Linderman Lab Presentation

Michael Murphy

Summer Internship

- Working with Libby and Dietrich on the fundamentals of **machine learning** and **computational neuroscience**
- Learning about the **math and statistics** concepts that underlie these fundamentals
- Applying what I have been learning to **Killifish data** using **Python** and tools such as **Numpy** and **Matplot.lib**

Killifish Keypoint Data

- Convolutional Neural Network traces the different keypoints of a Killifish from a video recording
- Recording is 200 days long, recording at 20 frames per second, tracking 12 different keypoints meaning there is 4.14 billion data points!
- I worked with one day of it, meaning 1.7 million frames of data

	frame_count	frame_timestamp	x_snout	y_snout	x_midbody	y_midbody	x_sidebody	y_sidebody	x_endbody	y_endbody	x_tail	y_tail	x_fan	y_fan
0	0	1.619461e+09	109.540108	142.976059	124.138733	135.067261	NaN	NaN	151.504974	123.650360	165.433395	113.957596	180.608887	105.863663
1	1	1.619461e+09	107.265854	142.104813	121.822556	136.475708	NaN	NaN	147.926788	123.891907	160.844559	115.370964	174.711700	106.121666
2	2	1.619461e+09	105.351967	142.368988	119.697525	136.510712	NaN	NaN	145.834869	123.651062	160.248383	116.183006	174.869812	107.406532
3	3	1.619461e+09	103.469025	142.543152	117.734238	136.160843	NaN	NaN	143.201263	124.122238	158.663406	117.667580	176.247086	109.998772
4	4	1.619461e+09	102.180344	142.780396	115.806297	136.204010	NaN	NaN	142.057327	123.676147	157.429688	117.932121	176.791046	110.939461
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1723651	1726808	1.619548e+09	84.945068	53.992599	96.640488	65.568367	NaN	NaN	108.316635	89.087357	109.255569	107.692741	109.788170	128.898438
1723652	1726809	1.619548e+09	83.437828	49.815742	97.773056	63.414818	NaN	NaN	112.321381	86.499474	113.034012	104.349236	106.420898	123.391586
1723653	1726810	1.619548e+09	82.824081	46.829994	97.266914	60.762054	NaN	NaN	114.773003	81.535423	120.126129	98.978592	110.384720	118.568207
1723654	1726811	1.619548e+09	81.770180	42.896610	94.724854	55.871910	NaN	NaN	115.567490	77.666855	122.965393	94.156998	117.896706	115.093361
1723655	1726812	1.619548e+09	81.606567	42.340202	93.756645	53.572468	NaN	NaN	114.121025	74.155083	124.118027	88.861328	125.768005	111.091743

1723656 rows × 14 columns

Goals

- Learn more about computational neuroscience, machine learning, math, and statistics

- Get hands on experience working with and visualizing big data with Python

- Participate in real science with a professional lab!

Math and Statistics

Differential Calculus

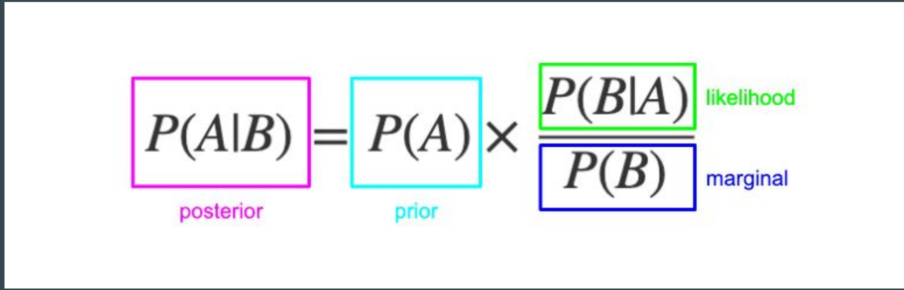
- Derivative rules
  - Power rule
  - Chain rule
- When derivative = 0, either local min or max
  - Determine which by computing the second derivative

Linear Algebra

- Vectors
- Matrices
- Dot Product
- Linear Combination
- Eigenvectors and Values

