Adjusted Limited Dependent Variable Mixture Models of Health State Utilities in R

Mark Pletscher*

2023-06-22

1 Introduction

Health-related quality of life is a key outcome in health technology assessments because it is patient-relevant and it is needed to calculate quality-adjusted life years. Quality of life instruments typically measure health problems in multiple domains using ordinal Likert scales. Value sets or valuation functions convert these profiles of ordinal measures into cardinal health-related utilities between 1 (perfect health) and minus infinity, where 0 represents death, and negative values represent health states worse than death. Because 100% quality of life represents perfect health, health state utilities are limited at 1. The lowest possible utility in a local value set further defines a lower limit of health state utilities in a local population. Thus, health state utilities are limited dependent variables. In addition, health state utilities often show gaps between 1 and the next smaller utility in the value set. These gaps occur more frequently in quality of life instruments with few levels in the Likert scales such as the EQ-5D-3L (Mulhern et al. 2018). A last but important particularity of health state utilities is that they can be the consequence of multiple latent classes, or they can exhibit multi-modal marginal densities (Hernández Alava et al. 2014).

Adjusted limited dependent variable mixture models are finite mixtures of normal distributions that account for limits, gaps between 1 and the next smaller utility value and multi-modality (Hernández Alava, Wailoo, and Ara 2012; Hernández Alava et al. 2013, 2014; Hernández Alava and Wailoo 2015; Mukuria et al. 2019). These features can improve empirical fit, parameter identification and predictive accuracy compared to standard regression models. Thus, adjusted limited dependent variable mixture models are particularly useful for mapping studies (Gray, Wailoo, and Hernández Alava 2018; Gray, Hernández Alava, and Wailoo 2018; Dixon, Hollingworth, and Sparrow 2020; Yang et al. 2019; Xu et al. 2020; Fuller et al. 2017; Pennington et al. 2020).

The R 'aldvmm' package is an implementation of the adjusted limited dependent variable mixture model proposed by Hernández Alava and Wailoo (2015) using normal component distributions and a multinomial logit model of probabilities of component membership.

The objectives of this vignette are to demonstrate the usage of the 'aldvmm' package, show important challenges of fitting adjusted limited dependent variable mixture models and validate the R implementation against the STATA® package (Hernández Alava and Wailoo 2015) using publicly available data.

2 Methods

Adjusted limited dependent variable mixture models are finite mixtures of normal distributions in K components c with conditional expectations $E[y|X,c] = X\beta^c$ and standard deviations σ^c. The probabilities of component membership are estimated using a multinomial logit model as $P[c|X] = exp(X\delta^c)/\sum_{k=1}^{K} exp(X\delta^k)$.

*Institute of Health Economics and Health Policy, Bern University of Applied Sciences, mark.pletscher@bfh.ch

The model accumulates the density mass of the finite mixture below a minimum value Ψ_1 at the value Ψ_1, and the density mass above a maximum value Ψ_2 at 1. If the maximum value Ψ_2 is smaller than 1, the model emulates a value set with a gap between 1 and the next smaller value.

$$y_i|c = \begin{cases} 1 & \text{if } y_i|c > \Psi_2 \\ \Psi_1 & \text{if } y_i|c \leq \Psi_1 \\ y_i|c & \text{if } \Psi_1 < y_i|c \leq \Psi_2 \end{cases} \tag{1}$$

In this vignette, we estimate the same models of post-operative EQ-5D-3L utilities as Hernández Alava and Wailoo (2015) and include post-operative Oxford Hip Scores (divided by 10) as the only explanatory variable x.

$$\begin{aligned} \text{Model 1:} \quad E[y|c,X] &= \beta_0^c + \beta_1^c x \\ P[c|X] &= \text{mlogit}(\delta_0^c) \\ \text{Model 2:} \quad E[y|c,X] &= \beta_0^c + \beta_1^c x \\ P[c|X] &= \text{mlogit}(\delta_0^c + \delta_1^c x) \end{aligned} \tag{2}$$

The `aldvmm()` function fits an adjusted limited dependent variable mixture model using the likelihood function from Hernández Alava and Wailoo (2015). The function calls `optimr::optimr()` to minimize the negative log-likelihood using analytical gradients from `aldvmm.gr()`. The `aldvmm()` function accepts all optimization methods available in `optimr::optimr()` except for "nlm", which requires a different implementation of the likelihood function. The default optimization method is "BFGS".

The model formula in `aldvmm()` is an object of class "formula" with two parts on the right-hand side of `~`. The first part on the left of the `|` delimiter represents the model of expected values of K normal distributions. The second part on the right of the `|` delimiter represents the model of probabilities of component membership from a multinomial logit model.

The 'aldvmm' package provides four options for the generation of starting values of the optimization algorithm.

1. "zero": A vector of zeroes (default).
2. "random": A vector of standard normal random values.
3. "constant": Parameter estimates of a constant-only model as starting values for intercepts and standard deviations, and zeroes for all other parameters.[1]
4. "sann": Parameter estimates of a simulated annealing algorithm.

The 'aldvmm' package obtains fitted values using the expected value function from Hernández Alava and Wailoo (2015). Covariance matrices and standard errors of parameters are obtained from a numerical approximation of the hessian matrix using `numDeriv::hessian()`. Standard errors of fitted values in the estimation data SE_i^{fit} and standard errors of predicted values in new data SE_i^{pred} are calculated using the delta method (Dowd, Greene, and Norton 2014; Whitmore 1986). G_i denotes the gradient of a fitted value with respect to changes in parameter estimates, Σ denotes the covariance matrix of parameters, and MSE denotes the mean squared error of fitted versus observed values in the estimation data.

$$SE_i^{fit} = \sqrt{G_i' \Sigma G_i} \tag{3}$$

$$SE_i^{pred} = \sqrt{MSE + G_i' \Sigma G_i} \tag{4}$$

[1] The auxiliary models for obtaining starting values are fitted using zero starting values.

The `aldvmm()` function returns an object of S3 class "aldvmm" for which methods for generic functions `print()`, `summary()`, `stats::predict()`, `stats::coef()`, `stats::nobs()`, `stats::vcov()`, `stats::model.matrix()`, `stats::formula()`, `stats::residuals()`, `stats::update()` and `sandwich::estfun()` are available. Objects of class "aldvmm" can be supplied to `sandwich::sandwich()`, `sandwich::vcovCL()`, `sandwich::vcovPL()`, `sandwich::vcovHAC()` and `sandwich::vcovBS()` to estimate robust or clustered standard errors (see example in the appendix). `sandwich::estfun()` calls `aldvmm.sc()` to return analytical gradients of the log-likelihood with respect to parameters for each observation, which allows fast computation of robust standard errors. `sandwich::vcovBS()` allows re-restimating the covariance matrix using bootstrapping with and without clustering, which can be particularly useful in cases with no valid covariance matrix from model fitting.

