Multivariate spatial models and the `multiKrig` class

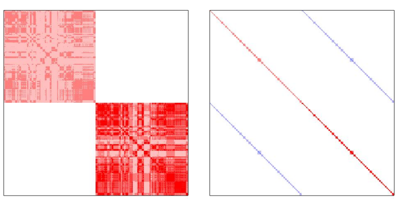
Stephan R. Sain, IMAGe, NCAR

ENAR Spring Meetings

March 15, 2009

Outline

- Overview of multivariate spatial regression models.
- Case study: pedotransfer functions and soil water profiles.
- The `multiKrig` class
  - Case study: NC temperature and precipitation.



A Spatial Regression Model

- A spatial regression model:

$$\underset{(n\times 1)}{\mathbf{Y}} = \underset{(n\times q)(q\times 1)}{\mathbf{X}\beta} + \underset{(n\times 1)}{\mathbf{h}} + \underset{(n\times 1)}{\epsilon}$$

  where
  - E$[\mathbf{h}] = \mathbf{0}$, Var$[\mathbf{h}] = \Sigma_{\mathbf{h}}$
  - E$[\epsilon] = \mathbf{0}$, Var$[\epsilon] = \sigma^2\mathbf{I}$.
  - $\mathbf{h}$ and ϵ are independent.

- $\mathbf{Y} \sim \mathcal{N}(\mathbf{X}\beta, \mathbf{V})$, $\mathbf{V} = \Sigma_{\mathbf{h}} + \sigma^2\mathbf{I}$

- $\widehat{\beta} = (\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-1}\mathbf{X}'\mathbf{V}^{-1}\mathbf{Y}$, $\widehat{\mathbf{h}} = \Sigma_{\mathbf{h}}\mathbf{V}^{-1}(\mathbf{Y} - \mathbf{X}\widehat{\beta})$

Multivariate Regression

- A multivariate, multiple regression model:

$$\underset{(n \times p)}{\mathbf{Y}} = \underset{(n \times q)(q \times p)}{\mathbf{X}\boldsymbol{\beta}} + \underset{(n \times p)}{\boldsymbol{\epsilon}}$$

  where

  – Each of the n rows of $\mathbf{Y}$ represents a p-vector observation.

  – Each of the p columns of $\boldsymbol{\beta}$ represent regression coefficients for each variable.

  – The rows of $\boldsymbol{\epsilon}$ represents a collection of iid error vectors with zero mean and common covariance matrix, $\boldsymbol{\Sigma}$.

Multivariate Regression

- MLEs are straightforward to obtain:

$$\begin{aligned} \underset{(q \times p)}{\widehat{\boldsymbol{\beta}}} &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} \\ \underset{(p \times p)}{\widehat{\boldsymbol{\Sigma}}} &= \frac{1}{n}\mathbf{Y}'\mathbf{P}\mathbf{Y} \end{aligned}$$

  where $\mathbf{P} = \mathbf{I} - \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$.

- Note that the columns of $\widehat{\boldsymbol{\beta}}$ can be obtained through p univariate regressions.

Vec and Kronecker

- The Kronecker product of an $m \times n$ matrix $\mathbf{A}$ and an $r \times q$ matrix $\mathbf{B}$ is an $mr \times nq$ matrix:

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \cdots & a_{1n}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \cdots & a_{2n}\mathbf{B} \\ \vdots & & \ddots & \vdots \\ a_{m1}\mathbf{B} & a_{m2}\mathbf{B} & \cdots & a_{mn}\mathbf{B} \end{bmatrix}$$

- Some properties:

$$\begin{aligned} \mathbf{A} \otimes (\mathbf{B} + \mathbf{C}) &= \mathbf{A} \otimes \mathbf{B} + \mathbf{A} \otimes \mathbf{C} \\ \mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}) &= (\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C} \\ (\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) &= \mathbf{AC} \otimes \mathbf{BD} \\ (\mathbf{A} \otimes \mathbf{B})' &= \mathbf{A}' \otimes \mathbf{B}' \\ (\mathbf{A} \otimes \mathbf{B})^{-1} &= \mathbf{A}^{-1} \otimes \mathbf{B}^{-1} \\ |\mathbf{A} \otimes \mathbf{B}| &= |\mathbf{A}|^m |\mathbf{B}|^n \end{aligned}$$

Vec and Kronecker

- The vec-operator stacks the columns of a matrix:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \qquad \text{vec}(\mathbf{A}) = \begin{bmatrix} a_{11} \\ a_{21} \\ a_{12} \\ a_{22} \end{bmatrix}$$

- Some properties:

$$\begin{aligned} \text{vec}(\mathbf{AXB}) &= (\mathbf{B}' \otimes \mathbf{A})\, \text{vec}\, \mathbf{X} \\ \text{tr}(\mathbf{A}'\mathbf{B}) &= \text{vec}(\mathbf{A})'\, \text{vec}(\mathbf{B}) \\ \text{vec}(\mathbf{A} + \mathbf{B}) &= \text{vec}(\mathbf{A}) + \text{vec}(\mathbf{B}) \\ \text{vec}(\alpha \mathbf{A}) &= \alpha\, \text{vec}(\mathbf{A}) \end{aligned}$$

Multivariate Regression Revisited

- Rewrite the multivariate, multiple regression model:

$$\underset{(np \times 1)}{\text{vec}(\mathbf{Y})} = \underset{(np \times qp)(qp \times 1)}{(\mathbf{I}_p \otimes \mathbf{X})\, \text{vec}(\boldsymbol{\beta})} + \underset{(np \times 1).}{\text{vec}(\boldsymbol{\epsilon})}$$

- What is Var[vec $\boldsymbol{\epsilon}$]?

