Appendix for Generalized Kernel Regularized Least Squares

Qing Chang and Max Goplerud

A Additional Theoretical Results

This section contains additional theoretical results. First, we provide a more detailed exposition of the Gaussian model. It unifies our presentation in the main text with more classical presentations of hierarchical/multilevel models (e.g., Hazlett and Wainstein 2022). It also allows us to derive our justification for robust/clustered standard errors as well as providing new standard errors for (traditional) KRLS. Second, we discuss the alternative estimation methods available in `mgcv` (`gam` vs `bam`). Finally, we explain how our software estimates average marginal effects.

A.1 Alternative Standard Errors

We consider the following Gaussian outcome model for our discussion on standard errors following Hazlett and Wainstein (2022). Extensions to non-Gaussian outcomes can be made using standard arguments (see Wood 2006). We do not assume any special structure on $\boldsymbol{\Omega}$ but are more general than the main text insofar as $\boldsymbol{\Sigma}$ need not be diagonal. We assume that $\boldsymbol{\Omega}$ and $\boldsymbol{\Sigma}$ are known and full rank. A rank-deficient $\boldsymbol{\Omega}$ can be addressed via an eigen-decomposition and adjusting the fixed effect design matrix $\boldsymbol{X}$. $\boldsymbol{\Sigma} = \sigma^2 \boldsymbol{I}$ and $\boldsymbol{\Omega} = 1/\sigma^2 \boldsymbol{S}_{\boldsymbol{\lambda}}$ recovers the model in the main text (Equation 4b).

$$\boldsymbol{y} \sim N\left(\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{\alpha}, \boldsymbol{\Sigma}\right) \tag{A.1}$$
$$p(\boldsymbol{\beta}) \propto 1; \quad \boldsymbol{\alpha} \sim N(\boldsymbol{0}, \boldsymbol{\Omega}^{-1}) \tag{A.2}$$

The penalized maximum likelihood estimator is shown below, mirroring Equation 5.

$$\begin{bmatrix} \hat{\boldsymbol{\beta}} \\ \hat{\boldsymbol{\alpha}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{X}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{X} & \boldsymbol{X}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{Z} \\ \boldsymbol{Z}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{X} & \boldsymbol{Z}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{Z} + \boldsymbol{\Omega} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{X}^T \\ \boldsymbol{Z}^T \end{bmatrix} \boldsymbol{\Sigma}^{-1}\boldsymbol{y} \tag{A.3}$$

Despite our different presentation, it can be easily verified (see also Wood 2017, p.80-81) that $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\alpha}}$ are identical to the standard multilevel estimators (e.g., Hazlett and Wainstein 2022, p. 51). In terms of appropriate estimators for the variance of $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\alpha}}$, Wood (2006) suggests two options; a "Bayesian" estimator that is the inverse of the Hessian of the log-posterior and a frequentist estimator. We consider each in detail. First, the Bayesian estimator $\mathcal{V}_B$ is shown below. This is justified based on the posterior distribution of $\{\boldsymbol{\beta}, \boldsymbol{\alpha}\}$ given $\boldsymbol{y}$ if $\boldsymbol{\Sigma}$ and $\boldsymbol{\Omega}$ are known (Wood 2006).

$$\mathcal{V}_B = \left[\begin{array}{cc} \boldsymbol{X}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{X} & \boldsymbol{X}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{Z} \\ \boldsymbol{Z}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{X} & \boldsymbol{Z}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{Z} + \boldsymbol{\Omega} \end{array} \right]^{-1}; \quad \mathcal{V}_B^{\beta} = \left(\boldsymbol{X}^T \left[\boldsymbol{\Sigma} + \boldsymbol{Z}\boldsymbol{\Omega}^{-1}\boldsymbol{Z}^T \right]^{-1} \boldsymbol{X} \right)^{-1}$$

Note that the upper left-block, corresponding to the estimated variance of $\hat{\boldsymbol{\beta}}$, denoted as $\mathcal{V}_B^{\beta}$ is identical to the "standard" multilevel model estimator for $\hat{\boldsymbol{\beta}}$ (see Wood 2017, p. 80 and Hazlett and Wainstein 2022, p. 52). Second, Wood (2006) suggests a "frequentist" estimator where, for some choice of variance of $\boldsymbol{y}$ denoted as $\mathrm{Var}(\boldsymbol{y})$, the estimator $\mathcal{V}_F$ is shown below.

$$\mathcal{V}_F = \mathcal{V}_B \left[\begin{array}{c} \boldsymbol{X}^T \\ \boldsymbol{Z}^T \end{array} \right] \boldsymbol{\Sigma}^{-1}\mathrm{Var}(\boldsymbol{y})\boldsymbol{\Sigma}^{-1}[\boldsymbol{X}, \boldsymbol{Z}]\mathcal{V}_B$$

There are different possible choices for $\mathrm{Var}(\boldsymbol{y})$. Following Wood (2006), we can assume the model is correct, condition on $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ and thus use $\mathrm{Var}(\boldsymbol{y}) = \boldsymbol{\Sigma}$. We denote this estimator as $\mathcal{V}_F^{Wood}$. When comparing $\mathcal{V}_B$ and $\mathcal{V}_F^{Wood}$, Wood (2006) suggests that $\mathcal{V}_B$ is preferable insofar as it reflects a coherent Bayesian model and the $\mathcal{V}_F$ estimator is likely to have considerably undercoverage due to the fact that neither $E[\hat{\boldsymbol{\beta}}] \neq \boldsymbol{\beta}$ nor $E[\hat{\boldsymbol{\alpha}}] \neq \boldsymbol{\alpha}$. A variety of work has corroborated this suggestion empirically and theoretically (see, amongst others, Wood 2017, Ch. 6.10 or Marra and Wood 2012). $\mathcal{V}_B$ has been shown to have usually rather good frequentist coverage despite coming from a regularized model. In general, this is our preferred estimator as it is naturally derived from our Bayesian and hierarchical interpretation of KRLS. However, $\mathcal{V}_F$ is useful insofar as it lends itself to "robust" estimators by choosing a specific choice of $\mathrm{Var}(\boldsymbol{y})$ that does not hue to the assumptions of the model. For example, if one suspected heteroskedastic errors, $\mathrm{Var}(\boldsymbol{y})$ could be swapped with the usual "meat" of the squared residuals, possibly scaled by a finite sample correction (Cameron and Miller 2015).

This idea is not novel to our paper, although we think it is perhaps underappreciated in the literature: Hazlett and Wainstein (2022) consider it in the case of clustered standard errors and a hierarchical model with a single random effect. Our formula, with the same corresponding "meat", recovers an identical variance matrix on $\hat{\boldsymbol{\beta}}$ to their proposal in Equation 8 (p. 51) with some re-arrangement. However, a benefit of this formulation is that the joint uncertainty on $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\alpha}}$ is quantified so any quantity of interest that includes *both* terms (i.e., most marginal effects and predicted values) can be fully incorporated.

Our justification does not assume any particular structure on $\boldsymbol{\Omega}$ or $\boldsymbol{\Sigma}$ so applies to generic generalized additive models as other types of robust standard errors (e.g., multiway clustering, Conley standard errors for spatial dependence, etc.; Cameron and Miller 2015). Following the arguments in Wood (2006) that generalize $\mathcal{V}_F$ and $\mathcal{V}_B$ to non-Gaussian outcomes, one could also apply our logic to non-Gaussian outcomes. Exploring these alternative standard errors in more detail is an interesting area for future research.

A.1.1 Implications for Traditional KRLS

We briefly consider the implications for traditional KRLS. Beyond justifying (cluster) robust standard errors, there is a more subtle point about the appropriate "regular" standard errors for KRLS. Hainmueller and Hazlett (2014, p. 153) propose frequentist standard errors, i.e. $\mathcal{V}_F$ assuming $\text{Var}(\boldsymbol{y}) = \sigma^2\boldsymbol{I}$. Noting that in their model, $\boldsymbol{\beta}$ does not exist (and $J = 1$), $\boldsymbol{Z} = \boldsymbol{K}$, $\boldsymbol{\Omega} = \frac{\lambda}{\sigma^2}\boldsymbol{K}$, and $\boldsymbol{\Sigma} = \sigma^2\boldsymbol{I}$, the possible variance estimators are shown below where $\mathcal{V}_{HH}$ is the suggestion in the original paper and coincides with $\mathcal{V}_F^{Wood}$.

$$\begin{gathered}\mathcal{V}_B = \sigma^2\left[\boldsymbol{K}\boldsymbol{K} + \lambda\boldsymbol{K}\right]^{-1} \\ \mathcal{V}_{HH} = \mathcal{V}_F^{Wood} = \sigma^2\left(\boldsymbol{K} + \lambda\boldsymbol{I}\right)^{-1}\left(\boldsymbol{K} + \lambda\boldsymbol{I}\right)^{-1}\end{gathered} \tag{A.4}$$

While exploring the coverage properties of these estimators in detail is outside of the scope of this paper, we note that the suggested variance-covariance matrix from Hainmueller and Hazlett (2014) is thus *not* the standard recommendation for generalized additive models ($\mathcal{V}_B$) and thus may have worse coverage properties than the Bayesian alternative. We examine this in a simple stylized example. We use a bivariate smoothing example from `mgcv`, shown below. We draw 250 observations indexed with $i \in \{1, \cdots, N\}$.

$$y_i|x_i, z_i \sim N\left(f(x_i, z_i), 1\right); \quad x_i \sim \text{Unif}(0,1); \quad z_i \sim \text{Unif}(0,1) \tag{A.5a}$$

$$f(x,z) = \pi^{\sigma_x^2}\sigma_z^2 \cdot \left(1.2 \cdot \exp\left[-\frac{(x-0.2)^2}{\sigma_x^2} - \frac{(z-0.3)^2}{\sigma_z^2}\right] + 0.8 \cdot \exp\left[-\frac{(x-0.7)^2}{\sigma_x^2} - \frac{(z-0.8)^2}{\sigma_z^2}\right]\right) \tag{A.5b}$$

In this case, we compare the coverage over the response surface itself, i.e. over the Cartesian product of a grid of 40 evenly spaced values of x (from 0 to 1) and 40 evenly spaced values of z (from 0 to 1). We repeat the simulation 200 times, i.e. generate 200 sets of $\boldsymbol{y}$ and report the average coverage on the predicted values across all simulations.

We consider `KRLS` with its default standard errors ($\mathcal{V}_{HH}$), `KRLS` with $\mathcal{V}_B$, `gKRLS` with default standard errors ($\mathcal{V}_B$), `gKRLS` with default standard errors and the two variables x_i and z_i also included linearly *outside* the kernel as fixed effects. This latter model is more comparable to the default splines in `mgcv` where the null space of the penalty contains an additive linear model. In this specification, as $\lambda \to \infty$, the model approaches a linear additive one in x and z versus an intercept only model for standard (g)KRLS. We finally compare against the tensor smoothing spline suggested by `gam`. Table A.1 shows the results. It reports the size of the average standard error on a predicted value, the coverage averaged across all points and simulations, and the mean absolute error (MAE) in prediction.

It illustrates the expected problem with `KRLS`'s default standard errors, $\mathcal{V}_{HH}$. While the point estimates are of high quality (better than `gam`), the coverage is very poor. By contrast, simply replacing the variance estimator with $\mathcal{V}_B$ immediately improves coverage dramatically—albeit remaining below nominal. In terms of `gKRLS`, we see that it performs equivalently to KRLS in performance—beating `mgcv`—and has similar coverage to the KRLS with $\mathcal{V}_B$. The model that includes x_i and z_i linearly outside of the kernel (`gKRLS`

Table A.1: Performance of Methods for Bivariate Smooth

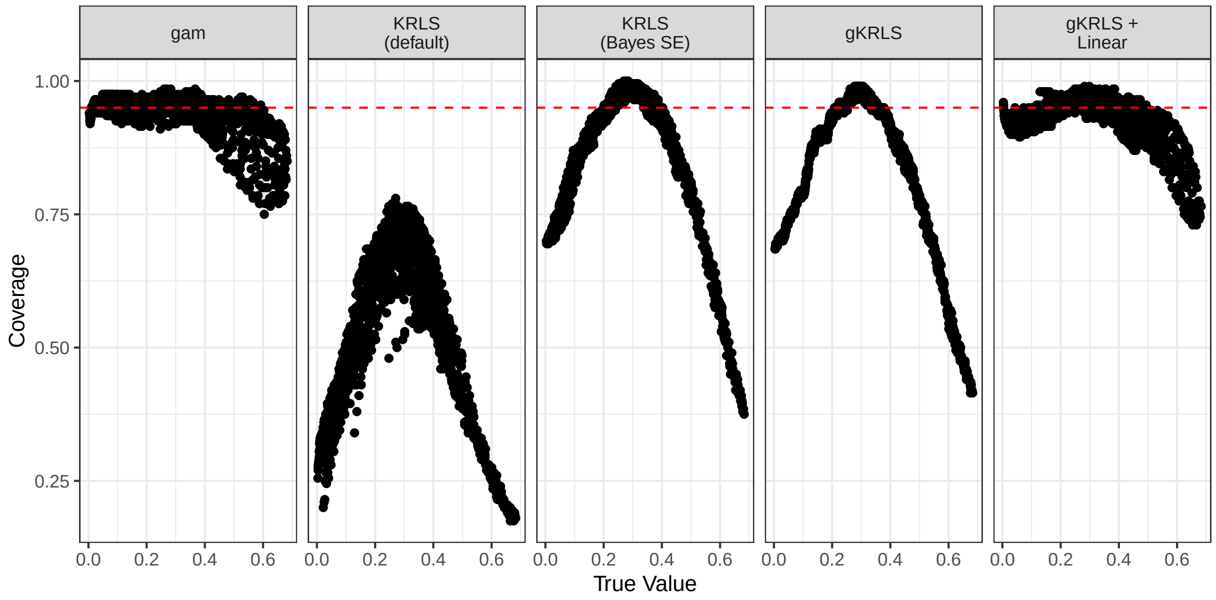
Method	Coverage	MAE	Average SE
`KRLS` (default)	0.526	0.170	0.086
`KRLS` (Bayes SE)	0.859	0.170	0.167
`gKRLS`	0.850	0.175	0.182
`gKRLS` + Linear	0.931	0.218	0.261
`gam`	0.939	0.234	0.277

Note: The coverage proportion of a 95% confidence interval across the entire grid of test values is shown. The mean absolute error (MAE) in prediction and the average standard error for each prediction is shown. These quantities are all averaged across 200 simulations.

+ Linear) has slightly worse performance but close to nominal coverage and improves in terms of MAE upon the tensor product smooth (`gam`).