3 Installation

The latest version of the 'aldvmm' package can be installed from cran.

```
install.packages("aldvmm")
```

4 Data

We analyze the same publicly available EQ-5D-3L utility data from English patients after hip replacement in 2011 and 2012 (NHS Digital 2013) as Hernández Alava and Wailoo (2015) in their description of the STATA® ALDVMM package.

```
temp <- tempfile()

url <- paste0("https://files.digital.nhs.uk/publicationimport/pub11xxx/",
              "pub11359/final-proms-eng-apr11-mar12-data-pack-csv.zip")

download.file(url, temp)
rm(url)

df <- read.table(unz(description = temp,
                     filename = "Hip Replacement 1112.csv"),
                 sep = ",",
                 header = TRUE)

unlink(temp)
rm(temp)

df <- df[, c("AGEBAND", "SEX", "Q2_EQ5D_INDEX", "HR_Q2_SCORE")]
df <- df[df$AGEBAND != "*" & df$SEX != "*", ]

df$eq5d <- df$Q2_EQ5D_INDEX
df$hr <- df$HR_Q2_SCORE/10

df <- df[stats::complete.cases(df), ]

set.seed(101010101)
df <- df[sample(1:nrow(df), size = nrow(df)*0.3), ]
```

The data includes 35'166 observations with complete information on patients' post-operative utilities, Oxford Hip Scores, age and sex. Like Hernández Alava and Wailoo (2015), we draw a 30% sub-sample of 10'549 observations from the population of complete observations of these variables. Although we follow a similar approach in data preparation preparation as Hernández Alava and Wailoo (2015), our random sample is not identical to the data used in their study. Post-operative EQ-5D-3L utilities from English value sets (Dolan 1997) show a bimodal distribution, limits at -0.594 and 1 and a gap between 1 and 0.883 (figure 1).

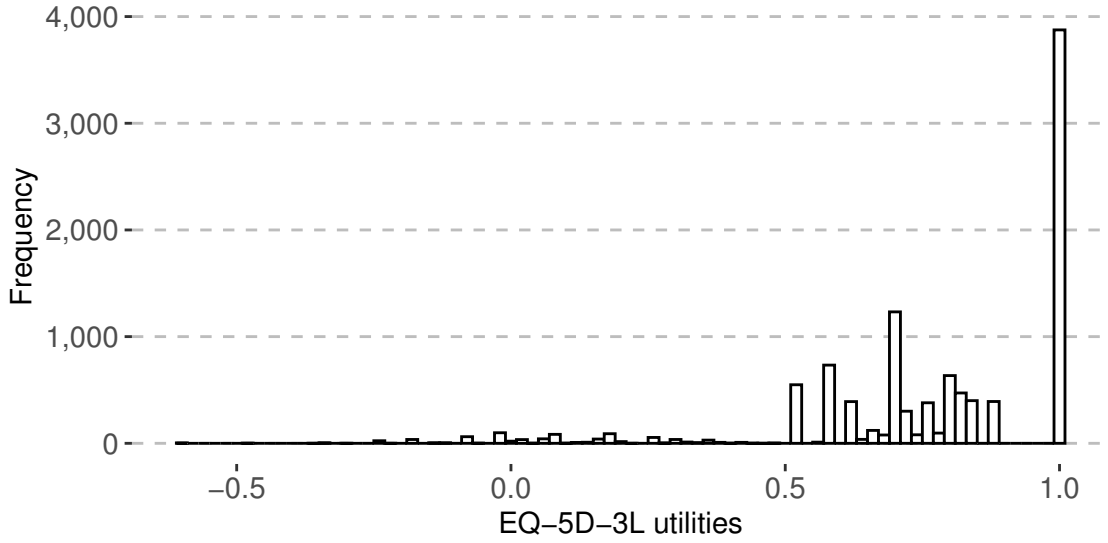

Figure 1: Frequency distribution of observed EQ-5D-3L utilities

5 Examples

5.1 Model 1: Default settings

We first fit model 1 with the default "BFGS" optimization method and "zero" initial values. The values 0.883 and -0.594 in the argument 'psi' represent the maximum and minimum values smaller than 1 in the English value set (Dolan 1997). As the data shows a bi-modal distribution (figure 1), we estimate a mixture of 2 normal distributions ('ncmp' = 2). `aldvmm()` returns an object of class "aldvmm".

```
library("aldvmm")

fit <- aldvmm::aldvmm(eq5d ~ hr | 1,
                      data = df,
                      psi = c(0.883, -0.594),
                      ncmp = 2)

summary(fit)

pred <- predict(fit,
                se.fit = TRUE,
                type = "fit")
```

We obtain a summary table of regression results using the generic function `summary()`. The model converges at a log-likelihood of -706 and an Akaike information criterion value of -1'399 (table 1). The function

`aldvmm()` returns the negative log-likelihood and Akaike information criterion, and smaller values suggest better fit.

The coefficients of the intercepts and covariates for the expected values $E[y|c, X]$ of the normal distributions can be interpreted as marginal effects on component means. 'lnsigma' denotes the natural logarithm of the estimated standard deviation σ^c. The coefficients of covariates in the multinomial logit model of probabilities of component membership are log-transformed relative probabilities. Our model only includes two components, and the multinomial logit model collapses to a binomial logit model. The intercept of -0.7281 means that the average probability of an observation in the data to belong to component 1 is $\exp(-0.7281)$ or 0.4828255 times the probability to belong to component 2.

Table 1: Regression results from model 1 with "BFGS" optimization method and "zero" starting values

		Estimate	Std. Error	t value	Pr(>|t|)	2.5 %	97.5 %
E[y|X, c]							
Comp1	(Intercept)	-0.4312	0.0224	-19.2471	0	-0.4751	-0.3873
	hr	0.3136	0.0065	47.9444	0	0.3008	0.3264
	lnsigma	-1.2479	0.0215	-57.9753	0	-1.2901	-1.2057
Comp2	(Intercept)	0.2358	0.0069	34.1725	0	0.2223	0.2493
	hr	0.1459	0.0019	76.359	0	0.1422	0.1497
	lnsigma	-2.4623	0.0178	-138.1184	0	-2.4972	-2.4273
P[c|X]							
Comp1	(Intercept)	-0.7281	0.0607	-12.0038	0	-0.847	-0.6092
N = 10549	ll = -706	AIC = -1399	BIC = -1348				

We obtain expected values of observations in the estimation data using the generic function `predict()`. Standard errors of fitted (estimation data) or predicted (new data) values are calculated using the delta method. Expected values exhibit a smoother distribution than observed values and do not show a gap between 1 and 0.883, because they are weighted averages of component distributions and 1 (figure 2).

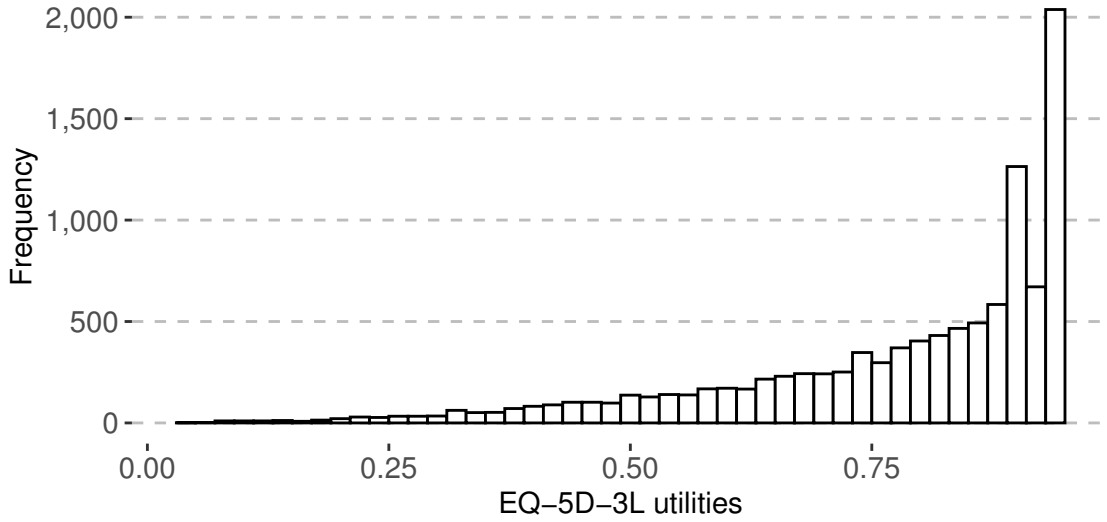

Figure 2: Expected values from base case model

Expected values of observations in the estimation data can also be used to calculate average incremental effects or average treatment effects on the treated. Average treatment effects on the treated compare predictions for treated individuals to predictions for the same individuals without the effect of the treatment

indicator. Standard errors of average treatment effects on the treated can be calculated using the delta method (see example code in the appendix).

