An Introduction to Principal Surrogate Evaluation with the pseval Package

by Michael C. Sachs, Erin E. Gabriel

Abstract We describe a new package called **pseval** that implements the core methods for the evaluation of principal surrogates in a single clinical trial. It provides a flexible interface for defining models for the risk given treatment and the surrogate, the models for integration over the missing counterfactual surrogate responses, and the estimation methods. Estimated maximum likelihood and pseudo-score can be used for estimation, and the bootstrap for inference. A variety of post-estimation methods are provided, including print, summary, plot, and testing. We summarize the main statistical methods that are implemented in the package and illustrate its use from the perspective of a novice R user.

Introduction

A valid principal surrogate endpoint, also called a specific nonmechanistic correlate of protection (Plotkin and Gilbert, 2012), can be used as a target for treatment improvement in early phase trials and, in the specific setting of evaluation, for predicting individual treatment effects post-licensure. A surrogate is considered to be valid if it provides reliable predictions of treatment effects on the clinical endpoint of interest. Frangakis and Rubin (2002) introduced the concept of principal stratification and the definition of a principal surrogate (PS). Informally, a post-treatment intermediate response variable is a principal surrogate if causal effects of the treatment on the clinical outcome only exist when causal effects of the treatment on the intermediate variable exist. The criteria for a PS have been modified and extended in more recent works, with most current literature focusing on wide effect modification as the primary criterion of interest.

The goal of PS evaluation is estimation and testing of how treatment efficacy on the clinical outcome of interest varies over subgroups defined by possible treatment and surrogate combinations of interest; this is an effect modification objective. The combinations of interest are called the principal strata and they include a set of unobservable counterfactual responses: responses that would have occurred under a set of conditions counter to the observed conditions. To finesse this problem of unobservable responses, a variety of clever trial designs and estimation approaches have been proposed. Several of these have been implemented in the **pseval** package (Sachs and Gabriel, 2016).

Methods

Notation

Let Z_i be the treatment indicator for subject i, where 0 indicates the control or standard treatment, and 1 indicates the experimental treatment. We currently only allow for two levels of treatment and assume that the treatment assignments are randomized. Let S_i be the observed value of the intermediate response for subject i. Since S_i can be affected by treatment, there are two naturally occurring counterfactual values of S_i: $S_i(1)$ under treatment, and $S_i(0)$ under control. Let s_z be the realization of the random variable $S(z)$, for $z \in \{0, 1\}$. The outcome of interest is denoted Y_i. We consider the counterfactual values of $Y_i(0)$ and $Y_i(1)$. We allow for continous, binary, count, and time-to-event outcomes, thus Y_i may be a vector containing a time variable and an event/censoring indicator, i.e. $Y_i = (T_i, \Delta_i)$ where $\Delta_i = 1$ if T_i is an event time, and $\Delta_i = 0$ if T_i is a censoring time. In event driven settings, $S_i(z)$ is only defined if the event, $Y_i(z)$, does not occur before the potential surrogate $S_i(z)$ is measured at a fixed time τ after entry into the study. The data analyses only include participants who have not experienced the event outcome by time τ.

Estimands

Criteria for S to be a good surrogate are based on risk estimands that condition on the potential intermediate responses. The risk is defined as a mapping g of the cumulative distribution function of $Y(z)$ conditional on the intermediate responses. The joint risk estimands conditions on the candidate surrogate under both level of treatment, $(S(1), S(0))$.

$$risk_1(s_1, s_0) = g\left\{F_{s_1}\left[Y(1)|S(0) = s_0, S(1) = s_1\right]\right\},$$

$$risk_0(s_1, s_0) = g\left\{F_{s_1}\left[Y(0)|S(0) = s_0, S(1) = s_1\right]\right\}.$$

For instance, for a binary outcome, the risk function may simply be the probability $risk_z(s_1, s_0) = P(Y(z) = 1|S(0) = s_0, S(1) = s_1)$, or for a time-to-event outcome the risk function may be the cumulative distribution function $risk_z(s_1, s_0) = P(Y(z) \leq t|S(0) = s_0, S(1) = s_1)$.

Currently we focus only on marginal risk estimands which condition only on $S(1)$, the intermediate response or biomarker under active treatment:

$$risk_1(s_1) = g\left\{F_{s_1}\left[Y(1)|S(1) = s_1\right]\right\},$$

$$risk_0(s_1) = g\left\{F_{s_1}\left[Y(0)|S(1) = s_1\right]\right\}.$$

Neither of the joint risk estimands are indentifiable in a standard randomized trial, as either $S(0)$ or $S(1)$ or both will be missing for each subject. In the special case where $S(0)$ is constant, such as the immune response to HIV antigens or Hep B in the placebo arm of a vaccine trial, the joint and marginal risk estimands are equivalent. This special case is referred to as case constant biomarker (CB) in much of the literature (Gilbert and Hudgens, 2008); i.e., $S_i(0) = c$ for subjects i. This may occur outside the vaccine setting when one considers the AUC of a treament drug as a surrogate; those recieving placebo will have no drug and therefore all placebo AUC will be 0 or undefined. Under assumptions given below, and in the case CB setting, the marginal risk estimand is indentifiable in the treatment arm; it is not indentifiable in the control arm without further assumptions or trial agumentation (Wolfson and Gilbert, 2010).

There are specific trial augmentations that allow for the measurement or imputation of the missing counterfactual Ss, in the control and treament arms. As well, Under one of these augmentations case CB can sometimes be induced by considering a function of the a candidate surrogate for evaluation. Greater detail on this point given below.

Specification of the distributions of $Y(z)|S(1)$ determines the likelihood, we will denote this as $f(y|\beta, s_1, z)$. If $S(1)$ were fully observed, simple maximum likelihood estimation could be used. The key challenge in estimating these risk estimands is solving the problem of conditioning on counterfactual values that are not observable for at least a subset of subjects in a randomized trial. This involves integrating out missing values based on some model, and under some set of assumptions and/or trial augmentations.

