

Etude du packing cristallin chez PR1 et PR2

Leslie REGAD et Maxime KERMARREC

2019-02-12

Contents

Objectifs	1
Données	1
Détermination des résidus impliqués dans le packing cristallin chez PR1 et PR2	2
Etude du type d'atomes impliqués dans le packing cristallin	5
Détermination du nombre de structures dans lequel un résidu est impliqué dans le packing cristallin	10
Détermination des résidus impliqués dans le packing cristallin chez PR1	13
Identification des résidus impliqués dans le packing cristallin chez au moins une structure de PR1 .	13
Calcul du nombre de structure dans lequel un résidu est impliqué dans le packing cristallin pour les PR1	16
Etude du type d'atomes impliqués dans le packing cristallin chez PR1	17
Etude du packing cristallin chez PR2	21
Identification des résidus impliqués dans le packing cristallin dans au moins une structure de PR2	21
Calcul du nombre de structures de PR2 dans lequel un résidu est impliqué dans le packing cristallin	22
Type d'atomes impliqués dans le packing cristallin chez PR2	24
Comparaison des résidus impliqués dans le packing cristallin chez PR1 et PR2	28
Localisation des résidus impliqués dans le packing cristallin	29
Pour PR	29
Pour PR1	30
Pour PR2	30
Etudier le lien entre la conservation des résidus impliqués dans le packing cristallin et l'espace cristallo des structures	33

du Vendredi 18/01/2018 au ..

Objectifs

Etudier la conservation du packing cristallin dans les 11 structures de PR2 et les 15 structures de PR1. Pour cela, on va identifier les résidus qui sont :

- inclus dans le packing cristallin chez toutes (ou la majorité (=80%)) des structures
- inclus dans le packing cristallin chez toutes (ou la majorité (=80%)) des structures de PR1
- inclus dans le packing cristallin chez toutes (ou la majorité (=80%)) des structures de PR2

Données

- 26 fichiers qui contiennent les atomes impliqués dans le packing cristallin pour chaque structure de PR1 et PR2
- le type de PR pour chaque code PDB

PDB code	type	remarque
3s45	PR2	
1hhp	PR1	
1hih	PR1	
1hii	PR2	
1hiv	PR1	
1hvp	PR1	
1hsh	PR2	
1hsi	PR2	
1ivp	PR2	
1sdt	PR1	(peu avoir des pbl de numérotation des résidus)
2hb3	PR1	
2hb4	PR1	(monomère)
2hpe	PR2	
2hpf	PR2	
2ien	PR1	(peu avoir des pbl de numérotation des résidus)
2mip	PR2	
2nph	PR1	(peu avoir des pbl de numérotation des résidus)
3phv	PR1	(monomère)
2z4o	PR1	(peu avoir des pbl de numérotation des résidus)
3ebz	PR2	(peu avoir des pbl de numérotation des résidus)
3ec0	PR2	(peu avoir des pbl de numérotation des résidus)
3ecg	PR2	(peu avoir des pbl de numérotation des résidus)
3ekv	PR1	
3nu3	PR1	(peu avoir des pbl de numérotation des résidus)
4hla	PR1	
4ll3	PR1	
1hsh	PR2	

```
Proteases = c("1hhp","1hih","1hii","1hiv","1hvp","1hsh","1hsi","1ivp","1sdt","2hb3","2hb4","2hpe","2hpf",
type = c("PR1","PR1","PR2","PR1","PR1","PR2","PR2","PR2","PR1","PR1","PR1","PR2","PR2","PR1","PR2","PR1",
names(type) = Proteases
```

Détermination des résidus impliqués dans le packing cristallin chez PR1 et PR2

Création d'une matrice sous R qui contient en lignes les protéines et en colonnes les positions. La case de la matrice contient :

- 1 quand le résidu a au moins 1 atome impliqués dans le packing dans la structure,
- 0 sinon.

Attention,

- certaines structures sont des monomères
- certaines structures ont des mauvaises numérotations des résidus. Normalement les résidus sont numérotés de 1 à 99 pour la chaîne A et la chaîne B. Certaines structures ont les résidus de la chaîne B numérotés de 101 à 109. Ainsi le résidu 101B = 1B

1. Etape 1 : récupération de la liste des fichiers PDB

```
listFile = dir("fileByProt3/")
```

2. Exemple sur le premier fichier

```
filein = listFile[1]
M = read.table(paste("fileByProt3",filein,sep="/"))
listAtom = unique(paste(as.character(M[,6]), as.character(M[,5]), sep="_"))
listAtomSyn = listAtom
listAtomSyn
```

```
[1] "1_A" "2_A" "3_A" "4_A" "5_A" "6_A" "7_A" "8_A" "9_A" "23_A" "24_A" "25_A" "26_A" "27_A"
[43] "98_A" "99_A"
```

3. Récupération de la liste des résidus pour chaque protéine

```
NbProt =(1:26)
NbResidues =(1:159)

for (i in NbProt){
  filein = listFile[i]
  M = read.table(paste("fileByProt3",filein,sep="/"))
  listAtom = unique(paste(as.character(M[,6]), as.character(M[,5]), sep="_"))
  listAtomSyn = unique(c(listAtom,listAtomSyn))
}

length(listAtomSyn)
```

```
[1] 159
```

Dans les 26 structures, il y a 159 atomes qui sont impliqués dans le packing cristallin dans au moins une structure.

4. Création de la matrice

```
matrice <- matrix(0, nrow=length(Proteases), ncol=length(listAtomSyn))
rownames(matrice) <- Proteases
colnames(matrice) <- listAtomSyn

for (i in 1:length(NbProt)) {
  filein = listFile[i]
  M = read.table(paste("fileByProt3",filein,sep="/"))
  Atome <- unique(c(paste(as.character(M[,6]), as.character(M[,5]), sep="_")))
  for (j in 1:length(NbResidues)) {
    for (y in 1:length(Atome)) {
      if ((listAtomSyn[j]) == (Atome[y])){
        matrice[i,j] = 1
      }
    }
  }
}

residue = colnames(matrice)
resA = residue[grepl("_A", residue)]
resA_ssA = as.numeric(gsub("_A","", resA))
names(resA_ssA) = resA

sort.resA = names(sort(resA_ssA))
```

```

resB = residue[grep("_B", residue)]
resB_ssB = as.numeric(gsub("_B","", resB))
names(resB_ssB) = resB

sort.resB = names(sort(resB_ssB))

sort.res = c(sort.resA, sort.resB)
matrice.sort = matrice[,sort.res]

matrice = matrice.sort
dim(matrice)

```

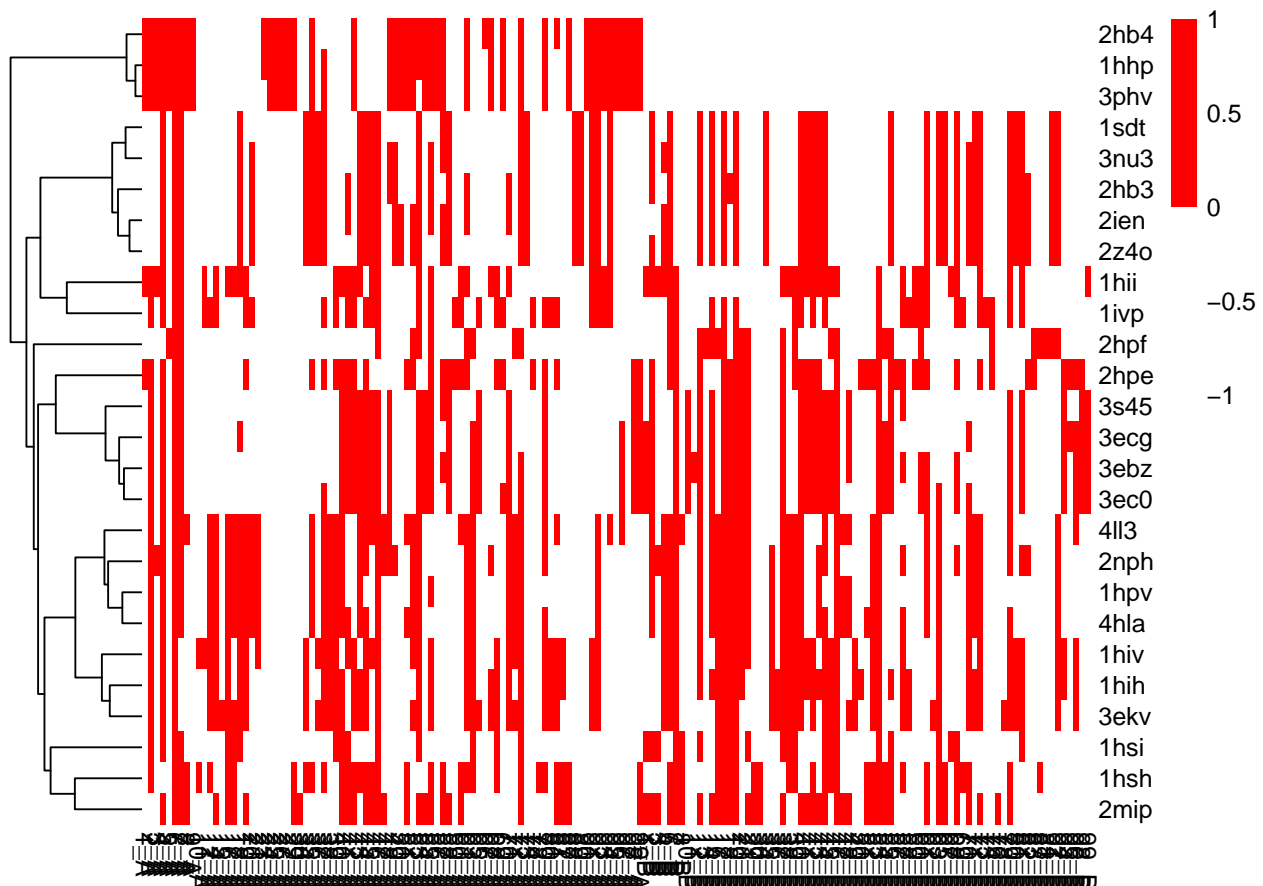
```
[1] 26 159
```

5. Coloration de la matrice b Utilisation la commande pheatmap du package pheatmap

```

pheatmap(matrice[-27:-29,], cluster_rows = TRUE, cluster_cols = FALSE, br=-1:1, col=c("white", "red"),
         clustering_distance_rows = "binary")

```



Les protéines sont classées suivant la distance 'binary' (aka asymmetric binary): The vectors are regarded as binary bits, so non-zero elements are 'on' and zero elements are 'off'. The distance is the proportion of bits in which only one is on amongst those in which at least one is on.

Analyse :

On voit tout d'abord :