A visual inspection of mean residuals over deciles of expected values (see example code of the modified Hosmer-Lemeshow test in the appendix) shows that model 1 fits the data poorly (figure 3). Although the test is overpowered with large data, and confidence bands should not be interpreted directly, the absolute over-prediction among individuals with low expected values is quite large.

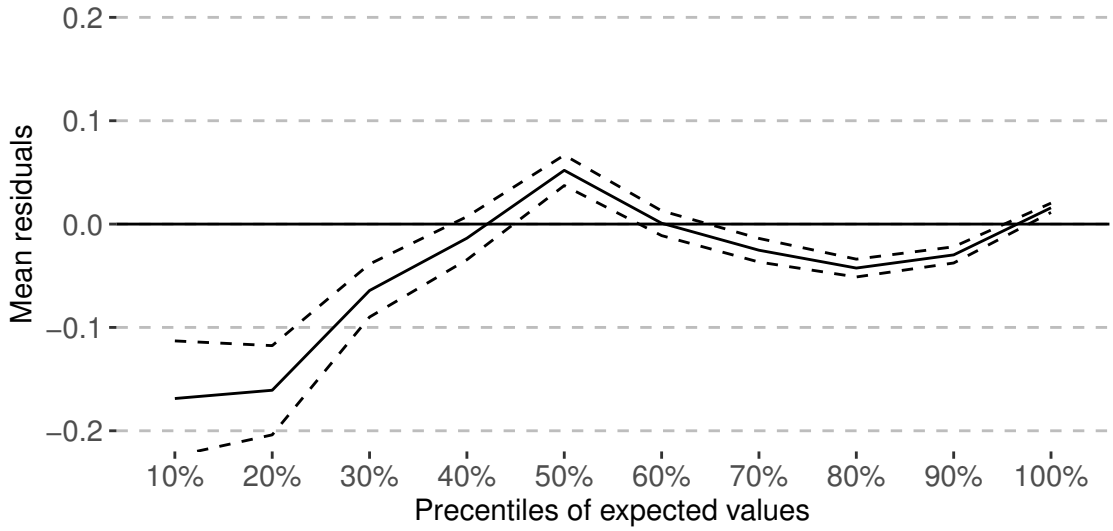

Figure 3: Mean residuals over deciles of expected values, "BFGS" with "zero" starting values

We can use the 'sandwich' package (Zeileis 2006) to calculate robust or clustered standard errors from objects of class "aldvmm" (see example code in the appendix).

5.2 Model 1: Comparison of optimization methods

Hernández Alava and Wailoo (2015) suggested that the likelihood function of the adjusted limited dependent variable mixture model with the English EQ-5D-3L data might have multiple local optima, and that the estimation is sensitive to initial values. We thus fit model 1 with all optimization algorithms and methods for generating initial values available in `aldvmm()` to assess the sensitivity of model fits to optimization settings and to find the maximum likelihood estimates.

```
init.method <- c("zero", "random", "constant", "sann")

optim.method <- c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "nlminb", "Rcgmin",
                  "Rvmmin","hjn")

fit1_all <- list()

for (i in init.method) {
  for (j in optim.method) {
    set.seed(101010101) # Seed for random starting values
    fit1_all[[i]][[j]] <- aldvmm::aldvmm(eq5d ~ hr | 1,
                                         data         = df,
                                         psi          = c(0.883, -0.594),
```

```
                                        ncmp         = 2,
                                        init.method  = i,
                                        optim.method = j)
  }
}
```

The maximum likelihood varies considerably across optimization methods and initial values which confirms the sensitivity of the model to changes in these settings (table 2). The most frequent log-likelihood is 706.32, but the Hooke and Jeeves Pattern Search Optimization ("hjn") with "zero" initial values converges at a log-likelihood 33'057.43.

The optimization methods "Nelder-Mead", "CG", "L-BFGS-B", and "Rvmmin" are particularly sensitive to changes of starting values. Only the method "nlminb" converge at the same log-likelihood of 706.32 regardless of initial values. However, even the "nlminb" algorithm converge at different parameters with starting values from a constant-only model than with the other starting values.

Table 2: Log-likelihood by optimization method

	Nelder-Mead	BFGS	CG	L-BFGS-B	nlminb	Rcgmin	Rvmmin	hjn
zero	-259.21	706.32	-634.81	-4513.33	706.32	-634.81	-8393.14	33057.43
random	-434.21	-616.85	150.59	-2830.18	706.32	706.32	-11074.58	-627.78
constant	354.36	706.32	-634.81	-634.81	706.32	-634.81	-10867.67	706.32
sann	706.19	706.32	580.53	706.31	706.32	706.32	706.32	706.32

The computation times of optimization routines were relatively short and did not vary considerably across methods expect for the "hjn" algorithm and simulated annealing "sann" starting values which requires more computation time (table 3).

Table 3: Estimation time [minutes] by optimization method

	Nelder-Mead	BFGS	CG	L-BFGS-B	nlminb	Rcgmin	Rvmmin	hjn
zero	0.14	0.09	0.15	0.04	0.10	0.45	0.04	18.37
random	0.14	0.14	0.16	0.06	0.11	0.51	0.04	0.44
constant	0.22	0.13	0.19	0.07	0.13	0.39	0.04	1.19
sann	2.09	2.13	2.17	2.23	2.05	2.49	2.04	4.75

Parameter estimates differ considerably across selected optimization algorithms (table 4). The default "BFGS" - "zero" approach converges at the same parameters as the "nlminb" - "zero" method. The solution of the "hjn" method is rather extreme with no effect of the Oxford Hip Score, a standard deviation of almost 0 in component 1 and a very low probability of membership of component 1. Also, the Modified Hosmer-Lemeshow test did not show better fit of the "hjn" solution compared to the other algorithms (figure 8 in the appendix). The solution of the "hjn" method suggests that a single-component model should be explored as well.

As adjusted limited dependent variable mixture models are frequently used for tasks that rely on predictions, we also compare expected values from the "BFGS", "Nelder-Mead" and "hjn" methods to expected values from the "nlminb" method. Expected values from the "Nelder-Mead" and "hjn" methods are smaller than "nlminb" values among observations with low and high expected values, while predictions from "BFGS" match predictions from the "nlminb" model fit closely (figure 4).

Table 4: Regression results of model 1 with zero starting values in "BFGS", "Nelder-Mead", "nlminb" and "hjn" algorithms

		BFGS	Nelder-Mead	nlminb	hjn
E[y|X, c]					
Comp1	(Intercept)	-0.4312	-0.0575	-0.4307	0.691
	hr	0.3136	0.2233	0.3135	0
	lnsigma	-1.2479	-1.8381	-1.2481	-36.737
Comp2	(Intercept)	0.2358	4.4022	0.2358	-0.149
	hr	0.1459	-0.9974	0.1459	0.25
	lnsigma	-2.4623	0.125	-2.4624	-1.589
P[c|X]					
Comp1	(Intercept)	-0.7281	3.8489	-0.728	-2.19
ll		-706.32	259.21	-706.32	-33057.43

