sleev: Semiparametric Likelihood Estimation with Errors in Variables

Jiangmei Xiong Sarah C. Lotspeich Joey B. Sherrill
Gustavo Amorim Bryan E. Shepherd Ran Tao

Data with measurement error in the outcome, covariates, or both are not uncommon, particularly with the increased use of routinely collected data for biomedical research. In settings with error-prone data, two-phase studies, where researchers validate a subsample of study data, can be used to obtain unbiased estimates. The sieve maximum likelihood estimator (SMLE), which combines the error-prone data on all records with the validated data on a subsample, is a highly efficient and robust estimator to analyze such two-phase data. However, given their complexity, a computationally efficient and user-friendly tool is needed to obtain SMLEs. This vignette introduces the `sleev` package for making semiparametric likelihood-based inference using SMLEs for error-prone two-phase data in settings with binary and continuous outcomes. Functions from this package can be used to analyze data with error-prone responses and covariates. Various examples are presented to provide users with guidance in handling different types of variables. To demonstrate the use of the functions in practice, they are applied to a two-phase dataset simulated to represent data obtained from the electronic health records of an HIV clinic.

1. Introduction

Routinely collected data are being used more frequently in biomedical research. For example, data extracted from electronic health records have been used in numerous studies as a cost-effective resource to obtain information on a large number of people. However, these data tend to be error-prone, often across multiple variables, and using these data without correcting for their error-prone nature could lead to biased estimates and misleading research findings (Duan et al. 2016). To avoid invalid study results, routinely collected data may undergo validation, in which trained experts carefully verify and extract data elements. However, it is usually only feasible to validate data for a subset of records or variables. After partial validation, researchers have two types of data: (i) error-prone pre-validation data for all records (phase

one data) and (ii) error-free validated data on a subset of records (phase two data). For analyses, the goal is then to combine the two types of data to obtain estimates that have low bias and are as efficient (i.e., have the smallest variance) as possible.

Building off of the measurement error and missing data literature, there are several types of approaches for combining such two-phase data with errors, including design-based methods (e.g., inverse-probability weighted estimators (Horvitz and Thompson 1952) and generalized raking estimators (Deville, Särndal, and Sautory 1993; Oh et al. 2021)) and model-based methods (e.g., maximum likelihood estimation (Carroll et al. 2006; Tang et al. 2015) and multiple imputation (Little and Rubin 1986; Cole, Chu, and Greenland 2006; Giganti et al. 2020)). Both design- and model-based estimators require the missing at random assumption for unbiased estimation, i.e., conditional on observed data, those records to be validated are assumed to be selected through random sampling. Design-based estimators also require that the probability of being selected for validation is non-zero for all records, whereas no such positivity assumption is required for model-based estimators. Because they make no model assumptions on the error mechanism, design-based estimators tend to be more robust but less efficient than model-based estimators (Bang and Robins 2005; Amorim et al. 2021).

A robust class of model-based estimators, the sieve maximum likelihood estimators (SMLEs), have recently been developed to analyze two-phase data with errors in both the outcome and covariates (Tao et al. 2021; Lotspeich et al. 2022). The SMLEs are semiparametric and robust because they avoid making parametric assumptions on the nuisance models of the error terms, and, as full-likelihood estimators, they remain highly efficient. Hence, they provide a nice balance between robustness and efficiency. Still, in practice these estimators can be difficult to implement, as they involve approximating nuisance conditional densities using B-splines (Schumaker 2007) and then maximizing the semiparametric likelihood via a sophisticated EM algorithm (Dempster, Laird, and Rubin 1977; Tao, Zeng, and Lin 2017).

In this vignette, we introduce the `sleev` package, which computes the SMLEs for linear and logistic regressions using partially-validated, error-prone data. The `sleev` package incorporates error-prone data on all records plus validated data on a subset of records to obtain efficient and robust estimates of regression parameters in a user-friendly manner. This vignette describes the SMLE method (Sections 2 and 3) and demonstrates the features of the `sleev` package and the application of functions in the package through a detailed illustration using simulated HIV data (Section 4).

2. Sieve maximum likelihood estimators for linear regression

Suppose that we want to fit a standard linear regression model for a continuous outcome Y and vector of covariates $\boldsymbol{X}$: $Y = \alpha + \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{X} + \epsilon$, where ϵ follows a normal distribution with mean zero and variance σ^2. Our goal is to obtain estimates of $\boldsymbol{\theta} = (\alpha, \boldsymbol{\beta}^{\mathrm{T}}, \sigma^2)^{\mathrm{T}}$. When we have error-prone data, Y and $\boldsymbol{X}$ are unobserved except for a subset of subjects whose records are validated. For the subjects whose records are not validated (the majority), only the error-prone

outcome $Y^* = Y + W$ and covariates $\boldsymbol{X}^* = \boldsymbol{X} + \boldsymbol{U}$ are observed in place of Y and $\boldsymbol{X}$, where W and $\boldsymbol{U}$ are the additive errors for the outcome and covariates, respectively. It is assumed that the measurement errors W and $\boldsymbol{U}$ are independent of ϵ. However, W and $\boldsymbol{U}$ can be correlated. Note that $\boldsymbol{X}^*$ can also include error-free covariates $\boldsymbol{Z}$, which can be incorporated as $\boldsymbol{X}^* = (\boldsymbol{X}_{\boldsymbol{0}}^{*\mathrm{T}}, \boldsymbol{Z}^{\mathrm{T}})^{\mathrm{T}}$, where $\boldsymbol{X}_{\boldsymbol{0}}^*$ denotes error-prone covariates. However, for simplicity, we do not include the expression of error-free covariates $\boldsymbol{Z}$ throughout Sections 2 and 3. With errors in our data, a naive regression analysis using error-prone variables Y^* and $\boldsymbol{X}^*$ could render misleading results (Fuller 2009).

