Evaluation of Tuning Parameters in the Genetic Algorithm

Florian Stijven

2024-03-20

In this document, we empirically examine the influence of the tuning parameters in the genetic algorithm (GA) that is used in the implementation of the value search estimator. Details on the value search estimator are provided at the end of this document. Throughout this document, the performance of the GA is assessed by the estimated value of the estimated optimal regime. The GA maximizes the estimated value in the regime parameters, hence optimization approaches that lead to larger estimated values are better. Throughout this document, an *optimization approach* is defined as the combination of tuning parameters (i.e., population size), starting values and number of independent runs in the GA.

The performance of the optimization approaches is assessed on all five outcomes and 20 imputed data sets. For computational reasons, we do not consider all 200 imputed data sets. Nonetheless, 20 imputed data sets are sufficient to draw valuable conclusions. This document contains two main parts:

1. The influence of the population size and the number of independent runs. In this part, the regime parameters estimated by Q-learning provide the starting values for the GA.
2. The influence of the type of starting values. Three ways of determining starting values are considered: (i) parameter estimates from Q-learning (as used in part 1), (ii) random starting values, and (iii) the zero vector as starting values.

# Performance of the Genetic Algorithm in the Value Search Estimator

A GA is used in the value search estimator that is implemented in the DynTxRegime R-package. This package relies on the GA that is implemented in the rgenoud R-package. While a GA typically has many tuning parameters, the most important tuning parameter is the population size. In the documentation of the rgenoud package, it is illustrated how the population size influences the performance of the GA for a wide variety of benchmark objective functions. In these illustrations, a population size between 1000 and 10.000 is used. Larger population sizes generally (but not always) lead to better solutions, but at the price of a greatly increased running time. This improved performance is expected because the theorems proving that GAs find good solutions are asymptotic in the population size (and number of generations). Tuning parameters regarding the number of generations is left unmodified at the default values in rgenoud.

There is no guarantee that the treatment regime found by the GA corresponds to the global optimum. A *seemingly* appropriate way to assess whether the tuning parameters are appropriate is to run the GA multiple times with a different seed (and possibly different starting values). If the estimated treatment regimes across these different runs are (nearly) equivalent, then this *could* be evidence that the population size and other tuning parameters are appropriate. Nevertheless, this does not provide an absolute guarantee that a good solution, let alone a global optimum will be found. In fact, one should instead compare different optimization approaches in terms of the corresponding maximized objective functions, i.e., the estimated values. Further on, it is illustrated with the Browne data that small variability between independent runs can indeed be a misleading indicator of good performance.

In the accompanying website for the book *Dynamic Treatment Regimes* by Tsiatis et al., use of DynTxRegime is illustrated. The population size used in those illustrations is 1000. This is also the default population size in the genetic algorithm implemented in rgenoud. The population size in the code of Zhang and Zhang (2018) varies between 700 and 3000, depending on the number of parameters in the regime. The GA is run only once in those applications. However, we show next that a single run with this population size is insufficient for our application. Additionally, it will be shown that a different population size can improve the performance of the GA.

In the remainder of this section, we will vary two aspects of the tuning parameters:

1. The population size is varied between 500, 1000, and 3000.
2. The number of independent runs of the GA is varied between 1 and 5. For a population size of 500, a scenario with 20 independent runs is additionally included.

Note that it is also important to consider how “independent” runs relate to each other across these different optimization approaches.

To ensure independent runs within each multi-run approach, distinct seeds were used:

* The seed 1 was used in all single-run approaches.
* The seeds 11:15 were used in multi-run approaches with 5 independent runs.
* The seeds 101:120 were used in the multi-run approach with 20 independent runs.

Note that the same seeds are used across approaches with the same number of independent runs.

