Non-parametric Regression or Smoothing on a Two Dimensional Lattice using the K-matrix

Stephen P. Smith
email: hucklebird@aol.com

Abstract. A two-dimensional lattice model is described that is able to treat border effects in a coherent way. The model belongs to a class of non-parametric regression models, coming with three smoothness parameters that are estimated from cross validation. The techniques use the K-matrix, which is a typically large and sparse matrix that is also symmetric and indefinite. The K-matrix is subjected to factorization, and algorithmic differentiation, using optimized software, thereby permitting estimation of the smoothness parameters and estimation of the two-dimensional surface. The techniques are demonstrated on real data.

1. Introduction

This paper treats non-parametric regression on a two-dimensional lattice. Smith (1997) describe non-parametric regression on a one-dimensional lattice. Smith estimated a smooth curve by penalizing the log-likelihood, i.e., subtracting α times the square of the approximate second derivative summed over the length of the curve. The parameter α was a smoothness parameter estimated by cross validation as part of differentiating a function of a Cholesky decomposition.

Non-parametric regression by penalizing the integral involving the square of the second derivative, as well as estimating the degree of smoothness by cross validation, are part of standard techniques in non-parametric statistics (Simonoff 1996, Chapter 5). The technique that uses algorithmic differentiation in the quantification of the prediction error from cross validation was also noted by De Hoog, Anderssen and Lukas (2011).

Lattice models are well described by Cressie (1991, Part II). These models may come with nearest-neighbor equations, one equation per lattice point[1] on a two-dimensional spatial grid with regular spacing. A particular challenge is how to treat nearest-neighbor relationships along the lattice border, and do this in a way that is coherent with the spatial model. Approximations are sometimes used to accommodate the edge effects.

Smith's (1997) approach used the Cholesky decomposition of the mixed model matrix. Rather than using the mixed model matrix, better sparse matrix handling is available with the K-matrix introduced by Smith (2001). Perhaps there is enough advantage to attempt non-parametric regression in two dimensions, which is the main goal of the present paper. The K-matrix is symmetric and indefinite, and related to the system of equations describe by Siegel (1965), but perhaps there are other early introductions to

[1] Where grid lines cross.

these systems. These indefinite systems are ubiquitous in application.

A two-dimensional lattice model that is very friendly to border effects is presented in Section 2. The components of the K-matrix are described in Section 3. Cross-validation and its connection to algorithmic differentiation is presented in Section 4. Section 5 provides a successful example of non-parametric regression on a lattice with 25 columns and 20 rows. The example is also enlarged into a lattice with 121 columns and 20 rows, demonstrating that missing values are easy to accommodate. Section 6 presents a short conclusion. Some useful alternative models are presented in Appendix A, including models that may accommodate uneven spatial steps.

2. Model Formulation

An example of a lattice showing 10 rows and 8 columns is provided by Figure 1, where each point represents a lattice value, u_{ij}, that is to be estimated and is surrounded by nearest-neighbors adjacent to the i-th row and j-th column.