We assume that the joint density of the complete data $(Y^*, \boldsymbol{X}^*, W, \boldsymbol{U})$ takes the form

$$\begin{aligned} P(Y^*, \boldsymbol{X}^*, W, \boldsymbol{U}) &= P(Y^*|\boldsymbol{X}^*, W, \boldsymbol{U}) P(W, \boldsymbol{U}|\boldsymbol{X}^*) P(\boldsymbol{X}^*) \\ &= P_{\boldsymbol{\theta}}(Y|\boldsymbol{X}) P(W, \boldsymbol{U}|\boldsymbol{X}^*) P(\boldsymbol{X}^*), \end{aligned}$$

where $P(\cdot)$ and $P(\cdot|\cdot)$ denote density and conditional density functions, respectively. Specifically, $P_{\boldsymbol{\theta}}(Y|\boldsymbol{X})$ then refers to the conditional density function of the linear regression model $Y = \alpha + \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{X} + \epsilon$. Denote the validation indicator variable by V, with $V = 1$ indicating that a record was validated and $V = 0$ otherwise. For records that do not undergo validation, their measurement errors $(W, \boldsymbol{U})$ are missing. Therefore, the contributions of these unvalidated subjects to the log-likelihood can be obtained by integrating out W and $\boldsymbol{U}$.

Let $(Y_i^*, \boldsymbol{X}_i^*, W_i, \boldsymbol{U}_i, V_i, Y_i, \boldsymbol{X}_i)$ for $i = 1, \ldots, n$ denote independent and identically distributed realizations of $(Y^*, \boldsymbol{X}^*, W, \boldsymbol{U}, V, Y, \boldsymbol{X})$ in a sample of n subjects. Then, the observed-data log-likelihood is proportional to

$$\begin{aligned} &\sum_{i=1}^{n} V_i \{\log P_{\boldsymbol{\theta}}(Y_i|\boldsymbol{X}_i) + \log P(W_i, \boldsymbol{U}_i|\boldsymbol{X}_i^*)\} \\ &+ \sum_{i=1}^{n} (1 - V_i) \log \left\{ \int \int P_{\boldsymbol{\theta}}(Y_i^* - w|\boldsymbol{X}_i^* - \boldsymbol{u}) P(w, \boldsymbol{u}|\boldsymbol{X}_i^*) \mathrm{d}w \mathrm{d}\boldsymbol{u} \right\}. \end{aligned} \tag{1}$$

Note that $P(\boldsymbol{X}^*)$ is left out, because the error-prone covariates are fully observed and thus $P(\boldsymbol{X}^*)$ can simply be estimated empirically.

Because the measurement error model, $P(W_i, \boldsymbol{U}_i|\boldsymbol{X}_i^*)$, is often unknown in practice, we prefer to leave it unspecified, and use nonparametric maximum likelihood estimation (NPMLE) to estimate it. NPMLE estimates $P(W, \boldsymbol{U}|\boldsymbol{X}^* = \boldsymbol{x}^*)$ with the m distinct observed $(W, \boldsymbol{U})$ values, $\{(w_1, \boldsymbol{u}_1), ..., (w_m, \boldsymbol{u}_m)\}$, from the validated subset. Because NPMLE estimates $P(W, \boldsymbol{U}|\boldsymbol{X}^* = \boldsymbol{x}^*)$ with the empirical density, it will not be applicable when $\boldsymbol{X}^*$ contains continuous elements, where only a small number of observations on $(W, \boldsymbol{U})$ will be associated with each $\boldsymbol{X}^* = \boldsymbol{x}^*$. In this situation, we estimate $P(W_i, \boldsymbol{U}_i|\boldsymbol{X}_i^*)$ with B-spline sieves.

Specifically, we approximate $P(w, \boldsymbol{u}|\boldsymbol{X}_i^*)$ and $\log P(W_i, \boldsymbol{U}_i|\boldsymbol{X}_i^*)$ by $\sum_{k=1}^{m} I(w = w_k, \boldsymbol{u} = \boldsymbol{u}_k) \sum_{j=1}^{s_n} B_j^q(\boldsymbol{X}_i^*) p_{kj}$ and $\sum_{k=1}^{m} I(W_i = w_k, \boldsymbol{U}_i = \boldsymbol{u}_k) \sum_{j=1}^{s_n} B_j^q(\boldsymbol{X}_i^*) \log p_{kj}$, respectively, where $B_j^q(\boldsymbol{X}_i^*)$ is the jth B-spline basis function of order q evaluated at $\boldsymbol{X}_i^*$, s_n is the dimension of

the B-spline basis, and p_{kj} is the coefficient associated with $B_j^q(\boldsymbol{X}_i^*)$ and $(w_k, \boldsymbol{u}_k)$. We note that the p_{kj} coefficients need to satisfy the constraints $\sum_{k=1}^m p_{kj} = 1$ and $p_{kj} \geq 0$ since they approximate conditional densities. The log-likelihood in expression (1) is now approximated by

$$\begin{aligned}&\sum_{i=1}^n V_i \left\{ \log P_{\boldsymbol{\theta}}(Y_i|\boldsymbol{X}_i) + \sum_{k=1}^m I(W_i = w_k, \boldsymbol{U}_i = \boldsymbol{u}_k) \sum_{j=1}^{s_n} B_j^q(\boldsymbol{X}_i^*) \log p_{kj} \right\} \\ &+ \sum_{i=1}^n (1 - V_i) \log \left\{ \sum_{k=1}^m P_{\boldsymbol{\theta}}(Y_i^* - w_k|\boldsymbol{X}_i^* - \boldsymbol{u}_k) \sum_{j=1}^{s_n} B_j^q(\boldsymbol{X}_i^*) p_{kj} \right\}.\end{aligned} \tag{2}$$

The maximization of expression (2) is carried out through an EM algorithm to find the SMLEs $\hat{\boldsymbol{\theta}}$ and $\hat{p}_{kj}$. The covariance matrix of the SMLE $\hat{\boldsymbol{\theta}}$ is obtained through the method of profile likelihood (Murphy and Van der Vaart 2000). Full details on the SMLE method for linear regression with error-prone data, including its theoretical properties, can be found in Tao et al. (2021).

