Latent Action Space

Blake Kell
Math 437: Computational Geometry

December 16, 2025

Abstract

High-dimensional datasets, where each data point is defined by dozens of interacting features, are notoriously difficult to visualize and interpret. The intrinsic geometric structure of such data is often obscured by the sheer volume of information. In this project, we use a 21-dimensional behavioral dataset as a cloud of points in a Euclidean space. We apply a pipeline of computational geometry and machine learning techniques to uncover the latent topology of this cloud.

Our approach begins by modeling the data's local geometry using a nearest-neighbor graph to capture the relative similarity between points. To visualize this high-dimensional shape, we embed it into a 2D plane using Laplacian Eigenmaps. This technique utilizes the spectral properties of the Graph Laplacian to perform a projection that preserves local neighborhood information. On this projected map, we utilize density-based clustering to identify "villages" of related points, which represent distinct behavioral archetypes. Finally, we apply Voronoi Diagram to these archetypes, partitioning the continuous latent space into discrete territories. The result is a region-encoded map where the abstract concept of "behavior" is translated into a navigable geometric surface.

1 Introduction

Interpreting or visualizing high-dimensional data is known to be difficult. To get usable information from these types of data set it is required to reduce the dimensionality while trying to maintain the highest possible amount of information. However, equally as important is defining the information you want to extract. I propose that these two things are the "What": What are the main features of the data?, and the "Where": Where is this data point relative to the others?. This question was necessary for this project due its motivation being creating an Action Space for an agent. Since agents traverse a State Space there is an inherent need to understand the relationship between these. However useful the "what" can be, the agent needs to know "where" it will end up. For that reason the usage of the Graph Laplacian over an Adjacency matrix is essential. While the overall component extraction is conceptually quite similar. The underlying difference in the structure it is extracted from, while small, allows for this information to be meaningfully different. The Graph Laplacian combines both the relationships of the nodes as well as their influence. This combination proves superior to a standalone Adjacency matrix since we can now normalize the influence of each node relative to their connectivity. This is important because otherwise our information would be heavily skewed by large and connected nodes and the nuance of smaller nodes would be lost leading to a structurally biased representation.

1.1 Data Description

The dataset used in this project is a proprietary set of personal logs where each entry represents a daily combination of quantitative and qualitative aspects. The quantitative features are measurable items, including activity durations (in minutes), wake and sleep times, and consumable intake (in mg or oz). The qualitative features are subjective scales related to emotional state and intensity. A short-answer summary of each day was also recorded but was not included in the feature analysis. For a detailed breakdown of all 21 features, please see Appendix A.

1.2 Motivation: The Infinite Action Space

The primary motivation for this project stems from the challenge of creating an effective action space for an agent in a complex environment. A traditional approach requires discretizing the world into a finite list of actions. This works well when the developer can guarantee an invariant action space, but it fails in open-world, continuous environments where it can lead to a critical loss of nuance, affecting agent performance. For this scenario, we propose a solution: by discovering the latent space encoded by high-dimensional behavioral data, we can create an action space that scales with experience and preserves the nuance of objectively different outcomes. The central idea is to use the clustering mechanics of the Graph Laplacian. This method finds the "Where" of the data, allowing us to learn its structure, group similar states, and map them to their behavioral results. In essence, this approach replaces explicit, hand-coded action creation with an implicit, learned one. The agent's choice is no longer "what to do," but "where to go" on the map, allowing it to directly understand the consequence of its action.

2 Methodology

The project was executed using a pipeline, implemented in Python using the following libraries: Pandas, NumPy, Scikit-learn, SciPy, and Matplotlib.

2.1 From points to a Graph

The initial dataset consisted of a chronological log of 37 days with 21 features each. To start we need to create an adjacency matrix. For this we used L2 norm (Euclidean distance) and set max neighbors to 10. The max neighbors selection was arbitrary and further research into how this might affect final result can be considered future work. Regardless once we have the adjacency matrix we can continue to the next step.

