Improved computational tool for OHDSI: Bayesian penalized regression

Separating known risk factors among the large number of potential confounders

Aki Nishimura[1], Yuxi Tian[1], and Marc A. Suchard[1,2,3]

[1]Department of Biomathematics, David Geffen School of Medicine at UCLA, Los Angeles, CA, USA

[2]Department of Biostatistics, UCLA Fielding School of Public Health, Los Angeles, CA, USA; [3]Department of Human Genetics, David Geffen School of Medicine at UCLA, Los Angeles, CA, USA

Background

Large-Scale Observational Studies Using the OHDSI Database:

- Number of subjects $n \approx 10^5 \sim 10^6$ and of potential confounders $p \approx 10^4 \sim 10^5$.
- Comparison of two alternative treatments are based on *propensity score* methods.

Current Analytic Tool and Promising Bayesian Alternative

- Propensity scores are computed by regressiong on all potential confounders.
- Large p makes it essential to select a subset of predictors via *penalized regression*.
- Current analytic tool relies on the widely-used *Lasso* based on ℓ^1 penalty.
- Computational bottleneck of Lasso is the calibration of sparsity level via cross-validation, limiting our ability to flexibly model relative predictor importances.
- *Bayesian* formulation of penalized regression can incorporate such additional flexibilities with litte additional computational costs.

Regression with Multiple Penalty Parameters

Lasso in its standard form

- Lasso estimates the regression coefficients $\boldsymbol{\beta}$ by minimizing a loss function

$$-\log L(\boldsymbol{y} \,|\, \boldsymbol{X}, \boldsymbol{\beta}) + \tau^{-1} \|\boldsymbol{\beta}\|_1 \tag{1}$$

where $L(\boldsymbol{y} \,|\, \boldsymbol{X}, \boldsymbol{\beta})$ is the likelihood of outcome $\boldsymbol{y}$ and τ^{-1} is a penalty parameter.

- Performance of Lasso critically depends on the tuning parameter τ^{-1}, whose calibration typically relies on computationally expensive cross-validation.

Independent penalties on known and unknown risk factors

- Automated literature screening may be used to identify known risk factors.
- Lasso, however, penalizes all the potential confounders equally, as if known risk factors are no more important than the rest.
- Use of separate penalty parameters τ_{risk}^{-1} and τ_{other}^{-1} is more realistic:

$$-\log L(\boldsymbol{y} \,|\, \boldsymbol{X}, \boldsymbol{\beta}) + \tau_{\text{risk}}^{-1} \|\boldsymbol{\beta}_{\text{risk}}\|_1 + \tau_{\text{other}}^{-1} \|\boldsymbol{\beta}_{\text{other}}\|_1 \tag{2}$$

but is computationally prohibitive for calibration via cross-validation.

Bayesian Penalized (Shrinkage) Regression

Bayesian Lasso

- The use of ℓ^1 penalty in Lasso can be interpreted as a Bayesian procedure of "shrinking" the regression coefficient estimates by placing a prior distribution

$$\pi_{\text{prior}}(\boldsymbol{\beta}) = \prod_{i=1}^{p} \tau^{-1} \exp\left(-\tau^{-1}|\beta_i|\right) = \tau^{-p} \exp\left(-\tau^{-1}\|\boldsymbol{\beta}\|_1\right) \tag{3}$$

- Under the Bayesian paradigm, our knowledge on the regression coefficients are summarized in the *posterior* distribution

$$\pi_{\text{post}}(\boldsymbol{\beta} \,|\, \boldsymbol{y}) \propto L(\boldsymbol{y} \,|\, \boldsymbol{X}, \boldsymbol{\beta}) \pi_{\text{prior}}(\boldsymbol{\beta}) \tag{4}$$

whose mode coincide with the minimizer of (1).

- The penalty parameter τ^{-1} is similarly estimated from the posterior $\pi_{\text{post}}(\boldsymbol{\beta}, \tau \,|\, \boldsymbol{y})$.

