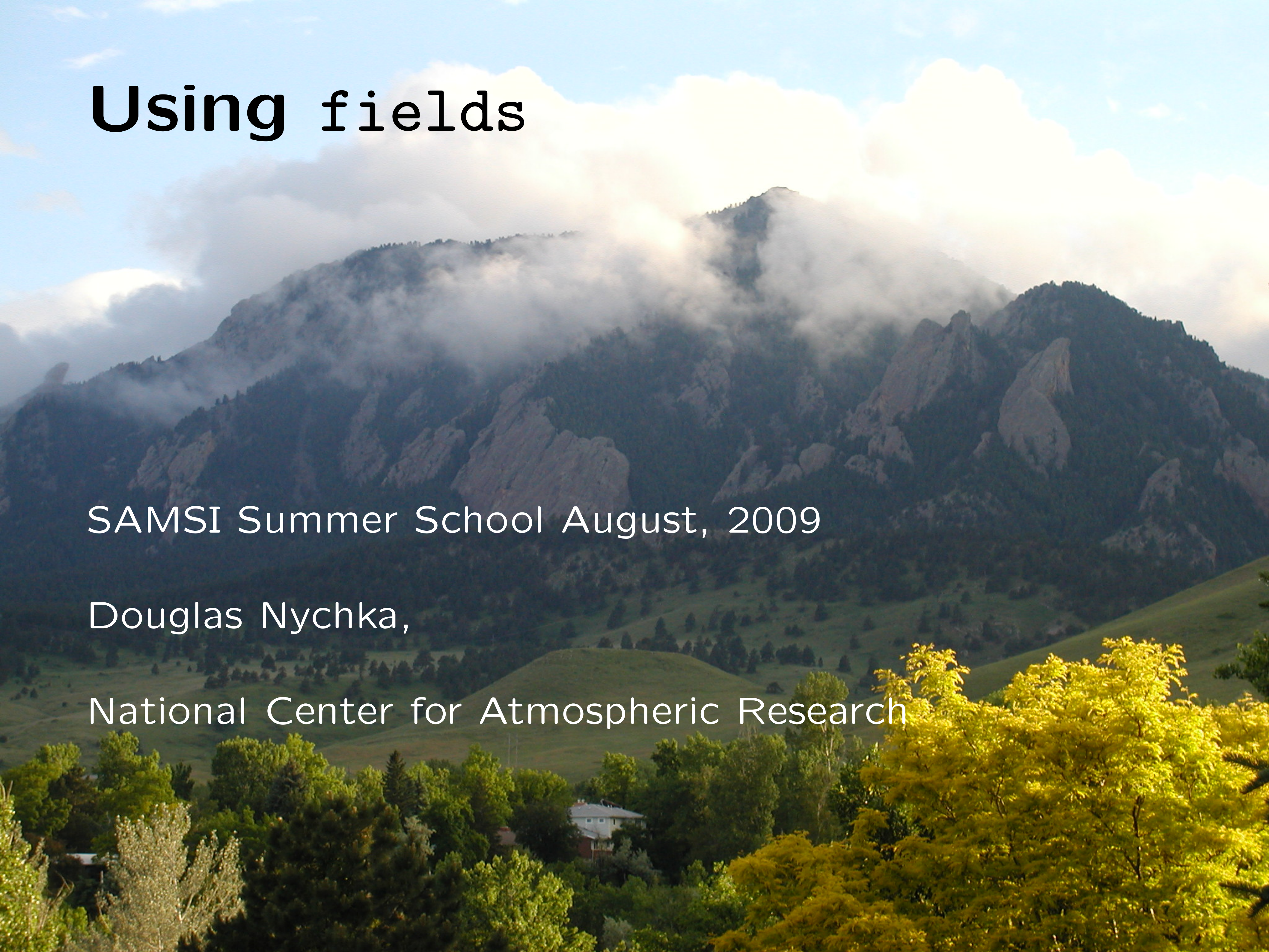

Using `fields`

SAMSI Summer School August, 2009

Douglas Nychka,

National Center for Atmospheric Research

Outline

- Krig function and basic operations

- Covariance functions

- Tps and mKrig

Supported by the National Science Foundation

`Krig`

Basic arguments are:

`x`: the matrix of locations (each row is a location)

`Y`: the vector of observations.

`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.

`cov.function`: name of covariance function
Default is `stationary.cov`

`...` Arguments to covariance function.
Usually need to put in something.

For example

```
object<- Krig(x,y, theta=2.0)

object<- Tps( x,y, m=3)

object<- Krig(x,y, theta=350, Distance="rdist.earth")
```

The returned object

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

`predict( object, xnew)` Predictions at new locations in `xnew`.

`plot( object)` Diagnostic plots of fit.

`summary( object)` Summary of fit.

Also, `predict.se` , `predict.surface`, `predict.se.surface`, `surface`

A simple 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 (x1, x2, theta = 1, C = NA,
         marginal = FALSE)
```

Computes the cross covariance of the locations in `x1` with those in `x2` and either:

- returns a matrix

or

- multiplies the matrix by the vector/matrix `C`.

`rdist` is a useful function that finds the Euclidean distance matrix between two sets of locations.

`theta` is the scale parameter for the exponential.

A stationary covariance function

```
stationary.cov ( x1, x2, Covariance = "Exponential",
           Distance = "rdist", Dist.args = NULL, theta = 1,
           C = NA, marginal = FALSE, ...)
```

Fitting a Matern: $\nu = 1.5$, $\theta = 350$ Using great circle distance.

```
  object<- Krig( x,y, Covariance="Matern",
                      nu= 1.5,   theta=350, Distance="rdist.earth")
```

What about λ?

`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

```
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.