Vertex Reconstruction

9/15

Taking a look at Polynomial VR

We currently use the full hit coordinate set (x,y,dx,dy) for each reconstructed vertex value (p, ip, oop), however we have strongly inferred that different reconstructed values may depend much more on one set of coordinates than another, so we tested it.

Ex. P depending more on x and dx

We used sklearn to train coefficients using a train/test split on a random run set of data, and perform a similar task as the original polynomial fit.

Recent work has been on injecting noise and testing how well different models hold up.

Fitting Coefficients

Example: [x,y]
degree 2

$$X_{\text{poly}} = \begin{bmatrix} x_1 & y_1 & x_1^2 & x_1 y_1 & y_1^2 \\ x_2 & y_2 & x_2^2 & x_2 y_2 & y_2^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & x_n^2 & x_n y_n & y_n^2 \end{bmatrix}$$

$$\min_a \sum_i (y_i - X_{\text{poly},i} \cdot a)^2$$

Spits out a bunch of coefficients, which are then applied to the testing set

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$$R^2 = 1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum (y_i - \bar{y})^2}$$

Lasso

$$\min_{a} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - X_{\text{poly},i}\, a\right)^2 \; + \; \alpha \sum_{j} |a_j|$$

It gets more complex once we have to start eliminating terms, using Lasso. We incorporate a cost function associated with each coefficient in the matrix which looks like this.

The minimizer now has to balance coefficient size as well as goodness of fit. Multiple different alphas are tried and minimized.
Can emphasize coefficients which are more important than others.

α = 0 -> Plain LSF
Small α -> smaller coefficients are driven to 0
Large α-> only largest coefficients survive

Results, no noise

{x, y, dx, dy}

=== Momentum (p) (train clean, test clean) ===
Degree 1: MSE=0.00175, R^2=0.9988
Degree 2: MSE=0.00154, R^2=0.9989
Degree 3: MSE=0.00148, R^2=0.9990
Degree 4: MSE=0.00168, R^2=0.9988
Degree 5: MSE=0.00342, R^2=0.9976

=== In-plane angle (ip) (train clean, test clean) ===
Degree 1: MSE=0.01884, R^2=0.9696
Degree 2: MSE=0.00763, R^2=0.9877
Degree 3: MSE=0.00709, R^2=0.9885
Degree 4: MSE=0.00605, R^2=0.9902
Degree 5: MSE=0.01183, R^2=0.9809

=== Out-of-plane angle (oop) (train clean, test clean) ===
Degree 1: MSE=2.72707, R^2=0.6636
Degree 2: MSE=2.55398, R^2=0.6849
Degree 3: MSE=2.40122, R^2=0.7038
Degree 4: MSE=2.31744, R^2=0.7141
Degree 5: MSE=2.37942, R^2=0.7065

{x, y ,dx, dy} with alpha pruning

=== p (train clean, test clean) ===
deg 1: R^2=0.9988, MSE=0.00175, alpha=0.0001456, 2/4 terms
deg 2: R^2=0.9989, MSE=0.00155, alpha=0.0009541, 6/14 terms
deg 3: R^2=0.9990, MSE=0.00148, alpha=0.0001758, 20/34 terms
deg 4: R^2=0.9990, MSE=0.00147, alpha=0.0002121, 33/69 terms
deg 5: R^2=0.9990, MSE=0.00143, alpha=0.00016, 50/125 terms

=== ip (train clean, test clean) ===
deg 1: R^2=0.9696, MSE=0.01884, alpha=0.0001, 4/4 terms
deg 2: R^2=0.9877, MSE=0.00763, alpha=0.000256, 9/14 terms
deg 3: R^2=0.9888, MSE=0.00692, alpha=0.0003089, 18/34 terms
deg 4: R^2=0.9899, MSE=0.00625, alpha=0.0002121, 38/69 terms
deg 5: R^2=0.9910, MSE=0.00559, alpha=0.0001099, 62/125 terms

=== oop (train clean, test clean) ===
deg 1: R^2=0.6635, MSE=2.72784, alpha=0.005429, 3/4 terms
deg 2: R^2=0.6826, MSE=2.57267, alpha=0.01389, 7/14 terms
deg 3: R^2=0.6781, MSE=2.60914, alpha=0.04292, 10/34 terms
deg 4: R^2=0.6885, MSE=2.52547, alpha=0.03556, 15/69 terms
deg 5: R^2=0.7094, MSE=2.35611, alpha=0.009541, 33/125 terms

Results, with noise

{x, y, dx, dy}

=== Momentum (p) (train clean, test noisy) ===
Degree 1: MSE=0.00210, R^2=0.9985
Degree 2: MSE=0.00190, R^2=0.9987
Degree 3: MSE=0.00185, R^2=0.9987
Degree 4: MSE=0.00204, R^2=0.9986
Degree 5: MSE=0.00377, R^2=0.9974

=== In-plane angle (ip) (train clean, test noisy) ===
Degree 1: MSE=0.02044, R^2=0.9670
Degree 2: MSE=0.00946, R^2=0.9847
Degree 3: MSE=0.00887, R^2=0.9857
Degree 4: MSE=0.00769, R^2=0.9876
Degree 5: MSE=0.01367, R^2=0.9779

=== Out-of-plane angle (oop) (train clean, test noisy) ===
Degree 1: MSE=2.72780, R^2=0.6635
Degree 2: MSE=2.55474, R^2=0.6849
Degree 3: MSE=2.40352, R^2=0.7035
Degree 4: MSE=2.32872, R^2=0.7127
Degree 5: MSE=2.37261, R^2=0.7073

{x, y ,dx, dy} with alpha pruning

=== p (train clean, test noisy) ===
deg 1: R^2=0.9985, MSE=0.00211, alpha=0.0001456, 2/4 terms
deg 2: R^2=0.9987, MSE=0.00191, alpha=0.0009541, 6/14 terms
deg 3: R^2=0.9987, MSE=0.00185, alpha=0.0001758, 20/34 terms
deg 4: R^2=0.9987, MSE=0.00182, alpha=0.0002121, 33/69 terms
deg 5: R^2=0.9988, MSE=0.00180, alpha=0.00016, 50/125 terms

=== ip (train clean, test noisy) ===
deg 1: R^2=0.9670, MSE=0.02044, alpha=0.0001, 4/4 terms
deg 2: R^2=0.9847, MSE=0.00946, alpha=0.000256, 9/14 terms
deg 3: R^2=0.9860, MSE=0.00867, alpha=0.0003089, 18/34 terms
deg 4: R^2=0.9873, MSE=0.00789, alpha=0.0002121, 38/69 terms
deg 5: R^2=0.9883, MSE=0.00727, alpha=0.0001099, 62/125 terms

=== oop (train clean, test noisy) ===
deg 1: R^2=0.6634, MSE=2.72856, alpha=0.005429, 3/4 terms
deg 2: R^2=0.6826, MSE=2.57322, alpha=0.01389, 7/14 terms
deg 3: R^2=0.6780, MSE=2.60993, alpha=0.04292, 10/34 terms
deg 4: R^2=0.6881, MSE=2.52864, alpha=0.03556, 15/69 terms
deg 5: R^2=0.7088, MSE=2.36096, alpha=0.009541, 33/125 terms

Next steps

Incorporate a ridge regression model for oop, should reduce terms to small coefficients but not outright get rid of them, like lasso does. Better when all terms matter a little instead of a few terms mattering a lot.

Apply to a signal run, compare output.

Look at mass reconstruction using new tailored coefficients- should be a finalized version of the polynomial vertex reconstruction and ready for implementation.