- What is the GLS estimator for vec($\boldsymbol{\beta}$)?

A Multivariate Spatial Model

- Extend the multivariate, multiple regression model:

$$\underset{(np \times 1)}{\text{vec}(\mathbf{Y})} = \underset{(np \times qp)(qp \times 1)}{(\mathbf{I}_p \otimes \mathbf{X})\, \text{vec}(\boldsymbol{\beta})} + \underset{(np \times 1)}{\text{vec}(\mathbf{h})} + \underset{(np \times 1),}{\text{vec}(\boldsymbol{\epsilon})}$$

where

$$\begin{aligned} \text{Var}[\text{vec}(\mathbf{h})] &= \boldsymbol{\Sigma}_{\mathbf{h}} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} & \cdots & \boldsymbol{\Sigma}_{1p} \\ \boldsymbol{\Sigma}'_{12} & \boldsymbol{\Sigma}_{22} & \cdots & \boldsymbol{\Sigma}_{2p} \\ \vdots & & \ddots & \vdots \\ \boldsymbol{\Sigma}'_{12} & \boldsymbol{\Sigma}'_{2p} & \cdots & \boldsymbol{\Sigma}_{pp} \end{bmatrix} \\ \text{Var}[\text{vec}(\boldsymbol{\epsilon})] &= \boldsymbol{\Sigma} \otimes \mathbf{I}_n \end{aligned}$$

A Multivariate Spatial Model

- One simplification to the spatial covariance matrix is to use a Kronecker form:

$$\boldsymbol{\Sigma}_{\mathbf{h}} = \boldsymbol{\rho} \otimes \mathbf{K}$$

  where

  - $\boldsymbol{\rho}$ is a $p \times p$ matrix of scale parameters
  - $\mathbf{K}$ is an $n \times n$ spatial covariance.

A Multivariate Spatial Model

- Extend the multivariate, multiple regression model:

$$\underset{(np \times 1)}{\text{vec}(\mathbf{Y})} = \underset{(np \times qp)(qp \times 1)}{(\mathbf{I}_p \otimes \mathbf{X})\,\text{vec}(\boldsymbol{\beta})} + \underset{(np \times 1)}{\text{vec}(\mathbf{h})} + \underset{(np \times 1)}{\text{vec}(\boldsymbol{\epsilon})}$$

  OR

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{h} + \boldsymbol{\epsilon}$$

- Now everything follows...

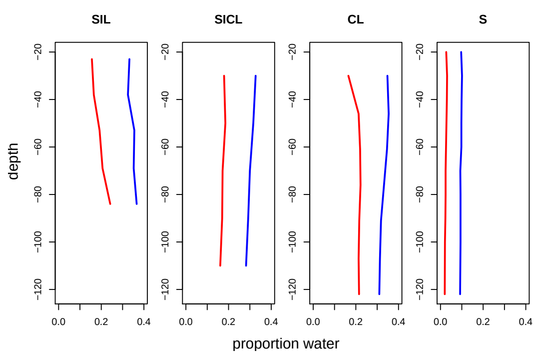
Case Study: Pedotransfer Functions

- Soil characteristics such as composition (clay, silt, sand) are commonly measured and easily obtainable.
- Unfortunately, crop models require water holding characteristics such as the wilting point or lower limit (LL) and the drained upper limit (DUL) which are not so easy to obtain.
  - Often the LL and DUL are a function of depth - soil water profile.

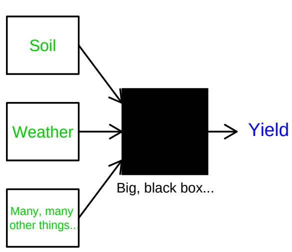
Case Study: Pedotransfer Functions

- *Pedotransfer functions* are commonly used to estimate LL and DUL.
  - Differential equations, regression, nearest neighbors, neural networks, etc.
  - Often specialized by soil type and/or region.
- Develop a new type of pedotransfer function that can capture the entire soil water profile (LL & DUL as a function of depth).
  - Characterize the variation!

Soil Water Profiles



The Big Picture



The CERES Crop Model

- Soil
  - Water holding characteristics
  - Bulk density
  - Etc.
- Weather (20 years)
  - Solar radiation
  - Temperature max/min
  - Precipitation

The Big Picture

- Given a complicated array of inputs, the CERES crop model will give the yields of, for example, maize.
- Deterministic output – variation in yields also of interest.
- Goals:
  - Establish a framework to study sources of variation in crop yields.
  - Assess impacts of climate change on crop yields.

Data

- $n = 272$ measurements on $N = 63$ soil samples
  - Gijsman et al. (2002)
  - Ratliff et al. (1983), Ritchie et al. (1987)
- Includes measurements of:
  - depth,
  - soil composition and texture
    - percentages of clay, sand, and silt
  - bulk density, organic matter, and
  - field measured values of LL and DUL.

