Fast KDAC

February 20, 2017

Given the objective function :

$$\begin{aligned} \min_{W} \quad & -\sum_{i,j} \gamma_{i,j} e^{-\frac{\operatorname{tr}\left(W^T A_{i,j} W\right)}{2\sigma^2}} \\ s.t \quad & W^T W = I \\ & W \in \mathbb{R}^{d \times q} \\ & A \in \mathbb{R}^{d \times d} \\ & \gamma_{i,j} \in \mathbb{R} \end{aligned} \tag{1}$$

To optimize this cost function, the original KDAC uses an optimization technique called Dimensional Growth (DG). It rewrites the cost function into separate columns of the W matrix, and solve the problem one column at a time in a Greedy fashion.

$$\begin{aligned} \min_{W} \quad & -\sum_{i,j} \gamma_{i,j} e^{-\frac{w_1^T A_{i,j} w_1}{2\sigma^2}} e^{-\frac{w_2^T A_{i,j} w_2}{2\sigma^2}} \ldots e^{-\frac{w_q^T A_{i,j} w_q}{2\sigma^2}} \qquad w_i = i\,\text{th column of } W \\ s.t \quad & W^T W = I \end{aligned}$$

For example, to solve the first column w_1, it ignores the rest of the columns and simplify the problem into :

$$\begin{aligned} \min_{w_1} \quad & -\sum_{i,j} \gamma_{i,j} e^{-\frac{w_1^T A_{i,j} w_1}{2\sigma^2}} \\ s.t \quad & w_1^T w_1 = 1 \end{aligned}$$

This problem could be solved using standard Gradient methods. Once w_1 is computed, DG then treats the exponential term as a constant $g(w_1)$ and solve for the next variable.

$$f(w_2) = -\sum_{i,j} \gamma_{i,j} e^{-\frac{w_1^T A_{i,j} w_1}{2\sigma^2}} e^{-\frac{w_2^T A_{i,j} w_2}{2\sigma^2}}$$

$$f(w_2) = -\sum_{i,j} \gamma_{i,j}\, g(w_1) e^{-\frac{w_2^T A_{i,j} w_2}{2\sigma^2}}$$

Without the orthgonality constraint, each stage of the optimization process could be solved using Gradient methods. However, the orthogonality constraint of $W^T W = I$ requires further complication to ensure compliance. To start, the initialization of each new column w_i must go through the Gram Schmit method to ensure its orthogonality against all previous columns. Further more, the gradient direction calculated during each iterations also must undergo Gran Schmit. By removing components from previous vectors, it ensures that each update of w_i maintains feasibility.

Dimension Growth was the original approach used to solve the optimization problem of equation (1), and it achieved its objective of demonstrating the viability of KDAC. However, as the technology approach its next developmental stage, the implementation of this technology on large scale data requires KDAC to adapt for a more implementable algorithm.

The complexity of Dimension Growth algorithm heavily increases time of code development. This issue is especially prominent when speed requirement forces the development to be done in C or on the GPU. In these cases, simpler algorithm using off the shelf techniques could significantly reduce the developmental time and therefore the cost of its implementation.

The convergence speed of Dimension Growth in KDAC is slow. As we know from optimization theory, the convergence rate for gradient methods heavily depend on the conditional value of the Hessian matrix. The conditional value is defined as the ratio between the maximum and the minimum eigenvalue of the Hessian matrix.

$$\text{condition value} = \frac{\text{eig}_{\max}(\nabla^2 f(x))}{\text{eig}_{\min}(\nabla^2 f(x))}$$

The ideal condition value is when the ratio is equal to 1 and the convergence rate slows down very quickly as we increase the condition value beyond 10. Given this fact, it would be instructive to study the Hessian matrix to potentially explain the slow convergence of gradient methods. The Hessian for each column r has the following form :

$$\nabla^2 f(w) = \sum_{i,j} \frac{\gamma_{i,j}}{\sigma^2} e^{-\frac{w_r^T A_{i,j} w_r}{2\sigma^2}} \left[A_{i,j} - \frac{1}{\sigma^2} A_{i,j} w_r w_r^T A_{i,j} \right]$$

From the Hessian matrix, we see that the condition value depends on the summation of matrix $A_{i,j}$ and $A_{i,j}w$ multiplied by some constant term, $\frac{\gamma_{i,j}}{\sigma^2} e^{-\frac{w_r^T A_{i,j} w_r}{2\sigma^2}}$. From this form, it is intuitive to find clues to size of the conditional value from the individual behaviors of $A_{i,j}$ and $A_{i,j}w$.

