# Compressing Parameters in Bayesian Logistic Sequence Prediction Models

Longhai Li

Joint work with Radford M. Neal

`longhai@utstat.toronto.edu`

Department of Statistics

University of Toronto

Toronto, Ontario,CANADA

The 3rd Monte Carlo Workshop

*Harvard University, 13 May 2007*

## Difficulty When Using High-Order Interactions

In many situations, the response depends on the high-order interactions of predictors. However, it is difficult to consider such high-order interactions due to the many parameters needed, e.g. there are $3^k$ interactions for $k$ binary features:

- Computationally intensive
- May overfit the data

Bayesian methods will not overfit the data, but still have computational difficulty. With more parameters, a Markov chain sampler

- Takes longer to update one iteration
- May need more iterations to converge
- May get trapped more easily in a local mode
- Requires more memory

# Our Solution: Compressing Parameters

Groups of predictors have the same value for all training cases. The number of groups, $G$, is much smaller than the number of parameters when considering high-order interactions. The likelihood function therefore depends only on the group sums:

$$\begin{aligned} L^{\beta}(\beta_{11},\ldots,\beta_{1,n_1},\ \ldots\ ,\beta_{G1},\ldots,\beta_{G,n_G}) &= L\left(\sum_{k=1}^{n_1}\beta_{1k},\ldots,\sum_{k=1}^{n_G}\beta_{Gk}\right) \\ &= L(s_1,\ \ldots\ ,s_G) \end{aligned}$$

We use priors as $\beta_{gk} \sim N(0,\sigma_{gk}^2)$ or $\beta_{gk} \sim$ Cauchy$(0,\sigma_{gk})$, because the priors of $s_g$'s can be found easily:

$$s_g \sim N\left(0,\ \sum_{k=1}^{n_g}\sigma_{gk}^2\right) \quad \text{or} \quad s_g \sim \text{Cauchy}\left(0,\ \sum_{k=1}^{n_g}\sigma_{gk}\right)$$

The posterior of the $s_g$'s given the training data $\mathcal{D}$:

$$P(\boldsymbol{s} \mid \mathcal{D}) = \frac{1}{c(\mathcal{D})} L(s_1,\ \ldots\ ,s_G)\, P_1^s(s_1)\ \cdots\ P_g^s(s_G)$$

## Splitting the Compressed Parameters

After obtaining the samples of $s_g$'s using MCMC, we can recover the original parameters, using the splitting distribution. The following shows for group $g$:

$$P(\beta_{g1}, \ldots, \beta_{g,n_g-1} \mid s_g) = \prod_{k=1}^{n_g-1} P_{gk}(\beta_{gk})\; P_{g,n_g}\left(s_g - \sum_{k=1}^{n_g-1} \beta_{gk}\right) \,/\, P_g^s(s_g)$$

The above distribution is unrelated to $\mathcal{D}$. To justify this, map original $\beta_{gk}$'s to a set of new parameters in light of training data:

$$(\beta_{g1}, \ldots, \beta_{g,n_g-1}, \beta_{g,n_g}) \Longrightarrow (\beta_{g1}, \ldots, \beta_{g,n_g-1}, s_g = \sum_{k=1}^{n_g} \beta_{gk})$$

The posterior of the new parameters is:

$$P(\boldsymbol{s}, \boldsymbol{\beta} \mid \mathcal{D}) = \frac{1}{c(\mathcal{D})} L\,(s_1, \ldots, s_G) \prod_{g=1}^{G} \prod_{k=1}^{n_g-1} P_{gk}(\beta_{gk}) P_{g,n_g}\left(s_g - \sum_{k=1}^{n_g-1} \beta_{gk}\right)$$

## Predicting for a Test Case

Storing all $\beta_{gk}$'s requires a huge amount of space. If we split $s_g$'s temporarily only for one test case, we need only split $s_g$ into two parts.

After re-indexing $\beta$'s, the function for making prediction for a test case can be written as:

$$a\left(\sum_{k=1}^{t_1}\beta_{1k},\ \ldots\ ,\sum_{k=1}^{t_G}\beta_{Gk},\ \sum_{k=1}^{t_*}\beta_{*k}\right) = a(s_1^t,\ \ldots\ , s_G^t, s_*^t) = a(\boldsymbol{s}^t, s_*^t)$$

Only need to sample from $P(s_g^t \mid s_g)$:

$$P(s_g^t \mid s_g) = P_g^{(1)}(s_g^t)\ P_g^{(2)}(s_g - s_g^t)\ /\ P_g^s(s_g)$$

where $P_g^{(1)}$ and $P_g^{(2)}$ are the priors of $\sum_{k=1}^{t_g}\beta_{gk}$ and $\sum_{k=t_g+1}^{n_g}\beta_{gk}$.

