Understanding Semiparametric Constraints

Caleb Leedy

19 January 2024

Introduction

The goal of this report is to understand how additional constraints effect semiparametric models. Inspired by Dr. Fuller's note, I set out to see if we could assess the amount of efficiency loss due to using semiparametric inference. First, let's define some notation.

Table 1: This table shows the relationship between different quantities of interest in each segment, their estimators, and the corresponding coefficients of their estimators.

Segment	Quantity of Interest	Estimator	Coefficient
A_{00}	$E[g \mid X]$	$\hat{\gamma}_{11}$	c_{11}
A_{10}	$E[g \mid X]$	$\hat{\gamma}_{21}$	c_{21}
A_{10}	$E[g \mid X, Y_1]$	$\hat{\gamma}_{22}$	c_{22}
A_{01}	$E[g \mid X]$	$\hat{\gamma}_{31}$	c_{31}
A_{01}	$E[g \mid X, Y_2]$	$\hat{\gamma}_{33}$	c_{33}
A_{11}	$E[g \mid X]$	$\hat{\gamma}_{41}$	c_{41}
A_{11}	$E[g \mid X, Y_1]$	$\hat{\gamma}_{42}$	c_{42}
A_{11}	$E[g \mid X, Y_2]$	$\hat{\gamma}_{43}$	c_{43}
A_{11}	$E[g \mid X, Y_1, Y_2]$	$\hat{\gamma}_{44}$	c_{44}

This is a similar framework to understanding the class of estimators that we discussed in `optest_update.pdf`, where we write the estimator as a weighted average of different expectation with functional coefficients.

$$\hat{\theta} = \frac{\delta_{11}}{\pi_{11}} g(Z) + \beta_0(\delta, c_0) E[g(Z) \mid X] + \beta_1(\delta, c_1) E[g(Z) \mid X, Y_1] + \beta_2(\delta, c_2) E[g(Z) \mid X, Y_2].$$

However, now we write the estimator as

$$\hat{\theta} = \sum_{k,t} c_{kt} \hat{\gamma}_{kt}$$

where $\hat{\gamma}_{kt} = \frac{\delta_{ij}}{\pi_{ij}} E[g \mid G_{ij}(X, Y_1, Y_2)]$ and ij corresponds to the segment A_{ij} associated with $\hat{\gamma}_{kt}$ in Table 1.

Methods

As noted by Dr. Fuller, we can understand parametric estimation as solving the following constrained optimization problems:

$$\text{minVar}\left(\sum_{k,t} c_{kt}\hat{\gamma}_{kt}\right) \text{ such that } \sum_{k,t} c_{kt} = 1$$

or

$$\text{minVar}\left(\sum_{k,t} c_{kt}\hat{\gamma}_{kt}\right) \text{ such that } \sum_{(k,t):(k,t)\neq(4,4)} c_{kt} = 0 \text{ and } c_{44} = 1.$$

To be robust to the outcome model, we can construct a similar optimization problem with different constraints:

$$\min \text{Var}\left(\sum_{k,t} c_{kt}\hat{\gamma}_{kt}\right) \text{ such that } c_{11} + c_{21} + c_{31} + c_{41} = 0, c_{22} + c_{42} = 0, c_{33} + c_{43} = 0, \text{ and } c_{44} = 1.$$

Likewise to be robust to the response model, we can have the problem:

$$\min \text{Var}\left(\sum_{k,t} c_{kt}\hat{\gamma}_{kt}\right) \text{ such that } c_{11} = \pi_{00}, c_{21} + c_{22} = \pi_{10}, c_{31} + c_{33} = \pi_{01}, \text{ and } c_{41} + c_{42} + c_{43} + c_{44} = \pi_{11}.$$

To be double robust (robust to the outcome and response model) we can combine the last two constraints. This is summarized in Table 9.

Table 2: This table identifies the different constraints for each model type.