Bayesian Lasso Computation

Characterizing the posterior distribution

- Regression coefficients are estimated by *Monte Carlo* simulation from a posterior.

Markov chain Monte Carlo algorithm for Bayesian Lasso

- Bayesian Lasso uses a Gibbs sampling algorithm based on a data augmentation scheme, introducing additional parameters $(\boldsymbol{\lambda}, \boldsymbol{\omega})$ in additions to $(\boldsymbol{\beta}, \tau)$.
- Gibbs sampler sequentially updates one of the parameters $(\boldsymbol{\beta}, \tau, \boldsymbol{\lambda}, \boldsymbol{\omega})$, and repeats this process for $100 \sim 1{,}000$'s of iterations.
- The computational bottleneck is the update of $\boldsymbol{\beta}$, that requires generating a random variable from the following high-dimensional Gaussian distribution:

$$\boldsymbol{\beta} \,|\, \tau, \boldsymbol{\lambda}, \boldsymbol{\omega} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \quad \text{where } \boldsymbol{\Sigma}^{-1} = \boldsymbol{X}^T \boldsymbol{\Omega} \boldsymbol{X} + \tau^{-2} \boldsymbol{\Lambda}^{-2} \text{ and } \boldsymbol{\mu} = \boldsymbol{\Sigma} \boldsymbol{X}^T (\boldsymbol{y} - 0.5) \tag{5}$$

with $\boldsymbol{\Omega} = \text{diag}(\boldsymbol{\omega})$ and $\boldsymbol{\Lambda} = \text{diag}(\boldsymbol{\lambda})$.

Speeding up Bayesian Lasso via Advanced Linear Algebra Techniques

Recasting a random variable as the solution of a linear system

- Random variable (5) can be generated as the solution of a linear system

$$\boldsymbol{\Phi}\boldsymbol{\beta} = \boldsymbol{v} \quad \text{for} \quad \boldsymbol{v} \sim \mathcal{N}\left(\boldsymbol{X}^{\intercal}(\boldsymbol{y} - 0.5), \boldsymbol{\Phi}\right) \text{ and } \boldsymbol{\Phi} = \boldsymbol{\Sigma}^{-1} \tag{6}$$

where $\boldsymbol{v}$ can be generated by sampling $\boldsymbol{\eta} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_n)$ and $\boldsymbol{\delta} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_p)$ independently and then setting

$$\boldsymbol{v} = \boldsymbol{X}^{\intercal} \boldsymbol{\Omega}^{1/2} \boldsymbol{\eta} + \tau^{-1} \boldsymbol{\Lambda}^{-1} \boldsymbol{\delta} + \boldsymbol{X}^{\intercal}(\boldsymbol{y} - 0.5) \tag{7}$$

Fast solution of (6) via conjugate gradient method

- *Conjugate gradient* (CG) algorithm allows us to solve the linear system (6) purely through the matrix-vector operation $\boldsymbol{w} \to \boldsymbol{\Phi}\boldsymbol{w}$, **without ever explicitly forming $\boldsymbol{\Phi}$**:
  - The vector $\boldsymbol{\Phi}\boldsymbol{w}$ can be computed through operations $\boldsymbol{w} \to \boldsymbol{X}\boldsymbol{w}$ and $\boldsymbol{u} \to \boldsymbol{X}^{\intercal}\boldsymbol{u}$ in addition to element-wise multiplications by $\tau, \boldsymbol{\lambda}$, and $\boldsymbol{\omega}$.
  - This is critical because computing $\boldsymbol{\Phi} = \boldsymbol{\Sigma}^{-1}$ through the formula (5) is very expensive — $O(n^2 p)$ operations when $\boldsymbol{X}$ is dense.
  - When $\boldsymbol{X}$ is sparse, CG automatically takes advantage of it as the algorithm only requires (sparse) matrix multiplications by $\boldsymbol{X}$ and $\boldsymbol{X}^{\intercal}$; no additional memory use for explicitly forming $\boldsymbol{\Phi}$.
- In case of Bayesian penalized regression, CG converges rapidly — often within a few hundred iterations — through an effective *preconditioning* strategy.
- Compared to the standard sampling method for (5), the proposed approach has big advantages both in terms of memory usage and number of arithmetic operations.

