Geometrical Construction of H-boxes in 2D Uniform Grid

Judah Ryoo

Apr 13, 2020

1 Introduction

In this report, we describe how to construct 'h-boxes' off an embedded line segment in an uniform 2D grid. An h-box is a non-physical quadrilateral cell of length h (uniform grid size) that covers a small cell and its neighboring cells, and has as its value the proportional average of the cell portions that it covers. Its shape is determined by the geometry of the small cell. The embedded line segment is a special 'geometrical obstacle' that is different from simply a boundary condition. It will represent a overtoppable barrier for our purpose, although infinitely thin. In 1D, h-box averages are simple to calculate as shown in the earlier notes. In 2D, there is an added complexity of the barrier possibly being at an angle to the grid. This introduces variation in the shapes of h-boxes; their widths will vary. Of course, when the barrier is aligned with the grid or parallel to either the x-axis or y-axis, the h-boxes are once again simple to calculate and very similar to the 1D case. In the angular case, we use various algorithms to calculate the h-box values.

2 Geometrical Terminologies and Definitions

Here we clarify our terminology when describing geometrical features in our algorithms.

1. barrier: the embedded line segment
2. physical grid: the 2D uniform domain
3. physical grid edge: the lines $x = x_i$ and $y = y_i$ where x_i, y_i are nodes of the grid
4. non-physical cell: cell that is not created by physical grid edge and barrier
5. physical grid point: the points (x_i, y_j)
6. h-box: a non-physical quadrilateral cell of length h (uniform grid size) that covers a small cell and its neighboring cells, and has as its value the proportional average of the cell portions that it covers
7. vertex: the four corners of h-box

8. side: vertex-to-vertex edge of h-box
9. edge: any line segment that is contained in the h-box or is on the side
10. layer: h-boxes addended next to one another across the barrier
11. small cell: any grid cell that is smaller than regular grid cell because of the barrier
12. fragmenting polygons: polygons whose sides are edges/sides of h-box and union is the h-box

3 Problem Setup

The domain of our problem will be uniformly spaced 2D grid. The embedded line segment, which will henceforth be called the 'barrier', will be placed anywhere within the grid. However, there are two conditions on the barrier's location. First, it cannot be placed too close to the domain's boundary, as the h-boxes will then extend outside of the domain. Second, the barrier cannot begin or end in the middle of a physical grid cell. It must always begin and end on a physical grid edge. This is to ensure that we actually form small cells along the barrier. Otherwise, there will be cells partially enclosed by the barrier, which will require more complex dynamics than our intended purpose. See figure 1. Note that the red barrier creates small cells along its length. In figure 2, the olive colored cells are the small cells that will be covered by the h-boxes.

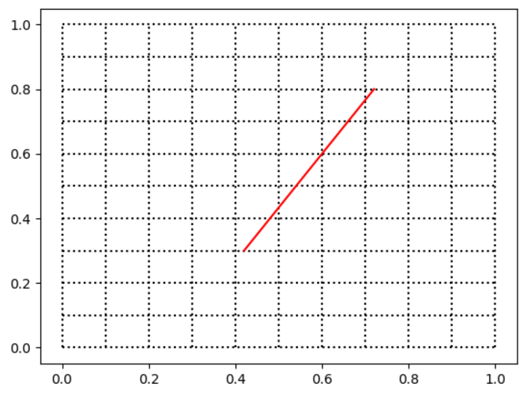


Figure 1: Example of barrier. The domain is $[0, 1] \times [0, 1]$ and $h = 0.1$.

3.1 H-boxes Setup

The first layer of h-boxes will extend normal to the barrier on both sides of the barrier, with half of their vertices being the intersections between the barrier and the physical

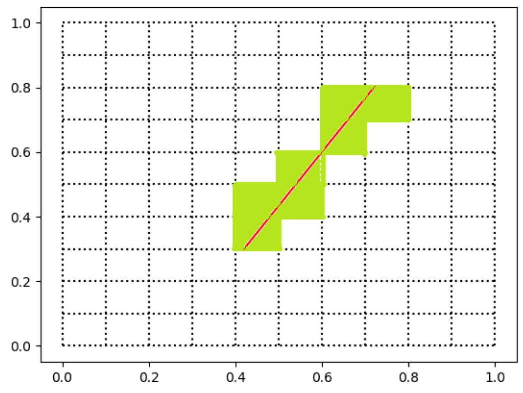
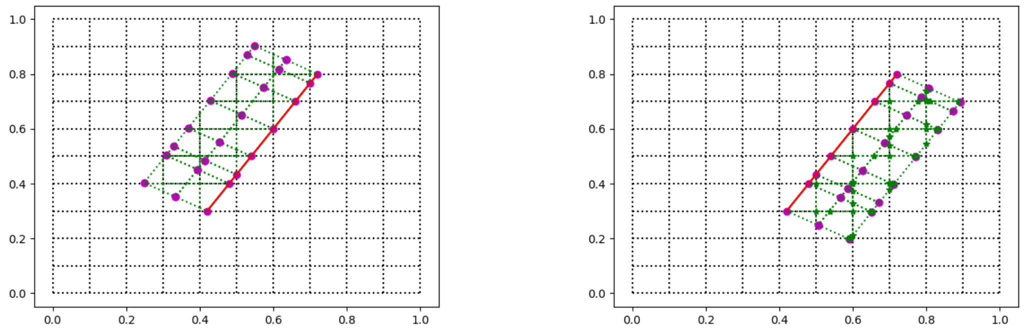