Due to the complexity of the Hessian form, it may be sufficiently instructive to simply look for an approximation of the Hessian to study its conditional value behavior. Given the problem :

$$f(w) = -\sum_{i,j} \gamma_{i,j} e^{-\frac{w^T A_{i,j} w}{2\sigma^2}} \tag{2}$$

We could approximate the equation (2), by using the Taylor Expansion around 0. Since gradient methods are techniques using the 1st order approximation, we could similarly make a 1st order approximate of the original function.

$$f(w) \approx -\sum_{i,j} \gamma_{i,j} \left(1 - \frac{w^T A_{i,j} w}{2\sigma^2} \right) \tag{3}$$

At this point, we find the approximate Hessian by taking the 2nd derivative.

$$\nabla f(w) \approx \sum \frac{\gamma_{i,j}}{\sigma^2} A_{i,j} w$$

$$\nabla^2 f(w) \approx \sum \frac{\gamma_{i,j}}{\sigma^2} A_{i,j}$$

From this form, we further simplied the Hessian matrix. And the simplified Hessian suggests a dominant influence of the summation of the $A_{i,j}$ matrix with some constant value $\gamma_{i,j}/\sigma^2$.

At this point, let's take a step back and ask how the eigenvalues of $A_{i,j}$ and $A_{i,j}\,w_r$ influence the conditional value depending on the type of data we handle. We first note that the $A_{i,j}$ matrix is formed with the following equation.

$$A_{i,j} = (x_i - x_j)(x_i - x_j)^T$$

Given a single gaussian cluster of data :

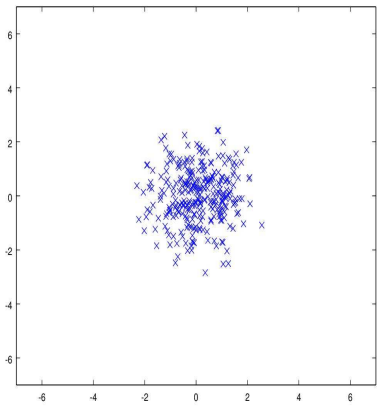

Figure 1.

If we calculate

$$A = \sum A_{i,j}$$

If we randomly generate 100 similar distributions with randomize mean, variance, sample size and dimension, the condition value stays within a small bounded range.

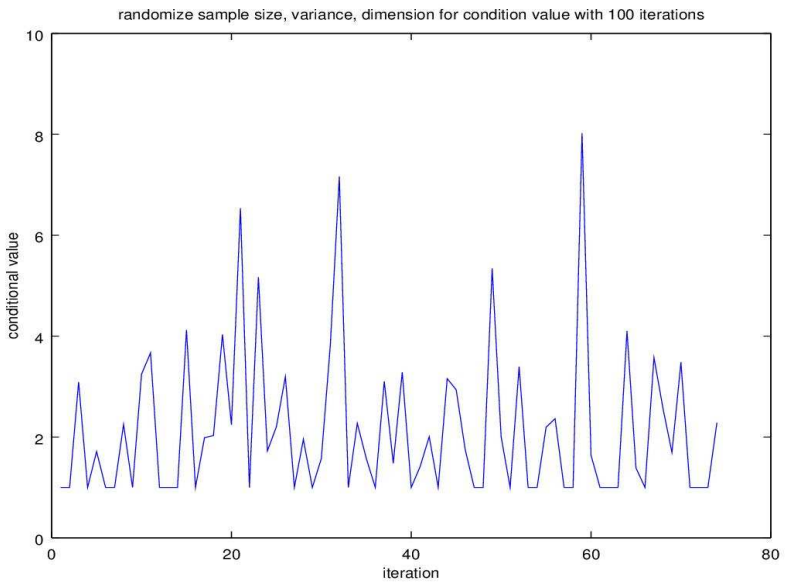

Figure 2.

