# Example 4: Standard Error of AUC

### Setting Up

First load the data and R packages you need into the R workspace.

```
load("PlayData_for_Examples.RData") ### load the data for the examples
source("LoadFunctions.R") #### Source the functions required to run the block bootstrap


#Install packages if required
if(!require(pROC)) { install.packages("pROC", repos = "http://cran.us.r-project.org"); require(pROC) }

library(pROC)
```

## Motivation

Imagine we want to know the standard error of the AUC (area under the ROC curve) of our model. Perhaps we have two models, and we want to see if variation in AUC is explained by sampling error or if it is due to the predictors in the model.

```
set.seed(42)

fit1 = glm (PresAbsresponse ~ temperature, data = dat, family="binomial")

auc(response=dat$PresAbsresponse, predictor= predict(fit1))
```

```
## Area under the curve: 0.7735
```

### Coding StatFunction argument

Define getAUC a function which will be the *StatFunction* argument to *BlockBootApply*.

```
GetAUC = function(dat){
fit1 = glm ("PresAbsresponse~temperature", data=dat, family="binomial")
AUC = auc(response=dat$PresAbsresponse, predictor= predict(fit1))
;
AUC
}
```

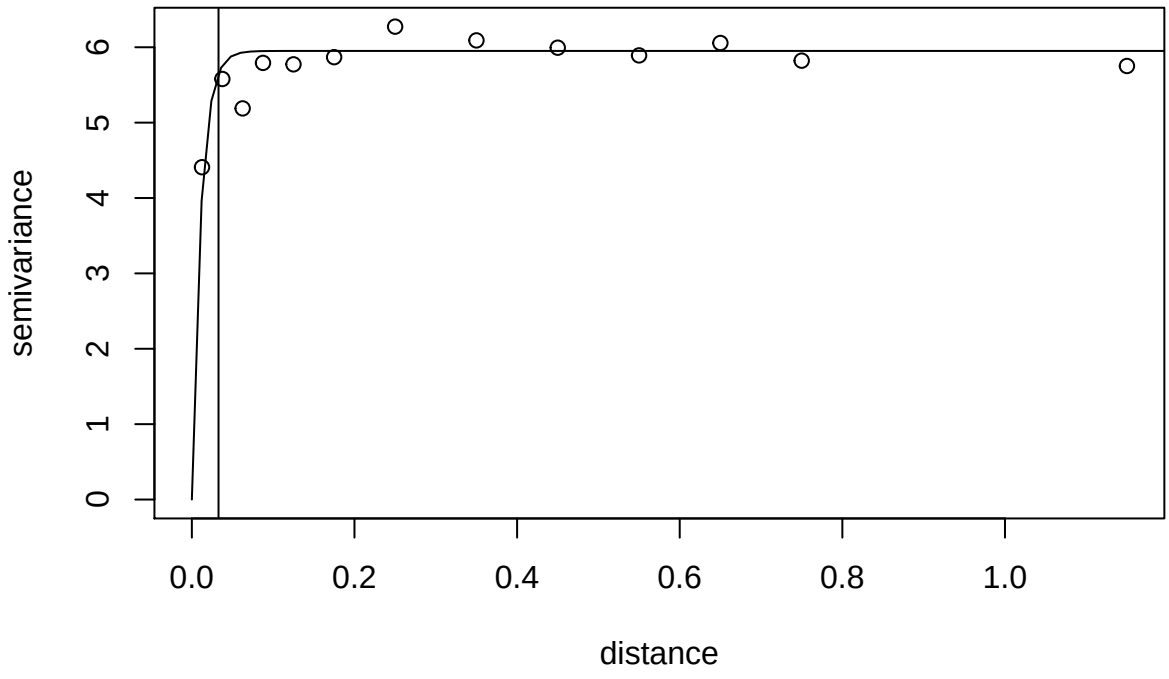
### Getting Variogram Practical Range

Fit a variogram on the residuals of fit1- notice we can see spatial autocorrelation. Use the variogram practical range as a tuning parameter for the Lahiri method of block selection. Always check the variogram visually. Sometimes you may need to change max.dist, breaks or initial conditions to get a sensible practical range. It is hard to automate this process.

```
ini.vals <- expand.grid(seq(0,10,l=100), seq(0,1,l=100))


variogram_block_length = select_b_length_off_variogram_envelope(x,y,resids = fit1$resid,
max.dist = 1.2, breaks=c(0,0.025,0.05,0.075,0.1,0.15,0.2,0.3,0.4,0.5,0.6,0.7,0.8,1.5),
ini=ini.vals)
```

```
## variog: computing omnidirectional variogram
## variofit: covariance model used is matern
## variofit: weights used: npairs
## variofit: minimisation function used: optim
## variofit: searching for best initial value ... selected values:
##               sigmasq phi    tausq kappa
## initial.value "5.96"  "0.01" "0"   "0.5"
## status        "est"   "est"  "est" "fix"
## loss value: 3853.7181489579
```




```
variogram_block_length
```

```
## [1] 0.03280132
```

### Run Block Bootstrap

Run Block Bootstrap, using Lahiri method of block size selection, with the variogram to select the tuning parameter. We search over 4 blocks sizes (0- i.e. IID, 0.05, 0.1, 0.2). BlockBootApply uses the size which

minimises the Empirical MSE of SE($\beta$). Use 0.05 as tuning block length as per variogram practical range ($tuning_block_length = 2$).

```
lookuptables.folderpathname = "LookupTables/"
```

```
Results.example4 = BlockBootApply (x = x ,y = y ,block_Ls = c(0,0.05,0.1,0.2), Grid_space = 0.01 ,
dat = dat , Stat.function = GetAUC,  tuning_block_length = 2, NBoot = 500,
method.block.length.select = "Lahiri", type="SE", lookuptables.folderpath=lookuptables.folderpathname)
```

Check sensitivity to tuning parameter. See that all columns are minimised by block length 0.1, so tuning block length wasn't terribly imporant here

```
Results.example4$Empirical.MSE
```

```
##                 tuninglength0 tuninglength0.05 tuninglength0.1
## blocklength0      0.001626589       0.02305761      0.07962379
## blocklength0.05   0.007378852       0.02219981      0.07119708
## blocklength0.1    0.029063621       0.04260590      0.09013901
## blocklength0.2    0.040642723       0.08243159      0.16230864
##                 tuninglength0.2
## blocklength0          0.1593789
## blocklength0.05       0.1441683
## blocklength0.1        0.1617980
## blocklength0.2        0.2629567
```

The block size parameter chosen is:

```
Results.example4$Lahiri_block_size
```

```
## [1] 0.05
```

This is the standard error (estimated via block bootstrap to account for spatial autocorrelation) of the AUC.

```
Results.example4$SE.estimate
```

```
## [1] 0.02576194
```