Comparison on Optimal Transformation

Qingyang Zhao, Wonho Bae

***Abstract*—Probability and statistic methods help us view large scale of data samples in an interpretable and disciplinary sense. After fitting all feature categories into random variables, it also becomes convenient to analyze the inner relationship between two objects by using the conception of dependency. One of the classical algorithms for finding correlated variables is Canonical Correlation Analysis (CCA). It extracts the linear dependency between two objects by linearly combining all the feature columns. However, the linear assumption largely constrains the performance of this algorithm since the real-world data tends to be generated from complex nonlinear functions. One of nonlinear extensions of the linear CCA is Alternating Conditional Expectation (ACE). The power of this algorithm is to reveal maximal correlation between two objects by nonlinearly transforming the feature representations of each object. This article first introduces CCA and ACE in terms of the optimal transformation algorithms. In addition, we give the proof of equivalence between OLS and CCA, MSE and ACE. In the experiment part, we compare CCA, ACE and Neural Network in several aspects by using polynomial, continuous and discontinuous functions.**

***Index Terms*—CCA, ACE, Neural Network**

I. Introduction

WHEN doing regression, researchers tend to find the intrinsic property embedded in the sample points. In a probability sense, this property is measured by dependency. Machine Learning scientists have done research on this field for decades. Plenty of fancy algorithms are developed to discover such characteristics. One of the classical approaches is Canonical Correlation Analysis (CCA). It measures the linear dependency between two random variables X and Y by maximizing the Pearson Correlation Coefficient. By projecting each of X and Y onto the principal directions found by CCA, we can efficiently extract linearly correlated features for each of X and Y space.

However, the power of CCA is limited because it is under the assumption that the features of both X and Y are linearly correlated. If X and Y are generated from some nonlinear functions, $f(X)$ and $g(Y)$, CCA might not capture the relationship between X and Y. For instance, if X and Y are the coordinates of an ellipse in two-dimensional space, the Pearson Correlation Coefficient is zero (Fig.1 [Yu, 2017]). It apparently shows the limitation of CCA.

In 1980s, a nonlinear optimal transformation method, Hirschfeld-Gebelein-Renyi Maximal Correlation [Breiman, 1985] has been discovered. This correlation coefficient generalizes CCA from linear case to nonlinear function. The advantage of this correlation coefficient is that if there exists function (regardless of linearity), the coefficient is 1.

Q. Z., is with University of California, Berkeley, CA 94720, USA. He is now with the Electrical Engineering and Computer Science Department, (e-mail: qingyang_zhao@berkeley.edu).

W.B. is with University of California, Berkeley, CA 94704, USA. He is an undergraduate student in Statistics Department at UC Berkeley, (e-mail: jhun324@berkeley.edu).

II. Background

A. Pearson Correlation Coefficient

The linear dependency of two Random Variables X and Y can be measured by Pearson Correlation Coefficient, denoted as $\rho(X,Y)$,

$$\rho(X,Y) = \frac{cov(X,Y)}{\sqrt{var(X)var(Y)}}$$

where $cov(X,Y)$ is the covariance matrix between X and Y, and $var(X)$ is the variance of random variable X. Note that this Correlation coefficient has the properties as follows:

1) Commutativity:

$$\rho(X,Y) = \rho(Y,X)$$

2) Affine Invariant:

$$\rho(aX+b, cY+d) = \rho(X,Y)$$

Affine Invariant property tells us that the linear dependency are not affected by the scale factor on the random variables, which means we can discover the underlying dependency of two random variables regardless of stretches as well as deviations.

3) Range between -1 and 1

$$-1 \leq \rho(X,Y) \leq 1$$

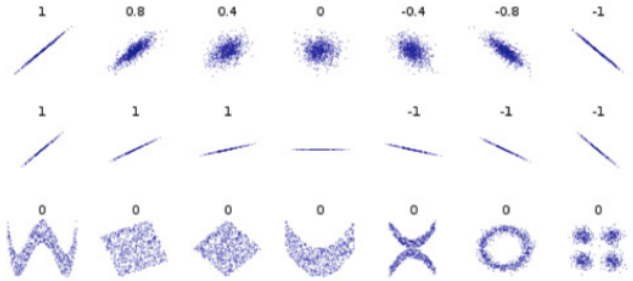

Figure 1. Pearson correlation coefficients for different shapes of data.

