Supplementary Materials for "Scalable algorithms for semiparametric accelerated failure time models in high dimensions"

Piotr M. Suder and Aaron J. Molstad*
Department of Statistics and Genetics Institute
University of Florida, Gainesville, FL

1 Proofs and derivations

1.1 Proof of Lemma 1

Let $\beta = \beta^{(t)}$, $\Gamma = \Gamma^{(t-1)}$, $\phi = \phi^{(t)}$ for notational convenience. The new iterate is

$$\begin{aligned}\theta^{(t)} &= \arg\min_{\theta \in \mathbb{R}^{|\mathcal{D}|}} \left\{ f_{\mathcal{D}}(\theta) + \lambda g(\beta) + \Gamma^\top \left\{ \theta - \mathrm{P}_{\mathcal{D}}(\log y - X\beta) \right\} + \frac{\rho}{2} \left\| \theta - \mathrm{P}_{\mathcal{D}}(\log y - X\beta) \right\|_2^2 \right\} \\ &= \arg\min_{\theta \in \mathbb{R}^{|\mathcal{D}|}} \sum_{k=1}^{|\mathcal{D}|} \left[\frac{1}{n^2} \left\{ \tilde{\delta}_{k,1}(\theta_k)^- + \tilde{\delta}_{k,2}(-\theta_k)^- \right\} + \Gamma_k \theta_k + \frac{\rho}{2} \left\{ \theta_k - \left(\phi_k + \rho^{-1}\Gamma_k \right) \right\}^2 \right] \qquad (1)\end{aligned}$$

where each k corresponds to one pair $(i, j) \in \mathcal{D}$, ϕ_k is the kth component of $\phi = \mathrm{P}_{\mathcal{D}}(\log y - X\beta) - \rho^{-1}\Gamma$, and $\tilde{\delta}_{k,\cdot} = (\delta_i, \delta_j)$. Thus, the optimization problem in (1) can be expressed as $|\mathcal{D}|$ separate univariate optimization problems. Specifically, for $k = 1, \ldots, |\mathcal{D}|$,

$$\begin{aligned}\theta_k^{(t)} &= \arg\min_{\theta \in \mathbb{R}} \left\{ \frac{1}{n^2} \left\{ \tilde{\delta}_{k,1}(\theta)^- + \tilde{\delta}_{k,2}(-\theta)^- \right\} + \Gamma_k \theta + \frac{\rho}{2} \left[\theta - \left(\phi_k + \rho^{-1}\Gamma_k \right) \right]^2 \right\} \\ &= \arg\min_{\theta \in \mathbb{R}} \left\{ \frac{1}{n^2} \left\{ \tilde{\delta}_{k,1}(\theta)^- + \tilde{\delta}_{k,2}(-\theta)^- \right\} + \Gamma_k \theta + \frac{\rho}{2}\theta^2 - \rho\theta \left(\phi_k + \rho^{-1}\Gamma_k \right) \right\} \\ &= \arg\min_{\theta \in \mathbb{R}} \left\{ \frac{1}{n^2} \left\{ \tilde{\delta}_{k,1}(\theta)^- + \tilde{\delta}_{k,2}(-\theta)^- \right\} + \frac{\rho}{2}\theta^2 - \rho\phi_k\theta \right\} \qquad (2)\end{aligned}$$

*Correspondence: amolstad@ufl.edu

Let $h(\theta) = h_1(\theta) + \frac{\rho}{2}\theta^2 - \rho\phi_k\theta$ where $h_1(\theta) = \frac{1}{n^2}\left\{\tilde{\delta}_{k,1}(\theta)^- + \tilde{\delta}_{k,2}(-\theta)^-\right\}$. Because the optimization in (2) is convex, we know that θ is optimal for (2) if and only if $0 \in \partial h(\theta)$ or equivalently,

$$0 = z + \rho\theta - \rho\phi_k$$

where $z \in \partial h_1(\theta)$ with $\partial h_1(\theta)$ being the subdifferential of h_1 at θ. It can be verified that

$$\partial h_1(\theta) = \begin{cases} -\frac{\tilde{\delta}_{k,1}}{n^2} & \text{if } \theta < 0 \\ \frac{\tilde{\delta}_{k,2}}{n^2} & \text{if } \theta > 0 \\ \left[-\frac{\tilde{\delta}_{k,1}}{n^2}, \frac{\tilde{\delta}_{k,2}}{n^2}\right] & \text{if } \theta = 0 \end{cases}.$$

Then, we have that

$$\partial h(\theta) = \partial h_1(\theta) + \rho\theta - \rho\phi_k = \begin{cases} \rho(\theta - \phi_k) - \frac{\tilde{\delta}_{k,1}}{n^2} & \text{if } \theta < 0 \\ \rho(\theta - \phi_k) + \frac{\tilde{\delta}_{k,2}}{n^2} & \text{if } \theta > 0 \\ \left[\rho(\theta - \phi_k) - \frac{\tilde{\delta}_{k,1}}{n^2}, \rho(\theta - \phi_k) + \frac{\tilde{\delta}_{k,2}}{n^2}\right] & \text{if } \theta = 0. \end{cases}$$

We will now show that the choices described in Lemma 1 yield $0 \in \partial h(\theta)$ going case-by-case.