- les 3 formes non complexées de PR1 sont isolées ce qui s'explique par le fait qu'elles n'ont pas de chaîne B
 - on n'a pas une séparation PR1-PR2
 - On remarque des positions qui sont impliquées dans le packing chez la majorité des structures ex 4, 6A, 7A
6. Représentation sur une structure 3D des résidus impliqués dans packing chez au moins une structure.
→ liste des résidus dans le packing dans au moins une structure de PR2 sous pymol :
7. tous les résidus impliqués dans le packing dans au moins une structure de PR prendre les colonnes de la matrice (matrice)
- ouvrir la structure de PR2 complexée avec le darunavir (DRV, code PDB 3EBZ).
 - Représenter la structure en cartoon : couleur ???
 - les résidus impliqués dans le packing en rouge

```
resSelA <- gsub("_A", "", colnames(matrice)[grep("_A", colnames(matrice))])
resSel2A <- paste(resSelA, collapse="+")
paste("select respackA, resid ", resSel2A, " and chain A", sep="")
```

```
[1] "select respackA, resid 1+2+3+4+5+6+7+8+9+10+11+12+14+15+16+17+18+19+20+21+23+24+25+26+27+29+30+34+35+36+37+38+39+40+41+42+43+44+45+46+47+48+49+50+51+52+53+54+55+56+57+58+59+60+61+62+63+64+65+66+67+68+69+70+71+72+73+74+75+76+77+78+79+80+81+82+83+84+85+86+87+88+89+90+91+92+93+94+95+96+97+98+99+100+101+102+103+104+105+106+107+108+109+110+111+112+113+114+115+116+117+118+119+120+121+122+123+124+125+126+127+128+129+130+131+132+133+134+135+136+137+138+139+140+141+142+143+144+145+146+147+148+149+150+151+152+153+154+155+156+157+158+159+160+161+162+163+164+165+166+167+168+169+170+171+172+173+174+175+176+177+178+179+180+181+182+183+184+185+186+187+188+189+190+191+192+193+194+195+196+197+198+199+200+201+202+203+204+205+206+207+208+209+210+211+212+213+214+215+216+217+218+219+220+221+222+223+224+225+226+227+228+229+230+231+232+233+234+235+236+237+238+239+240+241+242+243+244+245+246+247+248+249+250+251+252+253+254+255+256+257+258+259+260+261+262+263+264+265+266+267+268+269+270+271+272+273+274+275+276+277+278+279+280+281+282+283+284+285+286+287+288+289+290+291+292+293+294+295+296+297+298+299+300+301+302+303+304+305+306+307+308+309+310+311+312+313+314+315+316+317+318+319+320+321+322+323+324+325+326+327+328+329+330+331+332+333+334+335+336+337+338+339+340+341+342+343+344+345+346+347+348+349+350+351+352+353+354+355+356+357+358+359+360+361+362+363+364+365+366+367+368+369+370+371+372+373+374+375+376+377+378+379+380+381+382+383+384+385+386+387+388+389+390+391+392+393+394+395+396+397+398+399+400+401+402+403+404+405+406+407+408+409+410+411+412+413+414+415+416+417+418+419+420+421+422+423+424+425+426+427+428+429+430+431+432+433+434+435+436+437+438+439+440+441+442+443+444+445+446+447+448+449+450+451+452+453+454+455+456+457+458+459+460+461+462+463+464+465+466+467+468+469+470+471+472+473+474+475+476+477+478+479+480+481+482+483+484+485+486+487+488+489+490+491+492+493+494+495+496+497+498+499+500+501+502+503+504+505+506+507+508+509+510+511+512+513+514+515+516+517+518+519+520+521+522+523+524+525+526+527+528+529+530+531+532+533+534+535+536+537+538+539+540+541+542+543+544+545+546+547+548+549+550+551+552+553+554+555+556+557+558+559+560+561+562+563+564+565+566+567+568+569+570+571+572+573+574+575+576+577+578+579+580+581+582+583+584+585+586+587+588+589+590+591+592+593+594+595+596+597+598+599+600+601+602+603+604+605+606+607+608+609+610+611+612+613+614+615+616+617+618+619+620+621+622+623+624+625+626+627+628+629+630+631+632+633+634+635+636+637+638+639+640+641+642+643+644+645+646+647+648+649+650+651+652+653+654+655+656+657+658+659+660+661+662+663+664+665+666+667+668+669+670+671+672+673+674+675+676+677+678+679+680+681+682+683+684+685+686+687+688+689+690+691+692+693+694+695+696+697+698+699+700+701+702+703+704+705+706+707+708+709+710+711+712+713+714+715+716+717+718+719+720+721+722+723+724+725+726+727+728+729+730+731+732+733+734+735+736+737+738+739+740+741+742+743+744+745+746+747+748+749+750+751+752+753+754+755+756+757+758+759+760+761+762+763+764+765+766+767+768+769+770+771+772+773+774+775+776+777+778+779+780+781+782+783+784+785+786+787+788+789+790+791+792+793+794+795+796+797+798+799+800+801+802+803+804+805+806+807+808+809+810+811+812+813+814+815+816+817+818+819+820+821+822+823+824+825+826+827+828+829+830+831+832+833+834+835+836+837+838+839+840+841+842+843+844+845+846+847+848+849+850+851+852+853+854+855+856+857+858+859+860+861+862+863+864+865+866+867+868+869+870+871+872+873+874+875+876+877+878+879+880+881+882+883+884+885+886+887+888+889+890+891+892+893+894+895+896+897+898+899+900+901+902+903+904+905+906+907+908+909+910+911+912+913+914+915+916+917+918+919+920+921+922+923+924+925+926+927+928+929+930+931+932+933+934+935+936+937+938+939+940+941+942+943+944+945+946+947+948+949+950+951+952+953+954+955+956+957+958+959+960+961+962+963+964+965+966+967+968+969+970+971+972+973+974+975+976+977+978+979+980+981+982+983+984+985+986+987+988+989+990+991+992+993+994+995+996+997+998+999+1000+1001+1002+1003+1004+1005+1006+1007+1008+1009+1010+1011+1012+1013+1014+1015+1016+1017+1018+1019+1020+1021+1022+1023+1024+1025+1026+1027+1028+1029+1030+1031+1032+1033+1034+1035+1036+1037+1038+1039+1040+1041+1042+1043+1044+1045+1046+1047+1048+1049+1050+1051+1052+1053+1054+1055+1056+1057+1058+1059+1060+1061+1062+1063+1064+1065+1066+1067+1068+1069+1070+1071+1072+1073+1074+1075+1076+1077+1078+1079+1080+1081+1082+1083+1084+1085+1086+1087+1088+1089+1090+1091+1092+1093+1094+1095+1096+1097+1098+1099+1100+1101+1102+1103+1104+1105+1106+1107+1108+1109+1110+1111+1112+1113+1114+1115+1116+1117+1118+1119+1120+1121+1122+1123+1124+1125+1126+1127+1128+1129+1130+1131+1132+1133+1134+1135+1136+1137+1138+1139+1140+1141+1142+1143+1144+1145+1146+1147+1148+1149+1150+1151+1152+1153+1154+1155+1156+1157+1158+1159+1160+1161+1162+1163+1164+1165+1166+1167+1168+1169+1170+1171+1172+1173+1174+1175+1176+1177+1178+1179+1180+1181+1182+1183+1184+1185+1186+1187+1188+1189+1190+1191+1192+1193+1194+1195+1196+1197+1198+1199+1200+1201+1202+1203+1204+1205+1206+1207+1208+1209+1210+1211+1212+1213+1214+1215+1216+1217+1218+1219+1220+1221+1222+1223+1224+1225+1226+1227+1228+1229+1230+1231+1232+1233+1234+1235+1236+1237+1238+1239+1240+1241+1242+1243+1244+1245+1246+1247+1248+1249+1250+1251+1252+1253+1254+1255+1256+1257+1258+1259+1260+1261+1262+1263+1264+1265+1266+1267+1268+1269+1270+1271+1272+1273+1274+1275+1276+1277+1278+1279+1280+1281+1282+1283+1284+1285+1286+1287+1288+1289+1290+1291+1292+1293+1294+1295+1296+1297+1298+1299+1300+1301+1302+1303+1304+1305+1306+1307+1308+1309+1310+1311+1312+1313+1314+1315+1316+1317+1318+1319+1320+1321+1322+1323+1324+1325+1326+1327+1328+1329+1330+1331+1332+1333+1334+1335+1336+1337+1338+1339+1340+1341+1342+1343+1344+1345+1346+1347+1348+1349+1350+1351+1352+1353+1354+1355+1356+1357+1358+1359+1360+1361+1362+1363+1364+1365+1366+1367+1368+1369+1370+1371+1372+1373+1374+1375+1376+1377+1378+1379+1380+1381+1382+1383+1384+1385+1386+1387+1388+1389+1390+1391+1392+1393+1394+1395+1396+1397+1398+1399+1400+1401+1402+1403+1404+1405+1406+1407+1408+1409+1410+1411+1412+1413+1414+1415+1416+1417+1418+1419+1420+1421+1422+1423+1424+1425+1426+1427+1428+1429+1430+1431+1432+1433+1434+1435+1436+1437+1438+1439+1440+1441+1442+1443+1444+1445+1446+1447+1448+1449+1450+1451+1452+1453+1454+1455+1456+1457+1458+1459+1460+1461+1462+1463+1464+1465+1466+1467+1468+1469+1470+1471+1472+1473+1474+1475+1476+1477+1478+1479+1480+1481+1482+1483+1484+1485+1486+1487+1488+1489+1490+1491+1492+1493+1494+1495+1496+1497+1498+1499+1500+1501+1502+1503+1504+1505+1506+1507+1508+1509+1510+1511+1512+1513+1514+1515+1516+1517+1518+1519+1520+1521+1522+1523+1524+1525+1526+1527+1528+1529+1530+1531+1532+1533+1534+1535+1536+1537+1538+1539+1540+1541+1542+1543+1544+1545+1546+1547+1548+1549+1550+1551+1552+1553+1554+1555+1556+1557+1558+1559+1560+1561+1562+1563+1564+1565+1566+1567+1568+1569+1570+1571+1572+1573+1574+1575+1576+1577+1578+1579+1580+1581+1582+1583+1584+1585+1586+1587+1588+1589+1590+1591+1592+1593+1594+1595+1596+1597+1598+1599+1600+1601+1602+1603+1604+1605+1606+1607+1608+1609+1610+1611+1612+1613+1614+1615+1616+1617+1618+1619+1620+1621+1622+1623+1624+1625+1626+1627+1628+1629+1630+1631+1632+1633+1634+1635+1636+1637+1638+1639+1640+1641+1642+1643+1644+1645+1646+1647+1648+1649+1650+1651+1652+1653+1654+1655+1656+1657+1658+1659+1660+1661+1662+1663+1664+1665+1666+1667+1668+1669+1670+1671+1672+1673+1674+1675+1676+1677+1678+1679+1680+1681+1682+1683+1684+1685+1686+1687+1688+1689+1690+1691+1692+1693+1694+1695+1696+1697+1698+1699+1700+1701+1702+1703+1704+1705+1706+1707+1708+1709+1710+1711+1712+1713+1714+1715+1716+1717+1718+1719+1720+1721+1722+1723+1724+1725+1726+1727+1728+1729+1730+1731+1732+1733+1734+1735+1736+1737+1738+1739+1740+1741+1742+1743+1744+1745+1746+1747+1748+1749+1750+1751+1752+1753+1754+1755+1756+1757+1758+1759+1760+1761+1762+1763+1764+1765+1766+1767+1768+1769+1770+1771+1772+1773+1774+1775+1776+1777+1778+1779+1780+1781+1782+1783+1784+1785+1786+1787+1788+1789+1790+1791+1792+1793+1794+1795+1796+1797+1798+1799+1800+1801+1802+1803+1804+1805+1806+1807+1808+1809+1810+1811+1812+1813+1814+1815+1816+1817+1818+1819+1820+1821+1822+1823+1824+1825+1826+1827+1828+1829+1830+1831+1832+1833+1834+1835+1836+1837+1838+1839+1840+1841+1842+1843+1844+1845+1846+1847+1848+1849+1850+1851+1852+1853+1854+1855+1856+1857+1858+1859+1860+1861+1862+1863+1864+1865+1866+1867+1868+1869+1870+1871+1872+1873+1874+1875+1876+1877+1878+1879+1880+1881+1882+1883+1884+1885+1886+1887+1888+1889+1890+1891+1892+1893+1894+1895+1896+1897+1898+1899+1900+1901+1902+1903+1904+1905+1906+1907+1908+1909+1910+1911+1912+1913+1914+1915+1916+1917+1918+1919+1920+1921+1922+1923+1924+1925+1926+1927+1928+1929+1930+1931+1932+1933+1934+1935+1936+1937+1938+1939+1940+1941+1942+1943+1944+1945+1946+1947+1948+1949+1950+1951+1952+1953+1954+1955+1956+1957+1958+1959+1960+1961+1962+1963+1964+1965+1966+1967+1968+1969+1970+1971+1972+1973+1974+1975+1976+1977+1978+1979+1980+1981+1982+1983+1984+1985+1986+1987+1988+1989+1990+1991+1992+1993+1994+1995+1996+1997+1998+1999+2000+2001+2002+2003+2004+2005+2006+2007+2008+2009+2010+2011+2012+2013+2014+2015+2016+2017+2018+2019+2020+2021+2022+2023+2024+2025+2026+2027+2028+2029+2030+2031+2032+2033+2034+2035+2036+2037+2038+2039+2040+2041+2042+2043+2044+2045+2046+2047+2048+2049+2050+2051+2052+2053+2054+2055+2056+2057+2058+2059+2060+2061+2062+2063+2064+2065+2066+2067+2068+2069+2070+2071+2072+2073+2074+2075+2076+2077+2078+2079+2080+2081+2082+2083+2084+2085+2086+2087+2088+2089+2090+2091+2092+2093+2094+2095+2096+2097+2098+2099+2100+2101+2102+2103+2104+2105+2106+2107+2108+2109+2110+2111+2112+2113+2114+2115+2116+2117+2118+2119+2120+2121+2122+2123+2124+2125+2126+2127+2128+2129+2130+2131+2132+2133+2134+2135+2136+2137+2138+2139+2140+2141+2142+2143+2144+2145+2146+2147+2148+2149+2150+2151+2152+2153+2154+2155+2156+2157+2158+2159+2160+2161+2162+2163+2164+2165+2166+2167+2168+2169+2170+2171+2172+2173+2174+2175+2176+2177+2178+2179+2180+2181+2182+2183+2184+2185+2186+2187+2188+2189+2190+2191+2192+2193+2194+2195+2196+2197+2198+2199+2200+2201+2202+2203+2204+2205+2206+2207+2208+2209+2210+2211+2212+2213+2214+2215+2216+2217+2218+2219+2220+2221+2222+2223+2224+2225+2226+2227+2228+2229+2230+2231+2232+2233+2234+2235+2236+2237+2238+2239+2240+2241+2242+2243+2244+2245+2246+2247+2248+2249+2250+2251+2252+2253+2254+2255+2256+2257+2258+2259+2260+2261+2262+2263+2264+2265+2266+2267+2268+2269+2270+2271+2272+2273+2274+2275+2276+2277+2278+2279+2280+2281+2282+2283+2284+2285+2286+2287+2288+2289+2290+2291+2292+2293+2294+2295+2296+2297+2298+2299+2300+2301+2302+2303+2304+2305+2306+2307+2308+2309+2310+2311+2312+2313+2314+2315+2316+2317+2318+2319+2320+2321+2322+2323+2324+2325+2326+2327+2328+2329+2330+2331+2332+2333+2334+2335+2336+2337+2338+2339+2340+2341+2342+2343+2344+2345+2346+2347+2348+2349+2350+2351+2352+2353+2354+2355+2356+2357+2358+2359+2360+2361+2362+2363+2364+2365+2366+2367+2368+2369+2370+2371+2372+2373+2374+2375+2376+2377+2378+2379+2380+2381+2382+2383+2384+2385+2386+2387+2388+2389+2390+2391+2392+2393+2394+2395+2396+2397+2398+2399+2400+2401+2402+2403+2404+2405+2406+2407+2408+2409+2410+2411+2412+2413+2414+2415+2416+2417+2418+2419+2420+2421+2422+2423+2424+2425+2426+2427+2428+2429+2430+2431+2432+2433+2434+2435+2436+2437+2438+2439+2440+2441+2442+2443+2444+2445+2446+2447+2448+2449+2450+2451+2452+2453+2454+2455+2456+2457+2458+2459+2460+2461+2462+2463+2464+2465+2466+2467+2468+2469+2470+2471+2472+2473+2474+2475+2476+2477+2478+2479+2480+2481+2482+2483+2484+2485+2486+2487+2488+2489+2490+2491+2492+2493+2494+2495+2496+2497+2498+2499+2500+2501+2502+2503+2504+2505+2506+2507+2508+2509+2510+2511+2512+2513+2514+2515+2516+2517+2518+2519+2520+2521+2522+2523+2524+2525+2526+2527+2528+2529+2530+2531+2532+2533+2534+2535+2536+2537+2538+2539+2540+2541+2542+2543+2544+2545+2546+2547+2548+2549+2550+2551+2552+2553+2554+2555+2556+2557+2558+2559+2560+2561+2562+2563+2564+2565+2566+2567+2568+2569+2570+2571+2572+2573+2574+2575+2576+2577+2578+2579+2580+2581+2582+2583+2584+2585+2586+2587+2588+2589+2590+2591+2592+2593+2594+2595+2596+2597+2598+2599+2600+2601+2602+2603+2604+2605+2606+2607+2608+2609+2610+2611+2612+2613+2614+2
```

```

listAtomSyn2 = c()

for (i in NbProt){
  filein = listFile[i]
  N = read.table(paste("fileByProt3",filein,sep="/"))
  listAtom = (paste(as.character(N[,3])))
  listAtomSyn2 = (c(listAtom,listAtomSyn2))
}

names(listAtomSyn) = listAtomSyn2

for (j in 1:length(sort.res)) {
  for (k in 1:length(NbAtomeinPC)) {

    if (sort.res[j] == listAtomSyn[k]){

      #if (((listAtomSyn2[k]) == "C") || ((listAtomSyn2[k]) == "CA") || ((listAtomSyn2[k]) == "O"))
      if (is.element(listAtomSyn2[k], list.bk)==TRUE){

        matrice3[1,sort.res[j]] = matrice3[1,sort.res[j]] + 1
      }else{
        matrice3[2,sort.res[j]] = matrice3[2,sort.res[j]] + 1
      }
    }
  }
}

```

En moyenne on a :

```
apply(matrice3, 1, mean)
```

```

Backbone ChaîneLat
13.44025  24.54088

```

```
t.test(matrice3[1,],matrice3[2,], var.equal=TRUE, alternative="less")
```

Two Sample t-test

```

data: matrice3[1, ] and matrice3[2, ]
t = -3.9373, df = 316, p-value = 0.00005071
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
 -Inf -6.449546
sample estimates:
mean of x mean of y
 13.44025  24.54088