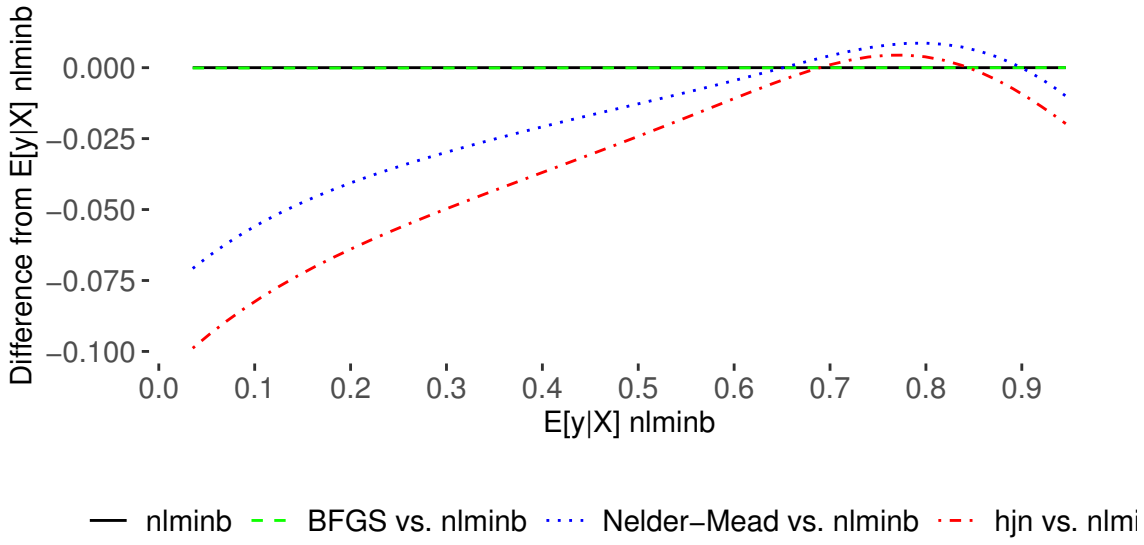

Figure 4: Deviation of expected values from "BFGS", "Nelder-Mead" and "hjn" versus "nlminb" with zero starting values

Based on the comparison of log-likelihoods, parameter estimates and predicted values, we deem the "nlminb" optimization method with "zero" starting values the most robust approach for the used data and model.

5.3 Model 1: Constrained optimization with user-defined initial values

We can also fit model 1 with user-defined starting values and box constraints. When constraints are imposed, the `aldvmm()` function uses the optimization method "L-BFGS-B", which shows to be very sensitive to starting values. We use zero initial values for all parameters except for the intercept in the multinomial logit which we set to the estimate from the "nlminb" optimization method with "zero" starting values (0.7283) (table 4). We impose a lower limit of -3 to the log-standard deviations in both components.

The `aldvmm()` function returns a warning that the covariance matrix included negative values on the diagonal. We see that these values are the variances of the intercept and the log-standard deviation in component 2 (table 5). The log-likelihood amounts to 628, which is smaller than the maximum log-likelihoods of model fits without constraints. The parameter estimates do not resemble any of the solutions of the unconstrained "BFGS", "Nelder-Mead", "nlminb" or "hjn" optimization methods with "zero" starting values (table 4),

which further emphasizes the difficulties in finding a global optimum of the likelihood with English EQ-5D-3L utilities after hip replacement.

```
init <- c(0,    0,   0,   0,    0,    0,    0.7283)
lo   <- c(-Inf, -Inf, -3,  -Inf, -Inf, -3, -Inf)
hi   <- c(Inf,  Inf,  Inf, Inf,  Inf,  Inf,  Inf)

fit1_cstr <- aldvmm::aldvmm(eq5d ~ hr | 1,
                            data = df,
                            psi = c(0.883, -0.594),
                            ncmp = 2,
                            init.est = init,
                            init.lo = lo,
                            init.hi = hi)

summary(fit1_cstr)
```

Table 5: Regression results of model 1 with the "L-BFGS-B" method, parameter constraints and user-defined starting values

		Estimate	Std. Error	t value	Pr(>|t|)	2.5 %	97.5 %
E[y|X, c]							
Comp1	(Intercept)	-0.092	0.0085	-10.8048	0	-0.1087	-0.0753
	hr	0.2325	0.0023	101.8493	0	0.228	0.237
	lnsigma	-1.6406	0.0089	-183.9809	0	-1.6581	-1.6231
Comp2	(Intercept)	0.3931	NaN	NaN	NaN	NaN	NaN
	hr	2.5249	7.4795	0.3376	0.7357	-12.1363	17.1861
	lnsigma	-0.7651	NaN	NaN	NaN	NaN	NaN
P[c|X]							
Comp1	(Intercept)	6.852	0.5388	12.7168	0	5.7958	7.9082
N = 10549	ll = 628	AIC = 1270	BIC = 1320				

5.4 Model 1: Single-component model

As the solution of the “hjn” algorithm includes a component with very low probability, we also estimate a single-component model.

```
fit <- aldvmm::aldvmm(eq5d ~ hr,
                      data = df,
                      psi = c(0.883, -0.594),
                      ncmp = 1,
                      init.method = "zero",
                      optim.method = "nlminb")

summary(fit)
```

The coefficients of the single-component model are relatively similar to the parameters in the second component of model 1 from the “hjn” algorithm (table 6). The Akaike information criterion amounts to 1’276 which is larger than the values of the “nlminb” (-706.32) and “hjn” (-33’057.43) solutions of the two-component model and thus suggests worse fit of the single-component model.[2]

[2]In the ‘aldvmm()‘ output, smaller values of the Akaike information criterion indicate better goodness of fit.

Table 6: Regression results of model 1 with 1 component, zero starting values in "nlminb" algorithm

		Estimate	Std. Error	t value	Pr(>|t|)	2.5 %	97.5 %
E[y|X, c]							
Comp1	(Intercept)	-0.0886	0.0085	-10.4099	0	-0.1053	-0.0719
	hr	0.2318	0.0023	101.4681	0	0.2273	0.2362
	lnsigma	-1.6363	0.0089	-184.2422	0	-1.6537	-1.6189
N = 10549	ll = 635	AIC = 1276	BIC = 1297				

5.5 Model 2: User-defined starting values

As an alternative specification, we explore model 2 with a coefficient of the Oxford Hip score in the multinomial logit model of component membership. For this fit, we use the method "nlminb" with estimates from Hernández Alava and Wailoo (2015) as starting values.

```
init <- c(-.40293118, .30502755, .22614716, .14801581, -.70755741, 0,
          -1.2632051, -2.4541401)

fit2 <- aldvmm::aldvmm(eq5d ~ hr | hr,
                       data = df,
                       psi = c(0.883, -0.594),
                       ncmp = 2,
                       init.est = init,
                       optim.method = "nlminb")

summary(fit2)
```

The Akaike information criterion of model 2 fitted using the "nlminb" method amounts to -1'867 which is smaller than the Akaike information criterion of model 1 (-706.32) with the same method. The smaller Akaike information criterion suggests that the increase in the log-likelihood after inclusion of a coefficient of the Oxford Hip Score on the probability of component membership is sufficiently large to justify the extra parameter.[3]

Table 7: Regression results of model 2 with user-defined starting values in the "nlminb" algorithm

		Estimate	Std. Error	t value	Pr(>|t|)	2.5 %	97.5 %
E[y|X, c]							
Comp1	(Intercept)	0.0031	0.0281	0.1084	0.9137	-0.0521	0.0582
	hr	0.0965	0.0116	8.3519	0	0.0739	0.1192
	lnsigma	-1.2676	0.0314	-40.4141	0	-1.3291	-1.2061
Comp2	(Intercept)	0.1816	0.0074	24.5105	0	0.1671	0.1961
	hr	0.1609	0.0019	85.4201	0	0.1573	0.1646
	lnsigma	-2.2809	0.0126	-181.0789	0	-2.3056	-2.2563
P[c|X]							
Comp1	(Intercept)	2.4455	0.1719	14.2293	0	2.1086	2.7824
	hr	-1.3902	0.0566	-24.5697	0	-1.5011	-1.2793
N = 10549	ll = -941	AIC = -1867	BIC = -1809				

[3] In the 'aldvmm()' output, smaller values of the Akaike information criterion indicate better goodness of fit.

