Bayesian Scientific Computing
Exercise for Day 2

Daniela Calvetti, Erkki Somersalo

Case Western Reserve University
Department of Mathematics, Applied Mathematics and Statistics

Lyngby, December 2019

Random Draws from Gaussian Prior

The goal is to draw random samples from smoothness priors in 1D and 2D.
1D case: Unit interval

$$I = [0, 1],$$

over which we define the second order derivative with vanishing Dirichlet boundary data,

$$\left(\frac{d^2}{dx^2}\right)_D : u \mapsto \frac{d^2 u}{dx^2} \quad u(0) = u(1) = 0.$$

We generate finite difference (FD) random draws of a random variable satisfying

$$\left(\frac{d^2}{dx^2}\right)_D X = W = \text{white noise}.$$

Random Draws from Gaussian Prior

2D case: Image area
$$\Omega = [0,1] \times [0,1].$$

Let Δ_D denote the Laplacian, with the domain of definition consisting of smooth functions vanishing at the boundary ("D" for Dirichlet):

$$\Delta_D \varphi = \frac{\partial^2 \varphi}{\partial x^2} + \frac{\partial^2 \varphi}{\partial y^2}, \quad \varphi\big|_{\partial\Omega} = 0.$$

The goal is to find a finite difference (FD) approximation of generating random variables X that satisfy

$$(-\Delta_D + \lambda^{-2})X = W = \text{white noise},$$

called the Whittle-Matérn (WM) prior model.

Random Draws from Gaussian Prior

FD discretization of the Dirichlet Laplacian: Consider first a 1D model.
Let $u : [0,1] \to \mathbb{R}$, $u(0) = u(1) = 0$.
Divide the interval in n intervals, denoting

$$x_j = jh, \quad 0 \leq j \leq n, \quad h = 1/n.$$

Finite difference approximation of the second derivative:

$$u''(x_j) = \frac{d^2 u}{dx^2}(x_j) \approx \frac{1}{h^2}(u_{j-1} - 2u_j + u_{j+1}), \quad 1 \leq j \leq n-1,$$

where $u_j = u(x_j)$.

Random Draws from Gaussian Prior

Recalling the boundary conditions, recalling that $1/h = n$:

$$
\begin{aligned}
u''(x_1) &= n^2(-2u_1 + u_2), \\
u''(x_2) &= n^2(u_1 - 2u_2 + u_3), \\
&\vdots \\
u''(x_{n-1}) &= n^2(u_{n-2} - 2u_{n-1}).
\end{aligned}
$$

In matrix form,

$$
U'' = \mathsf{L}_1 U,
$$

where $\mathsf{L}_1 \in \mathbb{R}^{(n-1)\times(n-1)}$ is

$$
\mathsf{L}_1 = n^2 \begin{bmatrix} -2 & 1 & & \\ 1 & -2 & 1 & \\ & \ddots & \ddots & \\ & & 1 & -2 \end{bmatrix}.
$$

Random Draws from Gaussian Prior

Task 1: *Write a code that generates the matrix* L_1. *For what follows, make it sparse, using the Matlab command* `spdiags`.

Now 1D random draws. Do the following:

1. Draw $w \sim \mathcal{N}(0, \mathsf{I}_{n-1})$. (`w = randn(n-1,1);`),
2. Solve the problem $\mathsf{L}_1 x = w$. (using "backslash" of Matlab),
3. Add the zero boundary values,

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{bmatrix} \rightarrow \begin{bmatrix} 0 \\ x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ 0 \end{bmatrix}.$$

4. Plot t vs. x, $t = [0, h, 2h, \ldots, (n-1)h, 1]^{\mathsf{T}}$.

Random Draws from Gaussian Prior

Moving to two dimensions: Let $u(x,y)$ be a function in Ω vanishing at the boundary.
Discretize by writing

$$u^{(1)} = \begin{bmatrix} u(x_1,y_1) \\ u(x_2,y_1) \\ \vdots \\ u(x_{n-1},y_1) \end{bmatrix}, \quad u^{(2)} = \begin{bmatrix} u(x_1,y_2) \\ u(x_2,y_2) \\ \vdots \\ u(x_{n-1},y_2) \end{bmatrix}, \ldots u^{(n-1)} = \begin{bmatrix} u(x_1,y_{n-1}) \\ u(x_2,y_{n-1}) \\ \vdots \\ u(x_{n-1},y_{n-1}) \end{bmatrix}.$$

Random Draws from Gaussian Prior

Stack the vectors in a big vector,

$$U = \begin{bmatrix} u^{(1)} \\ u^{(2)} \\ \vdots \\ u^{(n-1)} \end{bmatrix} \in \mathbb{R}^{(n-1)^2}.$$