Figure A.1 plots the average coverage for each observation by the true value; it shows that for more extreme fitted values, the coverage of KRLS and `gKRLS` is increasingly poor. Using the Bayesian standard errors improves KRLS considerably although it is still usually below nominal.

Figure A.1: Coverage by True Value



Note: The coverage proportion for each observation given the 95% confidence interval on the predicted value, averaged across 200 simulations, is shown. The true fitted value is on the horizontal axis.

Finally, we use a simple example that illustrates the importance of clustered standard errors. Sticking to the above simulation, we assume that there is a group g that is drawn randomly from one of forty clusters ($g \in \{1, \cdots, 40\}$). For two observations in the same cluster, our generative model assumes they are correlated with $\rho = 0.6$ and are otherwise uncorrelated. We re-estimate the same models from before, but also consider cluster-robust errors for `gam` and `gKRLS`. We see from Table A.2 that cluster-robust standard errors improve the coverage for the `gKRLS` and `mgcv` methods. We have written additional software to allow `mgcv` methods (`gam` and `bam`) to be used with the `sandwich` package. Without our software, there is a bug (at the time of writing [April 2023] affecting at least versions 3.0.2 and older) that results incorrect robust/clustered standard errors for Gaussian outcomes and certain models with non-canonical links.

Table A.2: Cluster-Robust Errors for Smoothing Methods

	Regular SE			Clustered SE		
Method	Coverage	MAE	Average SE	Coverage	MAE	Average SE
`gam`	0.830	0.159	0.146	0.910	0.159	0.177
`gKRLS`	0.775	0.148	0.126	0.879	0.148	0.153
`gKRLS` + Linear	0.844	0.156	0.151	0.909	0.156	0.172
`KRLS` (default)	0.522	0.150	0.072			

Note: The same quantities from Table A.2 are shown. The first three columns consider regular standard errors; the final three consider the cluster-robust version.

A.2 Alternative Estimation Methods

The main text notes that `mgcv` provides two methods for estimation: `gam` and `bam`. `gam` is designed to be a highly stable numerical procedure. However, it is sometimes slow especially when the number of observations is very large (e.g., the simulations in Appendix B) or when the number of parameters is very large (e.g., the replication of Newman 2016). The latter may often occur when a random intercept with many levels is included. In these cases, some alternative estimation technique is needed. Wood, Goude and Shaw (2015) develop `bam` to address these limitations. The two main innovations are as follows (see Wood 2017, p. 290 for a concise summary): First, `bam` slightly alters the estimation procedure for $\boldsymbol{\lambda}$. In the initial exposition in the main paper, for a proposed $\boldsymbol{\lambda}$, $\hat{\boldsymbol{\beta}}_{\boldsymbol{\lambda}}$ and $\hat{\boldsymbol{\alpha}}_{\boldsymbol{\lambda}}$ are estimated to convergence—e.g. using penalized iteratively re-weighted least squares (PIRLS) with multiple iterations for a non-Gaussian outcome. This can be expensive so `bam` optimizes $\boldsymbol{\lambda}$ after each step in PIRLS estimation. This is known as "performance orientated iteration" and can be less numerically stable than the main algorithm (Wood, Goude and Shaw, 2015).

Second, and perhaps more interestingly, `bam` never forms the entire design matrix. Rather, it splits the data into chunks of size p (10,000 by default) and then builds the matrix iteratively as needed. This allows it to exploit multiple cores—although we do not use this in our paper. This can lower the memory footprint of the algorithm considerably. One point of caution, however, is that the basis for the penalized terms (i.e., the hierarchical terms) are formed only on a random sample of chunk size p taken at the start of the algorithm. For our purposes, this means that `bam` freezes the size of the sketching matrix at $\delta(p)^{1/3}$. If $p = 10,000$ (default), this is around 21δ if $N > 10,000$. This chunk size can be modified using arguments to `bam` but may slow down the algorithm somewhat. Thus, while `bam` can be helpful for large-scale problems, one should be careful to check any potential impacts of chunk size p (perhaps by increasing δ).

We examine `bam` systematically throughout the appendices. The main places where the chunk size issue could cause different results would be for (i) the simulations in Section 4 where $N > 10,000$ and (ii) the analysis of Gulzar, Haas and Pasquale (2020) where $N = 30,000$ for the full-sample analysis (although note that $N < 10,000$ for the algorithms relying on sample-splitting). We find little evidence of difference between `bam` and `gam`.

A.3 Calculating Average Marginal Effects

Our accompanying software provides the ability to calculate average expected outcomes (e.g., average predicted probabilities) as well as "marginal effects" (e.g., first differences). For continuous predictors, we also include the ability to calculate the average marginal effect, i.e., the average of the partial derivative of the prediction with respect to a single covariate (Hainmueller and Hazlett, 2014); Appendix D provides a specific example.

Following existing software, we do this using numerical differentiation; Leeper (2016) provides a detailed discussion. This is especially important for complex models where a single covariate could appear multiple times and using an analytical approach is difficult to implement in a flexible fashion. We use following formula following Leeper (2016), shown for a two argument function for simplicity:

$$\frac{\partial f(x,y)}{\partial x} = \lim_{h \to 0} \frac{f(x+h,y) - f(x-h,y)}{2h}$$

In practice, some small h is used to approximate the derivative. The default setting for h is $h = max(|x|, 1)\sqrt{\epsilon}$ where $max(|x|, 1)$ ranges over data distribution that one is marginalizing over (following Leeper 2016) and ϵ is machine precision. h can be modified by the user if desired. Our package includes a function `legacy_marginal_effect` that can calculate the analytical average marginal effect in the simple case of a single kernel and some limited choices for outcome (e.g., Gaussian). Standard errors and confidence intervals on these quantities (or their averages) are calculated using the delta method.

B Additional Simulations: Scalability of `gKRLS`

This appendix provides additional simulations to complement Section 4. Appendix B.1 defines how we measure model performance and shows the error on estimating the average marginal effect. Appendix B.2 shows the performance of alternative sketching methods as the dataset grows to one million observations. Appendix B.3 breaks down estimation time by different parts of `gKRLS`.

B.1 Assessment of Model Performance

We consider two ways of assessing the performance of `gKRLS` following Hainmueller and Hazlett (2014). First, the main text considers the out-of-sample predictive accuracy. We do this by generating a dataset of identical size to the estimation data using the same data generating process. We then calculate the prediction for each observation in the test data and summarize the performance using the root mean squared error (RMSE).

Alternatively, we compare the estimated average marginal effect to the true value. The true value is computed by calculating the marginal effect for each observation in the training data (i.e. the analytical partial derivative of the data generating process with respect to the covariate) and then taking their average. Computing this quantity with respect to an out-of-sample dataset, i.e. the average effect on an identically sized population not used to estimate the original model, returns nearly identical results.

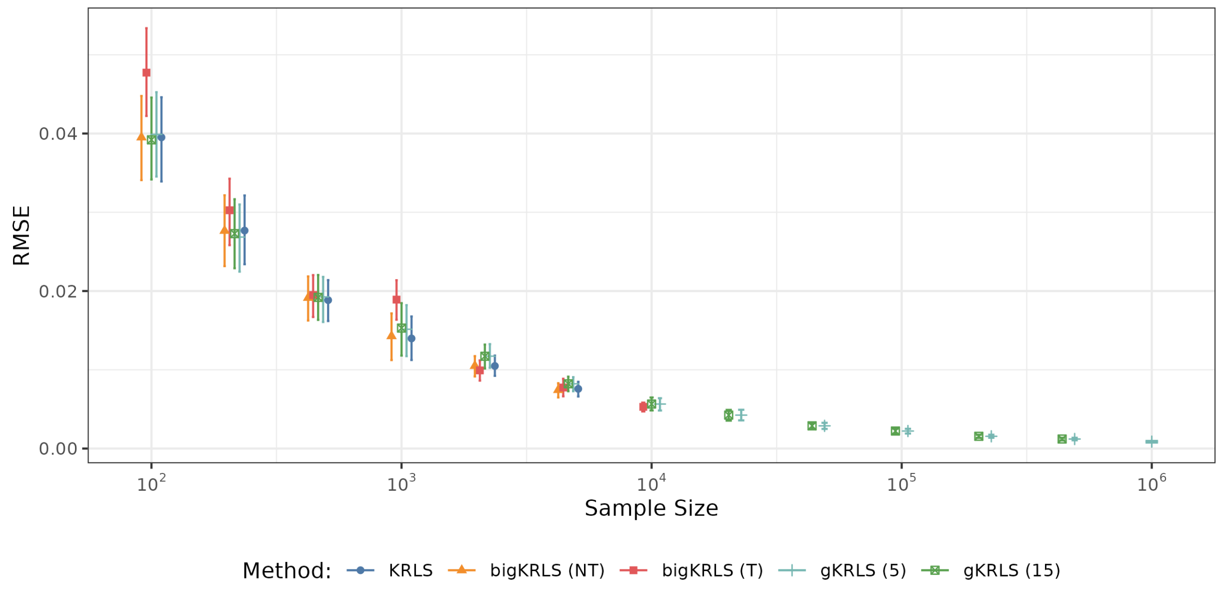
Figure A.2 shows the results for the five models in the main text on the root mean squared error (RMSE; averaged across fifty simulations and both variables) of the estimated average marginal effect. It shows a worse performance of bigKRLS with truncation ("bigKRLS (T)") for some sample sizes—although less dramatically than in the main text.

B.2 Alternative Sketch Methods

Yang, Pilanci and Wainwright (2017) raise a number of issues with random sketching—especially in the case of complex data where sub-sampling is likely to yield a bad representation of the original data; Lee and Ng (2020) provides a recent overview of different techniques that may address this problem. To explore this for our initial simulations, we also consider Yang, Pilanci and Wainwright (2017)'s Gaussian sketching; Appendix C.5 considers it for the second set of simulations.

Formally, for some sketching dimension M, Gaussian sketching generates a matrix that is $M \times N$ where each element is drawn from a $N(0, 1/\sqrt{M})$ following Yang, Pilanci and Wainwright (2017). In this case, the sketched kernel is some randomized combination of *all* observations—rather than simply being constructed on a subset of observations. Despite the beneficial nature of this method, it incurs a considerably higher computational burden in building the sketched kernel as the kernel matrix $\boldsymbol{K}$ $(N \times N)$ must be multiplied by a dense $N \times M$ matrix. If N is very large, this can be quite expensive to compute even once. One possible solution is the parallelization of $\boldsymbol{K}\boldsymbol{S}^T$.

Figure A.2: Error on Estimating Average Marginal Effect



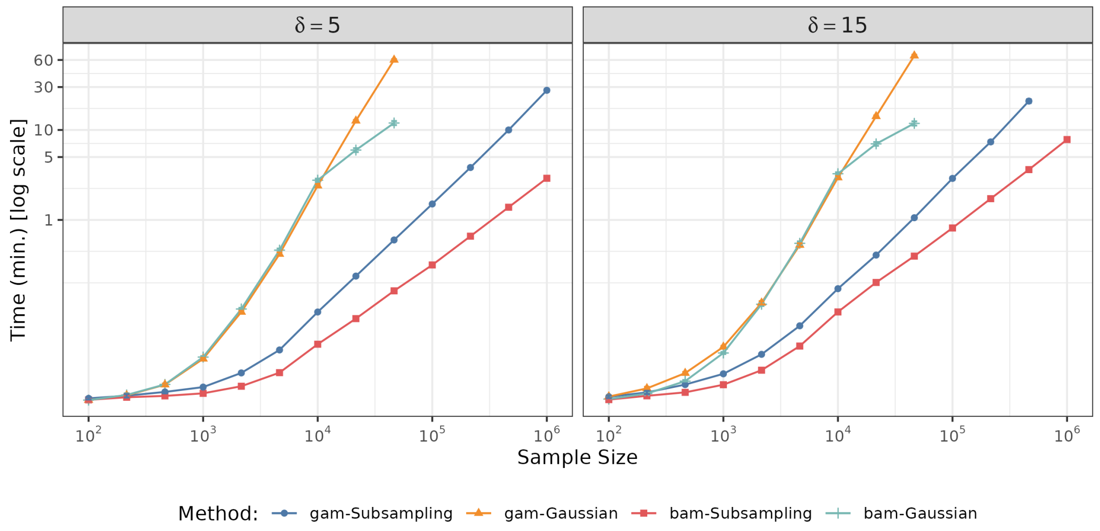
Note: The figure reports the RMSE of the estimated average marginal effect (AME) averaged across both covariates and fifty simulations. 95% confidence intervals are reported using a percentile bootstrap using 1,000 bootstrap samples.

Figure A.3 compares computational cost of Gaussian sketching. To examine alternative estimation methods designed for huge datasets (e.g., `gam` vs. `bam`; see Appendix A.2), we also compare results for `gam` and `bam` and types of sketching (sub-sampling sketching or Gaussian), denoted by "method-sketching type". We see that Gaussian sketching is considerably slower and rather expensive after around 45,000 observations.

As noted in the main text, when using `gam` (sub-sampling sketching; multiplier $\delta = 5$), estimation takes around 30-40 minutes for 1,000,000 observations. This figure illustrates that `bam` can help considerably. Estimation time is around three minutes, although as discussed in Appendix A.2 this is partially due to a freezing of the size of the sketching dimension.

We can also use the log-log plot to provide a rough estimate of the computational complexity of the various algorithms and sketching procedures; since the plot looks highly linear (above a certain sample size), we can use the slope p of that log-log line to estimate the complexity as roughly N^p for sufficiently large N. As expected, KRLS and bigKRLS is around N^3, with an estimated slope of around 3 when the sample size is over 100 for KRLS and 1,000 for bigKRLS. For the sub-sampling sketched methods, if we focus on a sample size above 40,000—as the relationship appears clearly linear—the estimated slope for the methods is around 1.3, suggesting a cost that increases much more slowly than traditional methods. When using `bam` (discussed in Appendix A.2), the slope is around 0.95, presumably because the size of the sketching dimension does not grow after

Figure A.3: Estimation Time of Different Methods



Note: This figure shows the average computational time in minutes averaged across fifty iterations. The left panel shows a multiplier of $\delta = 5$ and the right shows a multiplier of $\delta = 15$. The main text defines the abbreviations. 95% confidence intervals are shown.

$N > 10,000$. With Gaussian sketching, the estimated slope (when $N > 2500$) for `gam` is around 2.10; lower than the unsketched methods but considerably larger than the sub-sampling methods.