6 Comparison to STATA® results

To validate the R implementation of adjusted limited dependent variable mixture models, we estimate the four models presented in Hernández Alava and Wailoo (2015) as reference cases in R and STATA®.[4]

1. Model 1 with default options
2. Model 1 with parameter constraints
3. Model 1 with initial values from constant-only model
4. Model 2 with user-defined initial values

The parameter estimates and standard errors obtained in R are very similar to the results from STATA® (table 8 and table 9). R did not obtain any standard errors in reference model 2 while STATA® returns standard errors for the first component and the probability of belonging to component 1. Although reference models 1 and 3 yield different parameter estimates, they converge at the same log-likelihood which further supports the hypothesis of multiple local optima of the likelihood. The relative ordering of models is consistent across platforms.

Table 8: Comparison of point estimates to the results of the STATA package

		(1)		(2)		(3)		(4)	
		R	STATA	R	STATA	R	STATA	R	STATA
E[y|X, c]									
Comp1	(Intercept)	-0.4307	-0.427	-0.092	-0.092	0.2358	0.236	0.0031	0.006
	hr	0.3135	0.312	0.2325	0.232	0.1459	0.146	0.0965	0.095
	lnsigma	-1.2481	-1.251	-1.6406	-1.641	-2.4624	-2.463	-1.2676	-1.274
Comp2	(Intercept)	0.2358	0.236	100	100.000	-0.4307	-0.427	0.1816	0.182
	hr	0.1459	0.146	0	0.000	0.3135	0.312	0.1609	0.161
	lnsigma	-2.4624	-2.463	0	0.000	-1.2481	-1.251	-2.2809	-2.280
P[c|X]									
Comp1	(Intercept)	-0.728	-0.725	6.8564	6.855	0.728	0.725	2.4455	2.448
	hr							-1.3902	-1.393
N = 10549	ll	-706	715.84	628	-613.7	-706	715.84	-941	953.2

Table 9: Comparison of standard errors to the results of the STATA package.

		(1)		(2)		(3)		(4)	
		R	STATA	R	STATA	R	STATA	R	STATA
E[y|X, c]									
Comp1	(Intercept)	0.0224	0.022	NA	0.009	0.0069	0.007	0.0281	0.028
	hr	0.0065	0.006	NA	0.002	0.0019	0.002	0.0116	0.012
	lnsigma	0.0215	0.021	NA	0.009	0.0178	-0.018	0.0314	0.032
Comp2	(Intercept)	0.0069	0.007	NA		0.0224	0.022	0.0074	0.007
	hr	0.0019	0.002	NA		0.0065	0.006	0.0019	0.002
	lnsigma	0.0178	0.018	NA		0.0215	0.021	0.0126	0.013
P[c|X]									
Comp1	(Intercept)	0.0607	0.061	NA	0.540	0.0607	0.061	0.1719	0.172
	hr							0.0566	0.056
N = 10549	ll	-706	715.84	628	-613.7	-706	715.84	-941	953.2

[4]The STATA® and R code for model estimation is included in the appendix.

Fitted values show very similar marginal distributions on both platforms (figure 5). R does not return fitted values in reference case 2. The summary statistics of differences in fitted values between R and STATA® suggest that individual predictions are quite similar across platforms as well (table 10).

Standard errors of fitted values differ visibly between platforms (figure 6 and table 11). The difference is particularly pronounced in reference case 1, but the standard errors from STATA® seem quite extreme compared to all other reference cases. R does not return standard errors of fitted values in reference case 2.

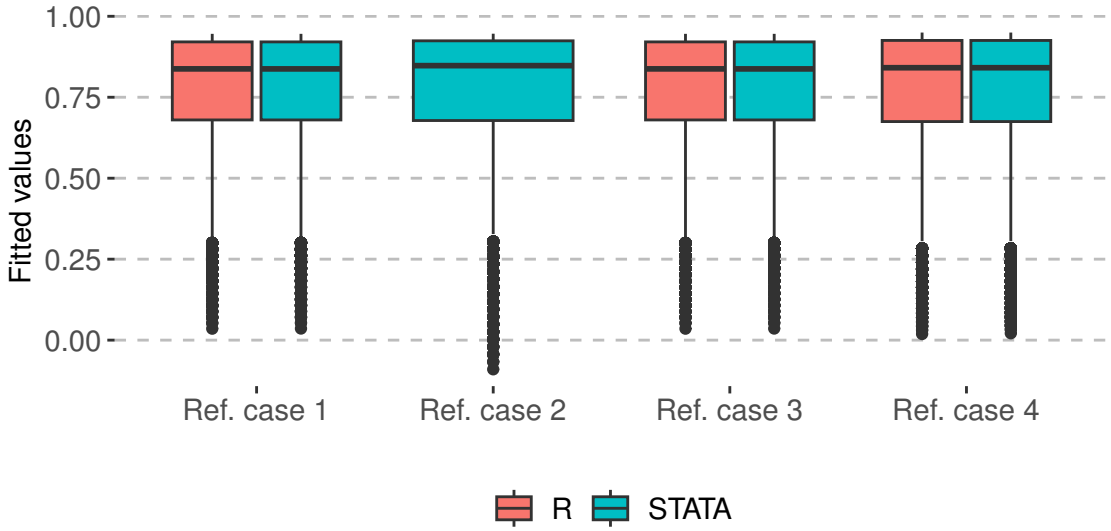

Figure 5: Fitted values in R and STATA

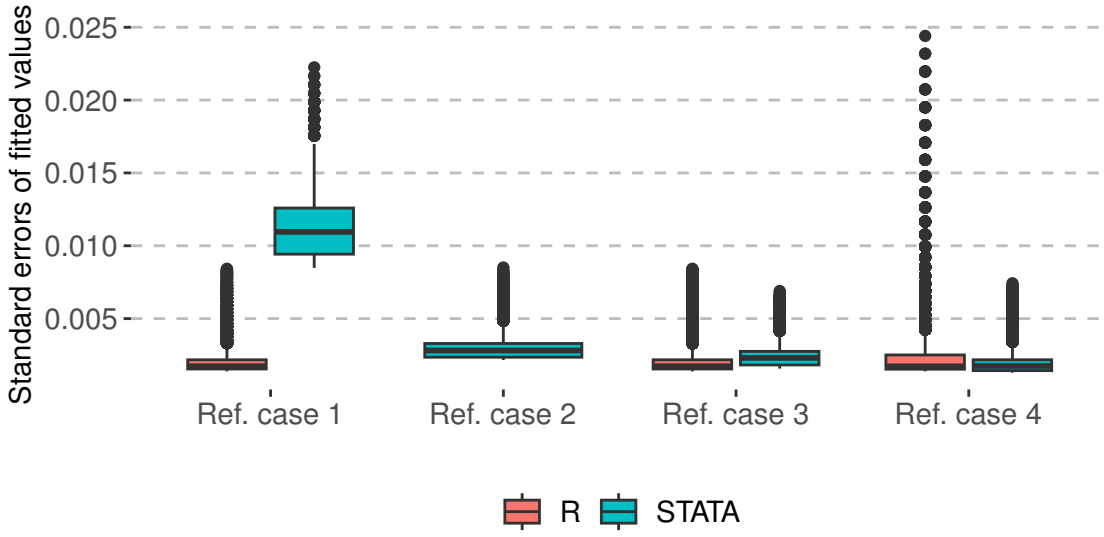

Figure 6: Standard errors of fitted values in R and STATA

The comparison of the R and STATA® packages shows that the R implementation sometimes behaves differently than the STATA® package, but the results are not indicative of technical errors in the R implementation.

