Kernel assignment: advanced topics in machine learning

YUAN ZHANG

SN: 17044633

Data: 24/11/2017

1 Feature Spaces

1.1 Describe a simple feature space

The proper classification for the left figure should be the black circle in the right figure.

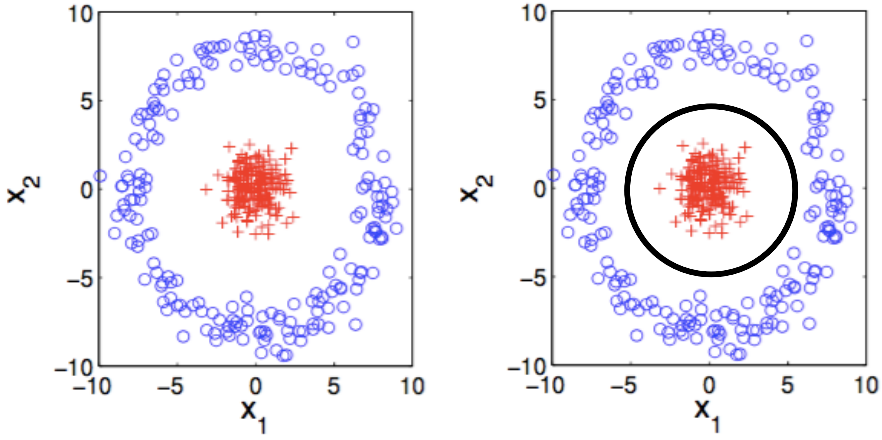


Figure 1 Dataset Figure and Its Classification Figure

We need to add a dimension to represent this classification. This dimension is the distance between data point and origin point, which is $x_1^2+x_2^2$.

So the feature space should be $\phi(x)=\begin{bmatrix} x_1 & x_2 & x_1^2+x_2^2 \end{bmatrix}^T$.

1.2 Derive the feature space

The eigendecomposition of K is $K=U\Lambda U^T$, $U=[\vec{u}_1 \ \cdots \ \vec{u}_m]$ Λ is a diagonal matrix with eigen values λ_i on the diagonal. K is positive semidefinite so λ_i's are nonnegative.

So $K=U\Lambda U^T=U\sqrt{\Lambda}*\sqrt{\Lambda}^T U^T=U\sqrt{\Lambda}\left(U\sqrt{\Lambda}\right)^T=MM^T$, $M=[\sqrt{\lambda_1}\vec{u}_1,\cdots,\sqrt{\lambda_m}\vec{u}_m]$

$$K_{ij}=\left\langle \Phi(x_i),\Phi(x_i)\right\rangle_{\mathrm{H}}=\left(MM^T\right)_{ij}=\left[\sqrt{\lambda_1}u_1^i,\cdots,\sqrt{\lambda_m}u_m^i\right]\left[\sqrt{\lambda_1}u_1^j,\cdots,\sqrt{\lambda_m}u_m^j\right]^T.$$

$$\therefore \Phi(x_i)=\left[\sqrt{\lambda_1}u_1^i,\cdots,\sqrt{\lambda_m}u_m^i\right]$$

2 Kernel dependence detection

2.1 Incomplete Cholesky for efficient COCO

For COCO problems, the solution is to solve the following generalized eigenvalue problems.

$$\begin{bmatrix} 0 & \frac{1}{n}\tilde{K}\tilde{L} \\ \frac{1}{n}\tilde{L}\tilde{K} & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \gamma \begin{bmatrix} \tilde{K} & 0 \\ 0 & \tilde{L} \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

