Project 2

Problem 3

Neumann-Scott Event RF

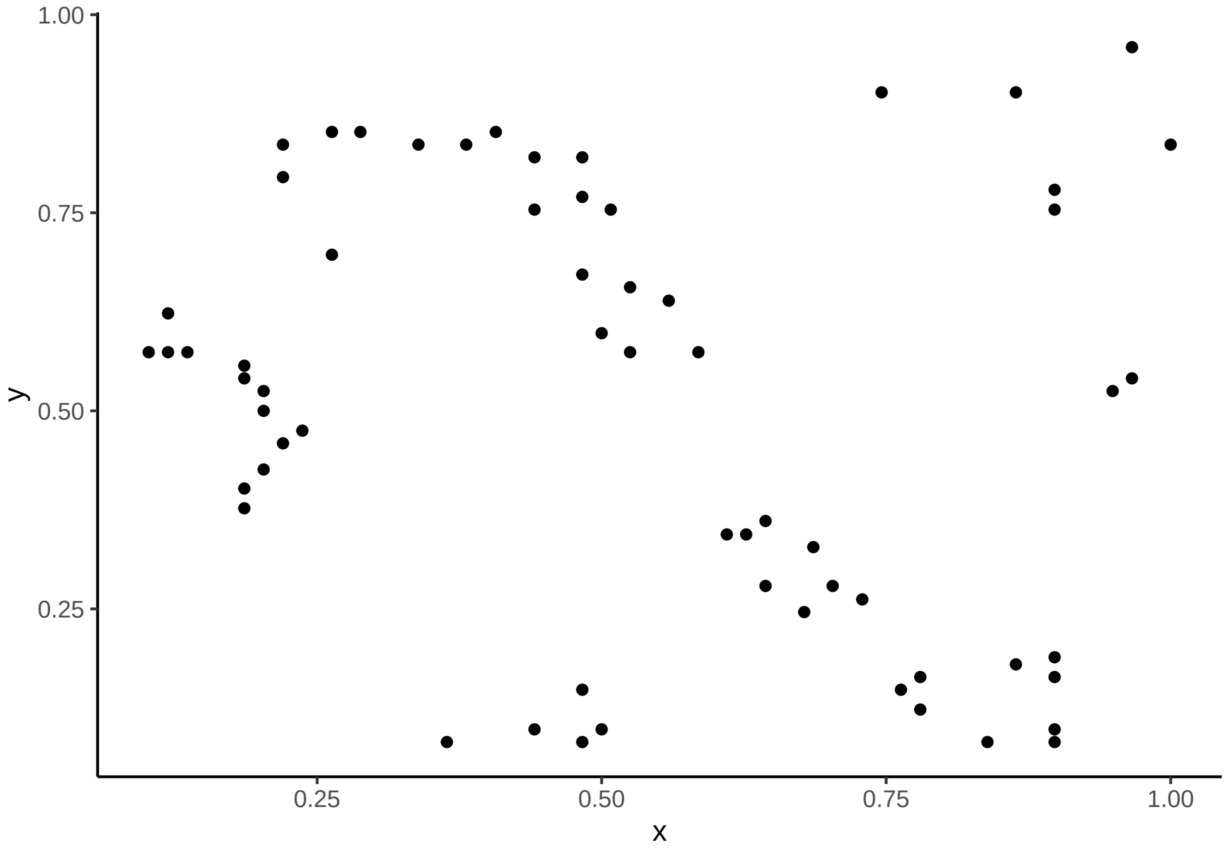
In the Neumann-Scott Event RF points are assumed distributed around some mother locations. The number of mother nodes often follow a poission distribution. With intensity λ_m, the number of points (children) around the mother node is then assume to follow a distribution according to the count pdf $p(k)$ and the intensity $p(\mathbf{x}|\mathbf{x}_j^M)$. The points are assumed independent, around a mother node. The intensity pdf is usually gaussian, with center in the mother node.

A possible border problem when dealing with a finite event space is that the intensity function, especially the guassian, might generate points outisde of the event space. A possibility is to use a function that enforces border conditions, i.e. the torus border condntions, when generating points.

Investigating the redwood data

We first plot the data:

```r
redwood <- read.csv("redwood.dat", header = F, sep = " ")
redwood <- as.data.frame(redwood)
colnames(redwood) <- c("x", "y")

ggplot(data = redwood) + geom_point(aes(x,y)) + theme_classic()
```



