UNIVERSITÄT
BIELEFELD
Fakultät für
Wirtschaftswissenschaften

On the initialization of multinomial probit models

Lennart Oelschläger

Bielefeld University
Department of Business Administration and Economics
Econometrics Group of Prof. Dr. Bauer

July 16, 2021

R nlm

Description

This function carries out a minimization of the function `f` using a Newton-type algorithm. See the references for details.

Usage

```
nlm(f, p, ..., hessian = FALSE, typsize = rep(1, length(p)),
    fscale = 1, print.level = 0, ndigit = 12, gradtol = 1e-6,
    stepmax = max(1000 * sqrt(sum((p/typsize)^2)), 1000),
    steptol = 1e-6, iterlim = 100, check.analyticals = TRUE)
```

Arguments

`f`
the function to be minimized, returning a single numeric value. This should be a function with first arg
arguments specified by the ... argument.

If the function value has an attribute called `gradient` or both `gradient` and `hessian` attributes, these wil
Otherwise, numerical derivatives are used. deriv returns a function with suitable `gradient` attribute an

`p`
starting parameter values for the minimization.

?

The multinomial probit model

$$\overbrace{\begin{pmatrix} X_{11} & \cdots & X_{1P} \\ \vdots & \ddots & \vdots \\ X_{J1} & \cdots & X_{JP} \end{pmatrix}}^{X}$$

The multinomial probit model

$$\overbrace{\begin{pmatrix} X_{11} & \dots & X_{1P} \\ \vdots & \ddots & \vdots \\ X_{J1} & \dots & X_{JP} \end{pmatrix}}^{X} \overbrace{\begin{pmatrix} \beta_1 \\ \vdots \\ \beta_P \end{pmatrix}}^{\beta}$$

The multinomial probit model

$$\overbrace{\begin{pmatrix} X_{11} & \dots & X_{1P} \\ \vdots & \ddots & \vdots \\ X_{J1} & \dots & X_{JP} \end{pmatrix}}^{X} \overbrace{\begin{pmatrix} \beta_1 \\ \vdots \\ \beta_P \end{pmatrix}}^{\beta} + \overbrace{\begin{pmatrix} \epsilon_1 \\ \vdots \\ \epsilon_J \end{pmatrix}}^{\epsilon}$$

The multinomial probit model

$$\overbrace{\begin{pmatrix} U_1 \\ \vdots \\ U_J \end{pmatrix}}^{U} = \overbrace{\begin{pmatrix} X_{11} & \dots & X_{1P} \\ \vdots & \ddots & \vdots \\ X_{J1} & \dots & X_{JP} \end{pmatrix}}^{X} \overbrace{\begin{pmatrix} \beta_1 \\ \vdots \\ \beta_P \end{pmatrix}}^{\beta} + \overbrace{\begin{pmatrix} \epsilon_1 \\ \vdots \\ \epsilon_J \end{pmatrix}}^{\epsilon}$$

The multinomial probit model

$$\overbrace{\begin{pmatrix} U_1 \\ \vdots \\ U_J \end{pmatrix}}^{U} = \overbrace{\begin{pmatrix} X_{11} & \dots & X_{1P} \\ \vdots & \ddots & \vdots \\ X_{J1} & \dots & X_{JP} \end{pmatrix}}^{X} \overbrace{\begin{pmatrix} \beta_1 \\ \vdots \\ \beta_P \end{pmatrix}}^{\beta} + \overbrace{\begin{pmatrix} \epsilon_1 \\ \vdots \\ \epsilon_J \end{pmatrix}}^{\epsilon}$$

$$y = \arg\max U$$

The multinomial probit model

$$\overbrace{\begin{pmatrix} U_1 \\ \vdots \\ U_J \end{pmatrix}}^{U} = \overbrace{\begin{pmatrix} X_{11} & \dots & X_{1P} \\ \vdots & \ddots & \vdots \\ X_{J1} & \dots & X_{JP} \end{pmatrix}}^{X} \overbrace{\begin{pmatrix} \beta_1 \\ \vdots \\ \beta_P \end{pmatrix}}^{\beta} + \overbrace{\begin{pmatrix} \epsilon_1 \\ \vdots \\ \epsilon_J \end{pmatrix}}^{\epsilon}$$