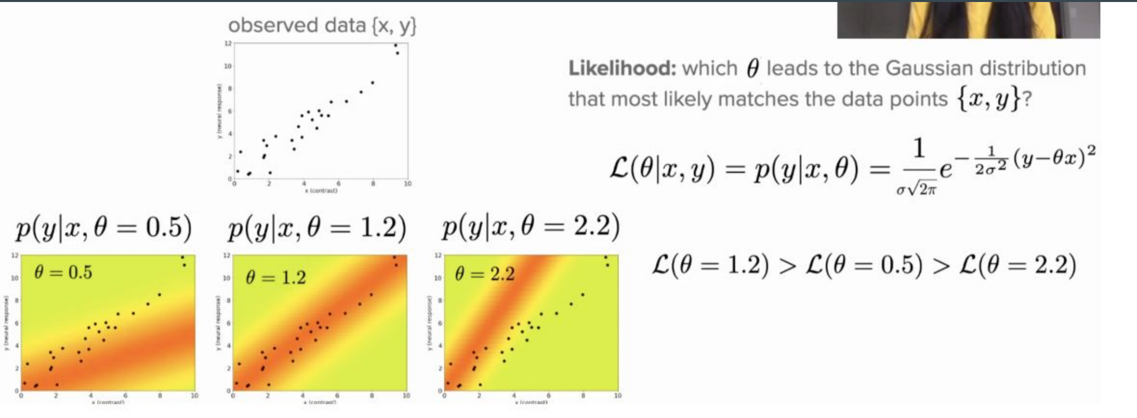
Statistics

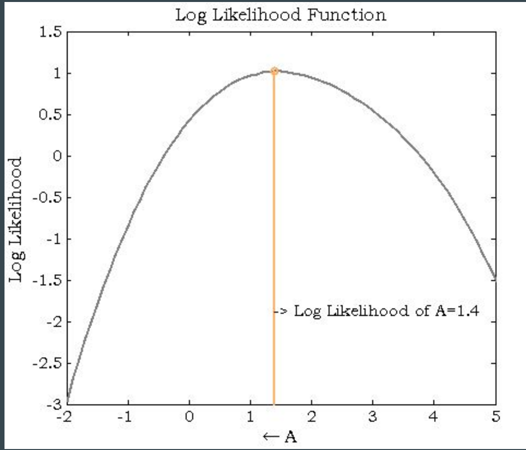
- Normal / Gaussian Distributions
- Bayes Rule
- Bayesian vs. Frequentist Approaches