Prinicipal surrogate criteria

Frangakis and Rubin (2002) gave a single criterion for a biomarker S to be a PS: causal effects of the treatment on the clinical outcome only exist when causal effects of the treatment on the intermediate variable exist. In general this can only be evaluated using the joint risk estimands, which consider not only the counterfactual values of the biomarker under treatment, but also under control $S(0)$. However, in the special case where all $S(0)$ values are constant, say at level C, such as an immune response to HIV in a HIV negative population pre-vaccination this criteria, often referred to as average causal necessity (ACN), can by written in terms of the marginal risk estimands as:

$$risk_1(C) = risk_0(C).$$

More recently, other works Gilbert and Hudgens (2008), Wolfson and Gilbert (2010), Huang and Gilbert (2011), Huang et al. (2013), Gabriel and Gilbert (2014), and Gabriel and Follmann (2016) have suggested that this criterion is both too restrictive and in some cases can be vacuously true. Instead most current works suggest that the wide effect modification (WEM) criterion is of primary importance, ACN being of secondary importance. WEM is given formally in terms of the risk estimands and a known contrast function h satisfying $h(x, y) = 0$ if and only if $x = y$ by:

$$|h(risk_1(s_1), risk_0(s_1)) - h(risk_1(s_1{}^*), risk_0(s_1{}^*))| > \delta,$$

for at least some $s_1 \neq s_1^*$ and $\delta > 0$, with the larger the δ the better the surrogate. Examples of contrast functions are the treatment efficacy, $h(x, y) = 1 - x/y$, and the risk difference $h(x, y) = x - y$. To evaluate WEM and ACN we need to identify the risk estimands, which condition on data that is missing for at least half of the subjects in a standard randomized trial.

Augmentation and assumptions

We first make three standard assumptions used in much of the literature for absorbing events outcomes:

- Stable Unit Treatment Value Assumption (SUTVA): Observations on the independent units in the trial should be unaffected by the treatment assignment of other units.

- Ignorable Treatment Assignment: The observed treatment assignment does not change the counterfactual clinical outcome.
- Equal individual risk up to the time of candidate surrogate measurement τ.

In time-to-event settings one more assumption is needed:

- Non-informative censoring.

It should be noted that the equal individual risk assumption requires that time-to-event analysis start at time τ, rather than at randomization.

Wolfson and Gilbert (2010) outlines how these assumptions are needed for identification of the risk estimands. Now to deal with the missing $S(1)$ values among those with $Z = 0$, we next focus on three trial augmentations: Baseline immunogenicity predictor (BIP), closeout placebo vaccination (CPV), a concept that was extend to the setting of general treatment trials under the name of closeout control treatment (CCT) in Gabriel and Follmann (2016), and baseline surrogate measurement (BSM). For further details on these augmentations, we refer you to Follmann (2006), Gilbert and Hudgens (2008), Gabriel and Gilbert (2014), and for further augmenations not yet implemented to Gabriel and Follmann (2016).

BIP

Briefly, a BIP W is any baseline measurement or set of measurements that is highly correlated with S. It is particularly useful if W is unlikely to be associated with the clinical outcome after conditioning on S, i.e. $Y \perp W|S(1)$; some of the methods leverage this assumption. The BIP W is used to integrate out the missing $S(1)$ among those with $Z = 0$ based on a model for $S(1)|W$ that is estimated among those with $Z = 1$. We describe how this model is used in the next section.

The assumptions needed for a BIP to be useful depend on the risk model used. If the BIP is included in the risk model, only the assumption of no interaction with treatment and the candidate surrogate are needed. However, if the BIP is not included in the risk model, the assumption that that clinical outcome is independent of the BIP given the candidate surrogate is needed. Although not a requirement for identification of the risk estimands, it has been found in most simulations studies that a correlation between the BIP and $S(1)$ of greater than 0.7 is needed for unbiased estimation in finite samples.

CPV or CCT

Under a CPV or CCT augmented design, control recipients that do not have events, or all willing control subjects for a non-event driven clinical outcome, are given the experimental treatment at the end of the follow-up period. Then, their intermediate response is measured at some time after that treatment. This measurement is then used as a direct imputation for the missing $S(1)$. The CPV augmentation was developed in the setting of vaccine trials, where the surrogate is an immune response and the outcome is infection. One set of conservative assumptions to use CPV as a direct imputation for $S(1)$ in a vaccine trial are given in Wolfson and Gilbert (2010) are:

- Individual time constancy of the true intermediate response under active treatment, $S(1) = S_{CPV}$ almost surely, for placebo recipients that are crossed over at the end of the trial, where S_{CPV} is the measurement of the candidate surrogate after crossover treatment of the placebo subjects.
- No events (infections) during the close-out period.

In the general treatment trial setting, the CCT augmentation can be used under the same Individual time constancy assumption, and the assumption that drop-out or unwillingness to receive close-out treatment is completely at random.

BSM

Gabriel and Gilbert (2014) suggested the baseline augmentation BSM, which is a pre-treatment measurement of the candidate PS, denoted S_B. The BSM may be a good predictor of $S(1)$ without any further assumptions. It can be used in the same way as a BIP. Alternatively you can transform $S(1) - S_B$ and use this as the candidate surrogate, further increasing the association with the BSM/BIP. Under the BSM assumption outlined in Gabriel and Gilbert (2014);

- Time constancy of the true intermediate response under control,

then $S(0) = S_{BSM}$ almost surely. You do not need this assumption to use a BSM, but if it holds then it induces the CB case, thus the joint and marginal risk estimands are equivalent.

Risk estimation

Estimated maximum likelihood

Let $f(y|\beta, s_1, z)$ denote the density of $Y|S(1), Z$ with parameters β. Further let R_i denote the indicator for missingness in $S_i(1)$. We proceed to estimate β by maximizing

$$\prod_{i=1}^{n} \{f(Y_i|\beta, S_i(1), Z_i)\}_i^R \left\{ \int f(Y_i|\beta, s, Z_i)\, d\hat{F}_{S(1)|W}(s|W_i) \right\}^{1-R_i}$$

with respect to β.

This procedure is called estimated maximum likelihood (EML) and was developed in Pepe and Fleming (1991). The key idea is that we are averaging the likelihood contributions for subjects missing $S(1)$ with respect to the estimated distribution of $S(1)|W$, denoted by $\hat{F}_{S(1)|W}(s|W_i)$. The model for this distribution is referred to as the integration model. Recall that a BIP W that is strongly associated with $S(1)$ is needed for adequate performance.

Closed-form inference is not available for EML estimates, thus we recommend use of the bootstrap for estimation of standard errors. It was suggested as an approach to principal surrogate evaluation by Gilbert and Hudgens (2008) and Huang and Gilbert (2011).

Pseudoscore

Huang et al. (2013) suggest a different estimation procedure that does have a closed form variance estimator. Instead of numerically optimizing the estimated likelihood, the pseudoscore approach iteratively finds the solution to weighted versions of the score equations. Pseudoscore estimates were also suggested in Wolfson (2009) and implemented for several special cases in Huang et al. (2013). We have implemented here only one of the special cases: categorical BIP and binary Y (S may be continuous or categorical). In addition to having closed form variance estimators, it has been argued that the pseudo-score estimators are more efficient than the EML estimators. The closed form variance estimates are not yet implemented.