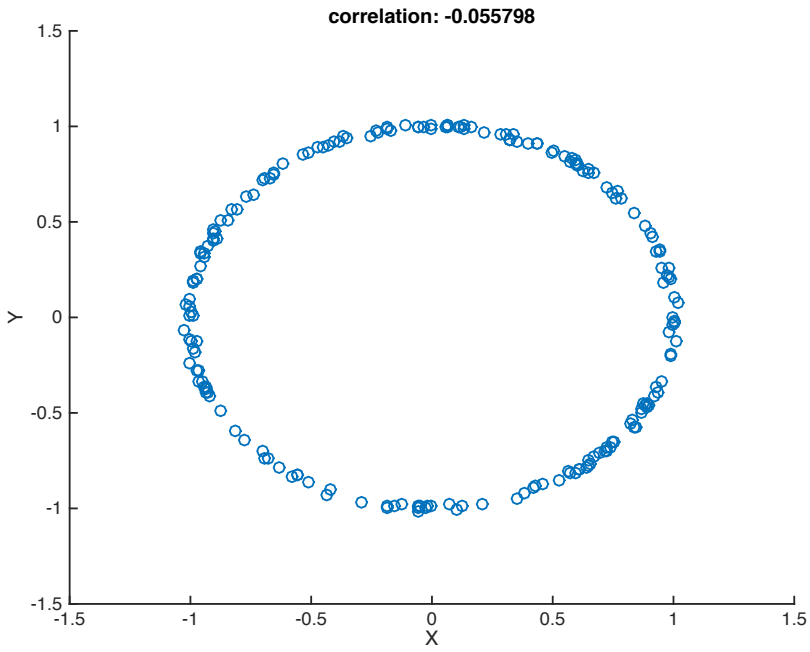
As this equation Ax= γ Bx is solved by MATLAB pack, we need to focus on the complexity of how to generalize A & B.

For the COCO computed exactly, $\tilde{K} = HKH, \tilde{L} = HLH$. These are all n*n matrix. The product of two n*n matrix needs n^3 multiplications and n^2*(n-1) additions. So the complexity of computing $\tilde{K}$ takes 2*n^3 multiplications and 2*n^2*(n-1) additions (two times matrix multiplications). Matrix $\tilde{L}$ is the same. For $\frac{1}{n}\tilde{K}\tilde{L}$, it needs n^3 + n^2 multiplications and n^2*(n-1) additions. $\frac{1}{n}\tilde{L}\tilde{K}$ is the same. So the total complexity is 6*n^3 + 2n^2 multiplications and 6*n^2*(n-1).

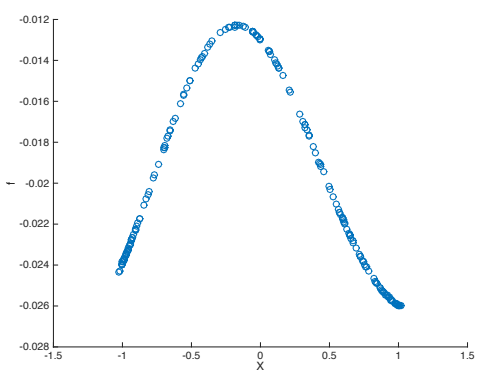
For cholesky-based COCO, $\tilde{K} = HKH = HR^T RH = (RH)^T RH$. R is t*n, H is n*n. So RH needs t*n^2 multiplications and t*n*(n-1) additions. RH is t*n, to compute $\tilde{K}$ from RH, t*n^2 multiplications and n*n*(t-1) additions are needed. The total cost of $\tilde{K}$ is 2* t*n^2 multiplications and n*(2nt-n-t) additions. $\tilde{L}$ is the same. For $\frac{1}{n}\tilde{K}\tilde{L}$, it needs n^3 + n^2 multiplications and n^2*(n-1) additions. $\frac{1}{n}\tilde{L}\tilde{K}$ is the same. So the total complexity is 4*t*n^2 + 2*n^3 + 2n^2 multiplications and 2* n*(2nt-n-t) + 2*n^2*(n-1).

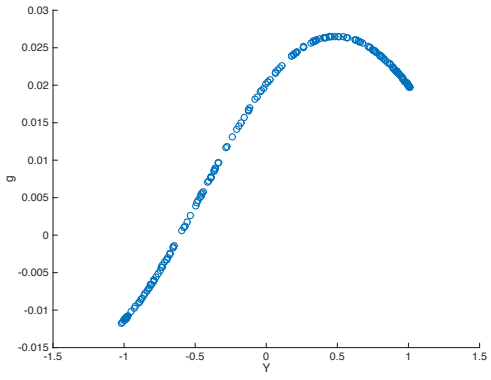
When t<n, the complexity of cholesky-based COCO is lower than exact COCO.