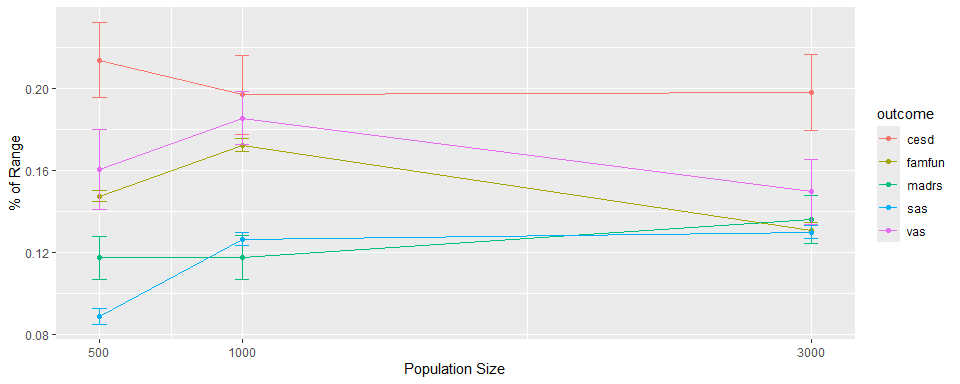
## Variation in Estimated Value in Independent Runs

In this subsection, we examine the variability between independent runs of the GA. As noted before, low variability between independent runs does not necessarily mean that the solutions are close to the global optimum. The results in this subsection should be interpreted with this in mind. In fact, the results in this and the next subsection provide a warning against using the variability between independent runs of the GA as a measure of performance.

In what follows, we will only consider the results with 5 independent runs for a population size of 500, 1000, and 3000. The standard deviation of the corresponding 5 values is computed and averaged over all imputed data sets for each of the 5 outcomes and 3 population sizes. These average standard deviations are summarized in the table and figure below. These average standard deviations summarize how much variability there is in multiple independent runs of the GA on the same data. The average standard deviation is given as a percentage of the range of the corresponding scale. This allows us to better appreciate the magnitude of the variation between multiple runs of the GA. The last column contains the corresponding standard error for the estimated average standard deviation.

| outcome | population\_size | Range of Scale | % of Range | SE for % of Range |
| --- | --- | --- | --- | --- |
| cesd | 500 | 60 | 0.214 | 0.018 |
| cesd | 1000 | 60 | 0.197 | 0.019 |
| cesd | 3000 | 60 | 0.198 | 0.018 |
| famfun | 500 | 3 | 0.148 | 0.003 |
| famfun | 1000 | 3 | 0.172 | 0.003 |
| famfun | 3000 | 3 | 0.131 | 0.004 |
| madrs | 500 | 60 | 0.117 | 0.010 |
| madrs | 1000 | 60 | 0.118 | 0.011 |
| madrs | 3000 | 60 | 0.136 | 0.012 |
| sas | 500 | 4 | 0.089 | 0.004 |
| sas | 1000 | 4 | 0.127 | 0.003 |
| sas | 3000 | 4 | 0.130 | 0.003 |
| vas | 500 | 100 | 0.160 | 0.019 |
| vas | 1000 | 100 | 0.185 | 0.013 |
| vas | 3000 | 100 | 0.150 | 0.016 |

The above table is visualized next. The error bars indicate the average standard deviation +/- one standard error.



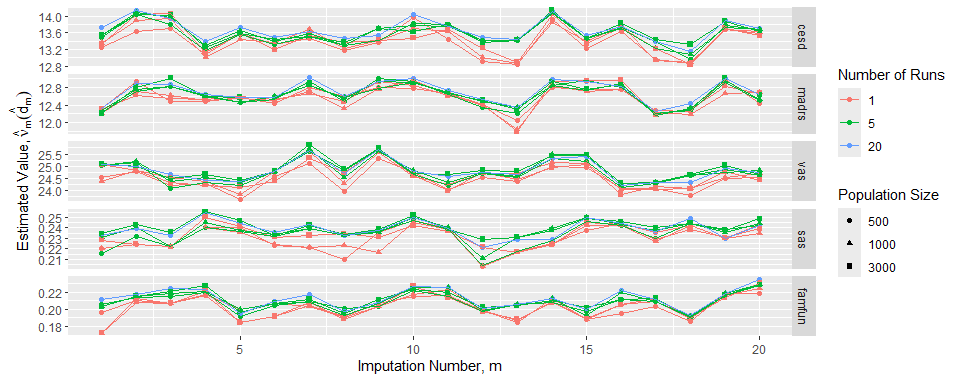
We can make three important observations with the above results.