Linear Regression

Maximum Likelihood Estimation



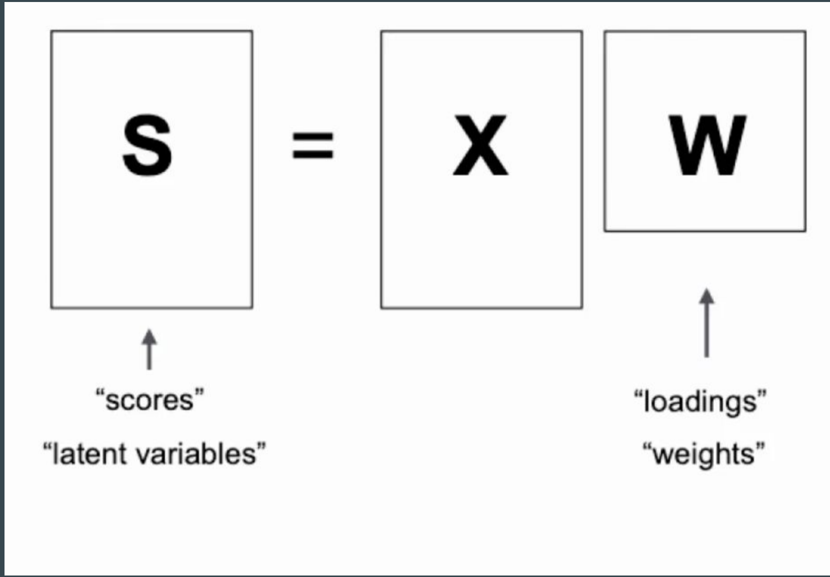


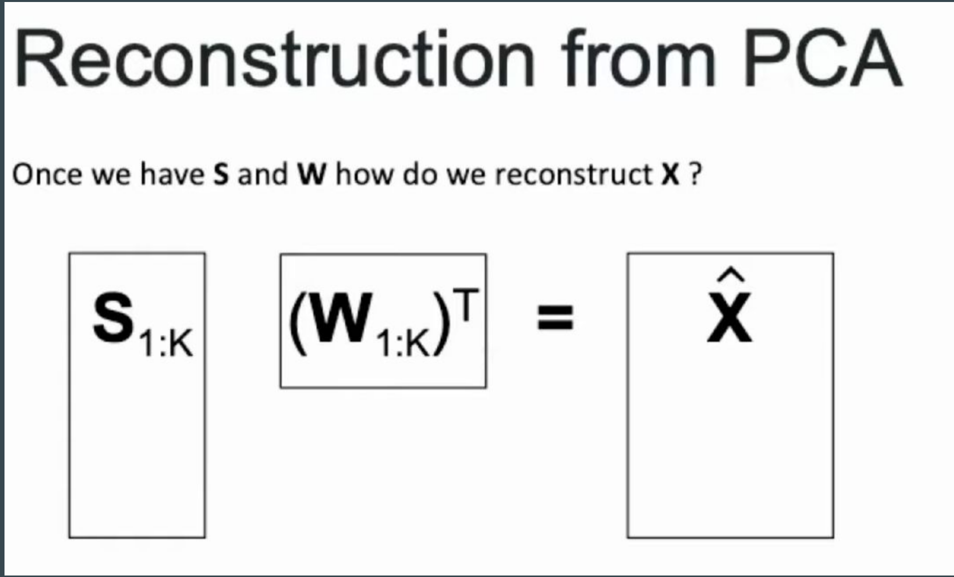
- If the noise of your data is Gaussian, you can estimate the likelihood of a given linear weight theta by plugging it in to a modified Gaussian density function

- By setting the derivative to 0, you can find the maximum likelihood value, for which the corresponding value of theta is the optimal linear weight

Principal Components Analysis

- Method of reducing dimensionality of your data
- Ties in linear algebra



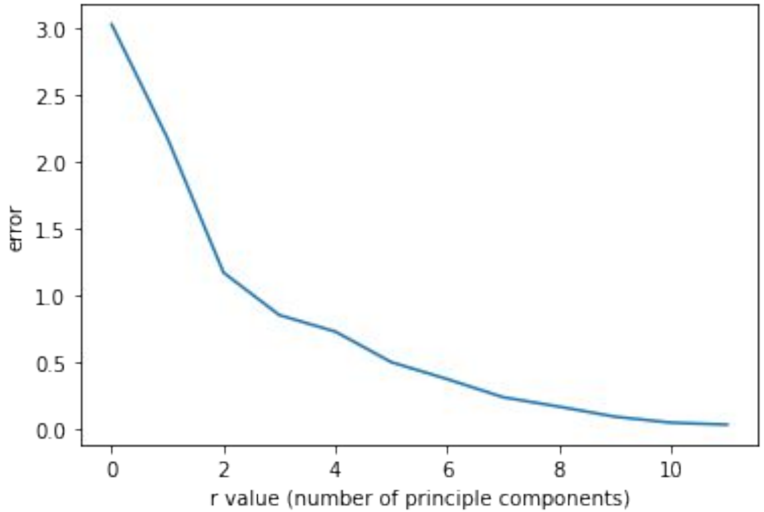


PCA on Killifish Data

	frame_count	frame_timestamp	x_snout	y_snout	x_midbody	y_midbody	x_sidebody	y_sidebody	x_endbody	y_endbody	x_tail	y_tail	x_fan	y_fan
0	0	1.619461e+09	109.540108	142.976059	124.138733	135.067261	NaN	NaN	151.504974	123.650360	165.433395	113.957596	180.608887	105.863663
1	1	1.619461e+09	107.265854	142.104813	121.822556	136.475708	NaN	NaN	147.926788	123.891907	160.844559	115.370964	174.711700	106.121666
2	2	1.619461e+09	105.351967	142.368988	119.697525	136.510712	NaN	NaN	145.834869	123.651062	160.248383	116.183006	174.869812	107.406532
3	3	1.619461e+09	103.469025	142.543152	117.734238	136.160843	NaN	NaN	143.201263	124.122238	158.663406	117.667580	176.247086	109.998772
4	4	1.619461e+09	102.180344	142.780396	115.806297	136.204010	NaN	NaN	142.057327	123.676147	157.429688	117.932121	176.791046	110.939461
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1723651	1726808	1.619548e+09	84.945068	53.992599	96.640488	65.568367	NaN	NaN	108.316635	89.087357	109.255569	107.692741	109.788170	128.898438
1723652	1726809	1.619548e+09	83.437828	49.815742	97.773056	63.414818	NaN	NaN	112.321381	86.499474	113.034012	104.349236	106.420898	123.391586
1723653	1726810	1.619548e+09	82.824081	46.829994	97.266914	60.762054	NaN	NaN	114.773003	81.535423	120.126129	98.978592	110.384720	118.568207
1723654	1726811	1.619548e+09	81.770180	42.896610	94.724854	55.871910	NaN	NaN	115.567490	77.666855	122.965393	94.156998	117.896706	115.093361
1723655	1726812	1.619548e+09	81.606567	42.340202	93.756645	53.572468	NaN	NaN	114.121025	74.155083	124.118027	88.861328	125.768005	111.091743

1723656 rows × 14 columns

- The 12 dimensional killyfish keypoint data can be modeled by just its first 4 principle components, with relatively low reconstruction error!