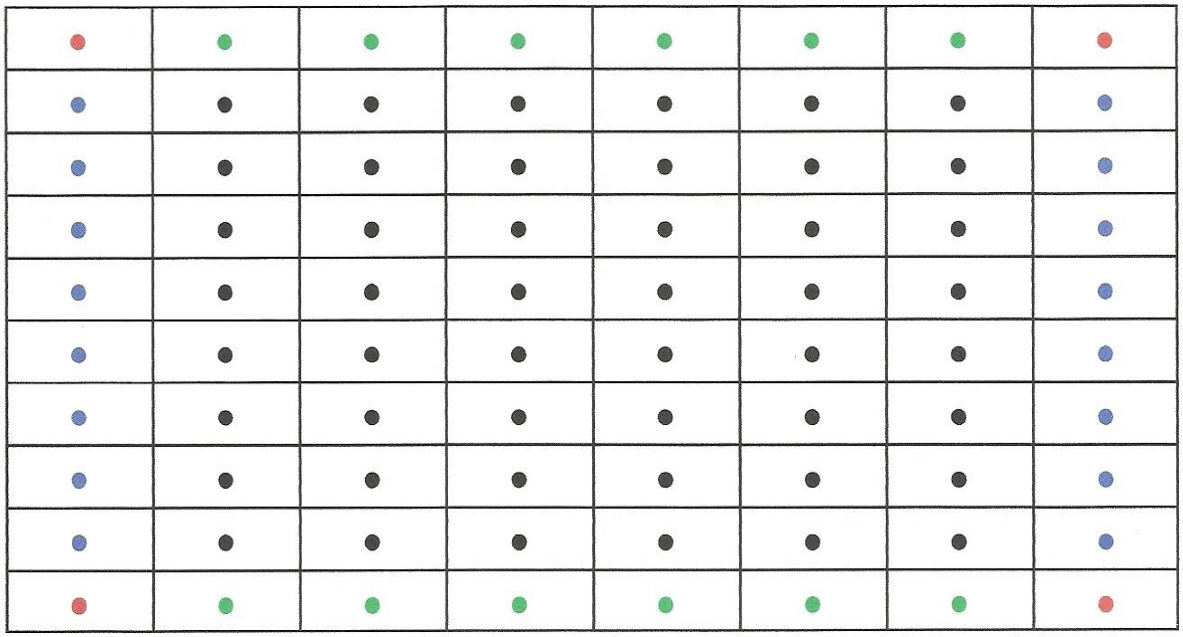

Figure 1. Lattice of 10 rows and 8 columns, where black dots are interior points and colored dots represents points long the border.

The stochastic properties of the lattice are described by nearest-neighbor interactions. Rather than describing these interactions by the set composed of one equation for each lattice value, as done historically, a new conventions is introduced that elegantly treats the boarder effects without the need for ad-hoc adjustments. The new convention uses at most two equations for each lattice point, one equation representing a relationship in

the length direction and the other in the width. Referring to Figure 1, the interior black dots are assigned two equations each, length and width. The values along the left and right edges of the lattice, the blue dots, are each assigned one equation representing width interactions. The green values in the lattice, those located on top and bottom, are each assigned one equation representing the length interactions. The corner values, the red nots, are given no equations. The equations are described by the following for the i-th row and j-th column.

Width or row interactions: $\alpha_{W:ij} = -u_{i-1\,j} + 2u_{ij} - u_{i+1\,j}$

Length or column interactions: $\alpha_{L:ij} = -u_{i\,j-1} + 2u_{ij} - u_{i\,j+1}$ (1)

From the point of view of a penalized likelihood only, variances are introduced as indicated below when both length and width equations are present.

$$Var\begin{bmatrix} a_{W:ij} \\ a_{L:ij} \end{bmatrix} = \begin{bmatrix} v_1 & c \\ c & v_2 \end{bmatrix}$$

If there is only one equation present because the i-th row and j-th column is a cell on the edge of the lattice, then only one variance term (v_1 or v_2) is needed for model specification.

The above formulation is general enough for non-rectangular lattices, coming with straight edges but with corners that form from 90° or 270° angles. Lattices with internal cut outs are also permitted. The specification can also be extended to three dimensions, where interior lattice cells come with three equations representing the length, width, depth dimensions and making the associated variance matrix a 3×3 matrix with six unknown parameters.

The covariances across lattice cells are treated as zeros, i.e., involving $\alpha_{W:ij}$ and $\alpha_{L:ij}$ from the ij-th cell matches with $\alpha_{W:ts}$ and $\alpha_{L:ts}$ from the ts-th cell. While the above formulation is fine from the point of view of a penalized likelihood, where the parameters v_1, v_2 and c define smoothness parameters for a non-parametric regression (as demonstrated below), it is noteworthy that the model so specified carries an incoherence. There are many more equations representing $\alpha_{W:ij}$ and $\alpha_{L:ij}$ than there are lattice values u_{ij}. Therefore, a hypothetical simulation of all the $\alpha_{W:ij}$ and $\alpha_{L:ij}$ will define an over-determined system of equations in terms of all the u_{ij}, and this only becomes an inconsistent system with probability one. It is remarkable that estimating v_1, v_2 and c using cross-validation comes with no complication, whereas a possible restricted maximum likelihood (REML) estimation of the same parameters is not recommended

because of model incoherence.[2]

All the nearest-neighbor interactions representing length and width directional equations can be assembled into a matrix equation given by the following.

$$\boldsymbol{\alpha}=\mathbf{Q}\mathbf{u} \qquad (2)$$

where $\boldsymbol{\alpha}$ is a column vector containing all the various $\alpha_{W:ij}$ and $\alpha_{L:ij}$, and $\mathbf{u}$ is a column vector containing all the various u_{ij}. The matrix $\mathbf{Q}$ assigns the coefficients 2, -1 and 0 to elements of $\mathbf{u}$ to represent all the linear equations given by (1). If a rectangular lattice has n rows and m columns, then $\mathbf{u}$ has $n{\cdot}m$ rows and $\boldsymbol{\alpha}$ has $2(n{\cdot}m -n-m)$ rows.

Define the variance of $\boldsymbol{\alpha}$ to be the block diagonal matrix, $\mathrm{Var}(\boldsymbol{\alpha})=\mathbf{B}$, with blocks of order 1 and 2 depending on whether the associated equation or equations represent an edge interaction or interior interactions. With v_1, v_2 and c specified, the penalty term that gets added to the log-likelihood is $-\frac{1}{2}\hat{\mathbf{u}}^{T}\mathbf{Q}^{T}\mathbf{B}^{-1}\mathbf{Q}\hat{\mathbf{u}}$, which penalizes the sum of squares of approximate second derivatives of the two-dimensional surface under estimation (i.e., $\hat{\mathbf{u}}$ as an estimate of $\mathbf{u}$), summed over both the length and width directions. The implementation of the penalized likelihood comes automatically with the K-matrix application described in Section 3, but further theoretical justification is skipped.