This plot is generated with the following code :

```
cv = []
for k = 1:100
        n = 20; %floor(40*rand());
        d = floor(5*rand());
        A = zeros(d,d);
        p = floor(5*rand())*randn(n,d);
```

```
            for m = 1:n
                    for n = 1:n
                            v = p(m,:) - p(n,:);
                            A = A + v'*v;
                    end
            end

            [U,S,V] = svd(A);
            D = diag(S) + 1;

            %max(D)/min(D)
            cv = [cv max(D)/min(D)];

    end

    plot(cv)
    xlabel('iteration')
    ylabel('conditional value')
    title('randomize sample size, variance, dimension for condition value with 100
    iterations')
```

From the plot of the conditional value, we can conclude that the conditional value stays bounded regardless of the sample size, variance, mean and dimensionality. This is for the case of a single cohesive cluster. However, what would happen if there are multiple clusters? As we vary the number of clusters as well as their distance apart, a clear pattern imerges. The following plot shows how the conditional value of 2 and 3 gaussian distributions. As we move them further apart from each other, the condition value explodes very quickly.

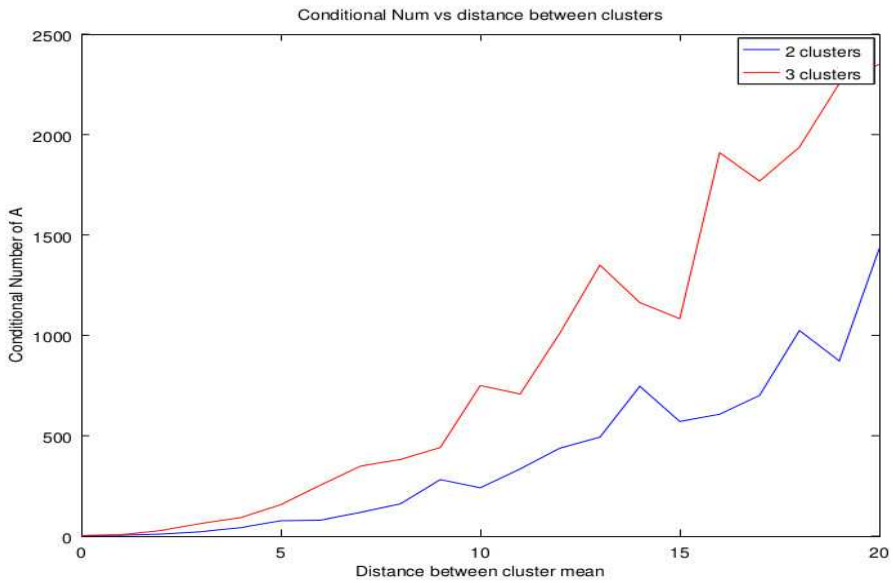

Figure 3.

In conjunction to the plot of a single cluster, this plot suggests that the conditional value is not bounded when the data type are separate into different clusters. As a matter of fact, the more clusters are involved the faster the conditional value grows. The purpose of showing ill-conditionality of this problem provide a reasoning to avoid Gradient method as an optimization approach. This allows us to lead into using a different optimization all together.

1 Fast KDAC

FKDAC is an alternative approach to Dimension Growth that estimates the optimal result without using gradient methods. We restate the optimization problem here :

$$\begin{array}{ll} \min & -\sum_{i,j} \gamma_{i,j}\, e^{-\frac{\mathrm{Tr}\left(W^T A_{i,j} W\right)}{2\sigma^2}} \\ s.t & W^T W = I \\ & W \in \mathbb{R}^{d\times q} \\ & A \in \mathbb{R}^{d\times d} \\ & \gamma_{i,j} \in \mathbb{R} \end{array} \tag{4}$$

1.1 Optimality Conditions

Due to the non-convex nature of the problem, FKDAC cannot guarentee to discover the global minimum. However, a sufficient optimality condition exists for a local minimum. According to Bertsekas in [1], Proposition 3.2.1 x^* is a optimal solution of it satisfies the following proposition.

Proposition 1. *(Second Order Sufficiency Conditions)*
Assume that f and h are twice continuously differentiable, and let $x^* \in \mathbb{R}^n$ and $\lambda^* \in \mathbb{R}^m$ satisfy the following 3 conditions :

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = 0 \qquad \text{Condition 1}$$

$$\nabla_\lambda \mathcal{L}(x^*, \lambda^*) = 0 \qquad \text{Condition 2}$$

$$z^T \nabla^2_{xx} L(x^*, \lambda^*)\, z > 0 \quad \text{for all } z \neq 0 \text{ with } \nabla h(x^*)^T z = 0 \qquad \text{Condition 3}$$

Then x^* is a strict local minimum of f subjected to h(x) = 0.