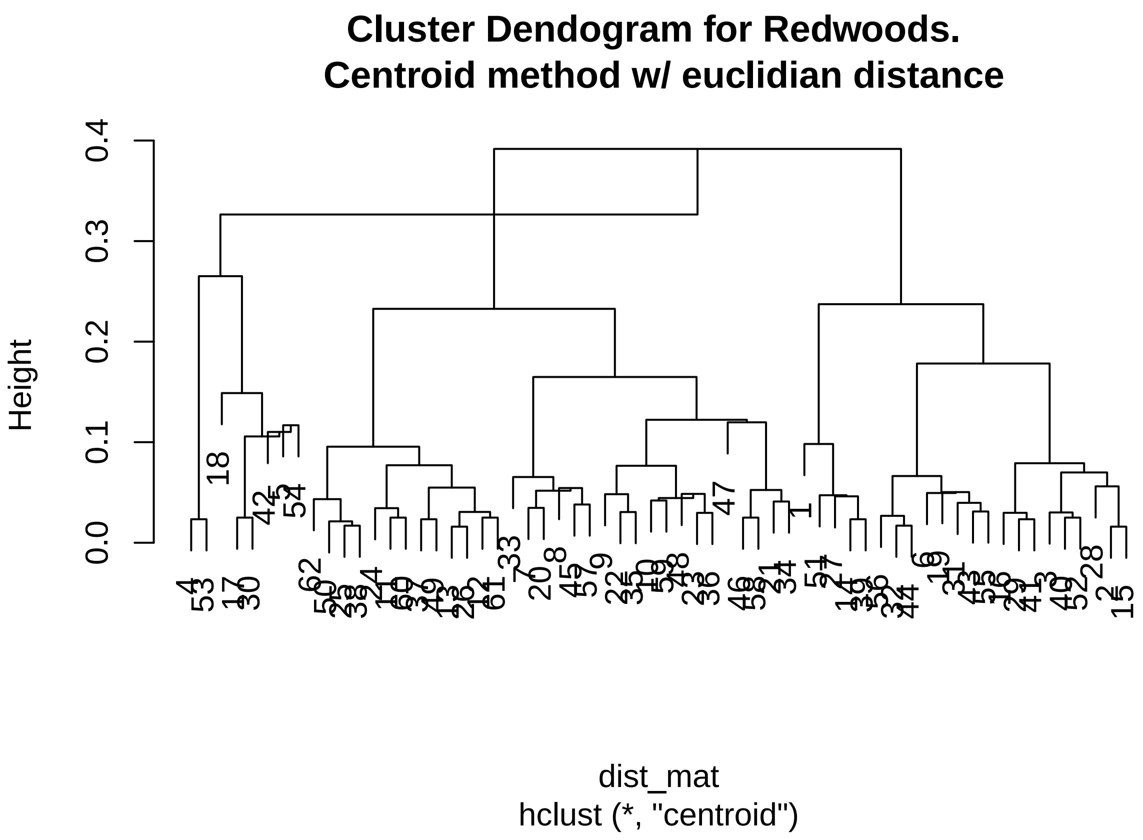
There seems to be some clusters of redwood. Want to fit an empirical Neumann Scott Event-ref.

From the data there seems to be 8 mother nodes, with clusters of redwoods around.

To avoid having to classify each point by hand, we use hierchical clustering with L2 distance to find the cluster and their centers. We use centroid clustering, and will use the centroids as estimation of position to the mother node. We have already counted 8 mothernodes, so we will use that estiamate.

```
# Doing hierchical clustering with centroid and eculidan
dist_mat <- dist(redwood, method = 'euclidean') # Calculate distances
hclust_centroid <- hclust(dist_mat, method = 'centroid') # Use centroid as moteher node
```

```
plot(hclust_centroid, main = "Cluster Dendogram for Redwoods. \n Centroid method w/ euclidian distance")
```

