Some SAR Processes

Short course Large spatial data.

create a 1D SAR weight matrix

Use a handy LatticeKrig function to do it. Note diagonal value muxst be greater than 2.

```
B<- LKDiag( c( -1, 3,-1), 10, full=TRUE)
print( B)
```

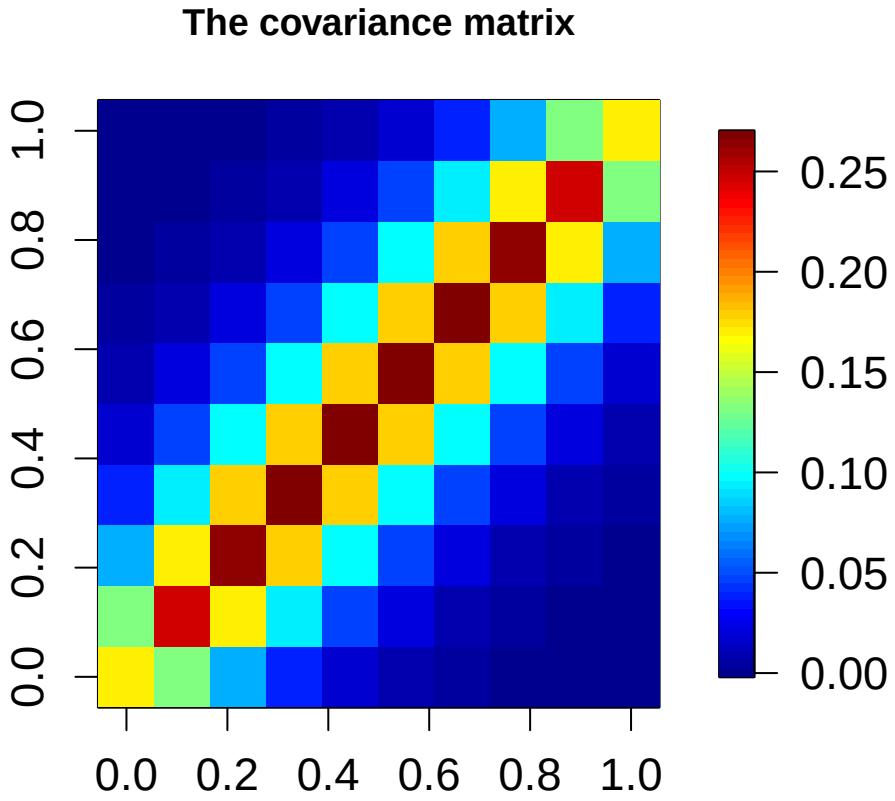
```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]    3   -1    0    0    0    0    0    0    0     0
##  [2,]   -1    3   -1    0    0    0    0    0    0     0
##  [3,]    0   -1    3   -1    0    0    0    0    0     0
##  [4,]    0    0   -1    3   -1    0    0    0    0     0
##  [5,]    0    0    0   -1    3   -1    0    0    0     0
##  [6,]    0    0    0    0   -1    3   -1    0    0     0
##  [7,]    0    0    0    0    0   -1    3   -1    0     0
##  [8,]    0    0    0    0    0    0   -1    3   -1     0
##  [9,]    0    0    0    0    0    0    0   -1    3    -1
## [10,]    0    0    0    0    0    0    0    0   -1     3
```

Now find precision matrix and the covariance matrix. The shapes are close to double exponential.

```
fields.style()
Q<- t( B) %*% B
print( Q)
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]   10   -6    1    0    0    0    0    0    0     0
##  [2,]   -6   11   -6    1    0    0    0    0    0     0
##  [3,]    1   -6   11   -6    1    0    0    0    0     0
##  [4,]    0    1   -6   11   -6    1    0    0    0     0
##  [5,]    0    0    1   -6   11   -6    1    0    0     0
##  [6,]    0    0    0    1   -6   11   -6    1    0     0
##  [7,]    0    0    0    0    1   -6   11   -6    1     0
##  [8,]    0    0    0    0    0    1   -6   11   -6     1
##  [9,]    0    0    0    0    0    0    1   -6   11    -6
## [10,]    0    0    0    0    0    0    0    1   -6    10
```

```
Omega<- solve( Q)
imagePlot( Omega)
title("The covariance matrix")
```



A 2D example

Now use LatticeKrig utilites to look at some SARs in 2D. This is for a 30X30 lattice. **NC.buffer=0** below means no extra grid points are added at the edges. This is not recommended as there are edge effects!

First define the mode with a setup function.

```
sDomain<- cbind( c(0,1), c( 0,1))
LKinfo<- LKrigSetup( sDomain, NC= 30, NC.buffer=0,  nlevel=1, a.wght=4.05)
#  print(LKinfo) for a summary of the model with its defautl choices.
```

Now use this specification to create some of the model components.

