A Projection free method for Generalized Eigenvalue Problem with a nonsmooth Regularizer (Supplemental Material)

Seong Jae Hwang[a] Maxwell D. Collins[a] Sathya N. Ravi[b] Vamsi K. Ithapu[a]
Nagesh Adluru[e] Sterling C. Johnson[d] Vikas Singh[ca]

[a]Dept. of Computer Sciences, University of Wisconsin - Madison, Madison, WI
[b]Dept. of Industrial and Systems Engineering, University of Wisconsin - Madison, Madison, WI
[c]Dept. of Biostatistics and Med. Informatics, University of Wisconsin - Madison, Madison, WI
[d]William S. Middleton VA Hospital, Madison, WI
[e]Waisman Center, Madison, WI

First, we provide some experimental results that are mentioned (but not shown) in the main paper; second, we also show the performance of our algorithm against a commercially available standard numerical optimization (nonlinear optimization) solver both in terms of accuracy and runtime in that order.

1. Accuracy with different r and p

Table (1) shows the classification results whereas table (2) shows regression results. In either case, the rows correspond to PCA rank (p). The columns represent the GEP and R-GEP models (with different choices of tensor decomposition rank r). Note that we omit the baseline and the standard PCA results in both the tables since they are redundant. It is clear from the accuracy results that introducing additional sources of information always increases the performance which our algorithm enables us to do so.

	GEP						R-GEP					
p	$r=1$	3	5	10	15	20	$r=1$	3	5	10	15	20
3	86.6	85.7	85.7	85.7	86.6	87.5	**90.3**	87.6	88.5	86.5	86.4	85.4
5	85.4	85.4	85.4	86.3	85.5	86.3	**89.5**	87.6	89.3	87.3	87.4	86.4
7	84.3	84.3	84.3	84.3	86.1	86.4	86.4	88.4	**88.4**	87.5	88.3	88.3
10	82.4	85.3	84.2	86.2	85.1	85.1	86.5	87.5	86.4	**89.3**	86.4	87.5
13	86.2	87.1	84.2	88.1	85.4	84.1	89.2	91.2	**91.2**	88.2	86.3	87.2
20	85.1	86.1	87.0	86.1	86.1	86.1	88.3	90.2	**90.8**	88.3	89.0	86.5

Table 1: Healthy versus diseased classification accuracy (10-fold cross validated) using GEP and R-GEP. p denotes the PCA rank and r is tensor rank.

2. Diminishing returns of $|K'|$

In this section we show that adding more regularization terms in the objective does not affect the accuracy of the model significantly as mentioned in section 3 of the main paper. This phenomenon is expected because in the PCA setting we assume that the principal component or the eigenvector corresponding to the leading eigenvalue is dominant, that is, most variance of the data can be explained well by the largest few eigenvectors. In the particular dataset (described in section 5 of the main paper) that we are interested in, it suffices to just include *the* largest eigenvector.

	GEP						R-GEP					
p	$r=1$	3	5	10	15	20	$r=1$	3	5	10	15	20
3	0.719	0.718	0.718	0.718	0.726	0.718	0.745	0.755	**0.771**	0.758	0.750	0.733
5	0.726	0.734	0.737	0.735	0.727	0.735	**0.769**	0.760	0.746	0.749	0.739	0.749
7	0.707	0.707	0.707	0.713	0.712	0.736	0.763	0.781	**0.785**	0.734	0.714	0.738
10	0.656	0.702	0.742	0.719	0.700	0.727	0.741	0.742	**0.762**	0.754	0.748	0.737
13	0.730	0.762	**0.765**	0.745	0.746	0.743	0.737	0.744	0.757	0.754	0.758	0.757
20	0.643	0.679	0.644	0.648	0.641	0.628	0.656	0.686	**0.705**	0.665	0.661	0.657

Table 2: Healthy versus diseased regression correlation coefficient (10-fold cross validated) using GEP and R-GEP. p denotes the PCA rank and r is tensor rank.