Package features

Typically, users would have to code up the likelihood, integration model, and perform the optimization themselves. This is beyond the reach of many researchers who desire to use these methods. The goal of **pseval** is to correctly implement these methods with a flexible and user-friendly interface, enabling researchers to implement and interpret a wide variety of models.

The **pseval** package allows users to specify the type of augmented design that is used in their study, specify the form of the risk model along with the distribution of $Y|S(1)$, and specify different integration models to estimate the distribution of $S(1)|W$. Then the likelihood can be maximized and bootstraps run. Post-estimation summaries are available to display and analyze the treatment efficacy as a function of $S(1)$. All of this is implemented with a flexible and familiar interface.

Package information

Usage

Here we will walk through some basic analyses from the point of view of a new R user. Along the way we will highlight the main features of **pseval**. We support binary, continuous, count, and time-to-event outcomes, thus we will also need to load the **survival** package (Therneau, 2015; Therneau and Grambsch, 2000).

Example dataset

First let's create an example dataset. The **pseval** package provides the function `generate_example_data` which takes a single argument: the sample size.

```
set.seed(1492)
fakedata <- generate_example_data(n = 800)
head(fakedata)
```

```
##   Z        BIP       CPV         BSM       S.obs  time.obs
## 1 0  0.3353179 1.4851399  0.45961614  0.35268095 0.3301972
## 2 0  1.4536863 2.6379400  1.39591042  1.46688905 0.1195136
## 3 0 -0.7243934        NA -0.62723499 -0.73190763 0.2631222
## 4 0 -0.1183592 0.9421504  0.07738308 -0.01833409 0.1373458
## 5 0 -0.2352566        NA -0.14971448 -0.18470242 0.8543703
## 6 0 -0.7782851 0.1159434 -0.65721609 -0.66313714 0.2200481
##   event.obs Y.obs      S.obs.cat         BIP.cat
## 1         0     0 (-0.198,0.503]  (0.0574,0.766]
## 2         1     0    (1.36, Inf]    (0.766, Inf]
## 3         1     1  (-Inf,-0.198]   (-Inf,-0.678]
## 4         1     0 (-0.198,0.503] (-0.678,0.0574]
## 5         1     1 (-0.198,0.503] (-0.678,0.0574]
## 6         1     0  (-Inf,-0.198]   (-Inf,-0.678]
```

The example data includes both a time-to-event outcome, a binary outcome, a surrogate, a BIP, CPV, and BSM, and a categorical version of the surrogate. The true model for the time is exponential, with parameters (intercept) = -1, S(1) = 0.0, Z = 0.0, S(1):Z = -0.75. The true model for binary is logistic, with the same parameter values.

In the above table S.obs.cat and BIP.cat are formed as `S.obs.cat <- factor(S.obs,levels=c(-Inf, quantile(c(S.0, S.1), c(.25, .5, .75), na.rm = TRUE), Inf))` and similarly for BIP.cat. Alternatively a user could input arbitrary numeric values to represent different discrete subgroups (e.g., 0s and 1s to denote 2 subgroups).

The "psdesign" object

We begin by creating a "psdesign" object with the synonymous function. This is the object that combines the raw dataset with information about the study design and the structure of the data. Subsequent analysis will operate on this psdesign object. It is designed to be analogous to the svydesign function in the **survey** package (Lumley, 2014, 2004). The first argument is the data frame where the data are stored. All subsequent arguments describe the mappings from the variable names in the data frame to important variables in the PS analysis, using the same notation as above. Other covariates or variables can be mapped to arbitrary variable names using the same syntax. An optional weights argument describes the sampling weights, if present. Our first analysis will use the binary version of the outcome, with continuous $S.1$ and the BIP labeled BIP. The object has a print method, so we can inspect the result.

```
binary.ps <- psdesign(data = fakedata, Z = Z, Y = Y.obs, S = S.obs, BIP = BIP)
binary.ps
```

```
## Augmented data frame:  800  obs. by  6  variables.
##   Z Y S.1     S.0 cdfweights    BIP
## 1 0 0  NA  0.3527          1  0.335
## 2 0 0  NA  1.4669          1  1.454
## 3 0 1  NA -0.7319          1 -0.724
## 4 0 0  NA -0.0183          1 -0.118
## 5 0 1  NA -0.1847          1 -0.235
## 6 0 0  NA -0.6631          1 -0.778
##
## Empirical TE:  0.526
##
## Mapped variables:
##   Z  ->  Z
##   Y  ->  Y.obs
##   S  ->  S.obs
##   BIP  ->  BIP
##
## Integration models:
##  None present, see ?add_integration for information on integration models.
##
```

```
## Risk models:
##  None present, see ?add_riskmodel for information on risk models.
##  No estimates present, see ?ps_estimate.
##  No bootstraps present, see ?ps_bootstrap.
```

The printout displays a brief description of the data, including the empirical treatment efficacy estimate, the variables used in the analysis and their corresponding variables in the original dataset. Finally the printout invites the user to see the help page for `add_integration`, in order to add an integration model to the psdesign object, the next step in the analysis.

Missing values in the S variable are allowed. Note that any cases where $S(1)$ is missing will be integrated over in the likelihood or score equations. Thus any cases that experienced an event prior to the time τ when the surrogate was measured should be excluded from the dataset. The equal individual risk assumption allows us to make causal inferences even after excluding such cases.

`psdesign` easily accommodates case-control or case-cohort sampling. In this case, the surrogate S is only measured on a subset of the data, inducing missingness in S by design. Let's modify the fake dataset to see how it works. We're going to sample all of the cases, and 20% of the controls for measurement of S.

```
fakedata.cc <- fakedata
missdex <- sample((1:nrow(fakedata.cc))[fakedata.cc$Y.obs == 0],
       size = floor(sum(fakedata.cc$Y.obs == 0) * .8))
fakedata.cc[missdex, ]$S.obs <- NA
fakedata.cc$weights <- ifelse(fakedata.cc$Y.obs == 1, 1, .2)
```

Now we can create the "psdesign" object, using the entire dataset (including those missing `S.obs`) and passing the weights to the `weights` field.

```
binary.cc <- psdesign(data = fakedata.cc, Z = Z, Y = Y.obs, S = S.obs,
                      BIP = BIP, weights = weights)
```

The other augmentation types can be defined by mapping variables to the names `BIP`, `CPV`, and/or `BSM`. The augmentations are handled as described in the previous section: CPV is used as a direct imputation for $S(1)$, and BSM is used as a direct imputation for $S(0)$. BIPs and BSMs are made available in the augmented dataset for use in the integration models which we describe in the next subsection.