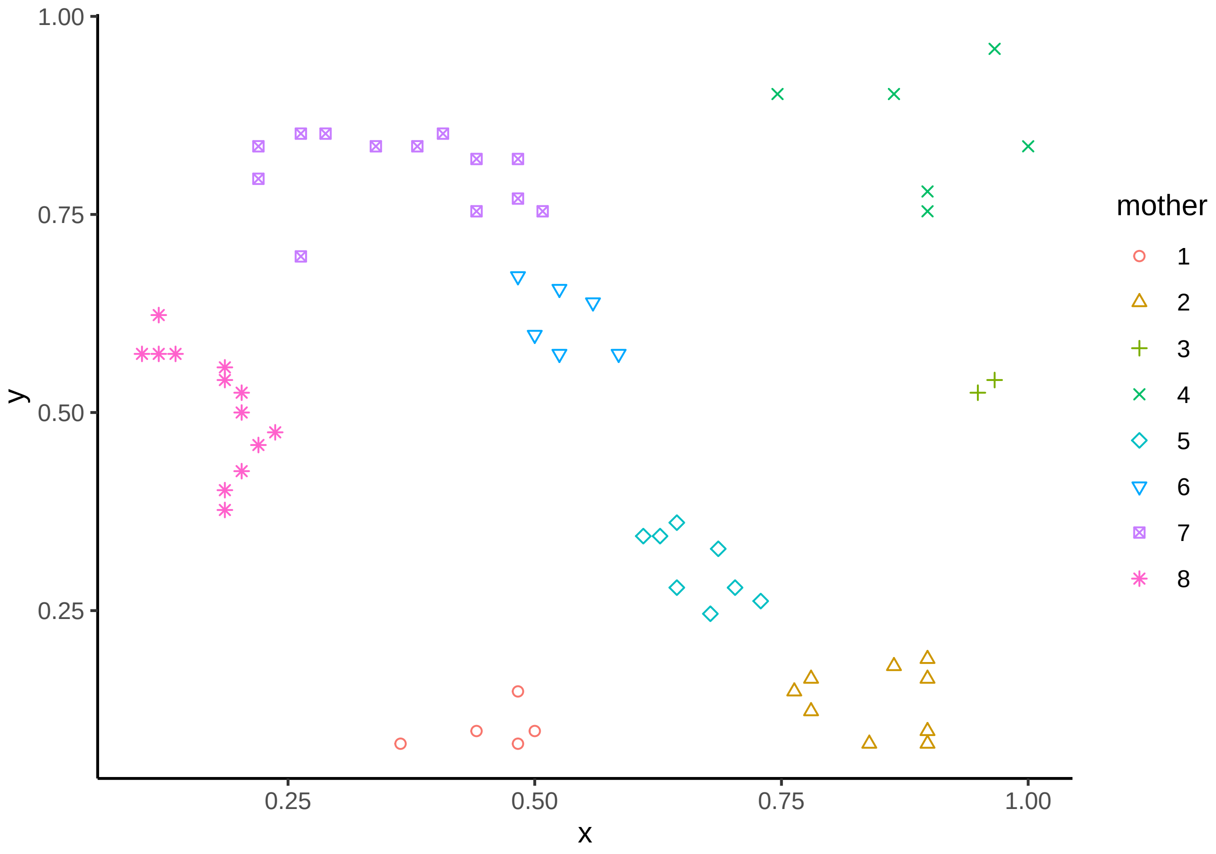


The cluster dendogram is illustrated in Figure ??

```
cut_centroid <- cutree(hclust_centroid, k = 8) # Find node of elements when we have 7 groups
```

```
redwood$mother = as.factor(cut_centroid)
```

```
ggplot(redwood) + geom_point(aes(x= x, y = y, color = mother, shape = mother)) + theme_classic() +  scal
```



Classiying by cluster we get the following plot. See ??

We know find the position of the centroids. (As hclus does not give us that we do it manually)

```r
# Finding centers and count for each mother
mothers <- redwood %>%
  group_by(mother) %>%
  summarise(x = mean(x),
            y = mean(y),
            n = n()) %>%
  as.data.frame()
```

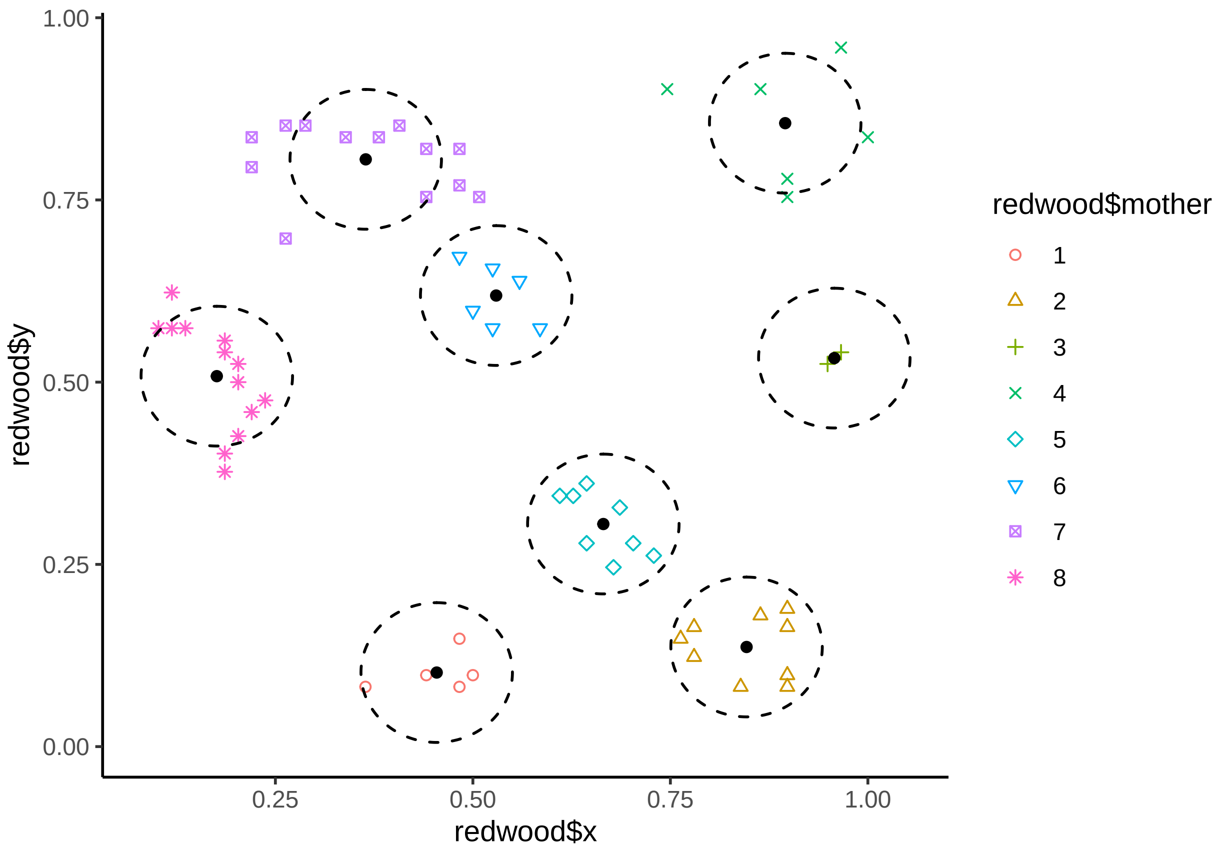
We also want to estiamte the standard deviation. If we assume that each cluster has independent standard deviation. And the each mother estimation is unbiased. We get: (as deviation in x-coordinate and y-coordiante are independent and equally normally distributed)

```r
# Estimating standard dev
redwood_w_mother <- left_join(redwood, mothers, by = c("mother"))
colnames(redwood_w_mother) <- c("x", "y", "mother", "x.m", "y.m", "n")
redwood_w_mother$dist_mother <- sqrt((redwood_w_mother$x - redwood_w_mother$x.m)^2 + (redwood_w_mother$
redwood_w_mother$x.dist <- redwood_w_mother$x - redwood_w_mother$x.m
redwood_w_mother$y.dist <- redwood_w_mother$y - redwood_w_mother$y.m

std2 <- 1/(length(redwood_w_mother$x.dist) + length(redwood_w_mother$y.dist) - 1)* (sum(redwood_w_mothe

q95 <- qnorm(0.95, mean = 0, sd = sqrt(std2))
```

