`sleev`: Semiparametric Likelihood Estimation with Errors in Variables

Jiangmei Xiong, Sarah C. Lotspeich, Gustavo Amorim, Bryan E. Shepherd and Ran Tao

May 4, 2022

1 Introduction

Due to the nature of routine data collection, errors in outcomes and covariates can be inevitable in observational studies; for added complexity, these errors can be correlated. To correct the errors, validation studies are often conducted, but usually only for subsets of the database due to their monetarily and time-wise expensive nature, especially for large datasets. In this vignette, we introduce the `sleev` package, which implements the sieve maximum likelihood estimation (SMLE) of Tao et al. (2021) and Lotspeich et al. (2021b) to conduct valid and efficient inference using the partially validated, error-prone data. The SMLE is robust as it models the distribution of errors nonparametrically. It allows for non-random validation sample selection mechanisms. The main functions in `sleev` are `linear2ph()` and `logistic2ph()`, which handle linear and logistic regression, respectively, in the presence of errors in both the outcome and covariates.

2 Overview of SMLE

2.1 Linear regression

Suppose we want to fit a standard linear regression model for a continuous outcome Y and vector of covariates $\boldsymbol{X}$: $Y = \alpha + \mathbf{X}\boldsymbol{\beta} + \epsilon$, where ϵ follows a normal distribution with mean zero and variance σ^2. When we have errors in variables, Y and $\mathbf{X}$ are unobserved except for a few subjects whose records are validated. For the majority of subjects whose records are not validated, only the error-prone outcome $Y^* = Y + W$ and covariates $\boldsymbol{X^*} = \boldsymbol{X} + \boldsymbol{U}$ are observed, 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. With potential errors in our data, a naive regression analysis using error-prone variables Y^* and $\boldsymbol{X^*}$ could render misleading results.

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

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

Define $\boldsymbol{\theta} = (\alpha, \boldsymbol{\beta}^{\mathrm{T}}, \sigma^2)^{\mathrm{T}}$. Now consider the indicator variable V, with $V_i = 1$ if subject i received a data audit and 0 otherwise. For subjects who did not receive the data audit, measurement errors $(W, \boldsymbol{U})$ are not available, so the contributions of these subjects to the log-likelihood can be obtained by integrating out W and $\boldsymbol{U}$:

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

Expression (1) is the log-likelihood of the observed data. For estimates of $\boldsymbol{\beta}$ that are robust to assumptions about the measurement error mechanism, $P(W_i, \boldsymbol{U}_i|\boldsymbol{X}_i^*)$, an approach like nonparametric maximum likelihood estimation (NPMLE) would be ideal. However, this method requires discrete estimation of $P(W, \boldsymbol{U}|\boldsymbol{X^*} = \boldsymbol{x^*})$ based on the empirical density. This estimator will have too few observations to be applicable since a small number of subjects will have $\boldsymbol{X}^* = \boldsymbol{x}^*$ and even fewer will have the different values of $(W, \boldsymbol{U})$. Instead, SMLE is used, wherein $\log \mathrm{P}(W_i, \boldsymbol{U}_i|\boldsymbol{X}_i^*)$ and $\mathrm{P}(W_i, \boldsymbol{U}_i|\boldsymbol{X}_i^*)$ are approximated by $\sum_{k=1}^{m} \mathrm{I}(w = w_k, \boldsymbol{u} = \boldsymbol{u}_k)\sum_{j=1}^{s_n} B_j^q(\boldsymbol{X}_i^*)p_{kj}$ and $\sum_{k=1}^{m} \mathrm{I}(w = w_k, \boldsymbol{u} = \boldsymbol{u}_k)\sum_{j=1}^{s_n} B_j^q(\boldsymbol{X}_i^*)\log p_{kj}$, respectively. Here: m is the number of distinct $pmbx^*$ values; $B_j^q(\cdot)$ represents the jth B-spline function of order q; s_n is determined by first-phase sample size n; and p_{kj} is coefficient with constraints $\sum_{k=1}^{m} p_{kj} = 1, p_{kj} \geq 0$, k here indicates correspondence to kth distinct $\boldsymbol{x}^*$ value and j indicates correspondence to jth B-spline. A B-spline basis is a type of piecewise polynomial function, and can be taken as a fitted curve on the data points. This can be seen as "smoothing" on the continuous $(\boldsymbol{X}^*, Y^*)$ to make estimation of the density tractable. The log-likelihood expression 1 is now being approximated by