Type	Constraints
Parametric 1	$\sum_{k,t} c_{kt} = 1$
Parametric 2	$\sum_{k,t:(k,t)\neq(4,4)} c_{kt} = 0, c_{44} = 1$
Outcome Robust	$c_{11}+c_{21}+c_{31}+c_{41} = 0, c_{22}+c_{42} = 0, c_{33}+c_{43} = 0,$ and $c_{44} = 1.$
Response Robust	$c_{11} = \pi_{00}, c_{21} + c_{22} = \pi_{10}, c_{31} + c_{33} =$ $\pi_{01},$ and $c_{41} + c_{42} + c_{43} + c_{44} = \pi_{11}$
Double Robust	$c_{11}+c_{21}+c_{31}+c_{41} = 0, c_{22}+c_{42} = 0, c_{33}+c_{43} = 0, c_{44} = 1, c_{11} =$ $\pi_{00}, c_{21}+c_{22} = \pi_{10}, c_{31}+c_{33} = \pi_{01},$ and $c_{41}+c_{42}+c_{43}+c_{44} = \pi_{11}$

Simulation Study

Simulation 1

We use the following simulation setup

$$\begin{bmatrix} x \\ e_1 \\ e_2 \end{bmatrix} \overset{ind}{\sim} N\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & \rho \\ 0 & \rho & 1 \end{bmatrix} \right)$$
$$y_1 = x + e_1$$
$$y_2 = \theta + x + e_2$$

Furthermore, $\pi_{11} = 0.2$ and $\pi_{00} = 0.3, \pi_{10} = 0.4$, and $\pi_{01} = 0.1$. The goal of this simulation study is to find $\theta = E[Y_2]$. In other words, $g(Z) = Y_2$. There are several algorithms for comparison which are defined as the following:

$$Oracle = n^{-1} \sum_{i=1}^{n} g(Z_i)$$
$$CC = \frac{\sum_{i=1}^{n} \delta_{11} g(Z_i)}{\sum_{i=1}^{n} \delta_{11}}$$
$$IPW = \sum_{i=1}^{n} \frac{\delta_{11}}{\pi_{11}} g(Z_i)$$

Furthermore, we include four existing estimators that we have already proposed in the past: WLS, Prop, PropInd, and SemiDelta.

The estimator WLS is a a weight least square estimator derived in the following manner. We now consider a normal model:

$$\begin{pmatrix} x_i \\ e_{1i} \\ e_{2i} \end{pmatrix} \overset{ind}{\sim} N\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 0 & \sigma_{11} & \sigma_{12} \\ 0 & \sigma_{12} & \sigma_{22} \end{bmatrix}, \right)$$

and define $y_{1i} = \theta_1 + x_i + e_{1i}$ and $y_{2i} = \theta_2 + x_i + e_{2i}$. Then $b_1 = b_2 = 1$. We define $\bar{z}_k^{(ij)}$ as the mean of y_k in segment A_{ij}. This means that we have means $\bar{z}_1^{(11)}$, $\bar{z}_2^{(11)}$, $\bar{z}_1^{(10)}$, and $\bar{z}_2^{(01)}$. Let $W = [\bar{z}_1^{(11)}, \bar{z}_2^{(11)}, \bar{z}_1^{(10)}, \bar{z}_2^{(01)}]'$, then for $n_{ij} = |A_{ij}|$, we have

$$Z - M\mu \sim N(\vec{0}, V)$$

where

$$M = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ and } V = \begin{bmatrix} \frac{\sigma_{11}}{n_{11}} & \frac{\sigma_{12}}{n_{11}} & 0 & 0 \\ \frac{\sigma_{12}}{n_{11}} & \frac{\sigma_{22}}{n_{11}} & 0 & 0 \\ 0 & 0 & \frac{\sigma_{11}}{n_{10}} & 0 \\ 0 & 0 & 0 & \frac{\sigma_{22}}{n_{01}} \end{bmatrix}.$$

Thus, the BLUE for $\mu = [\mu_1, \mu_2]'$ is

$$\hat{\mu} = (M'V^{-1}M)^{-1}M'V^{-1}W.$$

Hence, WLS is μ_2 as $g(X, Y_1, Y_2) = Y_2$.