In case of Gaussian priors used,

$$s_g^t \mid s_g\ \sim\ N\left(s_g\,\frac{(\sigma_g^{(1)})^2}{(\sigma_g^{(1)})^2+(\sigma_g^{(2)})^2}\,,\ (\sigma_g^{(1)})^2\left(1\ -\ \frac{(\sigma_g^{(1)})^2}{(\sigma_g^{(1)})^2+(\sigma_g^{(2)})^2}\right)\right)$$

## Splitting a Cauchy Variable

The density of the splitting distribution when using Cauchy priors:

$$P(s_g^t \mid s_g) = \frac{1}{C} \frac{1}{\left(\sigma_g^{(1)}\right)^2 + (s_g^t)^2} \frac{1}{\left(\sigma_g^{(2)}\right)^2 + (s_g^t - s_g)^2}$$

When $s_g \neq 0$ **or** $\sigma_g^{(1)} \neq \sigma_g^{(2)}$, the CDF is:

$$\begin{aligned} F(s_g^t\,;\, s_g, \sigma_g^{(1)}, \sigma_g^{(2)}) \quad = \quad & \frac{1}{C}\left[ r \log\left( \frac{(s_g^t)^2 + (\sigma_g^{(1)})^2}{(s_g^t - s_g)^2 + (\sigma_g^{(2)})^2} \right) + \right. \\ & p_0 \left( \arctan\left( \frac{s_g^t}{\sigma_g^{(1)}} \right) + \frac{\pi}{2} \right) + \\ & \left. p_s \left( \arctan\left( \frac{s_g^t - s_g}{\sigma_g^{(2)}} \right) + \frac{\pi}{2} \right) \right] \end{aligned}$$

We use inverse method to sample from above distribution, with inverse CDF found by some numerical method, such as Illinois method.

# Logistic Sequence Prediction Models (LSPM)

We want to model $P(x_{O+1} \mid x_1, \ldots, x_O)$, where $x_1, \ldots, x_O, x_{O+1}$ is a discrete sequence.

We use the linear logistic models with indicators for all interaction patterns:

$$P(x_{O+1} = k \mid \boldsymbol{x}_{1:O}, \boldsymbol{\beta}^{(1)}, \ldots, \boldsymbol{\beta}^{(K)}) = \frac{\exp(l\,(\boldsymbol{x}_{1:O}, \boldsymbol{\beta}^{(k)}))}{\sum_{j=1}^{K} \exp(l\,(\boldsymbol{x}_{1:O}, \boldsymbol{\beta}^{(j)}))}$$