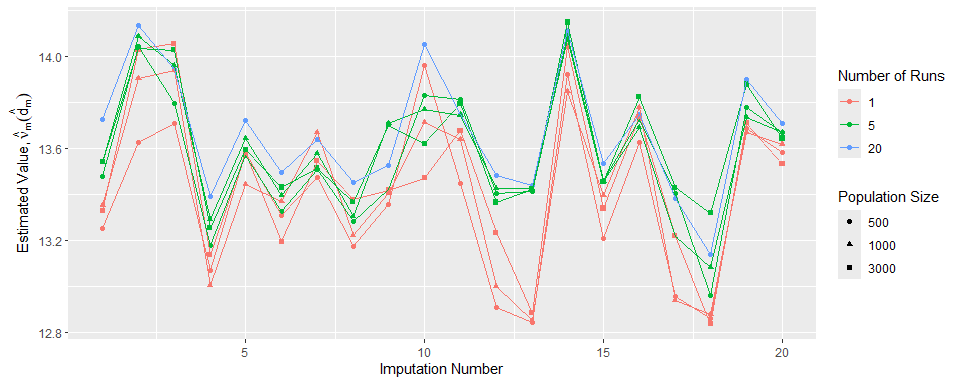
* The variation in the estimated value across multiple runs of the GA is relatively small in comparison to the ranges of the corresponding scales. Still, this variation is not negligible. Indeed, in individual data sets, the differences in estimated values across multiple runs and/or optimization approaches can be considerable. This is further shown in the next subsection.
* A larger population size does not generally lead to less variation among independent runs of the GA. The relation between this variation and the population size depends on the outcome.
* For SAS, the variation is considerably smaller with a population size of 500 than with a larger population size. Nonetheless, it is shown in the next subsection that 5 runs with a population size of 3000 is better than 20 runs with a population size of 500 for SAS. This illustrates that a small variation among independent runs of the GA does not mean that the tuning parameters are appropriate.

The above results should be interpreted with caution. Indeed, little variation among independent runs could mean two very different things: (i) the genetic algorithm often finds a similar local optimum and (ii) the GA often finds solutions close to the global optimum. Based on the above results, we cannot distinguish between these two situations since we do not know the global optimum. In what follows, we will examine this further by looking at which tuning parameters lead to the highest estimated values, i.e., which tuning parameters lead to solutions closest to the global optimum.

## Comparison of Optimization Strategies

In the following plot, we compare the performance of 5 different optimization strategies. In these five strategies, the number of independent runs and/or the population size are varied. Performance is assessed by the estimated value. Note that in the multi-run scenarios, only the run with the highest estimated value is retained.

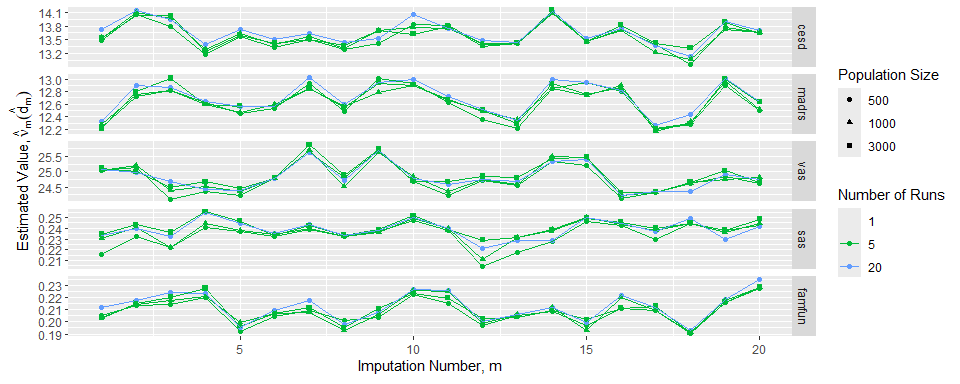




The above figure leads to important insights.