The remaining three estimators are drived from the following expression where the values for β are provided in Table 3. These are good estimators to compare to because Prop is the original proposed estimator. PropInd has the same form as Prop except the values for β is different. PropInd shares the same values of β as all of the new models with constraints. SemiDelta is useful because it is the best estimator in general so far.

$$\hat{\theta} = \frac{\delta_{11}}{\pi_{11}} g(Z) + \beta_0(\delta, c_0)E[g(Z) \mid X] + \beta_1(\delta, c_1)E[g(Z) \mid X, Y_1] + \beta_2(\delta, c_2)E[g(Z) \mid X, Y_2].$$

Table 3: This table displays the values of β for different estimator types.

Estimator	$\beta_0(\delta, c_0)$	$\beta_1(\delta, c_1)$
Prop	$\left(1 - \frac{(\delta_{10}+\delta_{11})}{(\pi_{10}+\pi_{11})} - \frac{(\delta_{01}+\delta_{11})}{(\pi_{01}+\pi_{11})} + \frac{\delta_{11}}{\pi_{11}}\right)$	$\left(\frac{\delta_{10}+\delta_{11}}{\pi_{10}+\pi_{11}} - \frac{\delta_{11}}{\pi_{11}}\right)$
PropInd	$\left(1 - \frac{(\delta_{10})}{(\pi_{10})} - \frac{(\delta_{01})}{(\pi_{01})} + \frac{\delta_{11}}{\pi_{11}}\right)$	$\left(\frac{\delta_{10}}{\pi_{10}} - \frac{\delta_{11}}{\pi_{11}}\right)$
SemiDelta	$c_0 \left(\frac{\delta_{11}}{\pi_{11}} - \frac{\delta_{00}}{\pi_{00}}\right)$	$c_1 \left(\frac{\delta_{11}}{\pi_{11}} - \frac{\delta_{10}}{\pi_{10}}\right)$

In the simulation results in Table 4, the new results have the same label as the value from the `Type` column in Table 9.

Table 4: Results from Simulation 1. True values: $\theta = 5, \rho = 0.5$. The test conducted for the T-statistic and P-value is a two sample test to see if the estimator is unbiased. This simulation uses $Y_1 = x + \varepsilon_1$, $Y_2 = \theta + x + \varepsilon_2$ where $X \sim N(0, 1)$ and $(\varepsilon_1, \varepsilon_2)$ come from a mean zero bivariate normal distribution with unit variance and covariance ρ. The segments are unbalanced with: $\pi_{11} = 0.2$, $\pi_{10} = 0.4$, $\pi_{01} = 0.1$, and $\pi_{00} = 0.3$.

Algorithm	bias	sd	tstat	pval
Oracle	-0.002	0.045	-1.669	0.048
CC	0.001	0.083	0.263	0.396
IPW	0.010	0.357	0.872	0.192
WLS	0.000	0.054	-0.065	0.474
$\hat{\theta}_{prop}$	-0.002	0.063	-0.768	0.221
$\hat{\theta}_{\delta}$	-0.001	0.064	-0.342	0.366
Parametric 1	-0.001	0.061	-0.509	0.305
Parametric 2	-0.001	0.061	-0.509	0.305
Outcome Robust	-0.001	0.061	-0.543	0.294
Response Robust	-0.001	0.061	-0.509	0.305
Double Robust	-0.001	0.061	-0.543	0.294

Simulation 2

Next, we used the same simulation setup except that we are interested in estimating $\theta = E[g] = E[Y_1^2 Y_2]$. We do not use the WLS estimator because it does not work.