In practice, we set the threshold residual cutoff eta equals to 0.00001. The matrix R's size is 6*200. Compared with K's size 200*200, the complexity really decreases.
We can run COCO.m code to generate data (x, y) as below.

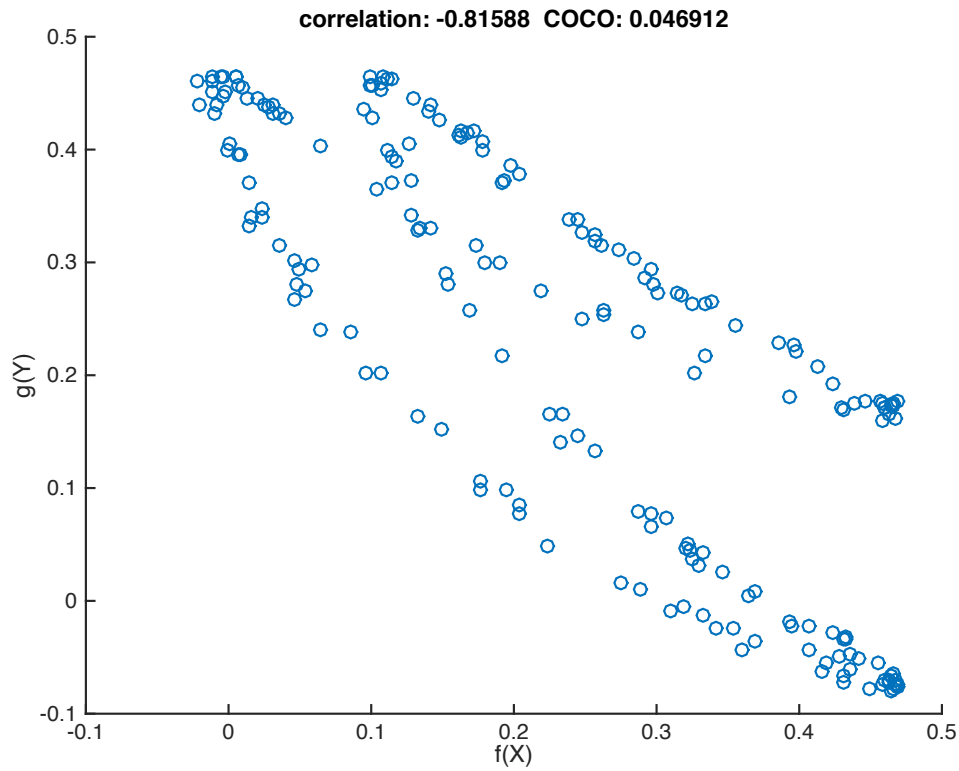


Then implement Cholesky-base COCO in MATLAB. We can get plot of our best-evaluated f and g as below.





Then map (x, y) via f and g functions, scatter the mapping results as below.



We can see that after mapping our data becomes more dependent (the correlation changes from -0.056 to -0.816), which gives us a better change to see the relationship between x and y.

2.1 Kernel CCA

(1) Kernel solution to the CCA problem.

Kernel CCA

Our goal: $\max \langle f, \hat{C}_{XY} g \rangle_F$

Subject to $\langle f, \hat{C}_{XX} f \rangle_F = 1, \quad \langle g, \hat{C}_{YY} g \rangle_G = 1$

Lagrange function:

$$L(f, g, \lambda, \gamma) = \langle f, \hat{C}_{XY} g \rangle_F - \frac{\lambda}{2}\left(\langle f, \hat{C}_{XX} f \rangle_F - 1\right) - \frac{\gamma}{2}\left(\langle g, \hat{C}_{YY} g \rangle_G - 1\right)$$

We have $X = [\vec{\phi}(x_1), \cdots, \vec{\phi}(x_n)]$, $Y = [\vec{\psi}(y_1), \cdots, \vec{\psi}(y_n)]$

$\hat{C}_{XY} = \frac{1}{n} XHY^T$, $\hat{C}_{XX} = \frac{1}{n} XHX^T$, $\hat{C}_{YY} = \frac{1}{n} YHY^T$, $f = XH\alpha$, $g = YH\beta$