Table 10: Summary statistics of differences of fitted values in R and STATA (positive values suggest larger values in STATA)

	Ref. case 1	Ref. case 2	Ref. case 3	Ref. case 4
Min.	-0.000038		-0.000038	-0.000051
1st Qu.	-0.000036		-0.000036	-0.000044
Median	-0.000033		-0.000033	0.000008
Mean	0.000037		0.000037	0.000035
3rd Qu.	0.000047		0.000047	0.000070
Max.	0.000567		0.000567	0.002319

Table 11: Summary statistics of differences of standard errors of fitted values in R and STATA (positive values suggest larger values in STATA)

	Ref. case 1	Ref. case 2	Ref. case 3	Ref. case 4
Min.	0.005897		-0.002063	-0.016984
1st Qu.	0.006898		-0.000551	-0.000823
Median	0.008690		0.000189	-0.000364
Mean	0.008917		0.000171	-0.000515
3rd Qu.	0.011050		0.001054	0.000354
Max.	0.013830		0.001530	0.000792

7 Discussion

Adjusted limited dependent variable mixture models are powerful tools for regression analysis of health state utilities. Unlike standard regression models, adjusted limited dependent variable mixture models account for limits, gaps and multi-modal distributions.

The comparison of different optimization methods with EQ-5D-3L utility data from English patients after hip replacement in 2011 and 2012 (NHS Digital 2013) showed that the likelihood function can be challenging to maximize and can converge at local optima or extreme solutions. Parameter estimates varied considerably across optimization methods and even across methods with the same maximum log-likelihood. However, fitted values were very similar across the four reference cases which suggests that the model is more robust for the identification of incremental and average marginal effects than for parameter identification.

The 'aldvmm' package offers a variety of optimization algorithms and methods for generating initial values which is an important strength in such challenging situations. It is essential to assess different optimization algorithms and methods for initial values before interpreting the parameter estimates or predictions of adjusted limited dependent variable mixture models.

The analysis of the EQ-5D-3L utility data also suggests that simpler models with fewer components should be considered when multi-component models are difficult to fit. Even single-component adjusted limited dependent variable mixture models can improve fit compared to traditional regression techniques because they account for limits and gaps.

Although coefficients can be interpreted as marginal effects within each component, they cannot be interpreted in terms of overall expected values. Thus, average marginal effects and average treatment effects need to be calculated from predictions using the generic function `predict()`. Standard errors of marginal effects or average treatment effects can be calculated using the delta method.

When no valid covariance matrix of parameters can be obtained or the user is in doubt of the validity of standard errors, the function `sandwich::vcovBS()` can be used to obtain bootstrapped standard errors. Due to the large variability of results across model fits, a large number of iterations might be needed which increases computation time. Other functions from the 'sandwich' package can also be used to estimate robust or clustered standard errors.

In situations with repeated utility measures, the 'aldvmm' package only allows fixed effects estimations with group and time fixed effects which can be an important limitation in the analysis of clinical data. However, time fixed effects can be an appropriate modeling strategy in the presence of general time trends and dynamic selection in the population, e.g. because health state utilities decrease over time and treated individuals survive longer and thus are over-represented in later measurements.

Possible extensions of the 'aldvmm' package could include additional component distributions, a mixed model implementation for repeated measures and an implementation of functions for calculating average marginal effects and their standard errors.

References

Dixon, Padraig, William Hollingworth, and John Sparrow. 2020. "Mapping to Quality of Life and Capability Measures in Cataract Surgery Patients: From Cat-PROM5 to EQ-5D-3L, EQ-5D-5L, and ICECAP-o Using Mixture Modelling." *MDM Policy & Practice* 5 (1): 2381468320915447.

Dolan, Paul. 1997. "Modeling Valuations for EuroQol Health States." *Medical Care*, 1095–1108.

Dowd, Bryan E, William H Greene, and Edward C Norton. 2014. "Computation of Standard Errors." *Health Services Research* 49 (2): 731–50.

Fuller, Gordon Ward, Monica Hernandez, David Pallot, Fiona Lecky, Mathew Stevenson, and Belinda Gabbe. 2017. "Health State Preference Weights for the Glasgow Outcome Scale Following Traumatic Brain Injury: A Systematic Review and Mapping Study." *Value in Health* 20 (1): 141–51.

Gray, Laura A, Mónica Hernández Alava, and Allan J Wailoo. 2018. "Development of Methods for the Mapping of Utilities Using Mixture Models: Mapping the AQLQ-s to the EQ-5D-5L and the HUI3 in Patients with Asthma." *Value in Health* 21 (6): 748–57.

Gray, Laura A, Allan J Wailoo, and Mónica Hernández Alava. 2018. "Mapping the FACT-b Instrument to EQ-5D-3L in Patients with Breast Cancer Using Adjusted Limited Dependent Variable Mixture Models Versus Response Mapping." *Value in Health* 21 (12): 1399–1405.

Hernández Alava, Mónica, and Allan Wailoo. 2015. "Fitting Adjusted Limited Dependent Variable Mixture Models to EQ-5D." *The Stata Journal* 15 (3): 737–50.

Hernández Alava, Mónica, Allan J Wailoo, and Roberta Ara. 2012. "Tails from the Peak District: Adjusted Limited Dependent Variable Mixture Models of EQ-5D Questionnaire Health State Utility Values." *Value in Health* 15 (3): 550–61.

Hernández Alava, Mónica, Allan Wailoo, Fred Wolfe, and Kaleb Michaud. 2013. "The Relationship Between EQ-5D, HAQ and Pain in Patients with Rheumatoid Arthritis." *Rheumatology* 52 (5): 944–50.

———. 2014. "A Comparison of Direct and Indirect Methods for the Estimation of Health Utilities from Clinical Outcomes." *Medical Decision Making* 34 (7): 919–30.

Mukuria, Clara, Donna Rowen, Sue Harnan, Andrew Rawdin, Ruth Wong, Roberta Ara, and John Brazier. 2019. "An Updated Systematic Review of Studies Mapping (or Cross-Walking) Measures of Health-Related Quality of Life to Generic Preference-Based Measures to Generate Utility Values." *Applied Health Economics and Health Policy*, 1–19.

Mulhern, Brendan, Yan Feng, Koonal Shah, Mathieu F Janssen, Michael Herdman, Ben van Hout, and Nancy Devlin. 2018. "Comparing the UK EQ-5D-3L and English EQ-5D-5L Value Sets." *Pharmacoeconomics* 36 (6): 699–713.

NHS Digital. 2013. "Finalised Patient Reported Outcome Measures (PROMs) in England - April 2011 to March 2012. Patient Reported Outcome Measures (PROMs)." *Https://Digital.nhs.uk/Data-and-Information/Publications/Statistical/Patient-Reported-Outcome-Measures-Proms/Finalised-Patient-Reported-Outcome-Measures-Proms-in-England-April-2011-to-March-2012* October 15, 2013.

Pennington, Becky M, Mónica Hernández Alava, Philip Hykin, Sobha Sivaprasad, Laura Flight, Abualbishr Alshreef, and John Brazier. 2020. "Mapping from Visual Acuity to EQ-5D, EQ-5D with Vision Bolt-on, and VFQ-UI in Patients with Macular Edema in the LEAVO Trial." *Value in Health* 23 (7): 928–35.

Whitmore, GA. 1986. "Prediction Limits for a Univariate Normal Observation." *The American Statistician* 40 (2): 141–43.

Xu, Richard Huan, Eliza Lai Yi Wong, Jun Jin, Ying Dou, and Dong Dong. 2020. "Mapping of the EORTC QLQ-C30 to EQ-5D-5L Index in Patients with Lymphomas." *The European Journal of Health Economics* 21 (9): 1363–73.