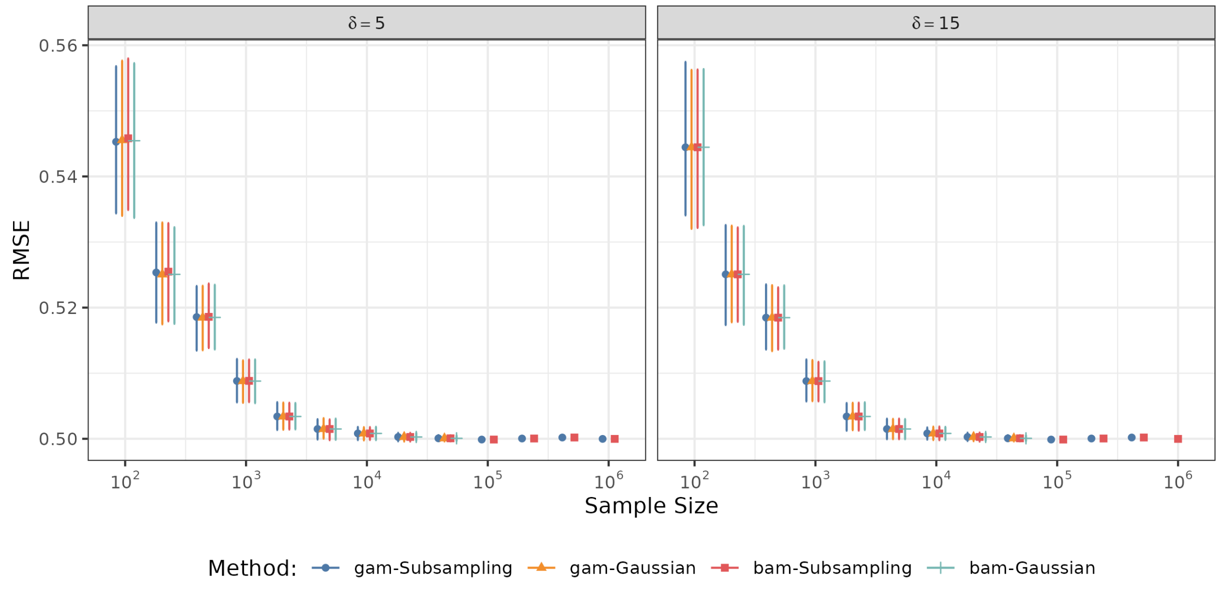
Figure A.4 shows that all sketching methods are equally accurate.

B.3 Estimation Time Disaggregation

When estimating `gKRLS` and using it for inference, there are three steps that the applied user may perform; it is useful to know which takes more time as sample size increases. First, the model must create the (sketched) kernel and estimate the parameters ("Estimation"); second, one might wish to perform prediction on a new dataset; we use one of the same size as the estimation data ("Prediction"). Finally, one might wish to calculate marginal effects ("Marg. Effects"); this requires repeated predictions on counterfactual versions of the estimation data. The total time is reported in the main text.

Figure A.5 reports the average time for each stage across sample size and for the four sketching methods considered. It uses a log-scale for readability. We see that, for sub-sampling sketching, estimation time becomes highly expensive as N (and the parameter dimension) grows very large and dominates the overall cost. For Gaussian sketching, the cost of calculating marginal effects becomes increasingly expensive as the cost of actually evaluating the kernel begins to dominate the computational cost. This also likely explains the explosion of computational time for estimation at an earlier stage than the sub-sampling

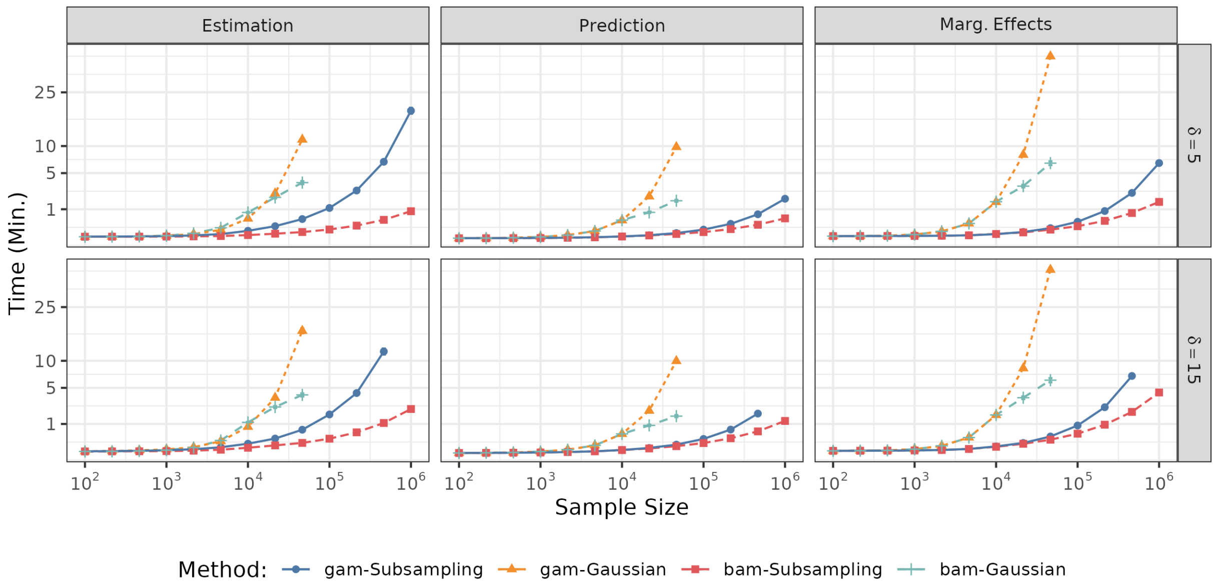
Figure A.4: Performance of Alternative Sketching and Estimation Methods



Note: This figure shows the root mean squared error (RMSE) of predicting the outcome on a dataset of the same size to the estimation data, averaged across the fifty simulations. The main text defines the abbreviation for each method. 95% confidence intervals are shown; they are calculated using a percentile bootstrap using 1,000 bootstrap samples.

method. As expected, `bam` provides considerable gains in speed—mostly in the estimation stage.

Figure A.5: Estimation Time of `gKRLS` by Step



Note: This figure shows the average computational time in minutes averaged across fifty iterations by each stage of estimation (“Estimation”, “Prediction”, “Marg. Effects”). The main text defines the abbreviation for each method. 95% confidence intervals are shown.

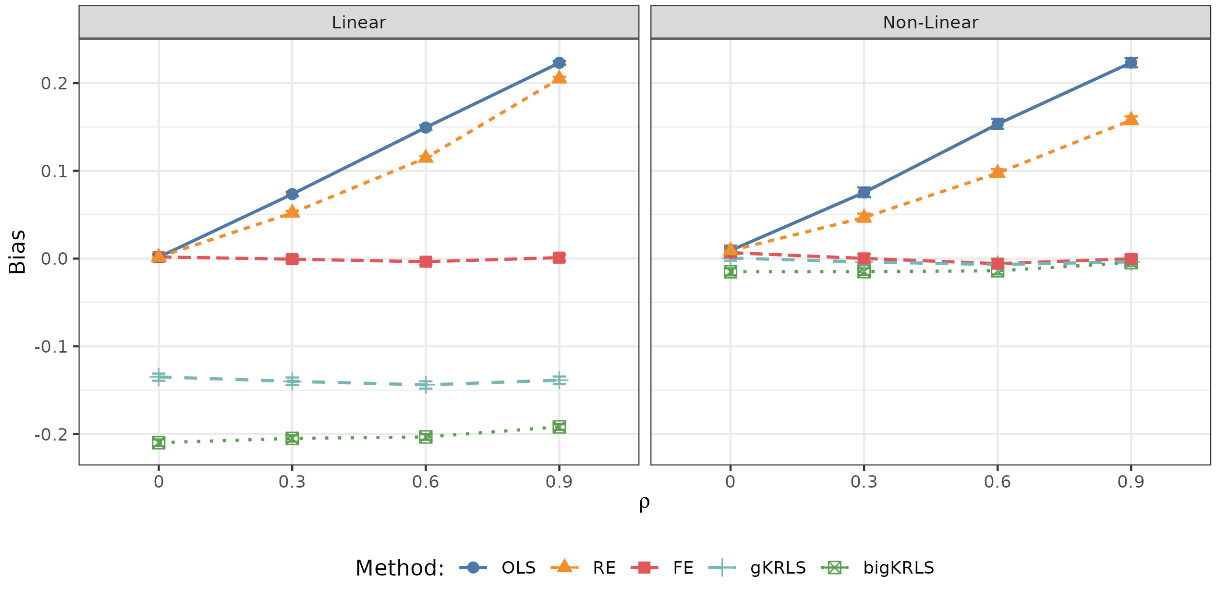
C Additional Simulations: `gKRLS` with Fixed Effects

This appendix contains additional information on the simulations in Section 4.1 where we include fixed effects in the data generating process. Appendix C.1 reports different performance metrics such as bias and out-of-sample predictive RMSE (see Appendix B.1 for definitions). Appendix C.2 reports the performance on the other covariate ($x_{i,2}$). Appendix C.3 varies the number of observations. Appendix C.4 considers different variants of `gKRLS` to understand what drives the improved performance versus bigKRLS. Appendix C.5 provides information on the impact of sketching in this more complicated scenario.

C.1 Alternative Performance Metrics

In addition to the RMSE, Figure A.6 shows the estimated bias of the methods in Figure 3 for the AME of the main covariate of interest ($x_{i,1}$). It shows, as expected, that the fixed effects estimator is unbiased and that the bias of the OLS and RE methods increase considerably as ρ increases. In the linear model, the kernel methods are biased downwards although the bias for `gKRLS` is considerably smaller. In the non-linear case, both have a small bias, although bigKRLS's is slightly larger at small ρ.

Figure A.6: Bias of AME



Note: The figure reports the bias of the estimated average marginal effect (AME) on the first covariate ($x_{i,1}$), averaged across all simulations, as ρ varies. The left panel shows the linear data generating process and the right shows the non-linear data generating process. 95% confidence intervals are shown; they are calculated using a percentile bootstrap using 1,000 bootstrap samples.

We next compare the out-of-sample predictive accuracy of the methods—estimated using the procedure in Appendix B.1. Figure A.7 reports the RMSE averaged across all 1,000 simulations. For the linear data generating process, the kernel methods perform worse than fixed effects or random effects, although the differences are more modest. In the non-linear case, we see the kernel methods perform the best although `gKRLS` out-performs bigKRLS by a considerable margin.

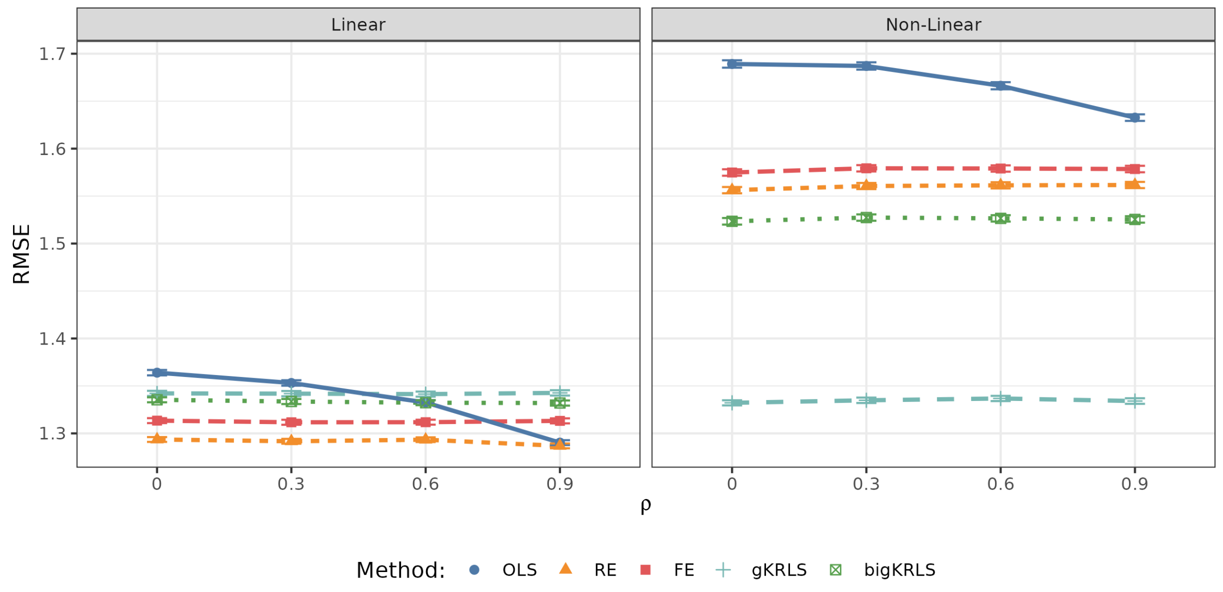
C.2 Performance on Other Covariate ($x_{i,2}$)

We replicate Figure 3 from the main text on the covariate $x_{i,2}$, i.e. one that is not correlated with the fixed effect. Figure A.8 shows that, in the linear case, all methods perform rather similarly at estimating this average marginal effect—including methods such as random effects or OLS. In the spline case, `gKRLS` clearly out-performs all other methods.

C.3 Varying Number of Observations Per Group

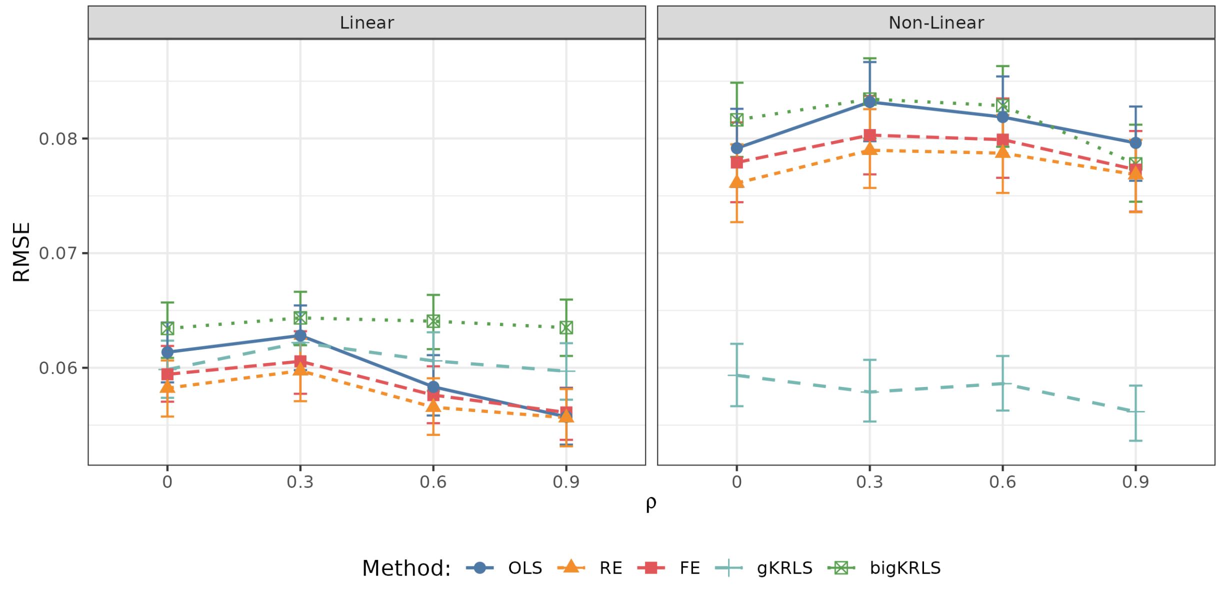
In the main text, we consider 50 groups ($J = 50$) and set the group size (i.e. number of observations per group) at 10. We vary that here and consider $T \in \{5, 10, 25, 50\}$.