Data

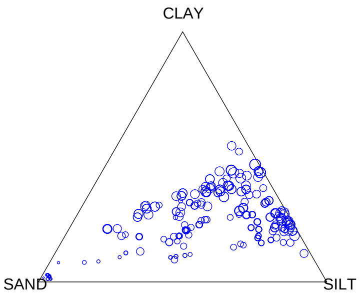
- The soil texture measurements form a composition

$$Z_{\text{clay}} + Z_{\text{silt}} + Z_{\text{sand}} = 1$$

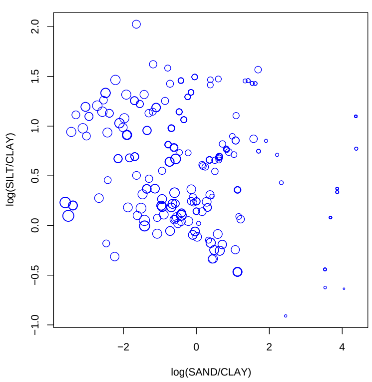
  and Z_{clay}, Z_{silt} ,Z_{sand} are the proportions of each soil component.
  - Not really three variables...
- To remove the dependence, the additive log-ratio transformation (Aitchinson, 1986) is applied, defining two new variables

$$X_1 = \log\left(\frac{Z_{\text{sand}}}{Z_{\text{clay}}}\right) \qquad X_2 = \log\left(\frac{Z_{\text{silt}}}{Z_{\text{clay}}}\right).$$

Data - Composition vs LL



Data - Composition vs LL



A Multi-objective Pedotransfer Function

- The model for the multi-objective pedotransfer function for a particular soil is

$$\mathbf{Y}_0 = \mathbf{T}_0\boldsymbol{\beta} + \mathbf{h}(\mathbf{X}_0) + \boldsymbol{\epsilon}(\mathbf{D}_0)$$

where

$$\mathbf{Y}_0 = \log \begin{bmatrix} \mathrm{LL}_1 \\ \vdots \\ \mathrm{LL}_d \\ \Delta_1 \\ \vdots \\ \Delta_d \end{bmatrix},$$

and d is the number of measurements (depths) and $\Delta_i = \mathrm{DUL}_i - \mathrm{LL}_i$.

A Multi-objective Pedotransfer Function

- The model for the multi-objective pedotransfer function for a particular soil is

$$\mathbf{Y}_0 = \mathbf{T}_0\boldsymbol{\beta} + \mathbf{h}(\mathbf{X}_0) + \boldsymbol{\epsilon}(\mathbf{D}_0)$$

where

$$\mathbf{T}_0 = \begin{bmatrix} \mathbf{1} & \mathbf{X}_0 & \mathbf{Z}_{\mathsf{LL},0} & & \mathbf{0} & \\ & \mathbf{0} & & \mathbf{1} & \mathbf{X}_0 & \mathbf{Z}_{\triangle,0} \end{bmatrix},$$

and

  - $\mathbf{X}_0$ is the transformed soil composition information
  - $\mathbf{Z}_{\mathsf{LL}}$ and $\mathbf{Z}_{\triangle}$ are additional covariates for LL and $\triangle$.
    - $\mathbf{Z}_{\mathsf{LL}}$ includes organic carbon
    - $\mathbf{Z}_{\triangle}$ includes linear and quadratic terms for depth

A Multi-objective Pedotransfer Function

- The model for the multi-objective pedotransfer function for a particular soil is

$$\mathbf{Y}_0 = \mathbf{T}_0\boldsymbol{\beta} + \mathbf{h}(\mathbf{X}_0) + \boldsymbol{\epsilon}(\mathbf{D}_0)$$

where

  - $\mathbf{h}(\mathbf{X}_0)$ is a two-dimensional spatial process that controls the smoothness of the contribution of $\mathbf{X}$
  - $\boldsymbol{\epsilon}(\mathbf{D}_0)$ is an error process that
    - accounts for the dependence in LL and $\triangle$ for a particular depth and
    - accounts for dependence across depths (one-dimensional spatial process).

A Multi-objective Pedotransfer Function

- Letting
  $\mathbf{Y} = \log\,[\mathsf{LL}_{11} \;\cdots\; \mathsf{LL}_{1d_1}\; \mathsf{LL}_{21} \;\cdots\; \mathsf{LL}_{Nd_N}\; \triangle_{11} \;\cdots\; \triangle_{1d_1}\; \triangle_{21} \;\cdots\; \triangle_{Nd_N}\,]'$,
  then $\mathbf{Y}$ is multivariate normal with

$$E[\mathbf{Y}] = \mathbf{T}\boldsymbol{\beta} \qquad \mathrm{Var}[\mathbf{Y}] = \boldsymbol{\Sigma}_{\mathbf{h}} + \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}$$

$$\begin{aligned} \boldsymbol{\Sigma}_{\mathbf{h}} &= \begin{bmatrix} \rho_1 & 0 \\ 0 & \rho_2 \end{bmatrix} \otimes \mathbf{K} \\ \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}} &= \mathbf{S} \otimes \mathbf{R}. \end{aligned}$$

with

  - $K_{ij} = k(\mathbf{X}_i, \mathbf{X}_j)$
  - $\mathbf{S}$ is the covariance of (LL, $\triangle$) at a fixed depth
  - $\mathbf{R}$ is the (spatial) covariance across depths

Covariance Structures

- The covariance function for h is the Matern family

$$C(d) = \sigma^2 \frac{2(\theta d/2)^\nu K_\nu(\theta d)}{\Gamma(\nu)}$$

where σ^2 is a scale parameter, θ represents the range, ν controls the smoothness.

  - $\sigma^2 = 1$ (the ρ controls the variances), $\nu = 1$, and θ is taken to be approximately the range of the data.
  - These choices represent a covariance structure that is consistent with the thin-plate spline estimator (large range, more smoothness).
  - Analogous to fixing the kernel and estimating the bandwidth with kernel estimators.

Covariance Structures

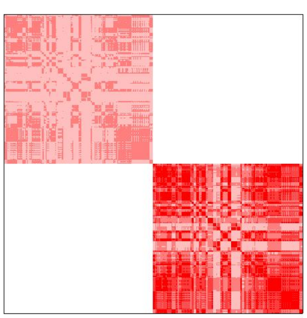
- The covariance function across depths is exponential

$$C(d) = \sigma^2 \exp(-d/\theta)$$

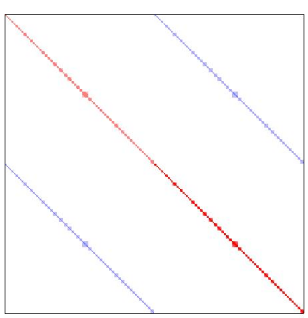
where again σ^2 is a scale parameter and θ represents the range.