Yang, Fan, Carlos KH Wong, Nan Luo, James Piercy, Rebecca Moon, and James Jackson. 2019. "Mapping the Kidney Disease Quality of Life 36-Item Short Form Survey (KDQOL-36) to the EQ-5D-3L and the EQ-5D-5L in Patients Undergoing Dialysis." *The European Journal of Health Economics* 20 (8): 1195–1206.

Zeileis, Achim. 2006. "Object-Oriented Computation of Sandwich Estimators." *Journal of Statistical Software* 16 (9): 1–16. https://doi.org/10.18637/jss.v016.i09.

Appendix

Example calculation of average treatment effects on the treated

```
# Create treatment indicator
#---------------------------

df$treated <- as.numeric(df$SEX == "2")

# Fit model
#----------

formula <- eq5d ~ treated + hr | 1

fit <- aldvmm(formula,
              data = df,
              psi = c(-0.594, 0.883))

# Predict treated
#----------------

# Subsample of treated observations
tmpdf1 <- df[df$treated == 1, ]

# Design matrix for treated observations
X1 <- aldvmm.mm(data = tmpdf1,
                formula = fit$formula,
                ncmp = fit$k,
                lcoef = fit$label$lcoef)

# Average expected outcome of treated observations
mean1 <- mean(predict(fit,
                      newdata = tmpdf1,
                      type = "fit",
                      se.fit = TRUE)[["yhat"]], na.rm = TRUE)

# Predict counterfactual
#-----------------------

# Subsample of counterfactual observations
tmpdf0 <- tmpdf1
rm(tmpdf1)
tmpdf0$treated <- 0

# Design matrix for counterfactual observations
X0 <- aldvmm.mm(data = tmpdf0,
                formula = fit$formula,
                ncmp = fit$k,
                lcoef = fit$label$lcoef)

# Average expected outcome of counterfactual osbervations
mean0 <- mean(predict(fit,
                      newdata = tmpdf0,
```

```
                      type = "fit",
                      se.fit = TRUE)[["yhat"]], na.rm = TRUE)

rm(tmpdf0)

# Standard error of ATET
#-----------------------

atet.grad <- numDeriv::jacobian(func = function(z) {

  yhat1 <- aldvmm.pred(par   = z,
                       X     = X1,
                       y     = rep(0, nrow(X1[[1]])),
                       psi   = fit$psi,
                       ncmp  = fit$k,
                       dist  = fit$dist,
                       lcoef = fit$label$lcoef,
                       lcmp  = fit$label$lcmp,
                       lcpar = fit$label$lcpar)[["yhat"]]

  yhat0 <- aldvmm.pred(par   = z,
                       X     = X0,
                       y     = rep(0, nrow(X0[[1]])),
                       psi   = fit$psi,
                       ncmp  = fit$k,
                       dist  = fit$dist,
                       lcoef = fit$label$lcoef,
                       lcmp  = fit$label$lcmp,
                       lcpar = fit$label$lcpar)[["yhat"]]

  mean(yhat1 - yhat0, na.rm = TRUE)

},
x = fit$coef)

se.atet <- sqrt(atet.grad %*% fit$cov %*% t(atet.grad))

# Summarize
#----------

out <- data.frame(atet = mean1 - mean0,
                  se = se.atet,
                  z = (mean1 - mean0) / se.atet)
out$p <- 2*stats::pnorm(abs(out$z), lower.tail = FALSE)
out$ul <- out$atet + stats::qnorm((1 + 0.95)/2) * out$se
out$ll <- out$atet - stats::qnorm((1 + 0.95)/2) * out$se

print(out)
```

Example calculation of robust and clustered standard errors

```
# Create cluster indicator
#--------------------------

df$grp <- as.factor(round(0.5 + runif(nrow(df)) * 5, 0))

# Fit model
#----------

formula <- eq5d ~ hr | 1

fit <- aldvmm(formula,
              data = df,
              psi = c(-0.594, 0.883))

# Calculate robust and clustered standard errors
#-----------------------------------------------

vc1 <- sandwich::sandwich(fit)
vc2 <- sandwich::vcovCL(fit, cluster = ~ grp)
vc3 <- sandwich::vcovPL(fit, cluster = ~ grp)
vc4 <- sandwich::vcovHAC(fit, cluster = ~ grp)
vc5 <- sandwich::vcovBS(fit)
vc6 <- sandwich::vcovBS(fit, cluster = ~ grp)

# Calculate test statistics
#--------------------------

lmtest::coeftest(fit)
lmtest::coeftest(fit, vcov = vc1)
lmtest::coeftest(fit, vcov = vc2)
lmtest::coeftest(fit, vcov = vc3)
lmtest::coeftest(fit, vcov = vc4)
lmtest::coeftest(fit, vcov = vc5)
lmtest::coeftest(fit, vcov = vc6)
```

Covariance matrices across optimization methods

Covariance matrices were incomplete or missing entirely (`FALSE`) in multiple optimization approaches (table 12)

Table 12: Covariance matrix by optimization method

	Nelder-Mead	BFGS	CG	L-BFGS-B	nlminb	Rcgmin	Rvmmin	hjn
zero	FALSE	TRUE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE
random	FALSE	TRUE	FALSE	FALSE	TRUE	TRUE	FALSE	FALSE
constant	FALSE	TRUE	TRUE	FALSE	TRUE	TRUE	FALSE	TRUE
sann	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE

Modified Hosmer-Lemeshow test

```
# Number of percentiles
ngroup <- 10

# Extract expected values and residuals
yhat <- fit1_all[["zero"]][["Nelder-Mead"]][["pred"]][["yhat"]]
res <- fit1_all[["zero"]][["Nelder-Mead"]][["pred"]][["res"]]

# Make groups
group <- as.numeric(cut(yhat, breaks = ngroup), na.rm = TRUE)

# Auxiliary regression
aux <- stats::lm(res ~ factor(group))

# Data set of predictions from auxiliary regressions
newdf <- data.frame(group = unique(group)[order(unique(group))])
predict <- predict(aux,
                   newdata = newdf,
                   se.fit = TRUE,
                   interval = 'confidence',
                   level = 0.95)

plotdat <- as.data.frame(rbind(
  cbind(group = newdf$group,
        outcome = "mean",
        value = predict$fit[ , 'fit']),
  cbind(group = newdf$group,
        outcome = "ll",
        value = predict$fit[ , 'lwr']),
  cbind(group = newdf$group,
        outcome = "ul",
        value = predict$fit[ , 'upr'])
))

# Make plot
plot <- ggplot2::ggplot(plotdat, aes(x = factor(as.numeric(group)),
                                     y = as.numeric(value),
                                     group = factor(outcome))) +
  geom_line(aes(linetype = factor(outcome)))
```