3. Sieve maximum likelihood estimators for logistic regression

For a binary outcome Y, we fit a logistic regression model instead:

$$P_{\boldsymbol{\theta}}(Y = 1|\boldsymbol{X}) = \left[1 + \exp\{-(\alpha + \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{X})\}\right]^{-1}$$

with parameters $\boldsymbol{\theta} = (\alpha, \boldsymbol{\beta}^{\mathrm{T}})^{\mathrm{T}}$. The joint density of $(Y^*, \boldsymbol{X}^*, Y, \boldsymbol{X})$ is

$$\begin{aligned}P(Y^*, \boldsymbol{X}^*, Y, \boldsymbol{X}) &= P(Y^*|\boldsymbol{X}^*, Y, \boldsymbol{X}) P(Y|\boldsymbol{X}, \boldsymbol{X}^*) P(\boldsymbol{X}|\boldsymbol{X}^*) P(\boldsymbol{X}^*) \\ &= P(Y^*|\boldsymbol{X}^*, Y, \boldsymbol{X}) P_{\boldsymbol{\theta}}(Y|\boldsymbol{X}) P(\boldsymbol{X}|\boldsymbol{X}^*) P(\boldsymbol{X}^*),\end{aligned}$$

where $P(Y|\boldsymbol{X}, \boldsymbol{X}^*) = P_{\boldsymbol{\theta}}(Y|\boldsymbol{X})$ follows from the assumption that Y and $\boldsymbol{X}^*$ are conditionally independent given $\boldsymbol{X}$ (i.e., $\boldsymbol{X}^*$ is a surrogate for $\boldsymbol{X}$). Similar to the linear regression case, the observed-data log-likelihood takes the form

$$\begin{aligned}&\sum_{i=1}^n V_i \{\log P_{\boldsymbol{\theta}}(Y_i|\boldsymbol{X}_i) + \log P(Y_i^*|\boldsymbol{X}_i^*, Y_i, \boldsymbol{X}_i) + \log P(\boldsymbol{X}_i|\boldsymbol{X}_i^*)\} \\ &+ \sum_{i=1}^n (1 - V_i) \log \left\{ \sum_{y=0}^1 \int \log P_{\boldsymbol{\theta}}(y|\boldsymbol{x}) P(Y_i^*|\boldsymbol{X}_i^*, y, \boldsymbol{x}) P(\boldsymbol{x}|\boldsymbol{X}_i^*) \mathrm{d}\boldsymbol{x} \right\}.\end{aligned} \tag{3}$$

We fit $P(Y^*|\boldsymbol{X}^*, Y, \boldsymbol{X})$ with an additional logistic regression model $P_{\boldsymbol{\gamma}}(Y^*|\boldsymbol{X}^*, Y, \boldsymbol{X})$ with $\boldsymbol{\gamma}$ denoting its parameters. We estimate $P(\boldsymbol{X}|\boldsymbol{X}^*)$ with NPMLE when $\boldsymbol{X}^*$ is discrete, and use

a B-spline approximation when $\boldsymbol{X}^*$ contains continuous components. Specifically, we approximate $P(\boldsymbol{x}|\boldsymbol{X}^*)$ and $\log P(\boldsymbol{X}_i|\boldsymbol{X}_i^*)$ in expression (3) by $\sum_{k=1}^{m} \mathrm{I}(\boldsymbol{x} = \boldsymbol{x}_k) \sum_{j=1}^{s_n} B_j^q(\boldsymbol{X}_i^*) p_{kj}$ and $\sum_{k=1}^{m} \mathrm{I}(\boldsymbol{X}_i = \boldsymbol{x}_k) \sum_{j=1}^{s_n} B_j^q(\boldsymbol{X}_i^*) \log p_{kj}$, respectively. Consequently, expression (3) can be approximated by

$$\begin{aligned} &\sum_{i=1}^{n} V_i \left\{ \log P_{\boldsymbol{\theta}}(Y_i|\boldsymbol{X}_i) + \log P_{\boldsymbol{\gamma}}(Y_i^*|\boldsymbol{X}_i^*, Y_i, \boldsymbol{X}_i) + \sum_{k=1}^{m} \mathrm{I}(\boldsymbol{X}_i = \boldsymbol{x}_k) \sum_{j=1}^{s_n} B_j^q(\boldsymbol{X}_i^*) \log p_{kj} \right\} \\ &+ \sum_{i=1}^{n} (1 - V_i) \log \left\{ \sum_{y=0}^{1} \sum_{k=1}^{m} P_{\boldsymbol{\theta}}(y|\boldsymbol{x}_k) P_{\boldsymbol{\gamma}}(Y_i^*|\boldsymbol{X}_i^*, y, \boldsymbol{x}_k) \mathrm{I}(\boldsymbol{x} = \boldsymbol{x}_k) \sum_{j=1}^{s_n} B_j^q(\boldsymbol{X}_i^*) p_{kj} \right\}. \end{aligned} \tag{4}$$

Similar to the linear regression case, we maximize expression (4) through an EM algorithm to obtain the SMLEs $\hat{\boldsymbol{\theta}}$, $\hat{\boldsymbol{\gamma}}$, and $\hat{\boldsymbol{p}}_{kj}$. Then, we use the method of profile likelihood to estimate the covariance of $\hat{\boldsymbol{\theta}}$. More details on the SMLEs, including the theoretical properties on the SMLEs for logistic regression with measurement error, can be found in Lotspeich et al. (2022).

4. Case study with mock data

4.1 Overview and installation of the `sleev` R package

The `sleev` package provides a user-friendly way to obtain the SMLEs for linear and logistic regression and their standard errors. The package can be installed through CRAN.

```
install.packages("sleev")
library("sleev")
```

The `sleev` package includes two main functions: `linear2ph()` and `logistic2ph()`, to fit linear and logistic regressions, respectively, under two-phase sampling with an error-prone outcome and covariates. The input arguments are similar for the two functions and listed in Table 1. In addition to the arguments for error-prone and error-free outcome and covariates, the user needs to specify the B-spline matrix $B_j^q(\boldsymbol{X}_i^*)$ to be used in the estimation of the error densities.

Table 1: Main arguments for the `linear2ph()` and `logistic2ph()` functions

Argument	Description
y_unval	Column name of unvalidated outcome in the input dataset.
y	Column name of validated outcome in the input dataset. `NA`s in the input will be counted as individuals not selected in phase two.

Argument	Description
x_unval	Column names of unvalidated covariates in the input dataset.
x	Column names of validated covariates in the input dataset. `NA`s in the input will be counted as individuals not selected in phase two.
z	Column names of error-free covariates in the input dataset.
b_spline	Column names of the B-spline basis in the input dataset.
data	Dataset
hn_scale	Scale of the perturbation constant in the variance estimation via the method of profile likelihood. The default is `1`.
se	Standard errors of the parameter estimators will be estimated when set to `TRUE`. The default is `TRUE`.
tol	Convergence criterion. The default is `0.0001`.
max_iter	Maximum number of iterations in the EM algorithm. The default is `1000`.
verbose	Print analysis details when set to `TRUE`. The default is `FALSE`.