  - The parameters $\sigma^2 = 1$ (the matrix $\mathbf{S}$ controls the variances) and θ is estimated from the data.
  - The multiple realizations of the soils allow for improved ability to estimate both scale and range parameters.

Covariance Structures



$$\mathbf{\Sigma_h} = \begin{bmatrix} \rho_1 & 0 \\ 0 & \rho_2 \end{bmatrix} \otimes \mathbf{K}$$



$$\mathbf{\Sigma_\epsilon} = \mathbf{S} \otimes \mathbf{R}$$

Spatial Smoothing

- Write

$$\begin{aligned}\boldsymbol{\Sigma}_{\mathbf{h}} + \boldsymbol{\Sigma}_{\epsilon} &= \begin{bmatrix} \rho_1 & 0 \\ 0 & \rho_2 \end{bmatrix} \otimes \mathbf{K} + \begin{bmatrix} s_{11} & s_{12} \\ s_{12} & s_{22} \end{bmatrix} \otimes \mathbf{R} \\ &= s_{11} \left[\begin{bmatrix} \eta_1 & 0 \\ 0 & \eta_2 \end{bmatrix} \otimes \mathbf{K} + \begin{bmatrix} 1 & v_{12} \\ v_{12} & v_{22} \end{bmatrix} \otimes \mathbf{R} \right] \\ &= s_{11} \boldsymbol{\Omega} \end{aligned}$$

- The amount of smoothing is due to the relative contributions of the variance components, i.e. η_1 and η_2.

- Different degrees of smoothing are allowed for LL and Δ.

- Also, this construction allows for different degrees of variation in the error terms for LL and the Δ variables.

The Estimator

- The model suggests an estimator of the form

$$\widehat{\mathbf{Y}}_0 = \mathbf{T}_0 \widehat{\boldsymbol{\beta}} + \mathbf{K}_0' \widehat{\boldsymbol{\delta}},$$

where

$$\mathbf{K}_0' = \begin{bmatrix} \eta_1 & 0 \\ 0 & \eta_2 \end{bmatrix} \otimes \mathbf{K}.$$

- To fit the model, we must estimate:
  - η_1, η_2 and s_{11}
  - $\boldsymbol{\beta}$, $\boldsymbol{\delta}$
  - $\mathbf{R}$ and the other entries of $\mathbf{S}$

REML

- Take the QR decomposition of $\mathbf{T}$

$$\mathbf{T} = [\mathbf{Q}_1 \ \ \mathbf{Q}_2] \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}.$$

- Then $\mathbf{Q}_2'\mathbf{Y}$ has zero mean and covariance matrix given by

$$\mathbf{Q}_2'(\boldsymbol{\Sigma}_{\mathbf{h}} + \boldsymbol{\Sigma}_{\epsilon})\mathbf{Q}_2.$$

- Maximize (numerically) the likelihood based on $\mathbf{Q}_2'\mathbf{Y}$ which is only a function of the covariance parameters.

- Estimates of $\boldsymbol{\beta}$ and $\boldsymbol{\delta}$ follow directly

$$\widehat{\boldsymbol{\beta}} = (\mathbf{T}'\widehat{\boldsymbol{\Omega}}^{-1}\mathbf{T})^{-1}\mathbf{T}'\widehat{\boldsymbol{\Omega}}^{-1}\mathbf{Y} \qquad \widehat{\boldsymbol{\delta}} = \widehat{\boldsymbol{\Omega}}^{-1}(\mathbf{Y} - \mathbf{T}\widehat{\boldsymbol{\beta}}).$$

An Iterative Approach

0. Initialize: compute $\mathbf{K}$ and set $\mathbf{S} = \mathbf{I}$ and $\mathbf{R} = \mathbf{I}$.

1. Estimate η_1 and η_2 (and s_{11}) via a simplified type of REML (grid search).

2. Then
$$\hat{\boldsymbol{\beta}} = (\mathbf{T}'\hat{\boldsymbol{\Omega}}^{-1}\mathbf{T})^{-1}\mathbf{T}^{-1}\hat{\boldsymbol{\Omega}}^{-1}\mathbf{Y} \qquad \hat{\boldsymbol{\delta}} = \hat{\boldsymbol{\Omega}}^{-1}(\mathbf{Y} - \mathbf{T}\hat{\boldsymbol{\beta}}).$$

3. Compute residuals and
   a. Update $\mathbf{S}$ ($\mathbf{R}$ fixed) – closed form solution.
   b. Update $\mathbf{R}$ ($\mathbf{S}$ fixed) – grid search for θ.

4. Repeat items 1-3 until convergence.

An Iterative Approach

- Let $\mathbf{Y} = \boldsymbol{\mu} + \mathbf{h} + \boldsymbol{\epsilon}$, where $\mathbf{h}$ and $\boldsymbol{\epsilon}$ are independent Gaussian random variables; the conditional distribution of $\mathbf{Y} - \boldsymbol{\mu} - \mathbf{h}$ given $\mathbf{h}$ is a zero mean Gaussian with covariance matrix $\boldsymbol{\epsilon}$.