$$\sum_{i=1}^{n} V_i\{\log \mathrm{P}_{\boldsymbol{\theta}}(Y_i|\boldsymbol{X}_i) + \sum_{k=1}^{m} \mathrm{I}(w = w_k, \boldsymbol{u} = \boldsymbol{u}_k) \sum_{j=1}^{s_n} B_j^q(\boldsymbol{X}_i^*) p_{kj}$$
$$+ \sum_{i=1}^{n} (1 - V_i) \log \left\{ \int \int \mathrm{P}_{\boldsymbol{\theta}}(Y_i^* - w|\boldsymbol{X}_i^* - \boldsymbol{u}) \sum_{k=1}^{m} \mathrm{I}(w = w_k, \boldsymbol{u} = \boldsymbol{u}_k) \sum_{j=1}^{s_n} B_j^q(\boldsymbol{X}_i^*) \log p_{kj} \mathrm{d}w \mathrm{d}\boldsymbol{u} \right\}. \tag{2}$$

The maximization of expression 2 is carried out through an expectation-maximization (EM) algorithm. The variance estimates of the parameters are obtained through the profile likelihood method (Murphy and Van der Vaart, 2000). Full details on this method, including its theoretical properties, can be found in Tao et al. (2021).

2.2 Binary outcomes: Logistic Regression

This setting is similar to the previous one, except that the outcome Y is binary and we would wish to fit a logistic regression model $\mathrm{P}_{\boldsymbol{\theta}}(Y = 1|\mathbf{X}) = [1 + \exp\{-(\alpha + \boldsymbol{\beta}\mathbf{X})\}]^{-1}$, where parameters $\boldsymbol{\theta} = (\alpha, \boldsymbol{\beta}^{\mathrm{T}})^{\mathrm{T}}$. Note that this is $\mathrm{logit}(Y = 1|\mathbf{X}) = \log(\mathrm{P}_{\boldsymbol{\theta}}(Y = 1|\mathbf{X})/[1 - \mathrm{P}_{\boldsymbol{\theta}}(Y = 1|\mathbf{X})]) = \alpha + \mathbf{X}\boldsymbol{\beta}$ after transformation. This time, the joint density of a complete observation is of the form

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

where $P(Y|\boldsymbol{X}, \boldsymbol{X}^*) = P(Y|\boldsymbol{X})$ follows from the assumption that Y and $\boldsymbol{X}^*$ are independent given $\boldsymbol{X}$ (i.e., surrogacy). Similar to the linear regression case, V is the data audit indicator, and unvalidated subjects will need to have their densities marginalize out the unobserved Y and $\boldsymbol{X}$. Again, the log-likelihood here ignores $P(\boldsymbol{X}^*)$, as it is fully observed.

$$\sum_{i=1}^{n} V_i\{\log P_\theta(Y_i|\boldsymbol{X}_i) + \log P(Y_i^*|\boldsymbol{X_i^*}, Y_i, \boldsymbol{X}_i) + \log P(\boldsymbol{X}_i|\boldsymbol{X_i^*})\} + \sum_{i=1}^{n} (1 - V_i) \log \left\{ \sum_{y=0}^{1} \int_{\boldsymbol{x}} \log P_\theta(y|\boldsymbol{x}) P(Y_i^*|\boldsymbol{X_i^*}, y, \boldsymbol{x}) P(\boldsymbol{x}|\boldsymbol{X_i^*}) d\boldsymbol{x} \right\}$$