Case 1: Suppose that $\phi_k - \frac{\tilde{\delta}_{k,2}}{\rho n^2} > 0$. Then, setting $\theta = \phi_k - \frac{\tilde{\delta}_{k,2}}{\rho n^2}$

$$\partial h(\theta) = \partial h\left(\phi_k - \frac{\tilde{\delta}_{k,2}}{\rho n^2}\right) = \rho\left(\phi_k - \frac{\tilde{\delta}_{k,2}}{\rho n^2} - \phi_k\right) + \frac{\tilde{\delta}_{k,2}}{n^2} = -\frac{\tilde{\delta}_{k,2}}{n^2} + \frac{\tilde{\delta}_{k,2}}{n^2} = 0$$

Case 2: Suppose that $\phi_k + \frac{\tilde{\delta}_{k,1}}{\rho n^2} < 0$. Then, setting $\theta = \phi_k + \frac{\tilde{\delta}_{k,1}}{\rho n^2}$

$$\partial h(\theta) = \partial h\left(\phi_k + \frac{\tilde{\delta}_{k,1}}{\rho n^2}\right) = \rho\left(\phi_k + \frac{\tilde{\delta}_{k,1}}{\rho n^2} - \phi_k\right) - \frac{\tilde{\delta}_{k,1}}{n^2} = \frac{\tilde{\delta}_{k,1}}{n^2} - \frac{\tilde{\delta}_{k,1}}{n^2} = 0$$

Case 3: Suppose that $\phi_k - \frac{\tilde{\delta}_{k,2}}{\rho n^2} \leq 0$ and $\phi_k + \frac{\tilde{\delta}_{k,1}}{\rho n^2} \geq 0$. Then $-\rho\phi_k - \frac{\tilde{\delta}_{k,1}}{n^2} \leq 0$ and $-\rho\phi_k + \frac{\tilde{\delta}_{k,2}}{n^2} \geq 0$, so setting $\theta = 0$ yields

$$\partial h(\theta) = \partial h(0) = \left[-\rho\phi_k - \frac{\tilde{\delta}_{k,1}}{n^2}, -\rho\phi_k + \frac{\tilde{\delta}_{k,2}}{n^2}\right] \ni 0,$$

as required. ■

1.2 Derivation of β update

In this section, we provide a derivation of the updating equation (16) from the main manuscript. Notice

$$
\begin{aligned}
\beta^{(t)} &= \arg\min_{\beta\in\mathbb{R}^p} \Big\{ \lambda g(\beta) + \frac{\rho}{2}\|\theta^{(t-1)} + \rho^{-1}\Gamma^{(t-1)} - \mathrm{P}_{\mathcal{D}}\left(\log y - X\beta\right)\|_2^2 \\
&\qquad\qquad + \frac{\rho}{2}(\beta - \beta^{(t-1)})^\top \left(\eta I_p - X^\top \mathrm{P}_{\mathcal{D}}^\top \mathrm{P}_{\mathcal{D}} X\right)(\beta - \beta^{(t-1)})\Big\} \\
&= \arg\min_{\beta\in\mathbb{R}^p} \Big\{ \left(\theta^{(t-1)} + \rho^{-1}\Gamma^{(t-1)} - \mathrm{P}_{\mathcal{D}}\log y + \mathrm{P}_{\mathcal{D}} X\beta\right)^\top \left(\theta^{(t-1)} + \rho^{-1}\Gamma^{(t-1)} - \mathrm{P}_{\mathcal{D}}\log y + \mathrm{P}_{\mathcal{D}} X\beta\right) + \\
&\qquad\qquad \frac{2\lambda}{\rho} g(\beta) + (\beta - \beta^{(t-1)})^\top \left(\eta I_p - X^\top \mathrm{P}_{\mathcal{D}}^\top \mathrm{P}_{\mathcal{D}} X\right)(\beta - \beta^{(t-1)})\Big\}
\end{aligned}
$$