```

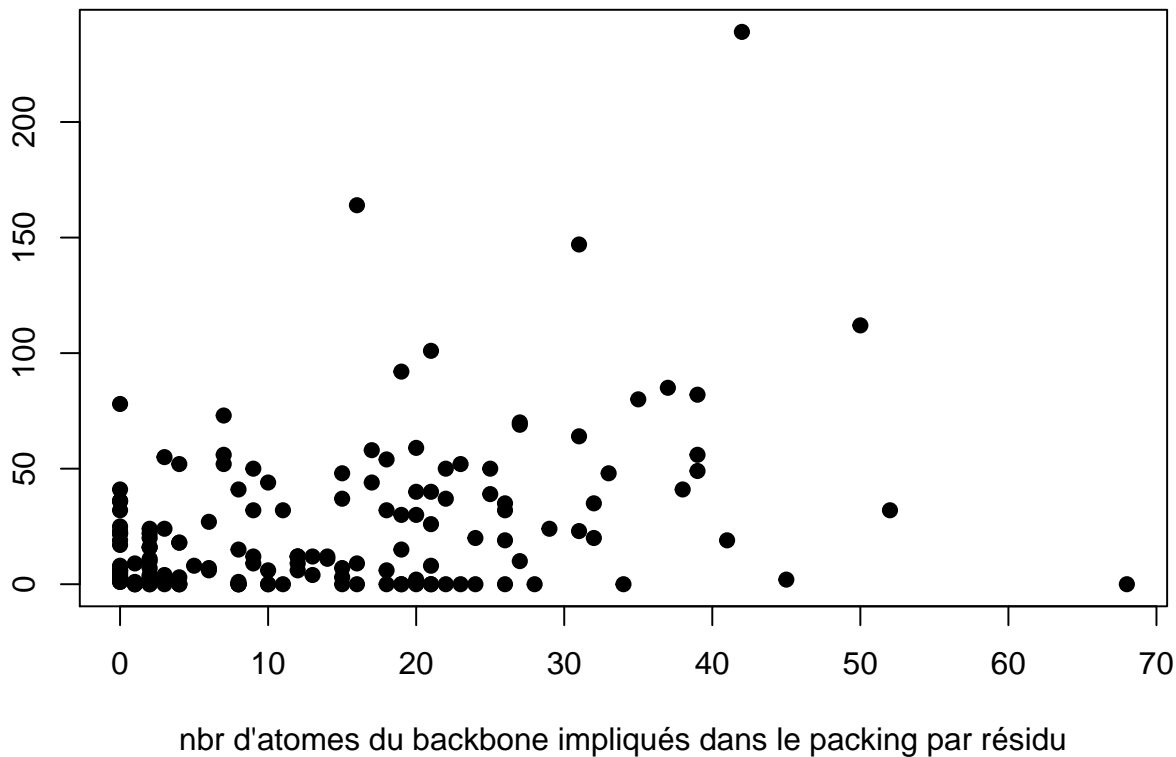
On voit qu'en moyenne les résidus ont plus d'atomes des chaînes latérales impliqués dans le packing cristallin que des atomes du backbone (p-value = 5.071e-05).

```

plot(matrice3[1,],matrice3[2,], xlab="nbr d'atomes du backbone impliqués dans le packing par résidu",
      ylab="nbr d'atomes des chaines latérales impliqués dans le packing par résidu", pch=19)

```

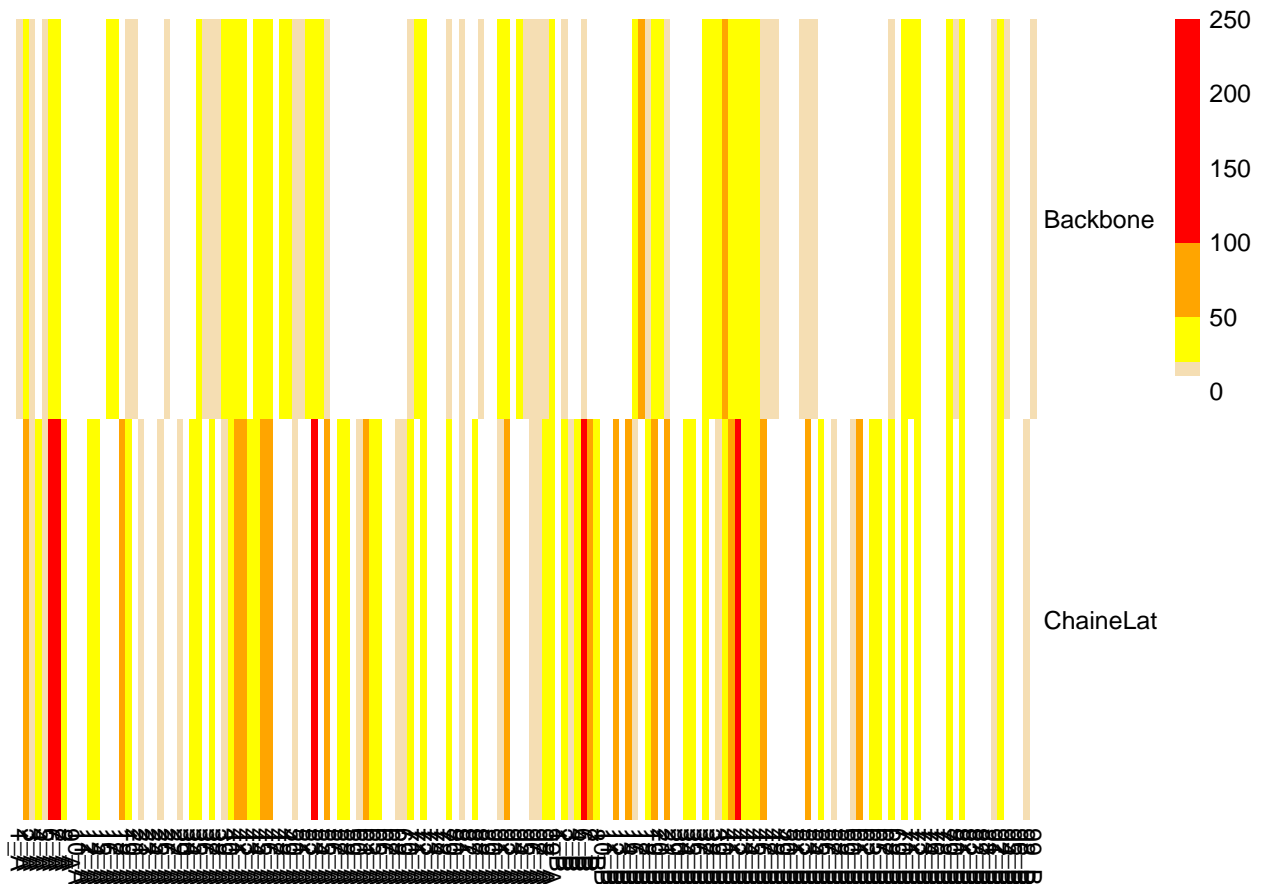
nbr d'atomes des chaînes latérales impliqués dans le packing par rés



On voit qu'il n'y a pas de lien entre le nombre d'atomes du backbone impliqués dans le packing par résidus et le nombre d'atomes des chaînes latérales impliqués dans le packing par résidus. Ce n'est pas parce qu'un résidu a beaucoup d'atomes de sa chaîne latérale impliqués dans le packing, qu'il aura beaucoup d'atomes de son backbone impliqués dans le packing.

Visualisation du type d'atomes impliqués dans le packing cristallin

```
pheatmap(matrice3, cluster_cols = FALSE, cluster_rows = FALSE, breaks = c(-1, 10, 20, 50, 100, 250),
          col = c("white", "wheat", "yellow", "orange", "red"))
```



A refaire pour le nombre de protéine : matrice avec des 0 et 1

```
matriceBackBonePR <- matrix(0, nrow=length(Proteases), ncol=length(NbResidues))

rownames(matriceBackBonePR) <- Proteases
colnames(matriceBackBonePR) <- sort.res

matriceChaineLatPR <- matrix(0, nrow=length(Proteases), ncol=length(NbResidues))

rownames(matriceChaineLatPR) <- Proteases
colnames(matriceChaineLatPR) <- sort.res

NbAtomeinPC = c(1:length(listAtomSyn2))
names(listAtomSyn) = listAtomSyn2
for (i in 1:length(Proteases)) {

  listAtomSyn = NULL
  listAtomSyn2 = NULL

  filein = listFile[i]
  N = read.table(paste("fileByProt3",filein,sep="/"))

  listAtomSyn = (paste(as.character(N[,6]), as.character(N[,5]), sep="_"))
```



```

listAtomSyn2 = (paste(as.character(N[,3])))

for (k in 1:length(listAtomSyn)) {
  for (j in 1:length(sort.res)) {

    if (listAtomSyn[k] == sort.res[j]){
      if (is.element(listAtomSyn2[k], list.bk)==TRUE){

        matriceBackBonePR[Proteases[i],sort.res[j]] = 1
      }else{
        matriceChaineLatPR[Proteases[i],sort.res[j]] = 1
      }
    }
  }
}

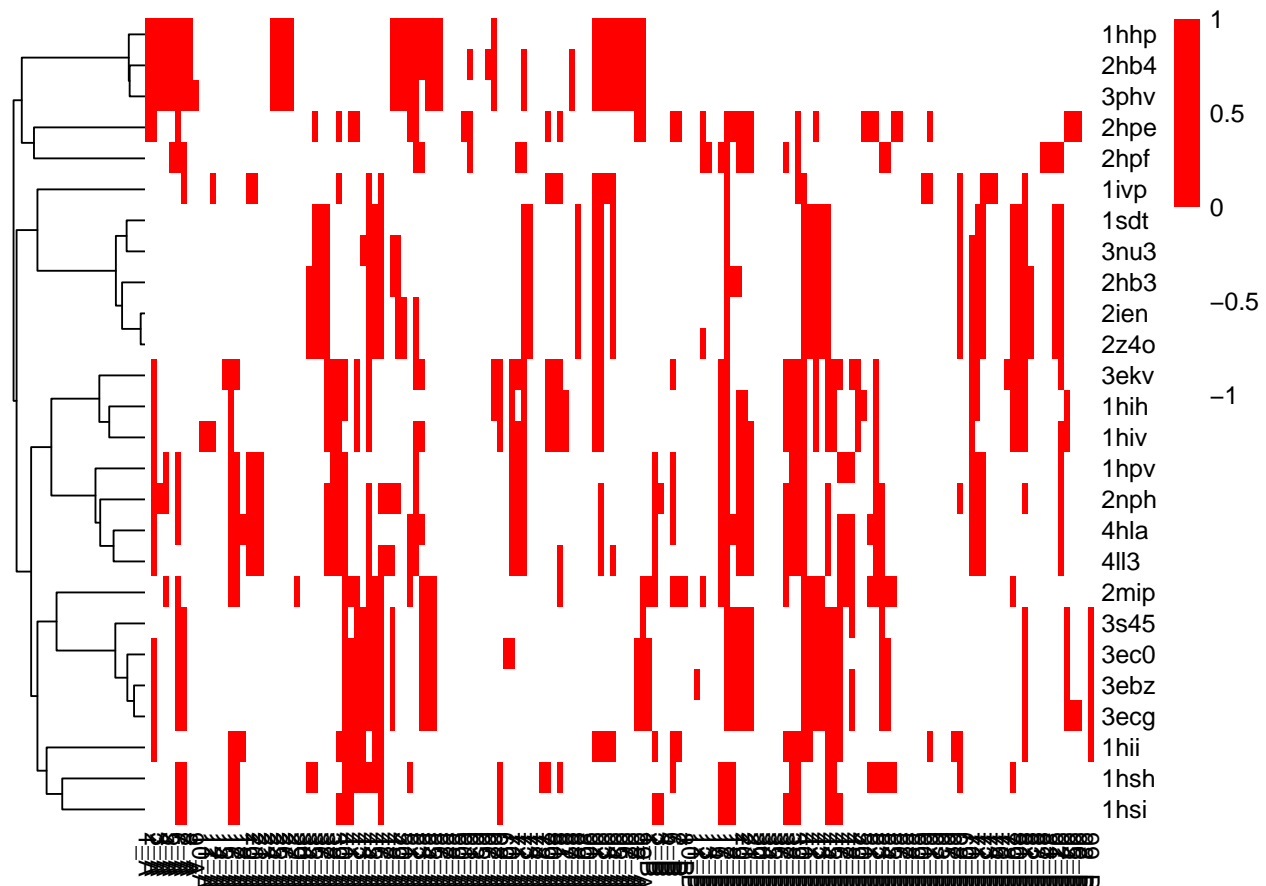
```

Visualisation des résultats

```

pheatmap(matriceBackBonePR[-27:-29,], cluster_rows = TRUE, cluster_cols = FALSE, br=-1:1,
  col=c("white", "red"), clustering_distance_rows = "binary")

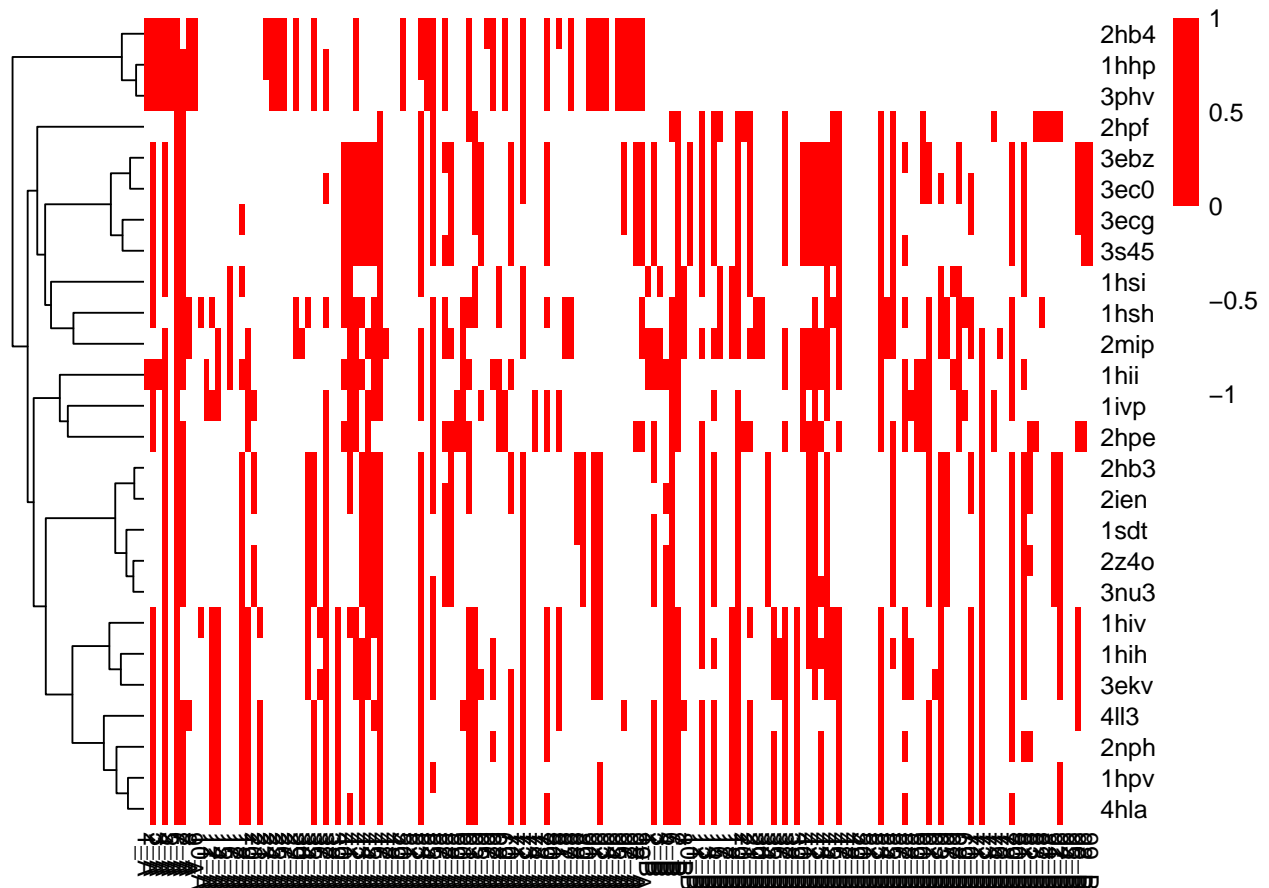
```



```

pheatmap(matriceChaineLatPR[-27:-29,], cluster_rows = TRUE, cluster_cols = FALSE, br=-1:1,
  col=c("white", "red"), clustering_distance_rows = "binary")

```



Détermination du nombre de structures dans lequel un résidu est impliqué dans le packing cristallin