Fig.1 shows the linear dependency between two random variables measured by Pearson correlation coefficient. The left-most column reveals that +1 represents for positive linear correlated and -1 verse vice. It is worth to mention that nonlinear structure in the third row cannot be captured by this correlation coefficient, thus the correlation equals to 0.

In a real dataset, we can only calculate the empirical expectation and covariance matrix. Thus, the correlation can be calculated as,

$$\rho(X,Y) = \frac{\sum_{i=1}^{n}(x_i - \tilde{x})\,(y_i - \tilde{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \tilde{x})^2 \sum_{i=1}^{n}(y_i - \tilde{y})^2}}$$

B. *Hirschfeld-Gebelein-Renyi Maximal Correlation*

Since the Pearson Correlation Coefficient only categories the linear dependency between random variable *X* and *Y*, clearly this metric dissatisfied the demands of most cases. Consider function, shown in Fig.

$$y = e^{sinx+4}$$

If we consider *x* and *y* are values of two random variables *X* and *Y*. It is obvious that there is nonlinear relation between *X* and *Y*. However, Pearson Correlation Coefficient shows that *X* and *Y* are irrelevant.

In this case, we define *Hirschfeld-Gebelein-Renyi maximal correlation* $\rho_{max}(X,Y)$[Renyi, 1959], which measures the nonlinear correlations.

$$\rho_{max}(X,Y) \triangleq \sup_{\substack{f:\mathcal{X}\to\mathbb{R},g:\mathcal{Y}\to\mathbb{R} \\ \mathbb{E}[f(X)]=\mathbb{E}[g(Y)]=0 \\ \mathbb{E}[f^2(X)]=\mathbb{E}[g^2(Y)]=0}} \mathbb{E}[f(X)g(Y)]$$

The main properties are listed as follows:

1) For bijective function, $f:\mathcal{X}\to\mathbb{R}, g:\mathcal{Y}\to\mathbb{R}$,

$$\rho_{max}\big(f(X),g(Y)\big) = \rho_{max}(X,Y)$$

Affine Invariant property tells us that the linear dependency are not affected by the scale factor on the random variables, which means we can discover the underlying dependency of two random variables regardless of stretches as well as deviation.

2) Range between -1 and 1

$$0 \leq \rho_{max}(X,Y) \leq 1$$

Some key points are:
When $\rho_{\max}(X,Y) = 1$, there exists function s.t. $f(X) = g(Y)$;
When $\rho_{\max}(X,Y) = 0$, *X* and *Y are independent.*

3) If X and Y are jointly Gaussian distribution,

$$\rho_{max}\big(f(X),g(Y)\big) = |\rho(X,Y)|$$

This means that, in Gaussian cases, the two correlations are equivalent, except for the absolute value.

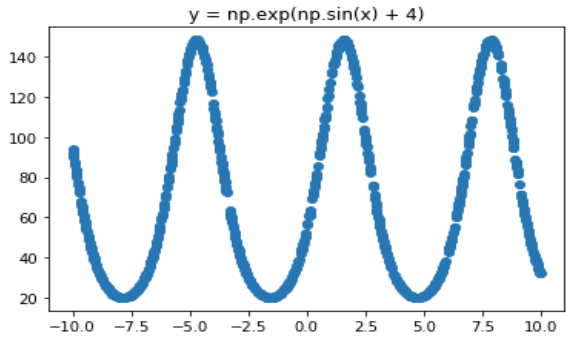

Figure 2. it is hard to estimate a nonlinear function using CCA.