Numpy

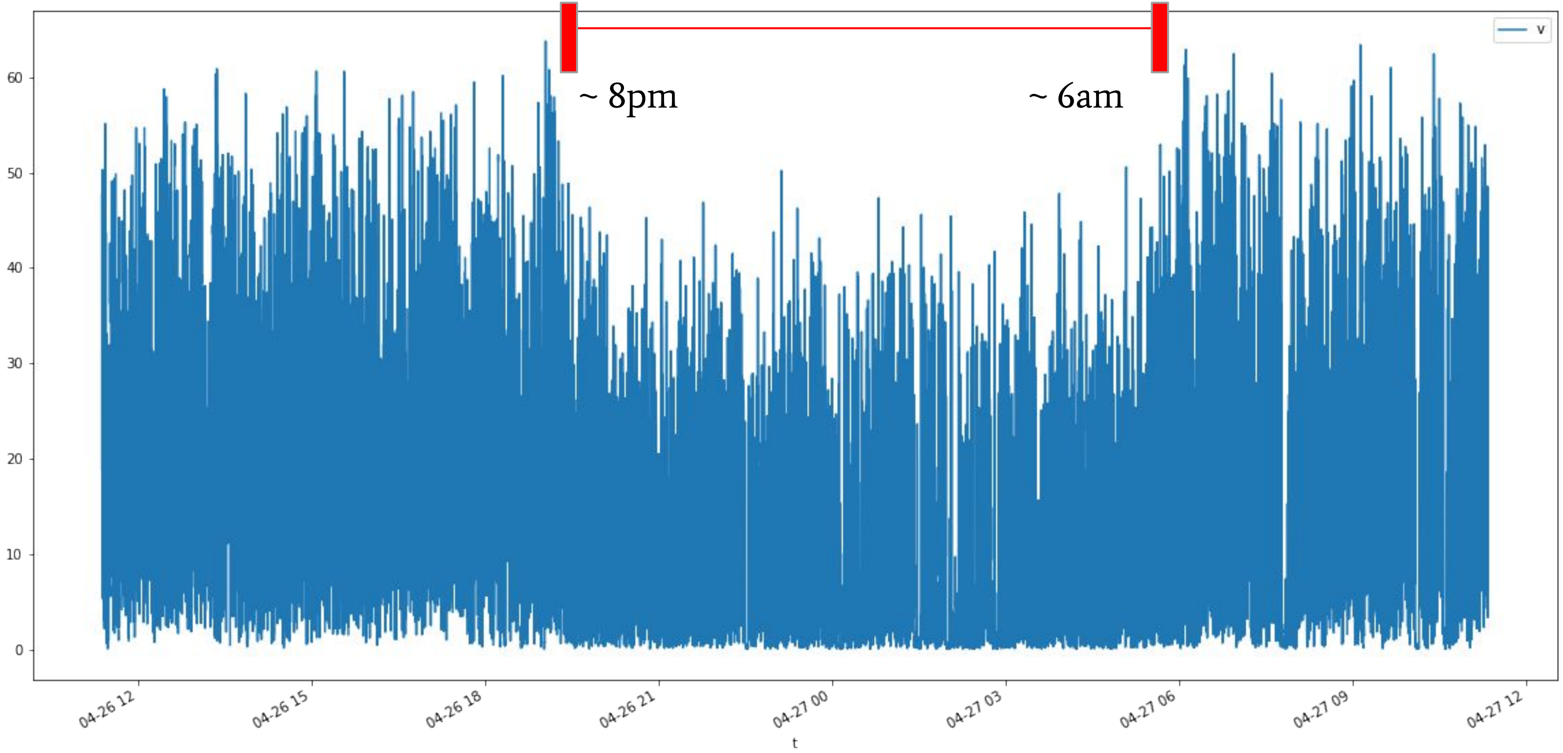
```
eigvals_st, eigvecs_st = np.linalg.eig(cov_st)

print(eigvals_st, eigvecs_st)
```

```
[ 6.40314495e+00  4.56719643e+00  9.74716365e-01 -5.17432518e-01
  3.60518594e-01  2.02530768e-01 -4.36588799e-02  3.01574051e-02
  2.22033390e-02 -2.84729261e-03  1.53732086e-03  1.96403237e-03]
   5.57366495e-01  1.89187581e-01 -1.77612131e-01  5.25388061e-01
   3.46165751e-02  2.25812165e-02  4.33823218e-01  8.38312222e-02]
 [-1.51168375e-01  3.67139538e-01  4.44731084e-01 -3.69947772e-02
  -2.08346943e-01  3.66937314e-01  1.16057321e-01 -5.47109717e-02
  -4.30246999e-01  4.68785105e-01  1.58256545e-01  1.37008062e-01]
 [ 3.72611165e-01  1.26017492e-01  3.19894876e-02  2.11159578e-01
   3.37664328e-01  3.12172179e-01  2.41057040e-01 -1.69777968e-01
  -1.58104256e-01 -3.32474173e-03 -5.43452046e-01 -4.25345691e-01]
 [-1.66753880e-01  4.04291509e-01  3.17038478e-01 -1.07817222e-01
```