Le nombre de structure dans lequel un résidu est impliqué dans le packing cristallin = la somme des colonnes de la matrice

1. Calculer le nombre de structure dans lequel un résidu est impliqué dans le packing cristallin

```
StructinPC = c("NbStructinPC", "NbStructinPC.PR1", "NbStructinPC.PR2")

matriceStruct <- matrix(0, nrow=length(StructinPC), ncol=length(sort.res))

rownames(matriceStruct) <- StructinPC
colnames(matriceStruct) <- sort.res

for (i in 1:length(NbResidues)){
  y = 0
  for (j in 1:length(Proteases[1:26])) {
    if (matrice[j,i] == 1){
      y = y+1
      matriceStruct[1,i] = y
    }
  }
}
```

```
}
```

Voir si ces deux lignes de codes sont nécessaires, car le vecteur type existe déjà mais ne contient pas les mêmes données

```
type = matrice[,ncol(matrice)]  
names(type)
```

```
[1] "1hhp" "1hih" "1hii" "1hiv" "1hvp" "1hsh" "1hsi" "1ivp" "1sdt" "2hb3" "2hb4" "2hpe" "2hpf" "2ien"
```

```
#matrice[names(which(type=="PR2")),]
```

```
matrice2 = matrice[, -ncol(matrice)]
```

```
NbStructinPC = apply(matrice[1:26,],2,sum)
```

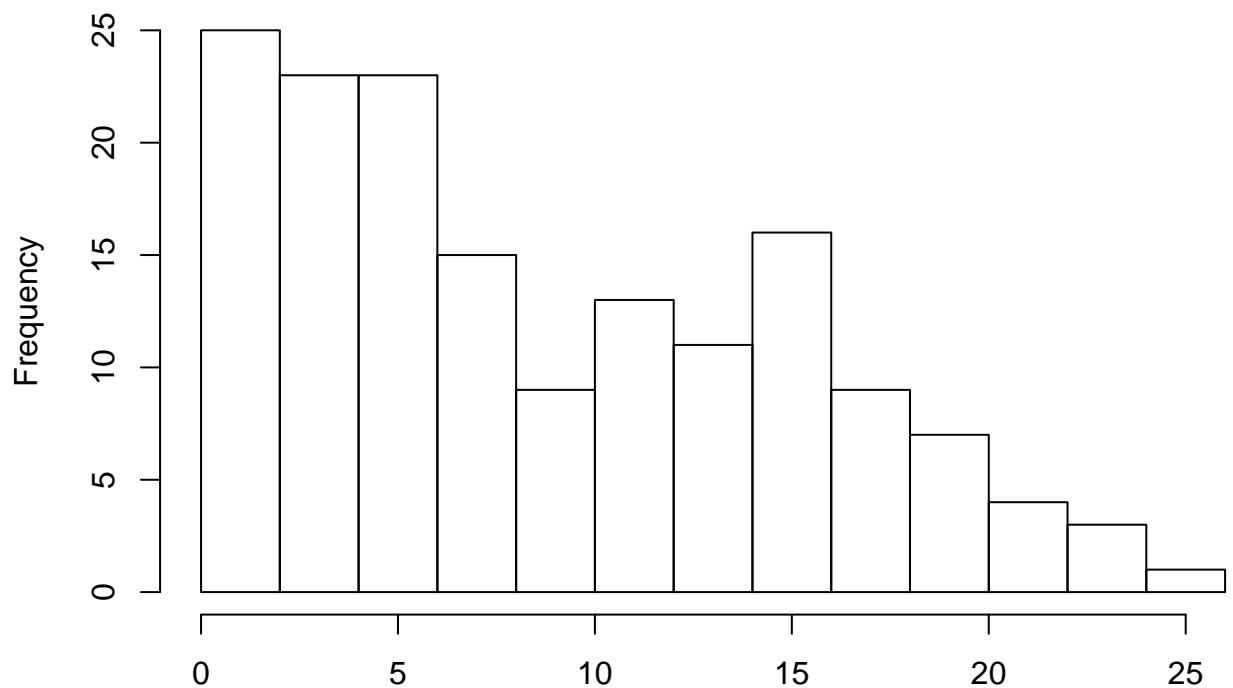
D'après ces résultats on voit que le résidu 6_A est impliqué dans le packing cristallin dans toutes les structures.

D'autres résidus sont retrouvés dans la majorité des structures (>80%) : 4_A, 6_A, 7_A, 46_A, 53_A, 72_A, 17_B, 19_B

2. Représenter ces valeurs graphiquement

```
hist(NbStructinPC,xlab="Nbr de structures ayant un résidu donné impliqué dans le packing")
```

Histogram of NbStructinPC

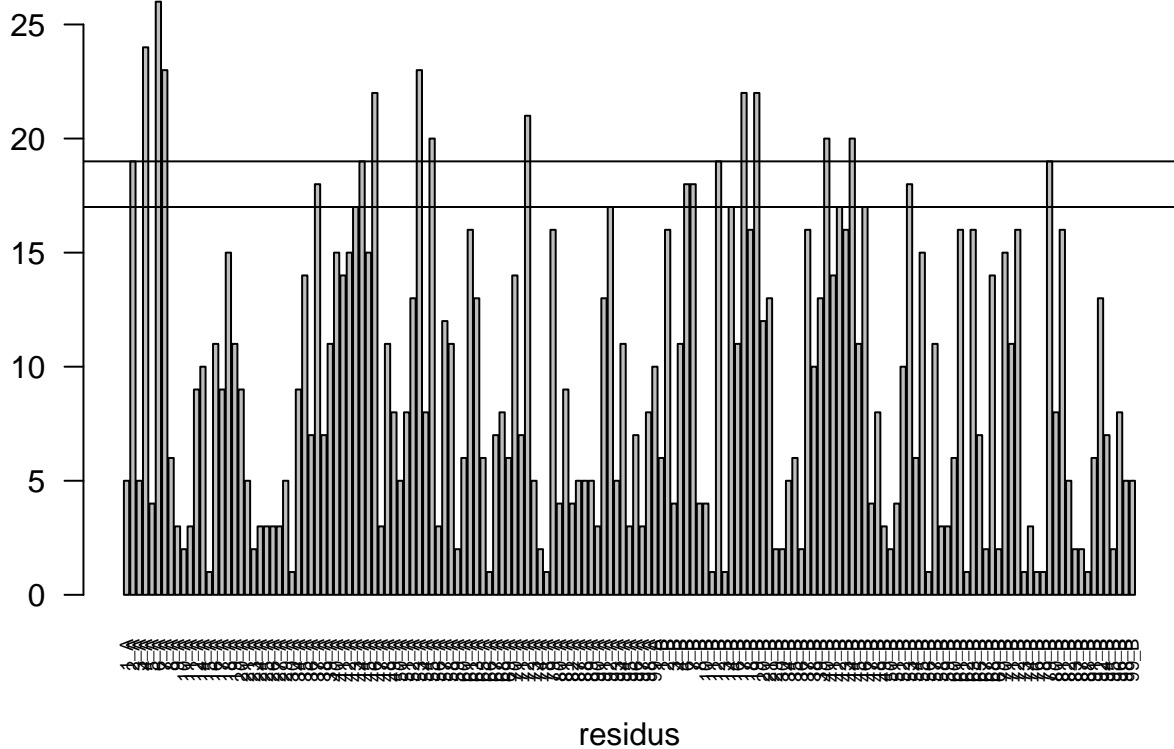


Nbr de structures ayant un résidu donné impliqué dans le packing

On va aussi faire un barplot pour voir le nombre pour chaque résidu

```
barplot(NbStructinPC, las = 2, cex.names = 0.6, xlab="residus", ylab="Nbr de structures ayant un résidu  
abline(h=c(19,17))
```

Nbr de structures ayant un résidu donné impliqué dans le packing



On voit des différences dans les chaînes A et B

3. Calculer la moyenne et écart type de ce nombre

```
mean(matriceStruct[1,])
```

```
[1] 9.157233
```

```
sd(matriceStruct[1,])
```

```
[1] 6.383855
```

On définit un résidu impliqué dans le packing dans la majorité des structures comme étant :

- les résidus de la chaîne A vu dans au moins 70% des structures = 19 structures
- les résidus de la chaîne B vu dans au moins 70% des structures = 17 structures

4. Calcul une p-value pour déterminer si les positions sont conservées

```
f.permut = function(vect){
  return(sample(vect))
}

occ <- apply(matrice,2,sum)

resA <- colnames(matrice)[grep("_A", colnames(matrice))]
resB<- colnames(matrice)[grep("_B", colnames(matrice))]
```

```

nbrSimul <- 10000

occ.SupbyPos <- rep(0, length = ncol(matrice))
names(occ.SupbyPos) <- colnames(matrice)
for( rep in 1:nbrSimul){
  mat.rand.A <- t(apply(matrice[,resA],1,f.permut))
  mat.rand.B <- t(apply(matrice[,resB],1,f.permut))
  matrice.random <- cbind(mat.rand.A, mat.rand.B)
  colnames(matrice.random) <- c(resA,resB)
  occ.rand <- apply(matrice.random,2,sum)
  diff <- occ.rand-occ
  occ.SupbyPos [names(which(diff > 0))] = occ.SupbyPos[names(which(diff > 0))] + 1
}

pval <- occ.SupbyPos/nbrSimul

```

Les positions qui sont sur-représentées sont (pvalue < 0.05) 2_A, 4_A, 6_A, 7_A, 18_A, 35_A, 37_A, 40_A, 41_A, 42_A, 43_A, 44_A, 45_A, 46_A, 52_A, 53_A, 55_A, 61_A, 63_A, 70_A, 72_A, 79_A, 91_A, 92_A, 2_B, 6_B, 7_B, 12_B, 14_B, 17_B, 18_B, 19_B, 21_B, 37_B, 39_B, 40_B, 41_B, 42_B, 43_B, 44_B, 46_B, 53_B, 55_B, 61_B, 63_B, 68_B, 70_B, 72_B, 79_B, 81_B, 92_B. Ces positions sont impliquées dans le packing dans la majorité des structures.

Détermination des résidus impliqués dans le packing cristallin chez PR1

Identification des résidus impliqués dans le packing cristallin chez au moins une structure de PR1

1. Matrice ne contenant que des protéines PR1

```
type = c("PR1","PR1","PR2","PR1","PR1","PR2","PR2","PR2","PR1","PR1","PR1","PR2","PR2","PR1","PR2","PR1")
names(type) = Proteases

ProtPR1 <- names(which(type=="PR1"))
MatricePR1 <- matrice[ProtPR1,]
```

- ## 2. Calcul de la significativité

```

nbrSimul <- 10000

f.computPval <- function(MatricePR1, nbrSimul){
  occ.PR1 <- apply(MatricePR1,2,sum)
  resA <- colnames(MatricePR1)[grep("_A", colnames(MatricePR1))]
  resB <- colnames(MatricePR1)[grep("_B", colnames(MatricePR1))]

  occ.SupbyPos.PR1 <- rep(0, length = ncol(MatricePR1))
  names(occ.SupbyPos.PR1) <- colnames(MatricePR1)

  for( rep in 1:nbrSimul){
    mat.rand.A <- t(apply(MatricePR1[,resA],1,f.permut))
    mat.rand.B <- t(apply(MatricePR1[,resB],1,f.permut))
    matrice.random <- cbind(mat.rand.A, mat.rand.B)
  }
}

```

```

    colnames(matrice.random) <- c(resA,resB)
    occ.rand <- apply(matrice.random,2,sum)
    diff <- occ.rand-occ.PR1
    occ.SupyPos.PR1 [names(which(diff > 0))] = occ.SupyPos.PR1[names(which(diff > 0))] +1
  }

  pval.PR1 <- occ.SupyPos.PR1/nbrSimul
  return(pval.PR1)
}

pval.PR1 <- f.computPval(MatricePR1, nbrSimul=20000)

length(which(pval.PR1 < 0.05/ncol(MatricePR1)))

```

```
[1] 24
```

```
res.conserv.PR1 <- names(which(pval.PR1 < 0.05))
```

3. Représentation sur une structure 3D des résidus impliqués dans le packing cristallin spécifiques de PR1

- Pour la visualisation utiliser la structure de PR1 complexée au DRV : PDB code : 2ien.
- résidus impliqués dans le packing cristallin spécifiques de PR1 = Résidus ceux qui sont très souvent impliqués dans le packing chez PR1 et très peu voir jamais chez PR2

Bien lister les résidus qui ont été colorés et comment ils ont été sélectionnés : quel seuil ?

```

resSel_A_PR1 <- gsub("_A", "", colnames(matrice)[grep("_A", colnames(matrice))])
resSel2_A_PR1 <- paste(resSel_A_PR1, collapse="+")
paste("select respackA, resid ", resSel2_A_PR1, " and chain A", sep="")

```

```
[1] "select respackA, resid 1+2+3+4+5+6+7+8+9+10+11+12+14+15+16+17+18+19+20+21+23+24+25+26+27+29+30+34+35+36+37+38+39+40+41+42+43+44+45+46+47+48+49+50+51+52+53+54+55+56+57+58+59+60+61+62+63+64+65+66+67+68+69+70+71+72+73+74+75+76+77+78+79+80+81+82+83+84+85+86+87+88+89+90+91+92+93+94+95+96+97+98+99+100"
```

```
#type[rownames(matrice)[j]] == "PR1"
```

```

count.allPR <- apply(matrice, 2, sum)
hist(count.allPR)