Take these into $L(f, g, \lambda, \gamma)$, we get

$$L = \frac{1}{n}\alpha^T H^T X^T X H Y^T Y H \beta - \frac{\lambda}{2}\left(\alpha^T H^T X^T \cdot \frac{1}{n} X H X^T X H \alpha - 1\right) - \frac{\gamma}{2}\left(\beta^T H^T Y^T \frac{1}{n} Y H Y^T Y H \beta - 1\right)$$

$\because H = H^T$, $H^2 = H$. $\tilde{K} = HKH = HX^T XH$; $\tilde{L} = HLH = HY^T YH$

$$\therefore L = \frac{1}{n}\alpha^T \tilde{K}\tilde{L}\beta - \frac{\lambda}{2}\left(\frac{1}{n}\alpha^T \tilde{K}^2 \alpha - 1\right) - \frac{\gamma}{2}\left(\frac{1}{n}\beta^T \tilde{L}^2 \beta - 1\right)$$

Derive wrt α, β, we get

$$\tilde{K}\tilde{L}\beta = \lambda \cdot \tilde{K}^2 \alpha \quad ①$$

$$\tilde{L}\tilde{K}\alpha = \gamma \tilde{L}^2 \beta \quad ②$$

$\alpha^T \times ① - \beta^T \times ②$,

$$\alpha^T \tilde{K}\tilde{L}\beta - \beta^T \tilde{L}\tilde{K}\alpha = \lambda \alpha^T \tilde{K}^2 \alpha - \gamma \beta^T \tilde{L}^2 \beta = \lambda - \gamma$$

$\therefore \lambda = \gamma$

Our restrictions become $\begin{cases} \tilde{K}\tilde{L}\beta = \lambda \tilde{K}^2 \alpha \\ \tilde{L}\tilde{K}\alpha = \lambda \tilde{L}^2 \beta \end{cases}$

which is $\begin{bmatrix} 0 & \tilde{K}\tilde{L} \\ \tilde{L}\tilde{K} & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \lambda \cdot \begin{bmatrix} \tilde{K}^2 & 0 \\ 0 & \tilde{L}^2 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$

This is generalised eigen value problems

The largest eigen value λ^*.

Then the max solution of CCA is $\alpha^T \tilde{K}\tilde{L}\beta \cdot \frac{1}{n} = \frac{\lambda^*}{n}$

(2) Regularizations

What went wrong?

Our CCA problem equlas to

$$\max_{\alpha,\beta} \frac{\alpha^T \tilde{K}\tilde{L}\beta}{(\alpha^T \tilde{K}^2\alpha)^{\frac{1}{2}}(\beta^T\tilde{L}^2\beta)^{\frac{1}{2}}} = \max_{V_1,V_2} \frac{V_1^T \cdot V_2}{\|V_1\|\,\|V_2\|} = \max_{V_1,V_2} \cos(V_1,V_2)$$

(Used by Bach and Jordan (2002a))

$V_1 \in V_1$, which is the colum space of $\tilde{K}$

$\tilde{K} = HKH$ is a subspace orthogonal to the vector composed of all ones. $\tilde{L}$ is the same.

As a result V_1 and V_2 are identical.

Because of free α,β, then max cosine will always be one, no matter different good ~~$\in$~~ inputs.

∴ We need to use regularization to restrict f, g (α,β)

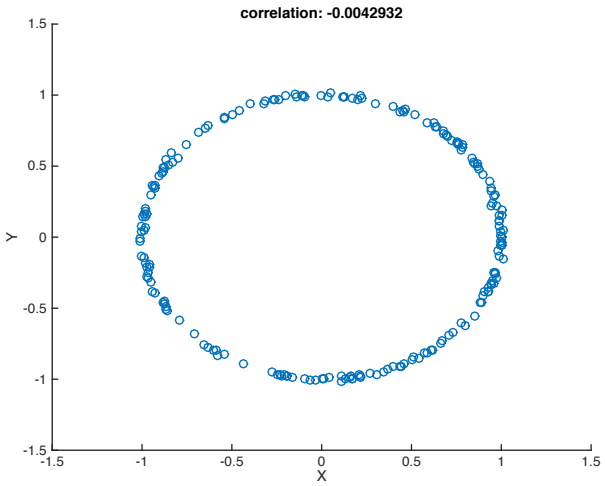
$$\langle f,f\rangle_F \le 1,\ \langle g,g\rangle_G \le 1 \Rightarrow \begin{cases} \tilde{K}\tilde{L}\beta = \lambda(\tilde{K}^2 + t\tilde{K})\alpha \\ \tilde{L}\tilde{K}\alpha = \lambda(\tilde{L}^2 + t\tilde{L})\beta \end{cases}$$