Notice that this is a very general sufficiency assumption that does not assume the form of the functions f or h. Using this requirement as a proof of optimality, we define how our cost function can satisfy the definition. We start by finding the gradient of the Lagrangian with respect to w.

$$\mathcal{L} = -\sum_{i,j} \gamma_{i,j}\, e^{-\frac{\mathrm{Tr}\left(W^T A_{i,j} W\right)}{2\sigma^2}} + \frac{\Lambda}{2}\left(1 - W^T W\right)$$

$$\frac{\partial \mathcal{L}}{\partial w} = \left[\sum_{i,j} \frac{\gamma_{i,j}}{\sigma^2}\, e^{-\frac{\mathrm{Tr}\left(W^T A_{i,j} W\right)}{2\sigma^2}} A_{i,j} W\right] - W\Lambda = 0$$

$$\left[\left[\sum_{i,j} \frac{\gamma_{i,j}}{\sigma^2}\, e^{-\frac{\mathrm{Tr}\left(W^T A_{i,j} W\right)}{2\sigma^2}} A_{i,j}\right] - \lambda I\right] W = 0$$

If we let :

$$\Phi(W) = \left[\sum_{i,j} \frac{\gamma_{i,j}}{\sigma^2}\, e^{-\frac{\mathrm{Tr}\left(W^T A_{i,j} W\right)}{2\sigma^2}} A_{i,j}\right]$$

We can rewrite the equation :

$$[\Phi(W) - \lambda I]\, W = 0$$

The equation above states that the optimal solution W^* is in the null space of the matrix $\Phi - \lambda I$. Or, from another perspective, we could rewrite the equation.

$$\Phi\,(W^*)W^* = W^*\,\Lambda \tag{5}$$

The equation above implies that the gradient of the Lagrangian is equal to 0 when W^* have columns of eigenvectors of $\Phi(w^*)$. Meeting this condition will satisfy the 1st condition of Proposition 1.

The 2nd condition of Proposition 1 requires that : $\nabla_\Lambda \mathcal{L} = 0$. From the cost function, we get :

$$\frac{\partial \mathcal{L}}{\partial\, \Lambda} = 1 - W^T W = 0 \tag{6}$$

With the conclusions from equation (5), it is obvious that if the optimal W^* have columns as unit norm eigenvectors, this condition is automatically satisfied.

To satisfy the final condtion, we first define :

$$\mathcal{D}f(X)[Z] := \lim_{t \to 0} \frac{f(X + t\,Z) - f(X)}{t}$$

We want to define a set of Z such that :

$$\nabla h(x^*)^T Z = 0$$

We know that $h(W) = W^T W - 1$, therefore :

$$\mathcal{D}h(W)[Z] = \lim_{t \to 0} \frac{(W + t\,Z)^T (W + t\,Z) - W^T W}{t} = Z^T W + W^T Z$$

From this, we define as set of Z such that :

$$S_Z = \{Z \colon Z^T W + W^T Z = 0\}$$

The 3rd optimality condition is therefore met when :

$$\mathrm{Tr}(Z^T \mathcal{D}f(W)[Z]) - \mathrm{Tr}(\Lambda\, Z^T Z) > 0 \qquad \forall\, Z \in S_Z$$

In Simpler terms, if the 2nd order gradient of the Lagrangian is Positive Definite, then the condition is satisfied. Due to the complexity of computing the Hessian in real time, a relaxation approach is discussed in a later section.

1.2 1st Order Relaxation

Given that $\Phi(W)$ is a function of W, it is difficult to take advantage of the eigenvalue/eigenvector relationship to find the solution.

$$\Phi\,(W^*)W^* = W^*\,\Lambda$$

A standard approach to circumvent this restriction is to treat W^* inside Φ as a separate variable and solve the problem iteratively.

$$\Phi\left(W_{k-1}\right)W = W\,\Lambda$$

The key is, therefore, finding a reasonable initialization value for W_0 . This section provide a theoretical suggestion for a logical initialization point. Given the following problem :

$$\min_{W} \; -\sum_{i,j} \gamma_{i,j}\, e^{-\frac{\mathrm{Tr}(W^T A_{i,j} W)}{2\sigma^2}}$$

$$s.t \qquad W^T W = I$$

Writing out the Larangian :

$$\mathcal{L} = -\sum_{i,j} \gamma_{i,j}\, e^{-\frac{\mathrm{Tr}(W^T A_{i,j} W)}{2\sigma^2}} + \frac{1}{2}\,\mathrm{Tr}(\Lambda\,(I - W^T W))$$

We simplify the Lagrangian by using the Taylor approximation on the problematic exponential term. The Taylor approximation is expanded up to the 1st order centering around 0.

