Efficient Sampling of Conditionally Gaussian Markov Random Fields on a Regular Lattice

Jacob Colvin

March 9, 2010

- MRF Papers

1. Markov Random Fields
   - Definition
   - MRF on a Lattice
   - Conditionally Gaussian MRF
   - Percision Matrix
   - Sparse Percision / Dense Covariance
2. Sampling
   - Naive Gibbs
   - Directly From Joint
   - Lavine - Multivariate DLM
   - Maskell, Orton, Gordon - Univariate DLM
3. Summary/Future

MRF Papers

- **Another Look at Conditionally Gaussian Markov Random Fields**, Michael Lavine, Duke University, 1998
- **Fast Sampling of Gaussian Markov Random Fields**, Havard Rue, Journal of the Royal Statistical Society. Series B (Statistical Methodology), Vol. 63, No. 2 (2001), pp. 325-338
- **Efficient Inference for Conditionally Gaussian Markov Random Fields**, Simon Maskell, Matthew Orton, Neil Gordon, Technical Report, University of Cambridge, 2002.

MRF: Definition

- Use a local definition to define a global distribution

$$x_i|x_{\partial i} \sim p(x_i|x_{\partial i})$$
$$y_i|x_i \sim p(y_i|x_i)$$

  (∂i just means the collection of neighbors of i)
- Easy to interpret, expecially spacial structures.
- By Markov property, this can be factored into a joint model. $X \sim MVN(0, \Sigma) = \prod_{i=1}^{N} p(x_i|x_{\partial i})$

MRF Nearest 4 Neighboorhood

		$x_{\partial i}$		
	$x_{\partial i}$	x_i	$x_{\partial i}$	
		$x_{\partial i}$		

(IxJ)

- Because of Markov Property, conditional on row i, rows $i-1$ and $i+1$ are independent.
- Columns work the same way.

Conditionally Gaussian MRF

$$x_i | x_{\partial i} \sim N(\bar{x}_{\partial i}, \sigma^2 / N_{\partial i})$$

$$X \sim MVM(0, \Sigma)$$

$$\Sigma^{-1} = \sigma^{-2}[T_I \otimes I_J + I_I \otimes T_J]$$

$$y_i | x_i \sim N(x_i, \tau^2)$$

$$p(Y|X) \sim N(0, \Sigma^{-1} = \sigma^{-2}(T_I \otimes I_J + I_I \otimes T_J) + \tau^{-2} I_I \otimes I_J)$$

$$T_k = \begin{bmatrix} 1 & -1 & 0 & \dots & 0 \\ -1 & 2 & \ddots & \ddots & \dots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \dots & \ddots & \ddots & 2 & -1 \\ 0 & \dots & 0 & -1 & 1 \end{bmatrix}_{(kxk)}$$

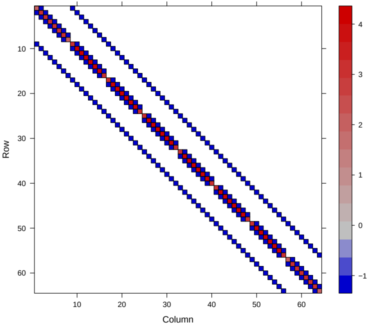
Matrix

$$P_{MRF} = \sigma^{-2} \begin{bmatrix} I_J & -I_J & 0 & \dots & 0 \\ -I_j & 2I_J & \ddots & \ddots & \dots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \dots & \ddots & \ddots & 2I_J & -I_J \\ 0 & \dots & 0 & -I_J & I_J \end{bmatrix} + \sigma^{-2} \begin{bmatrix} T_J & 0 & \dots & 0 \\ 0 & \ddots & \ddots & \dots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & T_J \end{bmatrix}$$

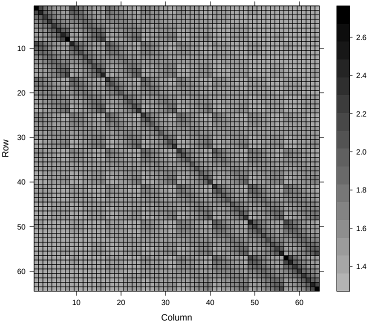
$$T_I = \begin{bmatrix} 1 & -1 & 0 & \dots & 0 \\ -1 & 2 & \ddots & \ddots & \dots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \dots & \ddots & \ddots & 2 & -1 \\ 0 & \dots & 0 & -1 & 1 \end{bmatrix}, H = \begin{bmatrix} 1 & -1 & 0 & \dots \\ 0 & 1 & -1 & \dots \\ \vdots & \ddots & \ddots & \ddots \end{bmatrix}_{(J-1)xJ}$$

Sparse Percision / Dense Covariance

(8x8) latice grid, (64x64) Percision/Covariance matrix w/ nugget.



