Genetic Matching Algorithms for Sensitivity to Hidden Bias Reduction

By Pierre Alkubeh

Introduction

One of the benefits of randomized-control studies for estimating treatment effect is it (presumably) accounts for biases caused by all confounding variables the researchers could and could not observe. However, observational studies have no inherent mechanism to account for the effect of confounders. Researchers often use matching to minimize the confounding bias of observed variables on the estimated effect; however, it does not account for bias caused by unobserved confounders.

Paul R. Rosenbaum (2005) put-forth a method for sensitivity analysis on matching estimates. The method assumes the presence of a single unobserved confounder, then measures the range of effect the unobserved, confounding variable has on the p-value at various weights to the importance of the confounder.

The first part of this paper proposes a genetic matching function that optimizes for the lowest sensitivity by creating a metric based on Rosenbaum's approach to quantify sensitivity to hidden bias and then using a genetic algorithm to optimize on that metric. It significantly improved robustness in RCT and observational datasets. The second part proposes, also using Rosenbaum's approach, a brute-force method to guess what possible confounding variables may look like by randomly generating variable and testing how well the created variables can minimize sensitivity to hidden bias. A method for determining what the random variables are in the context of any study is then proposed. When this method was applied to an RCT dataset, the method either could not find reasonable suggestion for confounders or none existed.

Part I: General Methodology

The programming language R will be used because of its widespread support and use in academia. The current most common way to perform genetic matching in R is to use the GenMatch() function in the Matching library. To allow for ease of comparison between methods, this paper will also use the GenMatch() function, and only write new fitness functions used by GenMatch to determine what to optimize for.

Part I: Sensitivity Matching a la Rosenbaum

The key to optimizing for robustness is quantifying a fitness value to represent exactly how robust the current matching setup is. Rosenbaum's approach gave two key values, the magnitude of the hidden bias represented by Γ, and the range of the estimated effect's p-value at the specific Γ. At $\Gamma = 1$, two units with the same observed covariates but different unobserved covariates have the same likelihood of receiving treatment, but at $\Gamma = 2$, one unit can be up to twice as likely to receive treatment because of the unobserved covariate.

The fitness function created focuses on lowering the upper-bound of the p-value at the lowest Γ until it becomes statistically significant (p-value <= 0.05), before attempting to lower the upper-bound p-value at the next level of Γ. There is no maximum value for Γ, so a cap of Γ=5 is suggested under the assumption that should results of a study still be statistically significant if a confounder's effect on treatment is so large that one unit is five times as likely to receive or not receive the treatment, then the study in question is robust. Incremental increase in the value of Γ is set at 0.25 for nuance without providing too-long a list of values. So, the algorithm will minimize sensitivity on a range of Γ from 1 to 5, in 0.25 increments. The function has these values set as the default but can be changed at input. This is not lexical optimization; it is a minimization optimization using fitness score created based on the levels of Γ specified. In the default Γ range of 1 to 5 in 0.25 increments, there are 5 Γ intervals. So an initial score of 5 is given, and for each level Γ that is statistically significant, 1 is subtracted from the score. The upper-bound p-value of the highest interval that is not statistically significant is added to the score to incentivize the lowering of that p-value.

To determine the effectiveness of this method, the results of the suggested genetic matching algorithm optimizing for robustness will be compared to the algorithm optimizing for p-values. The dataset of choice is the Lalonde dataset used by Dehejia & Wahba (1999). The observational version of the dataset is used, where controls were collected from PSID.

Table 1. Upper Bound p-value for Different Optimization Goals Using Lalonde

Γ	**p-value opt.**	**Robust opt.**
1	0.0001	0.0000

1.25	0.0093	0.0039
1.5	0.1050	0.0581
1.75	0.3702	0.2550
2	0.6793	0.5507
2.25	0.8791	0.7944
2.5	0.9644	0.9258

The p-value opt. column in table 1 are the p-value upper-bounds from sensitivity analysis results we obtain on the regular genetic matching algorithm that optimizes for the lexical p-value. The Robust opt. column are the results for the new genetic matching algorithm optimizing for robustness. At Γ=1, both methods are statistically significant. At Γ = 1.25, both are still significant, though the degree of significance is vastly different as robust optimization possesses an upper p-value half that of lexical p-value optimization. At Γ=1.5, while neither is statistically significant, note that the upper bound for the robust optimization is only slightly over the significance level and still *half* the p-value optimization value. Above Γ =1.5, the upper bounds are too high to even be considered, though the robust optimization maintains better results. For this dataset, genetic matching optimizing for lexical p-value is susceptible to minor hidden biases, while the robust optimization provides a model that is resistant to significant hidden bias.

