Non-convergence in iterative imputation

H. I. Oberman

Iterative imputation has become the de facto standard to accommodate for the ubiquitous problem of missing data. While it is widely accepted that this technique can yield valid inferences, these inferences all rely on algorithmic convergence. Our study provides insight into identifying non-convergence in iterative imputation algorithms. We show that these algorithms can yield correct outcomes even when a converged state has not yet formally been reached. In the cases considered, inferential validity is achieved after five to ten iterations, much earlier than indicated by diagnostic methods. We conclude that it never hurts to iterate longer, but such calculations hardly bring added value.

Introduction

Iterative imputation has become the de facto standard to accommodate for missing data. The aim is usually to draw valid inferences, i.e. to get unbiased, confidence-valid estimates that incorporate the effects of the missingness. Such estimates are obtained with iterative imputation by separating the missing data problem from the scientific problem. The missing values are imputed (i.e., filled in) using some sort of algorithm. And subsequently, the scientific model of interest is performed on the completed data. To obtain valid scientific estimates, both the missing data problem and the scientific problem should be appropriately considered. The validity of this whole process naturally depends on the convergence of the algorithm that was used to to generate the imputations.

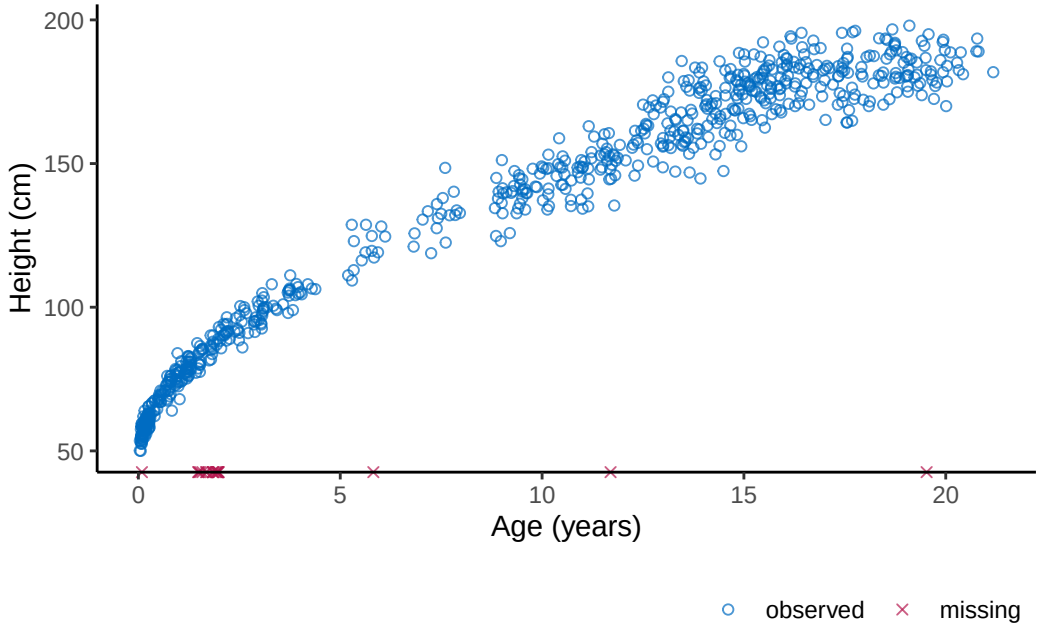
All inferences with imputed data rely on the convergence of the imputation algorithm. Yet, determining whether an algorithm has converged is not trivial. There has not been a systematic study on how to evaluate the convergence of iterative imputation algorithms. A widely accepted practice is visual inspection of the algorithm. Diagnosing convergence through visual inspection, however, may be undesirable for several reasons: 1) it may be challenging to the untrained eye, 2) only severely pathological cases of non-convergence may be diagnosed, and 3) there is not an objective measure that quantifies convergence (**buur18?**). Therefore, a quantitative diagnostic method to assess convergence would be preferred.

It is challenging to arrive upon a single point at which convergence has been reached. Since the aim is to converge to a distribution and not to a single point, the algorithm may produce some fluctuations even after it has converged. Because of this property, it may be more desirable to focus on *non*-convergence. Fortunately, there are non-convergence identifiers for other iterative algorithms, but the validity of these identifiers has not been systematically evaluated on imputation algorithms.