For survival outcomes, a key assumption is that the potential surrogate is measured at a fixed time τ after entry into the study. Any subjects who have a clinical outcome prior to τ will be removed from the analysis, with a warning. If `tau` is not specified in the `psdesign` object, then it is assumed to be 0. Survival outcomes are specified by mapping `Y` to a `Surv` object, which requires the **survival** package:

```
surv.ps <- psdesign(data = fakedata, Z = Z, Y = Surv(time.obs, event.obs), S = S.obs,
                        BIP = BIP, CPV = CPV, BSM = BSM)

## Warning in psdesign(data = fakedata, Z = Z, Y = Surv(time.obs,
## event.obs), : tau missing in psdesign: assuming that the
## surrogate S was measured at time 0.
```

Integration models

The EML procedure requires an estimate of $F_{S(1)|W}$, and we refer to this as the integration model. Details are available in the help page for `add_integration`. Several integration models are implemented, including a parametric model that uses a formula interface to define a regression model, a semiparametric model that specifies a location and a scale model is robust to the specification of the distribution, and a non-parametric model that uses empirical conditional probability estimates for discrete W and $S(1)$.

For this first example, let's use the parametric integration model. We specify the mean model for $S(1)|W$ as a formula. The predictor is generally a function of the BIP and the BSM, if available. We can add the integration model directly to the psdesign object and inspect the results. Note that in the formula, we refer to the variable names in the augmented dataset.

```
binary.ps <- binary.ps + integrate_parametric(S.1 ~ BIP)
binary.ps

## Augmented data frame:  800  obs. by  6  variables.
##   Z Y S.1     S.0 cdfweights    BIP
```

```
## 1 0 0  NA  0.3527          1  0.335
## 2 0 0  NA  1.4669          1  1.454
## 3 0 1  NA -0.7319          1 -0.724
## 4 0 0  NA -0.0183          1 -0.118
## 5 0 1  NA -0.1847          1 -0.235
## 6 0 0  NA -0.6631          1 -0.778
##
## Empirical TE:  0.526
##
## Mapped variables:
##   Z  ->  Z
##   Y  ->  Y.obs
##   S  ->  S.obs
##   BIP  ->  BIP
##
## Integration models:
##   integration model for  S.1 :
##       integrate_parametric(formula = S.1 ~ BIP )
##
## Risk models:
##  None present, see ?add_riskmodel for information on risk models.
##  No estimates present, see ?ps_estimate.
##  No bootstraps present, see ?ps_bootstrap.
```

We can add multiple integration models to a psdesign object, say we want a model for $S(0)|W$:

```
binary.ps + integrate_parametric(S.0 ~ BIP)
```

```
## Augmented data frame:  800  obs. by  6  variables.
##   Z Y S.1     S.0 cdfweights    BIP
## 1 0 0  NA  0.3527          1  0.335
## 2 0 0  NA  1.4669          1  1.454
## 3 0 1  NA -0.7319          1 -0.724
## 4 0 0  NA -0.0183          1 -0.118
## 5 0 1  NA -0.1847          1 -0.235
## 6 0 0  NA -0.6631          1 -0.778
##
## Empirical TE:  0.526
##
## Mapped variables:
##   Z  ->  Z
##   Y  ->  Y.obs
##   S  ->  S.obs
##   BIP  ->  BIP
##
## Integration models:
##   integration model for  S.1 :
##       integrate_parametric(formula = S.1 ~ BIP )
##   integration model for  S.0 :
##       integrate_parametric(formula = S.0 ~ BIP )
##
## Risk models:
##  None present, see ?add_riskmodel for information on risk models.
##  No estimates present, see ?ps_estimate.
##  No bootstraps present, see ?ps_bootstrap.
```

In a future version of the package, we will allow for estimation of the joint risk estimands that depend on both $S(0)$ and $S(1)$. We can also use splines, other transformations, and other variables in the formula:

```
library(splines)
binary.ps + integrate_parametric(S.1 ~ BIP^2)
binary.ps + integrate_parametric(S.1 ~ bs(BIP, df = 3))
binary.ps + integrate_parametric(S.1 ~ BIP + BSM + BSM^2)
```

To include additional baseline covariates in the model for $S(1)$, such as age or gender, these variables that are present in the data frame must be mapped in the psdesign function call so that they are visible in the subsequent functions:

```
binary.ps <- psdesign(data = fakedata, Z = Z, Y = Y.obs, S = S.obs, BIP = BIP,
                      BSM = BSM, age = age)
binary.ps + integrate_parametric(S.1 ~ BIP + age)
```

These are shown as examples, we will proceed with the simple linear model for integration. The other integration models are called integrate_bivnorm, integrate_nonparametric, and integrate_semiparametric. See their help files for details on the models and their specification.

The next step is to define the risk model.

Risk models and likelihoods

The risk model is the specification of the distribution for the outcome Y given $S(1)$ and Z. We accommodate a variety of flexible specifications for this model, for continous, binary, time-to-event, and count outcomes. We have implemented exponential and weibull survival models, and a flexible specification for binary models, allowing for standard or custom link functions. See the help file for add_riskmodel for more details.

Let's add a simple binary risk model using the logit link. The argument D specifies the number of samples to use for the simulated annealing, also known as empirical integration, in the EML procedure. In general, D should be set to something reasonably large, like 2 or 3 times the sample size.

```
binary.ps <- binary.ps + risk_binary(model = Y ~ S.1 * Z, D = 50, risk = risk.logit)
binary.ps
```

```
## Augmented data frame:  800  obs. by  6  variables.
##   Z Y S.1     S.0 cdfweights    BIP
## 1 0 0  NA  0.3527          1  0.335
## 2 0 0  NA  1.4669          1  1.454
## 3 0 1  NA -0.7319          1 -0.724
## 4 0 0  NA -0.0183          1 -0.118
## 5 0 1  NA -0.1847          1 -0.235
## 6 0 0  NA -0.6631          1 -0.778
##
## Empirical TE:  0.526
##
## Mapped variables:
##   Z  ->  Z
##   Y  ->  Y.obs
##   S  ->  S.obs
##   BIP  ->  BIP
##
## Integration models:
##   integration model for  S.1 :
##       integrate_parametric(formula = S.1 ~ BIP )
##
## Risk models:
##   risk_binary(model = Y ~ S.1 * Z, D = 50, risk = risk.logit )
##
##  No estimates present, see ?ps_estimate.
##  No bootstraps present, see ?ps_bootstrap.
```

Estimation and bootstrap

We estimate the parameters and bootstrap using the same type of syntax. We can add a "ps_estimate" object, which takes optional arguments start for starting values, and other arguments that are passed to the optim function in base R. The method argument determines the optimization method, we have found that "BFGS" works well in these types of problems and it is the default. Use "pseudo-score" as the method argument for pseudo-score estimation for binary risk models with categorical BIPs.