Among the terms, $\mathrm{P}(Y_i^*|\boldsymbol{X}_i^*, Y_i, \boldsymbol{X}_i)$ is fitted with an additional logistic regression model. $\mathrm{P}(\boldsymbol{X} = \boldsymbol{x}|\boldsymbol{X}_i^* = \boldsymbol{x}_i^*)$ is estimated with discrete probabilities when X is discrete, and SMLE is used when X is continuous. Following similar steps as in Section 2.1, we approximate $\log \mathrm{P}(\boldsymbol{X}_i|\boldsymbol{X}_i^*)$ and $\mathrm{P}(\boldsymbol{x}|\boldsymbol{X}_i^*)$ in the log-likelihood by $\sum_{k=1}^{m} \mathrm{I}(\boldsymbol{X}_i = \boldsymbol{x}_k) \sum_{j=1}^{s_n} \log p_{kj} B_j^q(\boldsymbol{X}_i^*)$ and $\sum_{k=1}^{m} \mathrm{I}(\boldsymbol{X}_i = \boldsymbol{x}_k) \sum_{j=1}^{s_n} p_{kj} B_j^q(\boldsymbol{X}_i^*)$, respectively. Again, the constrained maximization of the resulting expression is carried out through an EM algorithm and profile likelihood variance estimation. More details can be found in **?**.

2.3 SMLE Versus Existing Approaches

Apart from high statistical efficiency, the SMLE methods described stand out from existing methods because they can be used where both outcome and covariate error exists, do not assume distributions on the errors, and the full-likelihood specification can accommodate a multitude of informed audit designs, such as extreme-tail sampling.

The number of covariates in the B-spline basis is ideally two to three, due to the curse of dimensionality. In situations where we have more covariates, readers might consider dimension reduction on covariates (e.g., principal components analysis) or omitting error-free covariates that are independent from error-prone covariates. Both of these are promising workarounds.

3 Example with Data

The package can be installed through CRAN:

```
install.packages(sleev)
library(sleev)
data(mock.vccc) #assuming this data can be pulled from package
```

3.1 Data overview

In this section, we illustrate the usage of the function by using a simulated dataset. This dataset is constructed to imitate the Vanderbilt Comprehensive Care Clinic (VCCC) patient records, which have been fully validated and therefore contain validated and unvalidated versions of all variables. The VCCC cohort is a good candidate for the purpose of illustration. The data presented in this section are a mocked-up version of the actual data due to confidentiality, but the data structure and features, such as mean and variability, closely resemble the real dataset. Following the examples used with VCCC dataset in Lotspeich et al. (2021a) and Amorim et al. (2021), the mock dataset is created and the variable descriptions can be found in Table 1. The dataset can be loaded as

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

Name	Description	Status,Type
VL_unval	viral load at antiretroviral therapy (ART) initiation	error-prone outcome, continuous
VL_val	viral load at antiretroviral therapy (ART) initiation	validated outcome, continuous
ADE_unval	having an AIDS-defining event (ADE) within one year of ART initiation	error-prone outcome, binary
ADE_val	having an AIDS-defining event (ADE) within one year of ART initiation	validated outcome, binary
CD4_unval	CD4 count at ART initiation	error-prone covariate, continuous
CD4_val	CD4 count at ART initiation	validated covariate, continuous
ART	whether patient is ART naive at enrollment	error-free covariate, binary
Gender	gender of patient, 1 indicates male and 0 indicates female	error-free covariate, binary