Table 5: Results from Simulation 2. True values: $\theta = 10, \rho = 0.5$. The test conducted for the T-statistic and P-value is a two sample test to see if the estimator is unbiased. This simulation uses the same setup as Simulation 1 with $Y_1 = x + \varepsilon_1$, $Y_2 = \theta + x + \varepsilon_2$ where $X \sim N(0, 1)$ and $(\varepsilon_1, \varepsilon_2)$ come from a mean zero bivariate normal distribution with unit variance and covariance ρ. The segments are unbalanced with: $\pi_{11} = 0.2$, $\pi_{10} = 0.4$, $\pi_{01} = 0.1$, and $\pi_{00} = 0.3$.

Algorithm	bias	sd	tstat	pval
Oracle	-0.037	0.528	-2.214	0.014
CC	0.007	1.196	0.179	0.429
IPW	0.023	1.405	0.520	0.302
$\widehat{\theta}_{prop}$	-0.044	0.691	-2.027	0.021
$\widehat{\theta}_{\delta}$	-0.032	0.674	-1.521	0.064
Parametric 1	-0.004	0.301	-0.461	0.322
Parametric 2	-0.004	0.344	-0.344	0.366
Outcome Robust	-0.040	0.654	-1.950	0.026
Response Robust	-0.004	0.301	-0.461	0.322
Double Robust	-0.040	0.654	-1.950	0.026

Discussion

There has been concern about why the constrained results are the same in Table 4. This section aims to address these concerns.

Interestingly enough, the optimization algorithm does not have a lot of variability in the output. The variance for each coefficient is found in Table 6. Since these values are all basically zero, we can see that for each algorithm we are getting basically the same coefficients each iteration. The coefficients that we see are displayed in Table 7.

Table 6: This table displays the variance of each coefficient for each estimator. A number close to zero means that the optimization algorithm chose very similar values for each Monte Carlo simulation. A value of exactly zero indicated that the optimization algorithm chose the exact same number.

Estimator	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9
Para. 1	0	0	0	0	0	0	0	0	0
Para. 2	0	0	0	0	0	0	0	0	0
Outcome	0	0	0	0	0	0	0	0	0
Response	0	0	0	0	0	0	0	0	0
Double	0	0	0	0	0	0	0	0	0

Table 7: This table displays the estimated coefficients for each algorithm on iteration 1. Since the variance between iterations is small, these are reasonable for subsequent iterations.

Estimator	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9
Para. 1	0.3	0.400	0.000	0.100	0.000	0.200	0.000	0.000	0
Para. 2	0.3	0.400	0.000	0.100	0.000	0.200	0.000	-1.000	1
Outcome	0.3	-0.071	0.471	-0.194	0.294	-0.035	-0.471	-0.294	1
Response	0.3	0.400	0.000	0.100	0.000	0.200	0.000	0.000	0
Double	0.3	-0.071	0.471	-0.194	0.294	-0.035	-0.471	-0.294	1

Table 8: This table computes the F-statistic and P-value for an F-test comparing the `Full Model` with a nested model in `Reduced Model`. The computation of the F-statistic is found in the `get_f_statistic` function.

Full Model	Reduced Model	Fstat	P-Value
Parametric 1	Parametric 2	0.00	1
Parametric 1	Outcome Robust	1190.30	0
Parametric 1	Response Robust	0.00	1
Parametric 1	Double Robust	510.13	0
Parametric 2	Outcome Robust	1787.26	0
Parametric 2	Response Robust	0.00	1
Parametric 2	Double Robust	595.75	0
Outcome Robust	Double Robust	0.00	1
Response Robust	Double Robust	895.43	0

From the analysis in Table 8, we can see that there are significant differences between several of the models. Why is this different from the previous analysis in Table 4? The answer comes from how we compute the variance versus the sum of squares of error. The variance is computed from each estimator created in a Monte Carlo trial; whereas the sum of squares of error is summed from each individuals estimate for the mean of Y_2. For more details about how the F-test is computed see Appendix A and Appendix B.