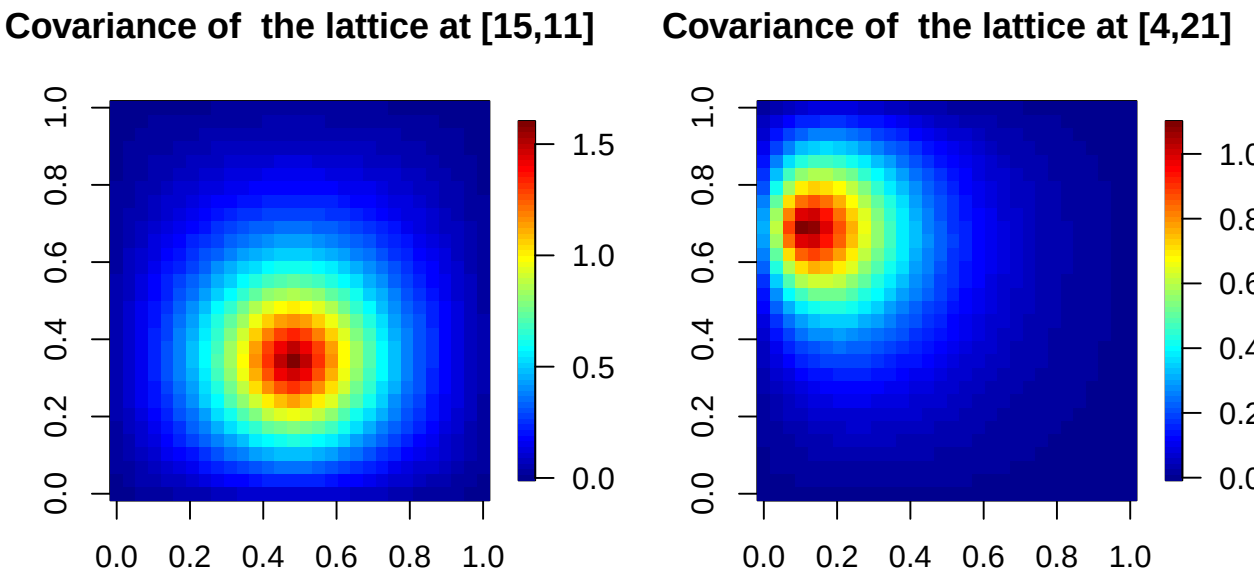
```
Qsparse<- LKrig.precision(LKinfo)
# covert the sparse format to a full one for plotting
Q<- spam2full( Qsparse)
# check
dim( Q)
```

```
## [1] 900 900
```

```
Omega<- solve( Q)
set.panel(1,2)
```

```
## plot window will lay out plots in a 1 by 2 matrix
```

```
par( pty="s")
#15 + 10*90
# pull off (15,10) and reshape to an image.
image.plot( matrix( Omega[,15+10*30], 30,30))
title("Covariance of  the lattice at [15,11]")
image.plot( matrix( Omega[,4+20*30], 30,30))
title("Covariance of  the lattice at [4,21]")
```




```
set.panel()
```

```
## plot window will lay out plots in a 1 by 1 matrix
```

Simulate some MRFs

How about 4? Note that the coefficient fields are big vectors. Have to reshape to look at them.