Table 1: The variables in mock VCCC dataset

Note that both CD4 and VL should both be transformed for the consistency in the scale of the variable. CD4 should be divided by 10 and square root transformed, and can hence be inerpreted as CD4 count per square centimeters. VL should be square root transformed:

```
mock.vccc$CD4_val[!is.na(mock.vccc$CD4_val)] <- sqrt(mock.vccc$CD4_val[!is.na(mock.vccc$CD4_val)]/10)
mock.vccc$CD4_unval <- sqrt(mock.vccc$CD4_unval)
mock.vccc$VL_val[!is.na(mock.vccc$VL_val)] <- log10(mock.vccc$VL_val[!is.na(mock.vccc$VL_val)])
mock.vccc$VL_unval <- log10(mock.vccc$VL_unval)
```

3.2 Fitting a linear model with linear2ph()

Suppose we are fitting a linear regression model where CD4 count at antiretroviral therapy (ART) initiation (CD4) is dependent on the viral load at ART initiation (VL), adjusting for gender (Gender). Both CD4 and VL are error-prone and partially validated, and Gender is error-free.

3.2.1 Setting up the B-spline for data

To analyze the dataset, we need to set up the B-spline basis on VL_unval (the error-prone viral load variable) and Gender and add it to the dataframe. The splines package is needed to do this:

```
install.packages(‘splines’)
library(splines)
```

Here we use a cubic B-spline basis with input degree=3 in our call to the function bs(). The number of basis functions, usually called the “number of sieves” and denoted s_n in Section 2, is set to be 20; this input is fed into bs() through the df = 20 parameter. The selection of number of sieves will be introduced in Section 3.2.4.

```
n <- nrow(mock.vccc)
Bspline_0 = splines::bs(x=mock.vccc$VL_unval[mock.vccc$Gender==0], df=nsieve, degree=3,
                    Boundary.knots=range(mock.vccc$VL_unval[mock.vccc$Gender==0]), intercept=TRUE)
Bspline_1 = splines::bs(x=mock.vccc$VL_unval[mock.vccc$Gender==1], df=nsieve, degree=3,
                        Boundary.knots=range(mock.vccc$VL_unval[mock.vccc$Gender==1]), intercept=TRUE)
Bspline = matrix(0, n, 2*nsieve)
Bspline[mock.vccc$Gender==0,1:nsieve] = Bspline_0
Bspline[mock.vccc$Gender==1,(nsieve+1):(2*nsieve)] = Bspline_1
colnames(Bspline) = paste("bs", 1:(2*nsieve), sep="") # name B-spline columns properly
```

df defines the number of equally-spaced internal knots and Boundary.knots defines at what range the internal knots are defined.

Note that in the code above, B-splines are set up separately for different stratum of the error-free binary variable Gender. Recall that we assumed the error-free binary predictor Gender is correlated with error-prone predictor VL_unval, therefore we fit B-spline functions to viral load for female and male patients separately. If some stratum has very few observations, it is recommended to combine it with other stratum, or fit B-spline with all data. Finally, assemble the variables and B-splines into one dataframe:

```
data = data.frame(cbind(mock.vccc, Bspline))
```

3.2.2 Fitting the model

Now, we can run the analysis

```
    res_linear = sleev::linear2ph(Y="CD4_val", X="VL_val", Y_unval="CD4_unval", X_unval="VL_unval",
                                  Z="Gender", Bspline=colnames(Bspline), data=data)
```

The argument inputs here are the corresponding column name in the dataset input, and they are in string format. Argument `Y` is the audited error-prone outcome variable, and `Y_unval` is the unaudited error-prone predictor variable. The meaning of `X` and `X_unval` follows the same pattern. `Z` is the error-free covariate. Note that the input of Bspline is the vector of column name inputs in the previous section. If you have more than one variable that fits into the type of argument, simply put in a vector of the column names like with `Bpline`.

The result is stored in a `list` object, which we called `res_linear`. We discuss the elements of this list in the Section that follows.