To sum up, this section introduces two definition of correlation coefficient i.e. Pearson and HGR correlation. In the next few sections, we will use the former one to find an optimal linear transformation a^TX and b^TY, and the later one to capture the optimal nonlinear transformation $g(X)$ and $g(Y)$. By implementing such transformations, we mapped the originally uncorrelated data into two random variables with high linear dependency. This benefits us in both understanding the patterns of the data and implementing dimension reduction.

III. Linear Feature Transformation Methods

The basic idea of Canonical Correlation Analysis is to find the optimal linear transformation of *X* and

Y that can maximize the Pearson Correlation Coefficient. However, before exploring CCA, we first examined some simple settings by using Ordinary Least Square to perform the transformations for both *X* and *Y*.

A. Ordinary Least Square for Linear Transformation

We considered a simple case where data samples $X \in \mathbb{R}^{n\times d}$ and $Y \in \mathbb{R}^{n}$. Ordinary Least Square (OLS) will solve this optimization problem,

$$\underset{w}{\text{minimize}} \|Y - w^T X\|_2^2$$

And the solution is,

$$w = (X^T X)^{-1} X^T Y$$

assuming $(X^T X)^{-1}$ is invertible. Now, we reconstruct our model as,

$$\underset{v,w}{\text{minimize}} \left\| v^T Y_{poly,norm} - w^T X_{norm} \right\|_2^2$$

By drawing the intention of CCA, in which we want to linearly transform both X and Y, we pick a polynomial feature Y_{poly} of Y, then normalize both X and Y by subtracting the mean.

$$Y_{poly,norm} = Y_{poly} - \bar{Y}$$

$$X_{poly,norm} = X_{poly} - \bar{X}$$

After taking the derivative to both *v* and *w,* we have

$$v = \left(Y_{poly,norm}^T Y_{poly,norm}\right)^{-1} \left(Y_{poly,norm}^T X_{poly,norm}\right) w$$

$$w = \left(X_{poly,norm}^T X_{poly,norm}\right)^{-1} \left(X_{poly,norm}^T Y_{poly,norm}\right) v$$

Then we set,

$$\Sigma_{YY} = Y_{poly,norm}^T Y_{poly,norm}$$

$$\Sigma_{XX} = X_{poly,norm}^T X_{poly,norm}$$

$$\Sigma_{XY} = X_{poly,norm}^T Y_{poly,norm}$$

$$v = \Sigma_{YY}^{-\frac{1}{2}} v^*$$

$$w = \Sigma_{XX}^{-\frac{1}{2}} w^*$$

where $\Sigma_{YY}^{-\frac{1}{2}} = U_{YY} \mathrm{D}_{YY}^{-\frac{1}{2}} \mathrm{V}_{YY}^T$, $\Sigma_{XX}^{-\frac{1}{2}} = U_{XX} \mathrm{D}_{XX}^{-\frac{1}{2}} \mathrm{V}_{XX}^T$

We found that $\Sigma_{YY}^{\frac{1}{2}} v$ is the largest eigenvector of a matrix,

$$\Sigma_{YY}^{-\frac{1}{2}} \Sigma_{XY}^T \Sigma_{XX}^{-1} \Sigma_{XY} \Sigma_{YY}^{-\frac{1}{2}}$$

and $\Sigma_{XX}^{\frac{1}{2}} v$ is the largest eigenvector of a matrix,

$$\Sigma_{XX}^{-\frac{1}{2}} \Sigma_{XY} \Sigma_{YY}^{-1} \Sigma_{XY}^T \Sigma_{XX}^{-\frac{1}{2}}$$

Later, we will show CCA gives the same solution.

B. Canonical Correlation Analysis

Now, we will go through CCA of which the objective function is,

$$\max_{u,v} \rho\left(w^T X, v^T Y\right)$$

which means we want to maximize the linear dependency after transforming X and Y. Given the definition above, the problem can be written as,

$$\max_{w,v} \frac{w^T \Sigma_{XY} v}{(w^T \Sigma_{XX} w)^{\frac{1}{2}} (v^T \Sigma_{YY} v)^{\frac{1}{2}}}$$

In this maximization problem, the objective is invariant of the scales of *w* and *v.* Thus, we assume they are both normalized. The solution for the optimization problem is as follows.