Calculate the FD differences of all subvectors, that is, find a matrix $\mathsf{D}_1 \in \mathbb{R}^{(n-1)^2 \times (n-1)^2}$ such that

$$\mathsf{D}_1 U = \begin{bmatrix} \mathsf{L}_1 u^{(1)} \\ \mathsf{L}_1 u^{(2)} \\ \vdots \\ \mathsf{L}_1 u^{(n-1)} \end{bmatrix} \in \mathbb{R}^{(n-1)^2}.$$

Random Draws from Gaussian Prior

Such vector is given by

$$\mathsf{D}_1 = \begin{bmatrix} \mathsf{L}_1 & & \\ & \ddots & \\ & & \mathsf{L}_1 \end{bmatrix}.$$

Task 2: *Write a code that generates the matrix* D_1. *Use the Matlab command* `kron`, *and to make sure that the matrix is sparse, use* `speye`.

Hint: If A and B are matrices, the Kronecker product of them is given by

$$\mathsf{A} \otimes \mathsf{B} = \begin{bmatrix} a_{11}\mathsf{B} & \cdots & a_{1n}\mathsf{B} \\ \vdots & & \vdots \\ a_{m1}\mathsf{B} & \cdots & a_{mn}\mathsf{B} \end{bmatrix}.$$

Random Draws from Gaussian Prior

Now we need the second derivatives w.r.t. y. We need a matrix $\mathsf{D}_2 \in \mathbb{R}^{(n-1)^2 \times (n-1)^2}$ such that

$$\mathsf{D}_2 U = n^2 \begin{bmatrix} -2u^{(1)} + u^{(2)} \\ u^{(1)} - 2u^{(2)} + u^{(3)} \\ \vdots \\ u^{(n-2)} - 2u^{(n-1)} \end{bmatrix}.$$

The matrix is

$$\mathsf{D}_2 = \begin{bmatrix} -2\mathsf{I} & \mathsf{I} & & \\ \mathsf{I} & -2\mathsf{I} & \mathsf{I} & \\ & \ddots & & \\ & & \mathsf{I} & -2\mathsf{I} \end{bmatrix}.$$

Random Draws from Gaussian Prior

Task 3: *Write a code that generates the matrix* D_2*. Again, use the Matlab command* `kron`*, and to make sure that the matrix is sparse, use* `speye`*.*

Now the FD approximation of the Dirichlet Laplacian can be written simply as

$$\mathsf{L} = \mathsf{D}_1 + \mathsf{D}_2 \in \mathbb{R}^{(n-1)^2 \times (n-1)^2}.$$

It is a sparse matrix, so despite its large size, it takes not much memory, and numerical operations are lightweight.

Task 4: *Form the Dirichlet Laplacian. Using* `spy`*, check the structure of it, and estimate the fill-in ratio, that is, the percentage of non-zero entries divided by the number of all entries.*

Random Draws from Gaussian Prior

Choose a correlation length $\lambda > 0$, e.g., $\lambda = 0.1$. Then form the matrix

$$\mathsf{M}_\lambda = \mathsf{L} - \frac{1}{\lambda^2}\mathsf{I}_{(n-1)^2}.$$

Task 6: *Generate the matrix* M_λ*, again, using* `speye`*.*

Now we are ready to generate random draws from the WM prior as follows:

1. Draw a vector
$$W \sim \mathcal{N}(0, \mathsf{I}^2_{(n-1)}) \quad (\mathtt{W} = \mathtt{randn}(1, (\mathtt{n}-1)^2));$$
2. Solve the system
$$\mathsf{M}_\lambda X = W.$$

Random Draws from Gaussian Prior

Task 7: *Generate a sample as described above, using the "backslash" to solve the linear system in Step 2.*

Observe: To visualize the outcome, you need to rearrange the vector components back in a matrix. if X is your vector, write first

$$\mathtt{X} = \mathtt{reshape}(\mathtt{X}, \mathtt{n}-1, \mathtt{n}-1).$$

Now your X is a matrix, but the zero boundary values are still missing, so you might want to add a zero frame. This can be done by

```
X_0 = zeros(n+1,n+1)
X_0(2:n,2:n) = X;
```

Random Draws from Gaussian Prior

Task 8: *Generate the matrix* X_0 *and visualize it by* `imagesc`*.*

Task 9: *Write a loop that iterates the whole process, producing several draws with different correlation lengths* λ *to confirm that the outcome is indeed what it should be.*