- Thus, the log-likelihood associated with the residuals is given by
$$-\frac{n}{2}|\mathbf{S}| - |\mathbf{R}| - \text{vec}(\mathbf{U})'(\mathbf{S}^{-1} \otimes \mathbf{R}^{-1})\,\text{vec}(\mathbf{U})$$

- The quadratic form can be written as
$$\text{tr}(\mathbf{S}^{-1}\sum_i\sum_j r^{ij}\mathbf{u}_j\mathbf{u}_i')$$
where r^{ij} is the ijth element of $\mathbf{R}^{-1}$ and $\mathbf{u}_i$ is the bivariate, unstacked residual for the ith observation.

An Iterative Approach

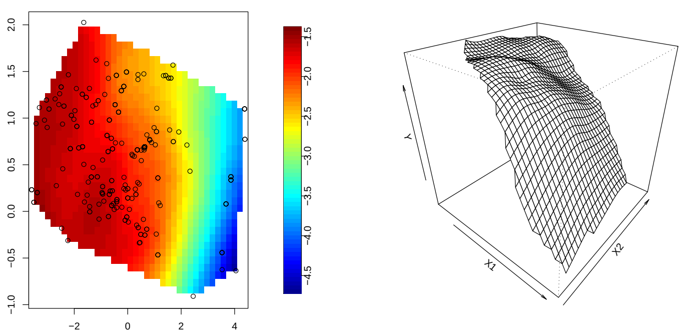
- An update for $\mathbf{S}$ can be written as
$$\begin{aligned}\hat{\mathbf{S}} &= \frac{1}{n}\sum_i\sum_j r^{ij}\mathbf{u}_j\mathbf{u}_i' \\ &= \frac{1}{n}\mathbf{U}'\mathbf{R}^{-1}\mathbf{U}\end{aligned}$$
where $\mathbf{U}$ is the $n \times 2$ matrix of unstacked residuals.

- Again, a simple grid search for θ is used to obtain a new value for $\mathbf{R}$.

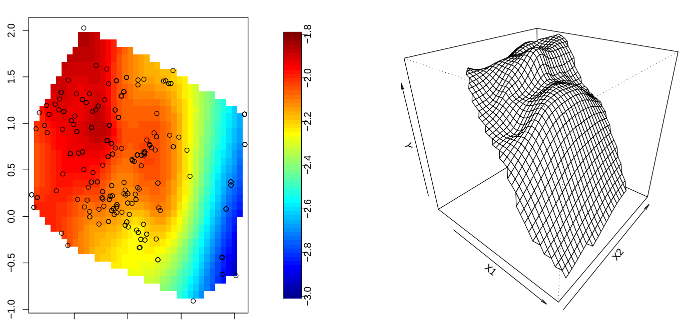
Parameter Estimates

	η_1	η_2	S_{11}	S_{22}	S_{12}	θ
REML	5.84	1.66	0.0765	0.0483	-0.0222	134.6
Iterative	5.74	2.21	0.0697	0.0445	-0.0217	144.2

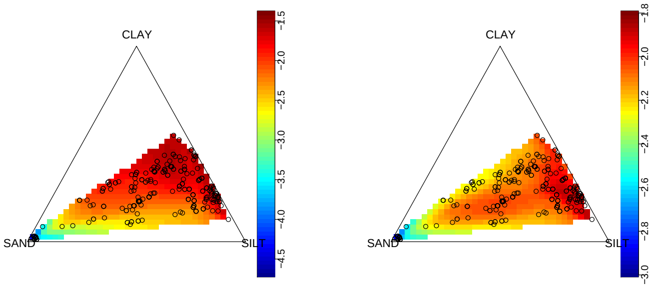
Soil Composition and LL



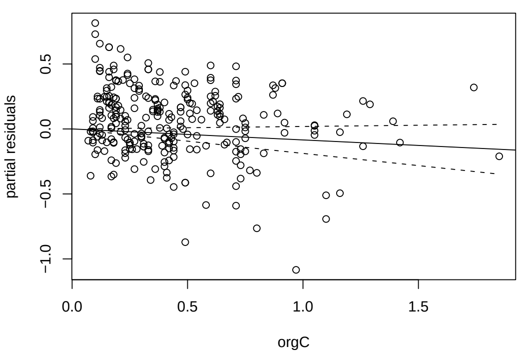
Soil Composition and $\triangle$



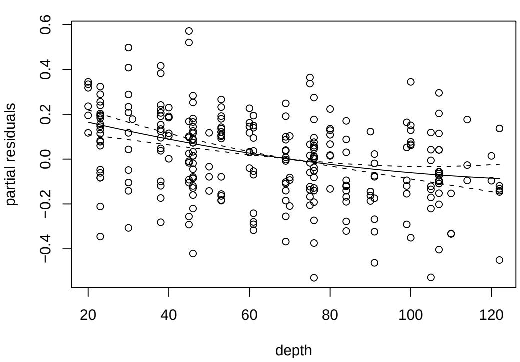
Soil Composition and LL/Δ



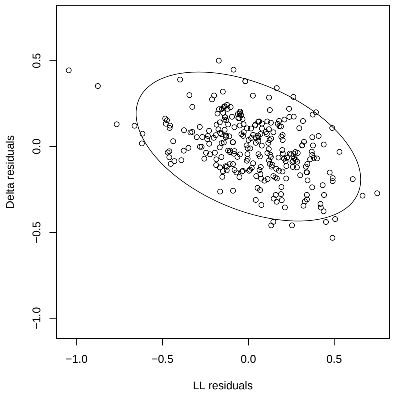
Organic Carbon and LL



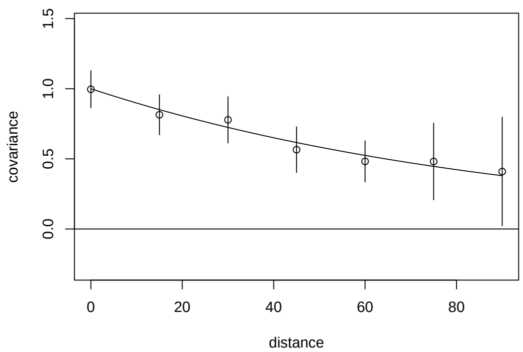
Depth and Δ



Residuals (Within Depth)



Spatial Covariance Across Depth



Prediction Error

- The real benefit of considering our estimator as a spatial process is in the interpretation with respect to the uncertainty of the estimator:
  - The thin-plate spline is a biased estimator with uncorrelated error; not easy to quantify the bias (interpolation error and smoothing error).
  - The spatial process estimator is unbiased, but with correlated error; more complicated error structure but conceptually straightforward to work with.