where

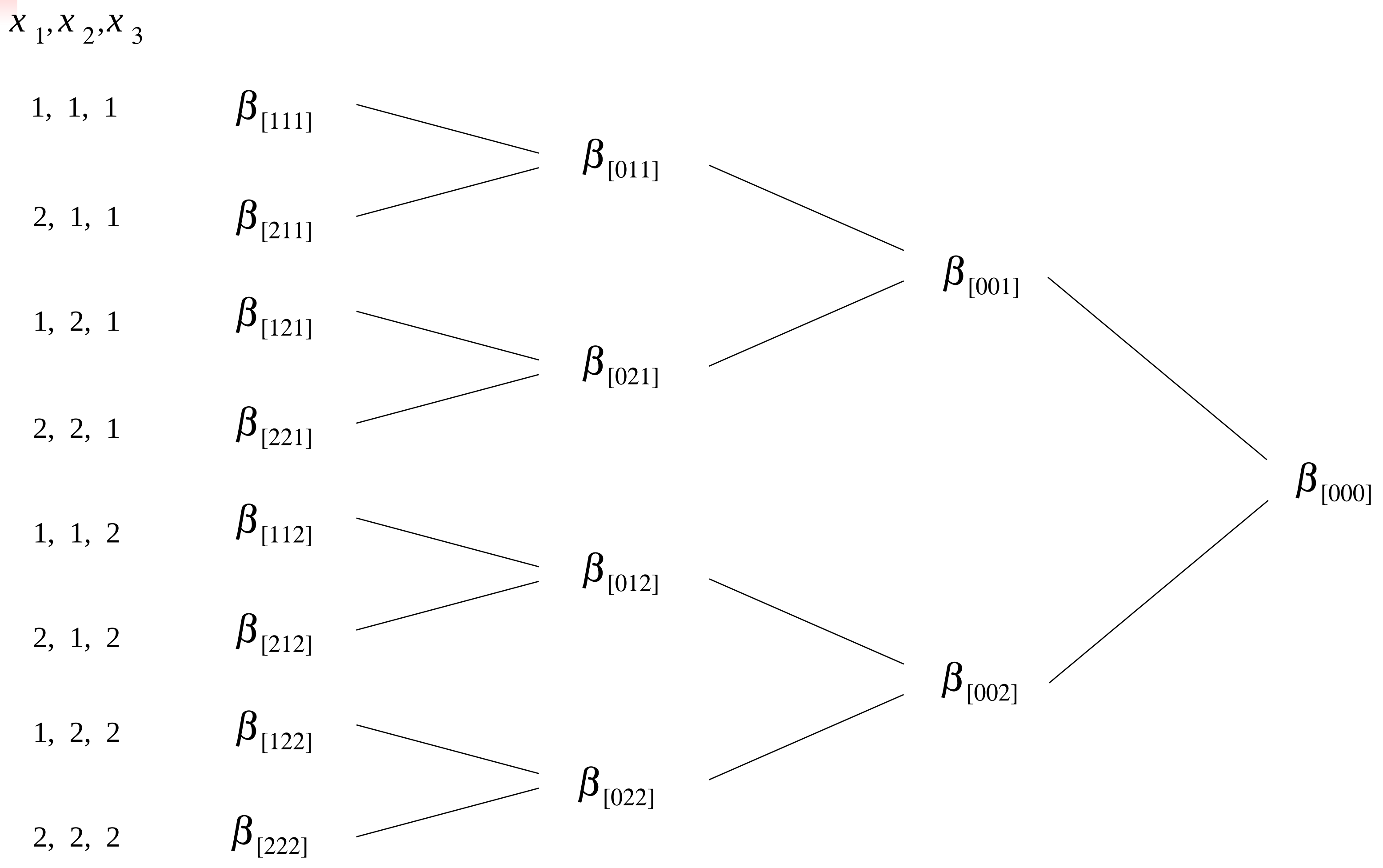
$$l\,(\boldsymbol{x}_{1:O}, \boldsymbol{\beta}^{(k)}) = \sum_{\mathcal{P} \in \boldsymbol{\mathcal{S}}} \beta_{\mathcal{P}}^{(k)}\; I(\boldsymbol{x}_{1:O} \in \mathcal{P}) = \beta_{[0\cdots 0]}^{(k)} + \sum_{t=1}^{O} \beta_{[0\cdots x_t \cdots x_O]}^{(k)}$$

where $\boldsymbol{\mathcal{S}}$ is the set of all patterns with all zero's at the start.

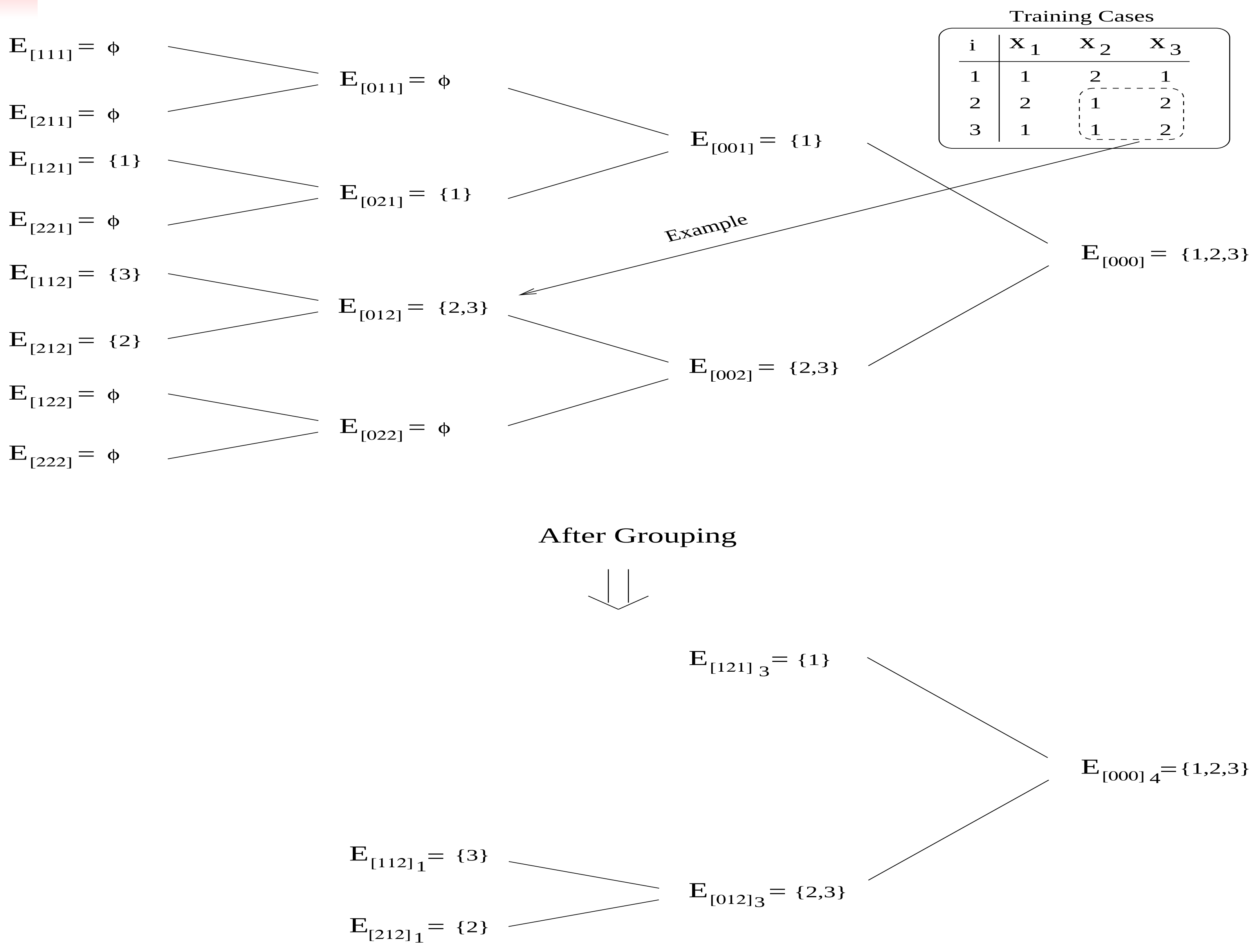
We use the following priors:

$$\begin{aligned} \sigma_t &\sim \text{Inverse-Gamma}(\alpha_t\,, (\alpha_t + 1)\, w_t), \text{ for } t = 0, \ldots, O \\ \beta_{\mathcal{P}}^{(k)} \mid \sigma_{o(\mathcal{P})} &\sim N(0, \sigma_{o(\mathcal{P})}^2) \text{ or Cauchy}(0, \sigma_{o(\mathcal{P})}), \text{ for } \mathcal{P} \in \boldsymbol{\mathcal{S}} \end{aligned}$$

## A Picture of Parameters of LSPM



# Grouping Parameters of LSPM

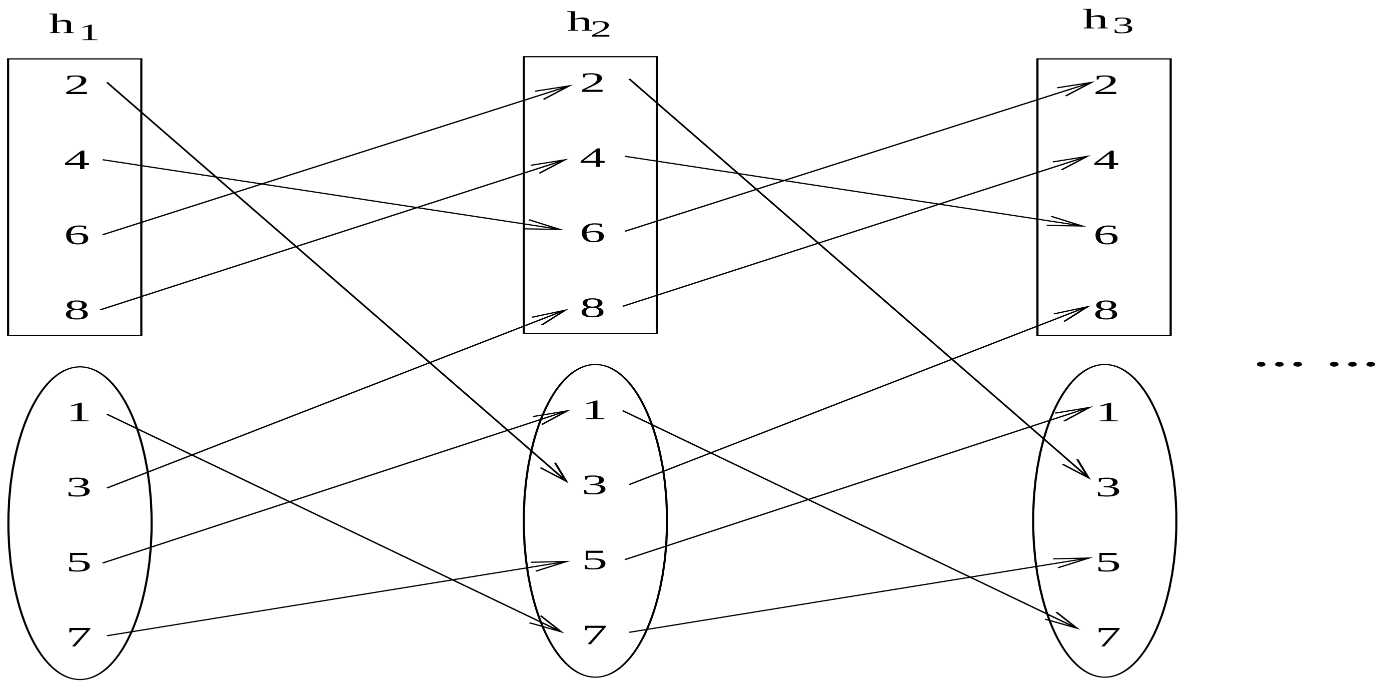


Training Cases

| i | $x_1$ | $x_2$ | $x_3$ |
|---|---|---|---|
| 1 | 1 | 2 | 1 |
| 2 | 2 | 1 | 2 |
| 3 | 1 | 1 | 2 |

## Generating Data with a Hidden Markov Model

To test our method, we used a hidden Markov model (HMM) to generate data. The arrows show the transition probabilities that are equal to $0.95$ for the hidden Markov chain. Other small transition probabilities are not shown. If the hidden state $h_t$ is in