Results of modified Hosmer-Lemeshow test

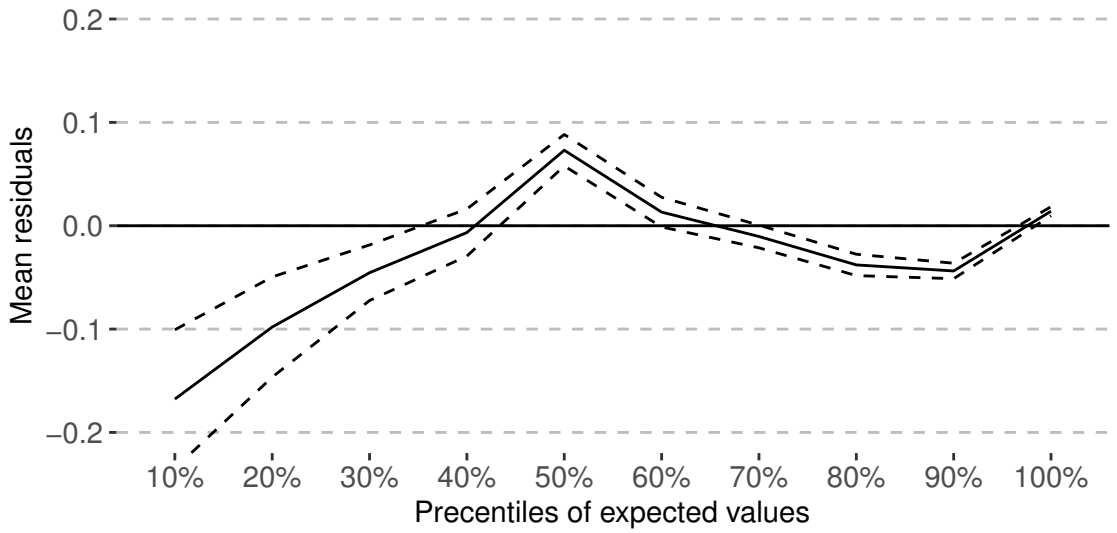

Figure 7: Mean residuals over deciles of expected values, "Nelder-Mead" with "zero" starting values

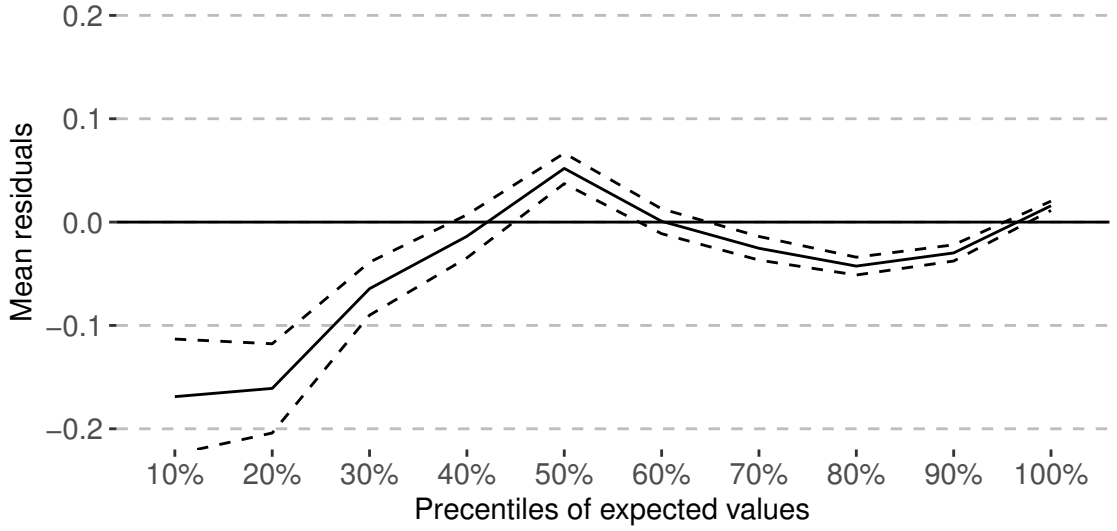

Figure 8: Mean residuals over deciles of expected values, "nlminb" with "zero" starting values

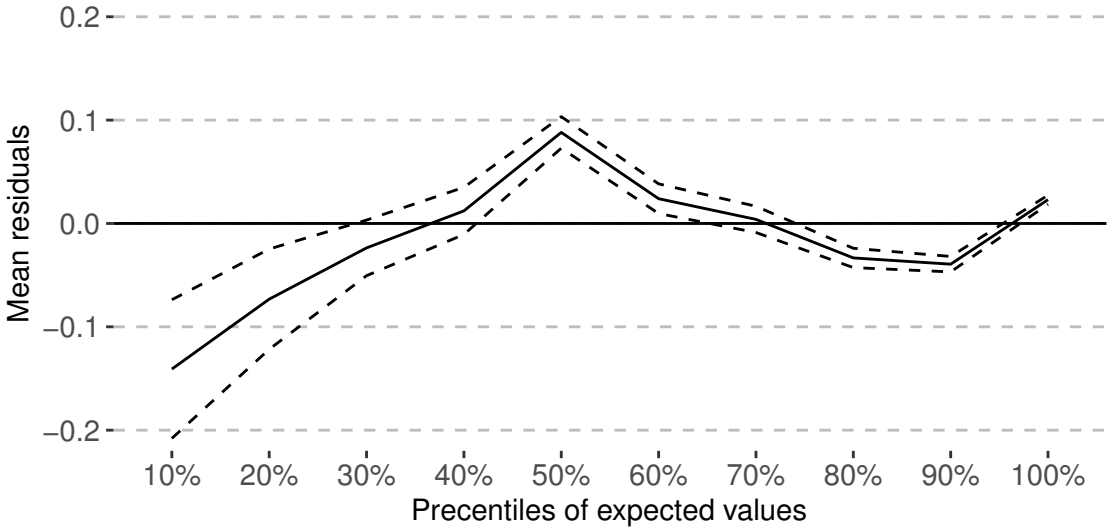

Figure 9: Mean residuals over deciles of expected values, "hjn" with "zero" starting values

R code for the estimation of reference cases

```
# (1) Reference case 1 with default optimization settings
fit1_default <- aldvmm::aldvmm(eq5d ~ hr | 1,
                               data = df,
                               psi = c(0.883, -0.594),
                               ncmp = 2,
                               init.method = "zero",
                               optim.method = "nlminb")

# (2) Reference case 1 with user-defined initial values and constraints on parameters
init <- c(0,    0,   0,   0,    0,    0,    0.7283)
lo   <- c(-Inf, -Inf, -3,  -Inf, -Inf, -3, -Inf)
hi   <- c(Inf,  Inf,  Inf, Inf,  Inf,  Inf,  Inf)

fit1_cstr <- aldvmm::aldvmm(eq5d ~ hr | 1,
                            data = df,
                            psi = c(0.883, -0.594),
                            ncmp = 2,
                            init.est = init,
                            init.lo = lo,
                            init.hi = hi)

# (3) Reference case 1 with initial values from constant-only model
fit1_const <- aldvmm::aldvmm(eq5d ~ hr | 1,
                             data = df,
                             psi = c(0.883, -0.594),
                             ncmp = 2,
                             init.method = "constant",
                             optim.method = "nlminb")

# (4) Reference case 2 with user-defined initial values.
init <- c(-.40293118, .30502755, .22614716, .14801581, -.70755741, 0,
          -1.2632051, -2.4541401)

fit2 <- aldvmm::aldvmm(eq5d ~ hr | hr,
                       data = df,
                       psi = c(0.883, -0.594),
                       ncmp = 2,
                       init.est = init,
                       optim.method = "nlminb")
```

STATA® code for the estimation of reference cases

```
* (1) Reference case 1
aldvmm eq5d hr, ncomponents(2)

* (2) Reference case 1 with constraints
matrix  input a = (0, 0, 0, 0, 0, 0, 0.7283)
constraint 1 [Comp_2]:hr10 = 0
constraint 2 [Comp_2]:_cons = 100
constraint 3 [lns_2]:_cons = 1e-30
aldvmm eq5d hr, ncomp(2) from(a) c(1 2 3)

* (3) Reference case 1 initital values from constant-only model
aldvmm eq5d hr, ncomp(2) inim(cons)

* (4) Reference case 2 user-defined initial values
matrix  input start = (.14801581, .22614716, .30502755, -.40293118, 0,
                       -.70755741, -2.4541401, -1.2632051)
aldvmm eq5d hr, ncomp(2) prob(hr) from(start)
```