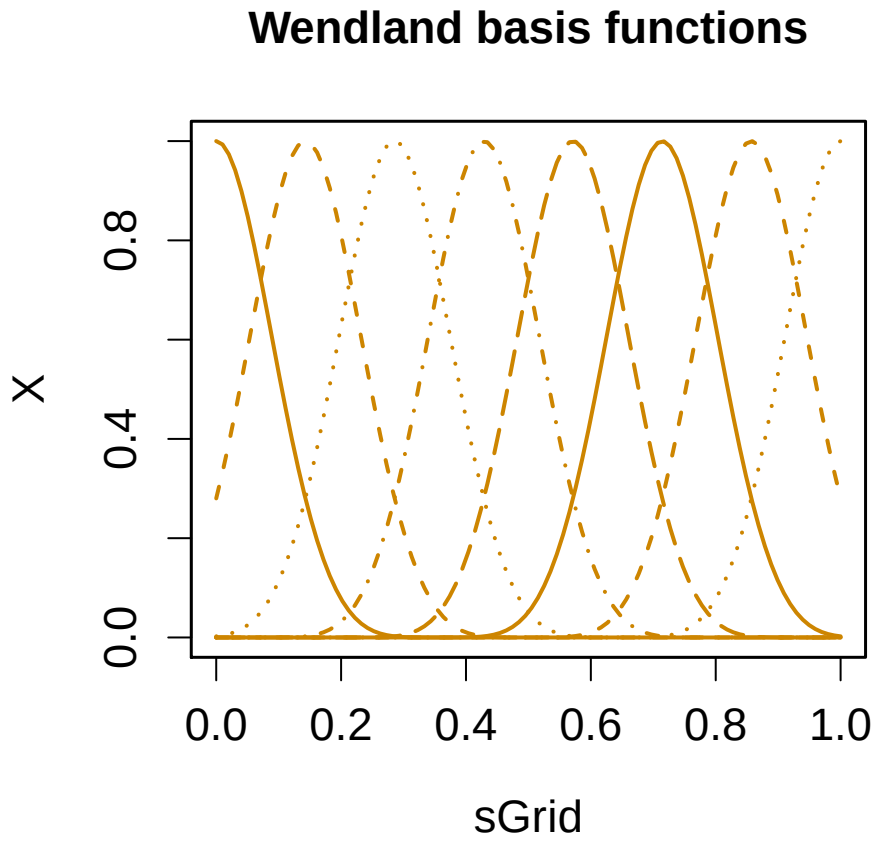
Day 2 Lab Part 2

ISI Short Course | June 1-3

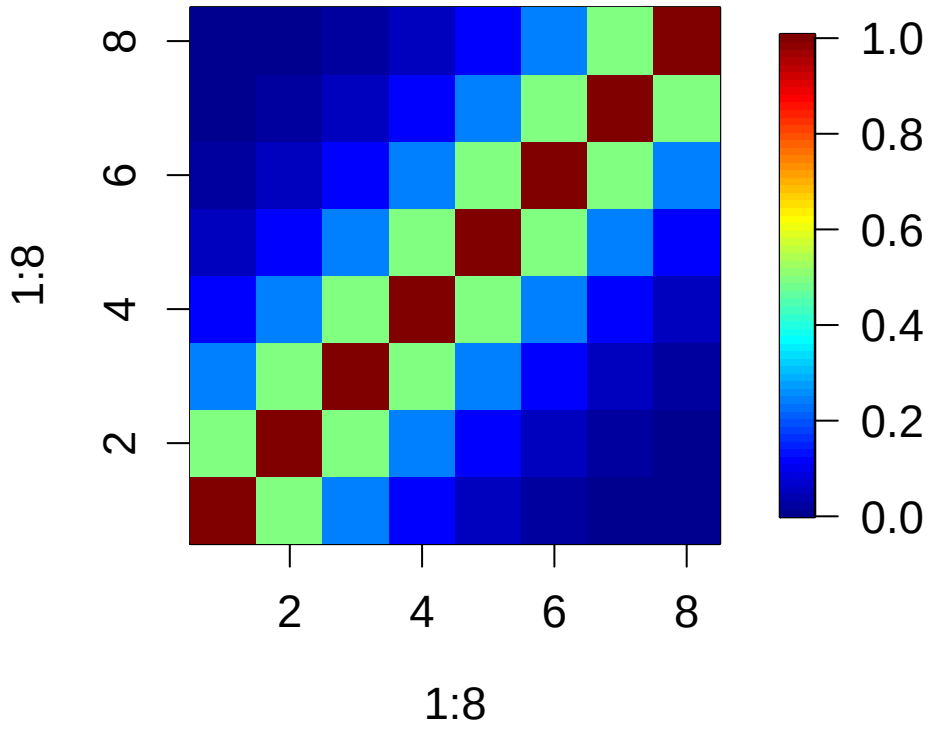
Exploring the implied covariance function in fixed rank Kriging

Recall Ω is the covariance among the coefficients. Take this to be exponential in this example.