We now illustrate how to obtain SMLEs using the `sleev` package. First, we briefly describe the data that will be used, and then we show how to fit a linear regression model in the presence of errors in both the outcome and covariates using the `linear2ph()` function. We will explain how to choose the dimension of the B-spline basis, s_n. We will also demonstrate two ways to handle the situation when there is more than one continuous covariate in the model, where the dimension of the B-spline basis increases exponentially with the number of continuous covariates. Finally, we will briefly demonstrate the use of `logistic2ph()` to fit a logistic regression model, which is largely analogous to the use of `linear2ph()`.

4.2 Overview of data

We illustrate the usage of the functions in the `sleev` package with a dataset constructed to mimic data from the Vanderbilt Comprehensive Care Clinic (VCCC) patient records from Giganti et al. (2020). People living with HIV who were admitted to the clinic between 1998 and 2011 are collected by VCCC. The VCCC cohort records are fully validated, meaning all observations have gold standard measures, making it an ideal dataset for illustrating the SMLEs. The VCCC dataset contains complete data for all 2087 patients; we use this number as the sample size for our simulated dataset.

The simulated VCCC data were created by sampling from distributions that are similar to the original dataset. For our illustrations, we assume that 835 (40%) patient records were validated. We selected the 835 patients through simple random sampling and hid the validated values for the remaining 1252 patients by setting them as missing. Table 2 describes the variables to be used in subsequent analyses.

Table 2: Data dictionary for the `mock.vccc` dataset

Name	Status	Type	Description
`ID`	error-free		Patient ID
`VL_unval`	error-prone	continuous	Viral load (VL) at antiretroviral therapy (ART) initiation
`VL_val`	validated	continuous	
`ADE_unval`	error-prone	binary	Had an AIDS-defining event (ADE) within one year of ART initiation: 1 - yes, 0 - no
`ADE_val`	validated	binary	
`CD4_unval`	error-prone	continuous	CD4 count at ART initiation
`CD4_val`	validated	continuous	
`Prior_ART`	error-free	binary	Experienced ART before enrollment: 1 - yes, 0 - no
`Sex`	error-free	binary	Sex at birth of patient: 1 - male, 0 - female
`Age`	error-free	continuous	Age of patient

The dataset is included in the `sleev` package, and it can be loaded by

```
data("mock.vccc")
```

Table 3 displays the first six rows of the VCCC dataset. Notice that patients $1, 3$, and 5 have `NA` listed for the `VL_val`, `ADE_val`, and `CD4_val` variables, which means that these patients were not selected for data validation. In contrast, patients $2, 4$, and 6 had their data validated, so these variables were not missing. For example, from the data validation, patient 2 had a viral load (VL) of 907 copies/mL^3 and no AIDS-defining events within one year of antiretroviral therapy (ART) initiation confirmed. However, the patient's validated CD4 at ART initiation was found to be 114 cells/mm^3, not 36.

Because of skewness, we often transform both CD4 and VL. In our analysis, CD4 was divided by 10 and square-root transformed and VL was $\log_{10}$-transformed.

```
mock.vccc$CD4_val_sq10 <- sqrt(mock.vccc$CD4_val / 10)
mock.vccc$CD4_unval_sq10 <- sqrt(mock.vccc$CD4_unval / 10)
mock.vccc$VL_val_l10 <- log10(mock.vccc$VL_val)
mock.vccc$VL_unval_l10 <- log10(mock.vccc$VL_unval)
```

Table 3: First six patients in the `mock.vccc` dataset

ID	VL_unval	VL_val	ADE_unval	ADE_val	CD4_unval	CD4_val	Prior_ART	Sex	Age
1	1358	NA	0	NA	465	NA	0	1	33
2	907	907	0	0	36	114	1	1	25
3	2284	NA	0	NA	263	NA	1	1	35
4	25473	25473	0	0	244	235	0	0	65
5	19	NA	0	NA	263	NA	1	1	37
6	36662	36662	0	0	30	30	1	0	47

4.3 Linear regression with mock data

We first illustrate the use of the `linear2ph()` function by fitting a linear regression model with CD4 count at ART (Y) regressed on VL at ART initiation (X), adjusting for sex at birth (Z). Both CD4 and VL are error-prone, partially-validated variables, whereas sex is error-free.

4.3.1 Setting up the B-spline basis for modeling the error mechanisms

To obtain the SMLEs, we first need to set up the B-spline basis for the covariates `VL_unval_l10` (the transformed error-prone VL variable from phase one) and `Sex`. The `spline2ph()` function in `sleev` packages can set up the B-spline basis, and combine it with the data input for the final analysis. The column names of the B-spline basis are set as `bs[num]`, where `[num]` is the index of the B-spline basis column.

Here, we use a cubic B-spline basis with the `degree = 3` argument in our call to the `spline2ph()` function. The size of the basis s_n is set to be 20, specified through the `size = 20` argument. The B-spline basis is set up separately for the two `Sex` groups by specifying argument `group = "Sex"`. The size of the B-spline basis assigned to each group is proportional to the sample size of that group. Stratifying the error distribution by sex allows the errors in `VL_unval_l10` to be heterogeneous between males and females. The described B-spline basis is constructed as follows.

```
sn <- 20
b_spline_names <- paste0("bs", 1:sn)
data.linear <- spline2ph(x = "VL_unval_l10", data = mock.vccc, size = sn,
                         degree = 3,  bs_names = b_spline_names,
                         group = "Sex")
```

Alternatively, if the investigator has prior knowledge that the errors in `VL_unval_l10` are likely to be homogeneous, one may fit a simpler model by not stratifying the B-spline basis by `Sex`. In this case we would not specify the `group` argument in this function.