rectangle, the $x_t$ is equal to $1$ with probability $0.95$, and $2$ with probability $0.05$. Reverse when the $h_t$ is in oval.



We generated $5500$ sequences of length $21$, $500$ of which were used as training cases and $5000$ as test cases. We want to predict $x_{21}$.

## Specifications of the Priors and Computations

The priors for $\sigma_t$'s:

$\sigma_0$ is fixed at $5$ for Cauchy and $10$ for Gaussian. For $t = 1, \ldots, 20$, $\alpha_t = 0.25$, $w_t = 0.1/t$. The quartiles of the priors for $\sigma_1$ are shown:

| $p$ | $0.01$ | $0.1$ | $0.2$ | $0.3$ | $0.4$ | $0.5$ | $0.6$ | $0.7$ | $0.8$ | $0.9$ | $0.99$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $q$ | $0.05$ | $0.17$ | $0.34$ | $0.67$ | $1.33$ | $2.86$ | $7.13$ | $22.76$ | $115.65$ | $1851.83$ | $1.85 \times 10^7$ |

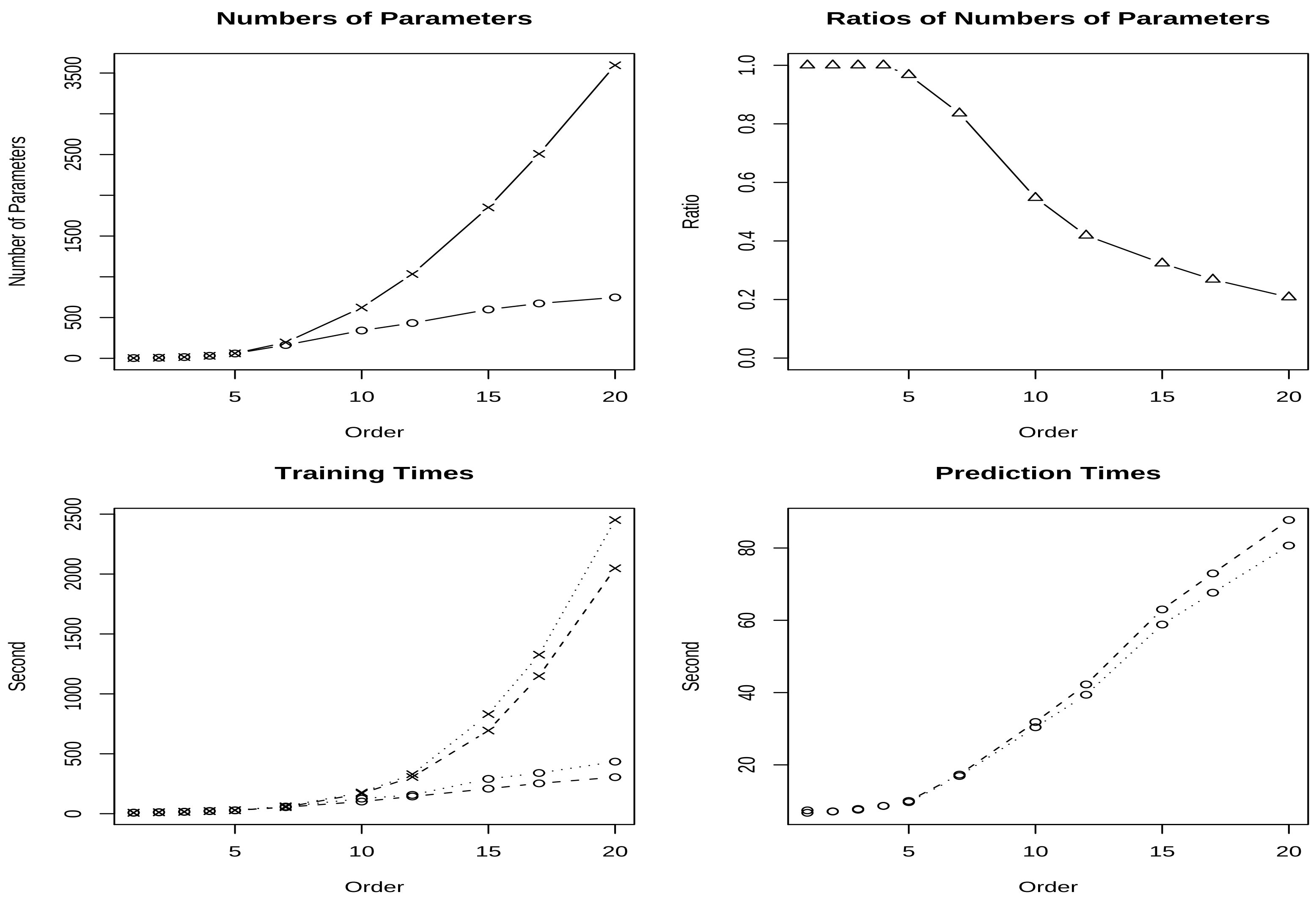
MCMC Computations:

- Slice sampling with "Stepping Out" + "Shrinkage" procedure, for both $\beta$'s and $\sigma$'s. There are two tuning parameters needed to set — $m$, the maximum times of "stepping out", and $w$, the size of stepping out. In sampling for $\beta$'s, $m = 50$,$w = 50$, for $\sigma$'s, $m = 50$ ,$w = 5$.
- Ran $2000$ iterations, with the first $750$ discarded, every $5$th iteration afterward used to make predictions.

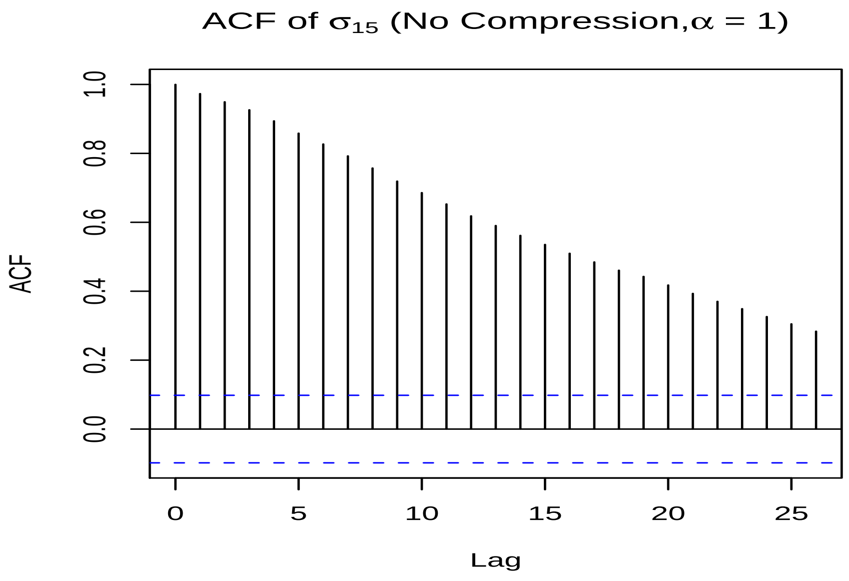
## Reduction of Parameters and Training Time