Results

We prototyped the proposed algorithm in Python and applied it to the replication of the warfarin vs dabigatran study of Graham et al. ($n = 72{,}489$ and $p = 22{,}175$).

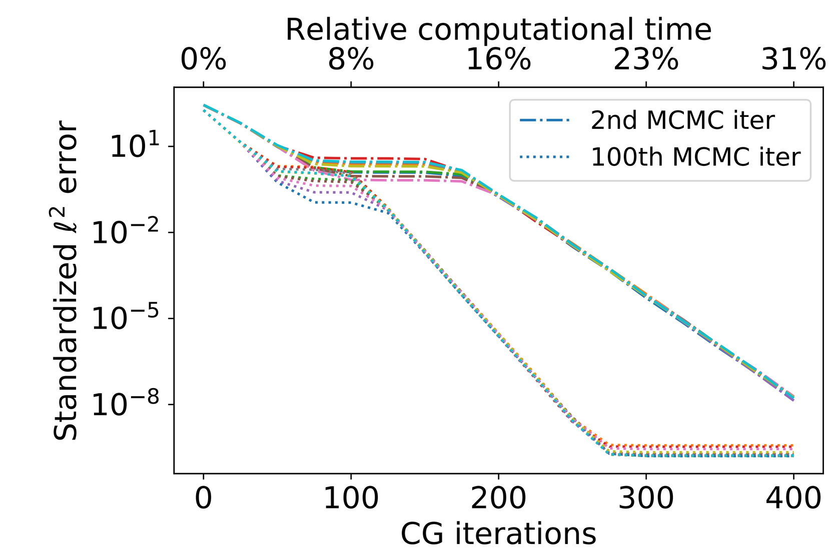

FIGURE 1 : Distance between the exact solution of (6) and iterative solutions; it is shown both as a function of the number of matrix-vector operations and that of computational time relative to the direct sampling method for (5). The error is computed as $\sum_i (\beta_{\text{exact},i} - \beta_{\text{cg},i})^2 / \hat{\nu}_i^2$ where $\hat{\nu}_i$ is an estimate of $\mathbb{E}_{\pi(\beta|\boldsymbol{y})}[\beta_i^2]$. Different colors indicate different draws of a random target vector $\boldsymbol{v}$ in (6). Dashed and dotted lines correspond to the values of $(\tau, \boldsymbol{\lambda}, \boldsymbol{\omega})$ and $\boldsymbol{\Phi}$ from different MCMC iterations.

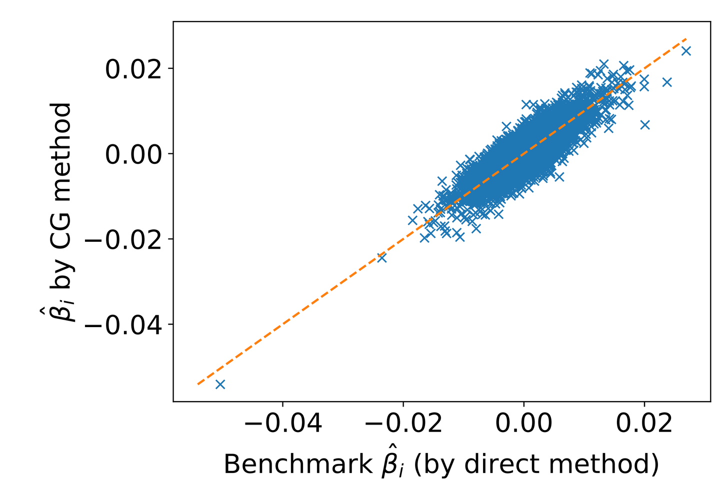

FIGURE 2 : Comparison of the regression coefficient estimates (posterior mean) from MCMC based on the direct and CG methods for sampling from (5).