If we assume that they a common standard deviation we get: 0.0033968

Plotting the redwood forest, with the mother nodes we get:

```
ggplot() +
  geom_point(aes(x= redwood$x, y = redwood$y, color = redwood$mother, shape = redwood$mother)) +
  theme_classic() +
  scale_shape_manual(values=1:nlevels(redwood$mother)) +
  geom_point(aes(x = mothers$x, y = mothers$y)) +
  geom_circle(aes(r = rep(q95, length(mothers$y)), y0 = mothers$y, x0 = mothers$x), linetype = "dashed"
```



```
# Mean of realization around mother
ns <- redwood %>% group_by(mother) %>% summarise(n = n()) %>% ungroup()
lambda.c <- mean(ns$n)
```

TODO: Q are mother nodes a part of the count ? TODO: Q what is the TORUS-3d algorithm not in my version of the note (p. 131). Is it to project outliers onto some torus? Then shifting them around some

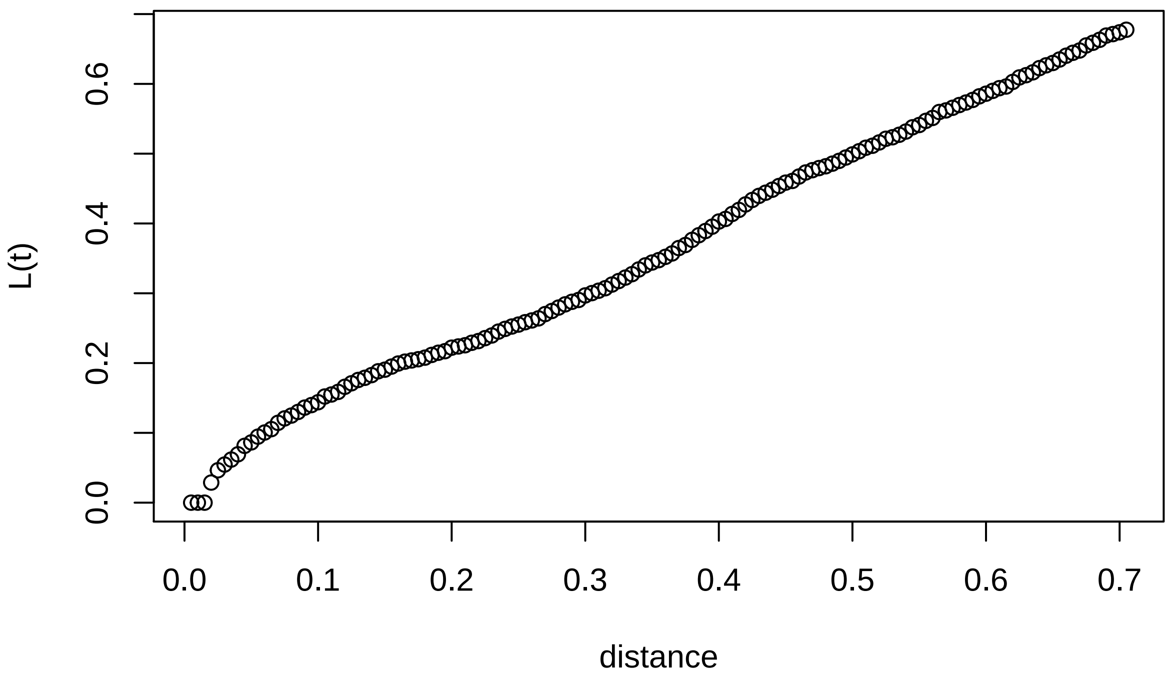
So the parameters of our model are:

- Poisson with $lambda = 8$ for number of mother nodes
- $\sigma_c^2 = 0.0033968$
- Poisson behind number of childrean with $lambda = $7.75
- Gaussian spread around mother node