so that ignoring terms not depending on β, we have

$$
\begin{aligned}
&= \arg\min_{\beta\in\mathbb{R}^p} \Big\{ \frac{2\lambda}{\rho} g(\beta) + \big(2\beta^\top X^\top \mathrm{P}_{\mathcal{D}}^\top \theta^{(t-1)} + 2\rho^{-1}\beta^\top X^\top \mathrm{P}_{\mathcal{D}}^\top \Gamma^{(t-1)} - 2\beta^\top X^\top \mathrm{P}_{\mathcal{D}}^\top \mathrm{P}_{\mathcal{D}} \log y + \\
&\qquad + \beta^\top X^\top \mathrm{P}_{\mathcal{D}}^\top \mathrm{P}_{\mathcal{D}} X\beta\big) + \big(\eta\beta^\top\beta - \beta^\top X^\top \mathrm{P}_{\mathcal{D}}^\top \mathrm{P}_{\mathcal{D}} X\beta - 2\eta\beta^\top\beta^{(t-1)} + 2\beta^\top X^\top \mathrm{P}_{\mathcal{D}}^\top \mathrm{P}_{\mathcal{D}} X\beta^{(t-1)}\big)\Big\} \\
&= \arg\min_{\beta\in\mathbb{R}^p} \Big\{ \frac{2\lambda}{\rho} g(\beta) + \eta\beta^\top\beta + 2\beta^\top \big(X^\top \mathrm{P}_{\mathcal{D}}^\top \theta^{(t-1)} + \rho^{-1} X^\top \mathrm{P}_{\mathcal{D}}^\top \Gamma^{(t-1)} + \\
&\qquad\qquad - X^\top \mathrm{P}_{\mathcal{D}}^\top \mathrm{P}_{\mathcal{D}} \log y - \eta\beta^{(t-1)} + X^\top \mathrm{P}_{\mathcal{D}}^\top \mathrm{P}_{\mathcal{D}} X\beta^{(t-1)}\big)\Big\} \\
&= \arg\min_{\beta\in\mathbb{R}^p} \Big\{ \frac{2\lambda}{\rho\eta} g(\beta) + \beta^\top\beta + 2\beta^\top \Big(\frac{1}{\eta} X^\top \mathrm{P}_{\mathcal{D}}^\top \theta^{(t-1)} + \frac{1}{\eta}\rho^{-1} X^\top \mathrm{P}_{\mathcal{D}}^\top \Gamma^{(t-1)} + \\
&\qquad\qquad - \frac{1}{\eta} X^\top \mathrm{P}_{\mathcal{D}}^\top \mathrm{P}_{\mathcal{D}} \log y + \frac{1}{\eta} X^\top \mathrm{P}_{\mathcal{D}}^\top \mathrm{P}_{\mathcal{D}} X\beta^{(t-1)} - \beta^{(t-1)}\Big)\Big\} \\
&= \arg\min_{\beta\in\mathbb{R}^p} \left[\frac{\lambda}{\rho\eta} g(\beta) + \frac{1}{2}\|\beta - \eta^{-1} X^\top \mathrm{P}_{\mathcal{D}}^\top \left\{\mathrm{P}_{\mathcal{D}}\left(\log y - X\beta^{(t-1)}\right) - \theta^{(t-1)} - \rho^{-1}\Gamma^{(t-1)}\right\} - \beta^{(t-1)}\|_2^2 \right] \\
&= \mathrm{Prox}_{(\lambda/\rho\eta)g}\left[\frac{1}{\eta} X^\top \mathrm{P}_{\mathcal{D}}^\top \left\{\mathrm{P}_{\mathcal{D}}(\log y - X\beta^{(t-1)}) - \theta^{(t-1)} - \rho^{-1}\Gamma^{(t-1)}\right\} + \beta^{(t-1)} \right].
\end{aligned}
$$

2 Variable selection accuracy results

Beyond model error and concordance, another important performance metric for an estimator is variable selection accuracy. The variable selection accuracy of $\hat{\beta}$, an estimator of β_*, can

be quantified through the true positive and true negative variable selection rates,

$$\frac{|\{j : \hat{\beta}_j \neq 0 \cap \beta_{*j} \neq 0, j \in [p]\}|}{|\{j : \beta_{*j} \neq 0, j \in [p]\}|}$$

and

$$\frac{|\{j : \hat{\beta}_j = 0 \cap \beta_{*j} = 0, j \in [p]\}|}{|\{j : \beta_{*j} = 0, j \in [p]\}|},$$

respectively, where $|\mathcal{A}|$ denotes the cardinality of a set $\mathcal{A}$. An estimate with a true positive rate of one has correctly identified all nonzero entries of β_*, but may have included many false positives. Conversely, an estimate with a true negative rate of one correctly identifies all entries which are zero, but may include many false negatives. Thus together, true positive and true negative rates give a sense of the overall variable selection accuracy.

In Figure 6, we display both true positive and true negative rates for the four considered methods under the same settings which gave rise to Figure 3 of the main manuscript (i.e., errors following a logistic distribution and β_* having ten nonzero entries). For, WLS-Or, the oracle version of the weighted least squares approach, we recorded these metrics for the tuning parameter which yielded the smallest value of equation (3) from main manuscript evaluated on the testing set. In Figure 7, we display variable selection rates under group-lasso penalization: the settings here are exactly those from Figure 4 of the main manuscript. In Figure 8, we display variable selection resutls under the same settings as in Figure 6, but without censoring.

Overall, in Figures 6, 7, and 8, we see that the true positive rate of the rank-based estimator is substantially higher than of either of the weighted least squares estimators. In contrast, the weighted least squares estimator tends to have higher true negative rate than does the rank-based estimator. Together, these results suggest the weighted least squares estimator yields smaller models, but tends to omit many important variables. This may partly explain why these estimators are outperformed in both simulation studies and real data analyses in terms of model error and concordance. When there is no censoring, as one may expect, the regularized weighted least square estimators have slightly better variable selection accuracy than does the regularized Gehan estimator.

