Derivation of Softmax Random Walk Algorithm

Objective:

To compute the probability of moving from a current position (x, y) to a neighboring position (i, j) by blending terrain slope and movement inertia, considering valid data and avoiding immediate return to the last position.

Preliminaries and Definitions:

Notations:

- (x, y): Current Position
- (i, j): Neighboring Position

Parameters:

- ϕ: Weighting factor for slope.
- ψ: Weighting factor for inertia.
- ω: Blending factor between inertia and slope.

Cosine Similarity:

Given two vectors A and B, the cosine similarity, $\text{cosine\_similarity}(A, B)$, is:

$$\text{cosine\_similarity}(A, B) = \frac{A \cdot B}{\|A\|_2 \|B\|_2}$$

Softmax Transformation:

For a vector z with components $z_1, z_2, \ldots z_k$, the softmax function, denoted as $\text{softmax}(z)$, is:

$$\text{softmax}(z)_j = \frac{e^{z_j}}{\sum_{l=1}^{k} e^{z_l}}$$

Elements Calculation:

Slope, $S_{i,j}$:

Given:

- e_c and e_n as elevations at current position and neighbor.
- dx and dy as absolute differences in x and y coordinates.
  The slope $S_{i,j}$ between positions is calculated as follows, considering the handling of invalid data and the distance for diagonal vs. orthogonal moves:

$$S_{i,j} = \begin{cases} \text{np.NINF} & \text{if } e_c \text{ or } e_n \text{ is invalid or } e_c = e_n = \text{no\_data\_value} \\ \frac{e_c - e_n}{\sqrt{2}} & \text{if } dx = dy \text{ (diagonal move)} \\ \frac{e_c - e_n}{1} & \text{if } dx \neq dy \text{ (orthogonal move)} \end{cases}$$

Inertia, $I_{i,j}$:

Using the last movement vector $V_{last} = [dx_{last}, dy_{last}]$ and the direction vector $V_{direction} = [i - x, j - y]$.

$$I_{i,j} = \left(\frac{1 + \text{cosine\_similarity}(V_{last}, V_{direction})}{2} \right)$$

Combining Slope and Inertia:

Raw Combined Weight:

Blend slope and inertia using:

$$W_{i,j}^{raw} = (1 - \omega)\psi I_{i,j} + \omega\phi S_{i,j}$$

Probabilistic Decision Model:

Softmax Transformation:

Compute the transition probabilities by applying the softmax function to the combined weights directly:

$$P((x, y) \rightarrow (i, j)) = \text{softmax}(W^{raw})_{i,j} \rightarrow \frac{e^{W^{raw}_{i,j}}}{\sum_{(k,l)\in\text{neighbors}} e^{W^{raw}_{k,l}}}$$

Here, (k, l) ranges over all positions neighboring (x, y) that are potential candidates for movement, excluding the current position itself and any positions that would constitute an immediate return to the last position.

Computation:

1. For (x, y), calculate $S_{i,j}$ and $I_{i,j}$ for all valid neighbors, excluding the immediate last position to prevent back-and-forth movement.
2. Compute the raw combined weight, $W^{raw}_{i,j}$, for each neighbor by blending inertia and slope.
3. Apply the softmax function to the combined weights to get probabilities for each neighbor.
4. Select a neighbor (i, j) based on these probabilities.
5. Update (x, y) to (i, j) and continue, ensuring to handle masked or invalid elevation data appropriately.