Literaturreview SAM-Analogs

The described SAM analogs in the literature were analyzed by:

a) which were tested
b) which of the tested were active

First load the required packages and load data

```
library(tidyr)
library(magrittr)
library(ggplot2)
library(stringr)
library(dplyr)

dat <- read.csv("~/litreview.csv", na.strings = c("NA", ""))
```

The data is in a type, which is not easy to work with:

```
head(dat)
```

```
##       citekey    enzymes                substrates    tested converted
## 1 Thomsen2013   PRMT1_wt        1;2;3;9;12;4;5;6;7 1;2;3;4;9     1;4;9
## 2    Wang2014       <NA>        1;9;10;11;24;25;26      <NA>      <NA>
## 3 Dalhoff2006  M.TaqI_wt                  1;2;9;10  1;2;9;10  1;2;9;10
## 4 Dalhoff2006  M.HhaI_wt                  1;2;9;10  1;2;9;10  1;2;9;10
## 5 Dalhoff2006 M.BcnIB_wt                  1;2;9;10  1;2;9;10  1;2;9;10
## 6   Singh2014    RebM_wt 1;12;13;9;2;14;10;3;28;27      <NA> 1;28;13;9
```

That is why the data need to be brought in a format, which is easier to interpret. For each substrate (keys 1:29) to occurence in each of the three cells *substrate*, *tested* and *converted* is extracted by means of `grepl()` and regular expression of the type `(((^X|\D)\D)|(X$))`, where X is the substrate number. The results of the three columns are converted into `TRUE = 1` or `FALSE = 0` and combined into a key of the form **XXX**, where **X** is either **0** or **1**.

```
for(i in 1:29){
  if(!exists("daf")){
    daf <- cbind(dat[,1:2],
                 paste(
                   grepl(
                     pattern = paste("(((^",as.character(i),"|\\D",as.character(i),")\\D)|(",as.charact
                     dat$substrates) %>% ifelse(1,0),
                   grepl(
                     pattern = paste("(((^",as.character(i),"|\\D",as.character(i),")\\D)|(",as.charact
                     dat$tested) %>% ifelse(1,0),
                   grepl(
                     pattern = paste("(((^",as.character(i),"|\\D",as.character(i),")\\D)|(",as.charact
                     dat$converted) %>% ifelse(1,0),
                   sep=""
                   ))
```

```
    }else{
      daf <- cbind(daf,
                   paste(
                     grepl(pattern = paste("(((^",as.character(i),"|\\D",as.character(i),")\\D)|(",as.ch
                           dat$substrates) %>% ifelse(1,0),
                     grepl(
                       pattern = paste("(((^",as.character(i),"|\\D",as.character(i),")\\D)|(",as.chara
                       dat$tested) %>% ifelse(1,0),
                     grepl(
                       pattern = paste("(((^",as.character(i),"|\\D",as.character(i),")\\D)|(",as.chara
                       dat$converted) %>% ifelse(1,0),
                     sep=""
                     ))
    }
  }

tmp<-expand.grid(A="S", B=1:29)
names(daf)[-c(1,2)] <- paste(tmp[,1], tmp[,2], sep=".")

head(daf[1:10])
```

```
##       citekey    enzymes S.1 S.2 S.3 S.4 S.5 S.6 S.7 S.8
## 1 Thomsen2013   PRMT1_wt 111 110 110 111 100 100 100 000
## 2    Wang2014       <NA> 100 000 000 000 000 100 000 000
## 3 Dalhoff2006  M.TaqI_wt 111 111 000 000 000 000 000 000
## 4 Dalhoff2006  M.HhaI_wt 111 111 000 000 000 000 000 000
## 5 Dalhoff2006 M.BcnIB_wt 111 111 000 000 000 000 000 000
## 6   Singh2014    RebM_wt 101 100 100 000 000 000 100 000
```

Next the **XX** keys are translated into three different classifiers C, T and NT, which mean *conversion*, *tested* and *not tested* respectively. The following rules are applied for the conversion:

- any three 0/1 combinations ending in 1 –> C
- any three 0/1 combinations ending in 1[01] –> T
- any three 0/1 combinations ending in 00 –> NT

```
daf <- apply(daf, 2, FUN = function(x){str_replace(string = x, pattern = "^[01]{2}1$", "C")}) %>% as.da
daf <- apply(daf, 2, FUN = function(x){str_replace(string = x, pattern = "^[01]{1}10$", "T")}) %>% as.d
daf <- apply(daf, 2, FUN = function(x){str_replace(string = x, pattern = "^[01]{1}00$", "NT")}) %>% as.

daf %<>% separate(enzymes, into = c("enzyme", "wt"), sep = "_", extra="drop")
daf$wt <- ifelse(daf$wt == "wt", T, F)
head(daf[1:10])
```

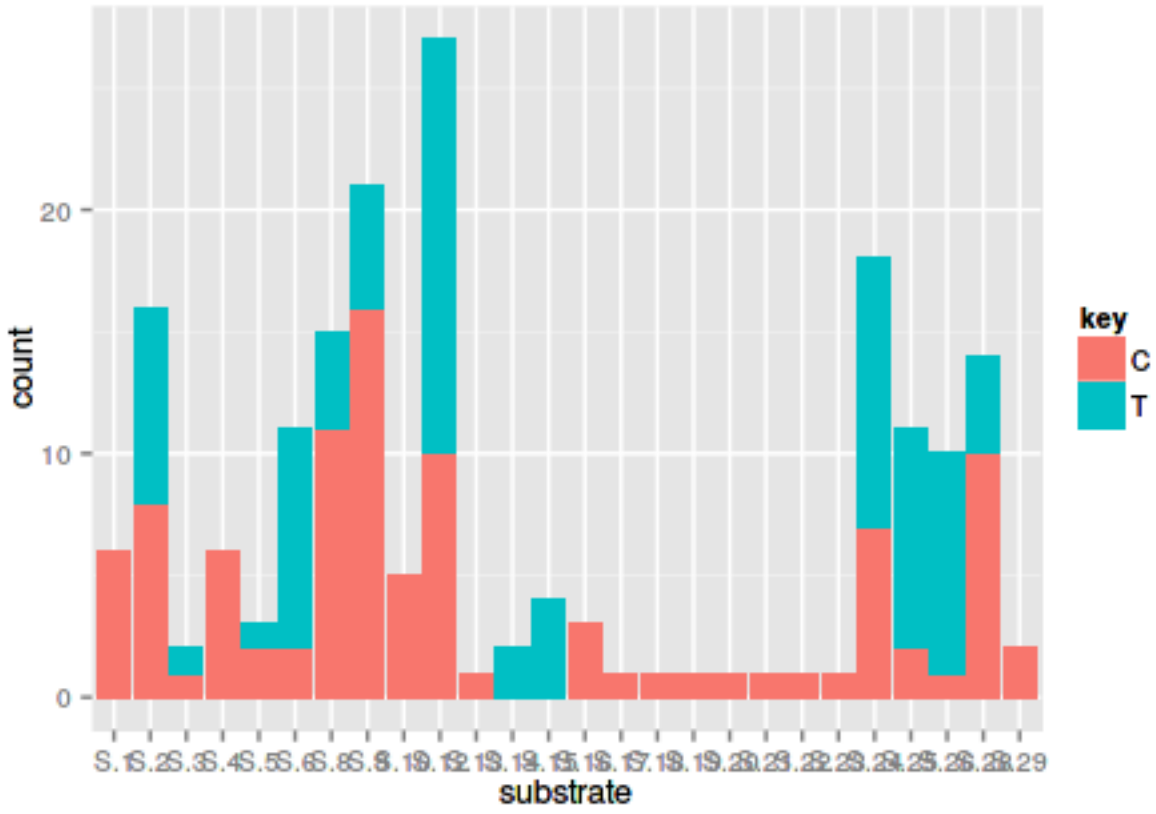
```
##       citekey  enzyme   wt S.1 S.2 S.3 S.4 S.5 S.6 S.7
## 1 Thomsen2013   PRMT1 TRUE   C   T   T   C  NT  NT  NT
## 2    Wang2014    <NA>   NA  NT  NT  NT  NT  NT  NT  NT
## 3 Dalhoff2006  M.TaqI TRUE   C   C  NT  NT  NT  NT  NT
## 4 Dalhoff2006  M.HhaI TRUE   C   C  NT  NT  NT  NT  NT
## 5 Dalhoff2006 M.BcnIB TRUE   C   C  NT  NT  NT  NT  NT
## 6   Singh2014    RebM TRUE   C  NT  NT  NT  NT  NT  NT
```