Figure A.7: Out-of-Sample Predictive Accuracy



Note: This figure shows the root mean squared error (RMSE) of predicting the outcome on a dataset of the same size to the estimation data, averaged across 1000 simulations, as ρ varies. The left panel shows the linear data generating process and the right shows the non-linear data generating process. 95% confidence intervals are shown.

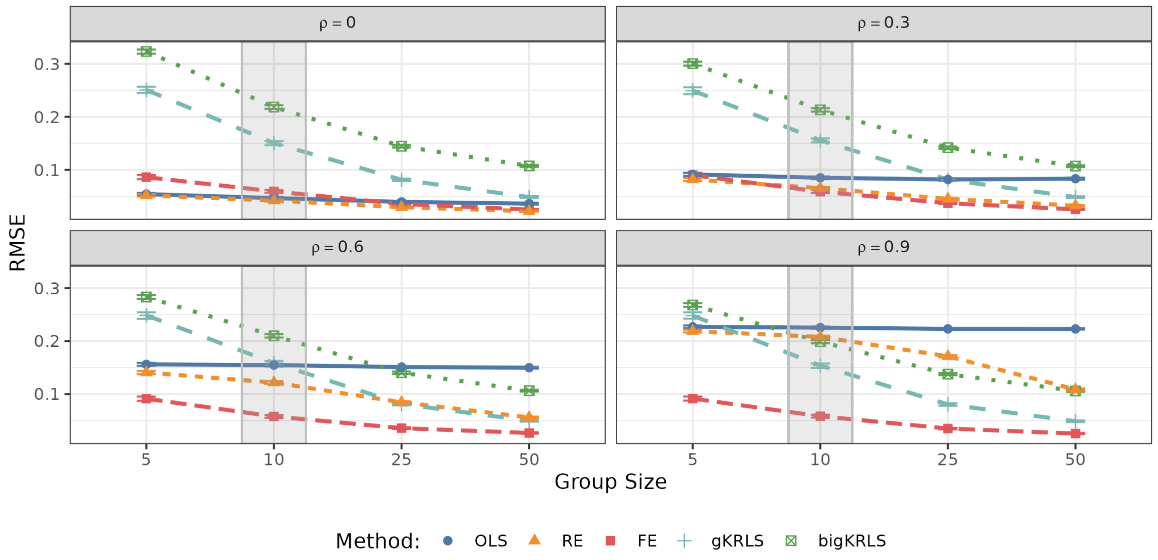
Figure A.8: RMSE on the Average Marginal Effect of $x_{i,2}$



Note: The figure reports the bias of the estimated average marginal effect (AME) on the second covariate ($x_{i,2}$), averaged across 1000 simulations, as ρ varies. The left panel shows the linear data generating process and the right shows the non-linear data generating process. 95% confidence intervals are shown; they are calculated using a percentile bootstrap using 1,000 bootstrap samples.

Figures A.9 and A.10 show the results as group size varies where the gray box indicates the results in the main text. Each panel displays a different value of ρ.

Figure A.9: Estimating AME for Varying Sample Size (Linear)



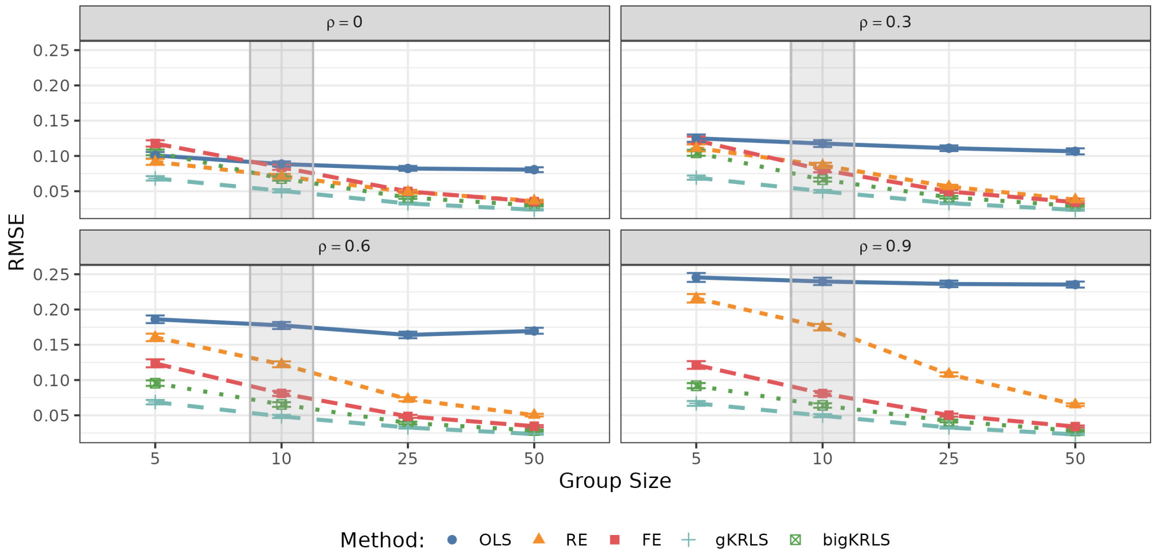
Note: This figure reports the RMSE of the estimated AME on the first covariate ($x_{i,1}$) at varying sample sizes for the linear data generating process. The shaded box indicates the values reported in the main analyses, i.e. $T = 10$. 95% confidence intervals are shown; they are calculated using a percentile bootstrap using 1,000 bootstrap samples.

Looking first at the linear case (Figure A.9), we see that as the cluster size increases, random effects and both kernel methods improve considerably. For larger cluster sample sizes, `gKRLS` is close to the RMSE for the fixed effect method although bigKRLS remains noticeably worse. This provides further evidence for the limitations of adding the fixed effects directly into the kernel. In the non-linear case, the story is similar—random effects improves with increasing sample size towards fixed effects—although `gKRLS` and bigKRLS are much closer in performance (as in the main text).

C.4 Understanding Why `gKRLS` Does Better

This section explores in more depth *why* `gKRLS` provides an improvement on bigKRLS. We first consider the most similar model to bigKRLS fit using `mgcv` (i.e. no sketching, standardized covariates, and fixed effects inside the kernel). We then vary each of these dimensions separately to the default settings in `gKRLS` and see which affects performance. This allows us to disaggregate what seems to improve performance in a more controlled setting. We consider the following models.

Figure A.10: Estimating AME for Varying Sample Size (Non-Linear)



Note: This figure reports the RMSE of the estimated AME on the first covariate $(x_{i,1})$ at varying sample sizes for the non-linear data generating process. The shaded box indicates the values reported in the main analyses, i.e. $T = 10$. 95% confidence intervals are shown; they are calculated using a percentile bootstrap using 1,000 bootstrap samples.

1. “Baseline”: gKRLS estimated with no sketching, fixed effects inside the kernel, standardized covariates (mean zero; variance one), GCV for penalty parameter selection.
2. “Mahal.”: “Baseline” but use Mahalanobis distance between covariates in the kernel.
3. “FE”: “Baseline” but include the fixed effects *outside* the kernel.
4. “Sketch(5)”: “Baseline” but use sketching with the default multiplier of five.
5. “Sketch(5)+REML”: Identical to “Sketch(5)“ but use REML instead of GCV to select λ.
6. “Naive”: This method includes the fixed effects in the kernel, but uses sketching, Mahalanobis distance, and GCV for penalty selection. It is “naive” as it simply swaps bigKRLS for gKRLS under the default settings of the package and mgcv.
7. “gKRLS (GCV)”: gKRLS estimated with sub-sampling sketching (multiplier of 5), fixed effects **outside** the kernel, Mahalanobis standardization, GCV for penalty parameter selection.
8. “gKRLS + Lin.”: This method includes the fixed effects and both covariates both inside and outside of the kernel. It ensures that as the amount of regularization increases, this shrinks towards a linear additive model.
9. “gKRLS”: The method shown in the main text. That is, “gKRLS (GCV)” but using REML for penalty parameter selection.

Figure A.11 reports the results where the estimated RMSE on the AME is divided by the RMSE from bigKRLS without truncation. Values above “1” indicate worse performance; values below 1 indicate better performance.

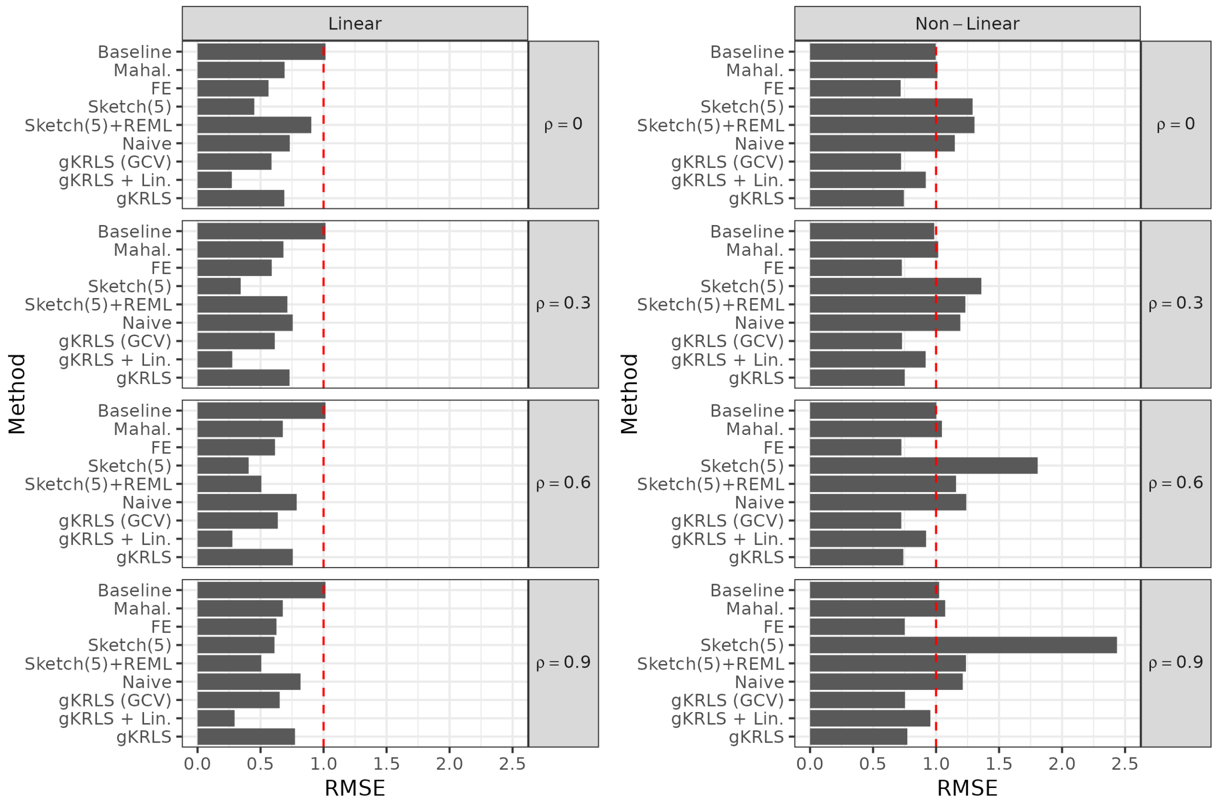
The first observation is that the “Baseline” gKRLS model that is as similar as possible to bigKRLS returns nearly equivalent performance. This corroborates the results in the first set of simulations (Section 4).

Next, consider the models that change only one feature of the “Baseline” model: “Mahal.”, “FE”, “Sketch(5)”. With a linear data generating process, all three result in improved performance. In the non-linear data generating process, “Sketch(5)” results in worse performance. After further investigating, this is seemingly a function of using GCV (Generalized Cross-Validation) to select λ; the bar below (“Sketch(5)+REML”) uses REML instead of GCV and finds a slight degradation in performance (see Appendix C.5) but nothing as catastrophic as with GCV. This may be due to some weaknesses of GCV, including a tendency to overfit for modestly sized datasets (see Wood 2011).

Of all of the simple modifications, note that “FE” is the only one that results in considerably improved performance. Thus, this provides evidence that putting the fixed effects outside of the kernel is what drives stronger performance of gKRLS.

Next, we examine the “Naive” specification; this simply swaps bigKRLS for gKRLS—using random sketching and Mahalanobis distance while keeping the fixed effects *inside* the

Figure A.11: Additional Versions of `gKRLS`



Note: This figure reports the RMSE of various methods for estimating the AME on $x_{i,1}$ relative to the RMSE of bigKRLS. The abbreviations are defined in the text.

kernel. In the non-linear setting, “Naive” incurs some penalty against a method with no sketching (“Baseline”). By contrast, “gKRLS (GCV)” that is the sketched, Mahalanobis, version of “FE” incurs a very slight penalty upon the unsketched version (“FE”). Methods that include fixed effects outside the kernel (“gKRLS (GCV)”, “gKRLS”, “gKRLS + Lin.”) improve considerably upon the “Baseline” and “Naive” model—especially with a non-linear data generating process. In terms of penalty parameter selection after using sketching and Mahalanobis distance (REML vs GCV; “gKRLS” and “gKRLS (GCV)”, respectively), we see a small negative impact for REML in the linear model, although there is little difference in the non-linear case.