4.3.2 Model fitting and result interpretation

Having constructed the B-spline basis, the SMLEs can be obtained by running the `linear2ph()` function on `data.linear`.

```
start.time <- Sys.time()
res_linear <- linear2ph(y_unval = "CD4_unval_sq10", y = "CD4_val_sq10",
                        x_unval = "VL_unval_l10", x = "VL_val_l10",
                        z = "Sex", b_spline = b_spline_names,
                        data = data.linear,  hn_scale = 1, se = TRUE,
                        tol = 1e-04, max_iter = 1000, verbose = FALSE)
paste0("Run time: ", round(difftime(Sys.time(), start.time,
                                    units = "secs"), 3), " sec")
```

```
[1] "Run time: 2.607 sec"
```

The `linear2ph()` function returns an object of class `linear2ph`, denoted by `res_linear` in the code above. An object of class `linear2ph` is a list containing five slots: `coefficients`, `covariance`, `sigma`, `converge`, and `converge_cov`. We should first check if the EM algorithms for estimating the regression coefficients and their covariance matrix converged by checking if `res_linear$converge` and `res_linear$converge_conv`, respectively, are `TRUE`.

```
c(res_linear$converge, res_linear$converge_cov)
```

```
[1] TRUE TRUE
```

The `coef()` function takes an object of class `linear2ph` and gives the regression coefficient estimates.

```
(res_linear_coef <- coef(res_linear))
```

```
 Intercept VL_val_l10         Sex
     4.821     -0.141       0.273
```

Similar to interpreting the output from a standard linear model (i.e., fitted with `lm()`), the output here indicates that, after adjusting for `Sex`, for every one-unit increase in the transformed viral load at ART initiation, there is expected to be a 0.141 decrease in the transformed CD4 count at ART initiation. The transformed CD4 count can be transformed back to the original scale for interpretation. For example, a female patient with a VL of 1000 copies/mL^3 is expected to have a CD4 count of approximately 193 cells/mm^3. The expected CD4 count

of this female patient is lower than a female patient with a viral load of 100 copies/mL^3, whose expected CD4 count is approximately 206 cells/mm^3. It is expected that the average transformed CD4 count for males is 0.273 higher than that for females, adjusting for VL.

The `summary()` function takes an object of class `linear2ph` and further returns the standard errors, z-statistics and p-values, alongside the point estimates of the regression coefficients. Based on the p-values, both VL and sex are associated with CD4 count at the 0.05 significance level.

```
summary(res_linear)
```

```
Call:
linear2ph(y_unval = "CD4_unval_sq10", y = "CD4_val_sq10", x_unval = "VL_unval_l10",
    x = "VL_val_l10", z = "Sex", b_spline = b_spline_names, data = data.linear,
    hn_scale = 1, se = TRUE, tol = 1e-04, max_iter = 1000, verbose = FALSE)

Coefficients:
           Estimate     SE Statistic  p-value
Intercept     4.821 0.1587     30.39 0.000000
VL_val_l10   -0.141 0.0398     -3.55 0.000389
Sex           0.273 0.1089      2.51 0.012229
```

The `summary()` function also gives the covariance matrix, which can be used to calculate confidence intervals for the outcome variable for a subset of patients. For example, suppose we want to know the 95% confidence interval of the expected CD4 count for male patients with VL of 1200 copies/mL^3. The upper and lower bounds are

$$(\text{upper}, \text{lower}) = (\text{mean} - 1.96 * \sqrt{var(\text{mean})}, \text{mean} + 1.96 * \sqrt{var(\text{mean})})$$

First, we need to calculate the estimated mean transformed CD4 count for this patient group by $\text{mean} = \beta_0 + \beta_1 * \text{VL} + \beta_2 * \text{Sex}$

```
x.vec <- matrix(data = c(1, log10(1200), 1), ncol = 1) # set up data matrix
est.mean <- res_linear_coef %*% x.vec # calculate estimated mean
est.mean
```

```
     [,1]
[1,] 4.66
```

Then, we use the estimated covariance matrix to compute the variance of the linear combination `est.mean`. The formula for the linear combination is

$$var(\text{mean}) = var(\beta_0 + \beta_1 * \text{VL} + \beta_2 * \text{Sex}) = \begin{bmatrix} 1 & \text{VL} & \text{Sex} \end{bmatrix} [3 \times 3 \text{ covariance matrix}] \begin{bmatrix} 1 \\ \text{VL} \\ \text{Sex} \end{bmatrix}$$

```
res_linear_cov <- summary(res_linear)$covariance
est.cov <- t(x.vec) %*% res_linear_cov %*% x.vec # covariance of est.mean
```

The 95% confidence interval of CD4 count $(\text{cells}/10\text{mm}^3)^{1/2}$ for this group is therefore

```
est.mean + c(-1.96, 1.96) * sqrt(est.cov)
```

```
[1] 4.55 4.76
```

4.4 Choosing the B-spline basis through cross-validation

When constructing the B-spline basis to estimate error models, one needs to specify the order of the B-spline functions q and the size of the B-spline basis s_n. It is customary to use cubic splines ($q = 3$) in practice. Quadratic and linear splines are also permissible, especially when the correlation between the covariates and their measurement errors is expected to be modest. The optimal size of the B-spline basis can be selected through k-fold cross-validation with the `cv_linear2ph()` function, which works as follows:

1. The data are split into k folds.
2. The number of iterations is the same as the number of folds, k. In each of k iterations, one fold is held out, and the SMLEs are estimated using the remaining $k-1$ folds. Then, the log-likelihood function in the hold-out fold is predicted using the fitted SMLEs.
3. The average predicted log-likelihood across the k iterations is calculated as a summary of performance.

The size of the B-spline basis that yields the largest average predicted log-likelihood will be chosen for subsequent analysis. The following code shows an example of using `cv_linear2ph()` to select the desirable size of the B-spline basis in the `mock.vccc` dataset. The number of folds is set to be $k = 5$.

```
# set for reproducibility of fold assignment
set.seed(1)
# different B-spline sizes
sns <- c(15, 20, 25, 30, 35, 40)
# vector to hold mean log-likelihood and run time for each sn
```

```r
pred_loglike.1 <- run.time.secs <- rep(NA, length(sns))
# get number of rows of the dataset
n <- nrow(mock.vccc)
# specify number of folds in the cross validation
k <- 5
# calculate proportion of female patients in the dataset
sex_ratio <- sum(mock.vccc$Sex == 0) / n
for (i in 1:length(sns)) {
  # constructing B-spline basis using the same process as in Section 4.3.1
  sn <- sns[i]
  b_spline_names <- paste0("bs", 1:sn)
  data.sieve <- spline2ph(x = "VL_unval_l10", data = mock.vccc, size = sn,
                          degree = 3,  bs_names = b_spline_names,
                          group = "Sex")

  # cross validation, produce mean log-likelihood
  start.time <- Sys.time()
  res.1 <- cv_linear2ph(y = "CD4_val_sq10", y_unval = "CD4_unval_sq10",
                        x ="VL_val_l10", x_unval = "VL_unval_l10", z = "Sex",
                        b_spline = b_spline_names, data = data.sieve,
                        nfolds = k, max_iter = 2000, tol = 1e-04,
                        verbose = FALSE)
  # save run time
  run.time.secs[i] <- difftime(Sys.time(), start.time, units = "secs")
  # save mean log-likelihood result
  pred_loglike.1[i] <- res.1$avg_pred_loglik
}
```