As with all genetic algorithms, the process is stochastic, and so better results can be achieved at higher population size, wait generations, and other parameters that increase the scope of the search. Additionally, sometimes running the exact same algorithm with the exact same parameters can lead to different results so it is advised that the program be run multiple times until an optimal or satisfactory balance is found.

The code for the function described is included in the link at the top of the first page. The “rbounds” library is necessary for the code to work.

Part I: Unexpected Finding

Interestingly, when this method was applied to the original Lalonde dataset collected from a randomized control study, it provided in significant improvement to the RCT’s p-values.

Table 2. Upper Bound p-value for Different Optimization Goals Using Lalonde

Γ	**p-value opt.**	**Robust opt.**
1	0.0027	0.0003
1.25	0.0967	0.0282
1.5	0.4600	0.2393

1.75	0.8194	0.6179
2	0.9370	0.8800
2.25	0.9951	0.9745
2.5	0.9995	0.9960

At Γ=1, both methods are statistically significant. At Γ = 1.25, the p-value optimization is no longer significant, but the robust optimization remains. At Γ=1.5, while neither is statistically significant, note that the upper bound for the robust optimization is *half* the p-value optimization. Above Γ =1.5, the upper bounds are too high to even be considered, though the robust optimization maintains better results. For this dataset, genetic matching optimizing for lexical p-value is susceptible to any minor hidden biases, while the robust optimization provides a model that is resistant to minor hidden biases, and twice as resistant to more significant hidden bias.

It is normally expected that an RCT is more robust than an observational study because the presence of an unobserved confounder is unlikely, regardless of analysis results. However, sensitivity analysis is just as necessary in randomized settings because the law of large numbers only nullifies the biases of normally distributed variables, and the assumption that all variables are distributed as such is not always true. Wealth or income distribution is a common variable in economics that is not normally distributed, it is bounded on the left side but unbounded and very skewed to the right. Additionally, some distributions are commonly mistaken for normal distributions but are not. For example, fat-tailed distributions are ones where the units at the tail of the distribution, the rare ones, have a disproportionate impact on the true mean effect, but because they are rare to observe, studies do not include their effect and so the sample effect is not representative of the true effect (Taleb, 2007). Therefore, sensitivity analysis and robustness optimization are still needed. This is neglected by sensitivity analysis methods such as those posed by Imbens (2003) because they assume present confounders are representative of unobserved confounders.

This does not mean the data used in the example was suffering from any of these issues. There are no variables or distributions that can be thought of that are relevant here, and so the RCT dataset is likely to be more robust than the observational dataset despite the results of the sensitivity analysis (note, however, that absence of evidence is not evidence of absence). As to why despite the optimization the Lalonde RCT data was still more sensitive, a possible explanation is the 445 sample size. This is dwarfed by the observational data containing over 16,000 observations which means more nuanced matching.

Part II: Predicting Confounders

There are scenarios where it is not enough to determine the robustness of results. In medical-drug development research for example, ignoring hidden confounders can have devastating consequences. Medical researchers need to determine what the confounder are in order to gain satisfactory results or to work on changing the drug so it is no longer influenced by said confounders. The problem with hidden confounders is you do not know what you do not know. However, given modern processing power, it is possible to systematically guess possible confounders through trial-and-error. I suggest the following method.

Randomly create a new variable (will be referred to using the Greek symbol kappa, κ) with values for all observations to represent the hidden confounder and see if matching using κ improves robustness. If κ does improve robustness, further information such as correlation with the recorded variables and distribution can be gathered to determine what κ really is. Multiple κ's will have to be created because it is a process reliant on trial-and-error, and even if many κ's are found to improve robustness there is no guarantee that those κ's are even real variables, if a hidden confounder even exists. This method is merely an attempt to guess the unknown through brute processing power.

For example, imagine a medical study is sensitive to a hidden bias caused by the unrecorded variable "blood pressure", but they have recorded and accounted for the variables of "glucose levels" and "heart-beats per minute", among other variables. Should the researchers attempt to find the confounder by creating κ's, they will eventually come across a κ that is highly correlated with the two recorded variables of "glucose levels" and "heart beats per minute", and will suspect "blood pressure" among other culprits as potential confounders to be recorded in future studies. It is clear this method is far from exact and requires on effort from the researcher, but it gives them a starting point to aim their suspicions. The question remains about how to systematically create and filter through κ's.