The observational equations are now joined with (2), and these N equations are given by k=1, 2, ...N:

$$y_k = u_{i(k),j(k)} + e_k$$

where the k-th observation belongs to the lattice cell in the i(k)-th row and j(k)-the column. There is no requirement that forces all lattice cells to hold actual observations, many missing observations are permitted. However, it is possible to have too few observations in a row, or column, to permit estimation.

The variance of e_k is denoted by σ^2, and the set of these are assumed uncorrelated. In matrix notation the observation equations become:

$$\mathbf{y}=\mathbf{X}\mathbf{u} + \mathbf{e} \qquad (3)$$

where $\mathbf{y}$ is a N×1 column vector of observations, $\mathbf{X}$ is an $N\times n{\cdot}m$ incidence matrix which selects the appropriate lattice effects in $\mathbf{u}$ and matches them to $\mathbf{y}$, and $\mathbf{e}$ is an N×1

[2] This is not to say that a REML application cannot be redefined in sensible terms. Setting the inverse of the variance matrix of $\mathbf{u}$ to be $\mathbf{Q}^{T}\mathbf{B}^{-1}\mathbf{Q}$ (these bolded arrays are defined below) leads to a coherent formulation, but this will induce some minor adjustments to the normal REML calculations that come with the K-matrix formulation.

column vector of random residuals. The variance matrix for **e** is denoted by:

Var(**e**)=**R**=σ^2 **I**.

3. Building the K-matrix

Models (2) and (3) are now inserted directly into the K-Matrix to produce the following.

$$\mathbf{K} = \begin{bmatrix} \mathbf{B} & & \mathbf{Q} & \\ & \mathbf{R} & \mathbf{X} & \mathbf{y} \\ \mathbf{Q}^T & \mathbf{X}^T & & \\ & \mathbf{y}^T & & \end{bmatrix}$$

Its is useful to redefine **K** by forcing σ^2=1, but leaving both estimation and cross-validation unaffected. This has the effect of redefining both **B** and **R**:

B ← **B** ÷ σ^2,

R ← **I** ,

and this convention is followed through the remainder of the paper. Rather than explicitly replacing **R** with **I**, its better to leave the generality represented by **R** to better describe the differentiation of matrix functions of **K** with respect to r_i, the i-th diagonal of **R**, but evaluated at **R**=**I**. The block-diagonal matrix **B** is now defined in terms of a reparameterization given as variance ratios: $\rho_1 = v_1 \div \sigma^2$, $\rho_2 = v_2 \div \sigma^2$ and ρ_3=c ÷ σ^2. The 1×1 blocks are given by replications of ρ_1 and ρ_2, and the 2×2 blocks are replications of:

$$\begin{bmatrix} \rho_1 & \rho_3 \\ \rho_3 & \rho_2 \end{bmatrix}$$

The matrix **K** is subjected to row and column permutations denoted by the matrix **P**, but leaving the last row and column fixed in the last position. Specifically, a matrix **P** is found such that $\mathbf{P}^T\mathbf{KP}=\mathbf{LDL}^T$ for a nominated set of parameters (ρ_1, ρ_2, ρ_3), where **L** is a lower triangular matrix with positive diagonals and **D** is a diagonal matrix with diagonals 1 and -1.

An initial calculation is required that involves dynamic exchanges of rows and columns

of **K**, its is computationally intensive and needed to calculate **P** and **L** for the first time with an initial set of parameters (ρ_1, ρ_2, ρ_3). This first step arrives at a quasi-symbolic factorization that defines the sparse structure of **L**. However, once **P** is determined and the sparse structure of **L** defined, new factorizations of $\mathbf{P}^T\mathbf{KP}$ then become available provided that the new parameters do not vary too much from the initial selection. Sparse-matrix computations are then readily optimized following the bordering algorithm.

