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

Jacob Colvin

March 9, 2010
1 Markov Random Fields
   - Definition
   - MRF on a Lattice
   - Conditionally Gaussian MRF
   - Precision Matrix
   - Sparse Precision / Dense Covariance

2 Sampling
   - Naive Gibbs
   - Directly From Joint
   - Lavine - Multivariate DLM
   - Maskell, Orton, Gordon - Univariate DLM

3 Summary/Future
Another Look at Conditionally Gaussian Markov Random Fields, Michael Lavine, Duke University, 1998


MRF: Definition

- Use a local definition to define a global distribution
  \[ p(x_i | x_{\partial i}) \]
  \[ p(y_i | x_i) \]

  (\( \partial i \) just means the collection of neighbors of \( i \))

- Easy to interpret, especially spatial structures.

- By Markov property, this can be factored into a joint model.
  \[ X \sim MVN(0, \Sigma) = \prod_{i=1}^{N} p(x_i | x_{\partial i}) \]
Because of Markov Property, conditional on row $i$, rows $i - 1$ and $i + 1$ are independent.

Columns work the same way.
Conditionally Gaussian MRF

\[ x_i | x_{\partial i} \sim N(\bar{x}_{\partial i}, \sigma^2 / N_{\partial i}) \]
\[ X \sim MVM(0, \Sigma) \]
\[ \Sigma^{-1} = \sigma^{-2} [T_I \otimes I_J + I_I \otimes T_J] \]
\[ y_i | x_i \sim N(x_i, \tau^2) \]
\[ p(Y | X) \sim N(0, \Sigma^{-1} = \sigma^{-2}(T_I \otimes I_J + I_I \otimes T_J) + \tau^{-2} I_I \otimes I_J) \]

\[ T_k = \begin{bmatrix}
1 & -1 & 0 & \ldots & 0 \\
-1 & 2 & \ddots & \ddots & \ldots \\
0 & \ddots & \ddots & \ddots & 0 \\
\vdots & \ddots & \ddots & \ddots & 0 \\
0 & \ldots & 0 & 2 & -1 \\
0 & \ldots & 0 & -1 & 1
\end{bmatrix}_{(k \times k)} \]
\[ P_{\text{MRF}} = \sigma^{-2} \left[ \begin{array}{cccccc}
I_J & -I_J & 0 & \cdots & 0 \\
-I_J & 2I_J & \ddots & \ddots & \cdots \\
0 & \ddots & \ddots & \ddots & 0 \\
\vdots & \ddots & \ddots & 2I_J & -I_J \\
0 & \cdots & 0 & -I_J & I_J 
\end{array} \right] + \sigma^{-2} \left[ \begin{array}{cccc}
T_J & 0 & \cdots & 0 \\
0 & \ddots & \ddots & \cdots \\
\vdots & \ddots & \ddots & 0 \\
0 & \cdots & 0 & T_J 
\end{array} \right] \]

\[ T_I = \left[ \begin{array}{cccc}
1 & -1 & 0 & \cdots & 0 \\
-1 & 2 & \ddots & \ddots & \cdots \\
0 & \ddots & \ddots & \ddots & 0 \\
\vdots & \ddots & \ddots & 2 & -1 \\
0 & \cdots & 0 & -1 & 1 
\end{array} \right], \quad H = \left[ \begin{array}{cccc}
1 & -1 & 0 & \cdots \\
0 & 1 & -1 & \cdots \\
\vdots & \ddots & \ddots & \ddots \\
(J-1) & \cdots & \cdots & \cdots 
\end{array} \right] \]
Sparse Percision / Dense Covariance

(8x8) lattice grid, (64x64) Precision/Covariance matrix w/ nugget.
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, \(O(I^3J^3)\).
- Best to use Cholesky decomp on a sparse matrix.
Sparse Cholesky

- $X \sim \text{MVN}(0, \Sigma)$
- $X = Az$ where $\Sigma = AA^t$, $A$ not unique.
- $A_\Sigma = \text{chol}(\Sigma) = \text{chol}(P^{-1})$
- $A_P = (\text{chol}(P)^t)^{-1}$
- 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 $\Sigma$, then matrix multiplication.
- Instead, cholesky on sparse $P$, then solve the sparse system $A_P^{-1}x = z$.

(note that in R, $\text{chol}(\Sigma) = A^t$)
Rue noted the importance of ordering the precision matrix into a band matrix, but this is only important for a non-regular lattice structure.

Ordering the precision matrix was $O(IJ^3)$, and sampling is $O(IJ^2)$.

In the case of a regular lattice, it sounds like sampling should be $O(IJ^2)$, and no ordering calculations need be made.
IDEA: Convert the MRF \((I \times J)\) lattice grid into a Multivariate DLM of \(I\) time steps and \(J\) dimensions.

\[
y_i | x_i \sim N(x_i, \tau^2 I_J)
\]

\[
0 | x_i \sim N(Hx_i, \sigma^2 I_{J-1})
\]

\[
x_i | x_{i-1} \sim N(x_{i-1}, \sigma^2 I_J)
\]

\[
p(x_1) \propto 1
\]

\[
p(x_1 | Y_1) = MVN((\sigma^{-2} T_J + \tau^{-2} I_J)^{-1}X, (\sigma^{-2} T_J + \tau^{-2} I_J)^{-1})
\]

where \(H' H = T_J\).
Then use DLM theory to do updating, FFBS etc.

Problem, is the inversion of a dense \((J \times J)\) covariance matrix for each row. Even with using 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.
Process each row as follows...

1. **Predict**: generate a prior for \( p(x_{i+1}^{1:J}|y_1^{1:i}) \)

2. **Update**: Forward filter row \( i + 1 \) left to right using Kalman Filter/DLM recursion to generate a sequence of filtering densities \( p(x_{i+1}^{1:j}|y_1^{1:i}, y_{i+1}^{1:j}) \)

3. **Smooth**: Smooth backwards to obtain the full joint density \( p(x_{i+1}^{1:J}|y_1^{1:i+1}) \)

\[
\begin{array}{cccccc}
\downarrow & \downarrow & \downarrow & \downarrow & \downarrow & (IxJ) \\
\rightarrow & \rightarrow & \rightarrow & \rightarrow & \rightarrow \\
\leftarrow & \leftarrow & \leftarrow & \leftarrow & \leftarrow
\end{array}
\]

And then backward sample like in FFBS, using 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 $O(n)$ 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.