# Getting started with IBGS

Lizhong Chen

July 5, 2026

## 1 Introduction

IBGS performs variable selection in ultrahigh dimensions, where the number of predictors p can far exceed the sample size n. The *iterated block Gibbs sampler* grows a small, stable set of important predictors by alternating random block screening and refinement, then records a long Gibbs run over the surviving candidates. The whole search runs in a single multicore C routine; models are scored by the AIC, BIC, AICc or extended BIC criterion. See ?IBGS for an overview and ?glmIBGS for the algorithm in detail.

This vignette walks through a small generalized-linear-model example. The same interface serves the Cox model (coxIBGS/coxGibbs) and the linear mixed model (lmeIBGS/lmeGibbs).

## 2 A small example

Simulate a sparse Gaussian problem with the signal in the first three predictors:

```
> library(IBGS)
> set.seed(1)
> n <- 100
> p <- 60
> x <- matrix(rnorm(n * p), n, p)
> y <- as.numeric(x[, 1:3] %*% c(3, -3, 3) + rnorm(n))
```

Run the block sampler and print the fit:

```
> fit <- glmIBGS(y, x, criterion = "BIC")
> fit

Iterated Block Gibbs Sampler (IBGS)
-----------------------------------
Family:     gaussian
Criterion:  BIC
Predictors: 60    |   Selected (marginal prob > 0.9): 3
  V1, V2, V3
Best model: BIC = 295.9,  size = 6,  visit freq = 0.002
Models retained: 10

Use summary() for the variable/model tables and plot() for diagnostics.
```

The `summary` method tabulates the selected variables and the top models:

```
> summary(fit)

IBGS fit: family = gaussian, criterion = BIC, predictors = 60

Selected variables (marginal prob > 0.9):
 variable marginal.prob best.coef
       V1         1.000     3.164
       V2         1.000    -2.727
       V3         1.000     3.027

Top 10 models (by BIC):
 rank       ic  freq size
    1 295.8930 0.002    6
    2 296.6836 0.002    9
    3 296.7184 0.001    8
    4 296.9731 0.002    7
    5 297.0812 0.001    6
    6 297.1188 0.001    6
    7 297.1349 0.001    8
    8 297.1739 0.002    7
    9 297.1833 0.001    9
   10 297.2015 0.001    7

Convergence diagnostics (on the BIC trace):
  Gelman-Rubin R-hat: 1.000 (upper 1.001)
  Geweke z:           1.119
  Effective size:     1433.0
  Autocorrelation lag 1: -0.137
```

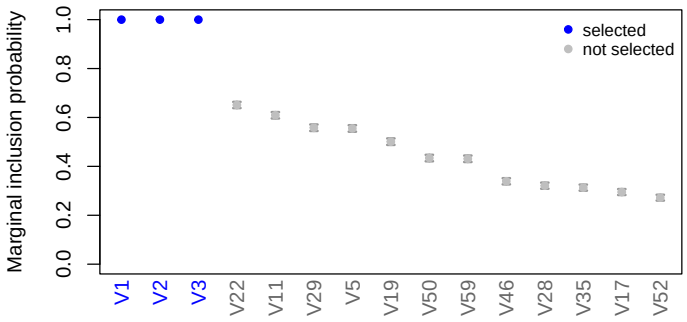
`coef` returns the coefficients of the best model (or, with `average = TRUE`, a model-averaged vector):

```
> head(coef(fit))

(Intercept)          V1          V2          V3          V4          V5
-0.05551364  3.16389564 -2.72707819  3.02697530  0.00000000 -0.18147957
```

The diagnostic plots show the marginal inclusion probabilities, the visit frequency of the top models, and a trace of the criterion sequence. Here is the inclusion-probability plot:

```
> plotMargProb(fit, n.vars = 15)
```




Finally, the `predict` method forms model-averaged predictions on new data, and `fitted` returns them for the training data:

```
> predict(fit, x[1:5, ])

[1]  1.424661  5.490532  3.911689  3.626576 -4.242257

> head(fitted(fit))

[1]  1.424661  5.490532  3.911689  3.626576 -4.242257  0.150048
```

# 3 Tuning the search

The defaults work well, but a few arguments control the search: `block.size` (predictors per screening block), `n.keep` (how many survive each screen), `threshold` (the inclusion-probability cut-off), `n.refine` (refinement rounds) and `n.draws` (run length). Set `n.cores` above 1 to screen blocks in parallel. For a manageable design the non-block sampler `glmGibbs` searches all predictors directly.