which is

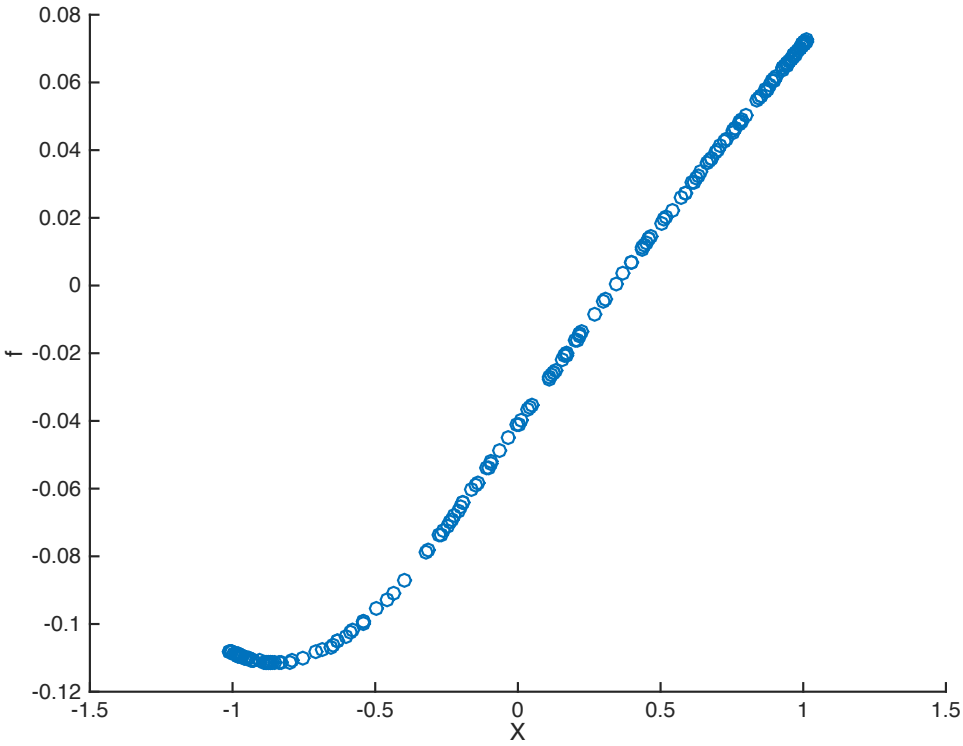
$$\begin{bmatrix} 0 & \tilde{K}\tilde{L} \\ \tilde{L}\tilde{K} & 0 \end{bmatrix}\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \lambda\begin{bmatrix} \tilde{K}^2 + t\tilde{K} & 0 \\ 0 & \tilde{L}^2 + t\tilde{L} \end{bmatrix}\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