The ps_bootstrap function takes the additional arguments n.boots for the number of bootstrap replicates, and progress.bar which is a logical that displays a progress bar in the R console if true.

It is helpful to pass the estimates as starting values in the bootstrap resampling. With estimates and bootstrap replicates present, printing the psdesign object displays additional information.

```
binary.est <- binary.ps + ps_estimate(method = "BFGS")
binary.boot <- binary.est + ps_bootstrap(n.boots = 500, progress.bar = FALSE,
                          start = binary.est$estimates$par, method = "BFGS")
binary.boot
```

```
## Augmented data frame:  800  obs. by  6  variables.
##   Z Y S.1     S.0 cdfweights    BIP
## 1 0 0  NA  0.3527          1  0.335
## 2 0 0  NA  1.4669          1  1.454
## 3 0 1  NA -0.7319          1 -0.724
## 4 0 0  NA -0.0183          1 -0.118
## 5 0 1  NA -0.1847          1 -0.235
## 6 0 0  NA -0.6631          1 -0.778
##
## Empirical TE:  0.526
##
## Mapped variables:
##   Z  ->  Z
##   Y  ->  Y.obs
##   S  ->  S.obs
##   BIP  ->  BIP
##
## Integration models:
##   integration model for  S.1 :
##       integrate_parametric(formula = S.1 ~ BIP )
##
## Risk models:
##   risk_binary(model = Y ~ S.1 * Z, D = 50, risk = risk.logit )
##
## Estimated parameters:
## (Intercept)         S.1           Z       S.1:Z
##      -0.920      -0.028      -0.220      -1.133
##  Convergence:  TRUE
##
## Bootstrap replicates:
##             Estimate boot.se lower.CL.2.5. upper.CL.97.5.
## (Intercept)   -0.920   0.182        -1.286         -0.580
## S.1           -0.028   0.128        -0.276          0.220
## Z             -0.220   0.250        -0.697          0.277
## S.1:Z         -1.133   0.214        -1.581         -0.780
##              p.value
## (Intercept) 4.02e-07
## S.1         8.27e-01
## Z           3.80e-01
## S.1:Z       1.29e-07
##
##   Out of 500 bootstraps,  500 converged ( 100 %)
##
##   Test for wide effect modification on 1 degree of freedom. 2-sided p value  < .0001
```

Do it all at once

The next code chunk shows how the model can be defined and estimated all at once.

```
binary.est <- psdesign(data = fakedata, Z = Z, Y = Y.obs, S = S.obs, BIP = BIP) +
  integrate_parametric(S.1 ~ BIP) +
  risk_binary(model = Y ~ S.1 * Z, D = 50, risk = risk.logit) +
  ps_estimate(method = "BFGS")
```

Plots and summaries

We provide summary and plotting methods for the psdesign object. If bootstrap replicates are present, the summary method does a test for wide effect modification. Under the parametric risk models implemented in this package, the test for wide effect modification is equivalent to a test of the null hypothesis that the $S(1) : Z$ coefficient is equal to 0. This is implemented using a Wald test using the bootstrap estimate of the variance.

Another way to assess wide effect modification is to compute the standardized total gain (STG) (Huang and Gilbert, 2011) and (Gabriel et al., 2015). This is implemented in the `calc_STG` function. The standardized total gain can be interpreted as the area sandwiched between the risk difference curve and the horizontal line at the marginal risk difference. It is a measure of the spread of the distribution of the risk difference, and is a less parametric way to test for wide effect modification. The `calc_STG` function computes the STG at the estimated parameters and at the bootstrap samples, if present. The function prints the results and invisibly returns a list containing the observed STG, and the bootstrapped STGS.

```
calc_STG(binary.boot, progress.bar = FALSE)
```

```
## $obsSTG
## [1] 0.3397774
##
## $bootstraps
##    STG.boot.se STG.lower.CL.2.5 STG.upper.CL.97.5
## V1   0.1243311        0.1573031         0.6382418
```

The summary method also computes the marginal treatment efficacy marginalized over $S(1)$ and compares it to the average treatment efficacy conditional on $S(1)$. This is an assessment of model fit. A warning will be given if the two estimates are dramatically different. These estimates are presented in the summary along with the empirical TE and the model-based marginal treatment efficacy that does not condition on S(1).

```
smary <- summary(binary.boot)
```

```
## Augmented data frame:  800  obs. by  6  variables.
##   Z Y S.1     S.0 cdfweights    BIP
## 1 0 0  NA  0.3527          1  0.335
## 2 0 0  NA  1.4669          1  1.454
## 3 0 1  NA -0.7319          1 -0.724
## 4 0 0  NA -0.0183          1 -0.118
## 5 0 1  NA -0.1847          1 -0.235
## 6 0 0  NA -0.6631          1 -0.778
##
## Empirical TE:  0.526
##
## Mapped variables:
##   Z  ->  Z
##   Y  ->  Y.obs
##   S  ->  S.obs
##   BIP  ->  BIP
##
## Integration models:
##   integration model for  S.1 :
##       integrate_parametric(formula = S.1 ~ BIP )
##
## Risk models:
##   risk_binary(model = Y ~ S.1 * Z, D = 50, risk = risk.logit )
##
## Estimated parameters:
## (Intercept)         S.1           Z       S.1:Z
##      -0.920      -0.028      -0.220      -1.133
##  Convergence:  TRUE
##
## Bootstrap replicates:
##             Estimate boot.se lower.CL.2.5. upper.CL.97.5.
## (Intercept)   -0.920   0.182        -1.286         -0.580
```

```
## S.1           -0.028   0.128        -0.276          0.220
## Z             -0.220   0.250        -0.697          0.277
## S.1:Z         -1.133   0.214        -1.581         -0.780
##              p.value
## (Intercept) 4.02e-07
## S.1         8.27e-01
## Z           3.80e-01
## S.1:Z       1.29e-07
##
##   Out of 500 bootstraps,  500 converged ( 100 %)
##
##   Test for wide effect modification on 1 degree of freedom. 2-sided p value  < .0001
##
## Treatment Efficacy:
## empirical  marginal     model
##     0.526     0.526     0.539
##  Model-based average TE is 2.3 % different from the empirical and 2.3 % different
##     from the marginal.
```

The `calc_risk` function computes the risk in each treatment arm, and contrasts of the risks. By default it computes the treatment efficacy, but there are other contrast functions available. The contrast function is a function that takes 2 inputs, the $risk_0$ and $risk_1$, and returns some one dimensional function of those two inputs. It must be vectorized. Some built-in functions are "TE" for treatment efficacy $= 1 - risk_1(s)/risk_0(s)$, "RR" for relative risk $= risk_1(s)/risk_0(s)$, "logRR" for log of the relative risk, and "RD" for the risk difference $= risk_1(s) - risk_0(s)$. You can pass the name of the function, or the function itself to `calc_risk`. See `?calc_risk` for more information about contrast functions.

