



Hamiltonian Monte Carlo Inference of Marginalized Linear Mixed-Effects Models

Jinlin Lai (presenter), Daniel Sheldon, Justin Domke

Outline

Background

Methods

Experiments

Future Works

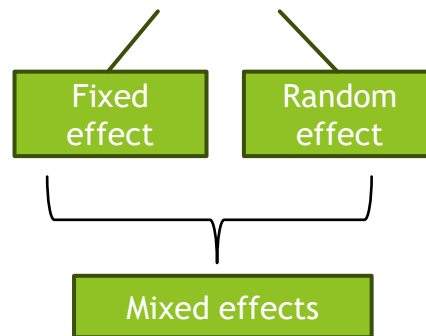
BACKGROUND

- Linear Mixed-Effects Models (LMMs)
- Bayesian Inference of LMMs

Linear Mixed-Effects Models

Linear mixed-effects models (LMMs) break regression coefficients into **fixed** and **random** effects:

$$y_n = x_n^T(\beta + u_{g_n}) + \epsilon, \epsilon \sim \text{normal}(0, \sigma)$$



Fixed effects are global, sharing among different data points.

Random effects are group specific, for M groups, there are M random effects $u_1, \dots, u_M$.

General Linear Mixed-Effects Models

In practice, a data point may belong to multiple orthogonal groups (gender, ethnicity, region, etc., which we call **classes**), and an effect may only multiply with some of the regressors. A more general LMM is

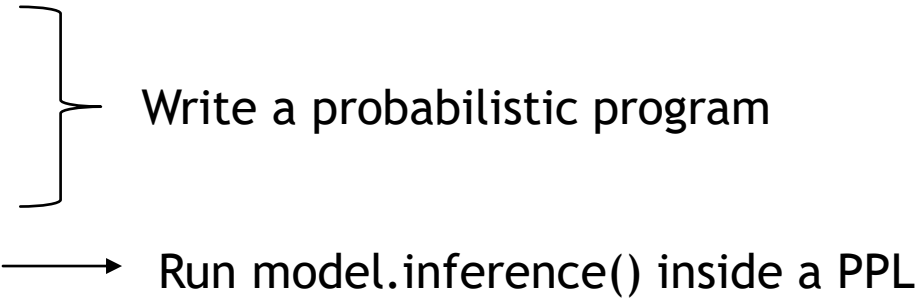
$$y_n = x_n^T \beta + z_{1,n}^T u_{1,g_{1,n}} + z_{2,n}^T u_{2,g_{2,n}} + \cdots + z_{L,n}^T u_{L,g_{L,n}} + \epsilon, \epsilon \sim \text{normal}(0, \sigma)$$

For data n , it belongs to $g_{1,n}, \dots, g_{L,n}$, whose corresponding random effects multiply with $z_{1,n}, \dots, z_{L,n}$.

The inference of LMMs requires estimation of hyperparameters, β, u, σ , which may further contain hierarchical structures.

Bayesian Inference of LMMs

For the parameters β, u, σ , practitioners would

- Assign priors on them
 - Form a probabilistic model
 - Run a Bayesian inference algorithm
 - Analyze with the obtained parameters
- 
- The diagram illustrates the workflow of Bayesian inference for Linear Mixed Models (LMMs). It shows a list of four steps on the left. A right-facing curly bracket groups the first two steps, 'Assign priors on them' and 'Form a probabilistic model', and points to the text 'Write a probabilistic program'. A horizontal arrow points from the third step, 'Run a Bayesian inference algorithm', to the text 'Run model.inference() inside a PPL'. The fourth step, 'Analyze with the obtained parameters', is not connected to any other text.
- Write a probabilistic program
- Run model.inference() inside a PPL

There are structures in the model. How can we make naïve inference faster?