The average predicted log-likelihoods and run time for the different s_n considered are:

```r
out <- data.frame(sns, pred_loglike.1, run.time.secs)
options(digits = 6)
shortest <- which.min(out$run.time.secs)
kable(out) %>%
  row_spec(shortest, background = "yellow")
```

sns	pred_loglike.1	run.time.secs
15	-919.862	13.5474
20	-919.355	19.1125
25	-920.191	13.0132

30	-919.915	16.0109
35	-920.914	24.1278
40	-920.684	17.9157

It can be seen that the model with $s_n = 20$ in the B-spline basis yields the highest average predicted log-likelihood, and is therefore chosen. We note that the average predicted log-likelihoods are fairly similar, indicating that the size of the B-spline basis does not impact the results very much in this dataset. This observation agrees with the results of Tao et al. (2021).

To confirm that there is negligible difference between the models fitted with different B-spline sizes in `mock.vccc`, we can compare the SMLEs with $s_n = 20$ and $s_n = 35$.

```
# same process as in Section 4.3, fit with sn = 35
sn.35 <- 35
b_spline_names <- paste0("bs", 1:sn.35)
data.sieve <- spline2ph(x = "VL_unval_l10", data = mock.vccc, size = sn.35,
                        degree = 3,  bs_names = b_spline_names,
                        group = "Sex")

start.time <- Sys.time()
fit.sn.35 <- linear2ph(y = "CD4_val_sq10", y_unval = "CD4_unval_sq10",
                       x = "VL_val_l10", x_unval = "VL_unval_l10", z = "Sex",
                       b_spline = b_spline_names, data = data.sieve,
                       hn_scale = 1, se = TRUE, tol = 1e-04,
                       max_iter = 1000, verbose = FALSE)
paste0("Run time: ", round(difftime(Sys.time(), start.time,
                                    units = "secs"), 3), " sec")
```

```
[1] "Run time: 9.731 sec"
```

```
# compare the coefficients to those from Section 4.3.2
summary(res_linear)$coefficients
```

```
            Estimate        SE Statistic     p-value
Intercept   4.820917 0.1586520  30.38673 0.000000000
VL_val_l10 -0.141317 0.0398341  -3.54764 0.000388705
Sex         0.272798 0.1088818   2.50545 0.012229410
```

```
summary(fit.sn.35)$coefficients
```

```
            Estimate       SE Statistic     p-value
Intercept   4.815033 0.159237  30.23811 0.000000000
VL_val_l10 -0.141407 0.040427  -3.49784 0.000469034
Sex         0.279226 0.109430   2.55163 0.010721977
```

The comparison shows that that the estimates, standard errors, z-statistics, and p-values of the parameters from the model with different B-spline sizes are very similar.

4.5 Example with two continuous covariates

In this section, we illustrate the use of the `linear2ph()` function with two continuous covariates using i) a bivariate B-spline basis and ii) a B-spline basis based on the first principle component (PC) of the covariates. Both approaches are reasonable, and they produce similar results in this example. The latter method is recommended for computational efficiency when there are more than two continuous covariates in the model. Suppose that we are fitting a model with CD4 count as the outcome and VL, age, and sex as covariates. This model is very similar to the model in Section 4.3, but with the addition of another error-free covariate age. Now, we have one binary and two continuous variables to be incorporated into the B-spline basis.

4.5.1 Bivariate B-spline

When there are two continuous covariates in the model, the B-spline basis is constructed from the tensor product of the one-dimensional B-spline bases for each variable. In this example, the two variables are VL and age, stratified by sex. Due to the curse of dimensionality, the choice of number of knots has more restrictions than when there is only one continuous covariate. For instance, in this example the smallest size for the one-dimensional cubic B-spline basis is 4. If we set one-dimensional s_n for each sex and each variable to be 4, the aggregate size of the multi-dimensional B-spline basis will be $4^2 + 4^2 = 32$, which is considered big with regards to the sample size we have available. Again, the `spline2ph()` function can be used, the only change is that the we supply both covariates `c("VL_unval_l10", "Age")` to the `x` argument.

```
sn_total = 4 ^ 2 + 4 ^ 2
bivariate_names <- paste0("bs", 1:sn_total)
data.bivariate <- spline2ph(x = c("VL_unval_l10", "Age"), size = 8,
                            degree = 3, bs_names = bivariate_names,
                            data = mock.vccc, group = "Sex",
                            split_group = FALSE)
```

The bivariate B-spline matrix is combined with the dataset and the corresponding column names are added as input arguments. Note that for the `linear2ph()` function, argument `hn_scale` is set to be 1/4 here, whereas it was set to 1 previously. This parameter controls the step size for the variance estimation using the method of profile likelihood (Murphy and Van der Vaart 2000). It tunes the numerical calculation of the profile log-likelihood and we can trouble-shoot the issue of occasional `NA` values in the covariance matrix by tuning `hn_scale`. In this case, there are `NA`s in the result when `hn_scale` is 1. We re-run the analysis with `hn_scale` set to $1/2, 1/4, 1/8$, and the variance estimates are very similar among these `hn_scale` values (data not shown). Therefore, we choose 1/4 to be the `hn_scale` value.

```
start.time <- Sys.time()
res_linear_bivariate <- linear2ph(y = "CD4_val_sq10",
                                  y_unval = "CD4_unval_sq10",
                                  x = "VL_val_l10", x_unval = "VL_unval_l10",
                                  z = c("Age", "Sex"),
                                  b_spline = bivariate_names,
                                  data = data.bivariate, hn_scale = 1/4,
                                  se = TRUE, tol = 1e-04, max_iter = 1000,
                                  verbose = FALSE)
paste0("Run time: ", round(difftime(Sys.time(), start.time,
                                    units = "secs"), 3), " sec")
```

```
[1] "Run time: 11.327 sec"
```

```
summary(res_linear_bivariate)$coefficients
```

```
           Estimate      SE Statistic  p-value
Intercept   5.02567 0.25515     19.70 0.000000
VL_val_l10 -0.13057 0.03938     -3.32 0.000913
Age        -0.00575 0.00475     -1.21 0.225746
Sex         0.28170 0.10901      2.58 0.009764
```