Dimensions: 64 x 64



Dimensions: 64 x 64

Naive Gibbs

- Easy, use the local definition of the MRF
- Horrid mixing, (remember motivation for FFBS).
- Useless to compare run time because the other methods are direct samples.

Directly From Joint

- Directly from $X|Y \sim MVN(\bar{Y}, (P_{MRF} + \tau^{-2} I_{IJ})^{-1})$
- $(P_{MRF} + \tau^{-2} I_{IJ})^{-1}$ is dense, but $P_{MRF} + \tau^{-2} I_{IJ}$ is sparse.
- If you use dense inversion algorithm, $O(I^3 J^3)$.
- Best to use Cholesky decomp on a sparse matrix.

Sparse Cholesky

- $X \sim MVN(0, \Sigma)$
- $X = Az$ where $\Sigma = AA^t$, A not unique.
- $A_\Sigma = chol(\Sigma) = chol(P^{-1})$
- $A_P = (chol(P)^t)^{-1}$
- While $A_\Sigma \neq A_P$, it is the case that $\Sigma = A_\Sigma A_\Sigma^t - A_P A_P^t$
- This way you don't have to invert P, then do cholesky on a dense Σ, then matrix multiplication.
- Instead, cholesky on sparse P, then solve the sparse system $A_P^{-1}x = z$.

(note that in R, $chol(\Sigma) = A^t$)

Rue - Sparse Matrix

- Rue noted the importance of ordering the percision matrix into a band matrix, but this is only important for a non regular lattice structure.
- Ordering the percision matrix was $\mathcal{O}(IJ^3)$, and sampling is $\mathcal{O}(IJ^2)$.
- In the case of a regular latice, it sounds like sampling should be $\mathcal{O}(IJ^2)$, and no ordering calculations need be made.

Lavine - Multivariate DLM

IDEA: Convert the MRF (IxJ) lattice grid into a Multivariate DLM of I time steps and J dimentions.

$$
\begin{aligned}
y_i|x_i &\sim N(x_i, \tau^2 I_J) && (\tau^{-2} I_I \otimes I_J, \text{ observations}) \\
0|x_i &\sim N(Hx_i, \sigma^2 I_{J-1}) && (\sigma^{-2} I_I \otimes T_J, \text{ pseudo obs}) \\
x_i|x_i - 1 &\sim N(x_{i-1}, \sigma^2 I_J) && (\sigma^{-2} T_I \otimes I_J, \text{ system}) \\
p(x_1) &\propto 1 \\
p(x_1|Y_1) &= MVN((\sigma^{-2} T_J + \tau^{-2} I_J)^{-1} X, (\sigma^{-2} T_J + \tau^{-2} I_J)^{-1})
\end{aligned}
$$

where $H'H = T_J$.

Lavine - Multivariate DLM

- Then use DLM theory to do updating, FFBS etc.
- Problem, is the inversion of a dense (JxJ) covariance matrix for each row. Even with useing Cholesky/SVD instead of direct matrix inversion, this is still $O(J^3)$ per row, making the algorithm $O(IJ^3)$ or quadratic with the number of cells for square lattice structures.

Maskell, Orton, Gordon - Univariate DLM

Process each row as follows...

1. $\downarrow$ Predict: generate a prior for $p(x_{i+1}^{1:J}|y_{1:i}^{1:J})$
2. $\rightarrow$ Update: Forward filter row $i+1$ left to right using Kalman Filter/DLM recursion to generate a sequence of filtering densities $p(x_{i+1}^{1:j}|y_{1:i}^{1:J}, y_{i+1}^{1:j})$
3. $\leftarrow$ Smooth: Smooth backwards to obtain the full joint density $p(x_{i+1}^{1:J}|y_{1:i+1}^{1:J})$

$\downarrow$ $\rightarrow$ $\leftarrow$	$\downarrow$ $\rightarrow$ $\leftarrow$	$\downarrow$ $\rightarrow$ $\leftarrow$	$\downarrow$ $\rightarrow$ $\leftarrow$	$\downarrow$ $\rightarrow$ $\leftarrow$

(IxJ)

And then backward sample like in FFBS, useing the same intuition. $O(IJ)$!!! - It's linear with the number of pixels or cells in the MRF.

Summary

- Matrix inversion, even if you can Cholesky, is very bad and is probably why your sampler is so slow.
- Exploit structure whenever possible. Many multivariate problems have some sort of spatial structure.
- Pay attention to $\mathcal{O}(n)$ notation if you want to scale your problem up very easily.
- Time series models are a natural way to think about building your model sequentially, then think of your actual model as the final smoothed joint distribution.
- Expand MOG for different neighborhood structures.
- Expand inference for static parameters.
- Convert strutured multivariate DLM into univariate DLM.