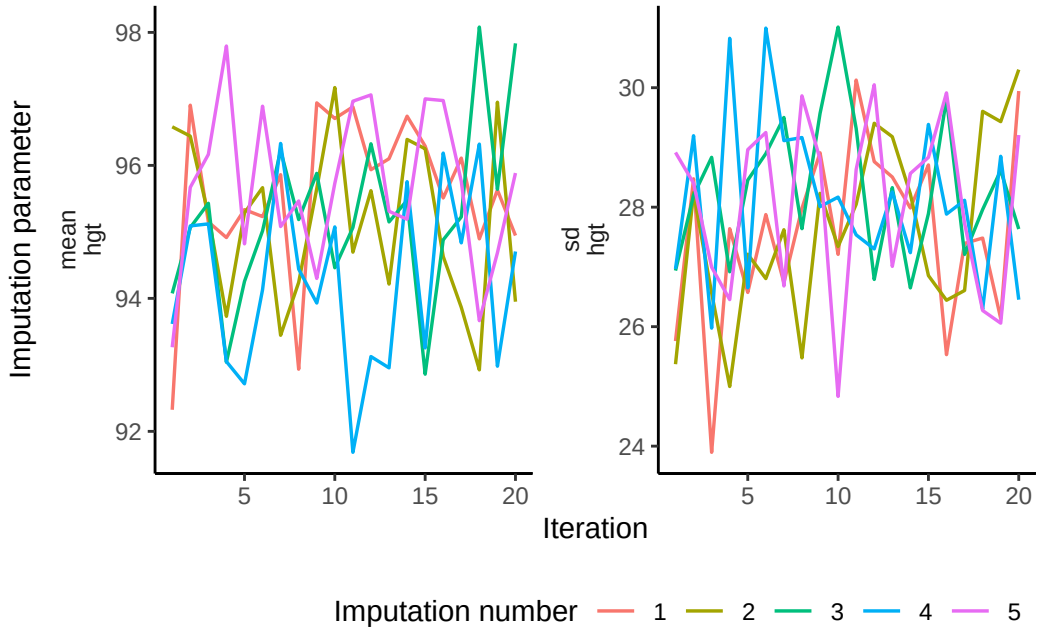
In this study, we explore different methods for identifying non-convergence in iterative imputation algorithms. We evaluate whether these methods are able to cover the extent of the non-convergence, and we also investigate the relation between non-convergence and the validity of the inferences. We translate the results of our simulation study into guidelines for practice, which we demonstrate by means of a motivating example.

Motivating Example

We use empirical incomplete data: the `boys` dataset from the `mice` package, which contains health-related data for 748 Dutch boys (Van Buuren and Groothuis-Oudshoorn 2011). Say, we're interested in the relation between children's heights and their respective ages, we could use a linear regression model to predict `hgt` from `age`. However, as figure XYZ shows, the variable `hgt` is not completely observed. To be able to analyze these data, we need to solve the missing data problem first.

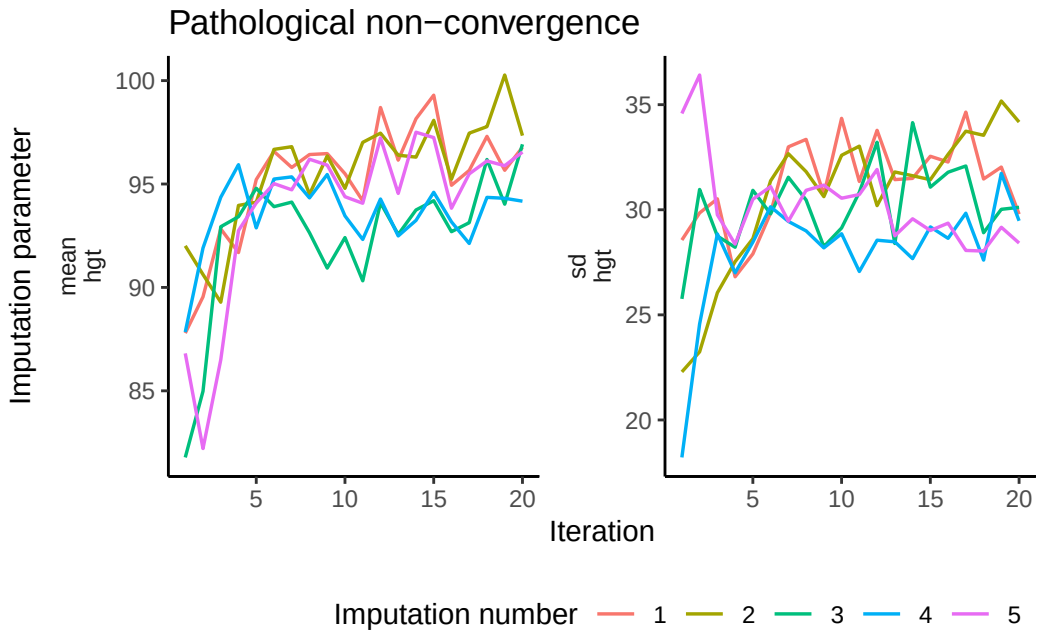


The incomplete height variable can be imputed based on auxiliary variables, such as weight. After imputation with `mice`, one would conventionally inspect the trace plots for signs of non-convergence.