METHODS

- Vectorization
- Marginalization

Marginalization

For the canonical form,

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}_1\mathbf{u}_1 + \mathbf{Z}_2\mathbf{u}_2 + \cdots + \mathbf{Z}_L\mathbf{u}_L + \boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim \text{normal}(\mathbf{0}, \boldsymbol{\Sigma}_y)$$

If we just look at the first class, usually we assign

$$\mathbf{u}_1 \sim \text{normal}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{u_1})$$

Then we can rewrite the canonical form as

$$\mathbf{y} = \mathbf{Z}_1\mathbf{u}_1 + \mathbf{b} + \boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim \text{normal}(\mathbf{0}, \boldsymbol{\Sigma}_y)$$

It is possible to exactly integrate $\mathbf{u}_1$ out from the model:

$$p(\mathbf{y}|\mathbf{u}_1, \dots)p(\mathbf{u}_1|\dots) \rightarrow p(\mathbf{y}|\dots)p(\mathbf{u}_1|\mathbf{y}, \dots)$$

Marginalization

The marginalized likelihood $p(\mathbf{y} | \dots)$ becomes more complicated:

$$p(\mathbf{y} | \dots) = \text{normal}(\dots, \mathbf{Z}_1 \boldsymbol{\Sigma}_{\mathbf{u}_1} \mathbf{Z}_1^T + \boldsymbol{\Sigma}_{\mathbf{y}})$$

$\mathbf{E} = \mathbf{Z}_1 \boldsymbol{\Sigma}_{\mathbf{u}_1} \mathbf{Z}_1^T + \boldsymbol{\Sigma}_{\mathbf{y}}$ is a dense $N \times N$ matrix. Evaluating $\log p(\mathbf{y} | \dots)$ naively becomes $O(N^3)$,

- The determinant $|\mathbf{E}|$ is $O(N^3)$.
- The computation $\mathbf{v}^T \mathbf{E}^{-1} \mathbf{v}$ is $O(N^3)$.
- Also, it is $O(N^3)$ to sample from $p(\mathbf{u}_1 | \mathbf{y}, \dots)$.

We use (a) linear algebra tricks and (b) structure of LMMs to speed up the above evaluations.

Summary of results

We consider the problem of marginalizing one or all random effects. Both marginalization approaches in this work have linear complexity with respect to N .

Table 1: Time complexities of different HMC approaches for the submodel involved in marginalization. Initialization is done once before the HMC loop. The log density is computed within each step of the leapfrog integrator. Recovery is performed for each sample from HMC. N is the number of observations, M is the dimension for one class of random effects, D is the dimension for all classes of random effects, L is the number of classes, d is the dimension for an effect of a group in a class.

Submodel	Approach	Initialization	Log density	Recovery
$p(\mathbf{u}_i, \mathbf{y} \boldsymbol{\Theta}, \mathbf{u}_{-i})$	HMC	-	$\mathcal{O}(Md^2 + NLd)$	-
	Naive marginalization	-	$\mathcal{O}(M^3 + N^3)$	$\mathcal{O}(M^3 + N^3)$
	Marginalize with lemmas	-	$\mathcal{O}(Md^2 + NLd + Nd^2)$	$\mathcal{O}(Md^2 + NLd + Nd^2)$
$p(\mathbf{v}, \mathbf{y} \boldsymbol{\Theta})$	HMC	-	$\mathcal{O}(Dd^2 + NLd)$	-
	Naive marginalization	-	$\mathcal{O}(D^3 + N^3)$	$\mathcal{O}(D^3 + N^3)$
	Marginalize with assumptions	$\mathcal{O}(D^3 + NL^2d^2)$	$\mathcal{O}(D^2 + NLd)$	$\mathcal{O}(D^2 + NLd)$

EXPERIMENTS

- Cross-effects Models
- Experiments in Cognitive Sciences

Cross-effects models - ETH instruction evaluation

Table 2: Running time in seconds for HMC, with or without marginalization. Mean and standard deviation over 5 independent runs are reported. Experiments are run on NVIDIA A40.

Method	No marginalization	Marginalize $\mathbf{u}_1$	Marginalize $\mathbf{u}_2$	Marginalize $\mathbf{u}_3$	Marginalize $\mathbf{u}$
Time (s)	13417 (98)	5004 (1468)	2607 (3)	3071 (4)	631 (12)