The estimate of **u**, given by $\hat{\mathbf{u}}$, is obtained by solving the following linear equations that are defined with ρ_1, ρ_2 and ρ_3 specified.

$$\begin{bmatrix} \mathbf{B} & & \mathbf{Q} \\ & \mathbf{R} & \mathbf{X} \\ \mathbf{Q}^T & \mathbf{X}^T & \end{bmatrix} \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \hat{\mathbf{u}} \end{bmatrix} = \begin{bmatrix} \mathbf{y} \\ \\ \end{bmatrix} \qquad (4)$$

The white space is this system is understood to contain zeros. Note that the coefficient matrix of equation (4), denoted by **C**, is the leading submatrix of **K** formed by excluding the last row and column of **K**. Moreover, the right-hand side of (4), denoted by **b**, is the last column of **K** with the last diagonal of **K** excluded. It is not surprising, therefore, that the solution to (4) is almost computed already once **L** is evaluated. Partition **P**, **D** and **L** by separating out the last row and column as indicated below.

$$\mathbf{P} = \begin{bmatrix} \overline{\mathbf{P}} & \\ & 1 \end{bmatrix}$$

$$\mathbf{D} = \begin{bmatrix} \overline{\mathbf{D}} & \\ & -1 \end{bmatrix}$$

$$\mathbf{L} = \begin{bmatrix} \overline{\mathbf{L}} & \\ \boldsymbol{\ell}^{\mathrm{T}} & \ell \end{bmatrix}$$

Therefore,

$$\mathbf{P}^T\mathbf{KP} = \begin{bmatrix} \overline{\mathbf{P}}^T\mathbf{C}\overline{\mathbf{P}} & \overline{\mathbf{P}}^T\mathbf{b} \\ \mathbf{b}^T\overline{\mathbf{P}} & \end{bmatrix} = \begin{bmatrix} \overline{\mathbf{L}}\,\overline{\mathbf{D}}\overline{\mathbf{L}}^T & \overline{\mathbf{L}}\,\overline{\mathbf{D}}\boldsymbol{\ell} \\ \boldsymbol{\ell}^T\overline{\mathbf{D}}\overline{\mathbf{L}}^T & \boldsymbol{\ell}^T\overline{\mathbf{D}}\boldsymbol{\ell} - \ell^2 \end{bmatrix} = \mathbf{L}\mathbf{D}\mathbf{L}^T,$$

and all that is needed is to extract from **L** the following upper triangular system that is readily solved by backward substitution to find **g**.

$$\overline{\mathbf{L}}^{T}\mathbf{g} = \boldsymbol{\ell}$$

With **g** computed the sought solution is given by the following.[3]

$$\overline{\mathbf{P}}\mathbf{g} = \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \hat{\mathbf{u}} \end{bmatrix}$$

4. Cross Validation using Algorithmic Differentiation

With $\hat{\mathbf{u}}$ calculated the estimate of the error vector is given by: $\hat{\mathbf{e}}=\mathbf{y}-\mathbf{X}\hat{\mathbf{u}}$. However, with cross validation the k-th observation is dropper in turn (for k=1, 2, ...N) from the data to estimate $u_{i(k),j(k)}$, as $\bar{u}_{i(k),j(k)}$, and this generates the k-th error $\bar{e}_k = y_k - \bar{u}_{i(k),j(k)}$ that is different. One half the prediction error sums of squares, PRESS, is estimated by the following.

$$PRESS = \frac{1}{2}\sum_{k=1}^{N} \bar{e}_k \times \bar{e}_k$$

The proposal is to estimate the parameters (ρ_1, ρ_2, ρ_3) by minimizing PRESS, these parameters acting as smoothness parameters that captures only the strength of nearest-neighbor interactions as found by cross validation that eliminates the direct measurement found for any particular cell.

Fortunately, it is not necessarily to calculate $\hat{\mathbf{u}}$ for each of N data sets where the k-th observation is dropped in turn. The preferred calculation makes an easy to perform adjustment for each element of $\hat{\mathbf{e}}$, where the k-th element is denoted by $\hat{e}_k = y_k - \mathbf{x}_k^T\hat{\mathbf{u}}$ and $\mathbf{x}_k^T$ is the k-th row of **X**. Smith (1997) describes the set of adjustments as an application of backward differentiation. With the present notation the k-th adjustment is given by the following.

[3] Because permutations are treated implicitly in most applications, this last multiplication is rarely performed.

$$\bar{e}_k = \frac{y_k - \mathbf{x}_k^T \hat{\mathbf{u}}}{1 - h_k},$$

where

$$h_k = \frac{\partial \log \left| \mathbf{X}^T \mathbf{W} \mathbf{X} + \mathbf{Q}^T \mathbf{B}^{-1} \mathbf{Q} \right|}{\partial w_k}$$

$$\mathbf{W} = diag\{w_k\} = \mathbf{R}^{-1}$$

The derivatives are evaluated at $\mathbf{W}=\mathbf{R}^{-1}=\mathbf{I}$. The computational advantage is now apparent, as the entire set of h_k , k=1, 2, ... N, is calculated with one reverse pass of backward differentiation. However, the above formulation was developed for the mixed model matrix, but here we are employing the K-matrix and therefore the above results must still be rewritten in terms friendly to the K-matrix. The alternative that's appropriate for the K-matrix follows.