```

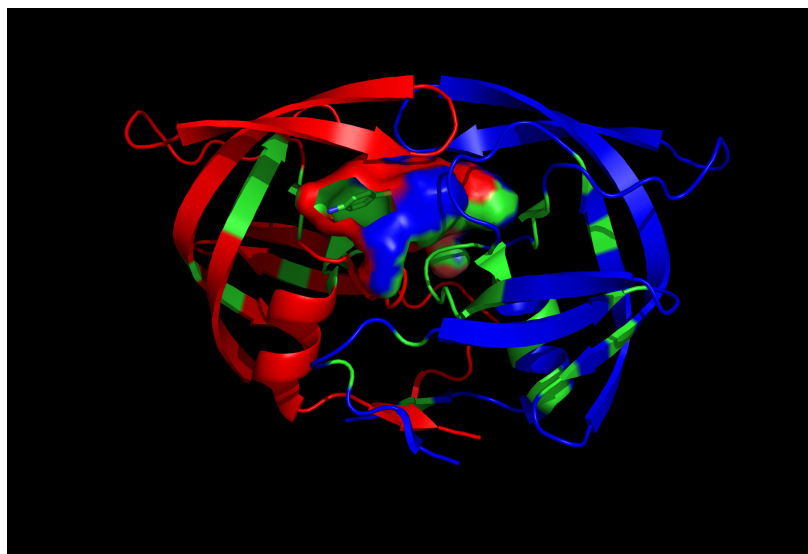
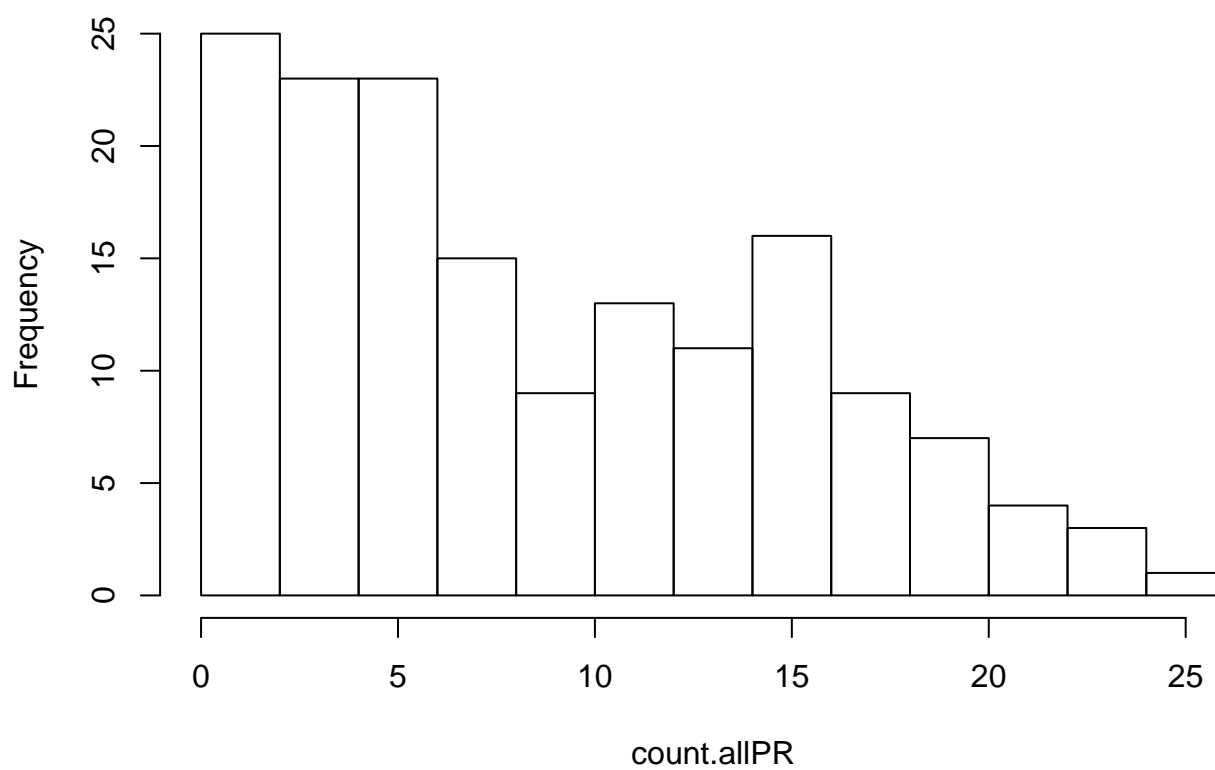


Figure 1: legende

Histogram of count.allPR



```
dim(matrice)
```

```
[1] 26 159
```

Calcul du nombre de structure dans lequel un résidu est impliqué dans le packing cristallin pour les PR1

1. Calculer le nombre de structure dans lequel un résidu est impliqué dans le packing cristallin pour les PR1

```
##version courte
NbStructinPC = apply(matrice,2,sum)

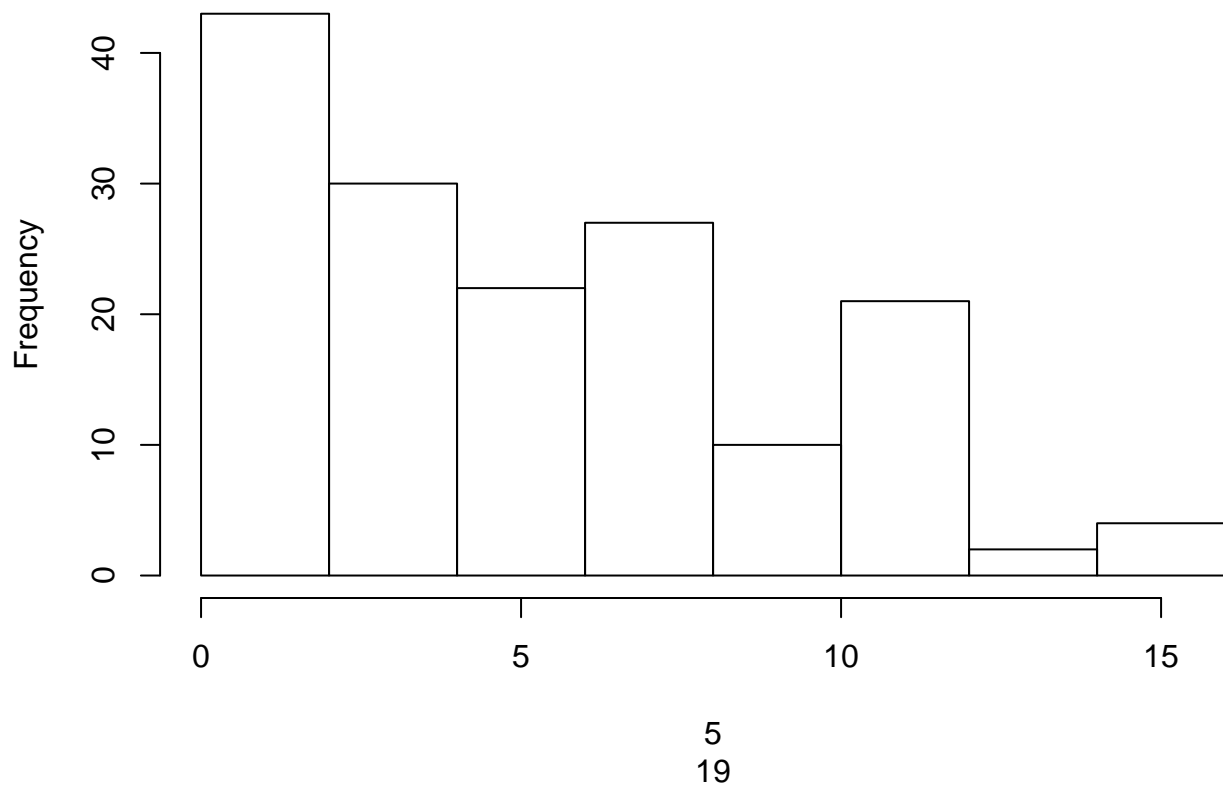
ind.PR1 = names(which(type == "PR1"))
NbStructinPC.PR1 = apply(matrice[ind.PR1,],2,sum)

matriceStruct = rbind(NbStructinPC,NbStructinPC.PR1)
```

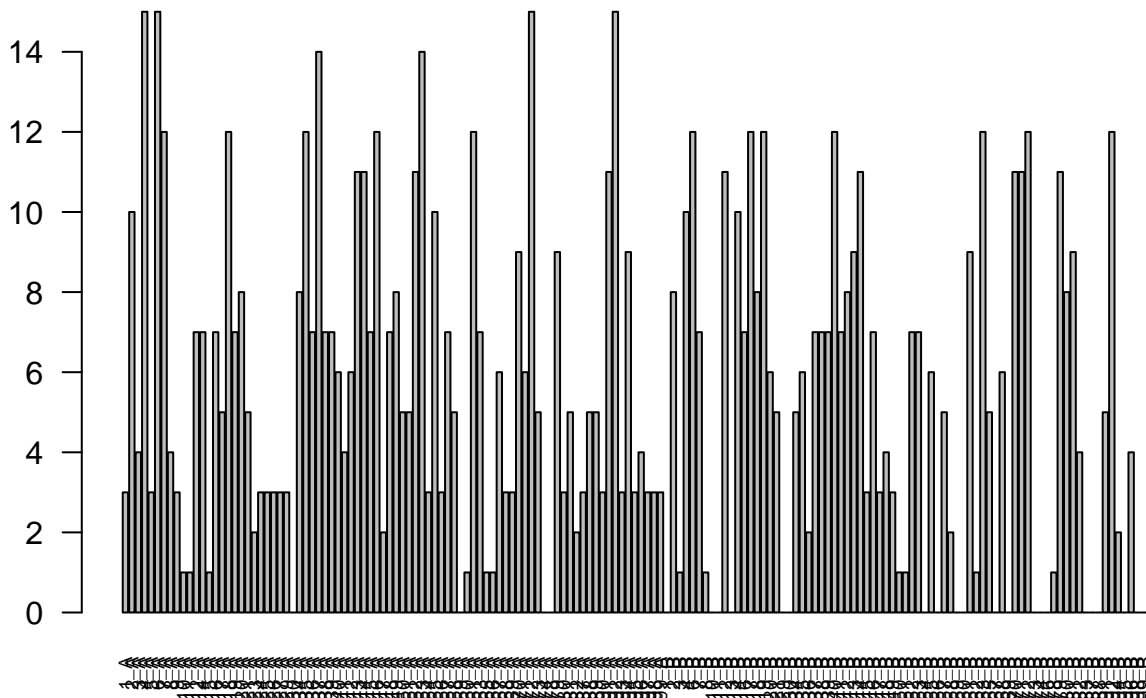
2. Représenter ces valeurs graphiquement

```
hist(NbStructinPC.PR1, xlab = matriceStruct[1,])
```

Histogram of NbStructinPC.PR1



```
barplot(NbStructinPC.PR1, las = 2, cex.names = 0.6)
```

3. Calculer la moyenne et écart type de ce nombre

```
mean(matriceStruct["NbStructinPC.PR1",])
```

```
[1] 5.408805
```

```
sd(matriceStruct["NbStructinPC.PR1",])
```

```
[1] 4.182867
```

Etude du type d'atomes impliqués dans le packing cristallin chez PR1

```
Localisation = c("Backbone","ChaineLat")
```

```
matrice3PR1 <- matrix(0, nrow=length(Localisation), ncol=length(NbResidues))
```

```
rownames(matrice3PR1) <- Localisation
```

```
colnames(matrice3PR1) <- sort.res
```

```
type.file = c("PR1","PR1","PR2","PR1","PR1","PR2","PR2","PR2","PR1","PR1","PR1","PR2","PR2","PR1","PR2")
```

```
names(type.file) = c("1hhp_packCryst.pdb","1hih_packCryst.pdb","1hii_packCryst.pdb","1hiv_packCryst.pdb")
```

```
ind.PR1 = names(which(type.file == "PR1"))
```

```
#ici je ne comprends pas pourquoi vous faites une double boucle
```

```
listAtomSyn = c()
```

```
listAtomSyn = NULL
```

```
listAtomSyn2 = NULL
```

```
pp =NULL
```

```

for (j in NbProt) {
  for (i in 1:length(ind.PR1)){
    filein = listFile[j]
    N = read.table(paste("fileByProt3",filein,sep="/"))
    if((filein) == (ind.PR1[i])){
      pp = c(pp, filein)
      listAtom = (paste(as.character(N[,6]), as.character(N[,5]), sep="_"))
      listAtomSyn = (c(listAtom,listAtomSyn))
    }
  }
}

for (j in NbProt){
  for (i in 1:length(ind.PR1)) {
    filein = listFile[j]

    if((filein) == (ind.PR1[i])){
      N = read.table(paste("fileByProt3",filein,sep="/"))
      listAtom = (paste(as.character(N[,3])))
      listAtomSyn2 = (c(listAtom,listAtomSyn2))
    }
  }
}

NbAtomeinPC.1 = c(1:length(listAtomSyn2))
names(listAtomSyn) = listAtomSyn2

for (j in 1:length(sort.res)) {
  for (k in 1:length(NbAtomeinPC.1)) {

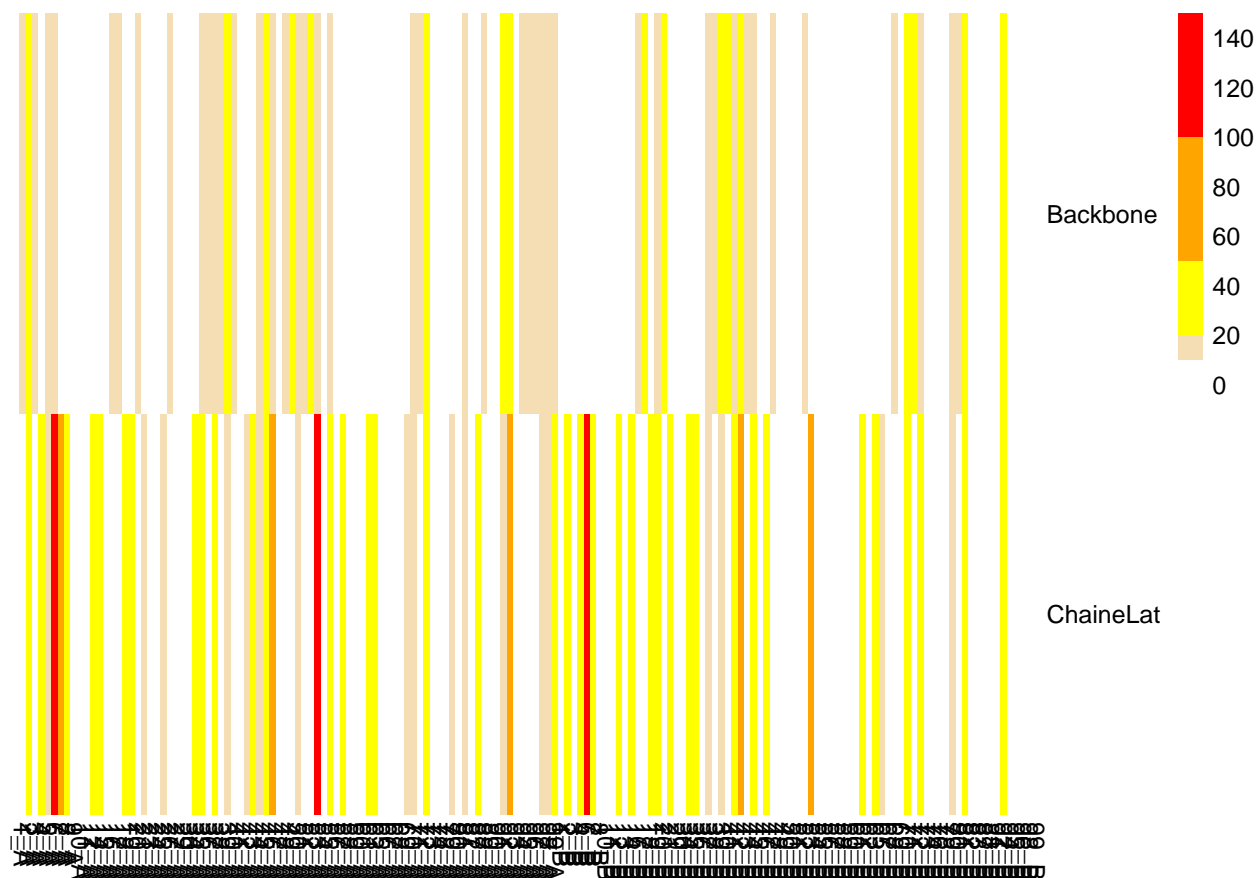
    if (sort.res[j] == listAtomSyn[k]){

      if (((listAtomSyn2)[k]) == "C") || (((listAtomSyn2)[k]) == "CA") || (((listAtomSyn2)[k]) == "O")

      matrice3PR1[1,sort.res[j]] = matrice3PR1[1,sort.res[j]] + 1
    }else{
      matrice3PR1[2,sort.res[j]] = matrice3PR1[2,sort.res[j]] + 1
    }
  }
}

pheatmap(matrice3PR1, cluster_cols = FALSE, cluster_rows = FALSE, breaks = c(-1, 10, 20, 50, 100, 150),