$$y = \arg\max U$$

$$\epsilon = L \underbrace{\eta}_{\sim N_J(0,I)} \sim N_J(0, LL' = \Sigma)$$

The multinomial probit model

$$\overbrace{\begin{pmatrix} U_1 \\ \vdots \\ U_J \end{pmatrix}}^{U} = \overbrace{\begin{pmatrix} X_{11} & \dots & X_{1P} \\ \vdots & \ddots & \vdots \\ X_{J1} & \dots & X_{JP} \end{pmatrix}}^{X} \overbrace{\begin{pmatrix} \beta_1 \\ \vdots \\ \beta_P \end{pmatrix}}^{\beta} + \overbrace{\begin{pmatrix} \epsilon_1 \\ \vdots \\ \epsilon_J \end{pmatrix}}^{\epsilon}$$

$$y = \arg\max U$$

$$\epsilon = L \underbrace{\eta}_{\sim N_J(0,I)} \sim N_J(0, LL' = \Sigma)$$

$$\beta = b + O \underbrace{\eta}_{\sim N_P(0,I)} \sim N_P(b, OO' = \Omega)$$

Level normalization

$$\Delta_J U = \Delta_J X\beta + \Delta_J \epsilon, \qquad \Delta_J = \begin{pmatrix} I_{J-1} & -1 \end{pmatrix} \in \mathbb{R}^{(J-1)\times J}$$

$$y = \begin{cases} i, & U_i = \max \Delta_J U > 0, i = 1, \ldots, J-1 \\ J, & \Delta_J U < 0 \end{cases}$$

$$\Delta_J \epsilon \sim N_{J-1}(0, \Delta_J L L' \Delta_J')$$

Level normalization

$$\Delta_J U = \Delta_J X\beta + \Delta_J \epsilon, \qquad \Delta_J = \begin{pmatrix} I_{J-1} & -1 \end{pmatrix} \in \mathbb{R}^{(J-1)\times J}$$

$$y = \begin{cases} i, & U_i = \max \Delta_J U > 0, i = 1, \dots, J-1 \\ J, & \Delta_J U < 0 \end{cases}$$

$$\Delta_J \epsilon \sim N_{J-1}(0, \Delta_J L L' \Delta_J')$$

Scale normalization

$$(\Delta_J L)_{11} = 1$$

Number of parameters to estimate

alternatives	covariates	$\#b$	$\#O$	$\#\Delta_J L$	total
J	P	P	$P \cdot (P+1)/2$	$(J-1) \cdot J/2 - 1$	
2	2	2	3	0	5
10	10	10	55	44	109

Number of parameters to estimate

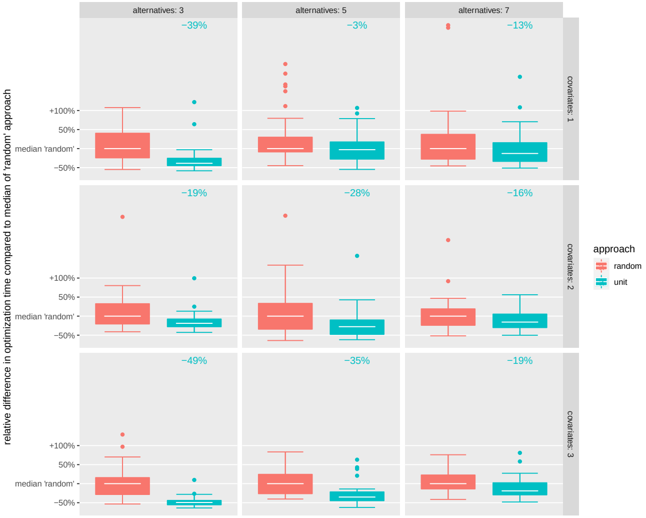
alternatives	covariates	$\#b$	$\#O$	$\#\Delta_J L$	total
J	P	P	$P \cdot (P+1)/2$	$(J-1) \cdot J/2 - 1$	
2	2	2	3	0	5
10	10	10	55	44	109

Log-likelihood

$$\log L(y) = \sum_{n,t,j} 1(y_{nt} = j) \overbrace{\Phi_{J-1}(-\Delta_j X_{nt} b \mid 0; \Delta_j (X_{nt} \Omega X_{nt}' + \Sigma) \Delta_j')}^{\Pr(y_{nt}=j)}$$

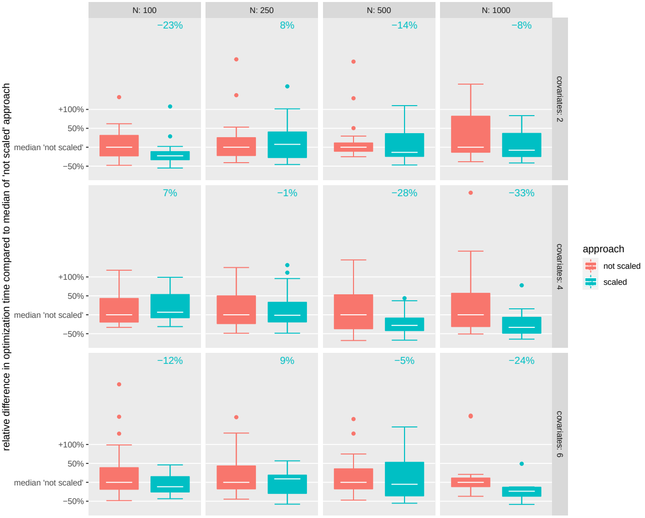
Approach: Unit

- Idea: Choose initial parameters s.t. $b = 0$, $\Omega = I$ and $\Sigma = I$
- Question: Better than random initialization?
- Simulation setting:
  - $N = 50$, $T = 10$
  - random covariates: $1, 2, 3$
  - alternatives: $3, 5, 7$


alternatives: 3
alternatives: 5
alternatives: 7
covariates: 1
covariates: 2
covariates: 3
−39%
−3%
−13%
−19%
−28%
−16%
−49%
−35%
−19%
+100%
50%
median 'random'
−50%
relative difference in optimization time compared to median of 'random' approach
approach
random
unit

Approach: Scaling

- Idea: Different ranges of covariate values may hinder optimization (e.g. 112 minutes travel time, EUR 5 travel cost)
- Question: Does standardization improve optimization time?
- Simulation setting:
  - $T = 10$, alternatives: 3
  - N: 100, 250, 500, 1000
  - random covariates: $2, 4, 6$
  - scale difference: $U[1, 10]$
  - use same random initial guesses, but mind the scales


N: 100
N: 250
N: 500
N: 1000
−23%
8%
−14%
−8%
7%
−1%
−28%
−33%
−12%
9%
−5%
−24%
+100%
50%
median 'not scaled'
−50%
covariates: 2
covariates: 4
covariates: 6
relative difference in optimization time compared to median of 'not scaled' approach
approach
not scaled
scaled

Approach: Subsample

- Idea: Estimate the model on a subsample based on
  - random subsampling (rs)
  - k-means (km)
  - model-based clustering (mc)
- Simulation setting:
  - $T = 10$, alternatives: 3, random covariates: 3
  - N: $100, 250, 500, 1000$
  - subsample proportion: $0.1, 0.25, 0.5$


N: 100
N: 250
N: 500
N: 1000
−7% 24% 3%
−20% −25% −18%
−19% −38% −46%
−69% −63% −58%
−35% −19%
−29% −37% −39%
−30% −22% −49%
−50% −28% −40%
−9% 2% 10%
−2% −5% −34%
−22% 5% −5%
−37% −19% −35%
subsample: 0.1
subsample: 0.25
subsample: 0.5
relative difference in optimization time compared to median of 'full' approach
+100%
50%
median 'full'
−50%
approach
full
km
mc
rs

Approach: Alternating optimization

- Idea: Alternating estimation of parameter groups
  - *b* separately (ao_b)
  - *b* and variances separately (ao_b_var)
  - *b*, variances and covariances separately (ao_full)
- Simulation setting:
  - $N = 50$, $T = 10$
  - random covariates: $1, 2, 3$
  - alternatives: $3, 5, 7$

UNIVERSITÄT BIELEFELD
Fakultät für Wirtschaftswissenschaften

Thank you!

Please let me know:

- To what extent is initialization an issue for your models?
- How do you initialize?