$$\bar{e}_k = \frac{y_k - \mathbf{x}_k^T \hat{\mathbf{u}}}{f_k},$$

where

$$f_k = \frac{\partial \log \|\mathbf{C}\|}{\partial r_k}$$

$$\mathbf{R} = diag\{r_k\}$$

The operation $||\mathbf{C}||$ represents the absolute value of $|\mathbf{C}|$. Note that **C** is still defined as the coefficient matrix of equation (4), being the lead submatrix of **K**. Therefore, $||\mathbf{C}||$ is a function of **K** and **L** as is required for backward differentiation. In particular, the following holds where **K** has order M×M and L_{kk} is the k-th diagonal of **L**.

$$\log \|\mathbf{C}\| = 2 \sum_{k<M} \log(L_{kk}) \qquad (5)$$

The backward differentiation of (5), leading to the calculation of the entire set of f_k, is described by Smith(2018a) where attention is given to the bordering algorithm to factorize a matrix **K** that is symmetric and indefinite. To minimize PRESS with respect to ρ_1, ρ_2, and ρ_3 requires setting the first derivatives to zero in an interactive scheme. Those derivatives are of the following form.

$$\frac{\partial PRESS}{\partial \rho} = \sum_k \bar{e}_k \frac{\partial \bar{e}_k}{\partial \rho}$$

where

$$\frac{\partial \bar{e}_k}{\partial \rho} = -\frac{1}{f_k}\left[\bar{e}_k \frac{\partial f_k}{\partial \rho} + \mathbf{x}_k^T \frac{\partial \hat{\mathbf{u}}}{\partial \rho}\right]$$

Because

$$\frac{\partial f_k}{\partial \rho} = \frac{\partial^2 \log\|\mathbf{C}\|}{\partial r_k \partial \rho}$$

Smith's (2018a) algorithm for second derivatives finds utility, and moreover, the forward derivatives for **L**, i.e., ∂**L**/∂ρ, are obtained in the same algorithm. The forward derivatives are used for computing ∂**û**/∂ρ as outlined below.

$$\bar{\mathbf{L}}^T \mathbf{g} = \boldsymbol{\ell} \quad \Rightarrow \quad \frac{\partial \bar{\mathbf{L}}^T}{\partial \rho}\mathbf{g} + \bar{\mathbf{L}}^T \frac{\partial \mathbf{g}}{\partial \rho} = \frac{\partial \boldsymbol{\ell}}{\partial \rho} \quad \Rightarrow \quad \bar{\mathbf{L}}^T \frac{\partial \mathbf{g}}{\partial \rho} = \frac{\partial \boldsymbol{\ell}}{\partial \rho} - \frac{\partial \bar{\mathbf{L}}^T}{\partial \rho}\mathbf{g}$$

Therefore, ∂**g**/∂ρ is available by backward substitution, and this vector contains ∂**û**/∂ρ.

The above information is enough to provide a steepest descent method to minimize PRESS. A supervised iteration is recommended to adjust the step size and to avoid overshooting. If overshooting occurs the iterations of PRESS will not be monotonically deceasing, which should be prevented by returning to the previous iteration before the overshot occurred and reducing the step size accordingly.

Smith (1997) also described a quasi-Newton iteration that worked well for a one-dimensional lattice. Here second derivatives are approximated by the following.

$$\frac{\partial PRESS}{\partial \rho_i \partial \rho_j} \approx \sum_k \frac{\partial \bar{e}_k}{\partial \rho_i} \frac{\partial \bar{e}_k}{\partial \rho_j}$$

This quasi-Newton iteration can be used as an alternative, but the method likely requires regularization (by adding a constant to the diagonals of the approximate Hessian) to improve its utility while implicitly adjusting step sizes down for large regularization constants. As before, the iterations should be supervised to avoid

overshooting.

Completely derivative free methods can also be used to minimize PRESS. Alternatively, second derivatives maybe calculated exactly if an automatic differentiation tool is available or if a serious commitment is made to software development beyond Smith (2018a). However, with all the first derivatives that are calculated from the steepest descent method (or quasi-Newton iteration), approximating the second derivatives using finite differences is a feasible side calculation that may permit an adaptation that eventually transforms into Newton steps during late stage iteration.

5. Example

The data used to illustrate the method comes from Mercer and Hall (1911) that describes wheat yield in pounds for 500 plots representing a lattice with 20 rows and 25 columns. The total area of the 500 plots is one acre, and a pictorial view of the data is presented in Figure 2. Cressie (1991) also presents these data, and uses them to illustrate spatial models.

Figure 2. Pictorial view of the raw wheat yield over 500 plots from Mercer and Hall (1911).

The first analysis uses the straight lattice model with 500 cells, with no missing observations. The K-matrix has order 1911×1911, and starts with 5554 non-zero elements before factorization.