Figure 2: Small cells formed by the barrier

grid. The length of an h-box will be $h = \Delta x$, or the uniform grid size. Note, however, that the width of the h-box will be determined by the distance between two consecutive intersecting vertices on the barrier. The reason why the intersections of the barrier and the physical grid are set as the top two vertices of an h-box is that the normally extended h-box will then cover the small cells. The other two vertices will be the points that extend from the first two vertices normally to the barrier at length of h. This is how the vertices of h-boxes are determined. In addition, we will create two layers of h-boxes, i.e. extend the h-box normally to the barrier with another length of h from the first h-box. This is to cover all the cells whose updates will be affected by the small cells' updates. See figure 3. In the next section we will now describe how we construct these h-boxes.



(a) 'up h-boxes'

(b) 'down h-boxes'

Figure 3: Example of h-boxes

4 Construction Method of the H-boxes

Constructing the h-boxes is a computational geometry problem. There are four main tasks:

1. Find intersections between line segments and identify possible grid points lying inside an h-box
2. Connect appropriate pairs of intersections to form edges and sides
3. Identify multi-sided polygons living in h-boxes formed by the edges/sides and find their areas
4. Identify indices of cells that the h-boxes cover

Once we obtain the areas of the polygons and the indices of the correspondingly covered cells, we can easily obtain the h-box values. We will now describe how each task is performed.

4.1 Find intersections between h-box edges and the grid and grid points lying in h-boxes

The first intersections we must identify are the intersections that the barrier itself makes with the uniform grid. These will serve as part of the vertices of the first layer of h-boxes across the barrier. To find the intersections, we use the fact that the barrier is a straight line and starts and ends at known coordinates. We walk through each physical grid edge that the barrier passes and find the intersections via basic trigonometry.

The more interesting intersections to identify are at the sides of h-box off the barrier. Namely, there are two 'types' of intersections: 1) at normal sides and 2) at parallel sides. The normal sides are the sides of h-box normal to the barrier, and parallel sides the ones parallel to the barrier. To find these intersections, we must find which physical grid edge(s) that the normal or parallel h-box side crosses. The only difficult part of algorising their discovery is knowing which physical edges to check for intersections. What one wants to avoid is go through every single physical edge. To avoid this, we obtain the indices of the vertices of the h-box. Then we can search one (or possibly two) grid edges left or right and up or down to the index, depending on which side of the barrier one needs to construct the h-boxes. We can find these indices as we know the index of the barrier's starting point and can walk up the vertices. Once we isolate the physical edges to check, we can easily find the intersections again by using bas trigonometry. These intersections will serve as endpoints of edges interior to the h-boxes.

We also need to find physical grid points that lie interior to h-boxes. This is because they will also be part of interior edges of h-boxes. In order to find which grid point lies in which h-box, we need a twofold algorithm. First, it needs to check appropriate 'candidate' grid points, and second, it needs to figure out whether a point lies within the h-box. The first task can be accomplished by finding the coordinates of the center of h-box, identifying its grid index, and choosing the four corners of the grid cell as the candidates. The second task can be accomplished by various point-in-polygon algorithms, out of which we use the simple ray-intersecting algorithm. This method checks whether a ray that starts from a candidate point crosses the edges of the h-box an odd or even number of times. If odd, then the point lives in the h-box, and if even, it is outside the h-box. We must be careful not to add grid points that lie directly on an edge, as that will instead be equivalent to an intersection found at the normal or parallel edges and create duplicates. Also, there might be at most two physical grid points that lie interior to the h-box, so we always check if a neighboring grid point also lies inside. Note that no more than two grid points can lie within an h-box since the greatest width possible for an h-box is $\sqrt{2}\Delta x$.

4.2 Connect appropriate pairs of intersections to form edges

Once the intersections and interior-lying grid points are found as above, (as well as the vertices of the h-boxes as mentioned in 'H-boxes Setup'), we link the appropriate points to form edges. There are two types of edges: the sides of an h-box (or edges lying on them) and the interior edges that are formed by the grid edges crossing into the h-box (see figure 3). Note that all the interior edges are horizontal or vertical. The way that these edges are formed is by counting how *many* intersections were found for each h-box above, isolating only these (by indexing), its four vertices, and any interior-lying grid point(s) per iteration of h-box, and linking any two of them that:

- consecutively share in x-coordinate or y-coordinate
- lie consecutively together on a side

The counting of how many intersections and isolating them for consideration is to manage and categorize all the points found above, which can be many, so that the right points are used to form the right edges for each h-box.

