

clustord Advanced Settings

Louise McMillan

2026-03-02

Contents

Introduction	1
Important algorithm settings	2
nstarts	2
maxiter and startmaxiter	2
Advanced algorithm settings	3
Convergence and stopping conditions	3
Parameter constraints	3
Label-switching	3
A note about notation	3
References	4

Introduction

The *clustord Tutorial* vignette covers the basics of how to use the package and the *Ordinal Models* vignette describes the different ordinal models within the package.

This vignette explains the advanced settings within the main `clustord()` model-fitting function.

```
library(clustord)
df <- read.table("eval_survey.txt")
colnames(df) <- paste0("Q", 1:ncol(df))
rownames(df) <- paste0("ID", 1:nrow(df))
long_df <- mat_to_df(df)
head(long_df)
```

```
##      Y ROW COL
## 1 6    1    1
## 2 7    2    1
## 3 7    3    1
## 4 6    4    1
## 5 7    5    1
## 6 6    6    1
```

Important algorithm settings

The `clustord()` algorithm is complex, so has many settings, but a handful of them are particularly important to understand for achieving good clustering results.

The key parameters are `maxiter` and `startmaxiter` inside the `control_EM` argument, and the `nstarts` argument. All of these are related.

The EM algorithm works by iteratively improving on the parameter estimates and estimated cluster memberships. It has to start with some estimates, but it is known to sometimes be sensitive to these initial estimates. It can, therefore, get stuck near a set of parameter estimates that are better than other similar values, but which are not the best.

nstarts

One simple way that the algorithm gets around this is to test multiple different starting points, and choose the best one. **nstarts** controls how many different starting points the algorithm tries. In general, the more complex your model, the more starts you should try, because when there are more parameters there is a greater chance of the algorithm getting stuck somewhere unhelpful.

The default number of starting points is 5. If you have only 2 or 3 clusters, and you're fitting cluster-only models, that will probably be enough. But if you are using the model with individual row/column effects and there are a lot of individual rows or columns, and especially if you are fitting interaction terms, then it would be a good idea to increase the number of random starts to 10 or 20.

maxiter and startmaxiter

`maxiter`, one of the entries in the `control_EM` argument, is the number of EM iterations. In the examples above, we checked each time whether the EM algorithm had converged **before** looking at the rest of the output. If the algorithm has not converged, try running it with more random starts, or running it again to use different random starts, or if you've used a random seed rerun it with a different seed.

But if you've already tried quite a few different random starts and/or different random seeds and you still can't get it to converge, then try increasing the number of iterations, because lack of convergence means that it hit the upper limit on the number of iterations before it reached convergence.

The default number of `maxiter` is 50, so you could try 100, for example.

`startmaxiter` is another setting in the `control_EM` argument, and this controls the number of EM iterations that the algorithm goes through for each random start. This is 5 by default, and it does **not** have to be very high. It takes a while for the EM algorithm to reach convergence, but it takes very few iterations for the algorithm to distinguish between different starting points. The differences between starting points are usually much bigger than the improvement that can be achieved in a few iterations.

The default number of `startmaxiter` is 5, but if you are using lots of random starts, e.g. at least 20, then you may want to change this value **down** to 2 or 3, for example, to save a bit of computing time.

If you want to set `maxiter` or `startmaxiter`, you have to input them as part of the `control_EM` argument, which is a list object. The `control_EM` list has other settings in it by default, but you do **not** have to set these if you don't want to; you can simply set the ones you want. This works the same way that the `control` argument in R works.

So, for example, you can run `clustord()` with additional random starts and for fewer starting iterations and more main iterations than the defaults:

```
fit <- clustord(Y ~ ROWCLUST + COL, model = "POM",
  RG = 2, long_df = long_df, control_EM = list(startmaxiter = 2,
    maxiter = 100), nstarts = 10)
```