Other arguments of the `calc_risk` function include `t`, the time at which to calculate the risk for time-to-event outcomes, `n.samps` which is the number of samples over the range of S.1 at which the risk will be calculated, and `CI.type`, which can be set to `"pointwise"` for pointwise confidence intervals or `"band"` for a simultaneous confidence band. `sig.level` is the significance level for the bootstrap confidence intervals. If the outcome is time-to-event and t is not present, then it will use the restricted mean survival time.

```
head(calc_risk(binary.boot, contrast = "TE", n.samps = 20), 3)
```

```
##           S.1          Y        R0        R1 Y.boot.se
## V1 -2.2756987 -1.7437221 0.2980453 0.8177536 1.1622104
## V2 -1.4262708 -1.1360482 0.2930970 0.6260692 0.6957994
## V3 -0.5973759 -0.3532827 0.2883149 0.3901715 0.3328793
##    Y.upper.CL.0.95 Y.lower.CL.0.95 R0.boot.se R0.upper.CL.0.95
## V1     -0.30455106       -3.780389 0.08994238        0.4766278
## V2     -0.05541741       -2.556452 0.06901664        0.4331237
## V3      0.29675970       -1.275280 0.04941685        0.4007098
##    R0.lower.CL.0.95 R1.boot.se R1.upper.CL.0.95 R1.lower.CL.0.95
## V1        0.1188411 0.06911306        0.9368827        0.6720409
## V2        0.1468385 0.07592515        0.7762517        0.4768403
## V3        0.1734875 0.05079824        0.5248493        0.2834909
```

```
head(calc_risk(binary.boot, contrast = function(R0, R1) 1 - R1/R0, n.samps = 20), 3)
```

```
##            S.1           Y        R0        R1 Y.boot.se
## V1 -0.97417991 -0.71327775 0.2904830 0.4976780 0.4840781
## V2 -0.11875337  0.05966359 0.2855748 0.2685364 0.1882398
## V3 -0.09236484  0.08009338 0.2854242 0.2625636 0.1820450
##    Y.upper.CL.0.95 Y.lower.CL.0.95 R0.boot.se R0.upper.CL.0.95
## V1       0.1395560      -1.6775172 0.05815888        0.4084405
## V2       0.4746753      -0.4357974 0.03903555        0.3909614
## V3       0.4835707      -0.4036444 0.03849822        0.3904263
##    R0.lower.CL.0.95 R1.boot.se R1.upper.CL.0.95 R1.lower.CL.0.95
## V1        0.1763127 0.06531273        0.6258814        0.3695242
## V2        0.2043944 0.03260224        0.3454742        0.1923057
## V3        0.2053113 0.03176977        0.3379838        0.1879937
```

It is easy to plot the risk estimates. By default, the plot method displays the TE contrast, but this can be changed using the same syntax as in `calc_risk`.

```
plot(binary.boot, contrast = "TE", lwd = 2)
abline(h = smary$TE.estimates[2], lty = 3)

expit <- function(x) exp(x)/(1 + exp(x))
trueTE <- function(s){

  r0 <- expit(-1 - 0 * s)
  r1 <- expit(-1 - 1.25 * s)
  1 - r1/r0

}

rug(binary.boot$augdata$S.1)
curve(trueTE(x), add = TRUE, col = "red")
legend("bottomright", legend = c("estimated TE", "95\\% CB",
                                 "marginal TE", "true TE"),
       col = c("black", "black", "black", "red"),
       lty = c(1, 2, 3, 1), lwd = c(2, 2, 1, 1))
```

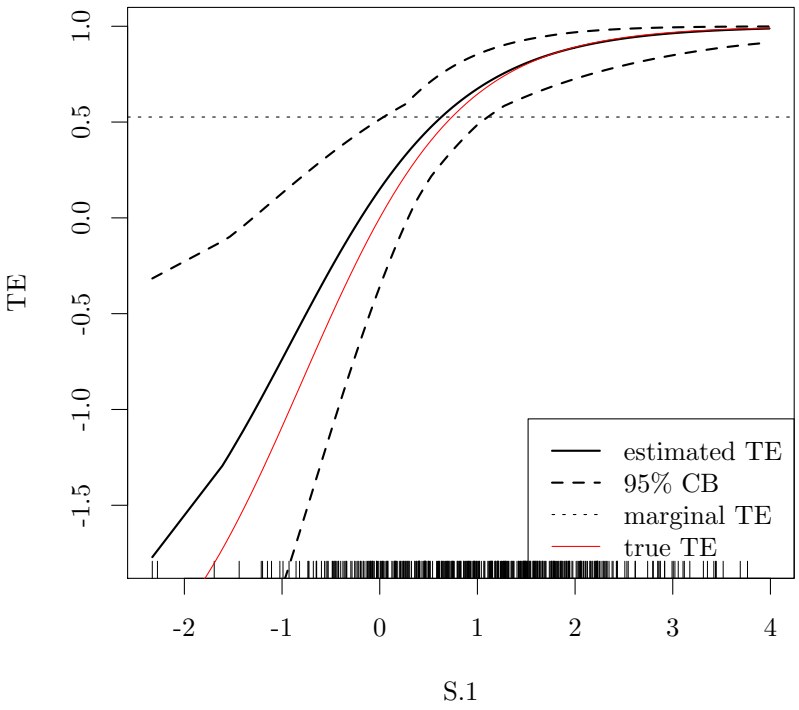

Figure 1: Plot showing the estimates using the example data, along with confidence bands (CB), and the true treatment efficacy (TE) curve.

