g_{B,K})$, the output of the generalized Djikstra's algorithm. Note that Figure 1a approximates the indicator function $\mathbb{1}_{\partial U}$ and that Figure 1b approximates the function

$$f(x, y) = \frac{1}{1 + 0.01 d_{\partial U}(x, y)}$$

References

[Cal24] Jeff Calder. *Lecture notes on viscosity solutions*. 2024. URL: `https://www-users.cse.umn.edu/~jwcalder/8590F18/viscosity_solutions.pdf`.

[KS98] Ron Kimmel and James A Sethian. "Computing geodesic paths on manifolds". In: *PNAS* (1998). DOI: `https://doi.org/10.1073/pnas.95.15.8431`.