4.5.2 Principal component analysis

When there are several continuous covariates in the model, it may be challenging to construct a multidimensional B-spline basis using the tensor product method from section 4.5.1. The challenge is due to the curse of dimensionality and is especially true when there is a relatively small validation sample. One way around this is to use principal component analysis (PCA) to first reduce the dimension of the continuous covariates and then construct the B-spline basis based on the first principle component (PC) rather than the original covariates. Here, we illustrate the use of this approach by using the first PC to reduce the dimension of the

continuous covariates from two to one. However, this approach is versatile (and probably more useful) when there are more than two continuous covariates. We use the `prcomp()` function in base R to perform PCA for the two continuous covariates. The two input variables are the error-prone unvalidated VL and error-free age.

```
VLAge_pca_all <- prcomp(x = mock.vccc[,c("VL_unval_l10", "Age")],
                        center = TRUE, scale. = TRUE)
mock.vccc$VLAge_pca <- VLAge_pca_all$x[,1]
```

The steps below are identical to what we did in Section 4.3, except that we construct the B-spline basis on the first PC of VL and age rather than on the original covariates.

```
sn <- 20
pca_names <- paste0("bs", 1:sn)
data_pca <- spline2ph(x = "VLAge_pca", size = sn, degree = 3,
                      bs_names = pca_names, data = mock.vccc,
                      group = "Sex", split_group = TRUE)

start.time <- Sys.time()
res_linear_pca = linear2ph(y = "CD4_val_sq10", y_unval = "CD4_unval_sq10",
                       x = "VL_val_l10", x_unval = "VL_unval_l10",
                       z = c("Age", "Sex"), b_spline = pca_names,
                       data = data_pca, hn_scale = 1/4, se = TRUE,
                       tol = 1e-04, max_iter = 1000, verbose = FALSE)
paste0("Run time: ", round(difftime(Sys.time(), start.time,
                                    units = "secs"), 3), " sec")
```

```
[1] "Run time: 3.193 sec"
```

```
summary(res_linear_pca)$coefficients
```

```
           Estimate     SE Statistic  p-value
Intercept   5.05416 0.2573     19.64 0.000000
VL_val_l10 -0.13668 0.0393     -3.48 0.000498
Age        -0.00584 0.0047     -1.24 0.213566
Sex         0.27907 0.1092      2.56 0.010601
```

Note that these results are very close to those generated when using the bivariate B-spline basis in section 4.5.1.

4.6 Fitting a logistic regression model with `logistic2ph()`

We now illustrate fitting a logistic regression model using `logistic2ph()`. Suppose we are interested in fitting a logistic regression model of having an AIDS-defining event (ADE) within one year of ART initiation on CD4 count at ART initiation (CD4), adjusting for whether the patient is ART naive at enrollment. Among the three variables, both ADE and CD4 are error-prone and partially validated, and ART is error-free.

We set up the B-spline basis for estimating the error mechanisms in a similar way as in Section 4.3.1. That is, we set up different B-spline bases within each stratum of ART status at enrollment. This allows the errors in CD4 count to be heterogeneous between patients who are and are not ART naive at enrollment. Again, we assemble the variables and B-splines into one data frame prior to fitting the SMLE.

```
# same process as in Section 4.3.1
sn <- 20
logistic_names <- paste0("bs", 1:sn)
data.logistic <- spline2ph(x = "CD4_unval_sq10", size = 20, degree = 3,
                           bs_names = logistic_names, data = mock.vccc,
                           group = "Prior_ART", split_group = TRUE)
```

Now, we obtain the SMLEs for the logistic regression model of interest by running the `logistic2ph()` function on our augmented dataset:

```
start.time <- Sys.time()
res_logistic <- logistic2ph(y = "ADE_val", y_unval = "ADE_unval",
                            x = "CD4_val_sq10", x_unval = "CD4_unval_sq10",
                            z = "Prior_ART", b_spline = logistic_names,
                            data = data.logistic, hn_scale = 1/2, se = TRUE,
                            tol = 1e-04, max_iter = 1000, verbose = FALSE)
paste0("Run time: ", round(difftime(Sys.time(), start.time,
                                    units = "secs"), 3), " sec")
```

```
[1] "Run time: 295.504 sec"
```

The arguments here are analogous to those of `res_linear`. Argument `hn_scale` is set to be 1/2, and it is set using the same method as in Section 4.5.2.

Like `linear2ph()`, the `logistic2ph()` function returns the results in a `list` of class `logistic2ph`, which we have stored in the object `res_logistic`. The `summary()` function takes an object of class `logistic2ph` and gives the coefficient estimates and corresponding standard errors, z-statistics, and p-values.

```
(res_logistic_coef <- summary(res_logistic)$coefficients)
```

```
             Estimate     SE Statistic p-value
Intercept      -0.846 0.3044     -2.78 0.00545
CD4_val_sq10   -0.542 0.0629     -8.61 0.00000
Prior_ART      -0.214 0.2706     -0.79 0.42961
```

The coefficient estimate associated with CD4 indicates that the odds of having an ADE within one year of ART initiation decreases with increasing CD4. Specifically, adjusting for whether a patient is ART naive at enrollment, a person with a CD4 count of 360 cells/mm^3 is estimated to have $\exp(-0.542) = 0.582$ times the odds of having an ADE within one year of ART initiation compared to a person with a CD4 count of 250 cells/mm^3. A 95% confidence interval for this odds ratio, computed in the usual manner, is

```
exp(res_logistic_coef[2, 1] + c(-1.96, 1.96) * res_logistic_coef[2, 2])
```

```
[1] 0.514 0.658
```

After adjusting for CD4 count, the estimated odds ratio for having an ADE within one year for ART naive patients versus patients not ART naive is $\exp(-0.214) = 0.807$, and the 95% confidence interval is

```
exp(res_logistic_coef[3,1] + c(-1.96, 1.96) * res_logistic_coef[3,2])
```

```
[1] 0.475 1.372
```

Based on these results, the association between having ADE within one year of ART initiation and CD4 is significant at the 0.05 level. However, the association between having ADE within one year of ART initiation and whether the patient is ART naive at enrollment is not.