```



A refaire pour le nombre de protéine : matrice avec des 0 et 1 pour PR1

```
ProtPR1 = c("1hhp","1hih","1hiv","1hvp","1sdt", "2hb3","2hb4","2ien","2nph","2z4o","3ekv","3nu3","3phv")

list.bk = c("C","O","N","CA")

matriceBackBonePR1 <- matrix(0, nrow=length(ProtPR1), ncol=length(NbResidues))

rownames(matriceBackBonePR1) <- ProtPR1
colnames(matriceBackBonePR1) <- sort.res

matriceChaineLatPR1 <- matrix(0, nrow=length(ProtPR1), ncol=length(NbResidues))

rownames(matriceChaineLatPR1) <- ProtPR1
colnames(matriceChaineLatPR1) <- sort.res

NbAtomeinPC = c(1:length(listAtomSyn2))
names(listAtomSyn) = listAtomSyn2
for (i in 1:length(ind.PR1)) {

  listAtomSyn = NULL
  listAtomSyn2 = NULL

  filein = ind.PR1[i]
```

```

N = read.table(paste("fileByProt3",filein,sep="/"))

listAtomSyn = (paste(as.character(N[,6]), as.character(N[,5]), sep="_"))

listAtomSyn2 = (paste(as.character(N[,3])))

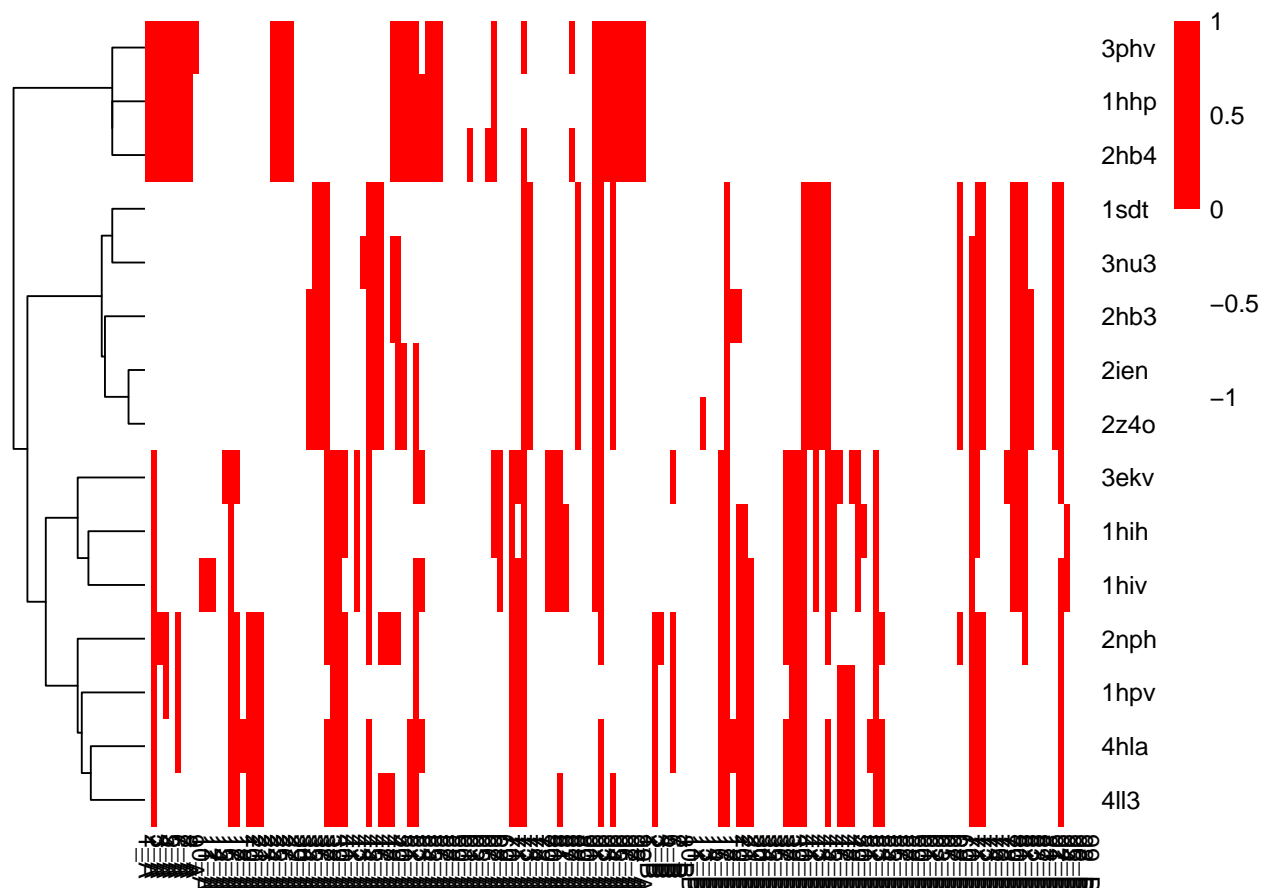
for (k in 1:length(listAtomSyn)) {
  for (j in 1:length(sort.res)) {

    if (listAtomSyn[k] == sort.res[j]){
      if (is.element(listAtomSyn2[k], list.bk)==TRUE){

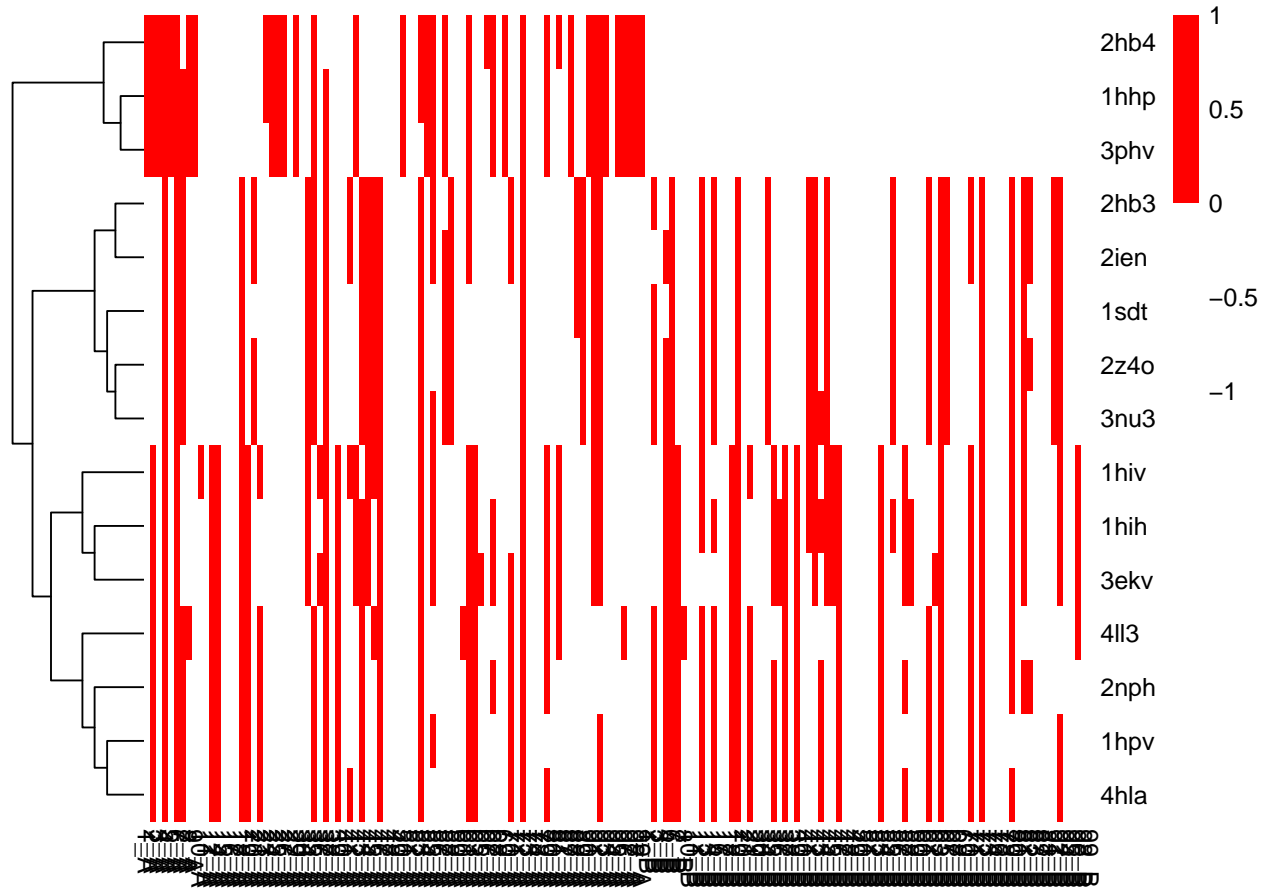
        matriceBackBonePR1[ProtPR1[i],sort.res[j]] = 1
      }else{
        matriceChaineLatPR1[ProtPR1[i],sort.res[j]] = 1
      }
    }
  }
}
}

pheatmap(matriceBackBonePR1[-27:-29,], cluster_rows = TRUE, cluster_cols = FALSE, br=-1:1, col=c("white", "black"))

```



```
heatmap(matriceChaineLatPR1[-27:-29,], cluster_rows = TRUE, cluster_cols = FALSE, br=-1:1, col=c("white", "black"))
```



Etude du packing cristallin chez PR2

To do

Identification des résidus impliqués dans le packing cristallin dans au moins une structure de PR2

1. Calcul de la matrice donnant si le résidu est impliqué dans le packing chez une des structures de PR2

```
type = c("PR1","PR1","PR2","PR1","PR1","PR2","PR2","PR2","PR1","PR1","PR1","PR2","PR2","PR1","PR2","PR1")
names(type) = Proteases
```

```
ProtPR2 = names(which(type=="PR2"))
```

```
MatricePR2 <- matrice[ProtPR2,]  
dim(MatricePR2)
```

```
[1] 11 159
```

- ## 2. Calcul de la significativité

```
pval.PR2 <- f.computPval(MatricePR2, nbrSimul=20000)
res.conserv.PR2 <- names(which(pval.PR2 < 0.05))
length(res.conserv.PR2)
```

[1] 40

3. différence entre PR1 et PR2

- résidus conservés chez PR1 et pas chez PR2

```
setdiff(res.conserv.PR1, res.conserv.PR2)
```

[1] "18_A" "35_A" "37_A" "52_A" "61_A" "70_A" "91_A" "92_A" "94_A" "4_B" "6_B" "63_B" "70_B" "71_B"

- résidus conservés chez PR2 et pas chez PR1

```
setdiff(res.conserv.PR2, res.conserv.PR1)
```

[1] "40_A" "41_A" "42_A" "45_A" "58_A" "63_A" "99_A" "7_B" "21_B" "37_B" "41_B" "45_B" "46_B" "53_B"

3. Représentation sur une structure 3D des résidus impliqués dans le packing cristallin spécifiques de PR2

- Pour la visualisation utiliser la structure de PR2 complexée au DRV : PDB code : 3EBZ.
- résidus impliqués dans le packing cristallin spécifiques de PR2 = Résidus ceux qui sont très souvent impliqués dans le packing chez PR2 et très peu voir jamais chez PR1

Bien lister les résidus qui ont été colorés et comment ils ont été sélectionnés : quel seuil ?

Calcul du nombre de structures de PR2 dans lequel un résidu est impliqué dans le packing cristallin

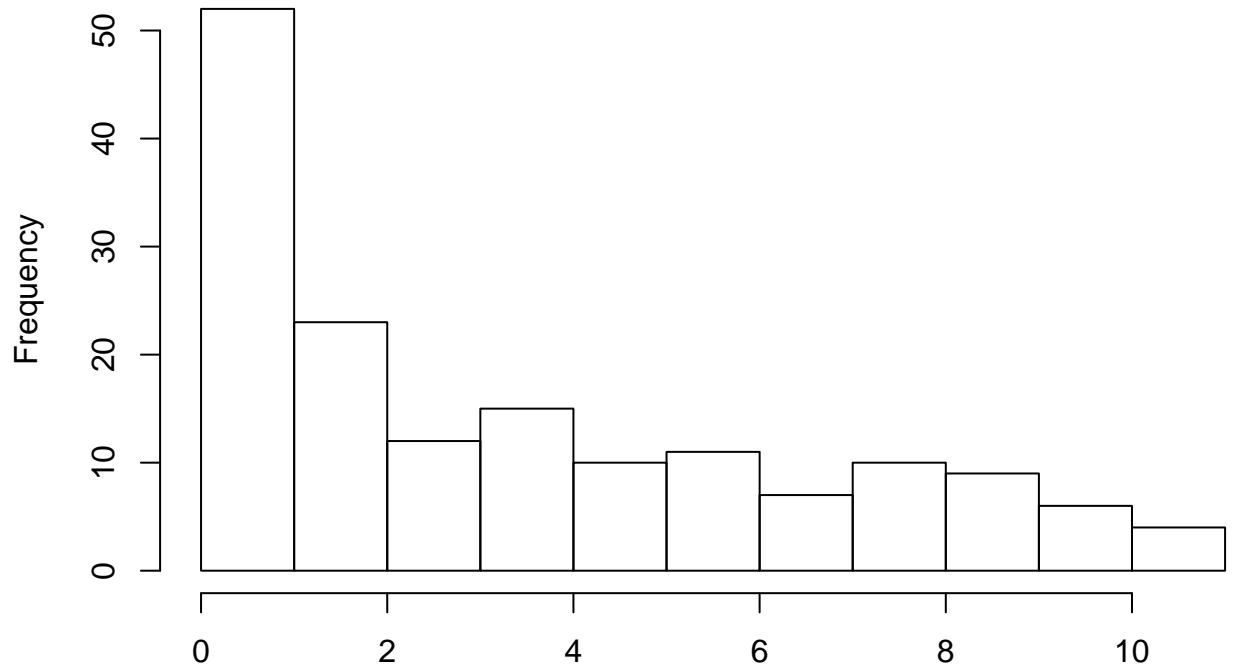
1. Calculer le nombre de structure dans lequel un résidu est impliqué dans le packing cristallin pour les PR2

```
ind.PR2 = names(which(type == "PR2"))
NbStructinPC.PR2 = apply(matrice[ind.PR2,], 2, sum)
matriceStruct = rbind(matriceStruct, NbStructinPC.PR2)
```

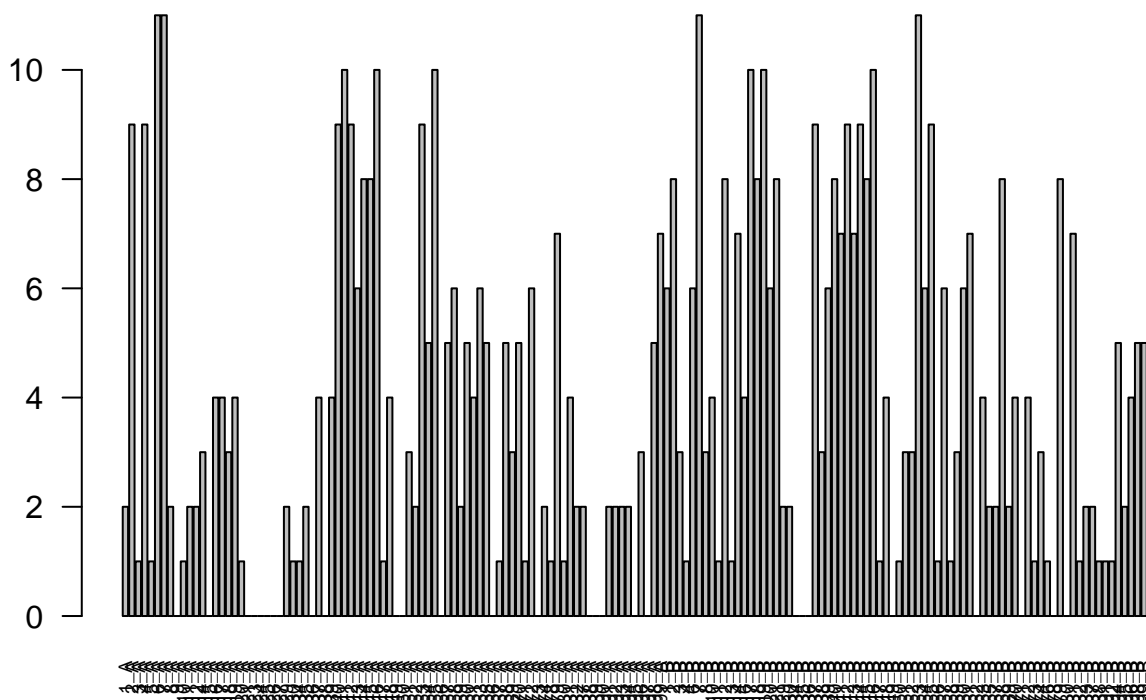
2. Représenter ces valeurs graphiquement

```
hist(NbStructinPC.PR2, xlab = "")
```

Histogram of NbStructinPC.PR2



```
barplot(NbStructinPC.PR2, las = 2, cex.names = 0.6)
```



3. Calculer la moyenne et écart type de ce nombre

```
mean(matriceStruct["NbStructinPC.PR2",])
```

```
[1] 3.748428
```

```
sd(matriceStruct["NbStructinPC.PR2",])
```

```
[1] 3.25298
```

Type d'atomes impliqués dans le packing cristallin chez PR2

```
Localisation = c("Backbone","ChaineLat")

matrice3PR2 <- matrix(0, nrow=length(Localisation), ncol=length(NbResidues))

rownames(matrice3PR2) <- Localisation
colnames(matrice3PR2) <- sort.res

ind.PR2 = names(which(type.file == "PR2"))

listAtomSyn2 = NULL
listAtomSyn = NULL

listAtomSyn = c()
for (j in NbProt) {
  for (i in 1:length(ind.PR2)){
    filein = listFile[j]
    if((filein) == (ind.PR2[i])){
      N = read.table(paste("fileByProt3",filein,sep="/"))
      listAtom = (paste(as.character(N[,6]), as.character(N[,5]), sep="_"))
      listAtomSyn = (c(listAtom,listAtomSyn))
    }
  }
}

for (j in NbProt){
  for (i in 1:length(ind.PR2)) {
    filein = listFile[j]
    if((filein) == (ind.PR2[i])){
      N = read.table(paste("fileByProt3",filein,sep="/"))
      listAtom = (paste(as.character(N[,3])))
      listAtomSyn2 = (c(listAtom,listAtomSyn2))
    }
  }
}

NbAtomeinPC.2 = c(1:length(listAtomSyn2))
list.bk = c("C","CA","N","O")
names(listAtomSyn) = listAtomSyn2

for (j in 1:length(sort.res)) {
  for (k in 1:length(NbAtomeinPC.2)) {

    if (sort.res[j] == listAtomSyn[k]){
```



```

    #if (((listAtomSyn2)[k])=="C") || (((listAtomSyn2)[k])=="CA") || (((listAtomSyn2)[k])=="O")
    if (is.element(listAtomSyn2[k], list.bk)==TRUE){

        matrice3PR2[1,sort.res[j]] = matrice3PR2[1,sort.res[j]] + 1
    }else{
        matrice3PR2[2,sort.res[j]] = matrice3PR2[2,sort.res[j]] + 1
    }
    }
}
matrice3PR2

```

	1_A	2_A	3_A	4_A	5_A	6_A	7_A	8_A	9_A	10_A	11_A	12_A	14_A	15_A	16_A	17_A	18_A	19_A	20_A	21_A	23_A
Backbone	4	6	0	1	1	25	12	0	0	0	0	1	0	0	13	18	2	2	2	0	
ChaineLat	2	22	2	19	0	93	46	11	0	1	2	2	10	0	10	0	7	11	2	0	

	68_A	69_A	70_A	71_A	72_A	73_A	74_A	78_A	79_A	80_A	81_A	82_A	87_A	88_A	89_A	90_A	91_A	92_A	93_A
Backbone	2	2	2	1	2	0	0	2	7	4	6	0	0	0	0	0	2	3	4
ChaineLat	15	4	18	0	14	0	3	0	22	0	4	3	3	0	0	0	0	0	0

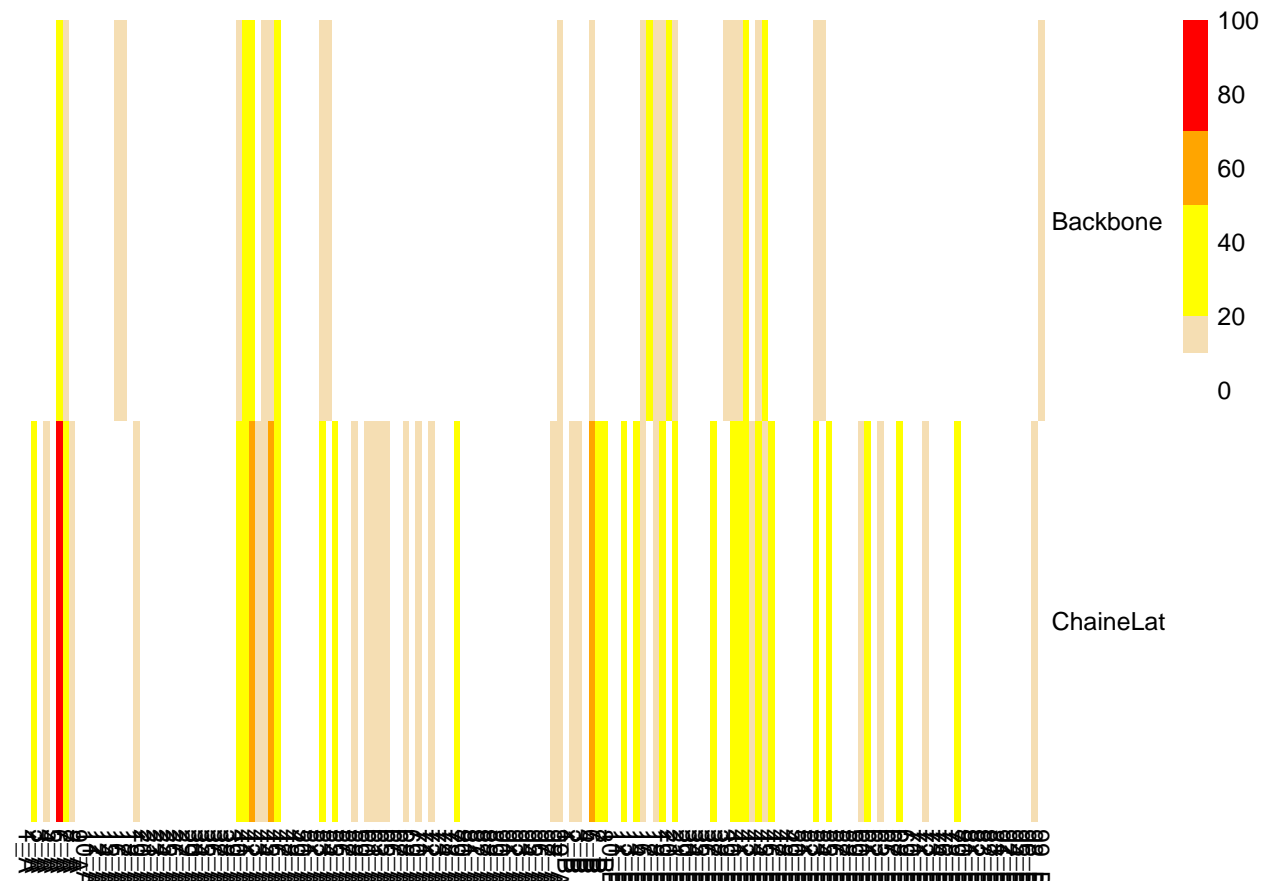
	48_B	49_B	50_B	51_B	52_B	53_B	54_B	55_B	56_B	57_B	58_B	59_B	60_B	61_B	62_B	63_B	65_B	67_B	68_B
Backbone	8	0	1	6	8	16	15	8	4	0	0	0	2	4	0	0	0	3	7
ChaineLat	0	0	0	0	0	39	7	34	0	10	3	3	16	21	0	15	7	4	30

```
matricetotale = matrice3PR1 + matrice3PR2
```

```

pheatmap(matrice3PR2, cluster_cols = FALSE, cluster_rows = FALSE, breaks = c(-1, 10, 20, 50, 70, 100),

```



A refaire pour le nombre de protéine : matrice avec des 0 et 1 pour PR2

```

ProtPR2 = c("1hii","1hsh","1hsi","1ivp","2hpe","2hpf","2mip","3ebz","3ec0","3ecg","3s45")

list.bk = c("C","O","N","CA")

matriceBackBonePR2 <- matrix(0, nrow=length(ProtPR2), ncol=length(NbResidues))

rownames(matriceBackBonePR2) <- ProtPR2
colnames(matriceBackBonePR2) <- sort.res

matriceChaineLatPR2 <- matrix(0, nrow=length(ProtPR2), ncol=length(NbResidues))

rownames(matriceChaineLatPR2) <- ProtPR2
colnames(matriceChaineLatPR2) <- sort.res

NbAtomeinPC = c(1:length(listAtomSyn2))
names(listAtomSyn) = listAtomSyn2
for (i in 1:length(ind.PR2)) {

  listAtomSyn = NULL
  listAtomSyn2 = NULL

  filein = ind.PR2[i]
  N = read.table(paste("fileByProt3",filein,sep="/"))

  listAtomSyn = (paste(as.character(N[,6]), as.character(N[,5]), sep="_"))

  listAtomSyn2 = (paste(as.character(N[,3])))

  for (k in 1:length(listAtomSyn)) {
    for (j in 1:length(sort.res)) {

      if (listAtomSyn[k] == sort.res[j]){
        if (is.element(listAtomSyn2[k], list.bk)==TRUE){

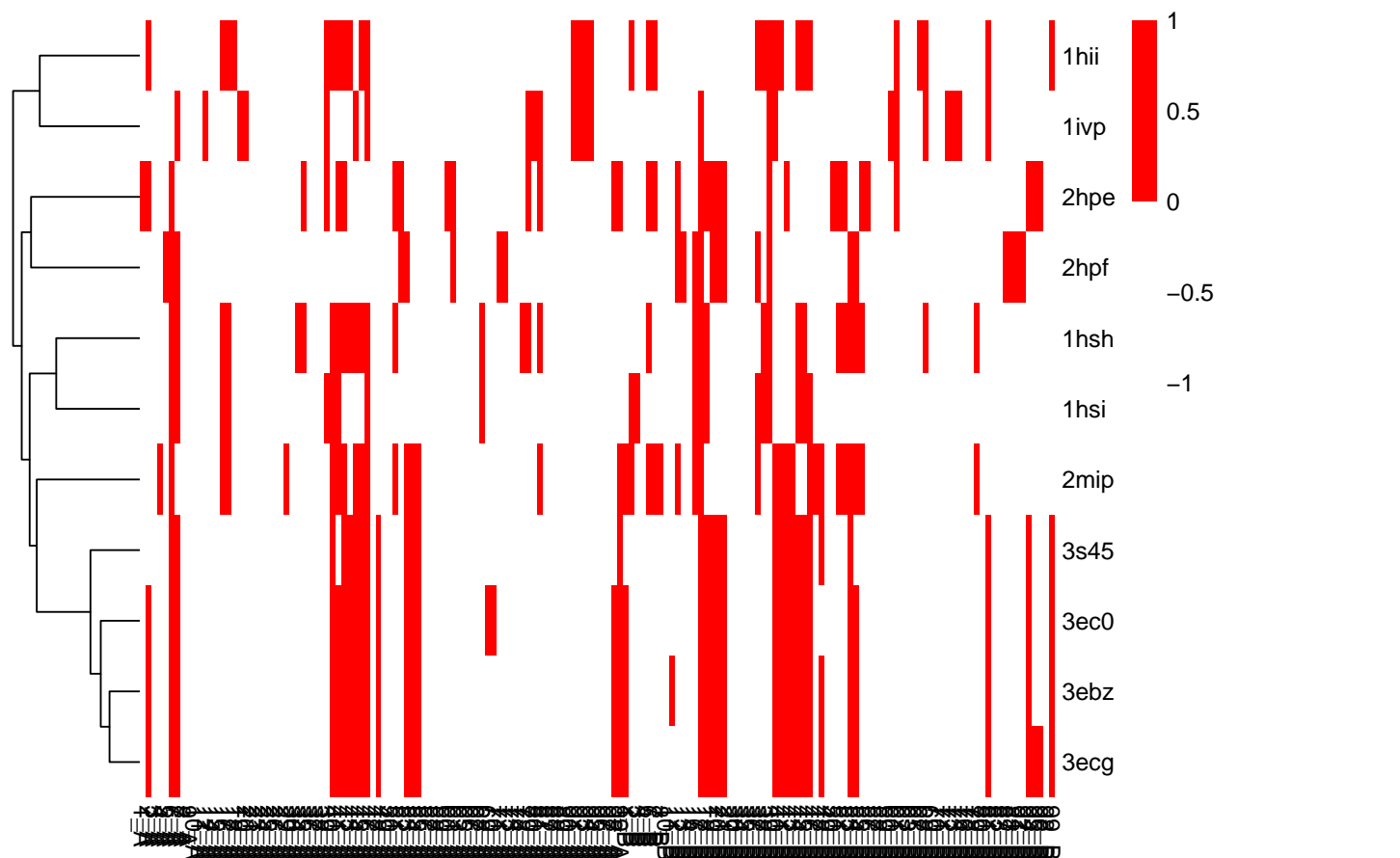
          matriceBackBonePR2[ProtPR2[i],sort.res[j]] = 1
        }else{
          matriceChaineLatPR2[ProtPR2[i],sort.res[j]] = 1
        }
      }
    }
  }
}