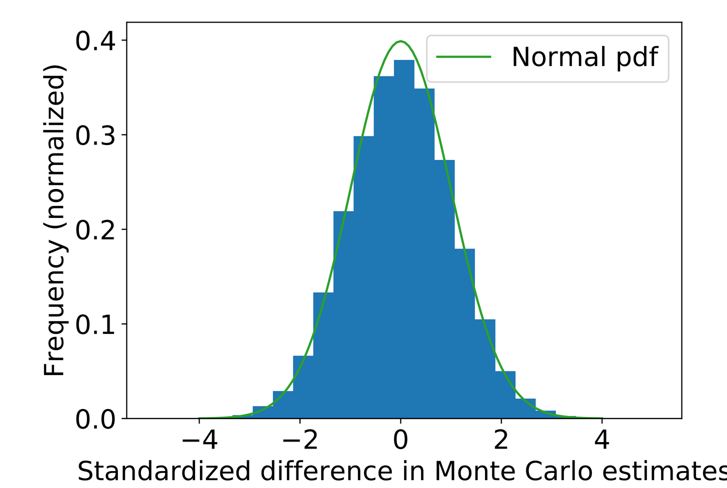

FIGURE 3 : Histogram for $(\hat{\beta}_{\text{exact},i} - \hat{\beta}_{\text{cg},i})/\hat{\sigma}_i$, where $\hat{\sigma}_i^2$ is an estimate of the Monte Carlo variance of $\hat{\beta}_{\text{exact},i} - \hat{\beta}_{\text{cg},i}$. Normality of the histogram indicates that the two MCMC outputs are indistinguishable.

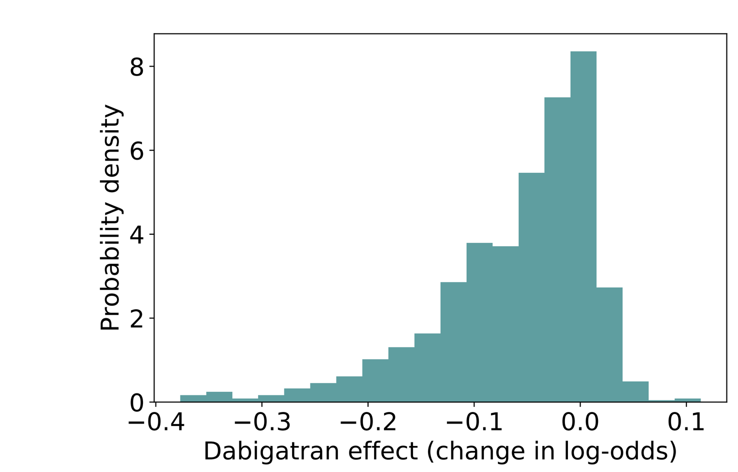

FIGURE 4 : Posterior distribution of the treatment effect — the change in the log-odds of brain hemorrhage for those treated with dabigatran instead of warfarin.

Conclusion

- The new algorithm is **over 7 times faster** than the traditional method when applied to a typical OHDSI dataset. Even in case of a single penalty parameter, this speed-up makes Bayesian Lasso competitive with the standard Lasso.
- Bayesian penalized regression easily accommodates literature-informed priors as well as further extensions. Once integrated into the OHDSI toolkit, it is expected to substantially increase its modeling and predictive capability.

References

[1] Polson NG, Scott JG, and Windle J (2014). The Bayesian bridge. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(4):713–733.

[2] Yuxi Tian M. JS and Suchard MA (2017). Synthetic and negative control evaluation framework for large-scale propensity score survival analysis. *Preprint*.

[3] Graham DJ, Reichman ME, Wernecke M, Zhang R, Southworth MR, Levenson M, et al. (2015). Cardiovascular, bleeding, and mortality risks in elderly Medicare patients treated with dabigatran or warfarin for non-valvular atrial fibrillation. *Circulation*, 131:157–164.