5. Summary and discussion

The `sleev` R package is a useful tool for analyzing two-phase validation studies with an error-prone outcome and covariates. It empowers users to perform linear regression for continuous outcomes and logistic regression for binary ones. The errors among variables can be correlated with each other and with additional error-free covariates. Conventional measurement error scenarios with errors in the outcome or covariates only are also accommodated. The resulting

SMLEs are statistically efficient and numerically stable while making minimal assumptions on the error distributions.

In this vignette, we demonstrate the usage of functions in the `sleev` package under different scenarios. We showcase the selection process of the size of a B-spline basis, which is not frequently seen in papers that involve the use of SMLEs. We also show the impact of the curse of dimensionality on the construction of the B-spline basis, and recommend PCA as a dimension reduction technique that can circumvent this challenge. We hope that users find the demonstrations in this vignette useful for their applications.

In the future, we plan to extend the SMLE to address errors in outcomes and covariates for models with count and time-to-event outcomes. Additional functions will be added to the `sleev` package as methods are developed for these settings.

Acknowledgement

This research was supported by the National Institute of Health grants R01HL094786, R01AI131771, and P30AI110527 and the 2022 Biostatistics Faculty Development Award from the Department of Biostatistics at Vanderbilt University Medical Center.

References

Amorim, G., R. Tao, S. Lotspeich, P. A. Shaw, T. Lumley, and B. E. Shepherd. 2021. "Two-Phase Sampling Designs for Data Validation in Settings with Covariate Measurement Error and Continuous Outcome." *Journal of the Royal Statistical Society. Series A, (Statistics in Society)*, 1368–89. https://doi.org/10.1111/rssa.12689.

Bang, H., and J. M. Robins. 2005. "Doubly Robust Estimation in Missing Data and Causal Inference Models." *Biometrics* 61 (4): 962–73. https://doi.org/10.1111/j.1541-0420.2005.00377.x.

Carroll, R. J., D. Ruppert, L. A. Stefanski, and C. M. Crainiceanu. 2006. *Measurement Error in Nonlinear Models: A Modern Perspective*. Chapman; Hall/CRC.

Cole, S. R., H. Chu, and S. Greenland. 2006. "Multiple-Imputation for Measurement-Error Correction." *International Journal of Epidemiology* 35 (4): 1074–81. https://doi.org/10.1093/ije/dyl097.

Dempster, A. P., N. M. Laird, and D. B. Rubin. 1977. "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society: Series B (Methodological)* 39 (1): 1–22.

Deville, J. C., C. E. Särndal, and O. Sautory. 1993. "Generalized Raking Procedures in Survey Sampling." *Journal of the American Statistical Association* 88 (423): 1013–20. https://doi.org/10.2307/2290793.

Duan, R., M. Cao, Y. Wu, J. Huang, J. C. Denny, H. Xu, and Y. Chen. 2016. "An Empirical Study for Impacts of Measurement Errors on EHR Based Association Studies." In *AMIA Annual Symposium Proceedings*, 2016:1764. American Medical Informatics Association. https://pubmed.ncbi.nlm.nih.gov/28269935/.

Fuller, Wayne A. 2009. *Measurement Error Models*. John Wiley & Sons. https://doi.org/10.1002/9780470316665.

Giganti, M. J., P. A. Shaw, G. Chen, S. S. Bebawy, M. M. Turner, T. R. Sterling, and B. E. Shepherd. 2020. "Accounting for Dependent Errors in Predictors and Time-to-Event Outcomes Using Electronic Health Records, Validation Samples, and Multiple Imputation." *The Annals of Applied Statistics* 14 (2): 1045. https://doi.org/10.1214/20-aoas1343.

Horvitz, D. G., and D. J. Thompson. 1952. "A Generalization of Sampling Without Replacement from a Finite Universe." *Journal of the American Statistical Association* 47 (260): 663–85. https://doi.org/10.2307/2280784.

Little, R. J. A., and D. B. Rubin. 1986. *Statistical Analysis with Missing Data*. John Wiley & Sons.

Lotspeich, S. C., B. E. Shepherd, G. Amorim, P. A. Shaw, and R. Tao. 2022. "Efficient Odds Ratio Estimation Under Two-Phase Sampling Using Error-Prone Data from a Multi-National HIV Research Cohort." *Biometrics* 78 (4): 1674–85. https://doi.org/10.1111/biom.13512.

Murphy, S., and A. Van der Vaart. 2000. "On Profile Likelihood." *Journal of the American Statistical Association* 95 (450): 449–65. https://doi.org/10.2307/2669386.

Oh, E. J., B. E. Shepherd, T. Lumley, and P. A. Shaw. 2021. "Raking and Regression Calibration: Methods to Address Bias from Correlated Covariate and Time-to-Event Error." *Statistics in Medicine* 40 (3): 631–49. https://doi.org/10.1002/sim.8793.

Schumaker, L. 2007. *Spline Functions: Basic Theory*. 3rd ed. Cambridge Mathematical Library. Cambridge University Press. https://doi.org/10.1017/CBO9780511618994.

Tang, L., R. H. Lyles, C. C. King, D. D. Celentano, and Y. Lo. 2015. "Binary Regression with Differentially Misclassified Response and Exposure Variables." *Statistics in Medicine* 34 (9): 1605–20. https://doi.org/10.1002/sim.6440.

Tao, R., S. C. Lotspeich, G. Amorim, P. A. Shaw, and B. E. Shepherd. 2021. "Efficient Semiparametric Inference for Two-phase Studies with Outcome and Covariate Measurement Errors." *Statistics in Medicine* 40 (3): 725–38. https://doi.org/10.1002/sim.8799.

Tao, R., D. Zeng, and D. Lin. 2017. "Efficient Semiparametric Inference Under Two-Phase Sampling, with Applications to Genetic Association Studies." *Journal of the American Statistical Association* 112 (520): 1468–76. https://doi.org/10.1080/01621459.2017.1295864.