The dataframe was then brought into long format and the resukts plotted as a histogram (leaving out the NT values)

```
daf %<>% gather(substrate, key, S.1:S.29)
head(daf)
```

```
##       citekey  enzyme   wt substrate key
## 1 Thomsen2013   PRMT1 TRUE       S.1   C
## 2    Wang2014    <NA>   NA       S.1  NT
## 3 Dalhoff2006  M.TaqI TRUE       S.1   C
## 4 Dalhoff2006  M.HhaI TRUE       S.1   C
## 5 Dalhoff2006 M.BcnIB TRUE       S.1   C
## 6   Singh2014    RebM TRUE       S.1   C
```

```
daf %>% dplyr::filter(key != "NT") %>%
  ggplot(data=.) + geom_bar(aes(x=substrate, fill=key))
```



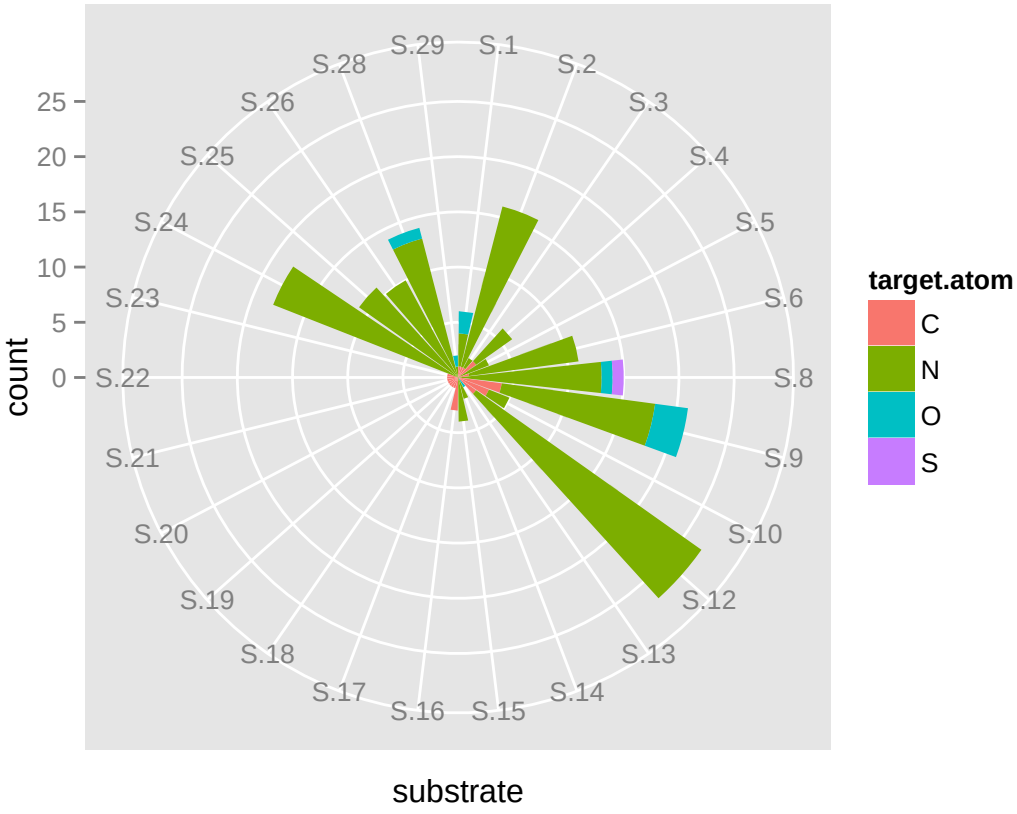
However this way is not really convenient for grasping the results. But first add the target molecule and atom to the table. Then make two separate dfs (`daf.atom` and `daf.mol`) for plotting. Split the `key` column into three separate columns for arithmetics. Calculate "*relative activity*" by dividing the number of conversions by the number of times tested.

```
hash_enz <- read.csv("~/enzyme_hash.csv")
hash_sub <- read.csv("~/substrate_hash.csv")
```

