Implicit manifold learning progress report:

Problem Definition:
given datapoints that supposedly lie on several different clusters we'll try to formulate each cluster as a combination of smoothly varying sub clusters where each of these sub-clusters has a polynomial formulation. The end goal is to derive moments of each sub-clusters distribution which can then be used in approximating the geodesic distance of the points A and B lying on the same sub-cluster the same way as Dollar paper with the difference that instead of a projection matrice, we have moments of the distribution that will guide as as to which small step that is being taken from A to B is more likely to keep us on the sub-cluster. (generalization to entire cluster using smoothness of variation between polynomials has to be decided?).

POP Formulation:(a separate program is defined for each cluster)

program parameters:

k: number of sub-clusters
mon_num: number of monomials of the defining polynomials.
data_num : number of data points

program variables & constraints:

explicit variables:

C_j (j=1:k)=> coefficient of the polynomial, fitting the j-th sub-cluster. (has mon_num vars).
B_i,j (i=1:data_num, j=1:k) => indicator variable for i-th datapoint for belonging to the j-th sub-cluster.
eps => error slack of fitting datapoints to sub-clusters (taken to be the same for all sub-clusters for now)

explicit constraints:

-eps<C_j.B_i,j.x_i<eps => constraint for fitting i-th datapoint (x_i) to j-th sub-cluster if B_i,j is set to 1.

B_i,j^2-B_i,j=0 => boolean constraint for indicator variables

N**ote:** constraint for forcing sum of B_i,j over all j-s (all assignments of datapoint i to sub-clusters) to 1 is not added so it's possible for a datapoint to not belong to any sub-cluster and as a result belong to a different cluster.

implicit variables:

X_j => the implicit variable for first degree moments of the data points lying on the j-th cluster.

l_j => the bound on distance of X_j from the j-th sub-cluster (in form of evaluation of the polynomial coefficients, not the actual euclidian distance).

vecEps => a vector of epsilons used in defining

implicit constraints:

$-eps < C_j.X_j < eps$ => constraint that the implicit X_j should be close to sub_cluster j.

$-l_j < C_j.X_j < l_j$ => the bound variable (difference with epsilon being that that’s a shared variable with all the implicit variables and the datapoints and appears in the objective term, whereas this one is defined for each implicit variable)

$(\mathrm{Sum}(B_{i,j}.x_i)$ on $i = (\mathrm{Sum}(B_{i,j})$on $i).X_j$ => this constraint links the implicit variableX_j to the empirical average of datapoints x_i lying on the j-th sub cluster.

New Smoothness constraint between clusters should be decided: