


OPSR: A Package for Estimating Ordinal Probit Switching Regression Models in R

Daniel Heimgartner 
ETH Zürich

Xinyi Wang 
MIT Boston

Abstract

When treatment is endogenous, selection bias might arise if unobserved factors simultaneously influence both the selection and outcome process. A possible cure in the case of cross-sectional data is to explicitly account for this error correlation and estimate the covariance matrix of the two processes. This is known as endogenous switching regression. The R package **OPSR** introduced in this article provides an easy-to-use, fast and memory efficient interface to ordinal probit switching regression, accounting for self-selection into an ordinal treatment. It handles log-transformed outcomes which need special consideration when computing conditional expectations and thus treatment effects.

Keywords: ordinal probit switching regression, endogenous switching regression, Heckman selection, selection bias, treatment effect, R.

1. Introduction

The goal of the program evaluation literature is to estimate the effect of a treatment program (e.g., a new policy, technology, medical treatment, or agricultural practice) on an outcome. To evaluate such a program, the “treated” are compared to the “untreated”. In an experimental setting, the treatment can be assigned by the researcher. However, in an observational setting, the treatment is not always exogenously prescribed but rather self-selected. This gives rise to a selection bias when unobserved factors influencing the treatment adoption also influence the outcome (also known as *selection on unobservables*). Simple group comparison no longer yield an unbiased estimate of the treatment effect. In more technical terms, the counterfactual outcome of the treated (“if they had not been treated”) does not necessarily correspond to the factual outcome of the untreated. For example, cyclists riding without a helmet (the “untreated”) might have a risk-seeking tendency. We therefore potentially overestimate the benefit of wearing a helmet if we compare the accident (severity) rate of the two groups. Risk-seeking is not readily measured and it is easy to imagine that it becomes part of the error in applied research and thus leading cause of a selection bias.

To properly account for the selection bias, various techniques exist, both for longitudinal and cross-sectional data. In the first case, difference in differences is a widely adopted measure. In the latter case, instrumental variables, matching propensity scores, regression-discontinuity design, and the endogenous switching regression model have been applied (Wang and Mokhtarian 2024). The latter method is particularly well-suited to correct for selection on unobservables (unlike other methods which only address and correct for selection on ob-

servables).

The seminal work by Heckman (1979) proposed a two-part model to address the selection bias that often occurs when modelling a continuous outcome which is only observable for a subpopulation. A very nice exposition of this model is given in Cameron and Trivedi (2005, Chapter 16). The classical Heckman model consists of a probit equation and continuous outcome equation. A natural extension is then switching regression, where the population is partitioned into different groups (regimes) and separate parameters are estimated for the continuous outcome process of each group. This model is originally known as the Roy model (Cameron and Trivedi 2005) or Tobit 5 model (Amemiya 1985). These classical models (the Tobit models for truncated, censored or interval data and their extensions) are implemented in various environments for statistical computing and in R's (R Core Team 2017) `sampleSelection` package (Toomet and Henningsen 2008).

Many different variants can then be derived by either placing different distributional assumptions on the errors and/or how the latent process manifests into observed outcomes (i.e., the dependent variables can be of various types, such as binary, ordinal, censored, or continuous) more generally known as conditional mixed-process (CMP) models. CMP models comprise a broad family involving two or more equations featuring a joint error distribution assumed to be multivariate normal. The Stata (StataCorp 2023) command `cmp` (Roodman 2011) can fit such models. The variant at the heart of this paper is an ordinal probit switching regression (OPSR) model, with an ordered treatment and continuous outcome. Throughout the text we use the convention that OPSR refers to the general methodology, while **OPSR** refers specifically to the package.

OPSR is available as a Stata command, `heckman` (Chiburis and Lokshin 2007), which however, does not allow distinct specifications for the continuous outcome processes (i.e., the same explanatory variables must be used for all treatment groups). The relatively new R package `switchSelection` (Potanin 2024) allows to estimate multivariate and multinomial sample selection and endogenous switching models with multiple outcomes. These models are systems of ordinal, continuous and multinomial equations and thus nest OPSR as a special case.

OPSR is tailored to one particular method, easy to use (understand, extend and maintain), fast and memory efficient. It handles log-transformed continuous outcomes which need special consideration for the computation of conditional expectations. It obeys to R's implicit modeling conventions (by extending the established generics such as `summary()`, `predict()`, `update()`, `anova()` among others) and produces production-grade output tables. This work generalizes the learnings from Wang and Mokhtarian (2024) and makes the OPSR methodology readily available. The mathematical notation presented here translates to code almost verbatim which hopefully serves a pedagogical purpose for the curious reader.

2. Model and software

In the following, we outline the ordinal probit switching regression model as well as list all the key formulas underlying the software implementation. **OPSR** follows the R-typical formula interface to a workhorse fitter function. Its architecture is detailed after the mathematical part.

As alluded, OPSR is a two-step model: One process governs the ordinal outcome and separate processes (for each ordinal outcome) govern the continuous outcomes. The ordinal outcome

can also be thought of as a regime or treatment. In the subsequent exposition, we will refer to the two processes as *selection* and *outcome* process.

We borrow the notation from Wang and Mokhtarian (2024) where also all the derivations are detailed. For a similar exhibition, Chiburis and Lokshin (2007) can be consulted. Individual subscripts are suppressed throughout, for simplicity.

Let \mathcal{Z} be a latent propensity governing the selection outcome

$$\mathcal{Z} = \mathbf{W}\boldsymbol{\gamma} + \epsilon, \quad (1)$$

where \mathbf{W} represents the vector of attributes of an individual, $\boldsymbol{\gamma}$ is the corresponding vector of parameters and $\epsilon \sim \mathcal{N}(0, 1)$ a normally distributed error term.

As \mathcal{Z} increases and passes some unknown but estimable thresholds, we move up from one ordinal treatment to the next higher level

$$Z = j \quad \text{if } \kappa_{j-1} < \mathcal{Z} \leq \kappa_j, \quad (2)$$

where Z is the observed ordinal selection variable, $j = 1, \dots, J$ indexes the ordinal levels of Z , and κ_j are the thresholds (with $\kappa_0 = -\infty$ and $\kappa_J = \infty$). Hence, there are $J - 1$ thresholds to be estimated. The probability that an individual self-selects into treatment group j is

$$\begin{aligned} \text{P}[Z = j] &= \text{P}[\kappa_{j-1} < \mathcal{Z} \leq \kappa_j] \\ &= \text{P}[\kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma} < \epsilon \leq \kappa_j - \mathbf{W}\boldsymbol{\gamma}] \\ &= \Phi(\kappa_j - \mathbf{W}\boldsymbol{\gamma}) - \Phi(\kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma}). \end{aligned} \quad (3)$$

The outcome model for the j^{th} treatment group is expressed as

$$y_j = \mathbf{X}_j\boldsymbol{\beta}_j + \eta_j, \quad (4)$$

where y_j is the observed continuous outcome, \mathbf{X}_j the vector of observed explanatory variables associated with the j^{th} outcome model, $\boldsymbol{\beta}_j$ is the vector of associated parameters, and $\eta_j \sim \mathcal{N}(0, \sigma^2)$ is a normally distributed error term. At this point it should be noted that \mathbf{X}_j and \mathbf{W} may share some explanatory variables but not all, due to identification problems otherwise (Chiburis and Lokshin 2007).

The key assumption of OPSR is now that the errors of the selection and outcome models are jointly multivariate normally distributed

$$\begin{pmatrix} \epsilon \\ \eta_1 \\ \vdots \\ \eta_j \\ \vdots \\ \eta_J \end{pmatrix} \sim \mathcal{N} \left(\begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & \rho_1\sigma_1 & \cdots & \rho_j\sigma_j & \cdots & \rho_J\sigma_J \\ \rho_1\sigma_1 & \sigma_1^2 & & & & \\ \vdots & & \ddots & & & \\ \rho_j\sigma_j & & & \sigma_j^2 & & \\ \vdots & & & & \ddots & \\ \rho_J\sigma_J & & & & & \sigma_J^2 \end{pmatrix} \right), \quad (5)$$

where ρ_j represents the correlation between the errors of the selection model (ϵ) and the j^{th} outcome model (η_j). If the covariance matrix should be diagonal (i.e., no error correlation), no selection-bias exists and the selection and outcome models can be estimated separately.

As shown in Wang and Mokhtarian (2024), the log-likelihood of observing all individuals self-selecting into treatment j and choosing continuous outcome y_j can be expressed as

$$\begin{aligned} \ell(\theta | \mathbf{W}, \mathbf{X}_j) &= \sum_{j=1}^J \sum_{\{j\}} \left\{ \ln \left[\frac{1}{\sigma_j} \phi \left(\frac{y_j - \mathbf{X}_j \beta_j}{\sigma_j} \right) \right] + \right. \\ &\left. \ln \left[\Phi \left(\frac{\sigma_j(\kappa_j - \mathbf{W}\boldsymbol{\gamma}) - \rho_j(y_j - \mathbf{X}_j \beta_j)}{\sigma_j \sqrt{1 - \rho_j^2}} \right) - \Phi \left(\frac{\sigma_j(\kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma}) - \rho_j(y_j - \mathbf{X}_j \beta_j)}{\sigma_j \sqrt{1 - \rho_j^2}} \right) \right] \right\} \quad (6) \end{aligned}$$

where $\sum_{\{j\}}$ means the summation of all the cases belonging to the j^{th} selection outcome, $\phi(\cdot)$ and $\Phi(\cdot)$ are the density and cumulative distribution function of the standard normal distribution.

The conditional expectation can be expressed as

$$\begin{aligned} \mathbb{E}[y_j | Z = j] &= \mathbf{X}_j \beta_j + \mathbb{E}[\eta_j | \kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma} < \epsilon \leq \kappa_j - \mathbf{W}\boldsymbol{\gamma}] \\ &= \mathbf{X}_j \beta_j - \rho_j \sigma_j \frac{\phi(\kappa_j - \mathbf{W}\boldsymbol{\gamma}) - \phi(\kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma})}{\Phi(\kappa_j - \mathbf{W}\boldsymbol{\gamma}) - \Phi(\kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma})}, \quad (7) \end{aligned}$$

where the fraction is the ordered probit switching regression model counterpart to the inverse Mills ratio (IMR) term of a binary switching regression model. We immediately see, that regressing \mathbf{X}_j on y_j leads to an omitted variable bias if $\rho_j \neq 0$ which is the root cause of the selection bias. However, the IMR can be pre-computed based on an ordinal probit model and then included in the second stage regression, which describes the Heckman correction (Heckman 1979). It should be warned, that since the Heckman two-step procedure includes an estimate in the second step regression, the resulting OLS standard errors and heteroskedasticity-robust standard errors are incorrect (Greene 2002).

To obtain unbiased treatment effects, we must further evaluate the ‘‘counterfactual outcome’’, which reflects the expected outcome under a counterfactual treatment (i.e., for $j' \neq j$)

$$\begin{aligned} \mathbb{E}[y_{j'} | Z = j] &= \mathbf{X}_{j'} \beta_{j'} + \mathbb{E}[\eta_{j'} | \kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma} < \epsilon \leq \kappa_j - \mathbf{W}\boldsymbol{\gamma}] \\ &= \mathbf{X}_{j'} \beta_{j'} - \rho_{j'} \sigma_{j'} \frac{\phi(\kappa_j - \mathbf{W}\boldsymbol{\gamma}) - \phi(\kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma})}{\Phi(\kappa_j - \mathbf{W}\boldsymbol{\gamma}) - \Phi(\kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma})}. \quad (8) \end{aligned}$$

As it is usual to log-transform the continuous outcome in regression analysis, we have to note, that in such cases the Equations 7-8 provide the conditional expectation of the log-transformed outcome. Therefore, we need to back-transform $\ln(y_j + 1)$ which yields

$$\mathbb{E}[y_j | Z = j] = \exp \left(\mathbf{X}_j \beta_j + \frac{\sigma_j^2}{2} \right) \left[\frac{\Phi(\kappa_j - \mathbf{W}\boldsymbol{\gamma} - \rho_j \sigma_j) - \Phi(\kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma} - \rho_j \sigma_j)}{\Phi(\kappa_j - \mathbf{W}\boldsymbol{\gamma}) - \Phi(\kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma})} \right] - 1 \quad (9)$$

for the factual case, and

$$\mathbb{E}[y_{j'} | Z = j] = \exp \left(\mathbf{X}_{j'} \beta_{j'} + \frac{\sigma_{j'}^2}{2} \right) \left[\frac{\Phi(\kappa_j - \mathbf{W}\boldsymbol{\gamma} - \rho_{j'} \sigma_{j'}) - \Phi(\kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma} - \rho_{j'} \sigma_{j'})}{\Phi(\kappa_j - \mathbf{W}\boldsymbol{\gamma}) - \Phi(\kappa_{j-1} - \mathbf{W}\boldsymbol{\gamma})} \right] - 1 \quad (10)$$

for the counterfactual case (Wang and Mokhtarian 2024).