During minimization it was noted that the estimate of ρ_1 tended close to zero while ρ_2 became relatively large. This was inducing a near singularity in **B**, and resulted in apparent rounding errors in the calculation of PRESS and its derivatives. Therefore, new permutations and a factorization were sought, but by setting the initial parameters

to $\rho_1=\rho_3=0$ and $\rho_2=10$.[4] This remedied the problem, and subsequent calculations were stable even with ρ_1 tending close to zero.

The non-zero structure of **L** contained only 35919 elements, and therefore the sparse-matrix calculations were very efficient. With the permutations and the sparse structure of **L** given, factorization by the bordering algorithm requires less than 1 second of computing time on a Windows machine that runs at 3.30 GHz with 8.0 GB of random access memory.

Using a combination of steepest descent and quasi-Newton iteration, PRESS reduced to 31.8339. The parameters were estimated as $\rho_1=0.03505$, $\rho_2=34.74$ and $\rho_3=-0.08637$, showing considerable column-to-column variation but little row-to-row variation. The estimated surface that is now smoothed by nearest-neighbor influences is displayed in Figure 3. It too shows the large column-to-column variation relative to the small row-to-row variation.

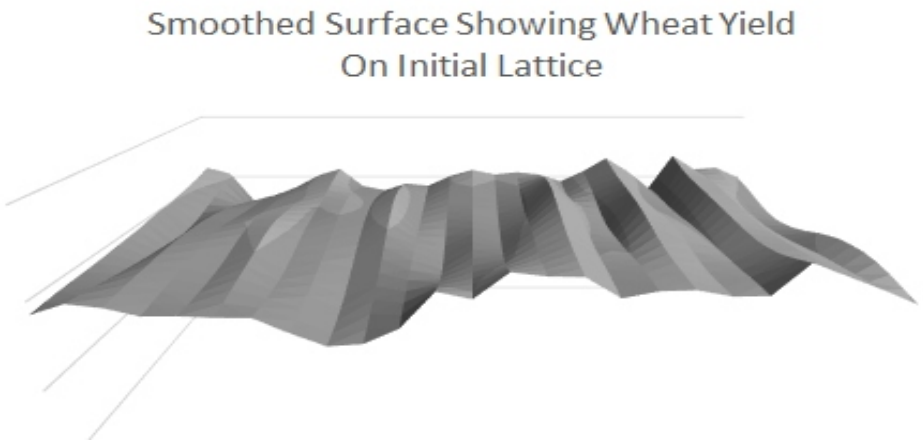

Figure 3. Estimated surface showing wheat yield on a lattice with 25 columns and 20 rows.

The non-smoothness on display in Figure 3 is entirely due to the small number of columns relative to the column-to-column variation. The 20 rows is well matched with the row-to-row variation. Therefore, increasing the number of columns should permit production of a surface that is smoother. The lattice was enlarged to 20 rows and 121 columns by adding four cells between cells with observations in the column dimension. The enlarged lattice contains 1920 cells with missing observations, and this creates no

[4] The software does a minimum degree ordering, but only when pivots are sufficiently removed from zero.

problem to the model or software. The resulting K-matrix is of order 7479×7479, and contains 21874 non-zero elements. To permute and factor the K-matrix the parameters were set to $\rho_1=\rho_3=0$ and $\rho_2=1$, noting that some of the column-to-column variation is dissipated with the finer mesh. The resulting **L** matrix needs non-sparse structure for 245706 non-zero elements. With the permutations and the non-sparse structure set, the computation of **L** by the bordering methods also uses less than 1 second.

The PRESS statistic reduced to 31.7754. The parameters were now estimated as $\rho_1=0.08917$, $\rho_2=-0.05573$ and $\rho_3=0.7482$. This smoothed out the estimated surface sufficiently, and is shown in Figure 4.

Figure 4. Estimated surface showing wheat yield on a lattice with 121 columns and 20 rows

In both minimizations, getting the iterations to converge based on first derivatives alone required some non-trivial adjustments to regularization constants, and step size. Even second derivatives estimated from finite differences proved helpful, because the quasi-Newton method that worked well in Smith (1997) for a one-dimensional lattice needed considerable help. The best way, or better ways, to carry out the minimization were not identified in the present study. No effort was made to evaluate derivative-free minimization, direction set methods, or even conjugate direction searches.

6. Conclusion

Sparse-matrix applications with the K-matrix proved very useful. The ability to exploit sparse structure is greater with the K-matrix than with the mixed model matrix or normal equations as Smith (1997) used. The mixed model matrix has the advantage that it may be symbolically factored, while any factorization of the K-matrix depends on a

parameter set and is only valid for those parameters that are close to the original set. However, the methods applicable to the K-matrix are useful even with **B** singular, unlike the methods based on the mixed model matrix. In fact, as demonstrated in the example a special permutation can be imposed on the K-matrix when **B** is singular or near singular thereby avoiding some serious rounding errors.