Appendix A: Computing the F-Test

To compute an F test we can use Equation 1.

$$F = \frac{(SSE_{Reduced} - SSE_{Full})/(DFE_{Reduced} - DFE_{Full})}{SSE_{Full}/DFE_{Full}}. \tag{1}$$

Let's assess what this would mean to compare models `Parametric 1` and `Outcome Robust` from Table 9.

Table 9: This table identifies the different constraints for each model type.

Type	Constraints
Parametric 1	$\sum_{k,t} c_{kt} = 1$
Parametric 2	$\sum_{k,t:(k,t)\neq(4,4)} c_{kt} = 0, c_{44} = 1$
Outcome Robust	$c_{11}+c_{21}+c_{31}+c_{41} = 0, c_{22}+c_{42} = 0, c_{33}+c_{43} = 0,$ and $c_{44} = 1.$
Response Robust	$c_{11} = \pi_{00}, c_{21}+c_{22} = \pi_{10}, c_{31}+c_{33} =$ $\pi_{01},$ and $c_{41}+c_{42}+c_{43}+c_{44} = \pi_{11}$
Double Robust	$c_{11}+c_{21}+c_{31}+c_{41} = 0, c_{22}+c_{42} = 0, c_{33}+c_{43} = 0, c_{44} = 1, c_{11} =$ $\pi_{00}, c_{21}+c_{22} = \pi_{10}, c_{31}+c_{33} = \pi_{01},$ and $c_{41}+c_{42}+c_{43}+c_{44} = \pi_{11}$

Consider the case where we run estimate each model on a data set with $n = 1000$ observations, and define the following notation: let $\hat{\theta}^{(P)}$ and $\hat{\theta}^{(OR)}$ be the estimated values of $\theta = E[Y_2]$ for the `Parametric 1` model and the `Outcome Robust` model respectively. Define the estimated coefficients to be $\hat{c}_j^{(P)}$ and $\hat{c}_j^{(OR)}$ where $j = 1, \dots 9$ for the `Parametric 1` and `Outcome Robust` models respectively. The one can compute the SSE with the following:

$$SSE = n^{-1}\sum_{i=1}^{n}(\hat{y}_{2i} - \theta)^2 \text{ where } \hat{y}_{2i} = \sum_{j=1}^{9}\hat{c}_j\hat{\gamma}_j$$

and $\hat{\gamma}_0 := \hat{\gamma}_{00} = \frac{\delta_{00i}}{\pi_{00}}E[Y_2 \mid x_i]$, $\hat{\gamma}_1 := \hat{\gamma}_{11} = \frac{\delta_{10i}}{\pi_{10}}E[Y_2 \mid x_i, y_{1i}]$, etc. Note that this is different from the previous estimate of the variance which used the Monte Carlo variance (standard deviation) defined by

$$\frac{1}{n-1}\sum_{b=1}^{B}(\hat{\theta}_b - \bar{\theta}_B)^2$$

where B is the number of Monte Carlo estimates and $\bar{\theta}_B = \frac{1}{B}\sum_{b=1}^{B}\hat{\theta}_b$.

Likewise, one can compute the degrees of freedom by noticing that each model has a degrees of freedom equal to nine minus the number of constraints. This means that we can compute

the model degrees of freedom and error degrees of freedom for each model type, which we do in Table 10

Table 10: This table displays the degrees of freedom for each model

Model	Model Degrees of Freedom	Error Degrees of Freedom
Parametric 1	$9-1=8$	$n-1-8=n-9$
Parametric 2	$9-1=8$	$n-1-8=n-9$
Outcome Robust	$9-4=5$	$n-1-5=n-6$
Response Robust	$9-4=5$	$n-1-5=n-6$
Double Robust	$9-8=1$	$n-1-1=n-2$

So continuing our first example[1], if $SSE^{(P)} = 993$ and $SSE^{(OR)} = 3428$ with $n = 1000$ then the F statistic is

$$F = \frac{(3428-993)/(3)}{993/(1000-9)} = 810.$$

The critical value we want to compare this with is the 0.95 quantile of $F_{3,993}$ which is 2.61. So there *is* a significant difference between the fit of these two model. (The p-value is basically zero.) The code to do this automatically is in Appendix B.