3.2.3 Interpreting model results

The `res_linear$coefficients` slot gives the coefficients, corresponding standard error estimates, t-statistics, and p-values for the outcome model:

```
    $coefficients
            Estimate         SE Statistic     p-value
Intercept  4.9911029 0.25477684 19.590096 0.000000000
VL_val    -0.1943532 0.06358564 -3.056557 0.002238946
Gender     0.2400633 0.17414664  1.378512 0.168045209
```

Similar to interpreting output from `lm()`, the output here indicates that for every 1-unit increase in the $\log_{10}$-transformed viral load at (ART) initiation, there is expected to be 0.194-units decrease in CD4 count per square centimeters at ART initiation, adjusting for gender. It is expected that the average $\log_{10}$-transformed viral load for males is 0.24-units higher than that for females, adjusting for viral load. Based on p-value output, the viral load is statistically significant and gender is not, at the 0.05 significance level.

The `res_linear$covariance` slot gives the covariance matrix, which can be useful for obtaining confidence intervals for linear combinations of covariates.

```
    $covariance
            [,1]          [,2]          [,3]
[1,]  0.06491124 -1.304698e-02 -2.271704e-02
[2,] -0.01304698  4.043134e-03 -2.889536e-05
[3,] -0.02271704 -2.889536e-05  3.032705e-02
```

3.2.4 Specifying B-spline parameters

There are two B-spline parameters that need specification: degree and number of sieves. Degree of 3, which is cubic spline, is most commonly used. The optimal number of sieves can be selected through utilizing the `sleev::cv_linear2ph()` function in this package. This function uses k-fold cross-validation to calculate the predicted log-likelihood under various B-spline specifications. To be more specific, to perform a k-fold cross validation, data is split into k parts. Among them, use $2, ..., k$ parts to fit the model, and test the model on the first part (usually obtain sum of residuals, etc). Repeat this process for all k parts and obtain the average measure of model fit.

The following code loops through different numbers of sieves and compare the log-likelihood of each fitted result. The number of sieves with highest log-likelihood will be chosen.

```
nsieves = c(5, 10, 15, 20, 25, 30, 40, 50)
pred_loglike.1 = rep(NA, length(nsieves))
for (i in 1:length(nsieves)) {
  nsieve = nsieves[i]
  Bspline = splines::bs(mock.vccc$VL_unval, df=nsieve, degree=3,
              Boundary.knots=range(mock.vccc$VL_unval), intercept=TRUE)
  colnames(Bspline) = paste("bs", 1:nsieve, sep="")
  # cubic basis

  data.sieve = cbind(mock.vccc, Bspline)
  ### generate data
  res.1 = sleev::cv_linear2ph(Y="CD4_val", X="VL_val", Y_unval="CD4_unval",
                   X_unval="VL_unval", Z="Gender", Bspline=colnames(Bspline),
                   data=data.sieve, nfold=5)
  pred_loglike.1[i] = res.1$avg_pred_loglik
}
data.frame(nsieves, pred_loglike.1, pred_loglike)
```

The output of this loop is:

```
   nsieves pred_loglike.1
 1       5      -1400.560
 2      10      -1400.540
 3      15      -1401.059
 4      20      -1400.092
 5      25      -1401.547
 6      30      -1400.790
 7      40      -1401.105
 8      50      -1401.050
```

It can be seen that 20 sieves has the lowest log-likelihood, and is therefore chosen. Note that the log-likelihoods are fairly flat across the changing number of sieves, which means that the number of sieves does not impact the results very much. This is also true in the simulations performed in Tao et al. (2021). This method of choosing number of sieves can also be used in the parallel function for logistic regression, `sleev::cv_logistic2ph()`, to be introduced next.

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

Suppose we are fitting logistic regression model of having an AIDS-defining event (ADE) within one year of ART initiation (`ADE`) on CD4 count at ART initiation (`CD4`), adjusting for whether 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.