This concludes the mathematical treatment and we briefly outline OPSR’s architecture which can be conceptualized as follows:

- We provide the usual formula interface to specify a model. To allow for multiple parts and multiple responses, we rely on the **Formula** package (Zeileis and Croissant 2010).
- After parsing the formula object, checking the user inputs and computing the model matrices, the Heckman two-step estimator is called in `opsr_2step()` to generate reasonable starting values.
- These are then passed together with the data to the basic computation engine `opsr.fit()`. The main estimates are retrieved using maximum likelihood estimation by passing the log-likelihood function `loglik_cpp()` (Equation 6) to `maxLik()` from the **maxLik** package (Henningsen and Toomet 2011).
- All the above calls are nested in the main interface `opsr()` which returns an object of class ‘opsr’. Several methods then exist to post-process this object as illustrated below.

The likelihood function `loglik_cpp()` is implemented in C++ using **Rcpp** (Eddelbuettel and Balamuta 2018) and relying on the data types provided by **RcppArmadillo** (Eddelbuettel and Sanderson 2014). Parallelization is available using **OpenMP**. This makes **OPSR** both fast and memory efficient (as data matrices are passed by reference).

3. Illustrations

We first illustrate how to specify a model using **Formula**’s extended syntax and simulated data. Then the main functionality of the package is demonstrated. We conclude this section by demonstrating some nuances, reproducing the core model of Wang and Mokhtarian (2024).

Let us simulate data from an OPSR process with three ordinal outcomes and distinct design matrices \mathbf{W} and \mathbf{X} (where $\mathbf{X} = \mathbf{X}_j \forall j$) by

```
R> sim_dat <- opsr_simulate()
R> dat <- sim_dat$data
R> head(dat)
```

	ys	yo	xs1	xs2	xo1	xo2
1	2	-0.5492	-0.702	-0.733	2.79248	-0.016
2	3	4.0957	-0.167	2.164	0.47481	-1.226
3	3	4.8617	0.533	1.641	0.00991	-1.524
4	2	1.1602	0.284	-1.175	0.19197	0.676
5	2	2.4321	-0.158	1.187	0.19337	1.178
6	2	0.0966	-1.236	0.935	1.46857	0.735

where `ys` is the selection outcome (or treatment group), `yo` the continuous outcome and `xs` respectively `xo` the corresponding explanatory variables.

Models are specified symbolically. A typical model has the form `ys | yo ~ terms_s | terms_o1 | terms_o2 | ...` where the `|` separates the two responses and process specifications. If the user wants to specify the same process for all continuous outcomes, two processes are enough (`ys | yo ~ terms_s | terms_o`). Hence the minimal `opsr()` interface call reads

```
R> fit <- opsr(ys | yo ~ xs1 + xs2 | xo1 + xo2, data = dat,
+   printLevel = 0)
```

where `printLevel = 0` omits working information during maximum likelihood iterations.

As usual, the fitter function does the bare minimum model estimation while inference is performed in a separate call

```
R> summary(fit)
```

Call:

```
opsr(formula = ys | yo ~ xs1 + xs2 | xo1 + xo2, data = dat, printLevel = 0)
```

Meta information:

BFGS maximization, 93 iterations

Return code 0: successful convergence

Runtime: 0.359 secs

Log-Likelihood: -1990

AIC: 4019

BIC: 4112

Number of regimes: 3

Number of observations: 1000 (176, 515, 309)

Estimated parameters: 19

Estimates:

	Estimate	Std. error	t value	Pr(> t)
kappa1	-2.0572	0.0960	-21.43	< 2e-16 ***
kappa2	1.0365	0.0701	14.79	< 2e-16 ***
s_xs1	1.0291	0.0576	17.87	< 2e-16 ***
s_xs2	1.5695	0.0760	20.65	< 2e-16 ***
o1_(Intercept)	0.8456	0.1045	8.09	6.0e-16 ***
o1_xo1	2.0776	0.0683	30.42	< 2e-16 ***
o1_xo2	1.1581	0.0649	17.84	< 2e-16 ***
o2_(Intercept)	0.9886	0.0487	20.28	< 2e-16 ***
o2_xo1	-0.9889	0.0505	-19.59	< 2e-16 ***
o2_xo2	1.5572	0.0465	33.50	< 2e-16 ***
o3_(Intercept)	1.0269	0.0806	12.74	< 2e-16 ***
o3_xo1	1.5376	0.0737	20.87	< 2e-16 ***
o3_xo2	-1.9998	0.0604	-33.10	< 2e-16 ***
sigma1	1.0055	0.0530	18.99	< 2e-16 ***
sigma2	1.1066	0.0366	30.27	< 2e-16 ***
sigma3	1.1195	0.0443	25.28	< 2e-16 ***
rho1	0.0222	0.1178	0.19	0.85039
rho2	0.4050	0.0665	6.09	1.2e-09 ***
rho3	0.2729	0.0815	3.35	0.00081 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Wald chi2 (null): 4982 on 8 DF, p-value: < 0

Wald chi2 (rho): 48 on 3 DF, p-value: < 0

The presentation of the model results is fairly standard and should not warrant further explanation with the following exceptions

1. The number of regimes along absolute counts are reported.
2. Coefficient names are based on the variable names as passed to the formula specification, except that "s_" is prepended to the selection coefficients, "o[0-9]_" to the outcome coefficients and the structural components "kappa", "sigma", "rho" (aligning with the letters used in Equation 6) are hard-coded (but can be over-written).
3. The coefficients table reports robust standard errors based on the sandwich covariance matrix as computed with help of the **sandwich** package (Zeileis 2006). `rob = FALSE` reports conventional standard errors.
4. Two Welch-tests are conducted. One, testing the null that all coefficients of explanatory variables are zero and two, testing the null that all error correlation coefficients (ρ) are zero. If the latter is rejected, selection bias is not an issue.

A useful benchmark is always the null model with structural parameters only. The null model can be derived from an 'opsr' model fit as follows

```
R> fit_null <- opsr_null_model(fit, printLevel = 0)
```

A model can be updated as usual

```
R> fit_intercept <- update(fit, . ~ . | 1)
```

where we have removed all the explanatory variables from the outcome processes.

Several models can be compared with a likelihood-ratio test using

```
R> anova(fit_null, fit_intercept, fit)
```

Likelihood Ratio Test

Model 1: ~Nullmodel

Model 2: ys | yo ~ xs1 + xs2 | 1

Model 3: ys | yo ~ xs1 + xs2 | xo1 + xo2

	logLik	Df	Test	Restrictions	Pr(>Chi)
1	-3308	8			
2	-2803	13	1011	5	<2e-16 ***
3	-1990	19	1625	6	<2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

If only a single object is passed, then the model is compared to the null model. If more than one object is specified a likelihood ratio test is conducted for each pair of neighboring models. As expected, both tests reject the null.

Models can be compared side-by-side using the **texreg** package (Leifeld 2013), which also allows the user to build production-grade tables as illustrated later.

```
R> texreg::screenreg(list(fit_null, fit_intercept, fit),
+   include.pseudoR2 = TRUE, include.R2 = TRUE, single.row = TRUE)
```