* In some imputed data sets, the difference between the optimization approaches is very substantial, e.g., first imputation for FAMFUN. In other imputed data sets, there is little difference between the optimization approaches, e.g., first imputation for MADRS.
* Multi-run approaches generally outperform the single-run approaches. This holds for all outcomes and most imputed data sets. This is expected behavior. Note that the runs in single-run approaches are *not* subsets of the runs in multi-run approaches. Hence, it is possible that a single-run approach incidentally yields a higher estimated value than the corresponding multi-run approach.
* There is no approach that consistently leads to the highest estimated value, even looking at each outcome separately.

Taking the above observations into account, we reproduce the plot above but now only including the multi-run approaches. This allows us to better examine the differences between the multi-run approaches.

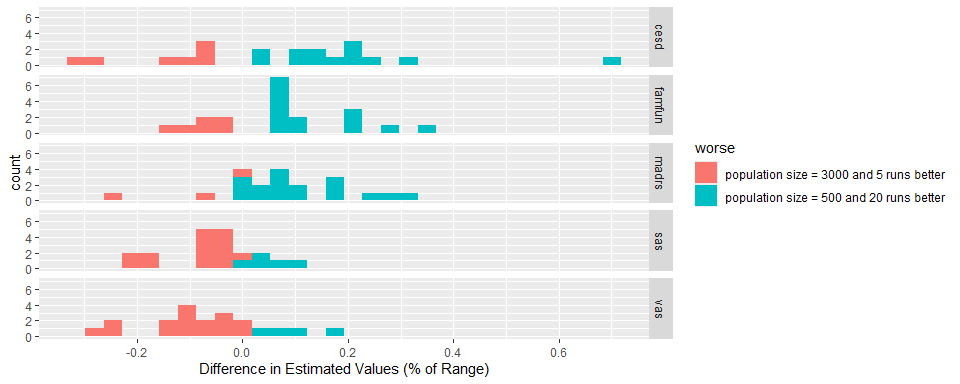


In the following table, we summarize the proportion of times each optimization approach is best across the imputed data sets, stratified by outcome. Note that the proportions for each outcome sometimes sum to values larger than 1. This is not an error since different optimization approaches lead to the same solution in some settings. In that case, both approaches are counted as “best” for that imputed data set. This table shows us that:

* 20 runs with a population size of 500 performs best for CESD, FAMFUN, and MADRS.
* 5 runs with a population size of 3000 performs best for SAS and VAS.

| population\_size | multi\_run | cesd | famfun | madrs | sas | vas |
| --- | --- | --- | --- | --- | --- | --- |
| 500 | 1 | 0.00 | 0.05 | 0.10 | 0.00 | 0.00 |
| 500 | 5 | 0.05 | 0.05 | 0.05 | 0.05 | 0.05 |
| 500 | 20 | 0.60 | 0.65 | 0.50 | 0.20 | 0.15 |
| 1000 | 1 | 0.05 | 0.05 | 0.05 | 0.00 | 0.05 |
| 1000 | 5 | 0.05 | 0.05 | 0.10 | 0.10 | 0.25 |
| 3000 | 1 | 0.05 | 0.00 | 0.15 | 0.05 | 0.00 |
| 3000 | 5 | 0.20 | 0.25 | 0.05 | 0.60 | 0.70 |

Hence, 2 different approaches perform best for the five outcomes. Because this table does not take into account the magnitude of the difference, we summarize the difference in estimated values between these two approaches next in histograms. Each histogram contains 20 such differences, 1 for each imputed data set.