Prediction Error

- The estimator can be written as

$$\begin{aligned}\widehat{\mathbf{Y}}_0 &= \mathbf{T}_0\widehat{\boldsymbol\beta} + \mathbf{K}_0'\widehat{\boldsymbol\delta} \\ &= \mathbf{A}_0\mathbf{Y},\end{aligned}$$

where

$$\begin{aligned}\mathbf{A}_0 &= \mathbf{T}_0(\mathbf{T}'\widehat{\boldsymbol\Omega}^{-1}\mathbf{T})^{-1}\mathbf{T}'\widehat{\boldsymbol\Omega}^{-1} \\ &+ \mathbf{K}_0\left(\widehat{\boldsymbol\Omega}^{-1} - \widehat{\boldsymbol\Omega}^{-1}\mathbf{T}(\mathbf{T}'\widehat{\boldsymbol\Omega}^{-1}\mathbf{T})^{-1}\mathbf{T}'\widehat{\boldsymbol\Omega}^{-1}\right).\end{aligned}$$

Prediction Error

- Hence,

$$\begin{aligned}\mathrm{Var}(\mathbf{Y}_0 - \widehat{\mathbf{Y}}_0) &= \mathrm{Var}(\mathbf{Y}_0 - \mathbf{A}_0\mathbf{Y}) \\ &= \mathrm{Var}(\mathbf{Y}_0) + \mathbf{A}_0\mathrm{Var}(\mathbf{Y})\mathbf{A}_0' - 2\mathbf{A}_0\mathrm{Cov}(\mathbf{Y},\mathbf{Y}_0).\end{aligned}$$

  - $\mathrm{Var}(\mathbf{Y}_0)$ and $\mathrm{Var}(\mathbf{Y})$ are computed by plugging in parameters estimates for $\boldsymbol\Sigma_{\mathbf{h}}$ and $\boldsymbol\Sigma_{\epsilon}$.
  - The covariance between $\mathbf{Y}_0$ and $\mathbf{Y}$ comes from **h** and is based on the distance between the transformed composition data.

Generation of Soil Profiles

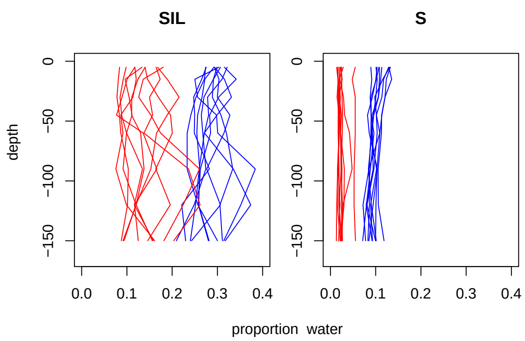
- Simulations of $\log LL$ and $\log \Delta$ were generated from a multivariate normal with mean $\mathbf{A}_0\mathbf{Y}$ and variance given by the prediction error.

- We use an average soil composition profile computed from the data and assumed to constant across all depths,

$$D = \{5, 15, 30, 45, 60, 90, 120, 150\}.$$

- Represent a draw from the posterior distribution of soil water profiles based on the estimated mean and covariance structure and the uncertainty gleaned from the data.

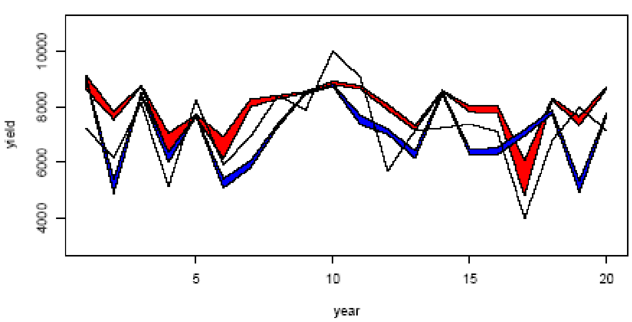
Generation of Soil Profiles



Application: Crop Models

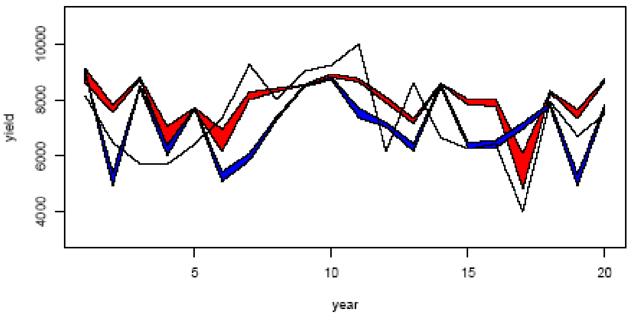
- Two soils (SIL, S)
  - Given the soil composition, organic carbon, depth, etc., 100 soil profiles (LL, DUL) were generated.
- Twenty years of weather (solar radiation, temperature min/max, and precipitation).
- Yield output generated from the CERES-Maize crop model.

