Scalable Bayesian sparse regression for OHDSI studies: Prior-preconditioned conjugate gradient sampler and `bayesbridge(r)` package

Akihiko Nishimura[a], Marc A. Suchard[b]

[a]Department of Biostatistics, Johns Hopkins Bloomberg School of Public Health

[b]Department of Biostatistics, University of California, Los Angeles

Background

The data heterogeneity across institutions has traditionally forced researchers to train predictive and causal models separately for each database, with no transfer of information across these models. Unsurprisingly, these statistical models exhibit worse performance on smaller databases with fewer training data [1]. This in particular has prevented researchers from asking clinical questions for which no single data source has sufficient information, such as how treatment effects may vary across patient subgroups. The OHDSI's network of OMOP-standardized databases provides an opportunity to borrow information across the heterogeneous data sources using Bayesian hierarchical analysis. The large-scale Bayesian regression model required to realize this opportunity, however, has traditionally been prohibitively expensive in terms of its computational burden. We develop a novel *prior-preconditioned conjugate gradient sampler* and accompanying Python/R packages `bayesbridge/bayesbridger` that overcomes this computational bottleneck, thereby paving the way for the hierarchical analysis of the OHDSI databases.

Methods

OHDSI has developed a causal inference framework that enables generating clinical evidences at scale, in a fully reproducible and data-driven manner [2]. A key component in this framework is the estimation of propensity scores for a large number of patients from tens of thousands of clinical covariates [3]. For this large-scale regression task, OHDSI's `ChortMethod` package relies on the L^1 regularized regression technique and its computationally efficient implementation via `Cyclops`. This approach works well, however, only for larger health databases that provide sufficiently large study cohorts consisting of, say, at least 10,000 patients who meet the inclusion criteria; otherwise, the estimated propensity score may fail to properly account for all the potential sources of confounding [4].

Bayesian hierarchical modeling is a natural solution to the issue of smaller databases. The Bayesian framework provides a principled way to borrow information across multiple databases while acknowledging their heterogeneity and to thereby improve propensity score and treatment effect estimates within each database. In particular, the L^1 regularized regression has a Bayesian analogue — Bayesian sparse regression based on a shrinkage prior — that allows for hierarchical extensions to the multi-database setting. Carrying out the posterior computation under Bayesian sparse regression models, however, has traditionally proved far more computationally demanding than its frequentist regularized regression conterpart. This has previously made the Bayesian approach computationally prohibitive at the scale of modern observational health databases.

To address this computational bottleneck, we develop the *prior-preconditioned conjugate gradient sampler* to accelerate the posterior computation under Bayesian sparse regression models [5]. Combined with the collapsed Gibbs update technique under the Bayesian bridge prior of Polson [6], the accelerated sampler makes it possible to deploy Bayesian sparse regression at the scale of OHDSI studies. We implement the scalable Bayesian sparse regression model in the Python package `bayesbridge` (https://github.com/OHDSI/bayes-bridge) and its R wrapper `bayesbridger` (https://github.com/OHDSI/BayesBridgeR) as part of HADES.

Results

The main classes in `bayesbridge` consist of `RegressionModel`, `RegressionCoefPrior`, and `BayesBridge`. For basic use, a user simply defines a regression model with a binary outcome y (numpy vector) and design matrix X (numpy or scipy sparse matrix) and, optionally, specify details on the bridge prior:

```
from bayesbridge import RegressionModel, RegressionCoefPrior, BayesBridge

model = RegressionModel(y, X, family='logit')
prior = RegressionCoefPrior(bridge_exponent=.25)
```

The `bridge_exponent` controls the degree of shrinkage induced by the prior, with 1/4 being a reasonable default value [7]. Having specified the model and prior, a user can sample from the posterior distribution via the Gibbs sampler:

```
bridge = BayesBridge(model, prior)
samples, mcmc_info = bridge.gibbs(
    n_burnin=100, n_post_burnin=1000, thin=1,
    coef_sampler_type='cg'
)
coef_samples = samples['coef']
```

We benchmark the accelerated Gibbs sampler (`coef_sampler_type='cg'`) and traditional one (`coef_sampler_type='cholesky'`) on the OHDSI replication of the FDA study comparing the effectivenesses and safety profiles of anti-coagulants dabigatran and warfarin for treating atrial fibrillation [8]. The data set consists of $n = 72{,}489$ patients, 27.3% of whom are on dabigatran, and $p = 22{,}175$ clinical covariates. Using an incident of gastrointestinal bleeding within a fixed follow-up window as the binary outcome of interest, we fit two large-scale Bayesian sparse logistic regression models: the propensity score model and the propensity-stratified outcome model with treatment-covariate interactions to capture potential subgroup effects. The posterior computation is carried out on 2015 iMac with an Intel Core i7 "Skylake" processor having four cores at 4GHz and 32GB of memory. With the traditional sampler, the computation requires 106 and 212 hours for 5,500 and 11,000 iterations of the Gibbs sampler for the propensity score and outcome model. The same computation requires 11.4 and 11.3 hours using the accelerated sampler, delivering **9.3** and **18.8**-fold speed-up.