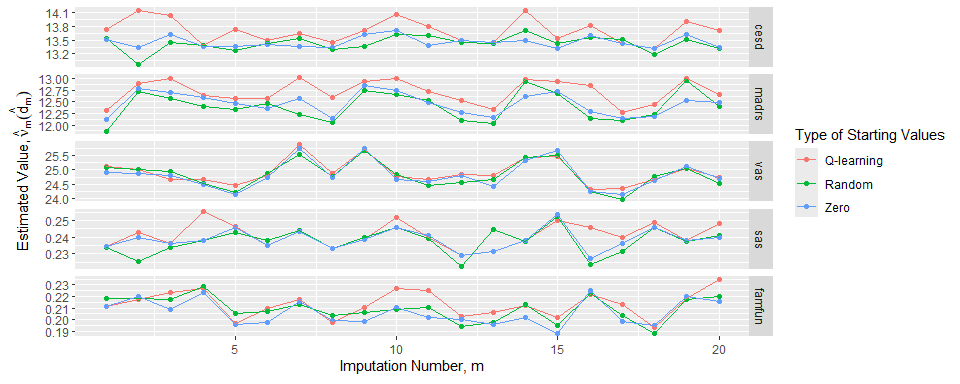


# Starting Values

In this section, we examine the influence of starting values. The following choices for starting values are considered:

1. The optimal regime is first estimated by Q-learning. This estimated parameter vector is then converted to a vector with unit norm. This normed vector is used as starting value. The class of regimes considered by Q-learning is the same as the class considered in the value search estimator. These starting values were used in the preceding sections.
2. Random starting values. These starting values are independently sampled from a uniform distribution with range .
3. All starting values are set to zero. This approach is used in the illustrations on the website of the *Dynamic Treatment Regimes* book by Tsiatis et al.

As optimization approach, we combine 5 runs with a population size of 3000 and 20 runs with a population size of 500. In the next plot, we compare the estimated values for identical optimization approaches, but different starting values.



The above plot indicates that (for the optimization approach considered) determining starting values as the estimated parameters from Q-learning performs best. The difference is very outspoken for CESD and MADRS, but not for FAMFUN, SAS, and VAS. The same table as before, summarizing the proportion of times a particular choice of starting values is best, is given next. This table confirms that starting values provided by Q-learning generally perform best.

| starting\_value | cesd | famfun | madrs | sas | vas |
| --- | --- | --- | --- | --- | --- |
| Q | 0.95 | 0.55 | 1 | 0.7 | 0.55 |
| random | 0.05 | 0.25 | 0 | 0.2 | 0.30 |
| zero | 0.15 | 0.20 | 0 | 0.4 | 0.15 |

# Conclusions

The default tuning parameters used by the DynTxRegime package result in suboptimal performance in our application. Researchers should be mindful of this when using this package. In an application with no missing data, finding the best tuning parameters is relatively straightforward, but additional issues arise when one is working with multiply imputed data sets. Based on the findings reported in this document, we can make the following recommendations to researchers applying the value search estimator after multiple imputation.

* One should be careful with interpreting a small variability between independent runs of the GA as implying appropriate tuning parameters. Instead, it is better to compare the estimated values directly between different optimization approaches.
* When searching for an optimal optimization approach with the GA, it is insufficient to look at what approach works best in a single (imputed) data set. Indeed, we found that no approach consistently performed best across all imputed data sets. Still, it might be sufficient to only select a subset of the imputed data sets for this purpose. In our application, the results were sufficiently clear for 20 imputed data sets.
* For our application, starting values provided by Q-learning performed best. The illustrations on the website of *Dynamic Treatment Regimes* use the zero vector as starting value.
* While combining all optimization approaches is in principle a viable strategy, computational load is an important limiting factor. This is even more so when one would use a resampling method such as the bootstrap or cross-validation to correct for possible overoptimism in the estimated value of the estimated regime.

# Details on the Value Search Estimator

The value search estimator estimates the optimal treatment regime within a restricted class of treatment regimes. This is done by maximizing an estimator for the value of a fixed regime, . In this document, the AIPW estimator was used for this purpose. This estimator requires the specification of a propensity score model and an outcome regression model:

* **Propensity score model**. Since treatment assignment was randomized, we estimate the propensity scores by the empirical proportions. This corresponds to a logistic regression model with only an intercept.
* **Outcome regression model**. A linear regression model with sex, past\_MDD, current\_MDD, phealth, age, madrs, sas, famfun, cesd, vas as main effects and interactions terms between treatment and sex, age, famfun, cesd, and past\_MDD.

The restricted class of treatment regimes considered in the value search estimator are all linear treatment regimes with the following covariates: sex, age, famfun, cesd, and past\_MDD.