The sparse-matrix methods were well matched with the two-dimensional lattice model described in the example; including factorization, cross validation and derivative calculations, even as the actual minimization was not perfected. In general, two-dimensional lattices require more work than one-dimensional lattices, but based on the present experience three dimensional lattices are within reach even as they require more work than two dimensions.

There are three worthy alternatives to the lattice model described in Section 2, and these are presented in Appendix A. All of the extra options are able to accommodate the edge effects in a coherent way with no added complexity, and all of them can be used with the K-matrix.

References

Cullis, B.R., and A.C. Gleeson, 1991, Spatial analysis of field experiments-an extension to two dimensions, *Biometrics*, 47: 1449-1460.

De Hoog, R.F., R.S. Anderssen and M.A. Lukas, 2011, Differentiation of matrix functionals using triangular factorization, *Mathematics of Computation*, 80: 1587-1600.

Cressie, N., 1991, *Statistics for Spatial Data*, John Wiley & Sons, New York.

Jones, R.H., and L.M. Ackerson, 1990, Serial correlation in unequally spaced longitudinal data, *Biometrika*, 77, 4, 721-731

Martin, R., 1979, A subclass of lattices processes applied to a problem in planar sampling, *Biometrika*, 66. 209-217.

Mercer, W.B., and A.D. Hall, 1911, The experimental error of field trials, *Journal of Agriculture Science*, 4: 107-132.

Siegel, I.H., 1965, Deferment of Computation in the Method of Least Squares, *Mathematics of Computation*, 19 (90): 329-331.

Simonoff, J.S., 1996, *Smoothing Methods in Statistics*, Springer, New York.

Smith, S.P., 1997, Sparse matrix tools for Gaussian models on lattices, *Computational Statistics & Data Analysis*, 26: 1-15.

Smith, S.P., 2001, Likelihood-based analysis of linear state-space models using the

Cholesky decomposition, *Journal of Computational and Graphical Statistics*, 10 (2): 350-369.

Smith, S.P., 2018a, The backward differentiation of the bordering algorithm for an indefinite Cholesky factorization, Data Structures and Algorithms, viXra archived #1707.0239.

Smith, S.P., 2018b, Autoregressive and rolling moving average processes using the K-matrix with discrete but unequal time steps, Statistics, viXra archived #1809.0279.

Appendix A

The following is a discussion of three possible alternatives to the model described in Section 2.

1. *Use a Separable Process.* The first approach is to give each spatial dimension an orientation like time that points from a beginning to an end. Then treat the two-dimensional spatial process as separable in two univariate processes where variance matrices are defined by the Kronecker product (Martin 1979). Cullis and Gleeson (1991) present a two-dimensional model that is separable and made from two autoregressive and rolling moving average (ARMA) processes. Smith (2018b) uses the same model and builds the K-matrix.

2. *Use Uneven Spatial Steps.* The second alternative is to embellish the first option that involves separable processes by defining a model on continuous spatial steps, where N observations correspond to N-1 uneven intervals for each spatial dimension. A general two-dimensional lattice then has no more than N^2 lattice cells[5], making the K-matrix with smaller order.

In general, a model with continuous time steps comes as a solution to a stochastic differential equation. Jones and Ackerson (1990) describe ARMA models with continuos time steps, and develops the state-space equations. It is easy to views these as components of a separable two-dimensional spacial process.

Its is preferred to generalize model (1) for continuous and uneven spatial steps, however, rather than holding to ARMA processes. The 2nd order stochastic differential equation $u(t)'' = \lambda \times \xi(t)$, where $\xi(t)$ represents white noise with an imagined time variable t representing one oriented space dimension, corresponds directly to a penalty $[u(t)'']^2$ that is appended to the log-likelihood in a non-parametric regression. Therefore, the better fit is to replace the non-parametric regression in one dimension with a solution to this stochastic differential equation. The process, $u(t)' = W(t)$ is a Wiener process, and

[5] By comparison, kriging, kernel regression and support vector machines may use dense matrix operations on matrices of order N. The saving grace is that the K-matrix works with sparse matrices.

u(t) is defined in the following state-space equations.

$$W(t) = W(t_o) + \varepsilon_1$$
$$u(t) = u(t_o) + \Delta \times W(t_o) + \varepsilon_2$$

where

$$\Delta = t - t_o$$
$$E\{\varepsilon_1\} = E\{\varepsilon_2\} = 0$$
$$Var\begin{Bmatrix}\varepsilon_1 \\ \varepsilon_2\end{Bmatrix} = \lambda \times \begin{bmatrix}\Delta & \frac{\Delta^2}{2} \\ \frac{\Delta^2}{2} & \frac{\Delta^3}{3}\end{bmatrix}$$

The errors, ϵ_1 and ϵ_2 are multivariate normal. Therefore, each space dimension comes with its own state-space model representing 2×1 state vectors that go through transitions. Initially there are two sets of state-space equations, each representing the collective row[6], or the collective column, as indicated below.