To find two consecutive intersections that lie on the same side, we indicate beforehand during the process of finding the intersections on the parallel or normal side whether we have two consecutive intersections. We then connect the two. If there are no intersections at the parallel or normal edge, we connect the two vertices of h-box. Finally, we connect the intersections with the vertices whose side they lie on. To do so, we use simple equation of a line ($y = mx + b$) to determine which side the intersections lie on, since we know the slope of each side.

One precaution we practice in forming edges is carefully determining tolerance values as edges can form between so-called 'intersections' that are actually the same point with machine precision level difference. To avoid forming these false edges, we remove duplicate intersections, avoid adding interior physical grid points that are in fact intersections on edges, and applying a distance criterion between the potential edge-points.

4.3 Identify multi-sided polygons formed by the edges and find their areas

The reason why we must form each side and edge formed by the vertices, intersections, and grid points is that they form polygons whose union gives the h-box. The polygons vary from triangles to sometimes octagons. To find these shapes, we again need to isolate the right edges to consider for each h-box and avoid considering all of them every time we identify the polygons. The method is the same as that used to isolate the right intersections to form edges: count and index. We count edges beforehand when forming them and store this number for each h-box. Then we draw the edges via indexing the list of edges for consideration.

If there are no intersections in any of the h-box sides, there are no inner polygons that are formed, and the only area to calculate is the quadrilateral h-box. The h-box value will be the value of the cell that it covers. In all other cases, there will be fragmenting polygons. Since their shapes and numbers vary from h-box to h-box, we search for them by 'cycling' around the edges and ensure we found all of them by checking that their areas sum up to the total h-box area. Cycling around the edges means that we start from an interior edge (always horizontal or vertical) and locate either clockwisely or counterclockwisely adjacent edge/side until a polygon is formed by these located edges. A polygon is formed when the edges collected form an undirected cycle.

To find whether an edge is located clockwisely or counterclockwisely, we evaluate the cross product of the two edges, with their sign determining the orientation. Here is again a precaution: sometimes small polygons will have edge vectors whose cross product values are very small. We must distinguish these from cross product values that are essentially zero (meaning the two edge vectors are colinear). A careful setting of tolerance value fixes this issue.

We cycle starting from every horizontal and vertical edge in both clockwise and counterclockwise directions. This will find the same polygons more than once for h-boxes that have more than one interior edges, so to distinguish between polygons we also determine the index of the physical grid cell which they lie in. Note that this will always be unique per h-box. We will describe how to find the index of the polygons in the next section. Finally, to find the area of the polygon, we use the standard equation of an area of a polygon A:

$$A = \frac{1}{2}\sum_{i=0}^{N-1}(x_i y_{i+1} - x_{i+1} y_i) \tag{1}$$

$$\tag{2}$$

where (x_i, y_i) are the coordinates of the vertices of the polygons ordered *clockwise*, and $(x_0, y_0) = (x_N, y_N)$ is the starting coordinate.

4.4 Identify indices of cells that the h-boxes cover

The final task of getting h-box averages is to identify what cells that the h-box covers so that their values can be added proportionally to the average. To do this, we find the centroid of the fragmenting polygons, or the center of the h-box if there are not any, and search for the interval of consecutive x axis physical edge values and that of y edge values where the centroid falls. The formula for a centroid of a polygon (c_x, c_y) is given by:

$$c_x = \frac{1}{6A} \sum_{i=0}^{N-1} (x_i + x_{i+1})(x_i y_{i+1} - x_{i+1} y_i) \tag{3}$$

$$c_y = \frac{1}{6A} \sum_{i=0}^{N-1} (y_i + y_{i+1})(x_i y_{i+1} - x_{i+1} y_i) \tag{4}$$

with the same notation as the area formula above. To avoid searching for every interval of physical edges, we start with a 'rough guess', which is the index of the center of h-box. Then we search left or right and up or down until we find the right interval and index.

4.5 Finding the average values

Once all the areas of polygons and their indices have been found, we average the covered cell values by their areas:

$$\tilde{Q}_k = \frac{1}{\sum_{P_{i,j}} A(P_{i,j})} \sum_{P_{i,j}} A(P_{i,j})\, Q_{i,j} \tag{5}$$

where $\tilde{Q}_k$ is the k^{th} h-box average value, $P_{i,j}$ is the fragmenting polygon that covers grid cell $C_{i,j}$ (the whole h-box if there are not any), $A(P_{i,j})$ is its area, and $Q_{i,j}$ is the value of conserved quantity in $C_{i,j}$. Notating H_k to be the h-box, we have $\sum_{P_{i,j}} A(P_{i,j}) = A(H_k)$.