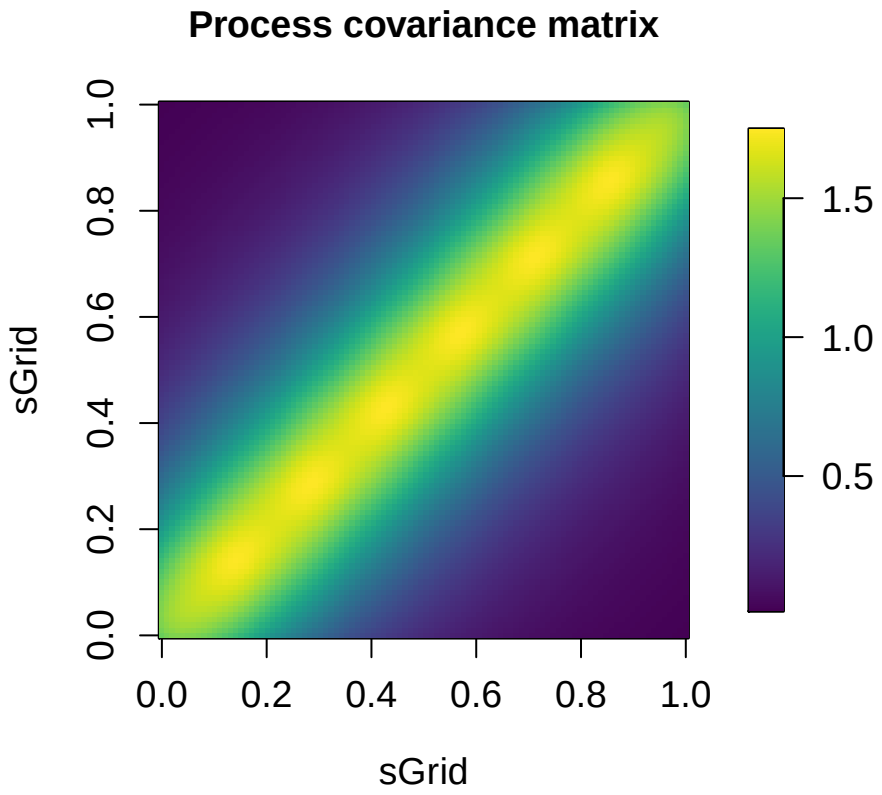
```
fields.style()
u<- seq( 0,1,length.out=8)
sGrid<- seq( 0,1,length.out= 100)

Omega<- Matern( rdist( u,u)/.2, smoothness=.5)
# smoothness .5 is also the negative exponential
X<- WendlandFunction( rdist( sGrid, u)/( (3*1.0)/8))

matplot( sGrid, X, type="l", col="orange3", lwd=2)
title("Wendland basis functions")
```



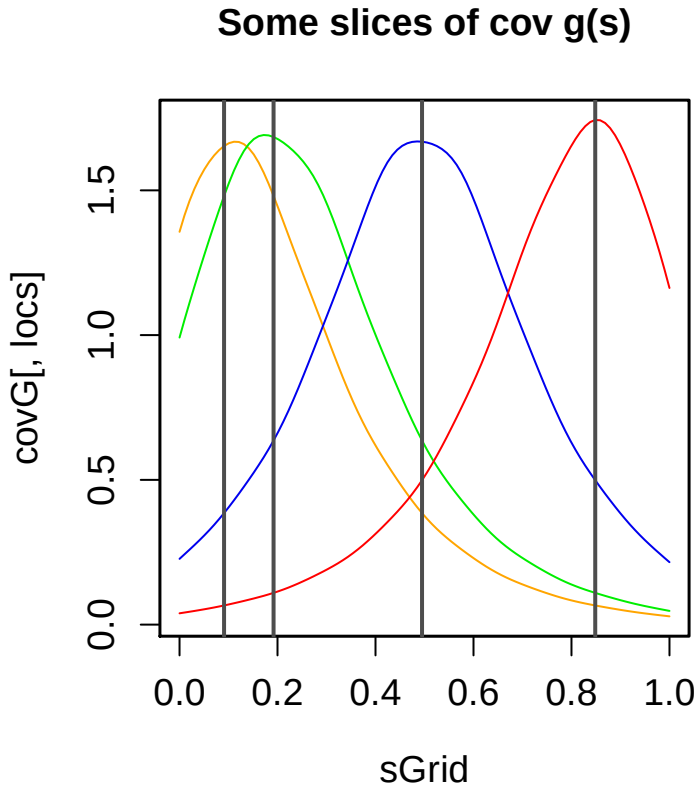

```
image.plot( 1:8,1:8, Omega)
title("Covariance of coefficients")
```




```
# covariance for the process
covG<- X%*%Omega%*%t(X)
image.plot(sGrid, sGrid,  covG, col=viridis(256) )
title("Process covariance matrix")
```