We display additional simulation results under normally distributed errors in the next subsection.

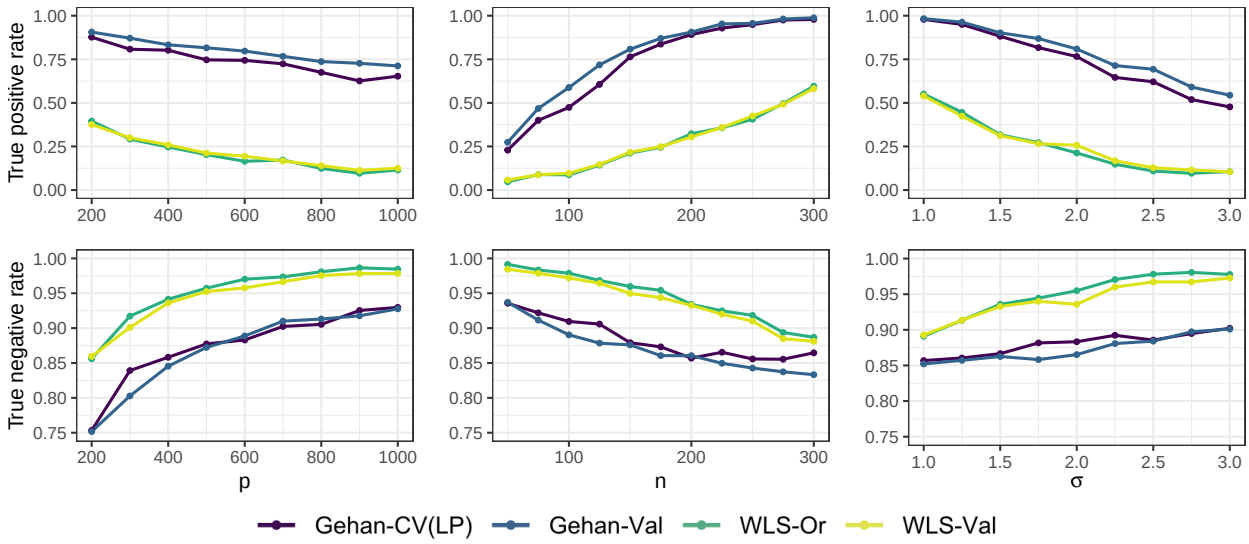

Figure 6: True positive rates (top row) and true negative rates (bottom row) for the four considered methods averaged over 100 independent replications with logistic errors, β_* having ten elements set equal to one, and g being the elastic net penalty with $\alpha = 0.5$.

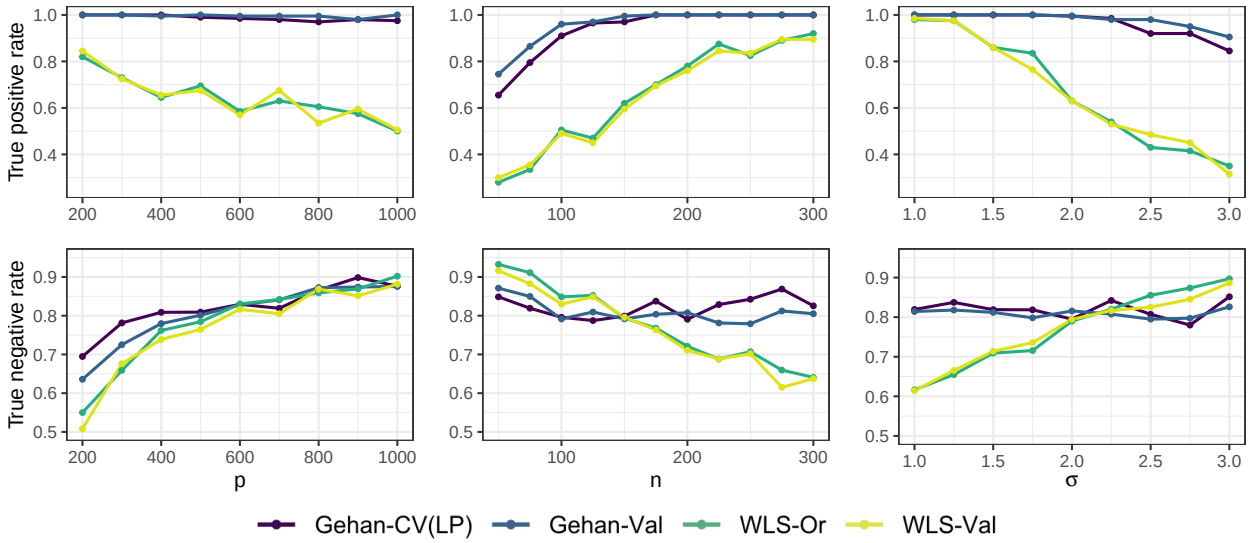

Figure 7: True positive rates (top row) and true negative rates (bottom row) for the four considered methods averaged over 100 independent replications with logistic errors, β_* having ten elements set equal to 0.5 (five in two different groups, with each group of size 10), and g being the group lasso penalty with $\alpha = 0$.