3.3.1 Setting up the B-spline for data

Again, we set up B-splines in similar way as in Section 3.2.1.

```
nsieve=20
Bspline_0 = splines::bs(mock.vccc$CD4_unval[mock.vccc$ART==0], df=nsieve,
degree=3,Boundary.knots=range(mock.vccc$CD4_unval[mock.vccc$ART==0]),intercept=TRUE)
Bspline_1 = splines::bs(mock.vccc$CD4_unval[mock.vccc$ART==1], df=nsieve, degree=3,
                        Boundary.knots=range(mock.vccc$CD4_unval[mock.vccc$ART==1]), intercept=TRUE)
Bspline = matrix(0, n, 2*nsieve)
Bspline[mock.vccc$ART==0,1:nsieve] = Bspline_0
Bspline[mock.vccc$ART==1,(nsieve+1):(2*nsieve)] = Bspline_1
colnames(Bspline) = paste("bs", 1:(2*nsieve), sep="") # name B-spline columns properly
```

That is, we have set up B-spline basis within stratum of different ART status at enrollment, because we have assumed that error-free variable `ART` is correlated with error-prone predictor `CD4_unval`. In other words, we assumed that the expected CD4 count differs between patients who are and aren't ART naive at enrollment, therefore the B-spline basis is calculated separately.

Finally, assemble the variables and B-splines into one dataframe.

```
data = data.frame(cbind(mock.vccc, Bspline))
```

3.3.2 Fitting the model

Now we run the analysis on the `data`, which augments the `mock.vccc` with the `Bspline`.

```
res_logistic = sleev::logistic2ph(Y="ADE_val", X="CD4_val", Y_unval="ADE_unval",
                X_unval="CD4_unval", Z="ART",Bspline=colnames(Bspline), data=data)
```

The arguments here are completely analogous to `res_linear`.

Like `sleev::logistic2ph`, the function returns the results in a `list`, which we store in object `res_logistic`.

3.3.3 Interpreting model results

`res_logistic$coeff` gives the coefficient estimates and corresponding standard error, t-statistic and p-value.

```
               coeff         se
Intercept -0.8890539 0.34468880
CD4_val   -0.5497318 0.08220148
ART       -0.1212772 0.30028783
```

The coefficient estimate associated with CD4 indicates that for every 1 unit increase in CD4 count per square centimeters, there are 0.55 decrease in log-odds of having an AIDS-defining event (ADE) within one year of ART initiation, adjusting for whether patient is ART naive at enrollment; ART naive patients at enrollment has lower log odds of 0.121 at having an AIDS-defining event (ADE) within one year than patients who are not ART naive at enrollment, adjusting for CD4 count. [placeholder: comment on p-values.]

4 Summary

The two functions introduced in this paper, `sleev::linear2ph()` and `sleev::logistic2ph()`, are very useful tools for drawing inferences from partially validated data while utilising both validated and unvalidated part of the data. Both of the functions can deal with error-prone outcome and covariates appearing simultaneously or individually, and do not make assumption on error distribution. We hope users find the demonstrations in the paper useful for their application.

References

G. Amorim, R. Tao, S. Lotspeich, P. A. Shaw, T. Lumley, and B. E. Shepherd. 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)*, page 1368–1389, 2021. URL `https://doi.org/10.1111/rssa.12689`.

S. C. Lotspeich, G. Amorim, P. A. Shaw, R. Tao, and B. E. Shepherd. Optimal multi-wave validation of secondary use data with outcome and exposure misclassification, 2021a.

S. C. Lotspeich, B. E. Shepherd, G. Amorim, P. A. Shaw, and R. Tao. Efficient odds ratio estimation under two-phase sampling using error-prone data from a multi-national HIV research cohort. *Biometrics*, 2021b. in press.

S. Murphy and A. Van der Vaart. On profile likelihood. *Journal of the American Statistical Association*, 95(450):449–465, 2000.

R. Tao, S. C. Lotspeich, G. Amorim, P. A. Shaw, and B. E. Shepherd. Efficient semiparametric inference for two-phase studies with outcome and covariate measurement errors. *Statistics in Medicine*, 40(3):725–738, 2021.