Finally, we consider the “gKRLS + Lin.” specification that includes the group indicators *and* two covariates as unpenalized fixed effects $\boldsymbol{\beta}$ as well as a kernel on the two continuous covariates. In this model, there is one hierarchical term ($J = 1$) and as $\lambda \to \infty$, this model converges to a linear additive model of the group indicators and two covariates. By contrast, simply placing a kernel on the two variables (with no linear component) would converge a model that predicts the mean for all observations as $\lambda \to \infty$. We see that in the linear case, where this model is true, this specification has very strong performance. However, in the non-linear model, it incurs some penalty versus the “default” `gKRLS`. Thus, this suggests that deciding whether include linear terms in addition to kernel is either tested empirically or motivated based on what theory suggests is a reasonable limiting case for the model as $\lambda \to \infty$.

C.5 Impact of Sketching

Give the somewhat varying results of sketching procedures depending on whether the fixed effects are included in the kernel or not, we conducted one additional set of simulations to more systematically understand the impact of sub-sampling sketching in this more complicated example. We focus on sub-sampling sketching and conducted the following set of simulations using the same data generating process in Section 4.1.

- Generate a set of simulated data; for $\delta \in \{1, 3, 5, 7, 9, 11, 13, 15\}$, estimate `gKRLS` 100 times (i.e., varying the sketching matrix).
- We consider two quantities of interest. First, what is the RMSE of the average marginal effect on $x_{i,1}$ (the key quantity of interest in the main text) across the 100 repetitions for each δ? We define this as follows, where AME* is the true AME as calculated in our main simulations (see Appendix B.1) and $\text{AME}_\delta^{(m)}$ is the AME estimated for the m-th repetition with sketching multiplier δ.

$$\text{RMSE}_\delta = \sqrt{\frac{1}{100}\sum_{m=1}^{100}\left[\text{AME}_\delta^{(m)} - \text{AME}^*\right]^2}$$

Next, we also calculate the AME for the *unsketched* method, define this as $\text{AME}_{\text{unsketch}}$; we can calculate the corresponding RMSE without sketching as $\text{RMSE}_{\text{unsketch}}$—

corresponding to the absolute error the unsketched model is deterministic for a single dataset:

$$\text{RMSE}_{\text{unsketch}} = |\text{AME}_{\text{unsketch}} - \text{AME}^*| = \sqrt{[\text{AME}_{\text{unsketch}} - \text{AME}^*]^2}$$

- Given those quantities for a single simulated dataset, we repeat this process 150 times as the quality of the unsketched estimates can vary across datasets. Define $s \in \{1, \cdots, 150\}$ as indexing this "outer" simulation. We average the estimated RMSEs across simulations and create a measure of "relative impact" of sketching, i.e. the relative increase in (averaged) RMSE for using sketching versus the unsketched estimates.

$$\overline{\text{RMSE}}_{\delta} = \frac{1}{150}\sum_{s=1}^{150} \text{RMSE}_{\delta}^{(s)}; \quad \overline{\text{RMSE}}_{\text{unsketch}} = \frac{1}{150}\sum_{s=1}^{150} \text{RMSE}_{\text{unsketch}}^{(s)}$$

$$\text{RelativeImpact}_{\delta} = \frac{\overline{\text{RMSE}}_{\delta} - \overline{\text{RMSE}}_{\text{unsketch}}}{\overline{\text{RMSE}}_{\text{unsketch}}}$$

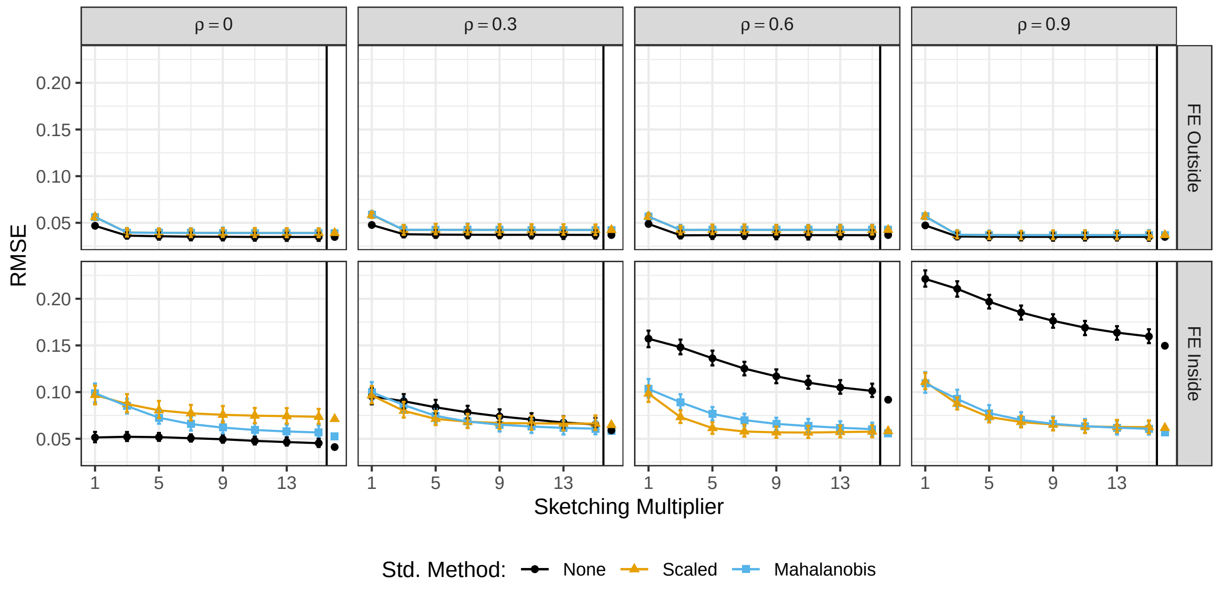
We consider all four ρ from the original simulations, sub-sampling sketching, and three different standardizations methods for the covariates (none [no standardization], scaled [i.e., all covariates have zero mean and unit variance], and Mahalanobis). Figure A.12 begins by showing the RMSE, averaged across simulations, for each combination. "FE Outside" denotes a model where the fixed effects are included outside the kernel and "FE Inside" denotes the one where they are included inside. We include the average RMSE of unsketched estimates as a dot on the right side of the figure after the black vertical bar.

As expected from the earlier simulations, including the fixed effects outside the kernel is highly amendable to sketching; at almost any multiplier, the RMSE is very close to the unsketched estimates, and this does not seem especially dependent on ρ or the standardization method.

By contrast, including the fixed effects inside the kernel shows more nuanced results. Before examining sketching, we note that including the fixed effects inside the kernel results in worse performance, on average, at every δ, ρ and standardization method than including them outside the kernel. However, given that choice of specification, there is more divergence between the standardization methods (especially with poor performance for no standardization as ρ grows). For this complex kernel, the convergence in the estimated RMSE to the unsketched method is much slower (i.e., a large multiplier δ is needed to closely approximate the RMSE of the unsketched method).

To show this more clearly, we report the standardized change in RMSE for sketching: $\text{RelativeImpact}_{\delta}$, defined above. This can be interpreted the proportional increase in RMSE that comes from using sketching with multiplier δ versus the unsketched method. Figure A.13 corroborates the above results. After a sketching multiplier of around $\delta = 3$, the RMSE for sketching when the fixed effects are outside the kernel are nearly identical to the unsketched RMSE. When fixed effects are included inside the kernel, there is a much

Figure A.12: RMSE For Varying Sketching Multiplier



Note: The RMSE for each sketching multiplier, ρ and model specification is shown. 95% confidence intervals, using a percentile bootstrap over the 150 datasets using 1,000 bootstrap samples, are shown. The RMSE of the unsketched method is shown to the right of the vertical bar.

more gradual decline in the relative impact of using sketching on the RMSE versus the unsketched estimates.

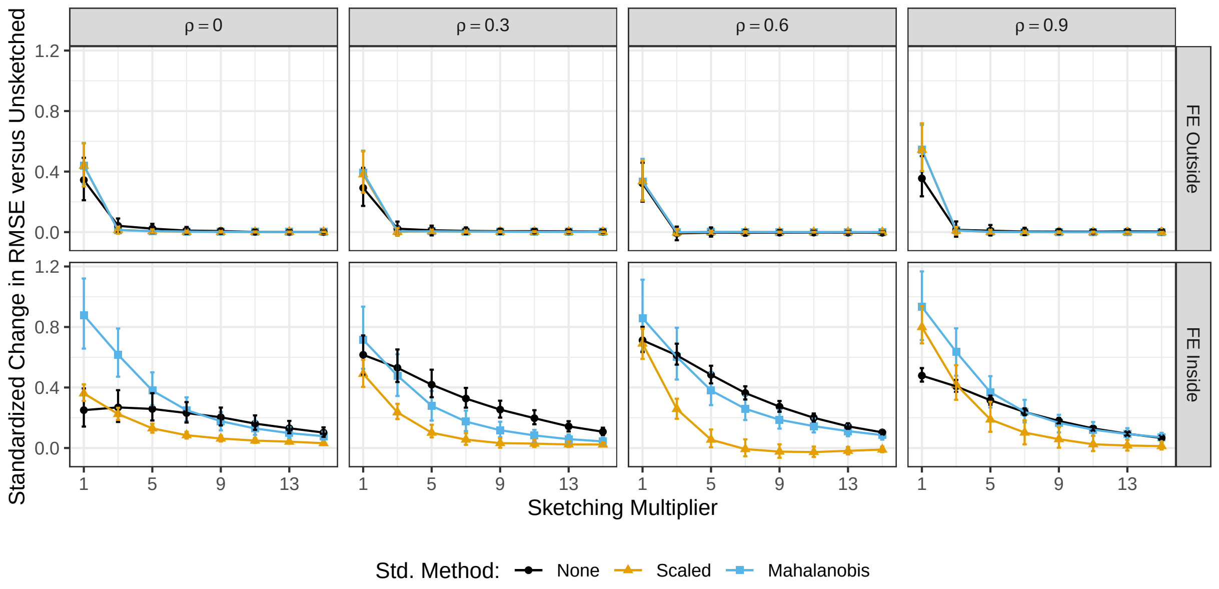
Interestingly, scaled covariates (mean-zero; variance one) seems to improve more quickly than Mahalanobis relative to the unsketched baseline, although note that the *absolute* performance of scaled vs. Mahalanobis standardization depends on ρ (see Figure A.12.

Overall, while more work is needed to understand the impacts of sketching with these complex kernels, we note that a likely cause is the fact that the sketched kernel likely contains zero or a few observations for each group and thus the estimation may be unreliable. This intuition is discussed further in Yang, Pilanci and Wainwright (2017) (Example 4) about issues with sub-sampling sketching when there is considerable variation in the data. Future research might explore better methods for sketching to improve performance even with kernels that include fixed effects (e.g., those discussed in Lee and Ng 2020). For example, stratified sampling based on the fixed effect may be helpful in improving performance.

D Additional Results: Newman (2016)

This appendix provides additional results for the analysis in Section 5. First, we provide information on the questions analyzed in Newman (2016). For the main analysis, the respondent is coded as "1", rejecting meritocracy, if they agreed with the statement pre-

Figure A.13: Relative Impact of Sketching



Note: This figure shows the relative impact of sketching (RelativeImpact$_\delta$) for each sketching multiplier, ρ and model specification (described in the main text). 95% confidence intervals are reported using a percentile bootstrap over the 150 datasets using 1,000 bootstrap samples.

sented to them or with both statements if both were presented—shown below. Otherwise, they are coded "0" (Newman 2016, p. 1013).

- "Success in life is pretty much determined by forces outside our control"
- "Hard work and determination are no guarantee of success for most people"

We consider the following two quantities. First, the average predicted probability $\bar{p}(e)$ as a function of earning inequality e where $\boldsymbol{x}_i$ indicates all other covariates in the model. Second, we consider the popular average marginal effect AME(e) where we evaluate the derivative using the finite difference method (Appendix A.3). Both quantities are calculated using the "observed value" strategy (e.g., Hanmer and Kalkan 2013) where one covariate (inequality) is set to some counterfactual value, all other covariates are held at their observed values, and the quantity of interest is the average across the observed sample.

$$\bar{p}(e) = \frac{1}{N}\sum_{i=1}^{N} \Pr(Y_i = 1|\boldsymbol{x}_i, e_i = e) \tag{A.6}$$

$$\text{AME}(e) = \frac{1}{N}\sum_{i=1}^{N} \left.\frac{\partial \Pr(Y_i = 1|\boldsymbol{x}_i, e_i)}{\partial e_i}\right|_{e_i=e} \tag{A.7}$$

For a secondary analyses as to the lack of a statistically detectable non-linear effect of economic inequality, we considered the following tests. Define e^*_{min} as the smallest value of earnings inequality considered (0.349), define e^*_{inf} as the model-specific inflection point—or the closest point in the grid of values considered, and define e^*_{max} as the largest value considered (1.119).

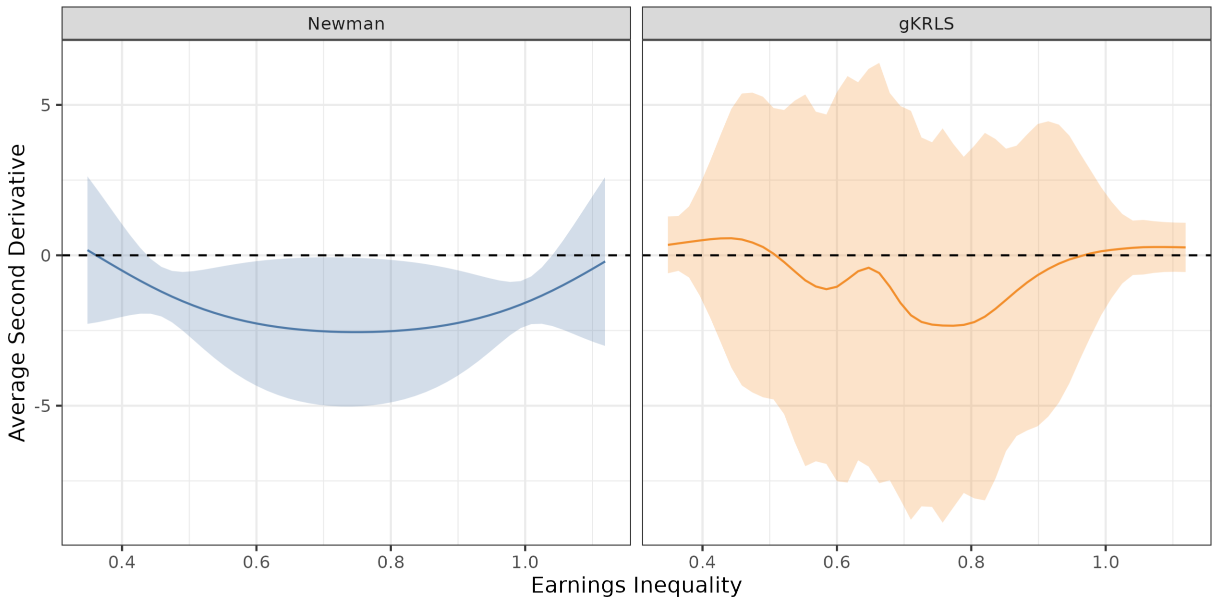
The first test examines the difference in average marginal effect at the end points, i.e. $\text{AME}(e^*_{max}) - \text{AME}(e^*_{min})$. For the original model in Newman (2016), the point estimate is -1.36 and the 95% confidence interval (using the delta method) is $[-2.05, -0.67]$. For the `gKRLS` model, the point estimate is -0.44 and the 95% confidence interval $[-1.28, 0.40]$ does contain zero. The second test compares the difference in changes in predicted probability above and below the inflection point, i.e. $\left[\bar{p}(e^*_{max}) - \bar{p}(e^*_{inf})\right] - \left[\bar{p}(e^*_{inf}) - \bar{p}(e^*_{min})\right]$. The 95% confidence interval on this quantity for the Newman (2016) model is $[-0.58, -0.07]$. The 95% confidence interval for `gKRLS` is $[-0.36, 0.04]$ and does contain zero.