Crop Yields



- SIL (red), S (blue), total annual precipitation (solid line)

Crop Yields



- SIL (red), S (blue), average annual temperature (solid line)

The `multiKrig` Class

- Create a multivariate spatial regression within the univariate `Krig` framework:

$$\mathbf{Y} = \mathbf{T}\boldsymbol{\beta} + \mathsf{h} + \epsilon$$

$$\mathbf{Y} = \begin{bmatrix} \mathbf{Y}_1 \\ \vdots \\ \mathbf{Y}_p \end{bmatrix} \qquad \mathbf{T} = \mathbf{I}_p \otimes \mathbf{X} \qquad \mathsf{h} = \begin{bmatrix} \mathsf{h}_1 \\ \vdots \\ \mathsf{h}_p \end{bmatrix} \qquad \epsilon = \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_p \end{bmatrix}$$

The `multiKrig` Class

- Create a multivariate spatial regression within the univariate `Krig` framework:

$$\mathbf{Y} = \mathbf{T}\boldsymbol{\beta} + \mathsf{h} + \epsilon$$

$$\begin{aligned} \mathrm{E}[\mathbf{Y}] &= \mathbf{T}\boldsymbol{\beta} \\ \mathrm{Var}[\mathbf{Y}] &= \Sigma_h + \Sigma_\epsilon \\ &= \begin{bmatrix} \rho_1 & & \\ & \ddots & \\ & & \rho_p \end{bmatrix} \otimes \mathbf{V}(\theta) + \begin{bmatrix} s_{11} & s_{12} & \cdots & s_{1p} \\ s_{21} & s_{22} & \cdots & s_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ s_{p1} & s_{p2} & \cdots & s_{pp} \end{bmatrix} \otimes \mathbf{I}_n \\ &= \rho_1 \mathbf{R} \otimes \mathbf{V}(\theta) + s_{11} \mathsf{S} \otimes \mathbf{I}_n \\ &= \rho_1 \left(\mathbf{R} \otimes \mathbf{V}(\theta) + \lambda \mathsf{S} \otimes \mathbf{I}_n \right) \end{aligned}$$

The `multiKrig` Class

- Create a multivariate spatial regression within the univariate `Krig` framework:

$$\mathbf{Y} = \mathbf{T}\boldsymbol{\beta} + \mathbf{h} + \epsilon$$

$$\begin{aligned} \mathsf{E}[\mathbf{Y}] &= \mathbf{T}\boldsymbol{\beta} \\ \mathsf{Var}[\mathbf{Y}] &= \boldsymbol{\Sigma}_h + \boldsymbol{\Sigma}_\epsilon \\ &= \rho_1 \left(\mathbf{R} \otimes \mathbf{V}(\theta) + \lambda \mathsf{S} \otimes \mathbf{I}_n \right) \end{aligned}$$

- Given S, $\mathbf{R}$, and θ, use `Krig` to estimate $\boldsymbol{\beta}$, ρ_1 and λ.

The `multiKrig` Class

- Issues:
  - Specifying `x`, `Y`, and `Z`
  - Mean function (`null.function`)
  - Covariance function (`cov.function`)
  - Error function (`wght.function`)
- Estimation (S, $\mathbf{R}$, and θ)

`Krig` Function

```
Krig  <-  function (x, Y, Z,
          null.function = "Krig.null.function",
          cov.function = "stationary.cov",
          wght.function = NULL,
          null.args = NULL, cov.args = NULL, wght.args = NULL)
```

- `x` is an $n \times q$ matrix of spatial locations
- `Y` is a n-vector of observations observations
- `Z` is a $n \times q$ matrix of additional covariates

multiKrig Function

```
multiKrig  <-  function(s,Y,Z,
               cov.function="multi.cov",cov.args=NULL,
               wght.function="multi.wght",wght.args=NULL)
```

- s is an $n \times q$ matrix of spatial locations
- Y is an $n \times p$ matrix of observations
- Z is either:
  - a $n \times q$ matrix of additional covariates, or
  - a list of $n \times q_i$ matrices of additional covariates

multiKrig Function

```
multiKrig  <-  function(s,Y,Z,
               cov.function="multi.cov",cov.args=NULL,
               wght.function="multi.wght",wght.args=NULL){
               ⋮
               d <- ncol(Y)
               n <- nrow(Y)
               Y <- c(Y)
               ⋮
               x <- expand.grid(1:n,1:d)
               nZ <- kronecker(diag(d),cbind(s,Z))
               ⋮
               obj  <-  Krig(x=x,Y=c(Y),Z=nZ,method="REML",
                         null.function="multi.null",
                         cov.function=cov.function,cov.args=cov.args,
                         wght.function=wght.function,wght.args=wght.args)
               ⋮
               }
```


- Y <- c(Y)

$$\mathbf{Y} = \begin{bmatrix} \mathbf{Y}_1 \\ \vdots \\ \mathbf{Y}_p \end{bmatrix}$$


```
multiKrig  <-  function(s,Y,Z,
               cov.function="multi.cov",cov.args=NULL,
               wght.function="multi.wght",wght.args=NULL){
               ⋮
               d <- ncol(Y)
               n <- nrow(Y)
               Y <- c(Y)
               ⋮
               x <- expand.grid(1:n,1:d)
               nZ <- kronecker(diag(d),cbind(s,Z))
               ⋮
               obj  <-  Krig(x=x,Y=c(Y),Z=nZ,method="REML",
                         null.function="multi.null",
                         cov.function=cov.function,cov.args=cov.args,
                         wght.function=wght.function,wght.args=wght.args)
               ⋮
               }
```

