Study of Conjugated Gradient in WDA-SMACOF and Fixed Points

1. Introduction

DACIDR [1] is a sequence clustering and visualization pipeline that uses WDA-SMACOF [2] as its core MDS method. WDA-SMACOF is a generalized solution to the DA-SMACOF algorithm with weighting, and it can be applied with various situations [3]. Traditional solution of SMACOF that solves non-linear MDS problems needed a matrix inversion of an order N matrix. Since matrix inversion has a time complexity of O(N3), it is not capable of handling large scale dataset. However, the convergence of SMACOF itself is within quadratic time. Therefore, one can use Conjugate Gradient method to approximate the matrix inversion in order to avoid the overwhelming time cost.

The STRESS function of WDA-SMACOF is given as:

$σ\left(X\_{T}\right)=\sum\_{i<j\leq N}^{}w\_{ij}(d\_{ij}(X\_{T})-\tilde{δ}\_{ij})^{2}$ ( 1 )

1. Algorithm Description

The detail of WDA-SMACOF can be found under [1]. Additionally, WDA-SMACOF needs to be updated to on some special cases. For some scientific applications, one needs to fix part of the data in one stage, and then varies other data in the following stage. The example is given as in Figure 1. Assume X is the whole dataset, and X1 is calculated in the previous stage, so nothing needs to be varied in here. And X2 needs to be varied according to the result from X1. So an updated WDA-SMACOF is needed. And it is referred to as Fixed WDA-SMACOF.

By expanding the object equation (1) with the temperature T, the function now is:

$σ\left(X\_{T}\right)=\sum\_{i<j\leq N}^{}w\_{ij}\tilde{δ}\_{ij}^{2}+ \sum\_{i<j\leq N}^{}w\_{ij}d\_{ij}^{2}\left(X\_{T}\right)-2 \sum\_{i<j\leq N}^{}w\_{ij}\tilde{δ}\_{ij}d\_{ij}(X\_{T})^{}$ ( 2 )

$=η\_{\tilde{δ}}^{2}+η^{2}\left(X\_{T}\right)-2ρ(X\_{T})$ ( 3 )

where the $\tilde{δ}\_{ij}$ is the distance in original dimension with termperature T, and XT is the N×L matrix where each row represents the coordinates under temperature T. For simplification of the equations, the X is used to denote XT in following notations within this section. Note that part of coordinates in X is fixed in this algorithm, so denote the fixed part in X as X1, and varied part of points in X as X2, where their corresponding number is N1 and N2. So the X can be divided as shown in. Note that $η\_{\tilde{δ}}^{2}$ is a constant, so only the second and third term of equation (3) needs to be modified.

|  |  |  |
| --- | --- | --- |
| Figure 1： Graph representation of dividing X into X1 and X2  | Figure 2：Graph representation of dividing V into 4 parts  | Figure 3：Graph representation of dividing B(Z) into 4 parts  |

Note that the second term means the square sum of distances between each pair of points in the target dimension under temperature T, so these distances are composed of 3 parts, the distances from points in X1 to X1, from points in X2 to X2, and from points in X1 to X2. And the second term $η^{2}\left(X\right)$ can be written as:

$η^{2}\left(X\right)=trX'VX$ ( 4 )

Where V is given in equation (5) and tr means the trace of a matrix.

$v\_{ij}=\left\{\begin{array}{c}-w\_{ij} if i \ne j\\-\sum\_{k\ne i}^{}v\_{ik} else if i=j \end{array} \right.$ 

Consider that X is composed of X1 and X2, and V can be decomposed in the same way as in Figure 2, equation (4) can be then decomposed into

$tr X^{'}VX=trX\_{1}'V\_{11}X\_{1}+trX\_{2}'V\_{22}X\_{2}+trX\_{1}'V\_{12}X\_{2}+trX\_{2}'V\_{21}X\_{1}$ ( 6 )

The matrices V11, V22, V12 and V21 are decomposed from the matrix V and shown in Figure 2. Note that all the diagonal values in V11 and V22 are as same as in V, so the first and second terms of equation (28) does not equal to d(X1) and d(X2). It also includes the values from distances between X1 and X2.

The third term of equation (3) is the sum of distances in between the mapping of X in the target dimension space multiplied by the distances in the original dimension space under temperature T. This term can be applied with *Cauchy-Schwarz inequality*, whereby an upper bound can be found as the following:

$-2ρ\left(X\right)=-2tr X^{'}B\left(X\right)X\leq -2tr XB\left(Z\right)Z$  ( 7 )

Where function B(X) and B(Z) can be calculated using equation (8).

$b\_{ij}=\left\{\begin{array}{c}\begin{array}{c}-\frac{w\_{ij}\tilde{δ}\_{ij}}{d\_{ij}\left(X\_{T}\right)} if i \ne j and d\_{ij}\left(X\_{T}\right)\ne 0\\-\sum\_{k\ne i}^{}b\_{ik} else if i=j \end{array}\\0 otherwise\end{array}\right.$ 

Note that Z is given as the previously estimated X, and in this case, X1 remains fixed during iterations. So Z can be decomposed as X1 and Z2, where Z2 means the previously estimated X2. So in this equation, the part for X1 actually equals to each other on the left and right of the equation. Therefore, denote B as for the matrix result from function B(Z), matrix B can be decomposed as same as in Figure 3. And left hand side of equation (7) is now:

$tr X^{'}B\left(X\right)X=trX\_{1}'B\_{11}X\_{1}+trX\_{2}'B\_{22}X\_{2}+trX\_{1}'B\_{12}X\_{2}+trX\_{2}'B\_{21}X\_{1}$( 9 )

Since the first term is fixed, only the rest of term in equation (7) needs to be updated with this inequality, which is now:

$-2tr X^{'}B\left(X\right)X\leq -2\left(trX\_{1}'B\_{11}X\_{1}+trX\_{2}'B\_{22}Z\_{2}+trX\_{1}'B\_{12}Z\_{2}+trZ\_{2}'B\_{21}X\_{1}\right)$  ( 10 )

Therefore, by taking equation (6) and (10) into equation (3), the STRESS function can now be written as:

$σ\left(X\_{T}\right)=η\_{\tilde{δ}}^{2}+trX\_{1}'V\_{11}X\_{1}+trX\_{2}'V\_{22}X\_{2}+trX\_{1}'V\_{12}X\_{2}+trX\_{2}'V\_{21}X\_{1}-2\left(trX\_{1}'B\_{11}X\_{1}+trX\_{2}'B\_{22}X\_{2}+trX\_{1}'B\_{12}X\_{2}+trX\_{2}'B\_{21}X\_{1}\right)$ ( 11 )

And the upper bound it is given as

$σ\left(X\_{T}\right)\leq η\_{\tilde{δ}}^{2}+trX\_{1}'V\_{11}X\_{1}+trX\_{2}'V\_{22}X\_{2}+trX\_{1}'V\_{12}X\_{2}+trX\_{2}'V\_{21}X\_{1}-2\left(trX\_{1}'B\_{11}X\_{1}+trX\_{2}'B\_{22}Z\_{2}+trX\_{1}'B\_{12}Z\_{2}+trZ\_{2}'B\_{21}X\_{1}\right)=τ(X\_{2},Z\_{2}) $ ( 12 )

Since X2 is the only varied part of this equation, by taking the partial derivative of equation (12), now it is:

$\frac{∂τ\left(X\_{2},Z\_{2}\right)}{∂X\_{2}}=(X\_{1}'V\_{12})'+V\_{21}X\_{1}+2V\_{22}X\_{2}-2B\_{22}Z\_{2}-B\_{21}X\_{1}-(X\_{1}'B\_{12})'$ ( 13 )

Since both V and B are symmetric matrices, so equation (34) now becomes

$\frac{∂τ\left(X\_{2},Z\_{2}\right)}{∂X\_{2}}=2V\_{21}X\_{1}+2V\_{22}X\_{2}-2B\_{22}Z\_{2}-2B\_{21}X\_{1}$ ( 14 )

By setting (35) to zero, the final majorization function is:

$V\_{22}X\_{2}=B\_{22}Z\_{2}-B\_{21}X\_{1}-V\_{21}X\_{1}$ 

Algorithm 1 Fixed WDA-SMACOF algorithm

Input: $∆$, W, $X$1, X2, $ε$ and $α$

Output: X as target dimension mapping

1. Generate random initial mapping X2.
2. Compute V21X1
3. $k=0$;
4. while $T\_{k}$ ≥ $T\_{min}$ do
5. Compute $T\_{k}$ and $\tilde{∆}\_{k}$
6. $t=0$;
7. while $σ(X^{|t|})-σ(X^{\left|t+1\right|})>ε$
8. Use CG to solve (15).
9. $t=t+1$;
10. end while
11. Cool down computational temperature $T\_{k+1} = α T\_{k}$;
12. $k=k+1$
13. end while
14. $X=$output of SMACOF based on $T\_{k}$
15. return $X$

Recall the Z2 is the estimated X2 in previous iteration, this is now the final formula for fixed WDA-SMACOF. Note that V21X1 is fixed from the start of the algorithm, so it only needs to be computed once and stored in static memory. Instead of having order N×N×L matrix multiplication in each iteration, now the calculation in each SMACOF iteration becomes N2×N2×L and N2×N1×L. The CG calculation now is in order N2 instead of N as well. Note that the V22 matrix is naturally a SPD matrix, so no approximation needs to be done beforehand for this algorithm. The detail of Fixed-WDA-SMACOF is given in algorithm 1.

1. Experimental Analysis

3.1 CG Iterations per SMACOF Iteration

When analyzing the time cost of WDA-SMACOF, Conjugate Gradient is the most time consuming part of WDA-SMACOF algorithm if there are multiple CG iteration per SMACOF iteration. The CG computation for WDA-SMACOF is essential as this is the main contribution for this algorithm that enables quadratic time complexity. In this set of experiments, the CG computation versus matrix inversion is tested, and the number of CG iterations needed per SMACOF iteration is tested. The time cost detail of WDA-SMACOF is also studied.

The number of CG iterations needed per SMACOF iteration in WDA-SMACOF is a key factor that dominates the total computation time for WDA-SMACOF. If the weights all equal 1, by increasing the data size alone, the CG iterations needed per SMACOF iteration is around 2 despite of the increasing data size. If the data size is fixed, and randomly chosen part of the distances as missing values, the number of CG iterations needed per SMACOF iteration is shown in Figure 4. It shows that if the weights are uniformly selected as missing, the CG iterations per SMACOF iteration won’t increases that rapidly. Even when the percentage of missing distances increases to 50%, the average CG iteration per SMACOF iteration is around 4, which is much less than the total size of data as 100k tested here. Figure 5 shows that by taking the average of all situations with percentage of missing distances increasing from 0.1 to 0.5, the number of CG iterations only increases from 3.35 to 3.74 with the data size increases from 20k to 100k. This is because the convergence rate of CG really depends on the eigenvalues found for the matrix on the left hand side of the formula Ax=b. If A has an evenly distributed eigenvalue, which means the direction found in the first CG iteration will make the cost function reaches a point where it is near the final destination. However, if the eigenvalues for A has property that the largest eigenvalue is much larger than the smallest eigenvalue, the number of iterations will be much larger. And during the study of this dissertation, this condition is very unlikely to happen by testing several of scenarios occurred with the dissimilarities generated using sequence alignment.

Sammon's Mapping is a popular non-linear MDS algorithm that solves equation (2) instead of equation (1). WDA-SMACOF can also solve this type of problem by leveraging the power of weight function. As shown in equation (2), SSTRESS can be converting to equation (1) with an updated weight, and the weight can be given as wij =1/δij. The weight matrix can then be calculated beforehand, and the eigenvectors of the given matrix V will be a lot different from the weight with only zeroes and ones. The time cost of WDA-SMACOF solving Sammon's STRESS will be higher than solving a randomly generated weight matrix because of this. And the number of CG iterations per SMACOF iteration is given by taking the average of total number of CG iterations divided by total number of SMACOF iteration. The result shows that despite of various dataset, the CG iterations needed per SMACOF iteration are around 200. Figure 6 illustrates the average number of CG iterations per SMACOF iteration needed for Sammon's Mapping, and it shows that for COG Consensus data the number of iterations needed are the smallest, which is around 110. The number of iterations for hmp16SrRNA data is the most among all three dataset, which is around 200. And for 2000 Artificial RNA data, the number of iterations is also around 170, which is only 11.3% less than processing the hmp16SrRNA data. But the data size of Artificial RNA data is only 1/5 of the 10k hmp16SrRNA data. This is because the number of CG iterations are mainly dominated by the eigenvectors of matrix V, rather than the data size. The following runs were finished using 600 cores on Futuregrid xRay. Figure 7 shows the average number of CG iterations per SMACOF iteration for WDA-SMACOF when using two different weightings. When use the weights all equals 1, the number of CG iterations per SMACOF iteration is always around 2 or 3. And for Sammon's Mapping, the number of CG iterations increases to around 200. However, even by increasing the data size 5 times with a same dataset, the number of CG iterations needed per SMACOF iteration only increases 1.4 times. One can assume that by increasing the data size won't increase computation time much because of this even with larger dataset as long as the dataset has a similar visualization data structure with Sammon's Mapping. Figure 8 shows the time cost of WDA-SMACOF with two different weighting processing the same dataset, and the number of SMACOF iterations were set to 100. And the time of WDA-SMACOF increases because of the number of CG iterations. It shows that WDA-SMACOF takes around 21 more time to finish Sammon's Mapping than dealing with the case where all weights equal 1 on 20k data. And it takes around 27 times longer when data size increases to 40k and 60k. Finally, in 80k and 100k cases, it takes 32 times longer to finish. So when processing with Sammon's mapping, WDA-SMACOF takes around 20 to 30 times longer to finish than dealing with trivial weights matrices.

Figure The number of CG iteration needed for 100k AM Fungal Data processed with parallel WDA-SMACOF. The percentage of missing distances increases from 0 to 0.5 and all missing distances are randomly chosen.

Figure The number of CG iteration needed for AM Fungal Data processed with parallel WDA-SMACOF. The data size varies from 20k to 100k. The number of iterations took over the average of number of CG iterations from the scenarios that percentage of missing distances increases from 0 to 0.5.

Figure The number of CG iteration needed for 2k Artificial RNA data, 4872 CGO Protein data and 10k hmp16S rRNA data processed with parallel WDA-SMACOF with Sammon's Mapping.

Figure The number of CG iteration needed for AM Fungal data processed with parallel WDA-SMACOF with Sammon's Mapping. The data size increases from 20k to 100k.

Figure Time Cost of WDA-SMACOF with equal weights compared with WDA-SMACOF with Sammon's Mapping using AM Fungal data. The data size increases from 20k to 100k. Each run takes 100 SMACOF iterations.

3.2 Accuracy and Time Cost of Fixed WDA-SMACOF

The Fixed WDA-SMACOF is compared with a normal interpolation technique called MI-MDS in order to check its time cost and accuracy. The time cost is calculated without the distance computation for the interpolation technique, i.e. MI-MDS, since distance computation is not included within the time cost of WDA-SMACOF. The accuracy is evaluated using normalized STRESS value for all of the data including in-sample and out-of-sample. The in-sample data were processed using WDA-SMACOF in order to get the most accurate result, then the rest out-of-sample data were either interpolated to the target dimension or being processed using fixed WDA-SMACOF. The number of nearest neighbor that used in MI-MDS is always the in-sample data size.

This experiment is carried out on the 4640 Artificial RNA data, where in-sample data size increased from 500 to 4000, and the out-of-sample data size decreases from 4140 to 640. The normalized STRESS value is given in Figure 9 and time cost is given in Figure 10. As shown in Figure 9, the normalized STRESS value of MI-MDS and WDA-SMACOF are both decreasing when in-sample size increases. This is because with the increasing in-sample size, the part of data that needs to be processed decreases. And the full WDA-SMACOF on the original data obviously has a lower STRESS value compared to the fixed WDA-SMACOF or interpolation techniques. On the other hand, the WDA-SMACOF is more accurate than MI-MDS because it also considered about the distances within the out-of-sample data. The difference between their normalized STRESS decreases when the in-sample size increases as well. And when the in-sample size reaches 4k, their differences is only 0.00002, which is different from the result from in-sample size 500, where their differences is 0.0007. This means the fixed WDA-SMACOF has a much larger advantage while processing data with a small in-sample size. Figure 10 illustrate the time cost of both the methods, and WDA-SMACOF is much slower than the MI-MDS method. This is because WDA-SMACOF has a quadratic time complexity while MI-MDS is linear. The time cost of WDA-SMACOF decreases with the out-of-sample data size, and the time cost of MI-MDS reaches maximum as in-sample size 2000. This is because the time cost of MI-MDS is O(N1×N2) where N1 is the in-sample size and N2 is the out-of-sample size.

Figure The normalized STRESS value comparison of WDA-SMACOF and MI-MDS. The in-sample data coordinates are fixed, and rest out-of-sample data coordinates can be varied.

Figure The time cost comparison of WDA-SMACOF and MI-MDS. The in-sample data coordinates are fixed, and rest out-of-sample data coordinates can be varied.

1. Conclusion and Futurework

In this section, WDA-SMACOF and Fixed WDA-SMACOF are proposed. WDA-SMACOF is an algorithm that can process input distance matrix with different weight associate with each distance, so that the particular needs from dissimilarities generated from pairwise sequence alignment can be met. During the performance analysis, WDA-SMACOF has been proved to have a lower time cost compare to the traditional matrix inversion solution. Fixed WDA-SMACOF is an extension to WDA-SMACOF, so that during the dimension reduction process, the coordinates of a certain part of data points can remain fixed. This is usually for solving problems associate with reference sequences and new sequences. The experiments showed that Fixed WDA-SMACOF has a higher accuracy than the interpolation solution by sacrificing computation time. This solution can be utilized for future work once a precise dimension reduction result needs to be obtained with fixed coordinates on a certain set of sequences in target dimension.

Although the time complexity of WDA-SMACOF is O(N2), it is around 30 times longer processing non-trivial weights (Sammon's Mapping) than processing trivial weights (0s and 1s). Future study of the convergence threshold for CG could significantly impact the total time cost of WDA-SMACOF. One may use the target dimension mapping from previous SMACOF iteration to set as starting position for CG in the next SMACOF iteration. This could potentially reduce the number of CG iterations needed per SMACOF iteration. And more experiments with Fixed WDA-SMACOF can be carried out so that it’s behavior on different kinds of dataset can be evaluated.

[1] Ruan, Yang, Saliya Ekanayake, Mina Rho, Haixu Tang, Seung-Hee Bae, Judy Qiu, and Geoffrey Fox. "DACIDR: deterministic annealed clustering with interpolative dimension reduction using a large collection of 16S rRNA sequences." In *Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine*, pp. 329-336. ACM, 2012.

[2] Ruan, Yang, and Geoffrey Fox. "A Robust and Scalable Solution for Interpolative Multidimensional Scaling with Weighting." In *eScience (eScience), 2013 IEEE 9th International Conference on*, pp. 61-69. IEEE, 2013.

[3] Ruan, Yang, Geoffrey House, Saliya Ekanayake, Ursel Schütte, James D. Bever, Haixu Tang, and Geoffrey Fox. "Integration of Clustering and Multidimensional Scaling to Determine Phylogenetic Trees as Spherical Phylograms Visualized in 3 Dimensions." *Proceedings of C4Bio* (2014): 26-29.