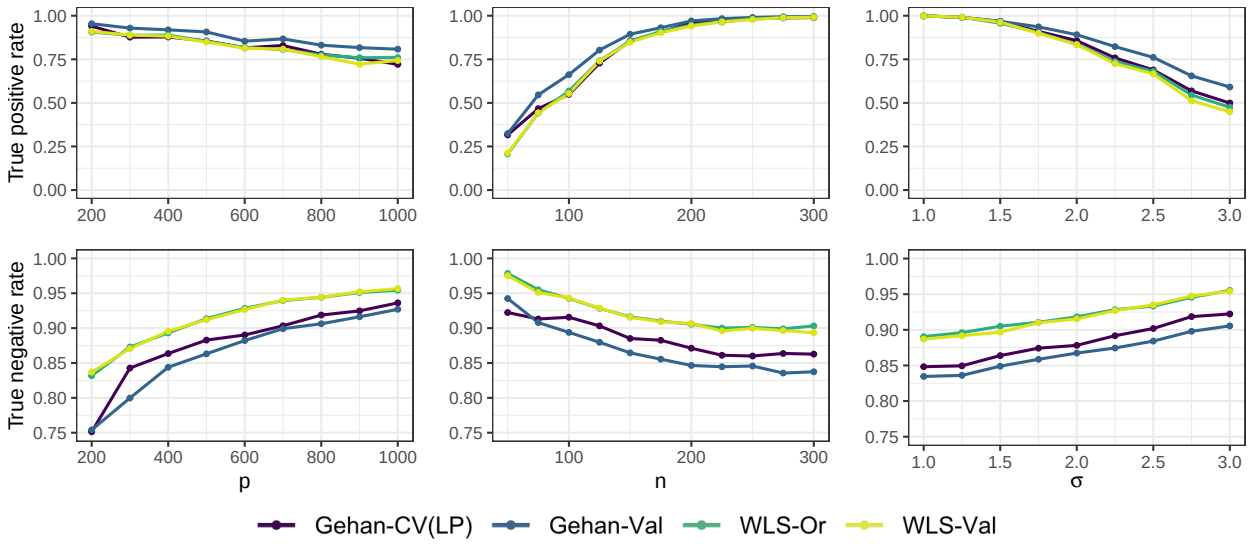

Figure 8: True positive rates (top row) and true negative rates (bottom row) for the four considered methods averaged over 100 independent replications with logistic errors, no censoring, β_* having ten elements set equal to one, and g being the elastic net penalty with $\alpha = 0.5$.

3 Simulation results under normal errors

In this section, we present additional simulation results mentioned in Section 6 of the manuscript. Figure 9 and 10 correspond to Figures 3 and 4 of the main manuscript, but under normal errors.

For the results presented in Figures 11 and 12, data were generated in the same manner as described in Section 6 with β_* having ten randomly selected components equal one and all others equal to zero. The only difference is that here, there is no censoring – all failure times are observed. This is an ideal scenario for the weighted least squares approach, particularly under normal errors, because one does not have to do any reweighted to account for censored observations.

We see under both logistic and normal errors without censoring, the methods' relative perform is remarkably similar. Under normal errors, the least squares estimator is better both in terms of model error and concordance under every scenario considered. With logistic errors, which have heavier tails than normal errors, the differences between the two approaches is less substantial. In many scenarios, the `Gehan-Val` performs as well as `WLS-Val`. When the error variance is large under logistic errors, even `Gehan-CV(LP)` outperforms the `WLS` estimators.