By default, plots of psdesign objects with bootstrap samples will display simultaneous confidence bands for the curve. These bands $L\alpha$ satisfy

$$P\left\{\sup_{s\in B}|\hat{TE}(s) - TE(s)| \leq L_\alpha\right\} \leq 1-\alpha,$$

for confidence level α. The alternative is to use pointwise confidence intervals, with the option `CI.type = "pointwise"`. These intervals satisfy

$$P\left\{\hat{L}_\alpha \leq TE(s) \leq \hat{U}_\alpha\right\} \leq 1-\alpha, \text{ for all } s.$$

Different summary measures are available for plotting. The options are "TE" for treatment efficacy $= 1 - risk_1(s)/risk_0(s)$, "RR" for relative risk $= risk_1(s)/risk_0(s)$, "logRR" for log of the relative risk, "risk" for the risk in each treatment arm, and "RD" for the risk difference $= risk_1(s) - risk_0(s)$. We can also transform using the `log` option of `plot`.

```
plot(binary.boot, contrast = "logRR", lwd = 2,
      col = c("black", "grey75", "grey75"))
plot(binary.boot, contrast = "RR", log = "y", lwd = 2,
      col = c("black", "grey75", "grey75"))
plot(binary.boot, contrast = "RD", lwd = 2,
      col = c("black", "grey75", "grey75"))
plot(binary.boot, contrast = "risk", lwd = 2, lty = c(1, 0, 0, 2, 0, 0))
legend("topright", legend = c("R0", "R1"), lty = c(1, 2), lwd = 2)
```

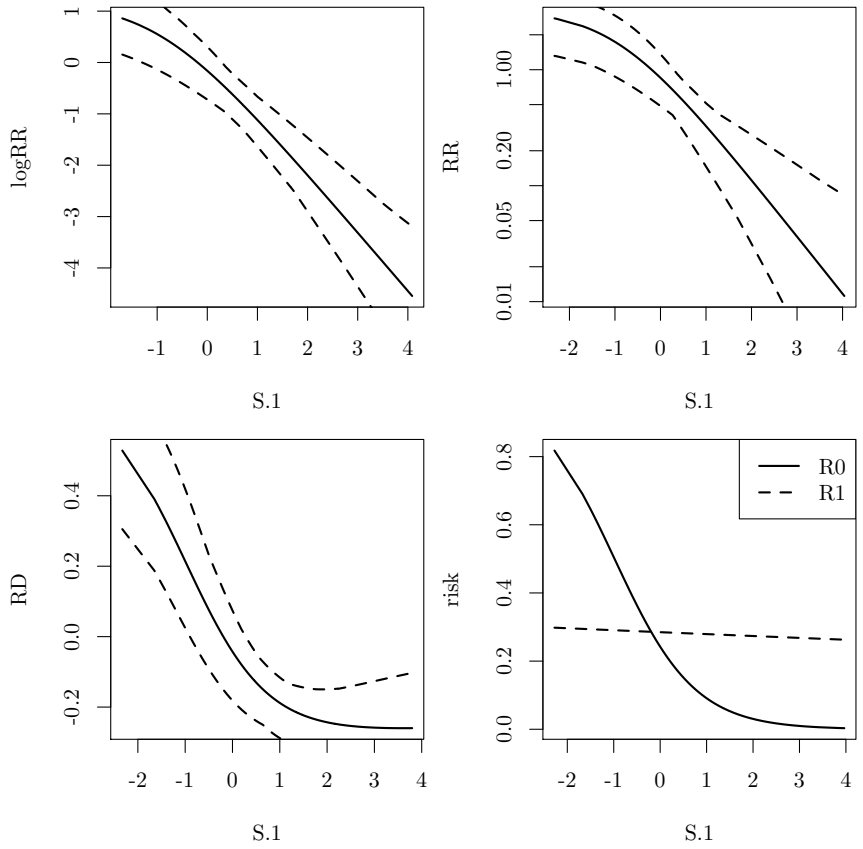

Figure 2: Plot illustrating ways that different risk contrast functions can be plotted.

The `calc_risk` function is the workhorse that creates the plots. You can call this function directly to obtain estimates, standard errors, and confidence intervals for the estimated risk in each treatment arm and transformations of the risk like TE. The parameter `n.samps` determines the number of points at which to calculate the risk. The points are evenly spaced over the range of S.1. Use this function to compute other summaries, make plots using **ggplot2** (Wickham, 2009) or **lattice** (Sarkar, 2008) and more.

```
te.est <- calc_risk(binary.boot, CI.type = "pointwise", n.samps = 200)
head(te.est, 3)
```

```
##          S.1         Y        R0        R1 Y.boot.se
## V1 -2.328509 -1.770899 0.2983546 0.8267105 1.1943792
## V2 -2.275699 -1.743722 0.2980453 0.8177536 1.1622104
## V3 -1.694556 -1.360959 0.2946547 0.6956675 0.8334835
##    Y.lower.CL.2.5 Y.upper.CL.97.5 R0.boot.se R0.lower.CL.2.5
## V1      -4.768551      -0.6276106 0.09125321       0.1460172
## V2      -4.671015      -0.6184582 0.08994238       0.1475266
## V3      -3.456249      -0.4524787 0.07557692       0.1648504
```

```
##    R0.upper.CL.97.5 R1.boot.se R1.lower.CL.2.5 R1.upper.CL.97.5
## V1        0.4890998 0.06786962       0.6663453        0.9237321
## V2        0.4856823 0.06911306       0.6556765        0.9183559
## V3        0.4513863 0.07721039       0.5287710        0.8307616
```

Summary and conclusion

We have implemented the core methods for principal surrogate evaluation in our **pseval** package. Our aim was to create a flexible and consistent user interface that allows for the estimation of a wide variety of statistical models in this framework. There has been some other work in this area. The **Surrogate** package implements the core methods for the evaluation of trial-level surrogates using a meta-analytic framework. It also has a wide variety of models, each implemented in a different function each with a long list of parameters (der Elst et al., 2016).

Our package uses the + sign to combine function calls into a single object. This is called "overloading the + operator" and is most famously known from the **ggplot2** package. Conceptually, this was appealing to us because it allows users to build up analysis objects starting from the design, and ending with the estimation. The distinct analysis concepts of the design, risk model specification, integration model, and estimation/bootstrap approaches are separated into distinct function calls, each with a limited number of parameters. This makes it easier for users to keep track of their models, makes it easier to understand the methods involved, and allows for the specification of a wide variety of models by mixing and matching the function calls. This framework will also make it easier to maintain the codebase, and to extend it in the future as the methods evolve. Our package is useful for novice and expert R users alike, and implements an important set of statistical methods for the first time.