2.2 Finding the "Where"

The general term is the Graph Laplacian but there exist two main forms the Unnormalized and Normalized Laplacian. The Normalized Laplacian defined as $\mathcal{L} = I - D^{-1/2}\,A\,D^{-1/2}$ is used for this project since the extra scaling by done by $D^{-1/2}$ is what normalizes each node relative to its degree. Allowing for that unbiased structural representation to happen. The Unnormalized $L = D - A$,where D is the degree matrix and A is the adjacency matrix, is more prone to biased represenation since it doesn't normalize the entries of A. Since the central challenge of this project is to find the structure of the latent space while preserving as much information as possible Normalized Laplacian is the natural choice. While the actual calculations were done under-the-hood by the SpectralEmbedding module from SciPy. The mechanism used are as, follows.

1. **Rayleigh Quotient:** The mechanism that allows us to set up to find the eigenpairs is the Rayleigh Quotient defined as

$$RQ_{\text{norm}}(f) = \frac{\langle f, Lf\rangle}{\langle f, Df\rangle} = \frac{f^T Lf}{f^T Df} = \frac{\sum_{(u,v)\in E}(f(u) - f(v))^2}{\sum_{v\in V} d_v\, f(v)^2}$$

Simply put the numerator represents the weighted sum of differences and the denominator represents the degree weighted function. In the first paragraph for this section it is mentioned how the Normalized Laplacian includes an extra scaling factor to normalize the adjacency entries. As you can see there is also a second degree weighted scaling in the denominator. The difference between these two degree weighted normalizations is that the numerators scaling is for the local set of a nodes. While the denominators degree weighted scaling is for the global set of nodes. This double normalization is what allows for the Normalized Laplacian to be so robust for calculating structural clusters.

2. **Generalized Eigenvalue Problem:** While the Rayleigh quotient is a nice way to understand the mechanisms of finding the eigenpairs. Computationally it is not as effective. For actual solving we must rewrite the Rayleigh Quotient for it to leverage existing computational methods. Since the RQ can be considered an optimization problem we can rewrite it with Lagrange Multiplier function. This assigns the numerator and denominator as separate functions. We then find its derivative since setting a derivative to zero finds us the critical points. The equation results in $Ly = \lambda Dy$.

3. **Computational:** Now that we have an equation that is easier to work with we can leverage iterative methods to find the eigenpairs. These can be like the Power or Shift-Inverse methods. Given that we are minimizing the variance over the constraint as shown in the RQ the eigenpairs we are searching for have the smallest eigenvalues. In fact the SpectralEmbedding module specifically uses the ARPACKS Shift-Inverse method to find these eigenvalues since finding all N-1 pairs on a connected component would be intractable. For reference the particular implementation used by SpectralEmbedding has a complexity of $O(n)$ whereas a traditional SVD is $O(mn^2)$.

4. **Eigenpairs:** Finally, once we have found the eigenpairs we can then use the first two non-zero values to find the vectors for embedding into 2D space. Since the first value is zero for connected components. This is given by minimizing over a normalized set of element by setting the vector to all ones, which results in a numerator equal to zero. Given that the RQ is trying to find a function that minimizes the total variance over the degree weighted function. We can interpret these functions as minimizing total variance when applied to the nodes. Since these functions are non-constant we can use them as vectors where each scaling factor acts as a coordinate for each of our points. By using two eigenvectors we effectively have given a x and y coordinate to each point. Which gives us our spectral embedding.

2.3 Clustering (DBSCAN)

Once we have our embedded 2D space, we can apply a density-based clustering algorithm. This is done to not assume the underlying groupings since another potential method K-Means would require that we propose cluster amount or iterate through cluster amounts minimizing a Silhouette Score that works best when clusters are all similarly shaped. Since we didn't know the amount of archetypes or could guarantee similar shapes DBSCAN seemed a more robust solution. The DBSCAN algorithm uses radius and number of neighbors in radius to classify clusters. In particular the heuristics passed allow for two types of points in a cluster to exist; a core point which has at least the minimum number of neighbors inside its radius or a border point where it doesn't have the minimum number of neighbors but it has a core point inside its radius. The combination of core and border points is what allows for DBSCAN to find more complex cluster patterns than K-Means could. The hyperparameters were tuned using a k-distance graph, resulting in an radius of 0.03 and a minimum number of neighbors of 3. This successfully identified 6 distinct clusters, filtering out 2 points as noise.