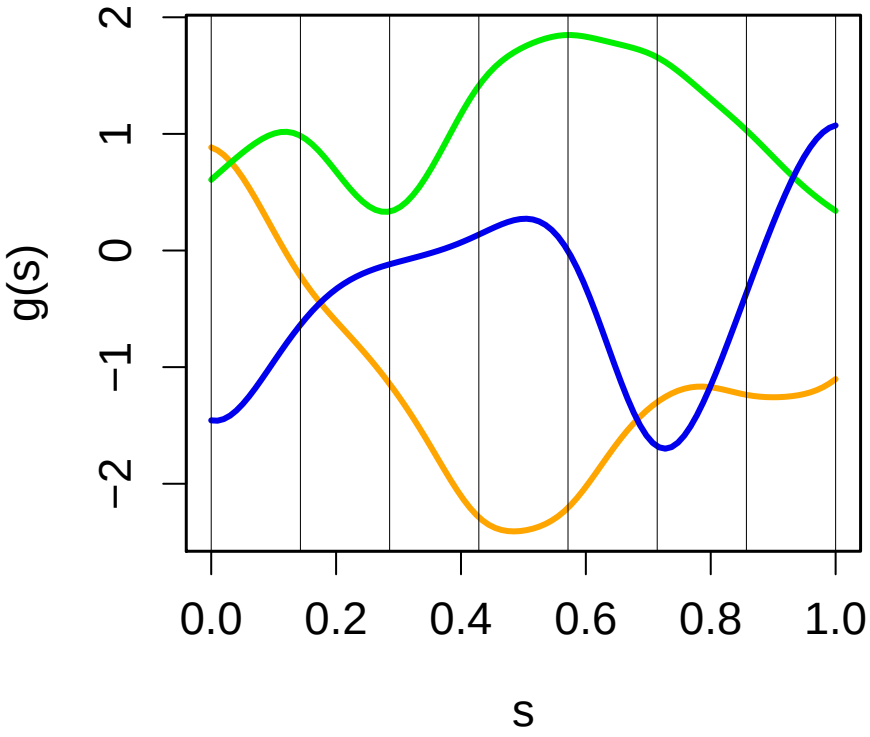

```
locs<- c( 10, 20, 50, 85)
matplot( sGrid, covG[,locs], type="l", lty=1)
xline( sGrid[locs], col="grey30", lwd=2)
title("Some slices of cov g(s)")
```



Simulation of the FRK process

Simulating 3 realizations (**M**) of g from this model. Here for efficient vectorized coding **gSim** is a matrix 100X3 where each column is a realization.

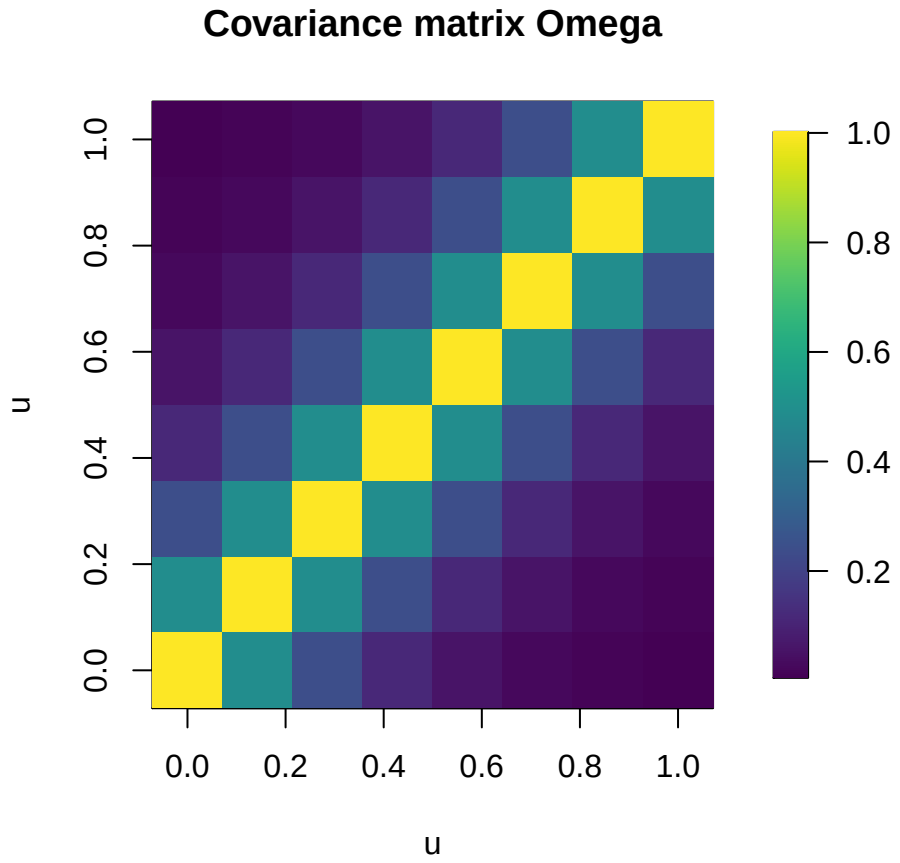
```
set.seed(2232)
B<- chol( Omega)
M<- 3
bigU<- matrix(rnorm( 8*M), 8,M )
c<- t(B)%*%bigU
gSim<- X%*%c
fields.style()
matplot( sGrid, gSim, type="l",
         lty=1, lwd=3, xlab="s", ylab= "g(s)")
# add knot locations for reference
xline( u, lwd=.5)
```