$\Sigma_{YY}^{\frac{1}{2}} v$ is the largest eigenvector of a matrix,

$$\Sigma_{YY}^{-\frac{1}{2}} \Sigma_{XY}^T \Sigma_{XX}^{-1} \Sigma_{XY} \Sigma_{YY}^{-\frac{1}{2}}$$

$\Sigma_{XX}^{\frac{1}{2}} v$ is the largest eigenvector of a matrix,

$$\Sigma_{XX}^{-\frac{1}{2}} \Sigma_{XY} \Sigma_{YY}^{-1} \Sigma_{XY}^T \Sigma_{XX}^{-\frac{1}{2}}.$$

In Part A and B, we conclude that in our case, CCA is equivalent to Ordinary Least Square. Later, in section five, we explore the performance of CCA on extracting linear dependency of polynomial, continuous (not polynomial) and discontinuous functions.

IV. Nonlinear Transformation Methods

In the previous part, we performed linear transformation on X and the polynomial features of Y. Now, we generalize the linear case into non-linear transformation by using a non-parametric algorithm called Alternating Conditional Expectation (ACE). However, instead of introducing ACE directly, we will explore the intuition behind it, which is mean

square error.

A. Mean Square Error

Suppose we have random variable X and Y, we intend to find a transformation of X and Y, i.e. $f(X)$ and $g(Y)$.

We would like to minimize the mean square error of these two transformations. And suppose we are looking for the normalized feature of both X and Y, since the scale of transformed feature will affect the objective value, but make no difference to the vector itself. Then we define MSE as,

$$MSE(X,Y) = \underset{\substack{\mathbb{E}[f(X)]=\mathbb{E}[g(Y)]=0 \\ \mathbb{E}[f(X)^2]=\mathbb{E}[g(Y)^2]=1}}{\text{minimize}} \mathbb{E}[(f(X)-g(Y))^2]$$

By expanding this minimization problem, we achieve,

$$MSE(X,Y) = 2 - 2\rho_{max}(X,Y)$$

Thus, we change the original optimization problem into maximizing the HGR maximal correlation $\rho_{max}(X,Y)$. This illustrates that minimize the mean square error of the two transformations are indeed equivalent to maximize the HGR correlation coefficient. Next, we introduce ACE to solve this optimization problem.

B. Alternating Conditional Expectation

The objective for ACE, as mentioned before, is to maximize maximal correlation. However, ACE can be applied to solve more generalized cases, in which X consists of multiple random variables,

$$\underset{\substack{\mathbb{E}[f_i(X)]=\mathbb{E}[g(Y)]=0 \\ \mathbb{E}[f_i(X)^2]=\mathbb{E}[g(Y)^2]=1}}{\text{minimize}} \mathbb{E}\left[\left(g(Y)-\sum_{i=1}^{p} f_i(X)\right)^2\right]$$

This minimization problem is calculated by implementing series of single function minimizations.

$$f_i(X) = \mathbb{E}\left[g(Y)-\sum_{j\neq i}^{p} f_j(X)\,|X_i\right]$$

$$g(X) = \mathbb{E}\left[\sum_{i=1}^{p} f_j(X)\,|Y\right] \Big/ \left\|\mathbb{E}\left[\sum_{i=1}^{p} f_j(X)\,|Y\right]\right\|$$