2.4 Geometric Construct

To finalize our interpretations of the discrete clusters, it was necessary to define a consensus for each cluster. For this, a centroid was calculated for each cluster, which is just the average of all the vectors in that cluster. To be clear, the values averaged were both the embedded coordinates and the original 21-dimensional feature vectors. This was achieved by maintaining an ID for each point across transformations to easily map to the original feature set. This dual mean was necessary since we wanted to interpret the essence of the centroid as well as have a 2D point to use for the Voronoi Diagram. By constructing the Voronoi Diagram on these centroids, we partition the entire latent space into regions. Any point within a specific Voronoi cell is geometrically closest to the behavioral archetype of that cell's centroid. On top of this Voronoi Diagram, we used a convex hull based on the initial embedded points to define a border between experienced and inexperienced behavior. The reason for this Convex Hull is that while Voronoi Regions are infinite, we can not guarantee that any new behavioral point will be clustered at any of the existing regions. As well as serving as a definite bound for our agents' action space. Since it is clear that the regions are based on a centroid calculated from a finite set of points, the entire region is technically not explored. However, given the intuition behind Voronoi Regions, it stands to reason that by creating these regions we have in fact given the agent a larger action selection space. Since for any point in the region, it is closer to some site than any other, from this it can estimate the behavioral outcome of said point. While initially the idea was to leave it as a coordinate space, the lack of definite bounds in continuous space leaves some ambiguity as to the precision allowed in action selection. To remedy this, triangulating the interior of a region could serve as a way to discretize the region, thereby removing the ambiguity. The triangulation could be as simple as triangulating based on the centroid or as complex as using the existing points as interior points to triangulate with. This particular avenue has not yet been explored, but intuitively seems that it could prove fruitful.

3 Results & Analysis

3.1 The Map of the Latent Behavioral Space

The final output of the pipeline is the geometric map shown in Figure 2. The map is composed of six colored territories, each representing a distinct behavioral archetype.

3.2 Behavioral Archetype Profiles

By analyzing the high-dimensional centroids associated with each Voronoi region, we can assign interpretations to the regions:

- **Archetype 0 (Emotionally-Driven Unproductive):** A non-workday coupled high with emotional states and low-productivity activities.
- **Archetype 1 (Passive & Unfocused):** A low-energy state, poor sleep, and a lot of passive media consumption.
- **Archetype 2 (Sleep Irregularity):** Very inconsistent sleep schedule, with work and consistent energy.
- **Archetype 3 (Stabilizing Sleep):** Similar external obligations but marked with higher energy and better sleep patterns.
- **Archetype 4 (High Energy):** A high energy day that switched between high analytical work and mental rest.