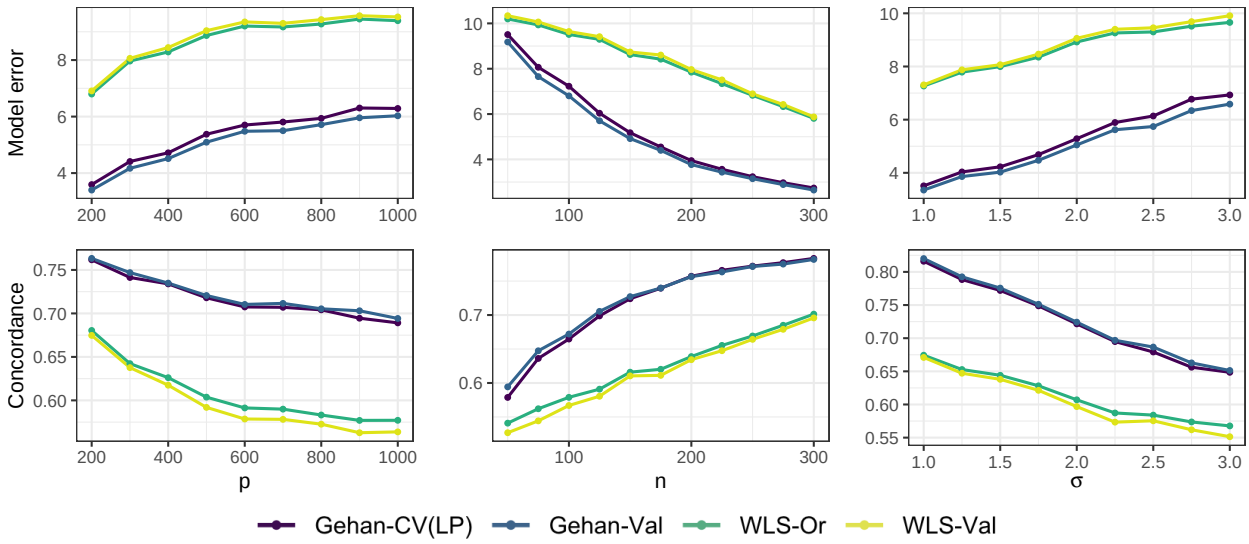

Figure 9: Model error (top row) and concordance (bottom row) for the four considered methods averaged over 100 independent replications with normal errors, β_* having ten elements set equal to one, and g being the elastic net penalty with $\alpha = 0.5$.

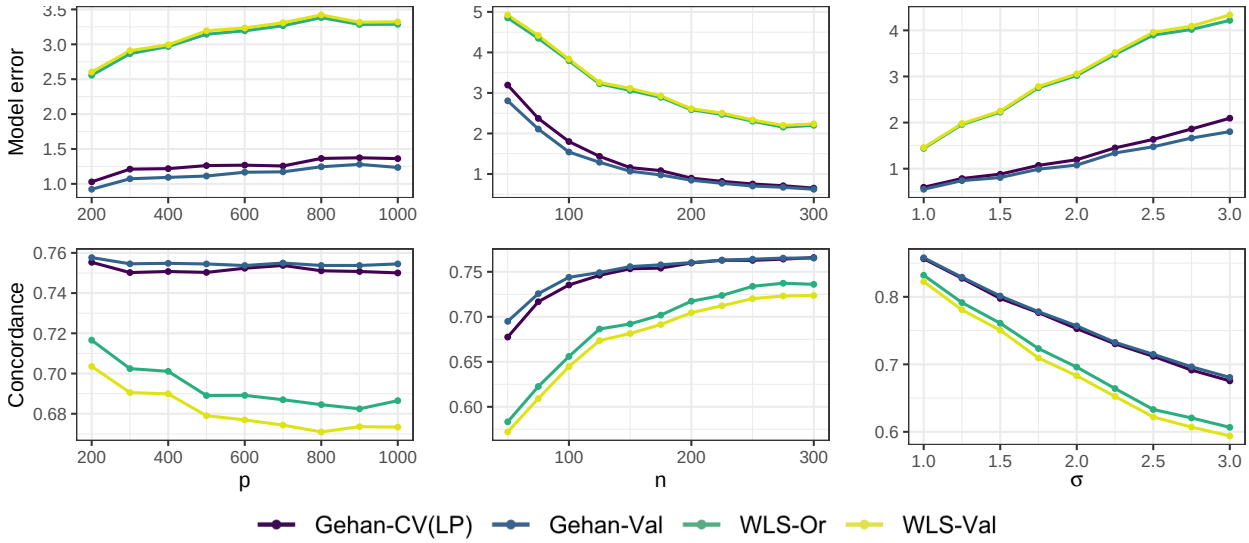

Figure 10: Model error (top row) and concordance (bottom row) for the four considered methods averaged over 100 independent replications with normal errors, β_* having ten elements set equal to 0.5 (five in two groups of size ten), and g being the sparse group lasso penalty with $\alpha = 0$.