$$\mathcal{L} \approx -\sum_{i,j} \gamma_{i,j} \left(1 - \frac{\mathrm{Tr}(W^T A_{i,j} W)}{2\,\sigma^2}\right) + \frac{1}{2}\,\mathrm{Tr}(\Lambda\,(I - W^T W))$$

Following the similar procedure by finding the derivative of $\mathcal{L}$ and setting it to zero.

$$\nabla\mathcal{L} \approx \sum_{i,j} \frac{\gamma_{i,j}}{\sigma^2} A_{i,j} W - W\,\Lambda = 0$$

We arrange the problem into standard eigenvalue/eigenvector problem.

$$\left[\sum_{i,j} \frac{\gamma_{i,j}}{\sigma^2} A_{i,j}\right] W = W\,\Lambda$$

From this, we see that Φ is no longer a function of W. If $W \in \mathbb{R}^{d\times q}$, the W_0 is therefore, q eigenvectors of the matrix Φ_0:

$$\Phi_0 = \sum_{i,j} \frac{\gamma_{i,j}}{\sigma^2} A_{i,j}$$

And Λ is the eigenvalue matrix.

$$\Lambda = \mathrm{eig}\left[\sum_{i,j} \frac{\gamma_{i,j}}{\sigma^2} A_{i,j}\right]$$

Using the W_0 discovered from this initialization point, we could similarly form an updated Φ_{n+1} using the original cost derivative without any approximation.

$$\Phi_{n+1} = \left[\sum_{i,j} \frac{\gamma_{i,j}}{\sigma^2}\, e^{-\frac{\mathrm{Tr}(W_n^T A_{i,j} W_n)}{2\sigma^2}} A_{i,j}\right]$$

Given that only q eigenvectors are chosen out of d total eigenvectors, and $q \ll d$. The total possible combinations could be extremely large depending on the values of d and q, more specifically it is equal to $\begin{pmatrix} d \\ q \end{pmatrix}$. Solving this problem becomes a difficult and separate combinatorial optimization.

One reasonable approach is to use the greedy algorithm to find q eigenvectors that produces the lowest cost. However, as we explore the theoretic motivation for choosing the optimal eigenvectors, we demonstrate in the next section the optimal eigenvectors without resorting to greedy methods.

2 Approximating w_{k+1}

Let q be the reduced the dimension and d as the original dimension. We have seen from the previous section that the stationary poins could be found by picking q eigenvectors from the Φ marix. However, since q could be significantly smaller than d, using the appropriate eigenvectors could effect the outcome. Without any prior knowledge, the problems requires the optimal solution to choose from potenial large outcomes of $\begin{pmatrix} d \\ q \end{pmatrix}$. To simpliy the selection process, this section provides a theoretical argument on how these vectors could be chosen intelligently. To demonstrate the idea, we start by simplifying the problem into a single column w vector. This is without much loss of generality since the idea could be generalized into higher dimensions. Again, assuming that the Lipschitz condition holds, we start by taking the 1st order Taylor Expansion around some w_k.

$$f(w) = -\sum_{i,j} \gamma_{i,j} e^{-\frac{w_k^T A_{i,j} w_k}{2\sigma^2}} + 2\left[\sum_{i,j} \gamma_{i,j} e^{-\frac{w_k^T A_{i,j} w_k}{2\sigma^2}} A_{i,j} w_k\right]^T (w_{k+1} - w_k) \tag{7}$$

The approximation generally assume that the constraint space is convex. For our case, since our constraint space is not convex, we must assume at least that the constraint space is convex within a certain radius.

$$w \in S \qquad s.t \qquad S \text{ is convex} \qquad \forall \quad \|w - w_k\| \leq \varepsilon$$

Looking at the right side of equation 6, the first term is identical to the current cost at w_k. If we assume that higher order terms are insignificant, we can achieve a lower cost as long as we pick w_{k+1} such that the 2nd term is a negative value. Let's look at the 2nd term more closely.

$$2\left[\sum_{i,j} \gamma_{i,j} e^{-\frac{w_k^T A_{i,j} w_k}{2\sigma^2}} A_{i,j} w_k\right]^T (w_{k+1} - w_k) \leq 0$$

Note that $A_{i,j}$ are symmetric. We can rewrite this expression :

$$w_k^T \left[\sum_{i,j} \gamma_{i,j} e^{-\frac{w_k^T A_{i,j} w_k}{2\sigma^2}} A_{i,j}\right] w_{k+1} \leq w_k^T \left[\sum_{i,j} \gamma_{i,j} e^{-\frac{w_k^T A_{i,j} w_k}{2\sigma^2}} A_{i,j}\right] w_k \tag{8}$$

Looking at the eqution above, variously conclusions could be drawn. First, by realizing that the quation :

$$v^T = w_k^T \left[\sum_{i,j} \gamma_{i,j} e^{-\frac{w_k^T A_{i,j} w_k}{2\sigma^2}} A_{i,j}\right]$$

is a vector, we could rewrite equation (7) as :

$$v^T w_{k+1} \leq v^T w_k$$

To draw the first conclusion, we note that $v^T w_{k+1}$ is a product of two vectors. To ensure that $v^T w_{k+1}$ is always less than or equal to $v^T w_k$, we simply pick w_{k+1} that minimizes the term $v^T w_{k+1}$.

$$\min_{w_{k+1}} v^T w_{k+1}$$

We know from geometry that the product of two vectors is minimized when they are opposite directions of each other. Therefore w_{k+1} is minimized when $w_{k+1} = -v$. From this approach, we can iteratively pick w_{k+1} by setting $w_{k+1} = -v$ to achieve a descending direction. However, this approach encounters the problem of discovering the step length accompanying the descending direction. Since the constraint space is not convex, the step length cannot be discovered by methods of interpolation.

Although finding a definite descent direction that stays on the Stiefel manifold is difficult, another insight could be drawn by making some obvious observations of the following equation.

$$w_k^T \Phi \, w_{k+1} \leq w_k^T \Phi \, w_k \tag{9}$$

Observations :

1. We have performed Taylor expansion around w_k
2. Any w chosen must have a norm of 1
3. If we have chosen a proper w_{k+1} that satisfies the inequality, it forms a descent direction.
4. If we continue to follow the descent direction, we would hit w^* at convergence
5. At the point of convergence $w_{k+1} = w_k$
6. At the point of convergence :

$$w_{k+1}^T \Phi \, w_{k+1} = w_k^T \Phi \, w_k \tag{10}$$

and no vector v exists such that :

$$w^{*T} \Phi \, w^* < v^T \Phi \, v \tag{11}$$

Conclusion :

Given the point of expansion w_k, there is no need to iteratively approach convergence because we already know w^*. The only vector w^* that satisfies equation (10) and has a norm of 1, is the least dominant eigenvector of Φ. Therefore, the least dominant eigenvector of Φ is the best approximation of w^* at the Taylor expansion of w_k.

Linking this equation to FKDAC, we have previously shown that the eigenvectors of Φ provide a stationary point such that $\nabla \mathcal{L} = 0$. However, it was not certain which eigenvectors are most appropriate. From the conclusion we have just drawn, it shows that the least dominant eigenvectors are the most reasonable.

3 Expand the same idea to multiple columns

After understanding the basic concept through the usage of a single column example, we can now expand the same idea to a more general multiple column case. Given the first order Taylor expansion.

$$f(W) = f(W_k) + \nabla f(W_k)^T (W - W_k) \tag{12}$$

Applying this equation to our problem, we get :

$$f(W) = f(w) = -\sum_{i,j} \gamma_{i,j} e^{-\frac{\mathrm{Tr}(W_k^T A_{i,j} W_k)}{2\sigma^2}} + \left[\sum_{i,j} \gamma_{i,j} e^{-\frac{\mathrm{Tr}(W_k^T A_{i,j} W_k)}{2\sigma^2}} \mathrm{Vec}(A_{i,j} W_k) \right]^T \mathrm{Vec}(W - W_k)$$

Note that the notation Vec(.) is the vectorization of a matrix. Similar to the single column case, we only need to concentrate on making the 2nd term negative.

$$\left[\sum_{i,j} \gamma_{i,j} e^{-\frac{\mathrm{Tr}(W_k^T A_{i,j} W_k)}{2\sigma^2}} \mathrm{Vec}(A_{i,j} W_k) \right]^T \mathrm{Vec}(W - W_k) \leq 0$$

Let :

$$\beta_{i,j} = \gamma_{i,j} e^{-\frac{\mathrm{Tr}(W_k^T A_{i,j} W_k)}{2\sigma^2}}$$

$$\left[\sum_{i,j} \beta_{i,j} (I \otimes A_{i,j}) \mathrm{Vec}(W_k) \right]^T [\mathrm{Vec}(W) - \mathrm{Vec}(W_k)] \leq 0$$

We now rearrange the terms and get :

$$\left[\sum_{i,j} \beta_{i,j} \left(I \otimes A_{i,j}\right) \mathrm{Vec}(W_k)\right]^T \mathrm{Vec}(W) \leq \left[\sum_{i,j} \beta_{i,j} \left(I \otimes A_{i,j}\right) \mathrm{Vec}(W_k)\right]^T \mathrm{Vec}(W_k)$$

$$\mathrm{Vec}(W_k)^T \left[\sum_{i,j} \beta_{i,j} \left(I \otimes A_{i,j}\right)\right] \mathrm{Vec}(W) \leq \mathrm{Vec}(W_k)^T \left[\sum_{i,j} \beta_{i,j} \left(I \otimes A_{i,j}\right)\right] \mathrm{Vec}(W_k)$$

Using the same logic to lower the upper bound, the W_k that minimizes the upper bound consists of the q least dominant eigenvectors of the matrix Φ.

4 Optimality condition

The Taylor expansion approach corresponds directly with the optimality condition with a convex constrained space. Given a problem of :

$$\begin{aligned} &\min \quad f(x) \\ &x \in X \end{aligned}$$

We know that given x^* as a local minimum in a convex constrained space of X, the following condition must be satisfied.

$$\nabla f(x^*)^T (x - x^*) \geq 0 \qquad \forall x \in X$$

Given that we have a non-convex constraint space, we must make more strict assumptions. Suppose x^* is a local minimum within a ball defined as $\mathcal{B}(x, \varepsilon) := \{x : \|x - x^*\| < \varepsilon\}$. Assume $\mathcal{B}(x, \varepsilon) \cap X$ is convex and f is convex within $\mathcal{B}(x, \varepsilon) \cap X$, then we state the following as the optimality condition.

$$\nabla f(x^*)^T (x - x^*) \geq 0 \qquad \forall x \in X \quad \text{and} \quad \|x - x^*\| \leq \varepsilon$$

Given a function of :

$$f(w) = -\sum \gamma_{i,j} e^{-\frac{w^T A_{i,j} w}{2\sigma^2}}$$

We create an approximate of the function :

$$f(w) \approx -\sum \gamma_{i,j} \left(1 - \frac{1}{2\,\sigma^2} w^T A_{i,j} w\right)$$

From the approximation, we find the gradient :

$$\nabla f(w) \approx \sum \frac{\gamma_{i,j}}{\sigma^2} A_{i,j} w$$

Using the approximated gradient, we could now check for the optimality condition.

$$w^{*T} \left[\sum \gamma_{i,j} A_{i,j}\right]^T (w - w^*) \geq 0$$

$$w^{*T} \left[\sum \gamma_{i,j} A_{i,j}\right]^T w \geq w^* \left[\sum \gamma_{i,j} A_{i,j}\right]^T w^*$$

From the equation above, we see that in order for w^* to be a local minimum, w^* must be chosen such that the left hand side is always larger than the right hand side for any w. This is only possible when w^* is the least dominant eigenvector of $\sum \gamma_{i,j} A_{i,j}$ matrix. From this perspective, we conclude that picking the least dominant eigenvector is a reasonable approximation for the local minimum of the original cost function.

5 Implementation Details of Cost function

The computation of cost function presented in this paper, is a complicated equation that slows down both the implementation and speed of the results.

$$\begin{aligned} \min_{W} \quad & -\sum_{i,j} \gamma_{i,j} e^{-\frac{\operatorname{tr}\left(W^T A_{i,j} W\right)}{2\sigma^2}} \\ s.t \quad & W^T W = I \\ & W \in \mathbb{R}^{d \times q} \\ & A \in \mathbb{R}^{d \times d} \\ & \gamma_{i,j} \in \mathbb{R} \end{aligned} \tag{13}$$

Instead of solving the function itself, it could be mostly easily done with the following equation.

$$\text{cost} = \text{HSIC}(XW, U) - \lambda\, \text{HSIC}(XW, Y)$$

The simplest way is to write a HSIC function, and pass $XW, U,$ and Y to compute the final cost. Although easy, this approach is not the fastest in terms of separating out the portion of the code that requires constant update, and the portion that remains constant. In this section, a faster approach to implement the cost function is outlined.

Starting with the original cost function :

$$\text{cost} = \text{HSIC}(XW, U) - \lambda\, \text{HSIC}(XW, Y)$$

Convert it into trace format.

$$\text{cost} = \text{Tr}\big(\tilde{K} H U U^T H\big) - \lambda\, \text{Tr}\big(\tilde{K} H Y Y^T H\big)$$

Where $\tilde{K}$ is the normalized kernel of XW, which could also be written as $\tilde{K} = D^{-\frac{1}{2}} K_{XW} D^{-\frac{1}{2}}$. Putting this into the cost function.

$$\text{cost} = \text{Tr}\Big(D^{-\frac{1}{2}} K_{XW} D^{-\frac{1}{2}} H U U^T H\Big) - \lambda\, \text{Tr}\Big(D^{-\frac{1}{2}} K_{XW} D^{-\frac{1}{2}} H Y Y^T H\Big)$$

When optimizing U, it is obvious that the 2nd portion does not effect the optimization. Therefore, U can be solved using the following form.

$$U = \underset{U}{\text{argmin}}\ \text{Tr}\big(U^T H D^{-1/2} K D^{-1/2} H U\big)$$

The situation get a bit more complicated if we are optimization for W. Using the combination of the rotation property and the combination of the 2 traces, the cost can be written as :

$$\text{cost} = \text{Tr}\big(\big[D^{-1/2} H (U U^T - \lambda Y Y^T) H D^{-1/2}\big] K\big)$$

In this form, it can be seen that the update of W matrix will only affect the kernel K and the degree matrix D. Therefore, it makes sens to treat the middle portion as a constant which we refer as Ψ.

$$\text{cost} = \text{Tr}\big(\big[D^{-1/2}\, \Psi D^{-1/2}\big] K\big)$$

Given that $\big[D^{-1/2}\, \Psi D^{-1/2}\big]$ is a symmetric matrix, from this form, we can convert the trace into an element wise product $\odot$.

$$\text{cost} = \big[D^{-1/2}\, \Psi D^{-1/2}\big] \odot K$$

To further reduction the amount of operation, we let d be a vector of the diagonal elements of $D^{-1/2}$, hence $d = \text{diag}\big(D^{-1/2}\big)$, this equality hold.

$$D^{-1/2}\, \Psi D^{-1/2} = [d d^T] \odot \Psi$$

Therefore, the final cost function can be written in its simplest form as :

$$\text{cost} = \Gamma = \Psi \odot [dd^T] \odot K$$

During update, as W update during each iteration, the matrix Ψ stays as a constant while dd^T and K update. The benefit of this form minimize the complexity of the equation, while simplify cost into easily parallelizable matrix multiplications. The equation also clearly separates the elements into portions that require an update and portions that does not.

6 Implementation Details of the Derivative

As it was shown from previous sections, the gradient of our cost function using the Gaussian Kernel has the following form.

$$\nabla f(W) = \left[\frac{1}{\sigma^2} \sum \gamma_{i,j} K_{i,j} A_{i,j} \right] W$$

It is often shown as :

$$\nabla f(W) = \Phi W$$

The key is therefore to find Φ.

$$\Phi = \frac{1}{\sigma^2} \sum \gamma_{i,j} K_{i,j} A_{i,j}$$

If we note that $A_{i,j} = (x_i - x_j)(x_i - x_j)^T$. It can be seen that the inner portion is identical to the cost function. The difference is the addition of the $A_{i,j}$ matrix and a constant of $\frac{1}{\sigma^2}$. These extra factors can be incorporate in the following form.

$$\Phi = \frac{1}{\sigma^2} Q^T \operatorname{diag}(\text{Vec}(\Gamma)) Q$$

Where :

$$Q = (X \otimes \mathbf{1}_n) - (\mathbf{1}_n \otimes X)$$

Note that $\otimes$ is a tensor product and $\mathbf{1}_n$ is a 1 vector with a length of n.

And :

$$\text{cost} = \Gamma = \Psi \odot [dd^T] \odot K$$

Since Q is a constant that never changes during the optimization, it could be calculated at the beginning and cached. During each W update using the gradient, Ψ and Q are considered as constants while K and D require a constant update. However, each time the U matrix is updated, Ψ must also be updated.

Here we outline the Algorithm for the W optimization update scheme.

1. Initialize $W_0 = 0$ for the first time, and W_k if U has been updated.
2. Calculate Q, Ψ and store them as constants
3. Calculate K, D, Φ
4. $W_{k+1} = \overrightarrow{\text{eig}}_{\min}(\Phi)$, pick q least dominant eigenvectors as W_{k+1}.
5. Repeat 3,4 until W convergence