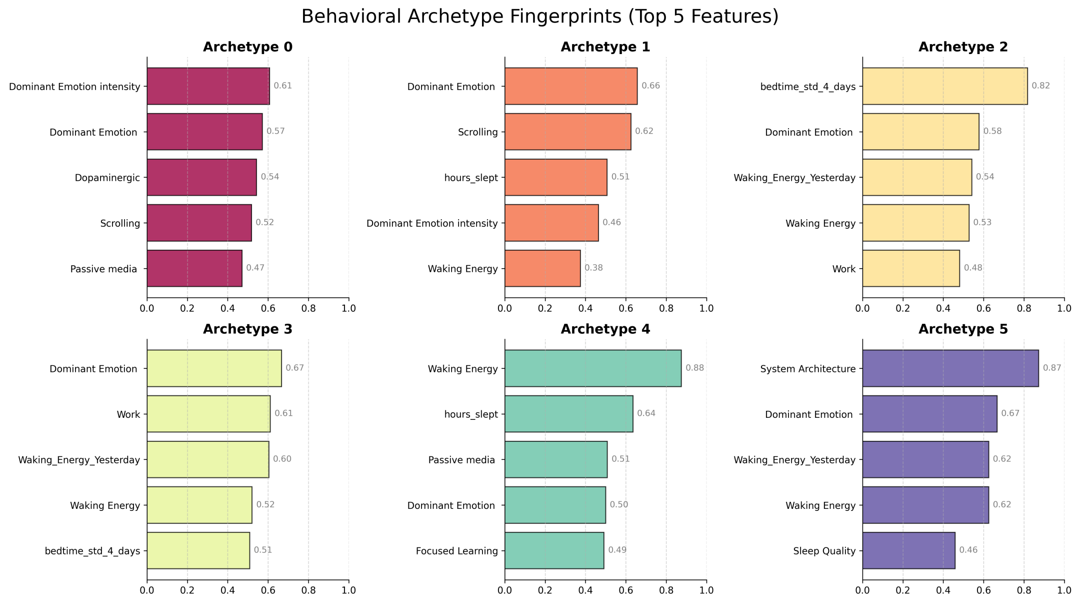

Figure 1: Distributions of each of the 6 archetypes. Top 5 highest features

- **Archetype 5 (Stable Energy):** Generative cognitive work with high energy and sleep quality.

4 Formalizing the agent

The Voronoi Diagram constructed in Section 3 provides a geometric representation of behavioral space, but an autonomous agent requires a learning rule as well. This allows it to determine the best action. We now formalize the structure where the agent's internal state dynamically warps its perception of the behavioral space. This creates a closed-loop system that models the interaction between physiology and habit formation.

4.1 The State Space

We define the state of the agent at time t by the tuple $S_t = (\mathbf{i}_t, \mathbf{b}_t, \mathcal{H}_t)$, where:

- $\mathbf{i}_t \in \mathbb{R}^n$: The **Internal State Vector** (e.g., Fatigue, Motivation, Stress).
- $\mathbf{b}_t \in \mathbb{R}^m$: The **Behavioral Feature Vector**
- $\mathcal{H}_t$: The **Habit Space**, represented by a moving average of set Behavioral vectors.

4.2 The Interaction Matrix (A)

The core mechanism linking physiology to behavior is the Interaction Matrix $A \in \mathbb{R}^{m\times n}$.

- Entries $A_{jk} \in \{1, -1, 0\}$ encode the relationship between internal state i and behavioral feature b.
- This matrix serves as the perceptual warping. At any time step, we compute the Warp Vector $\mathbf{w}_t$:

$$\mathbf{w}_t = A \cdot \mathbf{i}_t$$

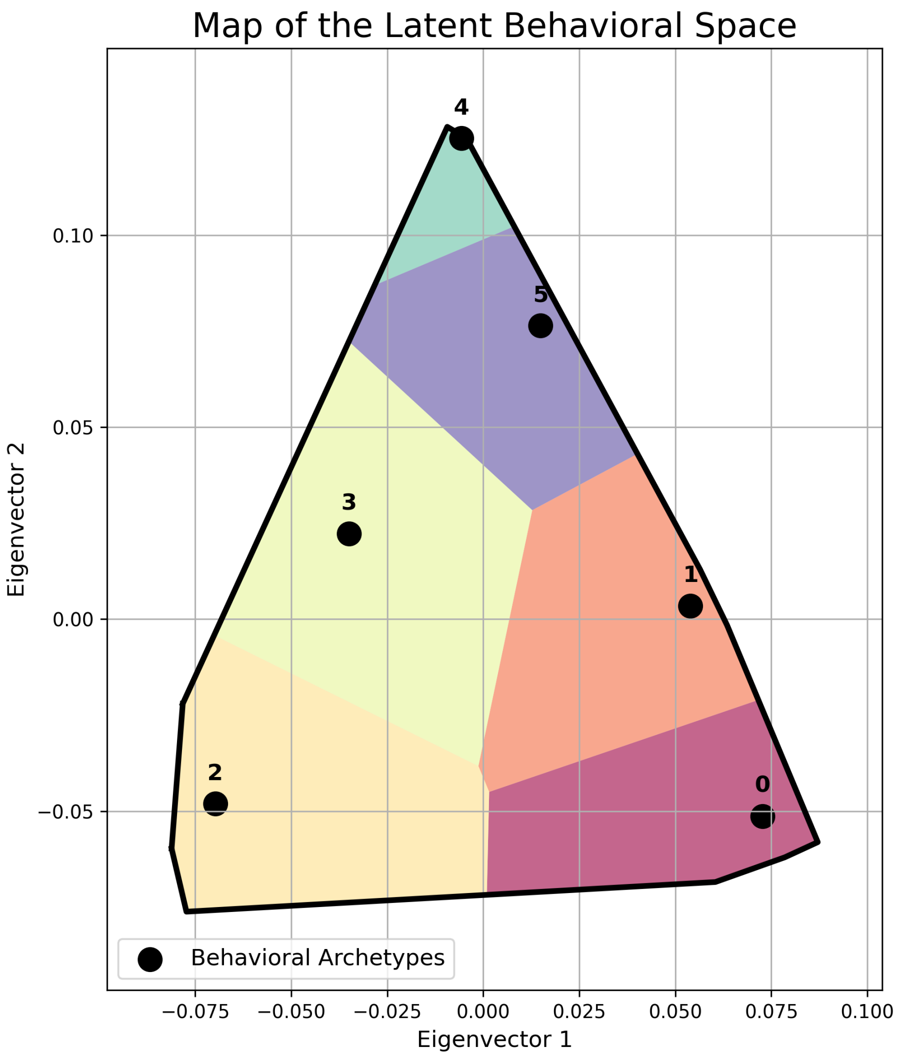

Figure 2: The Map of the Latent Behavioral Space. The 6 Voronoi regions represent distinct behavioral archetypes.

The vector $\mathbf{w}_t$ represents the current influence of the internal state for each behavioral feature. It acts as a weighting function on the behavioral space: if the agent is fatigued, $\mathbf{w}_t$ will negatively weight high-intensity features, basically influencing the perception of the region and manifesting it as a less desirable action.

4.3 Action Selection

The agent must select a target centroid $\boldsymbol{\mu}_k$ from the set of available Voronoi regions. This selection is based on a traditional learning step which includes a target and error. In this particular case since we want to approach an ideal habit space it becomes a trade-off between long-term goals and current needs.

We define the Utility Function $U(\boldsymbol{\mu}_k)$, as long-term goals vs short-term needs. Which has a parameter $\alpha \in [0, 1]$ that can be interpreted as "willpower":

$$U(\boldsymbol{\mu}_k) = \alpha \underbrace{(\mathbf{v}_{ideal} \cdot \boldsymbol{\mu}_k)}_{\text{Value Alignment}} + (1 - \alpha) \underbrace{(\mathbf{w}_t \cdot \boldsymbol{\mu}_k)}_{\text{Affordability}}$$

- $\mathbf{v}_{ideal}$: A static vector representing the ideal internal state

- α: The **Willpower Parameter**.
  - High α implies high willpower (chasing long term goals).
  - Low α implies low willpower (following immediate needs).

The agent selects the action $a^* = \text{argmax}_k\, U(\boldsymbol{\mu}_k)$.

4.4 The Closed Loop

Once an action is selected the system evolves via two mechanisms:

4.4.1 1. Memory

The resulting behavioral vector is inserted into the habit space $\mathcal{H}_{t+1}$. This updates the spectral embedding and shifts the centroids, allowing the agent to grow the territory of the selected behavior over time. Since our Habit Space consist of averaging the sets of behavioral vectors over time we can see how this shifts our current habit space to an ideal habit space.

4.4.2 2. Physiological Feedback

The execution of the behavior exerts a cost on the Internal State. We model this using the transpose of the interaction matrix:

$$\mathbf{i}_{t+1} = \mathbf{i}_t - (1+\alpha)(A^T \cdot \boldsymbol{\mu}_{selected})$$

Just as A maps Internal State to Behavioral Space, A^T maps Behavioral Output back to Internal State. This ensures the system is consistent with real world cases such as high-intensity actions draining the specific internal resources required to perform them. Willpower is added to the internal update to represent how willpower compounds the effect on internal states.