Part II: κ Generation

The most systematic way to create a κ list is to create values following a specified distribution, attempt every order combination between the newly created values and the units, measure if any of the combinations improves robustness, and then repeat with a new type of distribution. Even with the help of computers this is impossible. In a situation where there are 100 units observed, there are 10! = 9e157 possible combinations from one distribution. The time to match on those observations and then run sensitivity tests would be practically infinite, and that is only one attempted distribution. A different κ generation approach is necessary.

This paper suggests using a genetic algorithm to randomly generate a set of κ and search for the κ's that improve robustness based on fitness score described in part I. The genetic algorithm function is built from scratch and the code for said function is provided in the link above, and outputs the optimal κ's, their correlations with observed variables, and how well they improved robustness according to the metric described in part 1.

Part II: Analysis

The function was run five times on the Lalonde RCT dataset using the base observed variables only, a 25 population size, 10 generations four times, and the default options. While some of the generated κ's had fitness scores of 3 or more (meaning they were able to achieve statistical significance at three more Γ intervals than the original data could), none had a correlation with any of the observed variables over 10%. The immediate and most likely conclusion is that the generated κ's are completely random, and either no confounding variable exists or it is not part of the generated κ's.

In general, had there been any correlations, the researchers should have ran through ideas of what possible confounders could exist given that the confounder is related to the correlated variables, and just as informatively, unrelated to the other variables. Histograms or density plots of the κ's can also be created to check for any distributions unique to certain variables. Even then, there is no guarantee that such a confounder exists; this is merely a starting point for studies where there is a need to identify and account for confounders such as in medical studies, and there is the possibility or resources to run the experiment once more while collecting data on the new, suspected variables.

Conclusion

Part I proposed a minimization fitness function that took the number of Γ intervals chosen to construct a maximum value, then subtract based on the number of intervals that the matching is statistically significant on and add the upper-bound p-value it was not statistically significant on. This fitness function is combined with the GenMatch function to create a genetic matching algorithm to optimize for robustness. The method managed to significantly improve robustness on the Lalonde dataset (both observational and RCT).

Part II expanded on the work of part I to create a genetic algorithm that finds randomly created new covariates that improves robustness, calculating the correlation between the new and original covariates. The logic being, given enough randomization, if a hidden covariate exists, the proposed method will come across it and the correlations will serve as a clue to what the hidden covariate is. This method is severely limited because it provides nothing concrete. Finding no κ's that improve robustness is a soft indicator that no covariate exists but finding κ's that do is no indicator of anything as it could be caused due to randomness. Similar thinking is applied to presence of correlations. There were numerous κ's that significantly improved robustness in Lalonde, though no correlations were found. Even if the κ's exist in reality, the lack of correlations gives no direction to what they are. In the end the method is not analytical, and a hit or miss; however, this is a structural epistemological limitation surrounding hidden covariates. Therefore the method still holds significant potential for studies incentivized to find the hidden confounders where any method is better than no method.

It is not clear whether a genetic algorithm is optimal for generating and testing κ's. Moreover, the current proposed algorithm is inefficient and could be optimized further, especially considering it is computationally intensive.

The author would like to thank Alexis Diamond, Ph.D., and the author's Professor of Computational Sciences.

References

Dehejia, R. H., & Wahba, S. (1999). Causal effects in nonexperimental studies:

Reevaluating the evaluation of training programs. Journal of the American statistical Association, 94(448), 1053-1062.

Rosenbaum, P.R. (2005). Sensitivity Analysis in Observational Studies. In Brian S. Everitt & David C. Howell (Eds.), Encyclopedia of Statistics in Behavioral Science, vol. 4 (pp. 1809-1814). Chichester: John Wiley & Sons, Ltd.

Lalonde, Robert. “Evaluating the Econometric Evaluations of Training Programs with Experimental Data.” American Economic Review, September 1986, 76(4), pp. 604–20.

Nassim Nicholas Taleb (2007) Black Swans and the Domains of Statistics, The American Statistician, 61:3, 198-200, DOI: 10.1198/000313007X219996

Imbens, G. W. (2003). Sensitivity to exogeneity assumptions in program evaluation. American Economic Review, 126-132.