analyse de réseau pardi

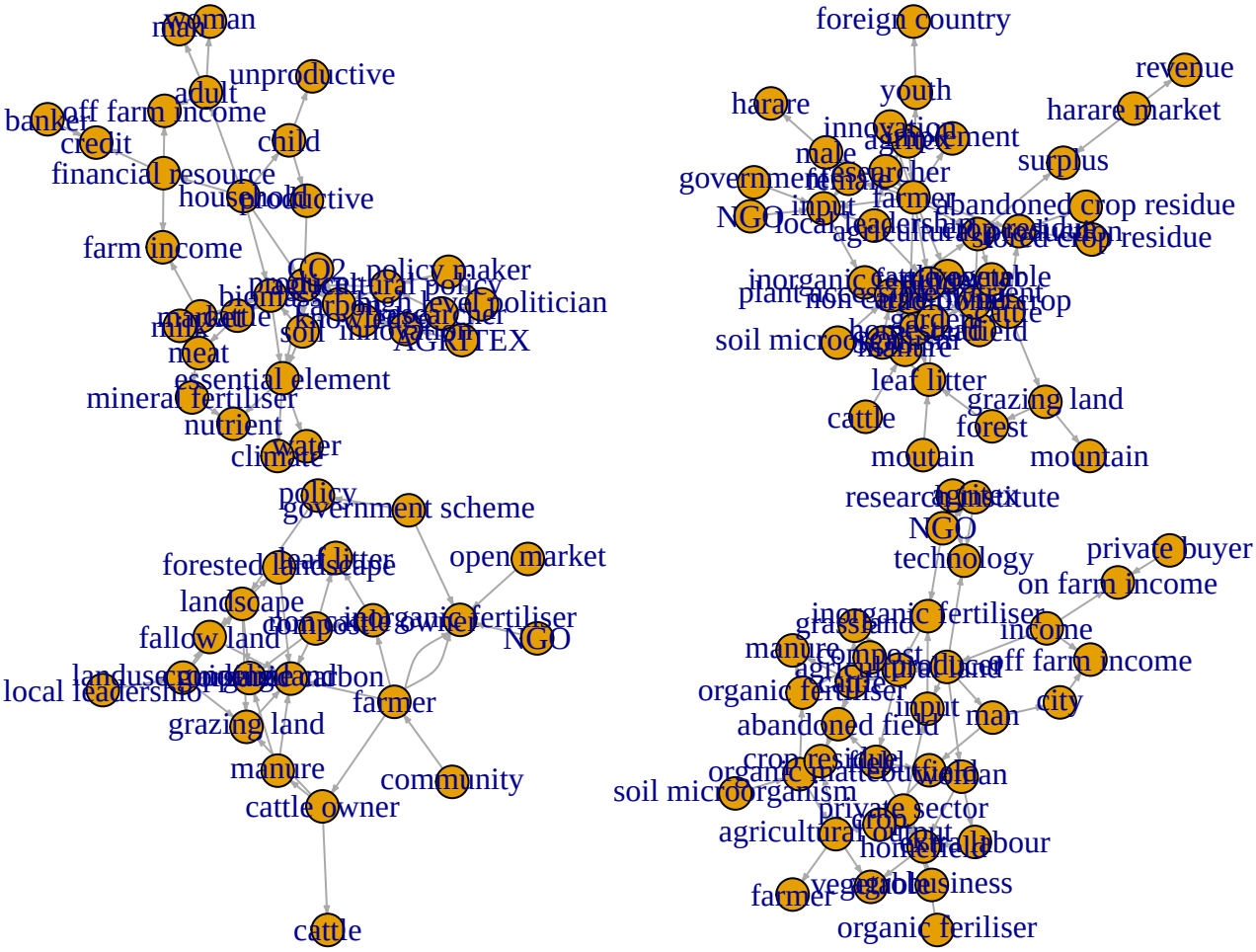
Delay

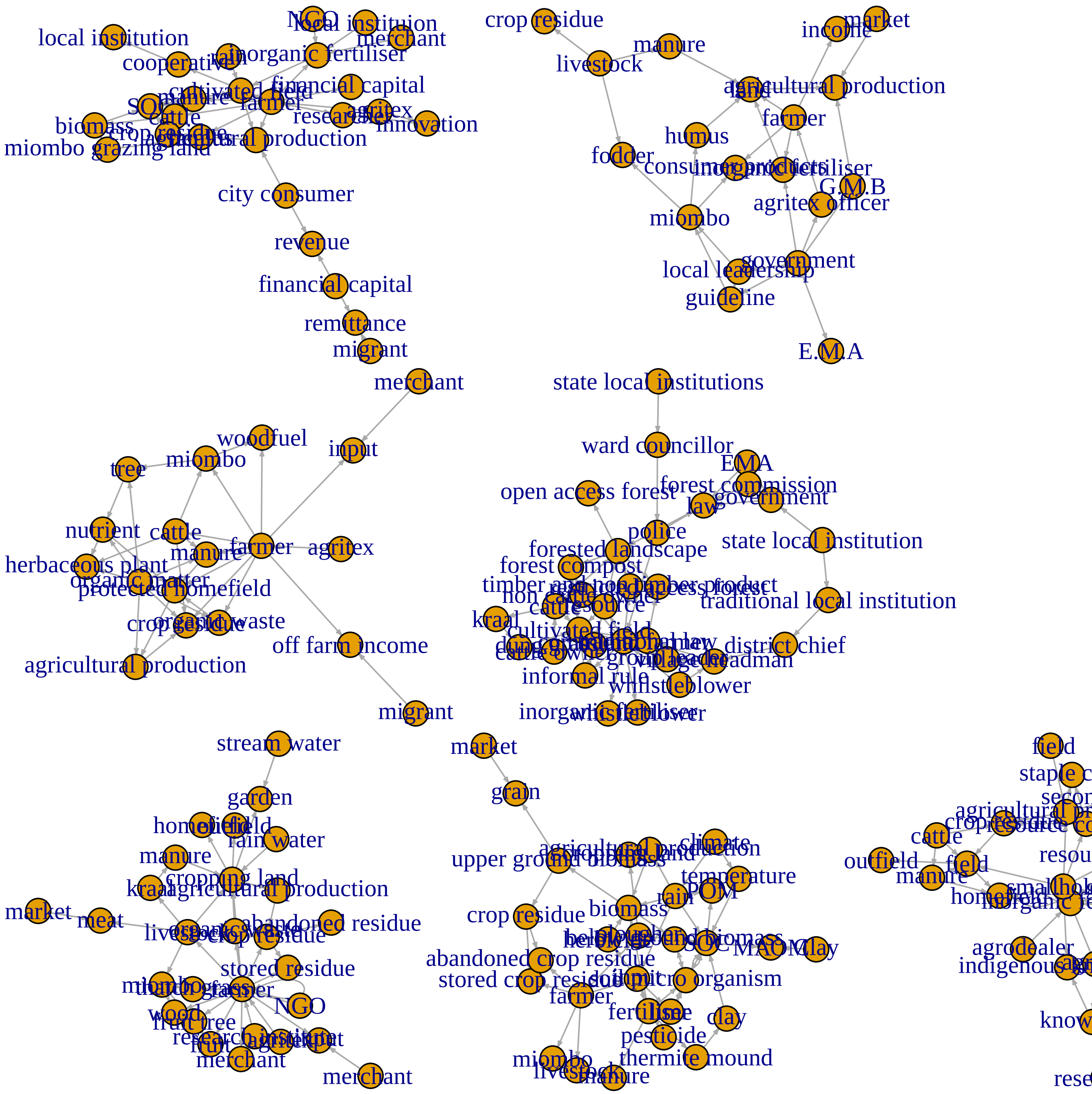
31/07/2020

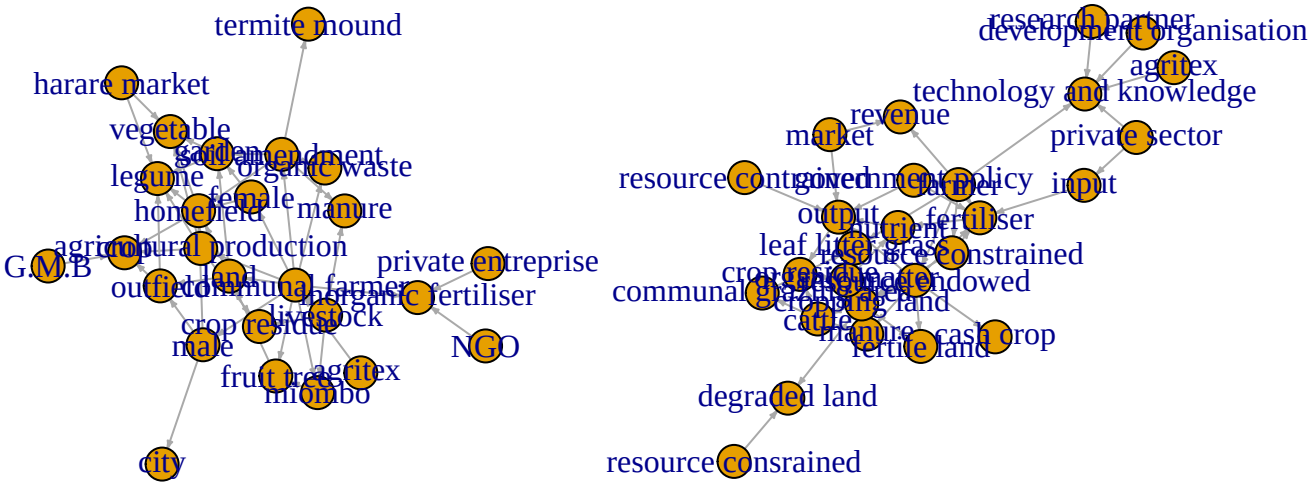
Présenter un réseau

Voilà l'ensemble des grapges d'interaction produit.

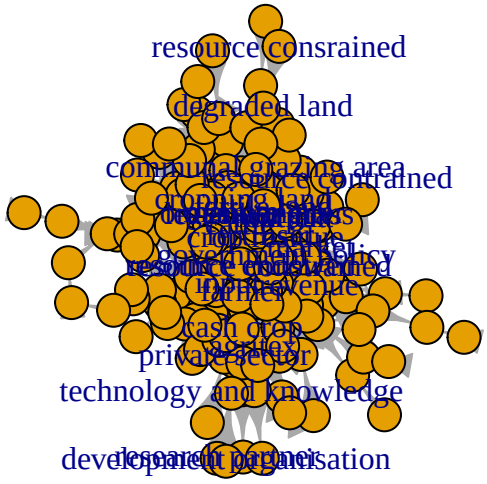
```
for(i in 1:length(v.file)){
plot(my_net_list[[i]], edge.arrow.size = 0.2)
}
```




local institution
cooperative
NGO
local instituion
merchant
inorganic fertiliser
financial capital
innovation
biomass
miombo grazing land
agritex
city consumer
revenue
financial capital
remittance
migrant
merchant
crop residue
livestock
manure
income
market
agricultural production
farmer
humus
fodder
G.M.B
agritex officer
miombo
government
local leadership
guideline
E.M.A
woodfuel
input
tree
miombo
nutrient
cattle
farmer
agritex
herbaceous plant
protected homefield
organic waste
crop residue
off farm income
agricultural production
migrant
state local institutions
ward councillor
EMA
open access forest
forest commission
law
government
police
forested landscape
state local institution
forest compost
traditional local institution
kraal
district chief
informal rule
whistleblower
stream water
garden
rain water
manure
cropping land
kraal
agricultural production
market
meat
abandoned residue
stored residue
NGO
merchant
market
grain
climate
upper ground biomass
temperature
crop residue
biomass
rain
SOM
abandoned crop residue
stored crop residue
farmer
micro organism
clay
pesticide
thermite mound
manure
field
cattle
outfield
manure
agrodealer




```
plot(big.g)
```



Distance Hamming

La distance de Hamming mesure le nombre minimum de substitutions nécessaires pour changer (transformer) un "objet" mathématique (c'est-à-dire des chaînes de caractères ou des binaires) en un autre.