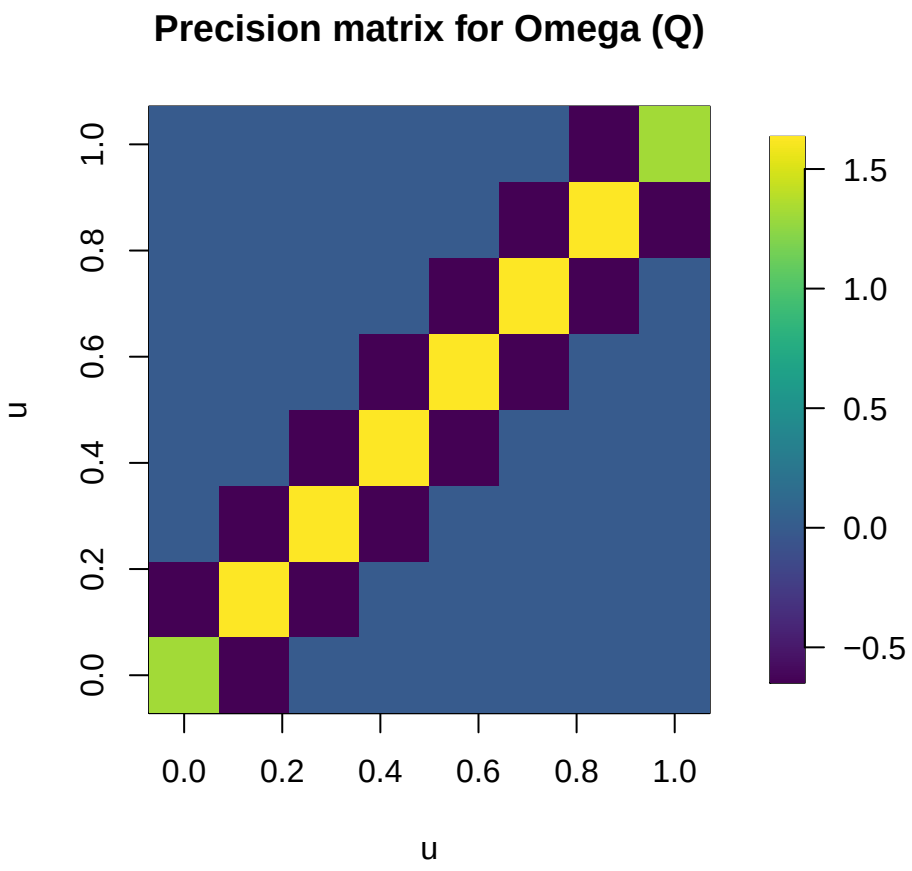
Precision matrix for coefficients

Investigating the precision matrix for the coefficients. For this exponetial there are many zeroes.

```
Q<- solve( Omega)
par( pty="s")
image.plot( u, u, Omega, col=viridis(256))
title("Covariance matrix Omega")
```




```
image.plot( u, u, Q, col=viridis(256))
title("Precision matrix for Omega (Q)")
```



Exercises

1. How does the sparseness of the precision matrix (**Q**) change when the smoothness is changed to 1.0 from .5?
2. Instead of 3 realizations generate 1000 (change 3 to 1000) and find the maximum value of the process for each of these (use the **apply** function across columns) Is the distrbution of the maxima normally distributed?