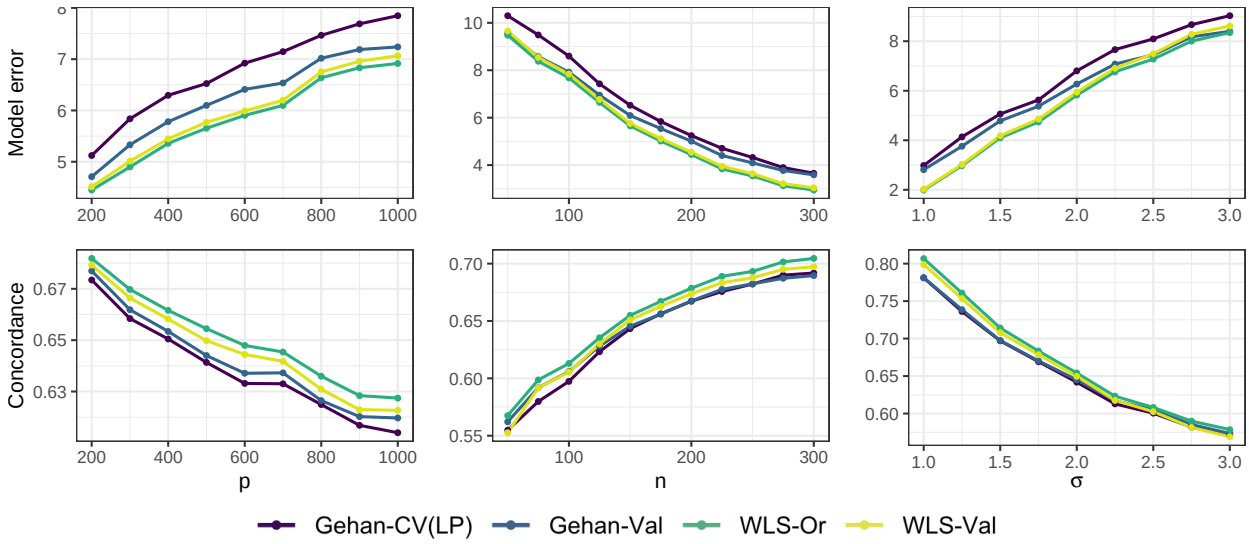

Figure 11: Model error (top row) and concordance (bottom row) for the four considered methods averaged over 100 independent replications with logistic errors, no censoring, β_* having ten elements set equal to one, and g being the elastic net penalty with $\alpha = 0.5$.

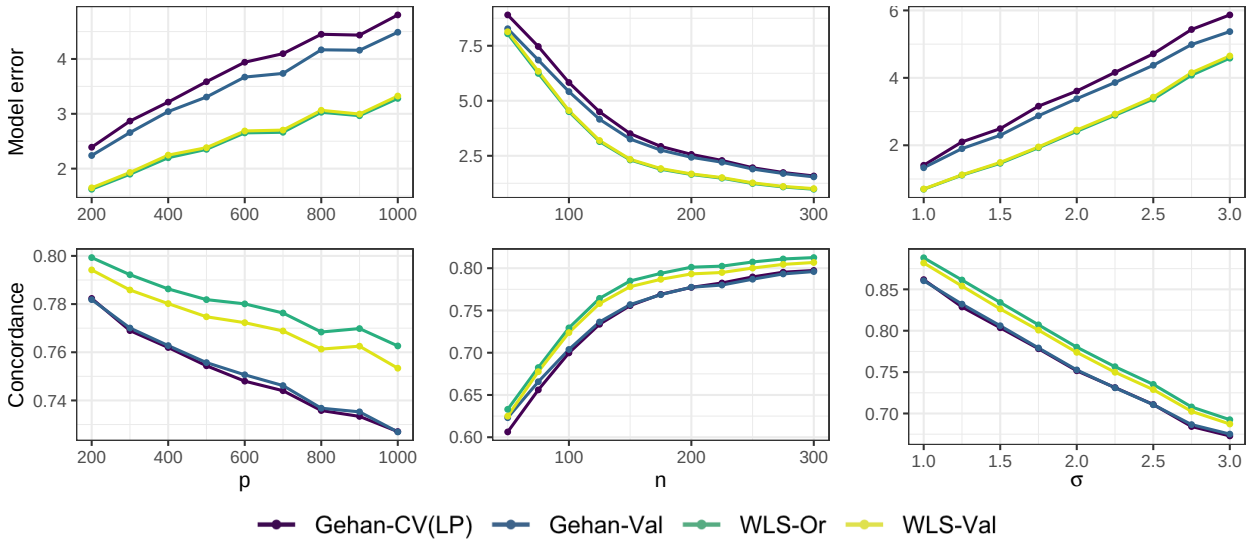

Figure 12: Model error (top row) and concordance (bottom row) for the four considered methods averaged over 100 independent replications with normal errors, no censoring, β_* having ten elements set equal to one, and g being the elastic net penalty with $\alpha = 0.5$.

4 Comparison to `hqreg`

As mentioned in the main manuscript, the R package `hqreg` can be used to compute $\arg\min_{\beta \in \mathbb{R}^p} h_M(\beta) + \lambda g(\beta)$ when g is the elastic net penalty. In the settings from Figure 2

of the main manuscript, `hqreg` runs nearly as fast as `penAFT`, and is nearly as accurate (in terms of objective function value at the returned estimate) when carefully implemented using the function `hqreg_raw`, rescaling the inputs, and tinkering with the convergence tolerances.

	Convergence tolerances				
	η_1	η_2	η_3	η_4	η_5
$M = n^2 10^2$	1.219	1.462	5.083	20.117	61.026
$M = n^2 10^4$	1.331	1.616	5.320	20.793	63.454
$M = n^2 10^6$	1.238	1.515	5.216	20.662	63.001

Table 3: Average solution path computing times for `hqreg` relative to `penAFT`. For example, a relative computing time of 5, indicates that `hqreg` took five times as long to compute than did `penAFT`. Various values of η at the different choices of M are described in the text. For reference, the average computing time of `penAFT` was 3.29 seconds.