En théorie des réseaux, elle peut donc être définie comme le nombre de connexions différentes entre deux réseaux (elle peut également être formulée pour les réseaux de taille inégale et pour les graphiques pondérés ou dirigés). Dans un cas simple où vous avez deux réseaux Erdos-Renyi (la matrice de contiguïté a 1 si la paire de nœuds est connectée et 0 sinon), la distance est mathématiquement définie comme suit :

$$\frac{1}{N \times (N-1)} \sum_{1 \leq i \neq j \leq N} |A_{ij}^{(1)} - A_{ij}^{(2)}|$$

Les valeurs qui sont soustraites sont les deux matrices de contiguïté.

```
combinaisons <- t(as.data.frame(combn(c(1:length(v.file)),2)))
```

```
g.similatity <-  data.frame()
for(i in 1:length(combinaisons[,1])){
  g1 <- combinaisons[i,1]
  g2 <- combinaisons[i,2]
```

```
  ##Cette solution vient de là : https://lists.gnu.org/archive/html/igraph-help/2008-04/msg00017.html
  int <- graph.intersection(my_net_list[[g1]],my_net_list[[g2]])
  n.dist <- ecount(my_net_list[[g1]])+ecount(my_net_list[[g2]])-2*ecount(int)

  g.similatity <- rbind(g.similatity, c(combinaisons[i,1], combinaisons[i,2], n.dist))

}
colnames(g.similatity) <- c("g1","g2","Hamming")
g.similatity
```

```
##    g1 g2 Hamming
## 1   1  2     116
## 2   1  3      86
## 3   1  4     101
## 4   1  5      84
## 5   1  6      78
## 6   1  7      83
## 7   1  8     102
## 8   1  9      87
## 9   1 10     100
## 10  1 11      92
## 11  1 12      96
## 12  1 13      97
## 13  2  3      90
## 14  2  4     111
## 15  2  5      90
## 16  2  6      92
## 17  2  7      87
## 18  2  8     114
## 19  2  9      95
## 20  2 10     108
## 21  2 11      98
## 22  2 12     104
## 23  2 13     109
## 24  3  4      85
## 25  3  5      68
## 26  3  6      60
## 27  3  7      67
## 28  3  8      84
## 29  3  9      71
## 30  3 10      84
## 31  3 11      74
## 32  3 12      78
## 33  3 13      79
## 34  4  5      85
## 35  4  6      79
## 36  4  7      78
## 37  4  8     101
## 38  4  9      90
## 39  4 10     101
## 40  4 11      83
## 41  4 12      91
## 42  4 13      92
```

```
## 43  5  6      62
## 44  5  7      59
## 45  5  8      88
## 46  5  9      71
## 47  5 10      86
## 48  5 11      72
## 49  5 12      78
## 50  5 13      81
## 51  6  7      59
## 52  6  8      78
## 53  6  9      63
## 54  6 10      74
## 55  6 11      66
## 56  6 12      68
## 57  6 13      73
## 58  7  8      83
## 59  7  9      60
## 60  7 10      77
## 61  7 11      67
## 62  7 12      75
## 63  7 13      72
## 64  8  9      89
## 65  8 10     100
## 66  8 11      92
## 67  8 12      94
## 68  8 13      97
## 69  9 10      79
## 70  9 11      77
## 71  9 12      75
## 72  9 13      82
## 73 10 11      90
## 74 10 12      90
## 75 10 13      93
## 76 11 12      82
## 77 11 13      83
## 78 12 13      91
```