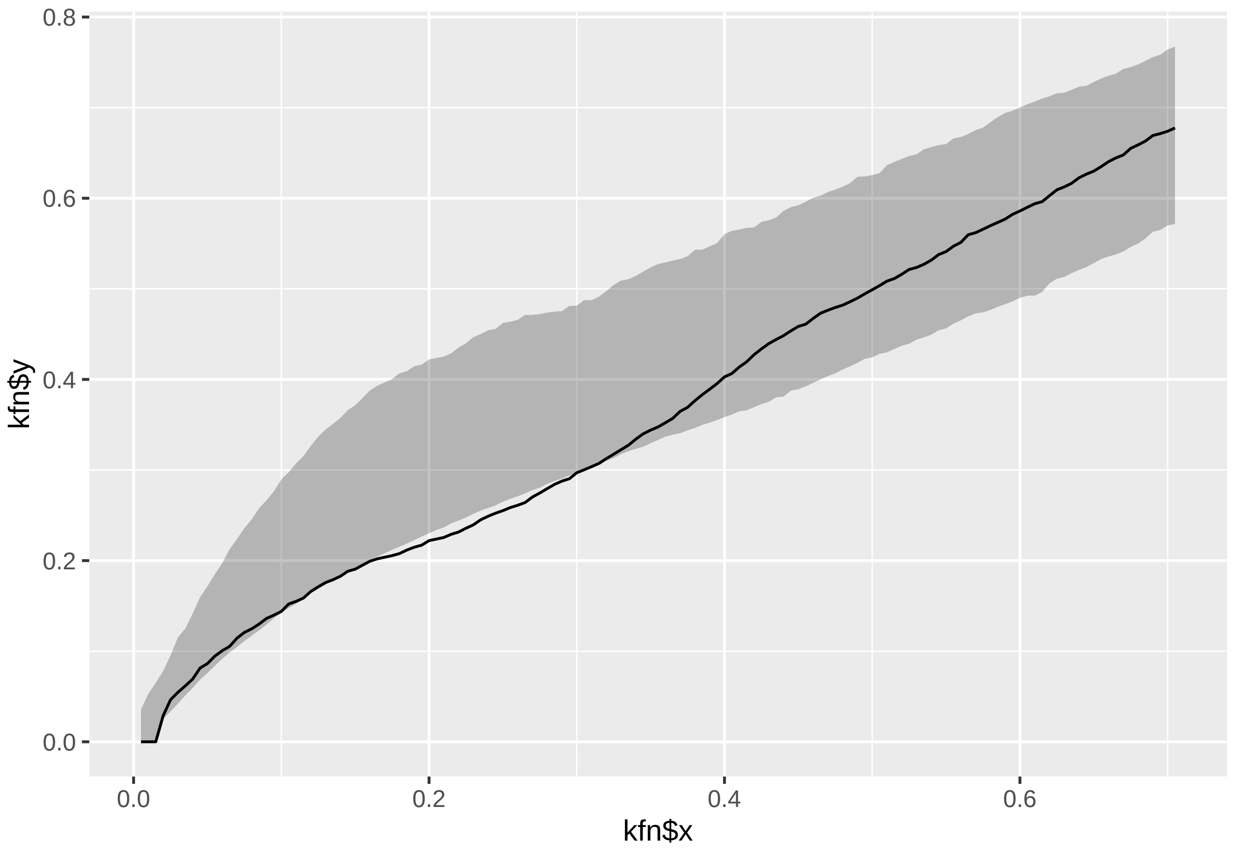
```
# ppinit does not seem to read properly so we do it manually
# redwood.pp <- ppinit("redwood.dat")
# plot(Kfn(redwood.pp, 1, k = 200), type="b", xlab="distance", ylab="L(t)")
```

```
redwood <- read.csv("redwood.dat", header = F, sep = " ")
redwood <- as.data.frame(redwood)
```

```r
colnames(redwood) <- c("x", "y")

D = ppregion(xl = 0, xu = 1, yl = 0, yu = 1)

redwood.pp2 <- list(
  x = redwood$x,
  y = redwood$y,
  area = D
)


kfn <- Kfn(redwood.pp2, 1, k = 200)
plot(kfn, type="b", xlab="distance", ylab="L(t)")
```