The third test considers the average *second* derivative, i.e., average of the derivatives of the individual marginal effects:

$$\frac{1}{N}\sum_{i=1}^{N} \left.\frac{\partial^2 \text{Pr}(Y_i = 1|\boldsymbol{x}_i, e_i)}{\partial e_i^2}\right|_{e_i = e}$$

This tests the idea that for a concave function (as posited by Newman 2016, p. 1011), the second derivative should be negative. Figure A.14 shows that, for the model used in Newman (2016), the average second derivative is usually negative and its confidence intervals do not contain zero except in the most extreme regions. For `gKRLS`, however, the confidence intervals contain zero at all points considered.

Figure A.14: Average Second Derivative of Earnings Inequality



Thus, none of the tests shows evidence of a statistically distinguishable effect of a "u-shaped" relationship using gKRLS.

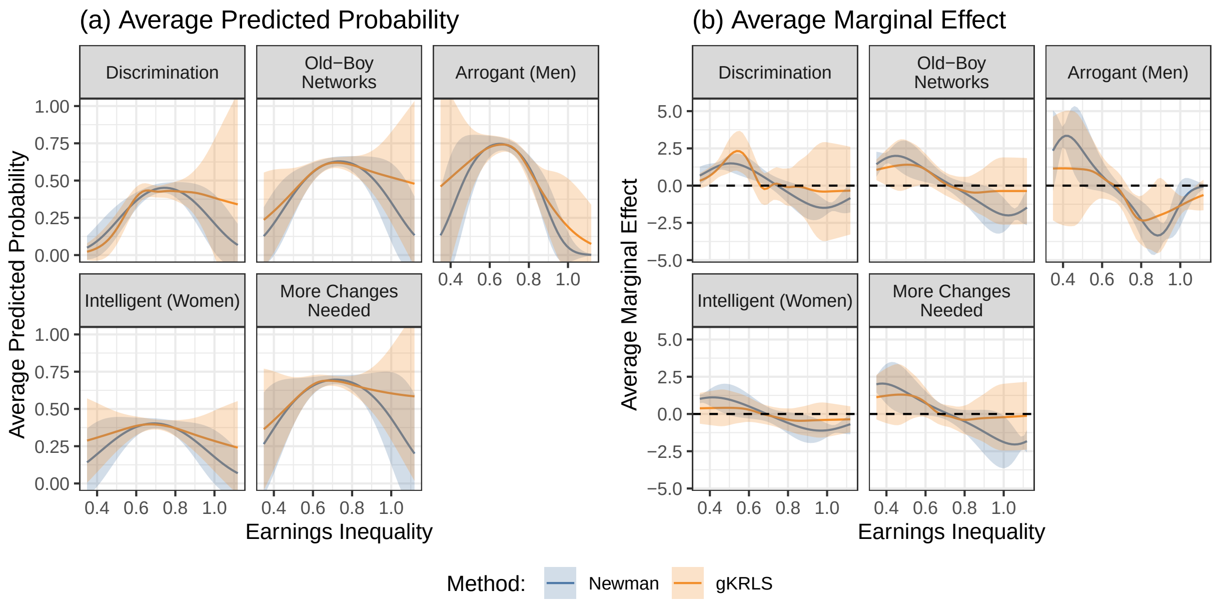
D.1 Additional Questions

The final test uses a secondary analysis in Newman (2016) where, on a different survey, the following five questions were analyzed. The below text is quoted directly from the supporting information in Newman (2016). Each question is colored in red with our annotation. For the two ordinal questions, Newman (2016) focuses on the probability of "major reason".

> **Barriers to Women's Professional Advancement**
>
> "As you may know, although women have moved into the work force in great numbers, very few top level business positions in this country are filled by women. There may be many reasons that there are so few women in high corporate positions. Here is a list of some of them. For each one, would you tell me whether you think it is a major reason, a minor reason, or not a reason why." (1) Discrimination: "Women are discriminated against in all areas of life, and business is no exception", and (2) Old-Boy Networks: "Women who try to rise to the top of major corporations get held back by the 'old-boy network"' [Q15B & Q15E]. Constructed variables each have three ordered categories, ranging from (1)-"Not a reason" (2)-"Minor reason" (3)-"Major reason."
>
> **Traits of Men and Women**
>
> "Now I would like to ask about some specific characteristics of men and women. For each one I read, please tell me whether you think it is generally more true of men or more true of women": (1) Intelligent [Q11A], and (2) Arrogant [Q11G]. For "Intelligent," constructed variable is dichotomous, and coded "1" for respondents who believed the trait is "More true of women" and "0" otherwise. For "Arrogant," constructed variable is dichotomous, and coded "1" for respondents who believed the trait is "More true of men," and "0" otherwise.
>
> **Gender Equality in Nation**
>
> "Which of these two statements comes closer to your own views—even if neither is exactly right": "This country has made most of the changes needed to give women equal rights with men" OR "The country needs to continue making changes to give women equal rights with men." [Q13]. Constructed variable coded "1" if respondent selected latter statement and "0" otherwise.

Figure A.15 shows the predicted probabilities and average marginal effects as a function of earnings inequality. As before, while the questions generally show an inverted "u-shape" estimated using gKRLS, there are concerns about a lack of a statistically detectable relationship. For most questions, the confidence intervals for the average marginal effects usually contain zero for most of the values for gKRLS but not for the specification in Newman (2016). Applying the additional tests in the previous section of the Appendix shows limited evidence for a statistically detectable effect for gKRLS—only one question passes either of the first two tests (men arrogant; difference in differences of $\bar{p}(e)$). The confidence interval on the average second derivative does not contain zero (and is negative) for many values of earnings inequality with the Newman (2016) model, but usually contains zero for the gKRLS model; Figure A.16 shows the results.

Figure A.15: Additional Questions in Newman (2016)



E Additional Results: Gulzar et al. (2020)

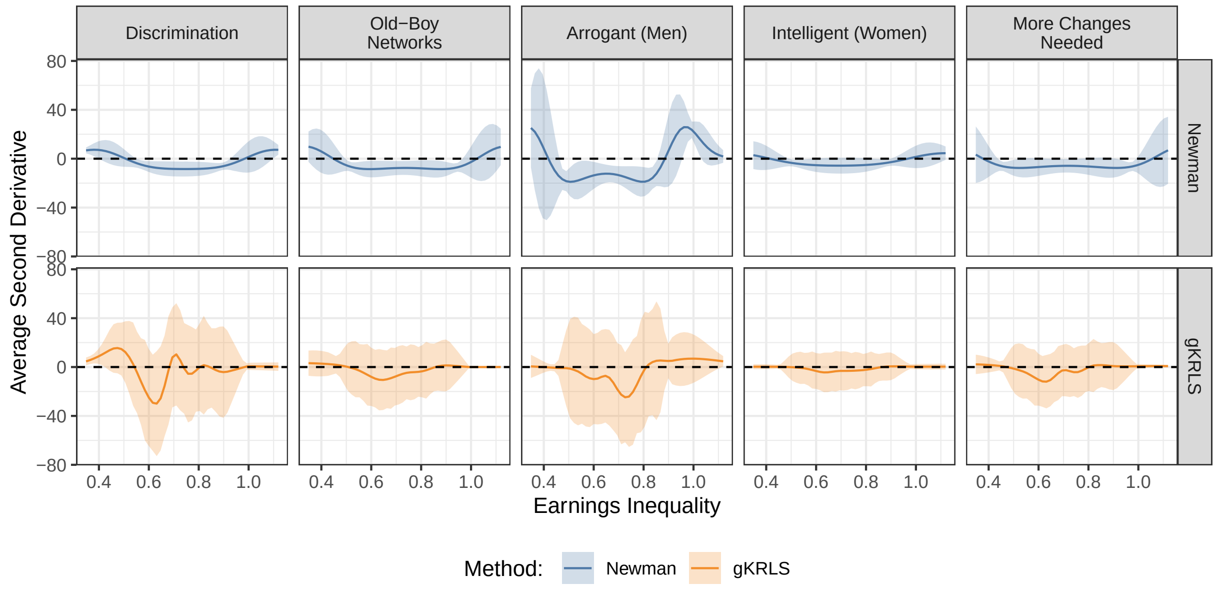
This appendix provides additional results for the analysis in Section 6.

E.1 Additional Results: Main Effects (Section 6)

We examine the sensitivity of the estimates to (i) the size of the sketching multiplier (5 or 15) and (ii) the use of bam or gam (see Appendix A.2 for a discussion). For each of the twelve outcomes, we ran the model fifty times. Note that for the double/debiased machine learning method (DML-PLR; DML-ATE), there is an additional source of randomness—the five folds that the data is split into—that is also accounted for in the uncertainty.

Figure A.17 presents the results for the two double/debiased machine learning methods (DML); the partially linear regression model (DML-PLR) and the average treatment effect model (DML-ATE). 95% confidence intervals are shown and the estimated results are sorted by their point estimates for clarity. We see that, for both methods, there is a large

Figure A.16: Average Second Derivative Additional Questions in Newman (2016)



amount of stability in the confidence intervals across repeated runs of the double/debiased machine learning procedure and the accompanying random sketching. It is rarely the case that re-running the model would change whether the confidence interval contains zero. The choice of `bam` vs `gam` and the multiplier also seem somewhat less important here. Across all methods, the ratio of the standard deviation of the point estimates to the average standard error is around 0.40 (for DML-ATE) and 0.20-0.25 (for DML-PLR).

Figure A.18 considers repeated estimation of the two `gKRLS` models that include either all variables in the kernel ("gKRLS (All)") or only the geographic coordinates ("gKRLS (Geog.)"), both described in the main text. It shows a broadly similar story to the DML methods. There is somewhat more variability when all variables are included in the kernel. Especially in this case, using the larger multiplier ($\delta = 15$) decreases the ratio of the standard deviation of the estimates to the average standard error from around 0.40 (comparable to DML-ATE) to around 0.20 (around 0.30 for `bam`).

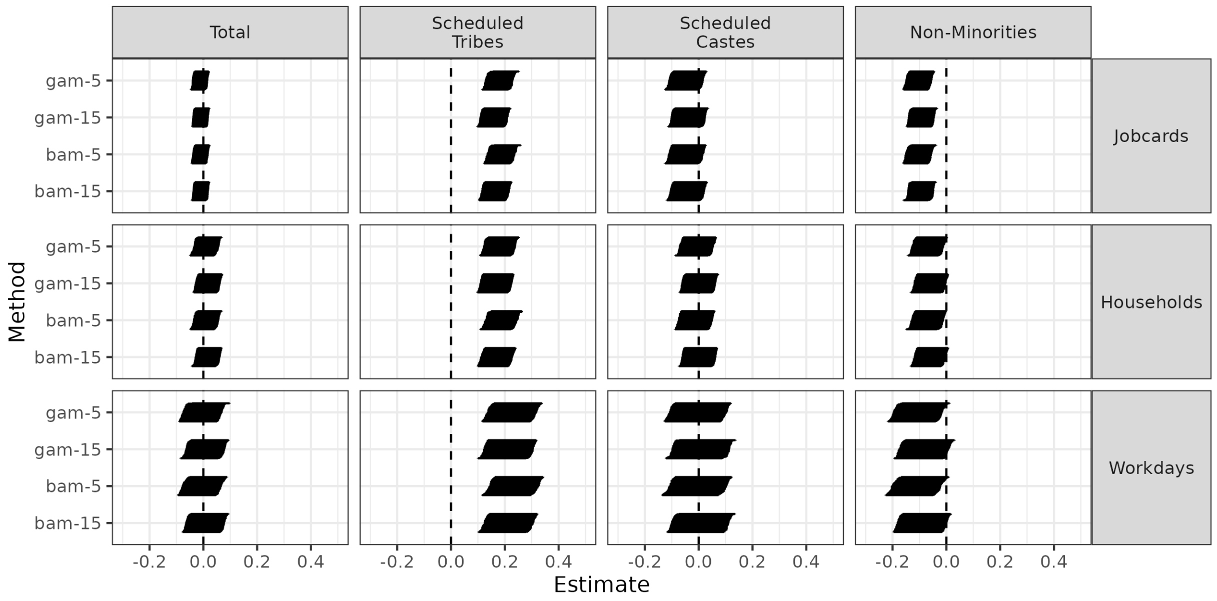
In terms of computational cost, Table A.3 reports the average estimation time for each method, averaged across all repeated estimations and outcome variables. It shows, as expected, that increasing the sketching multiplier can increase the cost considerably—especially for double/debiased machine learning methods or a kernel with many covariates ("gKRLS (All)"). `bam` provides considerable increases in speed especially in the case of the DML methods.