	Model 1	Model 2	Model 3
kappa1	-0.93 (0.05) ***	-2.06 (0.10) ***	-2.06 (0.10) ***
kappa2	0.50 (0.04) ***	1.03 (0.07) ***	1.04 (0.07) ***
sigma1	2.66 (0.14) ***	2.66 (0.13) ***	1.01 (0.05) ***
sigma2	2.18 (0.07) ***	2.19 (0.07) ***	1.11 (0.04) ***
sigma3	2.69 (0.11) ***	2.69 (0.11) ***	1.12 (0.04) ***
rho1		0.00 (0.12)	0.02 (0.12)
rho2		0.27 (0.07) ***	0.41 (0.07) ***
rho3		0.05 (0.10)	0.27 (0.08) ***
s_xs1		1.03 (0.06) ***	1.03 (0.06) ***
s_xs2		1.58 (0.08) ***	1.57 (0.08) ***
o1_(Intercept)	0.71 (0.21) ***	0.72 (0.31) *	0.85 (0.10) ***
o1_xo1			2.08 (0.07) ***
o1_xo2			1.16 (0.06) ***
o2_(Intercept)	0.92 (0.10) ***	0.97 (0.10) ***	0.99 (0.05) ***
o2_xo1			-0.99 (0.05) ***
o2_xo2			1.56 (0.05) ***
o3_(Intercept)	1.00 (0.15) ***	0.93 (0.20) ***	1.03 (0.08) ***
o3_xo1			1.54 (0.07) ***
o3_xo2			-2.00 (0.06) ***
AIC	6632.43	5631.68	4018.70
BIC	6671.69	5695.48	4111.94
Log Likelihood	-3308.21	-2802.84	-1990.35
Pseudo R ² (EL)	0.08	0.53	0.53
Pseudo R ² (MS)	-0.00	0.49	0.49
R ² (total)	0.00	0.01	0.81
R ² (1)	0.01	0.01	0.86
R ² (2)	0.00	0.03	0.76
R ² (3)	0.00	0.00	0.83
Num. obs.	1000	1000	1000

*** p < 0.001; ** p < 0.01; * p < 0.05

Finally, the key interest of an OPSR study almost certainly is the estimation of treatment

effects which relies on (counterfactual) conditional expectations as already noted in the mathematical exposition.

```
R> p1 <- predict(fit, group = 1, type = "response")
R> p2 <- predict(fit, group = 1, counterfact = 2, type = "response")
```

where `p1` is the result of applying Equation 7 and `p2` is the counterfactual outcome resulting from Equation 8. The following `type` arguments are available

- `type = "response"`: Predicts the continuous outcome according to the Equations referenced above.
- `type = "unlog-response"`: Predicts the back-transformed response if the continuous outcome was log-transformed according to Equations 9-10.
- `type = "prob"`: Returns the probability vector of belonging to `group`.
- `type = "mills"`: Returns the inverse Mills ratio.

Elements are `NA_real_` if the `group` does not correspond to the observed regime (selection outcome). This ensures consistent output length.

Now that the user understands the basic workflow, we illustrate some nuances by reproducing a key output of Wang and Mokhtarian (2024) where they investigate the treatment effect of telework on weekly vehicle miles driven. The data is attached, documented (`?telework_data`) and can be loaded by

```
R> data("telework_data", package = "OPSR")
```

A basic boxplot of the response variable against the three teleworking status is displayed in Figure 1. By simply looking at the data descriptively, we might prematurely conclude telework reduces vehicle miles driven. However, the whole value proposition of OPSR (and for models in general) is that we really are interested in a counterfactual. If the teleworkers self-select, the counterfactual is not simply the group average. More prosaically, if the usual teleworkers (UTW) would choose to be non-usual teleworkers (NUTW), they might travel more or less than the actual NUTWs.

■ daniehei: Maybe do this discussion and boxplot in Section 4 and omit here.

The final model specification reads

```
R> f <-
+   twing_status | vmd_ln ~
+   edu_2 + edu_3 + hhincome_2 + hhincome_3 + flex_work + work_fulltime +
+   twing_feasibility + att_proactivemode + att_procarowning + att_wif +
+   att_proteamwork + att_tw_effective_teamwork + att_tw_enthusiasm +
+   att_tw_location_flex |
+   female + age_mean + age_mean_sq + race_black + race_other + vehicle +
+   suburban + smalltown + rural + work_fulltime + att_prolargehouse +
+   att_procarowning + region_waa |
```

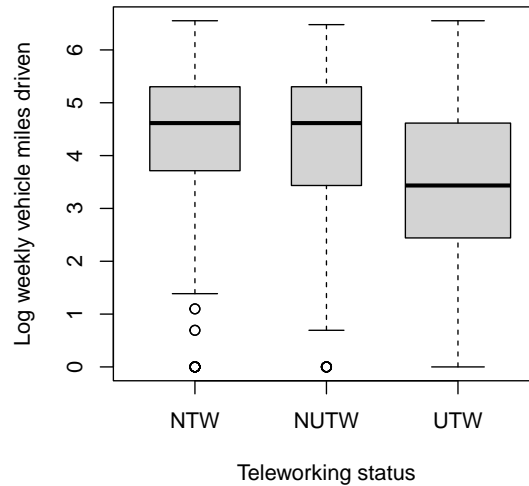


Figure 1: Log vehicle miles driven for different teleworking status.

```
+ edu_2 + edu_3 + suburban + smalltown + rural + work_fulltime +
+ att_prolargehouse + att_proactivemode + att_procarowning |
+ female + hhincome_2 + hhincome_3 + child + suburban + smalltown +
+ rural + att_procarowning + region_waa
```