- x <- expand.grid(1:n,1:d) $\quad \mathbf{x} = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ \vdots & \\ n & 1 \\ 1 & 2 \\ \vdots & \\ n & p \end{bmatrix}$

```
multiKrig   <-   function(s,Y,Z,
                 cov.function="multi.cov",cov.args=NULL,
                 wght.function="multi.wght",wght.args=NULL){
                 ⋮
                 d <- ncol(Y)
                 n <- nrow(Y)
                 Y <- c(Y)
                 ⋮
                 x <- expand.grid(1:n,1:d)
                 nZ <- kronecker(diag(d),cbind(s,Z))
                 ⋮
                 obj   <-   Krig(x=x,Y=c(Y),Z=nZ,method="REML",
                            null.function="multi.null",
                            cov.function=cov.function,cov.args=cov.args,
                            wght.function=wght.function,wght.args=wght.args)
                 ⋮
                 }
```


- nZ <- kronecker(diag(d),cbind(s,Z)) $\quad \mathbf{Z} = \begin{bmatrix} \mathbf{s}\,\mathbf{Z} & & \\ & \ddots & \\ & & \mathbf{s}\,\mathbf{Z} \end{bmatrix}$


```
multiKrig   <-   function(s,Y,Z,
                 cov.function="multi.cov",cov.args=NULL,
                 wght.function="multi.wght",wght.args=NULL){
                 ⋮
                 d <- ncol(Y)
                 n <- nrow(Y)
                 Y <- c(Y)
                 ⋮
                 x <- expand.grid(1:n,1:d)
                 nZ <- kronecker(diag(d),cbind(s,Z))
                 ⋮
                 obj   <-   Krig(x=x,Y=c(Y),Z=nZ,method="REML",
                            null.function="multi.null",
                            cov.function=cov.function,cov.args=cov.args,
                            wght.function=wght.function,wght.args=wght.args)
                 ⋮
                 }
```



```
multi.null   <-   function(x,Z=NULL,drop.Z=FALSE){
                  data <- data.frame(a=as.factor(x[,2]))
                  X <- model.matrix(~a,data=data,contrasts=list(a="contr.treatment"))
                  ⋮
                  return(cbind(X,Z))
                  }
```

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & \mathbf{s}\,\mathbf{Z} & 0 & 0 \\ 1 & 1 & 0 & 0 & \mathbf{s}\,\mathbf{Z} & 0 \\ 1 & 0 & 1 & 0 & 0 & \mathbf{s}\,\mathbf{Z} \end{bmatrix}$$

```
multiKrig   <-   function(s,Y,Z,
                 cov.function="multi.cov",cov.args=NULL,
                 wght.function="multi.wght",wght.args=NULL){
                 ⋮
                 obj   <-   Krig(x=x,Y=c(Y),Z=nZ,method="REML",
                            null.function="multi.null",
                            cov.function=cov.function,cov.args=cov.args,
                            wght.function=wght.function,wght.args=wght.args)
                 ⋮
                 }
```

Covariance Function

- Issue: **x** is now a matrix of indices.
- Solution: pass the spatial locations as an argument to the covariance function.


```
multiKrig   <-   function(s,Y,Z,
                 cov.function="multi.cov",cov.args=NULL,
                 wght.function="multi.wght",wght.args=NULL){
                 ⋮
                 cov.args$s <- s
                 obj   <-   Krig(x=x,Y=c(Y),Z=nZ,method="REML",
                            null.function="multi.null",
                            cov.function=cov.function,cov.args=cov.args,
                            wght.function=wght.function,wght.args=wght.args)
                 ⋮
                 }

multi.cov   <-   function(x1,x2,marginal=FALSE,C=NA,s,theta,rho,smoothness){
                 ⋮
                 ind <- unique(x1[,2])
                 temp   <-   stationary.cov(s[x1[,1][x1[,2]==1],],
                             s[x2[,1][x2[,2]==1],],
                             Covariance="Matern",theta=theta,smoothness=smoothness)
                 if   (length(ind)>1){
                      for   (i in 2:length(ind)){
                            temp2   <-   rho[i-1]*stationary.cov(s[x1[,1][x1[,2]==i],],
                                         s[x2[,1][x2[,2]==i],],
                                         Covariance="Matern",theta=theta,smoothness=smoothness)
                            d1 <- dim(temp)
                            d2 <- dim(temp2)
                            temp   <-   rbind(cbind(temp,matrix(0,d1[1],d2[2])),
                                        cbind(matrix(0,d1[1],d2[2]),temp2))
                      }
                 }
                 ⋮
                 return(temp)
                 }
```


Weight Function

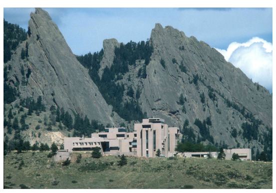
```
multiKrig   <-   function(s,Y,Z,
                 cov.function="multi.cov",cov.args=NULL,
                 wght.function="multi.wght",wght.args=NULL){
                 ⋮
                 obj   <-   Krig(x=x,Y=c(Y),Z=nZ,method="REML",
                            null.function="multi.null",
                            cov.function=cov.function,cov.args=cov.args,
                            wght.function=wght.function,wght.args=wght.args)
                 ⋮
                 }

multi.wght   <-   function(x,sp){
                  n <- length(unique(x[,1]))
                  ⋮
                  return(kronecker(solve(S),diag(n)))
                  }
```

Estimation

- `Krig` will estimate $\boldsymbol{\beta}$, ρ_1 and λ.
  - REML
  - GCV (not quite there...)
- How to estimate S, $\mathbf{R}$, and θ?

Thanks!



ssain@ucar.edu
www.image.ucar.edu/∼ssain

- Sain, S.R., Mearns, L., Shrikant, J., and Nychka, D. (2006), "A multivariate spatial model for soil water profiles," *Journal of Agricultural, Biological, and Environmental Statistics*, **11**, 462-480.