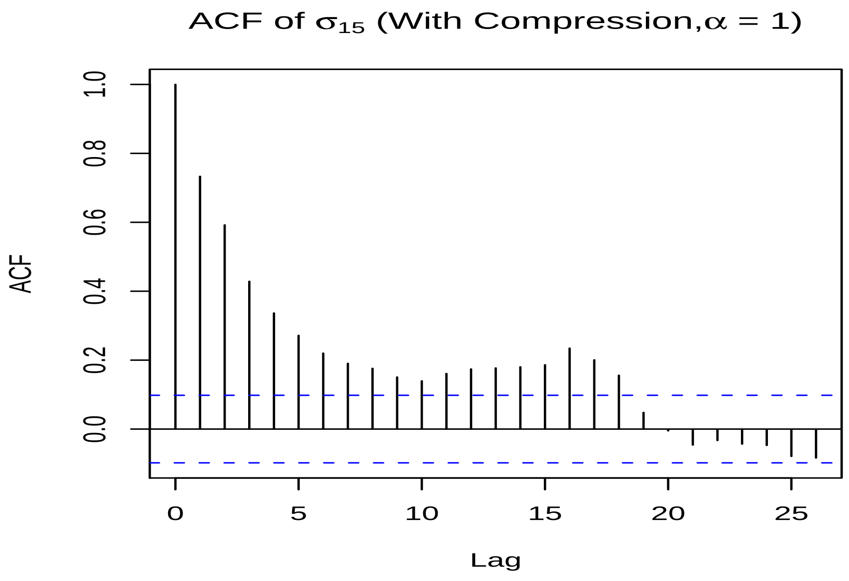
$\times$ — no compression, $o$ — with compression

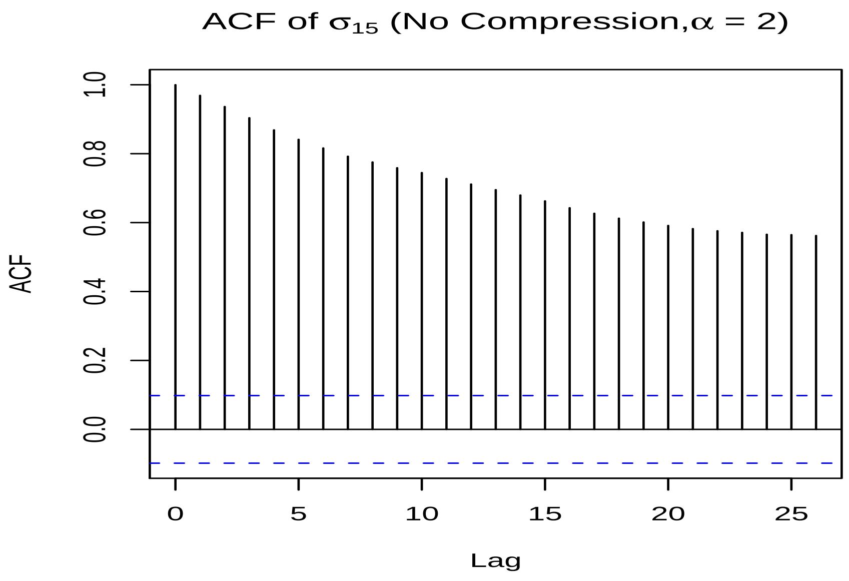
- - - — Gaussian Priors, $\cdots$ — Cauchy Priors