[1]SSE values might differ in Table 8 slightly depending on the actual data values.

Appendix B: R Functions

```r
#' get_f_statistic is a function to compute the F-statistic between two of
#' the constrained models.
#'
#' @param c_tab - A table with values for the estimated c coefficients
#' @param reduced_mod - A string indicating which model is the reduced model
#' @param full_mod - A string indicating which model is the full model
#' @param df - A data set simulated from the same data generating process that
#'    the c_tab coefficients were estimated from.
#'
#' @note We have previously shown that the variance of the estimated c_tab
#' variables is close to or exactly zero. Hence, conducting a new simulation
#' should not effect the results of the model much.
get_f_statistic <-
  function(c_tab, reduced_mod, full_mod, df, theta = true_theta, cov_e1e2) {

  vars_lst <- list(reduced_mod, full_mod)

  # SSE reduced
  red_cvec <-
    c_tab |>
    filter(iter == 1) |>
    filter(.data[["estimator"]] == vars_lst[[1]]) |>
    select(starts_with("c_")) |>
    as.numeric()

  # SSE full
  full_cvec <-
    c_tab |>
    filter(iter == 1) |>
    filter(.data[["estimator"]] == vars_lst[[2]]) |>
    select(starts_with("c_")) |>
    as.numeric()

  # Constructing gam_vec
  Eg <- theta
  Egx <- theta + df$X
  Egxy1 <- theta + df$X + cov_e1e2 * (df$Y1 - df$X)
  Egxy2 <- df$Y2
  EEg2 <- theta^2 + 2
```

```r
  EEgx2 <- theta^2 + 1
  EEgxy12 <- theta^2 + 1 + cov_e1e2^2
  EEgxy22 <- theta^2 + 2
  EEgxy1Egxy2 <- theta^2 + 1 + cov_e1e2^2

  g_11 <- df$delta_00 / df$prob_00 * (Egx)
  g_21 <- df$delta_10 / df$prob_10 * (Egx)
  g_22 <- df$delta_10 / df$prob_10 * (Egxy1)
  g_31 <- df$delta_01 / df$prob_01 * (Egx)
  g_33 <- df$delta_01 / df$prob_01 * (Egxy2)
  g_41 <- df$delta_11 / df$prob_11 * (Egx)
  g_42 <- df$delta_11 / df$prob_11 * (Egxy1)
  g_43 <- df$delta_11 / df$prob_11 * (Egxy2)
  g_44 <- df$delta_11 / df$prob_11 * (df$Y2)

  gam_mat <- base::cbind(g_11, g_21, g_22, g_31, g_33, g_41, g_42, g_43, g_44)

  # SSE
  sse_red <- sum((as.numeric(gam_mat %*% t(t(red_cvec))) - theta)^2)
  sse_full <- sum((as.numeric(gam_mat %*% t(t(full_cvec))) - theta)^2)

  # Error Degrees of Freedom
  df_vec <- nrow(df) - 1 - c(9 - 1, 9 - 2, 9 - 4, 9 - 4, 9 - 8)
  names(df_vec) <- c("parac", "zeroc", "outc", "respc", "doubc")

  df_red <- df_vec[reduced_mod]
  df_full <- df_vec[full_mod]

  fstat <- ((sse_red - sse_full) / (df_red - df_full)) / (sse_full / df_full)

  pval <- 1 - pf(fstat, df_red - df_full, df_full)
  return(list(f = fstat, p = pval))

}
```