```

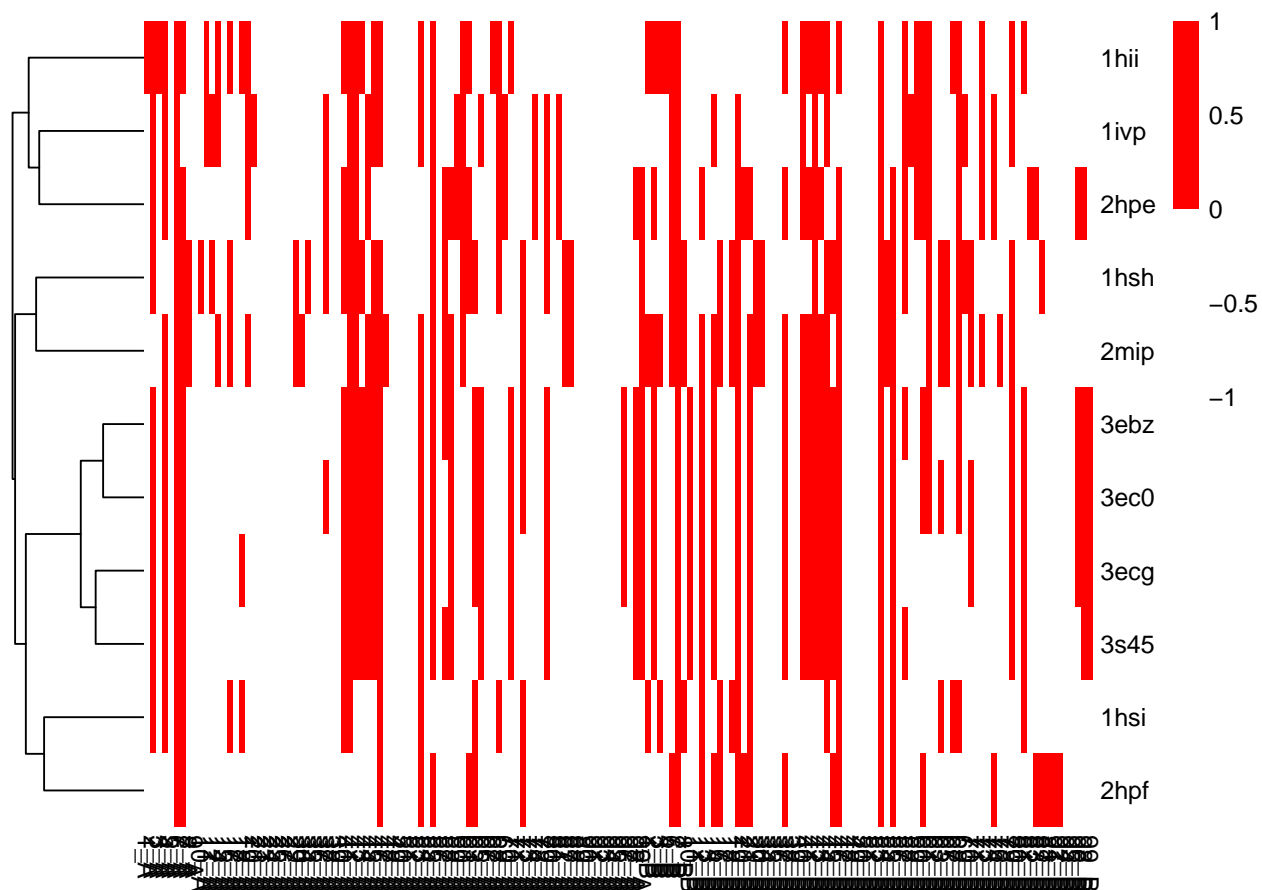
```

pheatmap(matriceBackBonePR2[-27:-29,], cluster_rows = TRUE, cluster_cols = FALSE, br=-1:1, col=c("white

```



```
pheatmap(matriceChaineLatPR2[-27:-29,], cluster_rows = TRUE, cluster_cols = FALSE, br=-1:1, col=c("white", "red"))
```

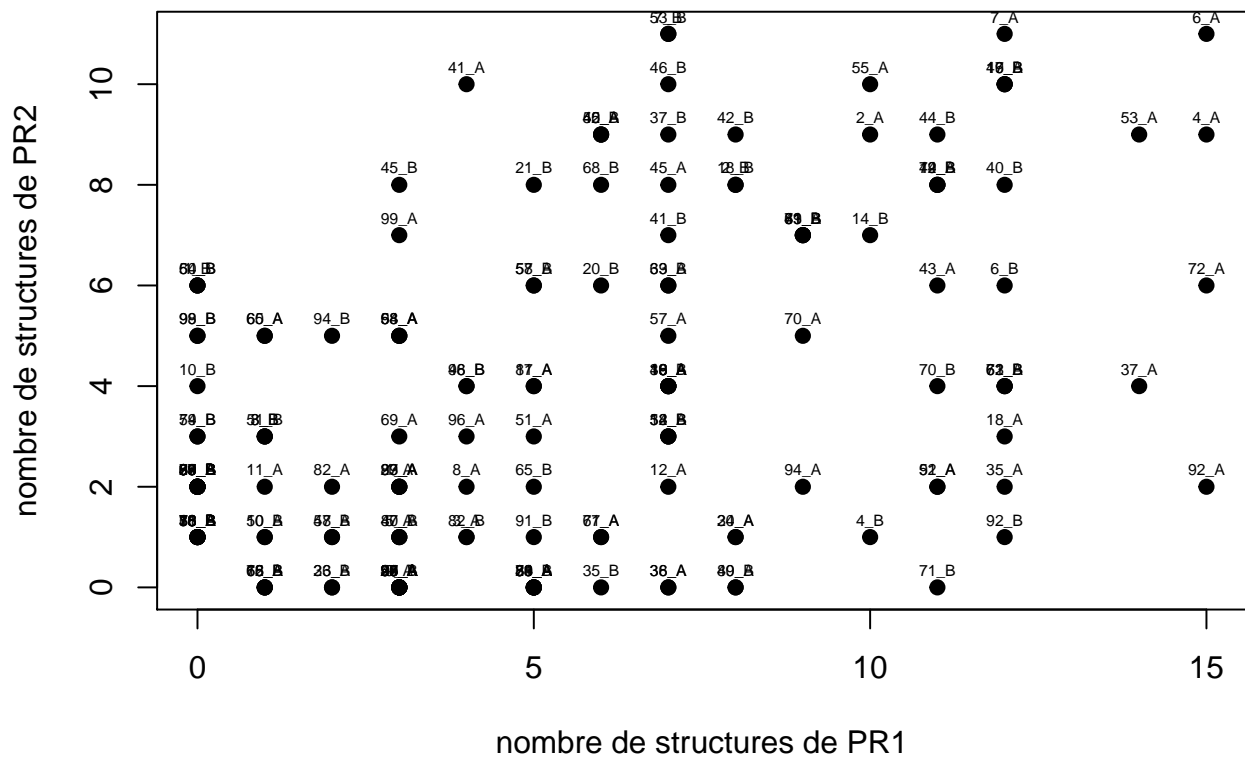


Comparaison des résidus impliqués dans le packing cristallin chez PR1 et PR2

Représentation du nombre de structures de PR1 ayant chaque résidu comme asymétrique en fonction du nombre de structures de PR2 ayant chaque résidu comme asymétrique

Refaire ce graphique en % car nombre PR1 n'est pas le même que le nombre de PR2

```
plot(matriceStruct["NbStructinPC.PR1",], matriceStruct["NbStructinPC.PR2",], pch = 19,
     xlab="nombre de structures de PR1",
     ylab="nombre de structures de PR2")
text(matriceStruct["NbStructinPC.PR1",], matriceStruct["NbStructinPC.PR2",],
     colnames(matriceStruct), pos=3, offset=0.3, cex=0.5 )
```



On calcule ensuite la corrélation entre ces deux variables

```
cor(matriceStruct["NbStructinPC.PR1",], matriceStruct["NbStructinPC.PR2",])
```

```
[1] 0.465773
```

Localisation des résidus impliqués dans le packing cristallin

Pour PR

```
RegionPacking = c("dimer1A","R1A","fulcrumA","catalyticA","R2A","elbowA","flapsA","cantileverA","R3A",
ResidueTotaux = c("1_A","2_A","3_A","4_A","5_A","6_A","7_A","8_A","9_A","10_A","11_A","12_A","13_A","14
regionfile <- read.table("description_regions.csv", sep=",")
regionV = (paste(as.character(regionfile[,2])))

for (i in 1:99) {
  regionV[i+99]=regionV[i]
}

regionSyn = NULL
for (i in 1:length(ResidueTotaux)) {
  region = (paste(as.character(regionV[i]),as.character(ResidueTotaux[i]), sep="_"))
  regionSyn[i] = region
}
names(ResidueTotaux) = regionSyn
```

```

matrice3PR <- matrix(0,nrow=length(Proteases),ncol=length(RegionPacking))
rownames(matrice3PR) <- Proteases
colnames(matrice3PR) <- RegionPacking

listAtomSyn = NULL

for (i in NbProt){
  filein = listFile[i]
  N = read.table(paste("fileByProt3",filein,sep="/"))
  listAtom = (paste(as.character(N[,6]), as.character(N[,5]), sep="_"))
  listAtomSyn = (c(listAtom,listAtomSyn))
}

listAtomSyn2 = NULL

for (i in NbProt){
  filein = listFile[i]
  N = read.table(paste("fileByProt3",filein,sep="/"))
  listAtom = (paste(as.character(N[,3])))
  listAtomSyn2 = (c(listAtom,listAtomSyn2))
}

#[grep("_A", colnames(matrice))]

names(listAtomSyn) = listAtomSyn2
for (j in 1:length(RegionPacking)) {
  for (k in 1:length(NbAtomeinPC.2)) {

    if (sort.res[j] == listAtomSyn[k]){

      matrice3PR[i,j] = 1
    }
  }
}

```

Pour PR1

Pour PR2

```

region.file <- read.table("description_regions.csv", sep=",")
as.character(region.file[4,2])

```

```
[1] "dimer"
```

```

listAtom = unique(paste(as.character(M[,6]), as.character(M[,5]), sep="_"))
listAtomSyn = unique(c(listAtom,listAtomSyn))

```

```
matrice3PR2 <- matrix(0, nrow=length(Proteases), ncol=length(NbResidues))
```

```

rownames(matrice3PR2) <- Proteases
colnames(matrice3PR2) <- sort.res

listAtomSyn = NULL
for (j in NbProt) {
  for (i in 1:length(ind.PR2)){
    filein = listFile[j]
    if((filein) == (ind.PR2[i])){
      N = read.table(paste("fileByProt3",filein,sep="/"))
      listAtom = (paste(as.character(N[,6]), as.character(N[,5]), sep="_"))
      listAtomSyn = (c(listAtom,listAtomSyn))
    }
  }
}
listAtomSyn2 = NULL

for (j in NbProt){
  for (i in 1:length(ind.PR2)) {
    filein = listFile[j]
    if((filein) == (ind.PR2[i])){
      N = read.table(paste("fileByProt3",filein,sep="/"))
      listAtom = (paste(as.character(N[,3])))
      listAtomSyn2 = (c(listAtom,listAtomSyn2))
    }
  }
}

list.bk = c("C","CA","N","O")
names(listAtomSyn) = listAtomSyn2

for (j in 1:length(sort.res)) {
  for (k in 1:length(NbAtomeinPC.2)) {

    if (sort.res[j] == listAtomSyn[k]){

      #if (((listAtomSyn2)[k]) == "C") || (((listAtomSyn2)[k]) == "CA") || (((listAtomSyn2)[k]) == "O")
      if (is.element(listAtomSyn2[k], list.bk)==TRUE){

        matrice3PR2[1,sort.res[j]] = matrice3PR2[1,sort.res[j]] + 1
      }else{
        matrice3PR2[2,sort.res[j]] = matrice3PR2[2,sort.res[j]] + 1
      }
    }
  }
}
matrice3PR2

```

1_A 2_A 3_A 4_A 5_A 6_A 7_A 8_A 9_A 10_A 11_A 12_A 14_A 15_A 16_A 17_A 18_A 19_A 20_A 21_A 23_A 24_A

1hhp	1	6	8	16	15	8	4	0	0	0	2	4	0	0	0	3	7	0	0	0
1hih	0	0	0	39	7	34	0	10	3	3	16	21	0	15	7	4	30	8	6	0
1hii	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1hiv	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1hvp	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1hsh	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1hsi	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1ivp	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1sdt	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2hb3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2hb4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2hpe	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2hpf	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2ien	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2mip	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2nph	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2z4o	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3ebz	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3ec0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3ecg	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3ekv	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3nu3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3phv	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3s45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4hla	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4l13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Etudier le lien entre la conservation des résidus impliqués dans le packing cristallin et l'espace cristallo des structures

1. Déterminer l'espace cristallographique de chaque structure en allant sur la site de la PDB (rcsb.org)

#BROUILLON

```

"1_A" "2_A" "3_A" "4_A" "5_A"      5
"6_A" "7_A" "8_A" "9_A"      4
"10_A" "11_A" "12_A" "14_A" "15_A" "16_A" "17_A" "18_A" "19_A" "20_A" "21_A" "23_A"      12
"24_A" "25_A" "26_A" "27_A" "29_A" "30_A"      6
"34_A" "35_A" "36_A"      3
"37_A" "38_A" "39_A" "40_A" "41_A" "42_A"      6
"43_A" "44_A" "45_A" "46_A" "47_A" "48_A" "49_A" "50_A" "51_A" "52_A" "53_A" "54_A" "55_A" "56_A" "57_A"
"59_A" "60_A" "61_A" "63_A" "65_A" "66_A" "67_A" "68_A" "69_A" "70_A" "71_A" "72_A" "73_A"
"74_A"      14
"78_A" "79_A"      2
"80_A" "81_A" "82_A"      3
"87_A" "88_A" "89_A" "90_A" "91_A" "92_A" "93_A" "94_A" "95_A"      9
"96_A" "97_A" "98_A" "99_A"      4

names (sort.res) = c("dimer","dimer","dimer","dimer","dimer",

```

```

      "R1","R1","R1","R1",
      "fulcrum","fulcrum","fulcrum","fulcrum","fulcrum","fulcrum","fulcrum","fulcrum","fulcrum","fulcrum","f
      ,"catalytic","catalytic","catalytic","catalytic","catalytic","catalytic"
      ,"R2","R2","R2",
      "flapelbow","flapelbow","flapelbow","flapelbow","flapelbow","flapelbow",
      "flaps","flaps","flaps","flaps","flaps","flaps","flaps","flaps","flaps","flaps","flaps","f
      "cantilever","cantilever","cantilever","cantilever","cantilever","cantilever","cantilever","can
      "R3","R3",
      "wall","wall","wall",
      "alphahelix","alphahelix","alphahelix","alphahelix","alphahelix","alphahelix","alphahelix","alp
      "dimer","dimer","dimer","dimer",)

Region2 = c("dimer","R1","fulcrum","catalytic","R2","flapelbow","flaps","cantilever","R3","wall","R4","f

"1_B"  "2_B"  "3_B"  "4_B"           4
"6_B"  "7_B"  "8_B"           3
"10_B" "11_B" "12_B" "13_B" "14_B" "16_B" "17_B" "18_B" "19_B" "20_B" "21_B"           11
"29_B" "30_B"           2
"34_B" "35_B" "36_B"           3
"37_B" "38_B" "39_B" "40_B" "41_B" "42_B"           6
"43_B" "44_B" "45_B" "46_B""47_B" "48_B" "49_B" "50_B" "51_B" "52_B" "53_B" "54_B" "55_B" "56_B" "57_B"

"59_B" "60_B" "61_B" "62_B" "63_B" "65_B" "67_B" "68_B" "69_B" "70_B" "71_B" "72_B" "73_B" "74_B"
                                10
"76_B" "78_B" "79_B"           3
"80_B" "81_B" "82_B" "83_B"           4
"87_B" "88_B" "91_B" "92_B" "94_B" "95_B"           6
"96_B" "98_B" "99_B"           3

names (sort.res) = c("dimer","dimer","dimer","dimer",
      "R1","R1","R1",
      "fulcrum","fulcrum","fulcrum","fulcrum","fulcrum","fulcrum","fulcrum","fulcrum","fulcrum","fulcrum","f
      ,"catalytic","catalytic"
      ,"R2","R2","R2",
      "flapelbow","flapelbow","flapelbow","flapelbow","flapelbow","flapelbow",
      "flaps","flaps","flaps","flaps","flaps","flaps","flaps","flaps","flaps","flaps","flaps","f
      "cantilever","cantilever","cantilever","cantilever","cantilever","cantilever","cantilever","can
      "R3","R3","R3",
      "wall","wall","wall","wall",
      "alphahelix","alphahelix","alphahelix","alphahelix","alphahelix","alphahelix","alphahelix",
      "dimer","dimer","dimer",)
test

```

