Response to Reviewer 1

February 11, 2017

We appreciate the helpful feedback from the reviewer. We have addressed your questions and comments. Below we give a point-by-point response to each of the questions:

Overall comment

Although the idea of a using a larger set of regularization parameters is interesting, the empirical results are incomplete and including additional scenarios would help strengthen the paper

We apologize for omitting simulation details. We have updated the paper accordingly.

We have also included a new example of matrix completion to illustrate the wide applicability of our method. This example moves away from the simple regression framework and considers matrix-valued data with partially observed entries. The problem now involves minimizing a penalized loss function with a nuclear norm penalty. This joint optimization problem has a much more complex differentiable space compared to the other examples. We had to rely on different representations of this differentiable space in order to (1) prove that the conditions of Theorem 1 were satisfied and (2) calculate the gradient. We added three new sections: Section 2.4.4 introduces low-rank matrix completion and illustrates how to transform the joint optimization problem into an equivalent smooth joint optimization problem; Section 3.4 provides simulation results; Section A.3.4 in the Appendix provides more details on how to calculate the gradient and shows the conditions in Theorem 1 are satisfied.

Specific Suggestions/comments

1. Can the authors point to examples in the literature where a large set of regularization parameters was used?

   We have updated the introduction with more examples of problems with multiple regularization parameters. We inserted the following paragraph into Section 1:

   > In recent years, there has been much interest in combining regularization methods to produce models with multiple desired characteristics. For example, the elastic net (Zou & Hastie 2003) combines the lasso and ridge penalties; and the sparse group lasso (Simon et al. 2013) combines the group lasso and lasso penalties. In Bayesian regression, a popular method for pruning irrelevant features is to use automatic relevance determination, which associates each feature with a separate regularization parameter (Neal 1996). Finally, neural networks commonly use regularization to control the weights at each node. Snoek et al. (2012) showed that using separate regularization parameters for each layer in a neural network can improve performance. From a theoretical viewpoint, multiple regularization parameters are required in certain cases to achieve oracle convergence rates. van de Geer & Muro (2014) showed that when fitting additive models with varying levels of smoothness, the penalty parameter should be higher for more "wiggly" functions and vice versa.

2. First line on p. 13, "the optimal regularization parameters $\boldsymbol{\lambda} = (\lambda_1, \lambda_2)^\top$" should be "the optimal regularization parameters $\boldsymbol{\lambda} = (\lambda_0, \lambda_1, ..., \lambda_M)^\top$"

Thank you for pointing this out. We have corrected this typo in the paper.

3. The authors note that the models considered in Sections 2.4.2 and 2.4.3 are usually employed with only two regularization parameters but propose using a larger set of regularization parameters. In the corresponding simulations in Sections 3.2 and 3.3, they compare using a larger set of regularization parameters against using two regularization parameters selected using grid search. The grid spaces considered are fairly small, so its not clear if the improved performance for gradient descent is due to the additional regularization parameters or due to the dependence of performance on the grid space. To provide additional insight into whats causing the difference in performance, could the authors also present results using gradient descent, Nelder-mead, and Spearmint for the two parameter case?

   We have added the results from gradient descent, Nelder-mead, and Spearmint for the two-parameter version of the joint optimizations for all the examples. The results are all displayed in Table A.1 in the Section A.6 of the Appendix. We include the text from Section A.6 for convenience:

   > The simulation results in Section 3 show that joint optimization problems with many penalty parameters can produce better models than those with only two penalty parameters. One may wonder if this difference is due to the method used to tune the penalty parameters. Here we present results from tuning the two-penalty-parameter joint optimization problems from Sections 3.2, 3.3, and 3.4 using gradient descent, Nelder-Mead, and Spearmint. As shown in Table A.1, the performance of these methods are very similar to grid search. Regardless of the method used to tune the two-penalty parameter joint optimization, the resulting models all have higher validation and test error compared to the models from the joint optimization problem with many penalty parameters tuned by gradient descent.

4. The authors report average performance and standard errors for the simulations done in Section 3. How many simulation runs were used in each example?

   We had thirty simulation runs for the examples in Section 3. We added this detail to Section 3.

5. In Section 3, two different starting values were considered for Nelder-mead and gradient descent. How sensitive were the results to the choice of starting values?

   The results for Nelder-Mead and gradient descent are indeed sensitive to their starting values, but gradient descent performs better than Nelder-Mead on average. We discuss the sensitivity of the two methods in more detail in Section A.5 of the Appendix. We tested multiple initialization points for the two methods on a smaller version of the sparse additive model. We plot the validation error as the number of initialization points increases. The validation error of both methods plateau quickly around four initialization points. Gradient descent manages to find penalty parameters with lower validation error. Also, given a random initialization, gradient descent tends to find penalty parameters with lower validation error compared to Nelder-Mead.

6. The empirical results for gradient descent depend on α, β, and δ. The authors mention ranges considered for these parameters in Section 2.5. How were these parameters ultimately selected in the evaluations in Sections 3 and 4?

   We apologize for omitting the exact values of the gradient descent procedure. We have now specified their values in Section 2.5. In particular, we use $\alpha = 0.001$, $\beta = 0.1$, and $\delta = 0.0005$.

7. In Sections 3.2 and 3.3, the authors created a training, validation, and test set, but in Section 3.1 they only consider a training and validation set. Why was a test set not considered in Section 3.1?

   We apologize for the confusion. We had originally omitted the test error for Section 3.1 since the goal was to illustrate that the validation loss values were similar between gradient descent and grid search. However we recognize that this omission may be confusing to the reader. To streamline the paper, we have included a new test error column to Table 1 in Section 3.1.

8. Table 1 should note that standard errors are given in parentheses.

   We have now included this clarification to Table 1.

9. For the simulations in Section 3.3, why did the authors set n = 60 in the first case and n = 90 in the other two cases?

   The first case originally had $n = 60$ since it had fewer model parameters to estimate. The reviewer's question made us realize that having the same n across the three examples allows the reader to better understand the trends in the table. Therefore we have updated our simulations such that all $n = 90$ across all simulations in Section 3.3.

10. g is undefined in Table 4

    We have removed g from Table 4 altogether. g was meant to indicate the number of groups in the true model. However we already specify in Section 3.3 that there are three groups in the true model.

11. In Table 4, could the authors provide intuition for why there is a large difference in validation error and test error for gradient descent?

    We have added a new paragraph in Section 3.3 to provide some intuition for the large difference between validation and test error. We include it below:

    > The validation errors of the un-pooled sparse group lasso models fit by gradient descent were much smaller than the test errors. This difference between the validation and test errors can be attributed to a change in the bias-variance tradeoff. The un-pooled sparse group lasso has more penalty parameters and thus a larger model space. That is, its fitted models have smaller bias but larger variance. Depending on the parameter-tuning method, the improvement in bias may outweigh the additional variance. Since gradient descent could effectively minimize the validation loss, the test error was low. On the other hand, Nelder-mead and Spearmint were unable to find penalty parameter settings with small validation loss, so their test errors were high.