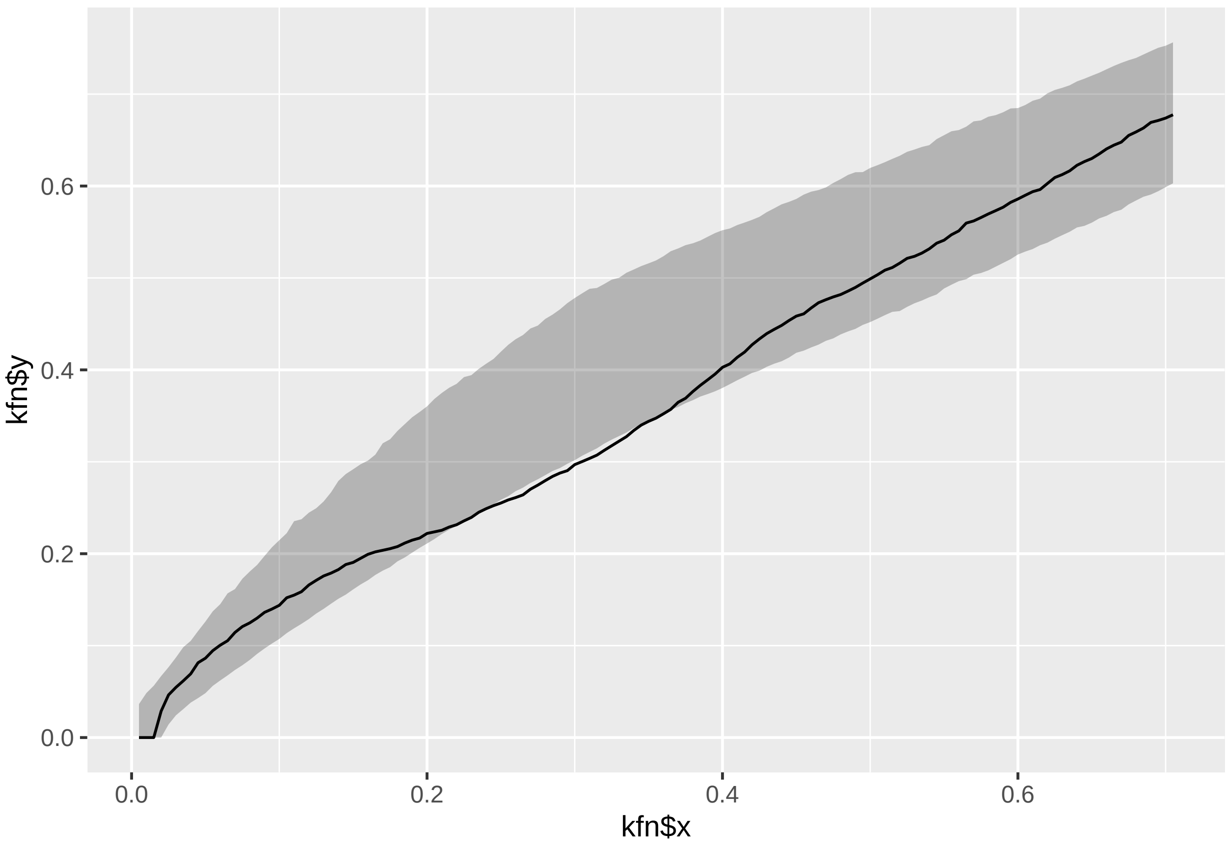
```r
# Distances we want to check out
neumann_scot_generate <- function(lambda.m, lambda.c, std2, seed){
  set.seed(seed)
  count = 0
  k <- rpois(n = 1, lambda = lambda.m)
  xs <- c()
  mothers <- c()
  for(j in 1:k){
    xj <- runif(2)
    n.child <- rpois(1, lambda = lambda.c)
    for(i in 1:n.child){
      x <- MASS::mvrnorm(n = 1, Sigma = std2*diag(2), mu = xj)
      # Just project for now
      if(x[1] < 0) x[1] <- 0
      if(x[1] > 1) x[1] <- 1

      if(x[2] < 0) x[2] <- 0
      if(x[2] > 1) x[2] <- 1
      xs <- rbind(xs, x)
    }
    count <- count + n.child
    mothers <- c(mothers, rep(j, n.child))
  }
  list(xs = xs, mothers = mothers)
}

lambda.m <- 8
n.rels <- 1000
rels <- lapply(1:n.rels, function(seed) neumann_scot_generate(lambda.m = lambda.m, lambda.c = lambda.c,
rels <- lapply(rels, function(rel) list(x = rel[,1], y = rel[,2], area = D))
kfns <- lapply(rels, function(rel) Kfn(rel, 1, k = 200))
bands <- sapply(kfns, function(kfn) kfn$y)
lower_band <- apply(bands, 1, function(x) quantile(x, probs = 0.025))
upper_band <- apply(bands, 1, function(x) quantile(x, probs = 0.975))
mid_band <- apply(bands, 1, function(x) quantile(x, probs = 0.5))

ggplot() + geom_ribbon(aes(x = kfn$x, ymax = upper_band, ymin = lower_band), alpha = 0.3) + geom_line(ae
```



Seems to be a bit bad a mid distances, a higher variance might on spread might help on that.

