How fields works

Douglas Nychka,
www.image.ucar.edu/~nychka

• Krig function and basic operations
• Covariance functions
• Tps and mKrig

Supported by the National Science Foundation           fields Short Course, Oct 2008
Basic arguments are:

\( \mathbf{x} \): the matrix of locations (each row is a location)

\( \mathbf{y} \): the vector of observations.

\( \mathbf{z} \): possible covariates.

\( m \): Sets spatial drift to \((m - 1)\) degree polynomial. Default is 2 – linear drift.

\( \lambda \): The smoothing parameter. Default is to find this by GCV.

\( \text{cov.function} \): name of covariance function

  Default is \text{stationary.cov}

... Arguments to covariance function.

  Usually need to put in something.
For example

\[
\text{object} <- \text{Krig( ozone2$lon.lat, ozone2$y[16,], theta=350)}
\]

or

\[
\text{object} <- \text{Tps( ozone2$lon.lat, ozone2$y[16,])}
\]

**The returned object**

The returned object is a list with the information needed to evaluate the function and compute prediction errors.

- `\text{predict( object, xnew)}` Predictions at new locations in `xnew`.
- `\text{plot( object)}` Diagnostic plots of fit.
- `\text{summary( object)}` Summary of fit.

Also, `\text{predict.se, predict.surface, predict.se.surface, surface}`
A model covariance function for Krig

This is a simple example of the exponential covariance.

Exp.simple.cov <-
 function(x1, x2, theta = 1, C = NA, marginal = FALSE) {
 if (is.na(C[1]) & !marginal) {
    return(exp(-rdist(x1, x2)/theta))
 }
 if (!is.na(C[1])) {
    return(exp(-rdist(x1, x2)/theta)%% C)
 }
 if (marginal) {
    return(rep(1, nrow(x1)))
 }
}

Always has three possible actions.
x1, x2, C and marginal are required.
Exp.simple.cov \((x_1, x_2, \theta = 1, C = \text{NA}, \text{marginal} = \text{FALSE})\)

Computes the cross covariance of the locations in \(x_1\) with those in \(x_2\) and either:
1. returns a matrix
or
2. multiplies the matrix by the vector/matrix \(C\).

\text{rdist} \ is \ a \ useful \ function \ that \ finds \ the \ Euclidean \ dis-
tance \ matrix \ between \ two \ sets \ of \ locations.

\text{theta} \ is \ the \ scale \ parameter \ for \ the \ exponential.
A more general stationary covariance function

\begin{verbatim}
stationary.cov ( x1, x2, Covariance = "Exponential", Distance = "rdist", Dist.args = NULL, theta = 1, C = NA, marginal = FALSE, ...)
\end{verbatim}

Fitting a Matern: \textit{shape} = 1.5, \textit{scale} = 350

object<- Krig( ozone2\$lon.lat, ozone2\$y[16,],
               Covariance="Matern",
               smoothness= 1.5, theta=350)

What about \(\lambda\)?

\texttt{lambda will still be found by GCV and is equivalent to finding the nugget variance by GCV.}
Tps – it is just a wrapper around Krig

```r
Tps (x, Y, m = NULL, p = NULL, scale.type = "range", ...) {
  x <- as.matrix(x)
  d <- ncol(x)
  if (is.null(p)) {
    if (is.null(m)) {
      m <- max(c(2, ceiling(d/2 + 0.1)))
    }
    p <- (2 * m - d)
  ....
  Krig( x, Y, cov.function = Rad.cov, m = m,
       scale.type = scale.type, GCV = TRUE,
       p = p, ...)
}
```

Returned object is just a specific call to Krig!
The depths of mKrig

an edited version and after setting up the data ...

Tmatrix <- fields.mkpoly(x, m)
tempM <- do.call(cov.function, c(cov.args, list(x1 = x, x2 = x)))
diag(tempM) <- (lambda/weights) + diag(tempM)

Mc <- do.call("chol", c(list(x = tempM), chol.args))
VT <- forwardsolve(Mc, x = Tmatrix, transpose = TRUE, upper.tri = TRUE)
qr.VT <- qr(VT)

d.coef <- qr.coef(qr.VT,
    forwardsolve(Mc, transpose = TRUE, y, upper.tri = TRUE) )

c.coef <- forwardsolve(Mc, transpose = TRUE,
    y - Tmatrix %*% d.coef, upper.tri = TRUE)
c.coef <- backsolve(Mc, c.coef)

.... now find residuals etc. and return object.
Some closing remarks

- Easy to get quick summaries, predictions and plots.
- Many comments throughout source code and consistent notation.
- Covariance function is very flexible
- Linear algebra can easily be overloaded to handle sparse matrices.