4.5 Example: The Fatigue Cycle

To illustrate how the system functions we will use a simplified scenario:
Internal States: $\mathbf{i}_t = [\text{Fatigue}, \text{Motivation}]^T$
Behavioral Features: $\mathbf{b} = [\text{Focus}, \text{Intensity}]^T$
Interaction Matrix:

$$A = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \quad \text{(Focus/Intensity hurt by Fatigue, helped by Motivation)}$$

Ideal Vector: $\mathbf{v}_{ideal} = [0.9, 0.8]^T$ (High focus, High intensity)
Available Archetypes:

- Archetype 2 (High-Intensity Work): $\boldsymbol{\mu}_2 = [0.9, 0.8]^T$
- Archetype 1 (Passive & Unfocused): $\boldsymbol{\mu}_1 = [0.2, 0.1]^T$

Step 1 - Morning (High Energy)

Internal State: $\mathbf{i}_0 = [0, 1]^T$ (No fatigue, High motivation)
Compute Warp Vector:

$$\mathbf{w}_0 = A \cdot \mathbf{i}_0 = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Set Willpower: $\alpha_0 = 0.8$

Compute Utilities:

$$U(\boldsymbol{\mu}_2) = 0.8(\underbrace{[0.9, 0.8] \cdot [0.9, 0.8]}_{1.45}) + 0.2(\underbrace{[1, 1] \cdot [0.9, 0.8]}_{1.7})$$
$$= 0.8(1.45) + 0.2(1.7) = 1.16 + 0.34 = \mathbf{1.50}$$

$$U(\boldsymbol{\mu}_1) = 0.8(\underbrace{[0.9, 0.8] \cdot [0.2, 0.1]}_{0.26}) + 0.2(\underbrace{[1, 1] \cdot [0.2, 0.1]}_{0.3})$$
$$= 0.8(0.26) + 0.2(0.3) = \mathbf{0.268}$$

Selection: Archetype 2 wins ($1.50 > 0.268$). The agent chooses high-intensity work.

Step 2 - Physiological Depletion

Execute action and compute cost (A^T):

$$A^T \cdot \boldsymbol{\mu}_2 = \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 0.9 \\ 0.8 \end{bmatrix} = \begin{bmatrix} -1.7 \\ 1.7 \end{bmatrix}$$

$$\mathbf{i}_1 = \mathbf{i}_0 - (1 + 0.8)(A^T \cdot \boldsymbol{\mu}_2) \qquad = \begin{bmatrix} 0 \\ 1 \end{bmatrix} - 1.8 \begin{bmatrix} -1.7 \\ 1.7 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} - \begin{bmatrix} -3.06 \\ 3.06 \end{bmatrix} = \begin{bmatrix} 3.06 \\ -2.06 \end{bmatrix}$$

Step 3 - Afternoon (Depleted)

Internal State: $\mathbf{i}_1 = [3.06, -2.06]^T$

Compute New Warp Vector:

$$\mathbf{w}_1 = A \cdot \mathbf{i}_1 = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 3.06 \\ -2.06 \end{bmatrix} = \begin{bmatrix} -5.12 \\ -5.12 \end{bmatrix}$$

Lower Willpower: $\alpha_1 = 0.5$

Compute New Utilities:

$$U(\boldsymbol{\mu}_2) = 0.5(1.45) + 0.5([-5.12, -5.12] \cdot [0.9, 0.8])$$
$$= 0.725 + 0.5(-8.704) = 0.725 - 4.352 = -\mathbf{3.627}$$

$$U(\boldsymbol{\mu}_1) = 0.5(0.26) + 0.5([-5.12, -5.12] \cdot [0.2, 0.1])$$
$$= 0.13 + 0.5(-1.536) = 0.13 - 0.768 = -\mathbf{0.638}$$

Selection: Archetype 1 wins ($-0.638 > -3.627$). The agent is forced toward recovery behavior.