and the model can be estimated by

```
R> start_default <- opsr(f, telework_data, .get2step = TRUE)
R> fit <- opsr(f, telework_data, start = start, method = "NM", iterlim = 50e3,
+           printLevel = 0)
```

where we demonstrate that

1. Default starting values as computed by the Heckman two-step procedure can be retrieved.
2. `start` values can be overridden (we have hidden the `start` vector here for brevity). If the user wishes to pass start values manually, some minimal conventions have to be followed as documented in `?opsr_check_start`.
3. Alternative maximization methods (here “Nelder-Mead”) can be used (as in the original paper).

With help of the `texreg` package, production-grade tables (in various output formats) can be generated with ease.

```
R> texreg::texreg(
+   fit, beside = TRUE, include.structural = FALSE,
```

	NTWer (535)	NUTWer (322)	UTWer (727)
Intercept	3.64 (0.27) ^{***}	2.49 (0.37) ^{***}	2.38 (0.26) ^{***}
Female	-0.21 (0.10) [*]		-0.36 (0.11) ^{***}
Age	0.01 (0.00) [*]		
Age squared	-0.00 (0.00)		
Race (ref: white)			
Black	-0.40 (0.24)		
Other races	-0.06 (0.18)		
Education (ref: high school or less)			
Some college		0.15 (0.33)	
Bachelor's degree or higher		0.62 (0.31) [*]	
Annual household income (ref: less than \$50,000)			
\$50,000 to \$99,999			0.47 (0.23) [*]
\$100,000 or more			0.31 (0.23)
Number of children			0.18 (0.06) ^{**}
Number of vehicles	0.12 (0.05) [*]		
Residential location (ref: urban)			
Suburban	0.07 (0.15)	0.45 (0.17) ^{**}	0.28 (0.14) [*]
Small town	0.47 (0.18) ^{**}	0.19 (0.29)	0.29 (0.28)
Rural	0.60 (0.23) ^{**}	0.81 (0.31) ^{**}	0.88 (0.34) ^{**}
Full time worker	0.45 (0.13) ^{***}	0.69 (0.17) ^{***}	
Attitudes			
Pro-large-house	0.18 (0.05) ^{***}	0.18 (0.08) [*]	
Pro-car-owning	0.14 (0.07) [*]	0.16 (0.09)	0.25 (0.06) ^{***}
Pro-active-mode		-0.18 (0.08) [*]	
AIC	7191.35	7191.35	7191.35
BIC	7491.94	7491.94	7491.94
Log Likelihood	-3539.67	-3539.67	-3539.67
R ² (total)	0.24	0.24	0.24
R ² (1)	0.28	0.28	0.28
R ² (2)	0.23	0.23	0.23
R ² (3)	0.21	0.21	0.21
Num. obs.	1584	1584	1584

*** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$

Table 1: Replica of Wang and Mokhtarian (2024), Table 3.

```
+ include.selection = FALSE, include.R2 = TRUE,
+ custom.model.names = custom.model.names,
+ custom.coef.names = custom.coef.names,
+ single.row = TRUE,
+ reorder.coef = reorder.coef, groups = groups, scalebox = 0.86,
+ booktabs = TRUE, dcolumn = TRUE, use.packages = FALSE, float.pos = "t!",
+ caption = "Replica of \\cite{Wang+Mokhtarian:2024}, Table 3.",
+ label = "tab:wang-replica"
+ )
```

Dot arguments (...) passed to `texreg()` (or similar functions) are forwarded to a S4 method `extract()` which extracts the variables of interest from a model fit (see also `?extract.opsr`). We demonstrate here that

1. The structural coefficients (κ , σ and ρ) and coefficients belonging to the selection component can be omitted (`include.structural = FALSE`, `include.selection = FALSE`).
2. The model components can be printed side-by-side (`beside = TRUE`).
3. Additional goodness-of-fit indicators can be included (`include.R2 = TRUE`).
4. The output formatting can be controlled flexibly, by reordering, renaming and grouping coefficients (the fiddly but trivial details are hidden here for brevity).