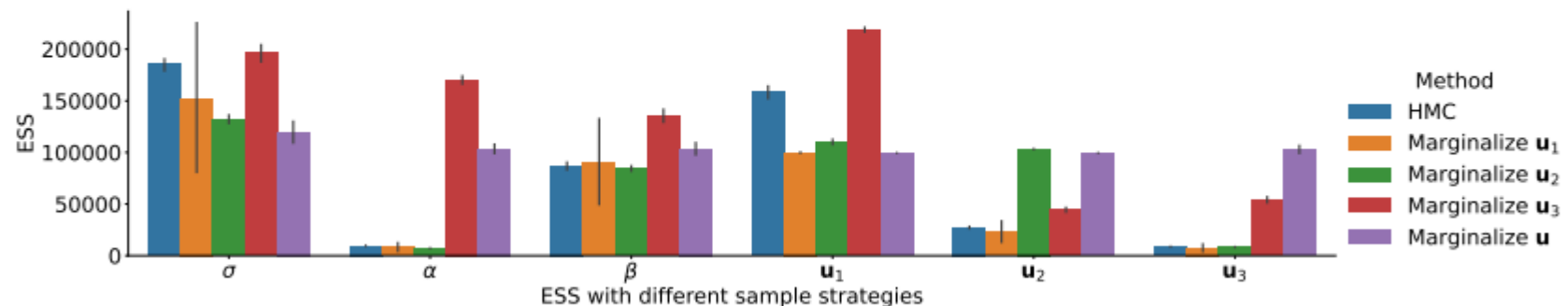


Figure 2: Average ESS for each variable on the instruction evaluation model with different HMC strategies. Numbers above the sample size 100,000 indicate effective sampling.