- One numpy function I found interesting: “np.linalg.eig”
- Input any matrix, and it will return the ordered eigenvectors and eigenvalues
- Gets rid of the need to do eigendecomposition

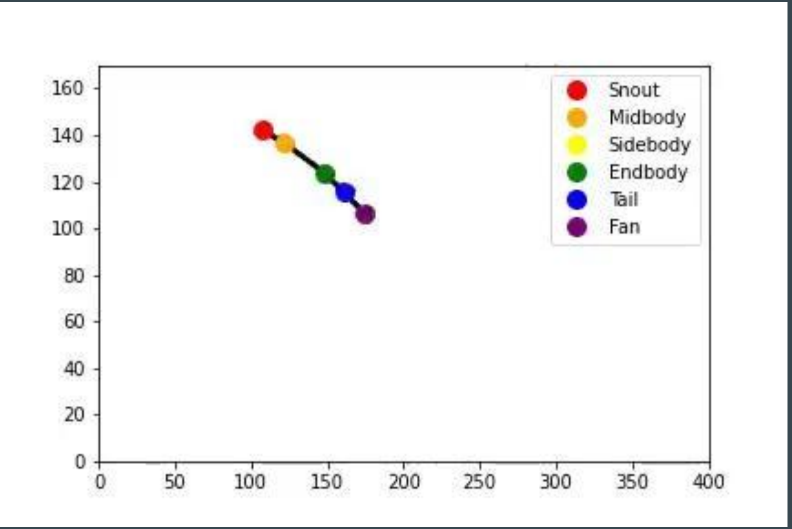
Observations of Data

- Velocity of the fish, plotted for every 100th frame of an entire day
- The killyfish moves slower / less during the hours of 8pm - 6am

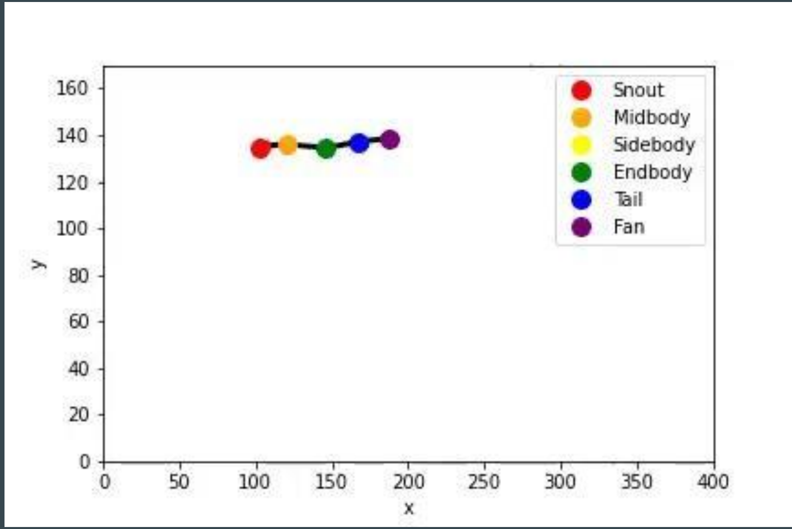


Animations

Fish at 11am



Fish at 11pm



- Random samples of 100 frames from these times
- Follows trend of fish moving slower / less at night
- Logical trend, because fish is sleeping!

Future Plans

- Compare different features over the Killyfish entire life span
- Continue to learn about neuroscience, machine learning, math, and statistics
- To continue my learning, I am attending the UC Cosmos summer program at UC Irvine, where I will be researching the applications of Data Science to the Health Sciences, and learning the programming language R.

Thank You!

- Thanks to the Linderman Lab for taking me in this summer!
- Thanks to Libby and Dietrich for spending so much time teaching me!
- Thanks to Scott for giving me this unique and amazing opportunity and introduction to real science!