Advanced algorithm settings

Convergence and stopping conditions

`control_optim` control list for the `optim` call within the M step of the EM algorithm. See the control list Details in the `optim` manual for more info. Please note that although `optim`, by default, uses `pgtol=0` and `factr=1e7` in the L-BFGS-B method, `clustord`, by default, alters these to `pgtol=1e-4` and `factr=1e11`, but you can use this `control_optim` argument in `clustord` to revert them to the defaults if you want. The reason for the change is that the chosen values in `clustord` reduce the tolerance on the log-likelihood function optimization in order to speed up the algorithm, and because the log-likelihood is on the scale of $1e4$ for <100 rows in the data matrix and $1e6$ for 5000 rows in the data matrix, tolerance at the default `optim` scale is not as important as the choice of model type and structure or the number of starting points. If one model is better than another, it will probably have a likelihood that is better by about the size of the data matrix, which is far larger than the tolerance in the optimization. If one starting point is better than another, it will probably have a likelihood that is better on about 1/10th or 1/100th the size of the data matrix, which is still far larger than the tolerance in the optimization. If you need accurate parameter estimates, firstly make sure to try more starting points, then perform model selection first, and then finally rerun the chosen model with greater tolerance, e.g. the `optim` defaults, `pgtol=0` and `factr=1e7`.

Parameter constraints

Label-switching

A note about notation

If you are looking at the cited journal articles by Pledger and Arnold (2014), Matechou et al. (2016), and Fernández et al. (2016 and 2019), the notation in those is slightly different than the notation used in this tutorial. The package and tutorial notation was changed to reduce confusion between the parameters in the row clustering and column clustering models.

This is a glossary of the notation used in `clustord` and the corresponding notation used in the articles.

The rest of the parameters retain the same names in this tutorial and the cited references.

Note also that, although it is theoretically possible in this model structure to add α_r and α_i to the same model, ie. row cluster effects **and** individual row effects, `clustord` does not allow this, and will warn you if you try to use `Y ~ ROWCLUST + ROW` or similar formulae. And the biclustering model, which has α_r and β_c , does not allow either individual row or individual column effects, partly because this would introduce too many parameters and be too difficult to fit correctly.

Table 1: Comparison of `clustord` notation with the notation used in the original journal articles.

α_r	rowc
β_j	col
γ_{rj}	rowc_col
β_c	colc
α_i	row
γ_{ic}	colc_row
γ_{rc}	rowc_colc

References

- Agresti, A. (2010). *Analysis of ordinal categorical data*. Vol. 656, John Wiley & Sons.
- Akaike, H. (1973). Maximum likelihood identification of Gaussian autoregressive moving average models. *Biometrika*, 60(2), 255-265.
- Anderson, J. A. (1984). Regression and ordered categorical variables. *Journal of the Royal Statistical Society – Series B (Methodological)*, pp. 1–30.
- Biernacki, C., Celeux, G., Govaert, G. (2000). Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(7), 719-725.
- Dempster, A. P., Laird, N. M. and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39, pp. 1–22.
- Fernández, D., Arnold, R. and Pledger, S. (2016). Mixture-based clustering for the ordered stereotype model. *Computational Statistics & Data Analysis*, 93, pp. 46–75.
- Fernández, D., Arnold, R., Pledger, S., Liu, I., & Costilla, R. (2019). Finite mixture biclustering of discrete type multivariate data. *Advances in Data Analysis and Classification*, 13, pp. 117–143.
- Jacques, J. and Biernacki, C. (2018). Model-based co-clustering for ordinal data. *Computational Statistics & Data Analysis*, 123, pp. 101–115.
- Lloyd, S. P. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2), pp. 129–137.
- MacQueen, J. B. (1967). Some Methods for classification and Analysis of Multivariate Observations. *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*. University of California Press, 1(14), pp. 281–297.
- Matechou, E., Liu, I., Fernández, D. Farias, M., and Gjelsvik, B. (2016). Biclustering models for two-mode ordinal data. *Psychometrika*, 81, pp. 611–624.
- McLachlan, G. J. and Basford, K. E. (1988) *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, New York.
- McLachlan, G. J. and Peel, D. (2000). *Finite Mixture Models*, (Vol. 299). John Wiley & Sons.
- O'Neill, R. and Wetherill, G. B. (1971). The present state of multiple comparison methods (with discussion). *Journal of the Royal Statistical Society (B)*, 33, pp. 218–250.
- Pledger, S. and Arnold, R. (2014). Multivariate methods using mixtures: Correspondence analysis, scaling and pattern-detection. *Computational Statistics and Data Analysis* 71, pp. 241–261.
- Schwarz, G. E. (1978). Estimating the dimension of a model. *Annals of Statistics*, 6(2): 461–464, doi:10.1214/aos/1176344136.