ACE Algorithm

Set $\theta(Y) = Y/\|Y\|$ and $\phi_1(X_1), \ldots, \phi_p(X_p) = 0$;
Iterate until $e^2(\theta, \phi_1, \ldots, \phi_p)$ fails to decrease;
Iterate until $e^2(\theta, \phi_1, \ldots, \phi_p)$ fails to decrease;
For $k = 1$ to p Do:
  $\phi_{k,1}(X_k) = E[\theta(Y) - \sum_{i\neq k} \phi_i(X_i) \mid X_k]$;
  replace $\phi_k(X_k)$ with $\phi_{k,1}(X_k)$;
End For Loop;
End Inner Iteration Loop;
  $\theta_1(Y) = E[\sum_{i=1} \phi_i(X_i) \mid Y]/\|E[\sum_{i=1}^{p} \phi_i(X_i) \mid Y]\|$;
  replace $\theta(Y)$ with $\theta_1(Y)$;
End Outer Iteration Loop;
  $\theta, \phi_1, \ldots, \phi_p$ are the solutions $\theta^*, \phi_1^*, \ldots, \phi_p^*$;
End ACE Algorithm.

Figure 3. Pseudocode for ACE

ACE solves an optimization problem by alternatively updating above two steps. The implementation of this algorithm is listed as above [Breiman, 1985].

V. Experiment

In this section, we tested CCA, ACE and then compared them with Neural Network on randomly generated data sets. Training data and test data were randomly generated using functions in the form of

$$y = g^{-1}(f(x) + error)$$

where $g: \mathbb{R} \to \mathbb{R}$, $f: \mathbb{R} \to \mathbb{R}$ and $error \sim N(0,1)$.

We chose three types of function, f: polynomial, continuous (non-polynomial) and discontinuous functions.

The dependent variable, x, was generated from $x \sim Uniform(-5,5)$, and response variable, y, was generated from $y = g^{-1}(f(x) + error)$. Using the generated data, we compared the performance of the ACE, CCA and Neural Network. Furthermore, to evaluate the performance of ACE on the case of multi-variable features, we compared it with a neural network using the classic example of ACE [D. Wang, 2004].

According to the paper [Breiman 1985], ACE can estimate any underlying functions, g and f. However, CCA performs only linear transformation on random variables. Therefore, in most cases, ACE will outperforms CCA. To make CCA more compatible with ACE, we gave an advantage to CCA by allowing it to use polynomial features on both X and Y to recover the intrinsic patterns in the data samples. This intuition originated from Taylor Approximation. By extracting polynomial features, CCA can project the features onto a particular direction. We interpreted this as it intends to find a

set of parameters that can best estimate the intrinsic functions of X and Y, which corresponds to f and g.

Both CCA and ACE ends up with finding a transformation of X and Y, $f(X)$ and $g(Y)$. However, to compare it with Neural Network, we need to test our result on test data. We use fourth order polynomial to fit the g^{-1} function. After training the data, we obtained $\hat{f}$ and $\hat{g}^{-1}$. Then we calculated $y_{test} = \hat{g}^{-1}\left(\hat{f}(x_{test})\right)$. The followings are the descriptions of the functions we used.

A. Polynomial Function

$$y = (\frac{1}{2}x^2 - x + 4 + 0.1 * error)^{\frac{1}{3}}$$

where $x \sim Uniform(-5, 5)$ and $error \sim Normal(0, 1)$

We treated a polynomial function as the simplest case since we used the polynomial features on both x and y for CCA.

B. Continuous Function

$$y = \exp(\sin x + 0.1 * \text{error})$$

where x $\sim Uniform(-5, 5)$ and $error \sim Normal(0, 1)$

This function is a "medium level" function since both exponential and sin functions can be well estimated by Taylor Expansion.

C. Discontinuous Function

$$y = (|x^2 + x - 2| + 2 + 0.1 * error)^{\frac{1}{2}}$$

where x $\sim Uniform(-5, 5)$ and $error \sim Normal(0, 1)$

We set this case to test CCA because Taylor Approximation does not work on discontinuous case.

As shown in the fig.4 below, ACE and Neural Network can accurately estimate the underlying functions, $g^{-1}f: \mathbb{R} \to \mathbb{R}$. CCA estimates the underlying function relatively worse than the others. In fact, for polynomial function, CCA gives the lowest error as shown in table 1. But, for continuous and discontinuous case, it performs much worse than the others.