m	$\beta = 0.0$	0.2	0.5	0.8	1.0
1	87.12	87.12	88.12	88.12	88.12
2	87.12	87.12	87.12	87.12	87.12
3	87.12	87.12	87.12	88.12	88.12

Table 3: Average accuracy (10-fold cross validated) using m regularization terms in addition to the base regularization term for $p = 10$ and $r = 5$. The base regularization term uses the largest eigenvector, and the m additional regularization terms are formulated using the next largest eigenvectors in a similar way. β is the ratio of the regularizer parameter to λ such that the parameters for the additional regularization terms are $\beta\lambda$. So when $\beta = 0.0$, the model is the same as having a single regularization term. Notice the small differences even with the additional regularization terms. The difference in accuracy between the result for $p = 10$ and $r = 5$ in Table 1 and this table is due to the difference in starting points.

3. Scalability

The following table shows the scalability of our algorithm.

N	100	500	1000	3000	5000	7000	10000	12000	15000	20000
runtime (sec)	3.55	1.41	3.01	35.11	78.49	165.23	365.20	779.61	857.11	1640.57

Table 4: Average runtime (10-fold cross validated) to achieve error $< 5\%$ for different problem sizes N with $\kappa = 3$ and $p = 20$.

4. KNITRO versus our algorithm

We used a commercially available solver KNITRO to test the performance including the run time and the accuracy. We used YALMIP [1] to model our problem with a few lines of MATLAB code. The results are shown in the table below. We initialized them both in the same way. Our algorithm always outperforms KNITRO in terms of the run time and also note that even though the objective value is equal or close, we can see that the accuracy of our algorithm is always better. Also, compared to KNITRO, the runtime of our algorithm is ~ 4 times faster for $p = 3$ and ~ 7 times faster for $p = 5$.

	$p=3,\ r=5$		$p=3,\ r=15$		$p=5,\ r=5$		$p=5,\ r=15$	
	KNITRO	R-GEP	KNITRO	R-GEP	KNITRO	R-GEP	KNITRO	R-GEP
acc	72.73	**83.48**	84.79	**86.32**	75.74	**88.32**	84.10	**87.53**
objval	-16394	**-16766**	**-24344**	-24330	-73208	**-73646**	**-24508**	-24496
epoch	8553	**5882**	10000	**5882**	6297	**5882**	10000	**5882**
runtime (sec)	1030.1	**276.7**	1312.4	**254.9**	1603.1	**221.2**	3060.8	**426.0**

Table 5: Comparison between KNITRO and R-GEP.

5. FOptM versus our algorithm

We also compared our algorithm with the algorithm presented in [2]. Recall that their algorithm requires that $D \succ 0$ and $\lambda = 0$ i.e., no regularization terms. In our case our $D \succeq 0$, so we used $D' := D + \epsilon I$ where I is the identity matrix of appropriate size for experimental purposes. We reformulate the GEP with D' to a standard EP (Eigenvalue Problem) by the following procedure. Let $D' = U^T \xi U$ be the eigen decomposition of D' with U be the matrix with eigenvectors and ξ be the diagonal matrix with positive eigenvalues on the diagonal. Setting $Z = \sqrt{\xi} UV$ we obtain the standard EP. The following reason explains the significant difference in the accuracy. In principle one has to choose D' to be the closest positive definite matrix to D which might be a hard problem on its own. Even in this simple technique we need to make sure that ϵ is as small as possible since a large ϵ might change the optimal solution and in principle there is not a best way to do this. The accuracy of our algorithm ($\lambda = 0$) when $p = 3, r = 5$ was 84.51% whereas the accuracy of FOptM was 83.90%. FOptM was faster than our algorithm but it is a well established fact that computing the eigen decomposition and inverses are expensive and unstable when the problem size increases making the reformulation impractical for large data instances.

References

[1] J. Lfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004. 2

[2] Z. Wen and W. Yin. A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 142(1-2):397–434, 2013. 3