Il ne reste plus qu'a produire une table de similarité

```
## créer une tablea de données lisible par les humains

#Table des nomes avec ID
table.names <- data.frame(ID = 1:length(v.file), name = str_extract(v.file, '.*(?=\\.csv)'))

tps1 <- left_join(g.similatity, table.names, by = c("g1" = "ID"))
tps2 <- left_join(g.similatity, table.names, by = c("g2" = "ID"))

final.data <- data.frame(name1=tps1$name, name2=tps2$name, Hamming=tps2$Hamming)
final.data[order(final.data$Hamming),]
```

```
##     name1  name2 Hamming
## 44   IN_5   IN_7      59
## 51   IN_6   IN_7      59
## 26   IN_3   IN_6      60
## 59   IN_7 IN_9_1      60
```

```
## 43   IN_5   IN_6      62
## 53   IN_6 IN_9_1      63
## 55   IN_6  OUT_1      66
## 27   IN_3   IN_7      67
## 61   IN_7  OUT_1      67
## 25   IN_3   IN_5      68
## 56   IN_6  OUT_2      68
## 29   IN_3 IN_9_1      71
## 46   IN_5 IN_9_1      71
## 48   IN_5  OUT_1      72
## 63   IN_7  OUT_3      72
## 57   IN_6  OUT_3      73
## 31   IN_3  OUT_1      74
## 54   IN_6 IN_9_2      74
## 62   IN_7  OUT_2      75
## 71 IN_9_1  OUT_2      75
## 60   IN_7 IN_9_2      77
## 70 IN_9_1  OUT_1      77
## 5    IN_1   IN_6      78
## 32   IN_3  OUT_2      78
## 36   IN_4   IN_7      78
## 49   IN_5  OUT_2      78
## 52   IN_6   IN_8      78
## 33   IN_3  OUT_3      79
## 35   IN_4   IN_6      79
## 69 IN_9_1 IN_9_2      79
## 50   IN_5  OUT_3      81
## 72 IN_9_1  OUT_3      82
## 76  OUT_1  OUT_2      82
## 6    IN_1   IN_7      83
## 40   IN_4  OUT_1      83
## 58   IN_7   IN_8      83
## 77  OUT_1  OUT_3      83
## 4    IN_1   IN_5      84
## 28   IN_3   IN_8      84
## 30   IN_3 IN_9_2      84
## 24   IN_3   IN_4      85
## 34   IN_4   IN_5      85
## 2    IN_1   IN_3      86
## 47   IN_5 IN_9_2      86
## 8    IN_1 IN_9_1      87
## 17   IN_2   IN_7      87
## 45   IN_5   IN_8      88
## 64   IN_8 IN_9_1      89
## 13   IN_2   IN_3      90
## 15   IN_2   IN_5      90
## 38   IN_4 IN_9_1      90
## 73 IN_9_2  OUT_1      90
## 74 IN_9_2  OUT_2      90
## 41   IN_4  OUT_2      91
## 78  OUT_2  OUT_3      91
## 10   IN_1  OUT_1      92
## 16   IN_2   IN_6      92
## 42   IN_4  OUT_3      92
```

```
## 66   IN_8  OUT_1      92
## 75 IN_9_2  OUT_3      93
## 67   IN_8  OUT_2      94
## 19   IN_2 IN_9_1      95
## 11   IN_1  OUT_2      96
## 12   IN_1  OUT_3      97
## 68   IN_8  OUT_3      97
## 21   IN_2  OUT_1      98
## 9    IN_1 IN_9_2     100
## 65   IN_8 IN_9_2     100
## 3    IN_1   IN_4     101
## 37   IN_4   IN_8     101
## 39   IN_4 IN_9_2     101
## 7    IN_1   IN_8     102
## 22   IN_2  OUT_2     104
## 20   IN_2 IN_9_2     108
## 23   IN_2  OUT_3     109
## 14   IN_2   IN_4     111
## 18   IN_2   IN_8     114
## 1    IN_1   IN_2     116
```

```
write.csv( x= final.data[order(final.data$Hamming),],file = "data_olivers/final.data.csv")
```

```
vertice.df <- unique(c(as.character(final.data$name1),as.character(final.data$name2)))
g <- graph_from_data_frame(d = final.data, vertices = vertice.df, directed = F)

m <- get.adjacency(g, attr = "Hamming", sparse = F)
l <- layout_with_mds(g, dist = m, dim = 2)
plot(g, layout = l)
```