More Experiments in Cognitive Sciences

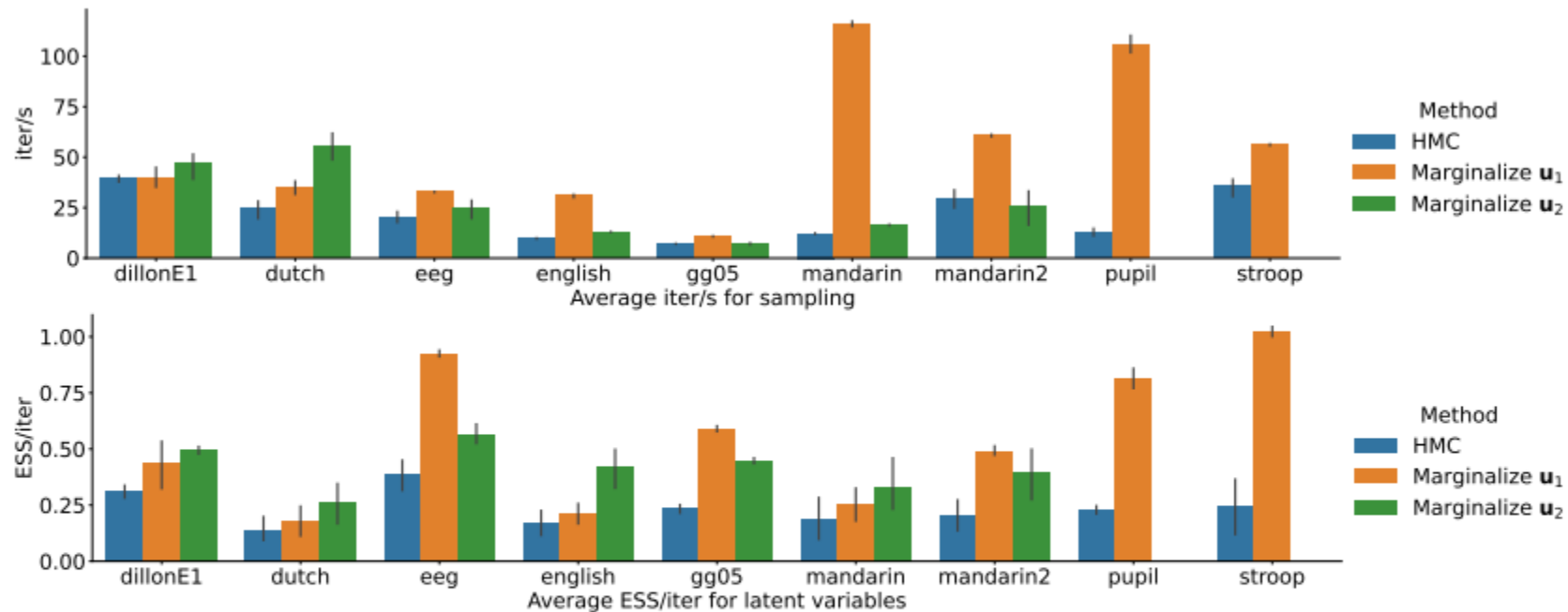
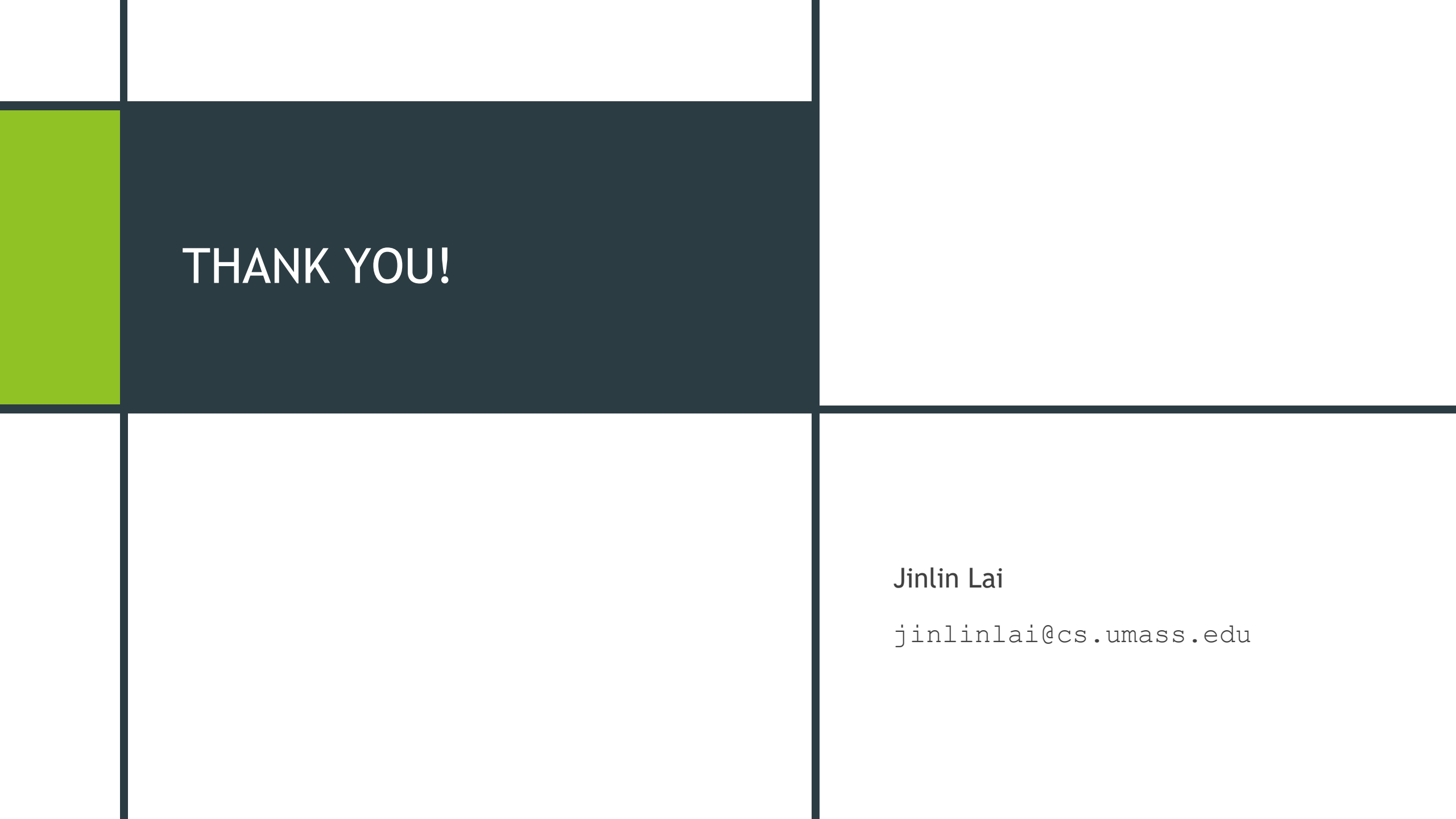


Figure 4: Experimental results for the 9 cognitive science datasets with and without marginalization. Each experiment is performed 5 times with different random seeds. Marginalization usually improves sampling speed measured by iterations per second (iter/s) and sample efficiency measured by ESS per iteration (ESS/iter).



THANK YOU!

Jinlin Lai

`jinlinlai@cs.umass.edu`