The result was consistent with our expectation. Since we used the polynomial features to estimate non-polynomial functions, the accuracies of CCA were highly sensitive to the degrees of polynomials. It was because some degrees of polynomials describe the underlying functions well while other degrees of polynomials do not describe the underlying functions accurately even though it maximizes the objective function, $\rho(w^T X, v^T Y)$. Unlike CCA, the neural network is relatively stable. Even with the 1 hidden layer, the neural network gives accurate estimation as long as it is trained with enough epochs. Finally, ACE gave the compatible result with the neural network. In fact, ACE might be more useful since it does not need any process of tuning hyper-parameters. In practice, it is highly powerful especially when dealing with the large data set.

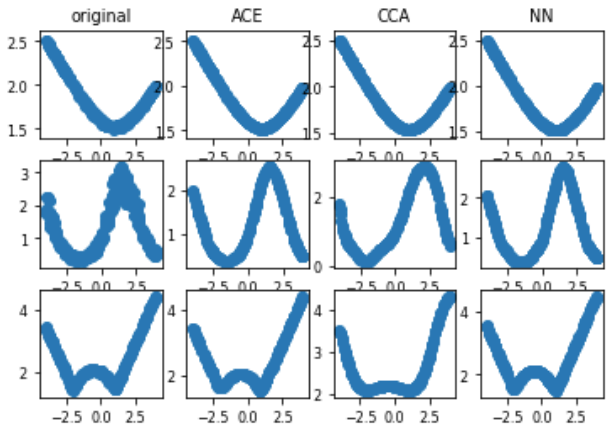

Figure 4. The plots of the original function and estimated functions using ACE, CCA and Neural Network. From the first row, each row corresponds to the different functions: polynomial, continuous and discontinuous functions as introduced before. ACE and Neural Network accurately estimated the functions while CCA estimated relatively worse than ACE or Neural Network.

	ACE	CCA	NN
poly	0.000140	0.000093	0.000163
sin	0.030014	0.074110	0.024029
discont	0.003284	0.293681	0.002457

Table 1. The average errors of ACE, CCA and Neural Network on test data.

VI. Conclusion

In this project, we explore both the linear and nonlinear transformation of X and Y. We first stated that both methods can be derived by minimizing the squared errors (OLS and MSE). Then, we compare CCA, ACE and Neural Network using different functions. Our conclusion is that they all have advantages and disadvantages. For CCA, although it only performs linear transformation, it gives the accurate estimation on polynomial underlying functions. As mentioned above, CCA is sensitive to

the degree of polynomial features, it can be tuned using validation data. Since CCA is linear transformation, it is relatively simple to reveal the true Y from $g(Y)$. ACE could recover all the three nonlinear underlying functions with significantly small errors on the test data. However, in the discontinuous case, the prediction was not as good as Neural Network. Our final point is ACE indeed has strong potential in estimating underlying patterns for a large data since this nonparametric method can achieve optimal transformation without tuning parameters and has strong theoretical backup in probability and information theory senses.

REFERENCES

[1] Breiman, Leo, and Jerome H. Friedman. "Estimating optimal transformations for multiple regression and correlation." Journal of the American statistical Association 80.391 (1985): 580-598.

[2] Stella Yu. (2017) University of California, Berkeley, On Canonical Correlation Analysis [Online]. Available: http://www1.icsi.berkeley.edu/~stellayu/1/20170926cca.pdf

[3] Rényi, Alfréd. "On measures of dependence." *Acta mathematica hungarica* 10.3-4 (1959): 441-451.

[4] Wang, Duolao, and Michael Murphy. "Estimating optimal transformations for multiple regression using the ACE algorithm." *Journal of Data Science* 2.4 (2004): 329-346.

[5] Michaeli, Tomer, Weiran Wang, and Karen Livescu. "Nonparametric canonical correlation analysis." *International Conference on Machine Learning*. 2016.

[6] Ryan, (2013) Measures of correlation, Online. Available: http://www.stat.cmu.edu/~ryantibs/datamining/lectures/12-cor3-marked.pdf