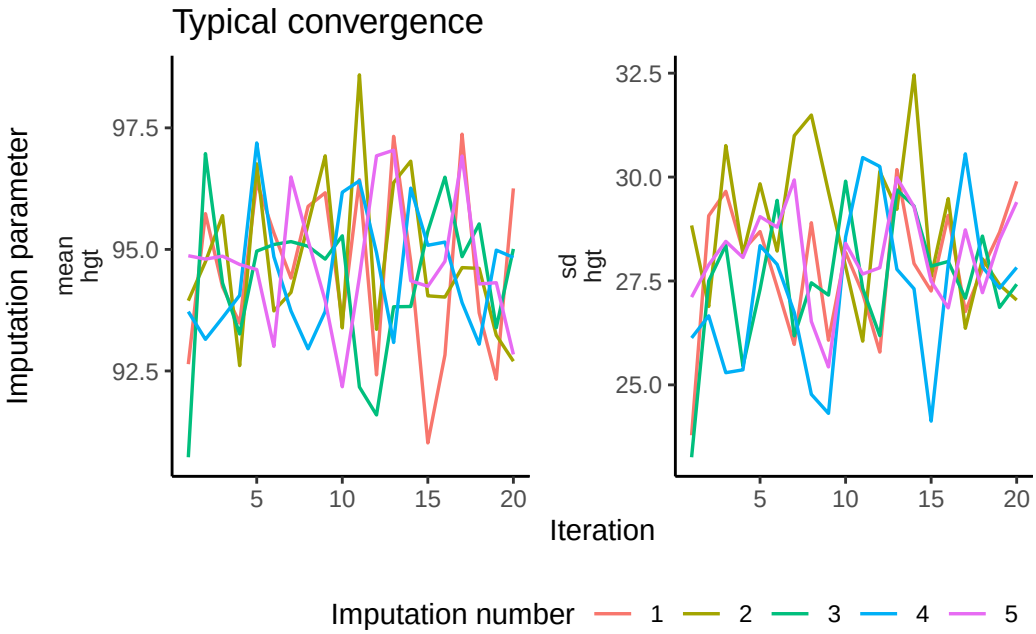



```
# impute missingness with mis-specified imputation model
meth <- make.method(boys) #define imputation model
meth["bmi"] <- "~I(wgt / (hgt / 100)^2)" #mis-specify model
nonconv <- mice( #impute missingness
  boys,
  meth = meth,
  maxit = 20,
  print = FALSE,
  seed = 60109
)

# # save results
# save(nonconv, file = "Results/example_nonconv.Rdata")


#####################
# Regular convergence
#####################
```

```r
# impute missingness with correctly specified imputation model
pred <- make.predictorMatrix(boys) #mend imputation method
pred[c("hgt", "wgt"), "bmi"] <- 0 #remove dependency
conv <- mice( #impute missingness
  boys,
  meth = meth,
  pred = pred,
  maxit = 20,
  print = FALSE,
  seed = 60109
)

plot_trace(nonconv, "hgt") +
  ggplot2::labs(title = "Pathological non-convergence")
```




```r
plot_trace(conv, "hgt") +
  ggplot2::labs(title = "Typical convergence")
```



From the traceplot of the chain means (see @ref(fig:case)A) it seems that mixing improves up-to 10 iterations, while trending is only apparent in the first three iterations.

Aims

- can we quantify non-convergence quantitatively?
- which parameter should we track?
- how many iterations are needed before convergence is reached?
- how many iterations are needed before valid inferences are reached?

Discussion

The potential scale reduction factor metric assumes over-dispersed starting values of the iterative algorithm. The MICE algorithm does not start in an over-dispersed state. MICE does not rely on starting values for parameters at all: instead, the algorithm is initialized based on a 'zeroth' iteration, where missing values are filled in using a random draw from observed values. Under MCAR, this would typically not yield an over-dispersed state at all, depending on the parameter of interest: means and variances are not affected, regression coefficients and other multivariate parameters start biased downwards (because sampling starting values distorts multivariate structures in the data). Whether this accrues to over-dispersion is questionable. Under MAR (or MNAR), some over-dispersion might occur when the observed data and imputed data differ impactfully. The algorithm should 'escape' the initial state (based

on the starting values drawn from observed data alone). Over iterations, all parameters of interest (e.g. means, variances, multivariate estimands) should converge towards a stable state. Moreover, the MICE algorithm is initialized with more information than a typical MCMC algorithm: parameter estimates depend not only on imputed data, but also on observed data that does not change over iterations. Therefore, the initial state of the algorithm is 'too good' to observe a relative decrease in $\widehat{R}$.

Van Buuren, Stef, and Karin Groothuis-Oudshoorn. 2011. "Mice: Multivariate Imputation by Chained Equations in R." *Journal of Statistical Software* 45 (1): 1–67. https://doi.org/10.18637/jss.v045.i03.