$\mathbf{b}_c=\mathbf{P}_c\mathbf{b}_c+\mathbf{e}_c$ or $\mathbf{0}=(\mathbf{I}-\mathbf{P}_c)\mathbf{b}_c+\mathbf{e}_c$
$\mathbf{b}_r=\mathbf{P}_r\mathbf{b}_r+\mathbf{e}_r$ or $\mathbf{0}=(\mathbf{I}-\mathbf{P}_r)\mathbf{b}_r+\mathbf{e}_r$

$Var\{\mathbf{e}_c\}=\mathbf{V}_c$
$Var\{\mathbf{e}_r\}=\mathbf{V}_r$

The vector $\mathbf{b}_c$, or $\mathbf{b}_r$, is intended to represent realization of W(t) and u(t) that come paired in any row, or column. The specification is enough to populated the matrices $\mathbf{P}_c$,$\mathbf{P}_r$, $\mathbf{V}_c$ and $\mathbf{V}_r$ based on two independent models that solve the stochastic differential equation separately for rows and columns; note that the parameter λ differs for rows and columns, hence $\lambda=\lambda_c$ or λ_r.

The actual model for the entire lattice is now given neatly using the Kronecker product that's denoted by ⊗, and is presented below.

$$\mathbf{0}=(\mathbf{I}-\mathbf{P}_c)\otimes(\mathbf{I}-\mathbf{P}_c)\,\mathbf{b}+\mathbf{e} \quad \text{or} \quad \mathbf{0}=(\mathbf{I}-\mathbf{P}_c\otimes\mathbf{P}_c)\,\mathbf{b}+\mathbf{e} \qquad (6)$$

$Var\{\mathbf{e}\} = \mathbf{V}_c \otimes \mathbf{V}_r$

The vector **b** is now intended to represent realization of W(t) and u(t) that come paired and populate the entire lattice, in an order that conforms with the Kronecker product. The model given by (6) feeds directly into the K-matrix.

[6] Thought of abstractly, and not an actual row.

Unlike model (1), model (6) only represents one equation for each element of **b**. Therefore, the model given in Section 2 does not represent a separable process.[7]

3. *Use Uneven Spatial Steps but On a Non-separable Process.* The third option jettisons the separable processes and the Kronecker product that had been used for the first and second alternatives, but keeps the ideas that each spatial dimension has a time-like orientation and that spatial steps are uneven. This leads back to the same solution to the stochastic differential equation that gave state-space equations for u(t) and W(t) that now populate the lattice. The goal is now to represent this model following the style of Section 2.

All the state-space equations are given by the following system of equations.

$$\mathbf{b}=\mathbf{P}\mathbf{b}+\epsilon \quad \text{or} \quad \mathbf{0}=(\mathbf{I}-\mathbf{P})\mathbf{b}+\epsilon \qquad (7)$$

The vector **b** is the same in (6). But **P** is now populated directly with the state-space equations for u(t) and W(t) at each lattice point, down each row and up each column. Therefore, (7) has many more equations than does (6). The matrix, Var{ϵ}, is also specified differently, and non-zero only corresponding to joints where a row crosses a column. For each joint there remains a possible 4×4 variance matrix that is a submatrix of Var{ϵ}, defined as the variances and covariances of ϵ_1 and ϵ_2 for each spatial dimension (see above solution for Alternative 2). The parameter λ serves as a proxy for ρ_1 or ρ_3 letting $\Delta = \Delta_W$ or Δ_L for the spatial change in width and length, at the particular joint. The entire 4×4 variance matrix is needed, however, and the following is suitable.

$$Var\begin{Bmatrix}\varepsilon_{1W}\\ \varepsilon_{2W}\\ \varepsilon_{1L}\\ \varepsilon_{2L}\end{Bmatrix} = \begin{bmatrix}\rho_1 B(\Delta_W,\Delta_W) & \rho_2 B(\Delta_W,\Delta_L)\\ \rho_2 B(\Delta_W,\Delta_L) & \rho_3 B(\Delta_L,\Delta_L)\end{bmatrix}$$

$$B(\Delta_x,\Delta_y) = \begin{bmatrix}\sqrt{\Delta_x\Delta_y} & \dfrac{\Delta_x\Delta_y}{2}\\ \dfrac{\Delta_x\Delta_y}{2} & \dfrac{\Delta_x^{3/2}\Delta_y^{3/2}}{3}\end{bmatrix}$$

In any of the three alternatives above, the observational equations are added to the K-matrix as originally prescribed (see Smith 2001).

While limiting the number of lattice cells to N^2 is advantages by keeping the K-matrix as small as possible, note that going from univariate variables (e.g, Section 2) to state vectors works in the opposite direction by increasing the order of **K**.

[7] At least on first impression.