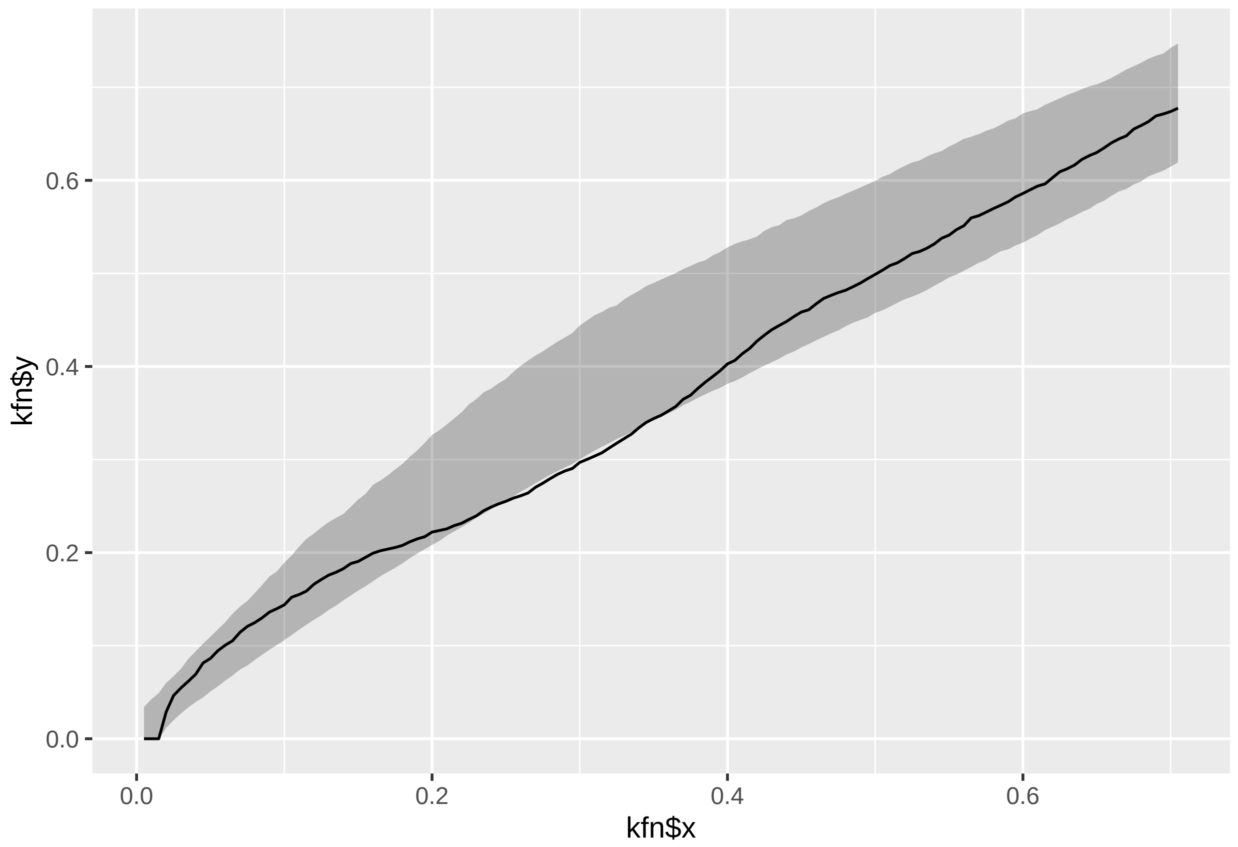
```
lambda.m <- 8
n.rels <- 1000
std2 <- std2*4
rels <- lapply(1:n.rels, function(seed) neumann_scot_generate(lambda.m = lambda.m, lambda.c = lambda.c,
rels <- lapply(rels, function(rel) list(x = rel[,1], y = rel[,2], area = D))
```

```
kfns <- lapply(rels, function(rel) Kfn(rel, 1, k = 200))
bands <- sapply(kfns, function(kfn) kfn$y)
lower_band <- apply(bands, 1, function(x) quantile(x, probs = 0.025))
upper_band <- apply(bands, 1, function(x) quantile(x, probs = 0.975))
mid_band <- apply(bands, 1, function(x) quantile(x, probs = 0.5))

ggplot() + geom_ribbon(aes(x = kfn$x, ymax = upper_band, ymin = lower_band), alpha = 0.3) + geom_line(ae
```



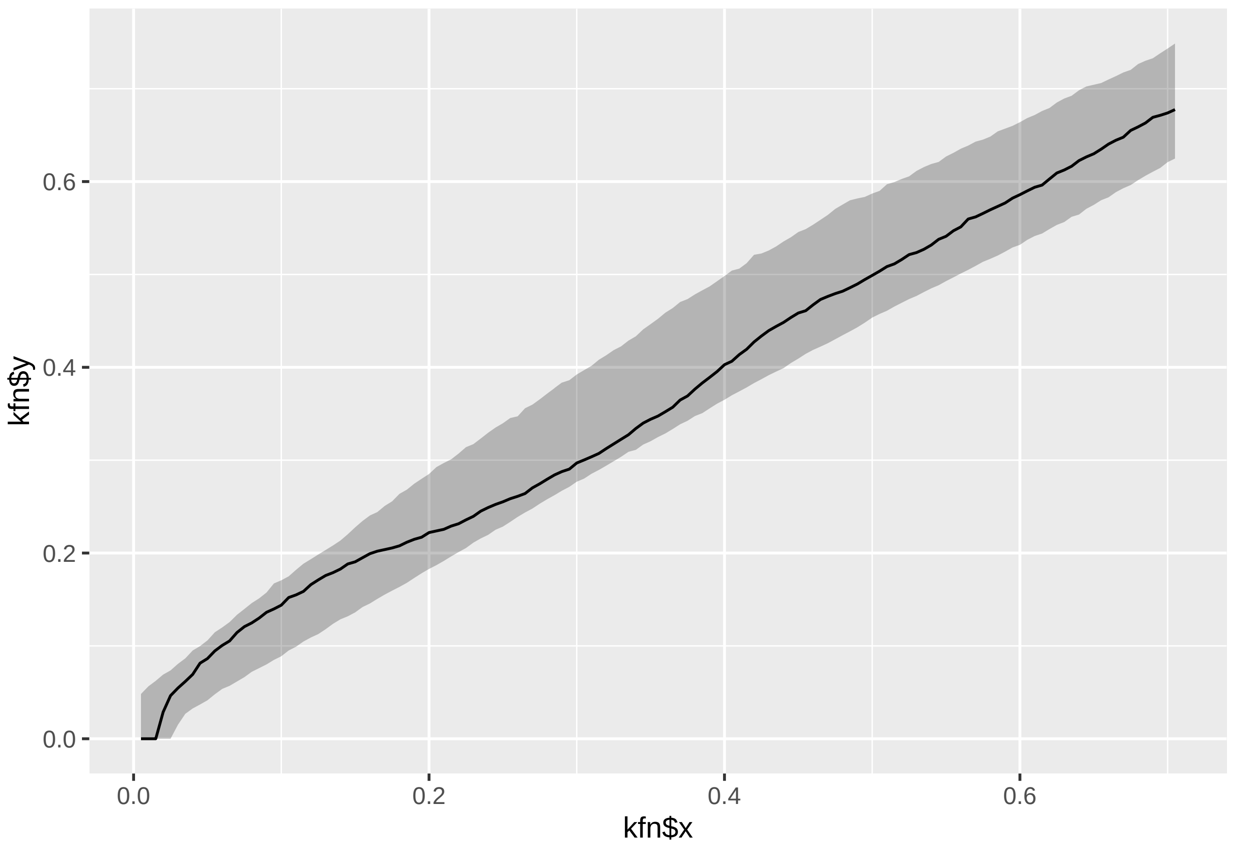
Helped somewhat, change lambda somewhat to see if it helps.

```
lambda.m <- 9.5
n.rels <- 1000
std2 <- std2
rels <- lapply(1:n.rels, function(seed) neumann_scot_generate(lambda.m = lambda.m, lambda.c = lambda.c,
rels <- lapply(rels, function(rel) list(x = rel[,1], y = rel[,2], area = D))
kfns <- lapply(rels, function(rel) Kfn(rel, 1, k = 200))
bands <- sapply(kfns, function(kfn) kfn$y)
lower_band <- apply(bands, 1, function(x) quantile(x, probs = 0.025))
upper_band <- apply(bands, 1, function(x) quantile(x, probs = 0.975))
mid_band <- apply(bands, 1, function(x) quantile(x, probs = 0.5))

ggplot() + geom_ribbon(aes(x = kfn$x, ymax = upper_band, ymin = lower_band), alpha = 0.3) + geom_line(ae
```