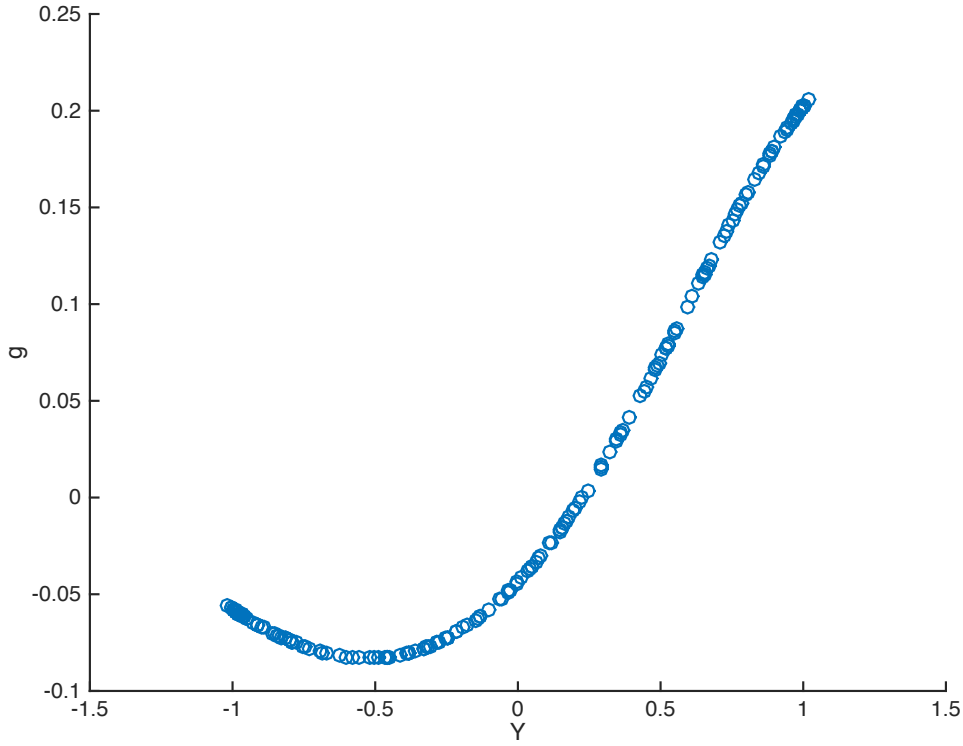
We have our CCA solutions.

Implement kernel CCA as above in MATLAB. We can run code to generate data (x, y) as below.

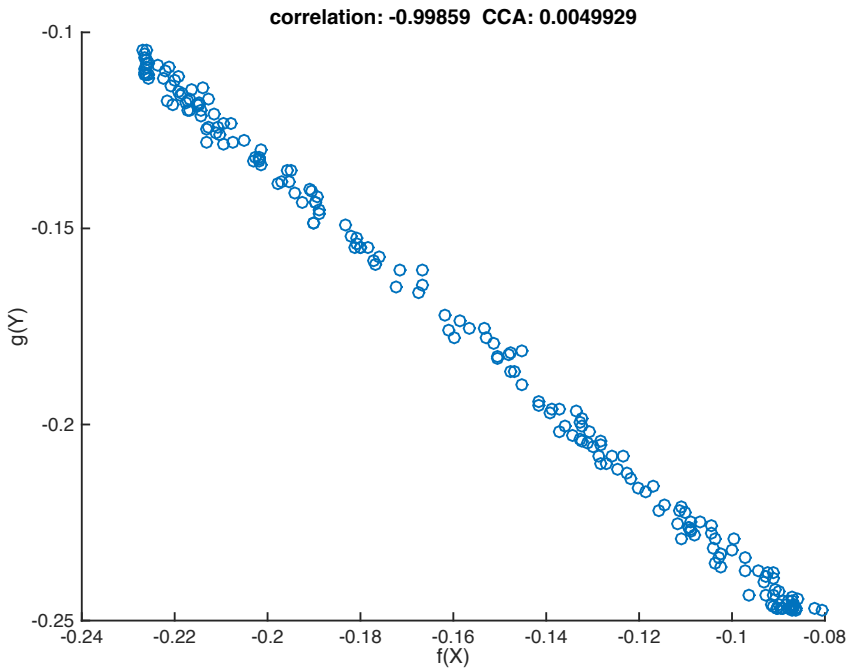


We can get plot of our best-evaluated f and g as below.