4.6 Limitations & Future Directions

- **Interaction Matrix** The current approach proposes to create one of the key components of the system by directly mapping internal features to behavioral features. For real data sets this might prove intensive and potentially impossible. Methods to compute these interactions could potentially be attention mechanism or some other form of pattern-matching.

- **Learning the Control Policy:** Since this system depends on willpower which should be a learned variable, future work should investigate which RL algorithms can learn this. In particular two methods of learning seem plausible one is a Hierarchical Agent where the action selection and state space observation exist at different time scales. Another approach could be actor-critic methods where the critic components learns values of states and uses its actor counterpart to reflect these values via action selection.

5 Conclusion

This project demonstrates that high-dimensional behavioral data possesses intrinsic geometric structure discoverable through spectral methods. By constructing a Voronoi diagram over behavioral archetypes, we create an **interpretable action space** where the consequence of an action is its position in behavioral space.

What we found most interesting after implementing this idea is that habit formation can be thought of as a optimization problem on a learned topology. Also, rather than hand-engineering discrete actions, an agent can explore its continuous behavioral space. Where its physiological constraints acting as a perceptual warp that influences its decisions.

While the fully functional agent has not been implemented, the geometric foundation has been established and allows for further exploration of the aforementioned limitations. In this project we try to connect geometry, reinforcement learning, linear algebra, and concepts of cognitive biology to approach continuous and high-dimensional space such as behavior from a hopefully insightful perspective.

References

[1] M. Belkin and P. Niyogi, Laplacian Eigenmaps for Dimensionality Reduction and Data Representation, *Neural Computation*, 15 (2003), no. 6, 1373–1396.

[2] S. L. Devadoss and J. O'Rourke, *Discrete and Computational Geometry*, Princeton University Press, Princeton NJ, 2011.

[3] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, *Proc. of the Second Int. Conf. on Knowledge Discovery and Data Mining (KDD-96)*, (1996), 226–231.

[4] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, New York NY, 2009.

[5] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, Distributed Representations of Words and Phrases and their Compositionality, *Advances in Neural Information Processing Systems 26*, (2013), 3111–3119.

[Chu] F. R. K. Chung, *Spectral Graph Theory*, CBMS Regional Conference Series in Mathematics, No. 92, American Mathematical Society, Providence RI, 1997.

A Data Dictionary

The 21 features used to encode each day are detailed below. Unless otherwise noted, time-based features are measured in minutes. Sleep and Wake time were a part of the form I filled out but combined into hours_slept for usability.

Sleep Quality A 1-10 scale measuring how uninterrupted sleep felt (1=Worst, 10=Best).

Waking Energy A 1-10 scale measuring subjective energy upon waking (1=Worst, 10=Best).

Work A 0-5 scale of work intensity (0=No work, 5=Extremely demanding).

Focused Learning Time spent on bottom-up cognitive skills (e.g., math, coding).

Skill practicing Time spent practicing a hobby (e.g., guitar, drawing).

Physical Endeavors Time spent on physical activity (e.g., gym, soccer).

Scrolling Time spent on social media or news feeds.

Dopaminergic Time spent on high-stimulus, low-reward activities (e.g., binge eating, aimless browsing).

Passive media Time spent consuming non-interactive content (e.g., watching shows).

Active Media Time spent on interactive entertainment (e.g., video games, chess).

Dominant Emotion A -5 to 5 scale representing mood polarity (-5=Depressive, 5=Manic).

Dominant Emotion intensity A 1-10 scale of how strongly the dominant emotion was felt.

System Architecture Time spent on top-down cognitive tasks (e.g., designing systems, planning).

NIC Nicotine intake, in milligrams (mg).

CAF Caffeine intake, in milligrams (mg).

ALC Alcohol intake, in fluid ounces (oz).

L-TY L-Tyrosine intake, in milligrams (mg).

WE Cannabis intake, in milligrams (mg).

– ***Engineered Features*** –

hours_slept Calculated sleep duration (wake time - sleep time).

Waking_Energy_Yesterday Lagging indicator; the ‘Waking Energy‘ value from the previous day.

bedtime_std_4_days The standard deviation of bedtime over the past 4 days, measuring sleep schedule consistency.