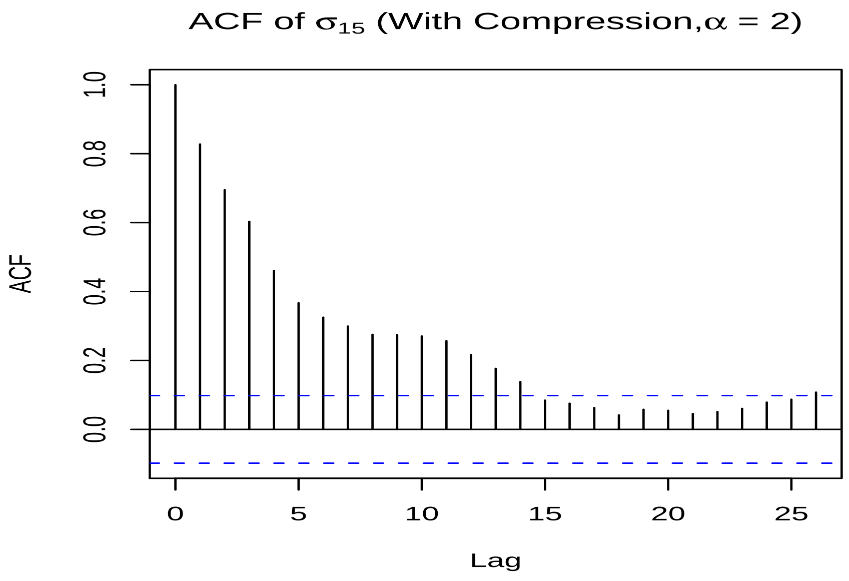


# Reduction of Autocorrelation of MC Samples





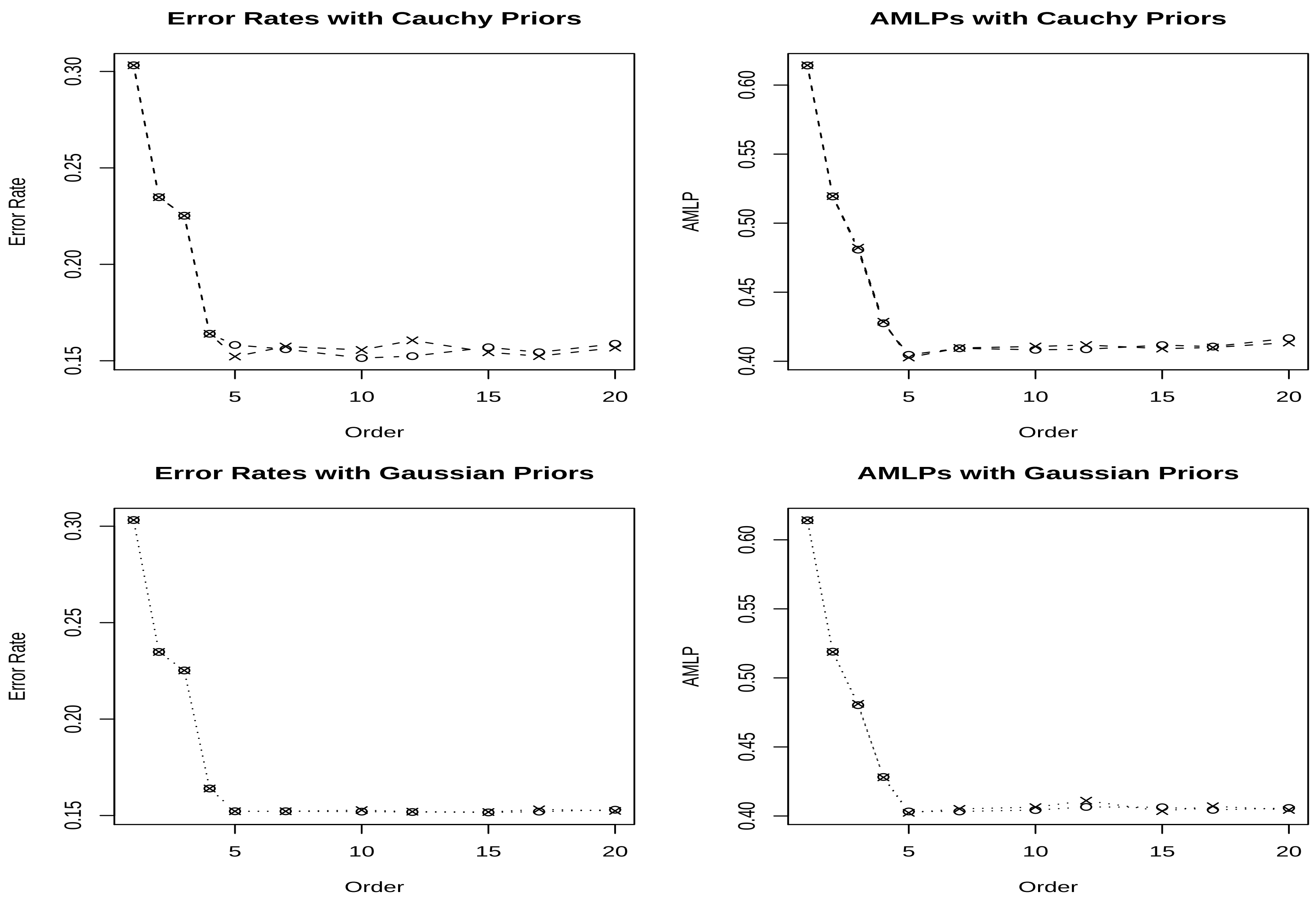




## Error Rates and Averge Minus Log Probabilities

$\times$ — no compression, $o$ — with compression

- - - — Gaussian Priors, $\cdots$ — Cauchy Priors



## Concluding Remarks

- We have proposed a method to greatly reduce the number of parameters for Bayesian models with high-order interactions, by compressing many parameters into $1$. Recovering the original parameters is easy.
- The number of compressed parameters for sequence prediction models will converge to a finite number when $O$ increases, although the number of the original ones will grow exponentially with $O$. Using our compression method, one can handle very high order Bayesian models in reasonable time with MCMC.
- The compression method is applicable to all problems using discrete features as predictors. We have also worked on general classification model with high-order interactions.