Documentation for the Multivariate Normal Gibbs sampler

Anand

February 19, 2007

The Gibbs samplers in `Normal.py` can be applied to parameters x whose children are distributed as follows: For each child c_i,

$$(c_i|x, F_i, a_i, \tau_i) \sim \mathrm{N}(F_i x + a_i, \tau_i).$$

1 Conjugate

The sampling method `cMVNormalWithMVNormalChildren` applies if x's prior is normal:

$$(x|\mu_p, \tau_p) \sim \mathrm{N}(\mu_p, \tau_p).$$

In this case, the log conditional probability up to constants not involving x, $\sum_i \log p_i(c_i|x, \ldots) + \log p_x(x|\ldots)$, is equal to

$$-\frac{1}{2}(c_i - F_i x - a_i)^T \tau_i (c_i - F_i x - a_i) - \frac{1}{2}(x - \mu_p)^T \tau_p (x - \mu_p),$$

which after completing the square yields

$$\begin{aligned}(x|\{c_i\}, \ldots) &\sim \mathrm{N}(\mu, \tau),\\ \tau &= \tau_p + \sum_i F_i^T \tau_i F_i,\\ \mu &= \tau^{-1}(\tau_p \mu_p + \sum_i F_i^T \tau_i (x_i - a_i)).\end{aligned}$$

This sampling method's constructor arguments are:

- `parameter`: The parameter representing x.
- `F_dict`: A dictionary, indexed by child objects, whose values are nodes whose values are the F matrices.
- `a_dict`: Same as `F_dict`, but the values of the nodes are the a arrays.
- `tau_dict`: The obvious.
- `prior_mu`: A node whose value is the prior mean of x.
- `prior_tau`: Same as `prior_mu`, but with the precision matrix.

2 Nonconjugate

The sampling method `cMVNormalWithMVNormalChildren` applies if x's prior $p_x(x)$ is non-normal. In this case, the log joint probability of x up to unimportant constants, $\sum_i \log p_i(c_i|x,\ldots) + \log p_x(x)$, is equal to

$$-\sum_i \frac{1}{2}(c_i - F_i x - a_i)^T \tau_i (c_i - F_i x - a_i) + \log p_x(x).$$

A Metropolis-Hastings step algorithm for this distribution which tends to be efficient is the following:

1. Propose a value x^p for x as if it had an uninformative prior ($\tau_p = \epsilon I$, with $\epsilon << 1$). That is,

$$\begin{aligned}(x_p|\{c_i\},\ldots) &\sim \mathrm{N}(\mu,\tau),\\ \tau &= \sum_i F_i^T \tau_i F_i,\\ \mu &= \tau^{-1}\sum_i F_i^T \tau_i (x_i - a_i).\end{aligned}$$

2. Accept the jump with probability

$$\min\left\{1, \frac{p_x(x^p)}{p_x(x)}\right\}.$$

This sampling method's constructor arguments are simply:

- `parameter`: The parameter representing x.
- `F_dict`: A dictionary, indexed by child objects, whose values are nodes whose values are the F matrices.
- `a_dict`: Same as `F_dict`, but the values of the nodes are the a arrays.
- `tau_dict`: The obvious.

3 Things that would be nice for Gibbs sampling in general

I started a class called Gibbs in `PyMC2/special_SamplingMethods/GibbsSampler.py`, but couldn't figure out how to make it work well. It would be nice if:

- We come up with some consistent class factory for making conjugate/ nonconjugate pairs of Gibbs samplers, maybe similar to what we're doing in distributions.py.
- The accessory dictionaries that get passed in to the constructor should be allowed to contain parameters, nodes, or simple ndarrays. The PyMC objects should have their values extracted automatically somehow.
- Are we willing to insist on consistent parent keying schemes, to avoid having to pass in the accessory dictionaries? If so, we need to standardize across all Gibbs samplers and distributions.py. Compatibility for weird models will require heavy use of utility nodes.
- Each Gibbs sampler holds a OneAtATimeMetropolis instance called 'default' or something which it uses if a Gibbs step fails.