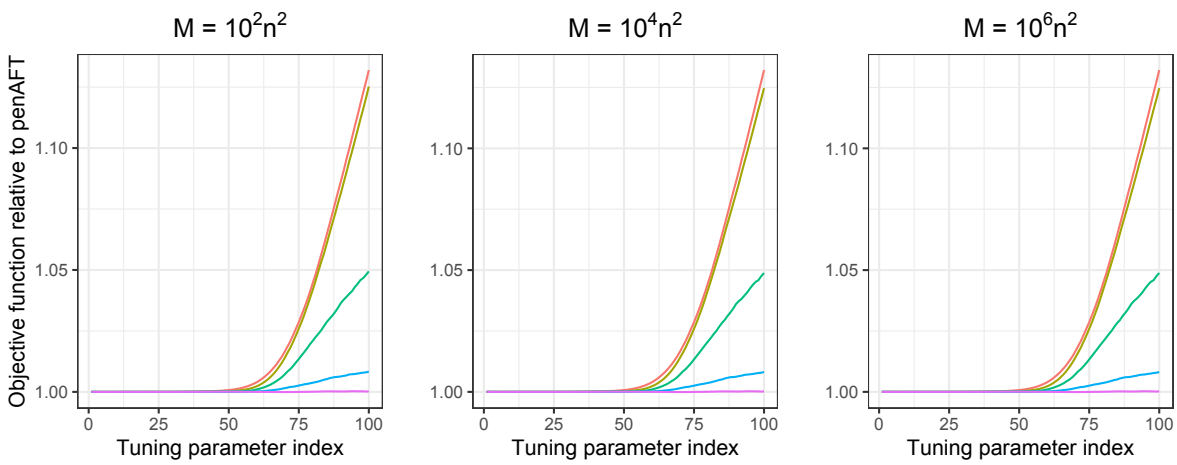

Figure 13: Average objective function values at termation for `hqreg` relative to `penAFT`. For example, a relative objective function value of 1.1 indicates that the objective function value at the `hqreg` solution was 10% larger than that at the `penAFT` solution. Red lines denote convergence tolerance η_1, gold denotes η_2, green denotes η_3, blue denotes η_4, and pink denotes η_5.

However, the dimensions considered in Figure 2 were chosen only to allow for computational feasibility of the two competitors. Here, we compare `penAFT` to `hqreg` in slightly higher dimensional settings. Specifically, we generate data from the same model as described in Section 5.1 with $n = 150$ and $p = 1000$. We randomly select 20 regression coefficients to be equal to one and set all others equal to zero.

To implement `hqreg`, we must choose M, the large constant defining h_M, and set the adjust the convergence tolerance. We found that for particular choices of M, the default convergence tolerance in `hqreg` either gave solutions too inaccurate, or took too long to compute. Here, we consider $n^2 10^2$, $n^2 10^4$ and $n^2 10^6$ as candidate values for M. For these settings we tried convergence tolerances $\eta \in \{\eta_1, \eta_2, \eta_3, \eta_4, \eta_5\}$ where we set set $\eta = \{10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}\}$ when $M = n^2 10^2$, set $\eta = \{10^{-7}, 10^{-8}, 10^{-9}, 10^{-10}, 10^{-11}\}$ when $M = n^2 10^4$, and set $\eta = \{10^{-8}, 10^{-9}, 10^{-10}, 10^{-11}, 10^{-12}\}$ when $M = n^2 10^6$.

In Table 3, we display the average computing times (over 100 independently generated datasets) for the entire solution path of 100 tuning parameter values (with $\kappa = 0.1$, the default) for `hqreg` relative to `penAFT` (i.e., a value of 1.1 means that on average, `hqreg` took 10% longer than `penAFT`). We see that for η_1, η_2, and η_3, computing times are similar to those when using `penAFT`. However, for η_4 and η_5, computing times are much longer than those for `penAFT`. In Figure 13, we display the average ratio of the objective function values at convergence for `hqreg` relative to that of `penAFT` at each tuning parameter value. As we can see, for large values of the tuning parameter, `hqreg` is reasonably accurate relative to `penAFT`. However, for smaller values of the tuning parameter, even using the computational time-consuming η_5 as a convergence tolerance does not always yield solutions as accurate as `penAFT`. Note further that `hqreg` employs many additional tricks such as the adaptive strong rule (which we could use in future versions of our package); whereas `penAFT` incurs additional computing times to determine the candidate tuning parameter set internally. To conclude, between the challenge of setting up the data correctly, choosing a particular M, and adjusting the convergence tolerance to be sufficiently strict, computing $\arg\min_{\beta \in \mathbb{R}^p} h_M(\beta) + \lambda g(\beta)$ with `hqreg` is doable but tedious, and is not likely to provide a solution as accurate as `penAFT` in a comparable amount of time.

We point out that this is not meant to be a criticism of `hqreg`: indeed, when the package is used on the problems for which it was intended, it performs quite well. However, when dealing with the special structure of h_M – specifically the input involving M – this seems problematic. Performing cross-validation may complicate issues even further: for example does M need be adjusted to account for the size of each fold? Fortunately, `penAFT` does not require a resolution to such concerns.