Does not seem to have any good effect. We try to increase variance again.

```
lambda.m <- 8
n.rels <- 1000
std2 <- std2*4
rels <- lapply(1:n.rels, function(seed) neumann_scot_generate(lambda.m = lambda.m, lambda.c = lambda.c,
rels <- lapply(rels, function(rel) list(x = rel[,1], y = rel[,2], area = D))
kfns <- lapply(rels, function(rel) Kfn(rel, 1, k = 200))
bands <- sapply(kfns, function(kfn) kfn$y)
lower_band <- apply(bands, 1, function(x) quantile(x, probs = 0.025))
upper_band <- apply(bands, 1, function(x) quantile(x, probs = 0.975))
mid_band <- apply(bands, 1, function(x) quantile(x, probs = 0.5))

ggplot() + geom_ribbon(aes(x = kfn$x, ymax = upper_band, ymin = lower_band), alpha = 0.3) + geom_line(ae
```



Realization is now within 95% confidence interval an we har happy with the fit. Ended up using

- Poisson with $lambda = 8$ for number of mother nodes
- $\sigma_c^2 = 0.0543488$
- Poisson behind number of childrean with \$lambda = \$7.75
- Gaussian spread around mother node

Display three realiations of the model with this parameters to what was observed (we overlay our grouping to the plots) :

```r
df1 <- neumann_scot_generate(lambda.m = lambda.m, lambda.c = lambda.c, std2 = std2, seed = 1)
df1 <- as.data.frame(df1)
colnames(df1) <- c("x", "y", "mothers")
df1$mothers <- as.factor(df1$mothers)
df2 <- neumann_scot_generate(lambda.m = lambda.m, lambda.c = lambda.c, std2 = std2, seed = 2)
df2 <- as.data.frame(df2)
colnames(df2) <- c("x", "y", "mothers")
df2$mothers <- as.factor(df2$mothers)
df3 <- neumann_scot_generate(lambda.m = lambda.m, lambda.c = lambda.c, std2 = std2, seed = 3)
df3 <- as.data.frame(df3)
colnames(df3) <- c("x", "y", "mothers")
df3$mothers <- as.factor(df3$mothers)

p1 <- ggplot() +
  geom_point(aes(x= df3$x, y = df3$y, color = df3$mothers, shape = df3$mothers)) +
  theme_classic() +
  scale_shape_manual(values=1:nlevels(df3$mothers))
```

```r
p2 <- ggplot() +
  geom_point(aes(x= df2$x, y = df2$y, color = df2$mothers, shape = df2$mothers)) +
  theme_classic() +
  scale_shape_manual(values=1:nlevels(df2$mothers))

p3 <- ggplot() +
  geom_point(aes(x= df1$x, y = df1$y, color = df1$mothers, shape = df1$mothers)) +
  theme_classic() +
  scale_shape_manual(values=1:nlevels(df1$mothers))
```

See that it is quite normal with overlapping data from mothers, the real intensity for number of mothers might thus be even higher. But again it is difficult to say.

Ovreall the figures seems to match quite well.

Problem 4: Repulsive event spatial variables

The Strauss model can be expressed on the conditional form as:

$$p(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k | k_d = k) \propto \prod_{i,j \in \{1,\dots,k\}} exp(-\phi(\boldsymbol{\tau}_{ij})) \tag{1}$$

Where $\$\boldsymbol{\tau}$_{ij} = | **x**_i - **x**_j | Usually we have:

$$\phi(\tau) = \begin{cases} \phi_0, & 0 \le \tau \le \tau_0 \\ \phi_0 exp\{(\phi_1(\tau - \tau_0)), & \tau \ge \tau_0 \end{cases} \tag{2}$$