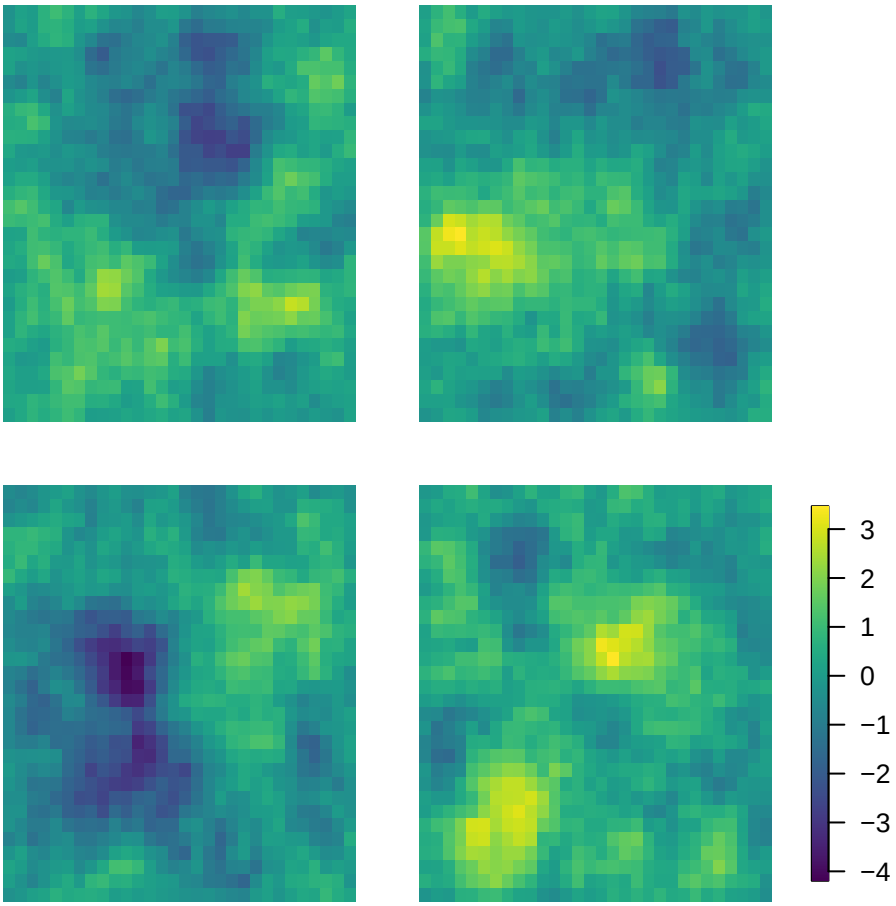
```
set.seed(123)
look <-  LKrig.sim( LKinfo=LKinfo, M=4 ,just.coefficients=TRUE )
```

Now plot them – this takes way more code than generating them!

```
# take a look
set.panel( 2,2)
```

```
## plot window will lay out plots in a 2 by 2 matrix
```

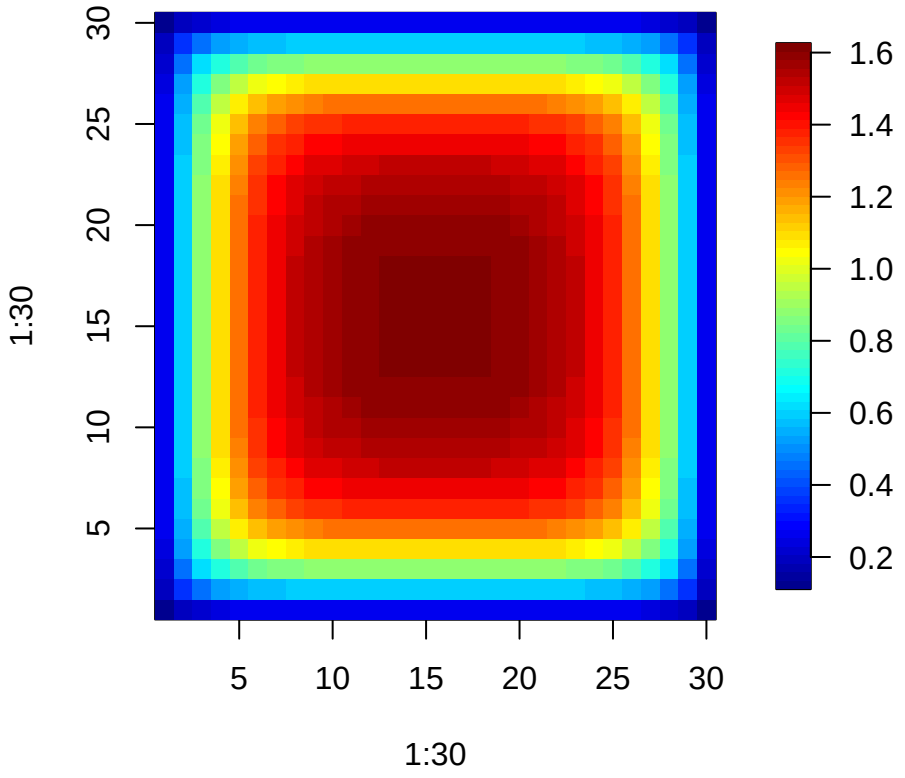
```
par( mar=c( 1,1,1,1), oma=c( 0,0,0,4))
zr<- range( c( look))
for ( k in 1:4){
  image( matrix( look[,k], 30,30), axes=FALSE,
              xlab="", ylab="", zlim= zr,
         col=viridis(256))
}
par( oma=c( 0,0,0,1))
image.plot( legend.only=TRUE, col=viridis(256), zlim=zr)
```



Exercises

1. How does the covariance at the 155 location change when **a.wght=4.01** ? How about **a.wght=10**?
2. Take a look at the process variances (**diag(Omega)**)

```
image.plot( 1:30, 1:30, matrix( diag( Omega),30,30))
```



You can see some edge effects.

How many rows and columns of the lattice should you ignore so that the variances are approximately constant? Say within about 90%? The LatticeKrig default is 5 on each side.