In fact, a further performance boost is possible through the use of graphical processing unit (GPU), for which `bayesbridge` is optimized. A user can take advantage of this feature by supplying `cupy` (sparse) matrix to the `RegressionModel` class:

```
import cupy as cp
import cupyx as cpx

X = cp.asarray(X) # Or `cpx.scipy.sparse.csr_matrix(X)` if `X` is a scipy sparse matrix
model = RegressionModel(y, X, family='logit')
```

When combining the algorithmic and GPU accelerations, `bayesbridge` completes the posterior computation for the propensity score and outcome models in 0.62 and 0.61 hours, delivering **171** and **347**-fold speed-up.

The R package `bayesbridger` provides an R interface similar to the original Python package, using `reticulate` as its backend. A user first has to set a Python environment via the provided utility functions:

```
library(bayesbridger)
setup_python_env(
  envname = "bayesbridge",
  python_path = guess_anaconda_path()
)
configure_python(envname = "bayesbridge")
```

The user can then fit a large-scale Bayesian sparse regression model in the same manner as before:

```
model <- create_model(y, X) # `X` is either a 2-dimensional array or sparseMatrix
prior <- create_prior(bridge_exponent=.25)

bridge <- instantiate_bayesbridge(model, prior)
gibbs_output <- gibbs(
  bridge, n_burnin = 100L, n_iter = 1100L, thin = 1L,
  coef_sampler_type = "cg"
)
mcmc_samples <- gibbs_output$samples
```

Conclusion

The accelerated sampler and its optimized implementation provided by `bayesbridge(r)` address the critical computational bottleneck that has previously prevented us from leveraging the Bayesian machinery for OHDSI studies. The `bayesbridge(r)` packages are under active development to allow for the hierarchical extension of Bayesian sparse regression, which in particular accounts for the heterogeneity in data encoding by incorporating ontological relations between clinical covariates [9]. This tool will enable OHDSI to fully exploit all its constituent databases and thus amplify its ability to generate clinically impactful insight from observational health data.

References

[1] J. M. Reps, M. J. Schuemie, M. A. Suchard, P. B. Ryan, and P. R. Rijnbeek, "Design and implementation of a standardized framework to generate and evaluate patient-level prediction models using observational healthcare data," *Journal of the American Medical Informatics Association*, vol. 25, no. 8, pp. 969–975, 2018.

[2] M. J. Schuemie *et al.*, "Large-scale evidence generation and evaluation across a network of databases (LEGEND): Assessing validity using hypertension as a case study," *Journal of the American Medical Informatics Association*, vol. 27, no. 8, pp. 1268–1277, 2020.

[3] Y. Tian, M. J. Schuemie, and M. A. Suchard, "Evaluating large-scale propensity score performance through real-world and synthetic data experiments," *International Journal of Epidemiology*, vol. 47, no. 6, pp. 2005–2014, 2018.

[4] A. Nishimura *et al.*, "Alpha-1 blockers and susceptibility to COVID-19 in benign prostate hyperplasia patients: An international cohort study," *medRxiv*, 2021.

[5] A. Nishimura and M. A. Suchard, "Prior-preconditioned conjugate gradient for accelerated Gibbs sampling in 'large n & large p' sparse Bayesian regression," *Journal of the American Statistical Association*, 2022.

[6] N. G. Polson, J. G. Scott, and J. Windle, "The Bayesian bridge," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 76, no. 4, pp. 713–733, 2014.

[7] A. Nishimura and M. A. Suchard, "Shrinkage with shrunken shoulders: Gibbs sampling shrinkage model posteriors with guaranteed convergence rates," *Bayesian Analysis*, 2022.

[8] D. J. Graham *et al.*, "Cardiovascular, bleeding, and mortality risks in elderly Medicare patients treated with dabigatran or warfarin for non-valvular atrial fibrillation," *Circulation*, vol. 131, pp. 157–164, 2015.

[9] X. Ding, P. Nagy, C. G. Chute, and A. Nishimura, "Bayesian analytics of multi-institutional health data via ontologically-informed hierarchical model." 2022+.