Appendix

Additional examples

Plot both types of CI

```
plot(binary.boot, contrast = "TE", lwd = 2, CI.type = "band")
sbs <- calc_risk(binary.boot, CI.type = "pointwise", n.samps = 200)
lines(Y.lower.CL.2.5 ~ S.1, data = sbs, lty = 3, lwd = 2)
lines(Y.upper.CL.97.5 ~ S.1, data = sbs, lty = 3, lwd = 2)
legend("bottomright", lwd = 2, lty = 1:3,
       legend = c("estimate", "simultaneous CI", "pointwise CI"))
```

Plot with ggplot2

```
library(ggplot2)
TE.est <- calc_risk(binary.boot, n.samps = 200)
ggplot(TE.est,
       aes(x = S.1, y = Y, ymin = Y.lower.CL.0.95, ymax = Y.upper.CL.0.95)) +
  geom_line() + geom_ribbon(alpha = .2) + ylab(attr(TE.est, "Y.function"))
```

Case-control design

```
cc.fit <- binary.cc + integrate_parametric(S.1 ~ BIP) +
  risk_binary(D = 10) + ps_estimate()
cc.fit
```

Survival outcome

```
surv.fit <- psdesign(fakedata, Z = Z, Y = Surv(time.obs, event.obs),
                     S = S.obs, BIP = BIP, CPV = CPV) +
  integrate_semiparametric(formula.location = S.1 ~ BIP, formula.scale = S.1 ~ 1) +
  risk_exponential(D = 10) + ps_estimate(method = "BFGS") + ps_bootstrap(n.boots = 20)
surv.fit
plot(surv.fit)
```

Continuous outcome

```
fakedata$Y.cont <- log(fakedata$time.obs + 0.01)
cont.fit <- psdesign(fakedata, Z = Z, Y = Y.cont,
                     S = S.obs, BIP = BIP, CPV = CPV) +
  integrate_semiparametric(formula.location = S.1 ~ BIP, formula.scale = S.1 ~ 1) +
  risk_continuous(D = 10) + ps_estimate(method = "BFGS") + ps_bootstrap(n.boots = 20)
cont.fit
plot(cont.fit, contrast = "risk")
```

Categorical S

`S.obs.cat` and `BIP.cat` are factors:

```
with(fakedata, table(S.obs.cat, BIP.cat))

cat.fit <- psdesign(fakedata, Z = Z, Y = Y.obs,
                     S = S.obs.cat, BIP = BIP.cat) +
  integrate_nonparametric(formula = S.1 ~ BIP) +
  risk_binary(Y ~ S.1 * Z, D = 10, risk = risk.probit) + ps_estimate(method = "BFGS")
cat.fit
plot(cat.fit)
```

Pseudo-score

Categorical W allows for estimation of the model using the pseudo-score method for binary outcomes. *S* may be continuous or categorical:

```
cat.fit.ps <- psdesign(fakedata, Z = Z, Y = Y.obs,
                     S = S.obs, BIP = BIP.cat) +
  integrate_nonparametric(formula = S.1 ~ BIP) +
  risk_binary(Y ~ S.1 * Z, D = 10, risk = risk.logit) +
  ps_estimate(method = "pseudo-score") +
  ps_bootstrap(n.boots = 20, method = "pseudo-score")
summary(cat.fit.ps)
plot(cat.fit.ps)
```

Bug reports

- Please file bugs and suggestions here as a github issue: https://github.com/sachsmc/pseval/issues.

Bibliography

W. V. der Elst, P. Meyvisch, A. Alonso, H. M. Ensor, and C. J. W. . G. Molenberghs. *Surrogate: Evaluation of Surrogate Endpoints in Clinical Trials*, 2016. URL https://CRAN.R-project.org/package=Surrogate. R package version 0.1-79. [p290]

D. Follmann. Augmented designs to assess immune response in vaccine trials. *Biometrics*, 62(4): 1161–1169, 2006. [p279]

C. Frangakis and D. Rubin. Principal stratification in causal inference. *Biometrics*, 58(1):21–29, 2002. [p277, 278]

E. E. Gabriel and D. Follmann. Augmented trial designs for evaluation of principal surrogates. *Biostatistics*, 17(3):453–467, 2016. [p278, 279]

E. E. Gabriel and P. B. Gilbert. Evaluating principal surrogate endpoints with time-to-event data accounting for time-varying treatment efficacy. *Biostatistics*, 15(2):251–265, 2014. [p278, 279]

E. E. Gabriel, M. C. Sachs, and P. B. Gilbert. Comparing and combining biomarkers as principle surrogates for time-to-event clinical endpoints. *Statistics in Medicine*, 34(3):381–395, 2015. ISSN 1097-0258. doi: 10.1002/sim.6349. [p286]

P. Gilbert and M. Hudgens. Evaluating candidate principal surrogate endpoints. *Biometrics*, 64(4): 1146–1154, 2008. [p278, 279, 280]

Y. Huang and P. B. Gilbert. Comparing biomarkers as principal surrogate endpoints. *Biometrics*, 67(4): 1442–1451, 2011. [p278, 280, 286]

Y. Huang, P. B. Gilbert, and J. Wolfson. Design and estimation for evaluating principal surrogate markers in vaccine trials. *Biometrics*, 69(2):301–309, 2013. [p278, 280]

T. Lumley. Analysis of complex survey samples. *Journal of Statistical Software*, 9(1):1–19, 2004. [p281]

T. Lumley. *Survey: analysis of complex survey samples*, 2014. R package version 3.30. [p281]

M. Pepe and T. Fleming. A nonparametric method for dealing with mismeasured covariate data. *Journal of the American Statistical Association*, 86(413):108–113, 1991. [p280]

S. A. Plotkin and P. B. Gilbert. Nomenclature for immune correlates of protection after vaccination. *Clinical Infectious Diseases*, 54(11):1615–1617, 2012. [p277]

M. C. Sachs and E. E. Gabriel. *pseval: Methods for Evaluating Principal Surrogates of Treatment Response*, 2016. R package version 1.3.0. [p277]

D. Sarkar. *Lattice: Multivariate Data Visualization with R*. Springer, New York, 2008. URL http://lmdvr.r-forge.r-project.org. ISBN 978-0-387-75968-5. [p289]

T. M. Therneau. *A Package for Survival Analysis in S*, 2015. R package version 2.38. [p280]

T. M. Therneau and P. M. Grambsch. *ModModel Survival Data: Extending the Cox Model*. Springer, New York, 2000. [p280]

H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2009. ISBN 978-0-387-98140-6. URL http://had.co.nz/ggplot2/book. [p289]

J. Wolfson. *Statistical Methods for Identifying Surrogate Endpoints in Vaccine Trials*. Doctor of philosophy dissertation, University of Washington; Department of Biostatistics, Chair: Gilbert, Peter, 2009. [p280]

J. Wolfson and P. Gilbert. Statistical identifiability and the surrogate endpoint problem, with application to vaccine trials. *Biometrics*, 66(4):1153–1161, 2010. [p278, 279]

Michael C Sachs
Unit of Biostatistics, Institute of Environmental Medicine, Karolinska Institute
Nobel väg 13
17165 Stockholm, Sweden
michael.sachs@ki.se

Erin E Gabriel
Biostatistics Research Branch, National Institute of Allergy and Infectious Disease
5601 Fishers Lane
Rockville, MD 20892
USA
erin.gabriel@nih.gov