4. Case study

daniehei: TU+ case study. Intended to demonstrate how to use **OPSR** in a real-world example. Don't comment on what a function does but interpret the output substantively.

5. Summary and discussion

Computational details

The results in this paper were obtained using R 4.4.2 with the **OPSR** 0.1.2.9001 package. R itself and all packages used are available from the Comprehensive R Archive Network (CRAN) at <https://CRAN.R-project.org/>.

Acknowledgments

References

- Amemiya T (1985). *Advanced Econometrics*. Harvard University Press, Cambridge, Massachusetts.
- Cameron AC, Trivedi PK (2005). *Microeconometrics: Methods and Applications*. Cambridge University Press, Cambridge.
- Chiburis R, Lokshin M (2007). “Maximum Likelihood and Two-Step Estimation of Ordered-Probit Selection Model.” *The Stata Journal*, **7**(2), 167–182. doi:10.1177/1536867X0700700202.
- Eddelbuettel D, Balamuta JJ (2018). “Extending R with C++: A Brief Introduction to **Rcpp**.” *The American Statistician*, **72**(1), 28–36. doi:10.1080/00031305.2017.1375990.
- Eddelbuettel D, Sanderson C (2014). “**RcppArmadillo**: Accelerating R with high-performance C++ linear algebra.” *Computational Statistics and Data Analysis*, **71**, 1054–1063. doi:10.1016/j.csda.2013.02.005.

- Greene WH (2002). *LIMDEP Version 8.0 Econometric Modeling Guide, vol. 2*. Econometric Software, Plainview, New York.
- Heckman J (1979). “Sample Selection Bias as a Specification Error.” *Econometrica*, **47**(1), 153–161. doi:10.2307/1912352.
- Henningsen A, Toomet O (2011). “**maxLik**: A package for maximum likelihood estimation in R.” *Computational Statistics*, **26**(3), 443–458. doi:10.1007/s00180-010-0217-1.
- Leifeld P (2013). “**texreg**: Conversion of Statistical Model Output in R to L^AT_EX and HTML Tables.” *Journal of Statistical Software*, **55**(8), 1–24. doi:10.18637/jss.v055.i08.
- Potantin B (2024). **switchSelection**: *Endogenous Switching and Sample Selection Regression Models*. R package version 2.0.0, URL <https://CRAN.R-project.org/package=switchSelection>.
- R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL <https://www.R-project.org/>.
- Roodman D (2011). “Fitting Fully Observed Recursive Mixed-Process Models with **cmp**.” *The Stata Journal*, **11**(2), 159–206. doi:10.1177/1536867X110110020.
- StataCorp (2023). *Stata Statistical Software*. StataCorp LLC, College Station, Texas. URL <https://www.stata.com/>.
- Toomet O, Henningsen A (2008). “Sample Selection Models in R: Package **sampleSelection**.” *Journal of Statistical Software*, **27**(7), 1–23. doi:10.18637/jss.v027.i07.
- Wang X, Mokhtarian PL (2024). “Examining the Treatment Effect of Teleworking on Vehicle-Miles Driven: Applying an Ordered Probit Selection Model and Incorporating the Role of Travel Stress.” *Transportation Research Part A*, **186**, 104072. doi:10.1016/j.tra.2024.104072.
- Zeileis A (2006). “Object-Oriented Computation of Sandwich Estimators.” *Journal of Statistical Software*, **16**(9), 1–16. doi:10.18637/jss.v016.i09.
- Zeileis A, Croissant Y (2010). “Extended Model Formulas in R: Multiple Parts and Multiple Responses.” *Journal of Statistical Software*, **34**(1), 1–13. doi:10.18637/jss.v034.i01.

Affiliation:

Daniel Heimgartner
Institute for Transport Planning and Systems
Eidgenössische Technische Hochschule Zürich
IFW C 46.1
Haldeneggsteig 4
8092 Zürich, Switzerland
E-mail: daniel.heimgartner@ivt.baug.ethz.ch

Xinyi Wang
Department of Urban Studies and Planning
Massachusetts Institute of Technology
105 Massachusetts Avenue
Cambridge, MA 02139
E-mail: xinyi174@mit.edu