E.2 Additional Results: Heterogeneous Effects

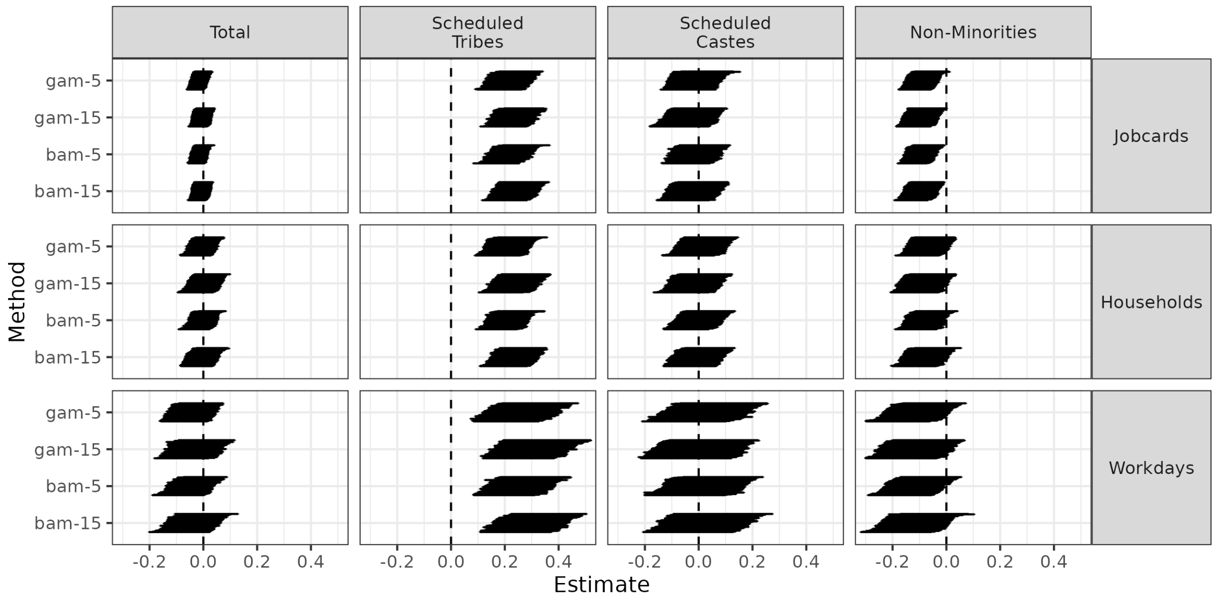
We conduct an additional application of `gKRLS` on the Gulzar, Haas and Pasquale (2020) to estimate heterogeneous effects. There is a large and active literature on how to use

Figure A.17: Repeated Estimation of Double/Debiased Machine Learning with Sketching

(a) Partially Linear Regression (PLR)



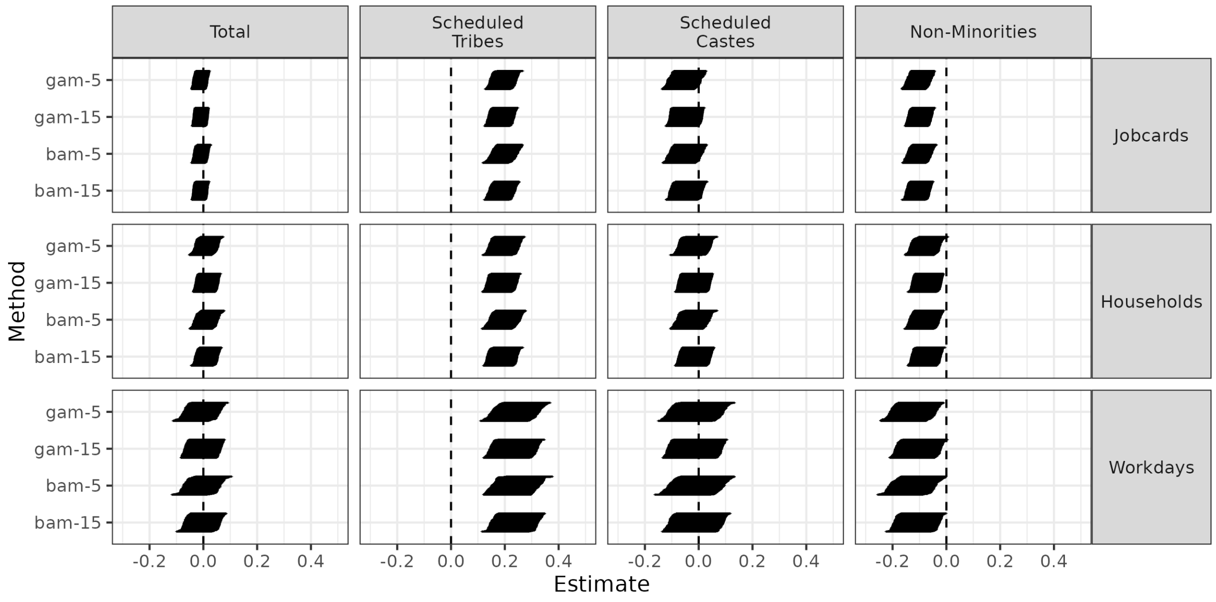
(b) Average Treatment Effect (ATE)



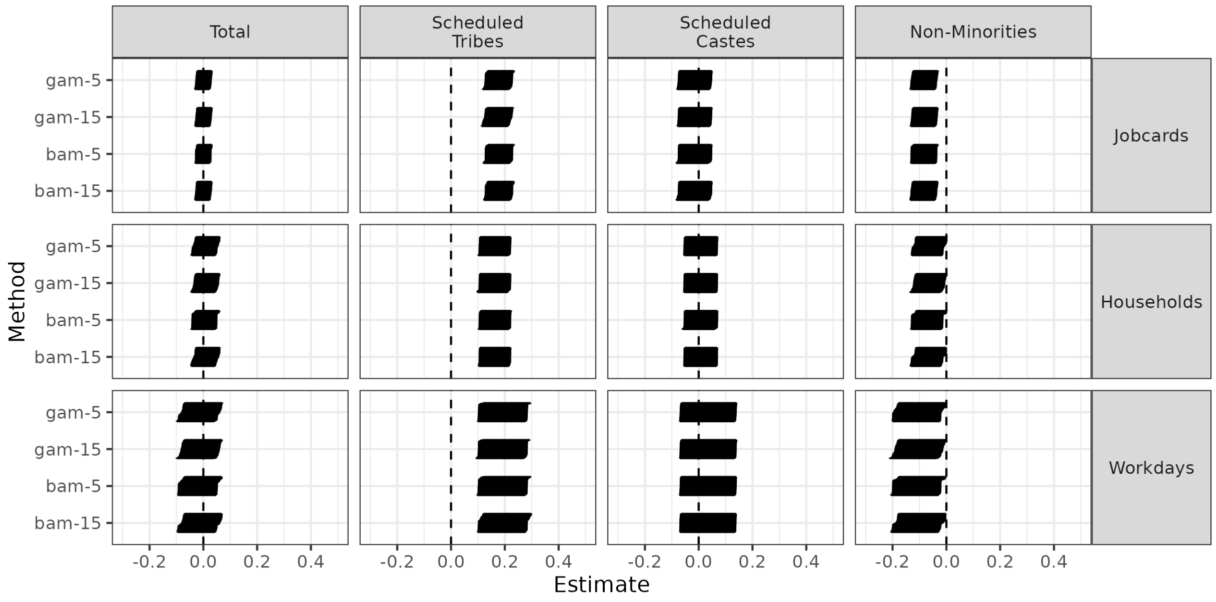
Note: This figure reports the estimated effects and 95% confidence intervals from fifty repetitions of each model for each outcome and group. The horizontal axis is truncated as occasionally the confidence interval from a method is very large. The top panel shows thee results of DML partially linear regression (DML-PLR) and the bottom shows the results for the DML algorithm for estimating the ATE (DML-ATE).

Figure A.18: Repeated Estimating `gKRLS` with Sketching

(a) All Variables in Kernel



(b) Geographic Kernel



Note: This figure reports the estimated effects and 95% confidence intervals from fifty repetitions of each model for each outcome and group. The horizontal axis is truncated as occasionally the confidence interval from a method is very large. The top panel shows the results of `gKRLS` that includes all covariates ("gKRLS (All)") and the bottom includes results from a method that only includes the geographic coordinates ("gKRLS (Geog.)" .

Table A.3: Run Time of DML and gKRLS Methods on Gulzar et al. (2020)

Method	DML		gKRLS	
	PLR	ATE	Geog	All
bam-5	0.63	1.37	0.04	0.07
bam-15	2.31	6.40	0.09	0.26
gam-5	7.54	5.71	0.48	1.00
gam-15	45.14	38.15	0.63	5.93

Note: This table reports the average estimation time in minutes on a computer with 8GB of RAM and 1 core, averaged across the simulations and outcome variables. "DML" reports the time of two methods (DML-PLR; DML-ATE) discussed in the main text. "`gKRLS`" reports the two kernel methods ("gKRLS (Geog.); "gKRLS (All)") discussed in the main text. "Method" indicates the method used with `bam` or `gam` and the sketching multiplier ("-5"; "-15").

arbitrary machine learning algorithms to estimate heterogeneous effects (e.g., Künzel et al. 2019; Nie and Wager 2021). We focus on a recent method ("R-learner") by Nie and Wager (2021) as representing a current state-of-the-art method.

The R-learner is a type of meta-learner (see Künzel et al. 2019 for a general review) that allows the researcher to use an arbitrary machine learning algorithm to estimate heterogeneous treatment effects. The core innovation of the R-learner is to estimate a heterogeneous treatment effect function $\tau^*(.)$ (as a function of a vector of pre-treatment covariates X_i—noting their notation *differs* from that in the rest of our paper) by minimizing the following empirical analogue $\tilde{\tau}(.)$ on a dataset with N observations where $m^*(x)$ and $e^*(x)$ represent the conditional mean outcome—$E[Y_i|X_i = x]$—and treatment propensity—$P(W_i = 1|X_i = x)$, respectively. $\Lambda_n\{\tau(.)\}$ regularizes the estimated $\tau(.)$ function.

$$\tilde{\tau}(.) = \underset{\tau}{\operatorname{argmin}} \frac{1}{N}\sum_{i=1}^{N} \left([Y_i - m^*(X_i)] - [W_i - e^*(X_i)]\,\tau(X_i)\right)^2 + \Lambda_n\{\tau(.)\} \tag{A.8}$$

One of the meanings of "R" in R-learner gestures at the fact that this function depends on *residualizing* the observed outcome Y_i and the treatment W_i from their expected values ($m^*(X_i)$ and $e^*(X_i)$, respectively). This ensures a more robust objective function when trying to estimate the heterogeneous treatment effects (Nie and Wager 2021). The key difficulty of estimating $\tilde{\tau}(.)$ is that the two key functions $m^*(.)$ and $e^*(.)$ are unknown so $\tilde{\tau}(.)$ cannot be estimated directly. A second key innovation of Nie and Wager (2021, p. 301) is to first estimate m^* and e^* and use these "pilot estimates" to create a feasible version for estimating $\tilde{\tau}(.)$.

They propose obtaining estimates $\hat{m}$ and $\hat{e}$ using a procedure known as "cross-fitting." This process is similar in implementation to ensemble methods (stacking, SuperLearning) and starts by separating the data into K-folds. Using K-1 folds of the data, one estimates the conditional mean outcome (i.e. predict Y_i with covariates X_i) and the propensity score (i.e. predicting treatment W_i with covariates), and then generate predictions on the held out fold. By cycling through all of the folds, one gets an out-of-sample prediction for each observation. These are combined as shown below to estimate the heterogeneous treatment effect function $\hat{\tau}(.)$. Formally, the R-learner algorithm is sketched below (Nie and Wager, 2021, p. 301).

1. Split the data into K folds. Fit $\hat{m}$ and $\hat{e}$ using cross-fitting with some machine learning algorithm, i.e. hold out one fold and estimate the model using the other $K-1$.

2. Estimate $\tau(.)$ using the out-of-sample predictions for each observation. Define $\hat{m}^{-k(i)}(X_i)$ as the estimate of $m^*(x)$ that does not include the fold k of which i is a member; similarly define $\hat{e}^{-k(i)}(X_i)$. The objective is shown below where some machine learning algorithm is used to estimate $\hat{\tau}(.)$.[1]

$$\hat{\tau}(.) = \underset{\tau}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \left(\left[Y_i - m^{-k(i)}(X_i) \right] - \left[W_i - e^{-k(i)}(X_i) \right] \tau(X_i) \right)^2 + \Lambda_n\{\tau(.)\} \tag{A.9}$$

While any machine learning algorithm can be used for the R-learner, there is an especially interesting reason to use `gKRLS`. Nie and Wager (2021) prove that if one uses traditional KRLS in estimating $\hat{\tau}(.)$ (and some weaker assumptions on the quality of the estimates of $\hat{m}$ and $\hat{e}$), then the resulting estimator $\hat{\tau}(.)$ has a "quasi-oracle" property. Roughly speaking, this means that the accuracy on estimating $\hat{\tau}(.)$ is asymptotically equivalent to the accuracy one would obtain if the researcher *knew* $m^*(X_i)$ and $e^*(X_i)$ exactly. Thus, there is no loss of information from having to use the estimated analogues. This provides a formal justification for using the "pilot estimates" $\hat{m}^{-k(i)}(X_i)$ and $\hat{e}^{-k(i)}(X_i)$ in the final estimation of $\hat{\tau}(.)$.

In their numerical experiments, Nie and Wager (2021) use Sonnet and Hazlett (2018)'s implementation of traditional KRLS for Gaussian and binary outcomes for estimating $\hat{m}(.)$, $\hat{e}(.)$ and $\hat{\tau}(.)$. However, this has the disadvantages discussed in Sections 2 and 3 (e.g., requiring cross-validation for the binary case and being quite slow). Indeed, Nie and Wager (2021) consider a coarse grid of only thirteen values when calibrating λ and only examine problems of 500 or 1000 observations. Thus, `gKRLS` allows for the theoretical promise of the R-learner when combined with KRLS to be scaled to much larger datasets.

We apply this to the Gulzar, Haas and Pasquale (2020) application, using `gKRLS` for all machine learning procedures where we include the covariates linearly as well as kernel

[1] In the spirit of their accompanying code, we perform one final step of K-fold cross-validation (with the same folds) for the final estimation of $\hat{\tau}(.)$. If one were to estimate effects on truly out-of-sample data, the $\hat{\tau}(.)$ estimated on the entire training data could be used.

that includes all of the covariates. The formula in pseudo-code (see Appendix F) is shown below where "`x1 + x2 + ...`" indicates the covariates.[2]

```
y ~ x1 + x2 + ... + s(x1, x2, ..., bs = "gKRLS")
```

The procedure takes under three minutes on a machine with 8 GB of RAM and a single core. Our initial examinations of the estimated heterogeneous effects suggested a key role of geography. Figure A.19 shows that, in general, one state—Himachal Pradesh—exhibits quite different patterns of estimated treatment effects. It shows the distribution of effects for each observation in each state across the four groups and three outcomes. In total, their analysis includes nine states. We also include a symbol (■) to indicate the estimated treatment effect, without controls, for each variable inside of each state. It is reassuring that the treatment effects obtained from the R-learner are similar to those calculated by simply calculating the difference-in-means within each state.