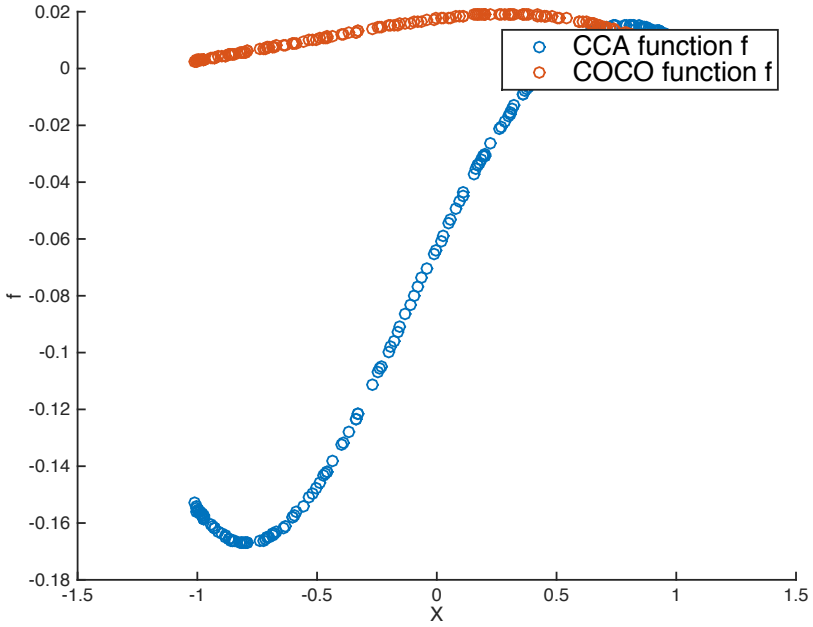


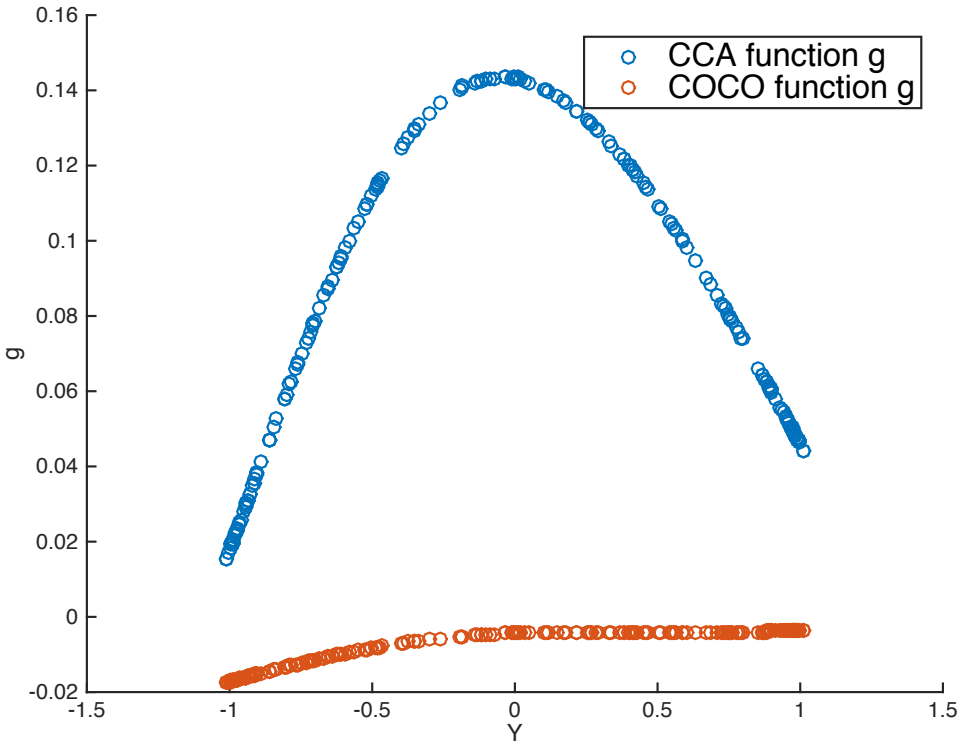
Then map (x, y) via f and g functions, scatter the mapping results as below.



We can see that after mapping our data becomes more dependent (the correlation changes from -0.004 to 0.999), which gives us a better change to see the relationship between x and y.

We now compare COCO and CCA using the same data.
Firstly, we compare the functions generated from two methods. We plot functions in the same figure as below.





We can see that functions generated from COCO have a much narrower range compared with those generated from CCA. This is because the constraints of two methods are different. So that it is natural that mapping with CCA gives a larger correlations.

However, computing CCA is more expensive than COCO, especially for the larger dataset.