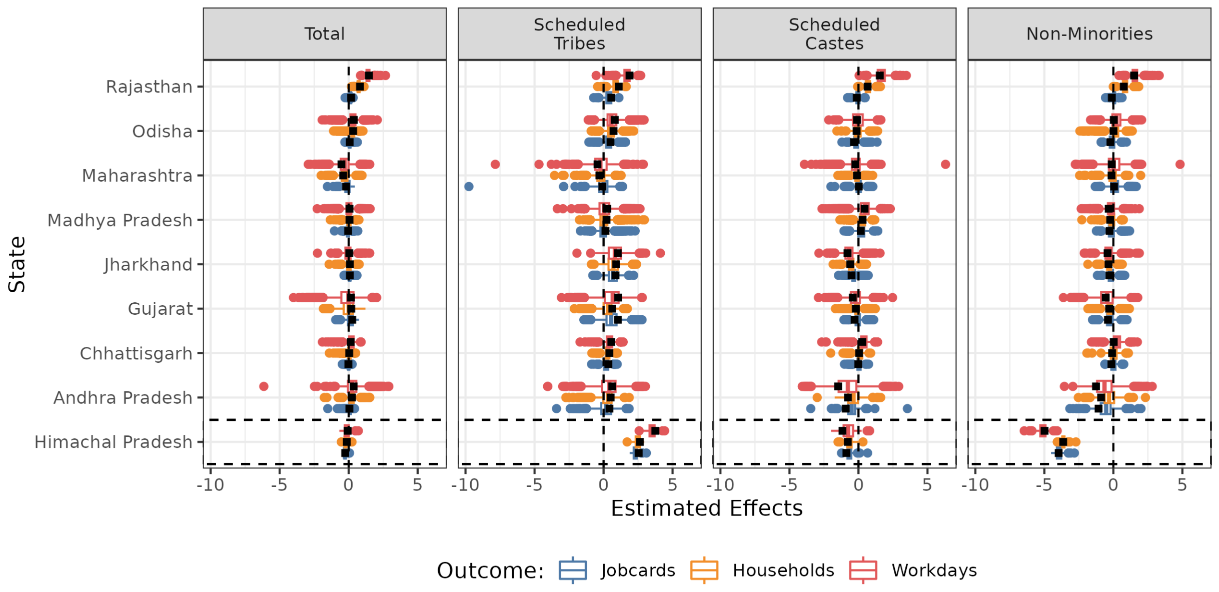
In substantive terms, Himachal Pradesh has much larger effects for the targeted minority group (Scheduled Tribes). We also find slightly negative effects for the non-targeted minority group (Scheduled Castes) and large negative effects for the other individuals (non-minorities). While fully exploring the reason for the differential effects in Himachal Pradesh is outside of the scope of this paper, we note that it is distinct amongst the states considered in that it (i) has the smallest population, (ii) has the largest share of the *non-targeted* minority group (Scheduled Castes; 25% of the population in 2001), and (ii) has the smallest share of the targeted group (4% of the population in 2001). This suggests a possibility of an interesting qualification of the effect of an electoral quota scheme to benefit a small minority group when there is another larger, although still historically disadvantaged group, who is not affected by the scheme. We found similar distributions of estimates by state if we used different multipliers, e.g., 5 vs. 15, and `gam` instead of `bam`.

F Software

This appendix briefly discusses software packages for estimating KRLS and provides a brief demonstration on how to use the core function in the `gKRLS` package.

[2]After our first explorations of the Gulzar, Haas and Pasquale (2020) data, we noted that two controls (share of scheduled tribes in 1991 and 2001) are extremely highly correlated with the treatment indicator in Himachal Pradesh (0.87 and 0.94, respectively; standard error of 0.039 and 0.027, respectively) but in no other state (correlations ranging from -0.05 and 0.25; standard errors ranging from 0.010 to 0.031). No other control variable's correlation with treatment within a state with treatment is higher than 0.47 (standard error of 0.070). Following supplemental analyses in the original paper, we find that this imbalance on "share of scheduled tribes in 2001" in Himachal Pradesh between treated and untreated units does not decrease as the geographic bandwidth decreases, although it does decline in all other states. The imbalance on "share of scheduled tribes in 1991" does decrease although more slowly. Exploring this and the implications for Gulzar, Haas and Pasquale (2020) in more detail is outside of the scope of this project. However, we exclude these two covariates for the heterogeneous effect analysis as their inclusion led to unstable estimates for Himachal Pradesh given minor changes in specification. Their exclusion has a limited effect on the results in the main text (e.g., Figure 5).

Figure A.19: Heterogeneous Effects in Gulzar, Haas and Pasquale (2020)



Note: This figure shows the distribution of heterogeneous treatment effects by state, outcome, and group using the R-learner approach described in the main text. The ■ indicates the treatment effect estimated within each state using a difference-in-means estimator.

F.1 Existing Software Packages

In the main paper, we consider `KRLS` and bigKRLS as the two well-known packages for estimating KRLS to political scientists. They are only able to assume a Gaussian outcome and must include all predictors in the kernel. An additional package `KSPM` (Schramm et al. 2020) provides additional flexibility by allowing multiple kernels and additive terms. It is still considerably less flexible than `mgcv` (e.g., a lack of other methods of penalization, the use of non-Gaussian outcomes, sketching, selection of penalty parameter using something other than LOO-CV, etc.). It does have some novel features (e.g., the use of alternative kernels besides the Gaussian one; interactions between kernels) that are straightforward to include in future development of `gKRLS`. Unfortunately, preliminary experiments also suggested that it was considerably slower than either `KRLS` or `mgcv` and thus it was not explored further in our empirical analyses.

F.2 Software Demonstration

We demonstrate how to estimate `gKRLS`, average marginal effects, and double/debiased machine learning.

```
# to install gKRLS
# install.packages('gKRLS')
# load SuperLearner before gKRLS/mgcv to address clashing object names
```

```r
library(SuperLearner)
library(gKRLS) # this also load "mgcv" and "sandwich"
library(DoubleML)
library(caret) # for sample splitting

# simulate data
n <- 1000
x1 <- rbinom(n, 1, 0.45)
x2 <- rnorm(n, 0, 1)
x3 <- rnorm(n, 0, 1)
x4 <- rnorm(n, 0, 1)
state <- as.factor(sample(letters[1:10], n, replace = TRUE))
error <- rnorm(n, 0, 0.5)

# target function
y <- 0.6*x1 +0.15*x2 -0.2*x1*x2 + 0.3*x3^3 - 0.2*x4^2 + error

data <- data.frame(y, x1, x2, x3, x4, state)
```

We estimate three versions of `gKRLS`; one that includes all variables in the kernel, one that includes some in a linear fashion (i.e., fixed effects for state), and one that includes two kernels.

```r
# All variables in kernel: "gam" is exported from "mgcv"
gkrls_est <- gam(y ~ s(x1, x2, x3, x4, bs = "gKRLS"),
    data = data)
summary(gkrls_est)

# State fixed effect
gkrls_fx <- gam(y ~ state + s(x1, x2, x3, x4, bs = "gKRLS"),
    data = data)

# Multiple kernels
gkrls_dk <- gam(
    y ~ state + s(x1, x2, bs = "gKRLS") +
        s(x3, x4, bs = "gKRLS"),
    data = data)
```

We can estimate predictions and marginal effects using the accompanying functions. If `individual=TRUE`, the individual marginal effect (i.e. for each observation) in addition to the average is returned. If marginal effects on only certain variables are desired, this can be specified using the `variables` argument.

```
gkrls_pred <- predict(gkrls_fx, newdata = data)
gkrls_ame <- calculate_effects(gkrls_fx,
    variables = c("x1", "x2"),
    continuous_type = "derivative", individual = T)
```

A variety of estimation options can be set using the `gKRLS` argument to `xt`. This is further described on the package documentation and some options are shown below, e.g. changing multiplier and sketching method.

```
gkrls_alt <- gam(y ~ s(x1, x2, x3, x4, bs = "gKRLS",
        xt = gKRLS(sketch_method = "gaussian",
            sketch_multiplier = 15)),
    data = data)
```

We can also integrate gKRLS with double/debiased maching learning method.

```
# double/debiased machine learning
ml_g <- LearnerRegrBam$new()
ml_g$param_set$values$formula <- ~ s(x2, x3, x4, bs = "gKRLS")
ml_m <- LearnerRegrBam$new()
ml_m$param_set$values$formula <- ~ s(x2, x3, x4, bs = "gKRLS")
ml_g$param_set$values$method <- "REML"
ml_m$param_set$values$method <- "REML"

data_DML <- double_ml_data_from_data_frame(
  df = data,
  y_col = "y",
  d_cols = "x1",
  x_cols = setdiff(names(data), c("y", "x1", "state"))
)
# Fit Partial Linear Regression
dml_plr <- DoubleMLPLR$new(data_DML, ml_g, ml_m)
dml_plr$fit()
```

Finally, we show how to use `gKRLS` to estimate heterogeneous effects using the `SuperLearner` package and the R-learner (Nie and Wager 2021) discussed in Section E.2.

```
# Set x1 as "treatment" for this analysis
data$treatment <- data$x1
data$x1 <- NULL
# Define function for R learner
heteff_SL <- function(...){suppressMessages(
    SL.mgcv(..., bam = T, method = "REML",
    formula = ~ s(x2, x3, x4, bs = "gKRLS")))
}
# Create the folds for post-stratification using
# caret. Ensure they are the same across both
# initial estimations
id <- createFolds(1:nrow(data), 5)

# Estimate the conditional mean function
fit_SL_m <- SuperLearner(Y = data$y,
 X = data, family = 'gaussian',
 SL.library = 'heteff_SL',
 cvControl = list(V = 5, validRows = id))

# Estimate the propensity score
fit_SL_e <- SuperLearner(Y = data$treatment,
 X = data, family = 'binomial',
 SL.library = 'heteff_SL',
 cvControl = list(V = 5, validRows = id),
 verbose = T)

# Extract estimated propensity scores
estimated_ps <- fit_SL_e$Z[,1]
# Truncate to avoid extreme scores
estimated_ps[estimated_ps < 0.01] <- 0.01
estimated_ps[estimated_ps > 0.99] <- 0.99
# Get the R-learner outcome
data$resid_PS <- data$y - estimated_ps
data$resid_outcome <- data$y - fit_SL_m$Z[,1]
data$rlearner_outcome <- data$resid_outcome/data$resid_PS

# Esimate heterogeneous effect
fit_SL_R <- SuperLearner(Y = data$rlearner_outcome,
 X = data, family = 'gaussian',
 obsWeights = data$resid_PS^2,
 SL.library = 'heteff_SL',
 cvControl = list(V = 5, validRows = id),
 verbose = T)
# Get the cross-validated estimates of heterogeneous
# treatment effect using held-out data
data$stacked.heteffect <- fit_SL_R$Z[,1]
# Get the estimates fit on the entire dataset
data$fullsample.heteffect <- fit_SL_R$SL.predict[,1]
```

```
# To predict for new out-of-sample data
predict(fit_SL_R, newdata = data[1:5,])
```

References

Cameron, A Colin and Douglas L Miller. 2015. “A Practitioner’s Guide to Cluster-Robust Inference.” *Journal of Human Resources* 50(2):317–372.

Gulzar, Saad, Nicholas Haas and Benjamin Pasquale. 2020. “Does Political Affirmative Action Work, and for Whom? Theory and Evidence on India’s Scheduled Areas.” *American Political Science Review* 114(4):1230–1246.

Hainmueller, Jens and Chad Hazlett. 2014. “Kernel Regularized Least Squares: Reducing Misspecification Bias with a Flexible and Interpretable Machine Learning Approach.” *Political Analysis* 22(2):143–168.

Hanmer, Michael J and Kerem Ozan Kalkan. 2013. “Behind the Curve: Clarifying the Best Approach to Calculating Predicted Probabilities and Marginal Effects from Limited Dependent Variable Models.” *American Journal of Political Science* 57(1):263–277.

Hazlett, Chad and Leonard Wainstein. 2022. “Understanding, Choosing, and Unifying Multilevel and Fixed Effect Approaches.” *Political Analysis* 30(1):46–65.

Künzel, Sören R, Jasjeet S Sekhon, Peter J Bickel and Bin Yu. 2019. “Metalearners for Estimating Heterogeneous Treatment Effects using Machine Learning.” *Proceedings of the National Academy of Sciences* 116(10):4156–4165.

Lee, Sokbae and Serena Ng. 2020. “An Econometric Perspective on Algorithmic Subsampling.” *Annual Review of Economics* 12:45–80.

Leeper, Thomas J. 2016. “Interpreting Regression Results using Average Marginal Effects with R’s margins.”.
URL: *https://s3.us-east-2.amazonaws.com/tjl-sharing/assets/AverageMarginalEffects.pdf*

Marra, Giampiero and Simon N Wood. 2012. “Coverage properties of confidence intervals for generalized additive model components.” *Scandinavian Journal of Statistics* 39(1):53–74.

Newman, Benjamin J. 2016. “Breaking the Glass Ceiling: Local Gender-Based Earnings Inequality and Women’s Belief in the American Dream.” *American Journal of Political Science* 60(4):1006–1025.

Nie, Xinkun and Stefan Wager. 2021. “Quasi-Oracle Estimation of Heterogeneous Treatment Effects.” *Biometrika* 108(2):299–319.

Schramm, Catherine, Sébastien Jacquemont, Karim Oualkacha, Aurélie Labbe and Celia M. T. Greenwood. 2020. "KSPM: A Package For Kernel Semi-Parametric Models." *The R Journal* 12(2):82–106.

Sonnet, Luke and Chad Hazlett. 2018. "Kernel Regularized Logistic Regression: Avoiding Misspecification Bias while Maintaining Interpretability for Binary Outcome Regressions." *Working Paper* .

Wood, Simon N. 2006. "On confidence intervals for generalized additive models based on penalized regression splines." *Australian & New Zealand Journal of Statistics* 48(4):445–464.

Wood, Simon N. 2011. "Fast Stable Restricted Maximum Likelihood and Marginal Likelihood Estimation of Semiparametric Generalized Linear Models." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 73(1):3–36.

Wood, Simon N. 2017. *Generalized Additive Models*. Chapman and Hall/CRC.

Wood, Simon N, Yannig Goude and Simon Shaw. 2015. "Generalized Additive Models for Large Data Sets." *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 64(1):139–155.

Yang, Yun, Mert Pilanci and Martin J Wainwright. 2017. "Randomized Sketches for Kernels: Fast and Optimal Nonparametric Regression." *The Annals of Statistics* 45(3):991–1023.