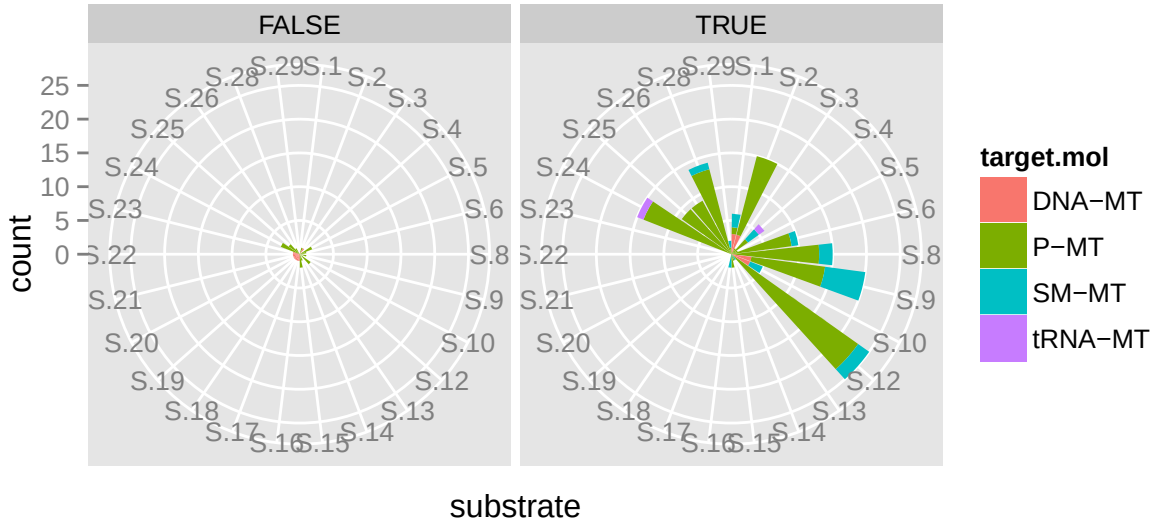
```
daf <- merge(daf, hash_enz, by.x = "enzyme", by.y = "enzyme")
daf <- merge(daf, hash_sub, by.x = "substrate", by.y = "substrate")

daf$type <- factor(daf$type, levels = c("aliphatic", "allylic", "propargylic", "aromatic", "Se-derivati
daf <- daf[order(daf$type),]

rm(hash_enz, hash_sub)
head(daf)
```

```
##   substrate enzyme      citekey   wt key target.mol target.atom  name
## 1       S.1   RebM    Singh2014 TRUE   C      SM-MT           O ethyl
## 2       S.1   GLP1 Bothwell2012 TRUE  NT       P-MT           N ethyl
## 3       S.1 PRDM16    Binda2011 TRUE  NT       P-MT           N ethyl
## 4       S.1 SETDB1    Binda2011 TRUE  NT       P-MT           N ethyl
## 5       S.1   GLP1    Wang2011a TRUE  NT       P-MT           N ethyl
## 6       S.1   TPMT      Lee2010 TRUE  NT      SM-MT           S ethyl
##        type
## 1 aliphatic
## 2 aliphatic
## 3 aliphatic
## 4 aliphatic
## 5 aliphatic
## 6 aliphatic
```

```
daf %>% dplyr::filter(key != "NT") %>%
  ggplot(data=.) + geom_bar(aes(x=substrate, fill=target.atom)) + coord_polar()
```



```
daf %>% dplyr::filter(key != "NT") %>%
  ggplot(data=.) + geom_bar(aes(x=substrate, fill=target.mol)) + coord_polar() + facet_grid(~wt)
```




```
daf.mol <- daf %>% group_by(target.mol, substrate, name, type, key, wt) %>%
  summarise(count = n())
daf.atom <- daf %>% group_by(target.atom, substrate, name, type, key, wt) %>%
  summarise(count = n())
daf %<>% group_by(target.mol, target.atom, substrate, name, type, key, wt) %>%
  summarise(count = n())


daf.atom <- tidyr::spread(daf.atom, key = key, value = count, fill = 0)
daf.mol <- tidyr::spread(daf.mol, key = key, value = count, fill = 0)
daf <- tidyr::spread(daf, key = key, value = count, fill = 0)

daf.atom %<>% mutate(T = T+C) %>% mutate(C.T = (C/T))
daf.mol %<>% mutate(T = T+C) %>% mutate(C.T = (C/T))
daf %<>% mutate(T = T+C) %>% mutate(C.T = (C/T))
```

PLot heat-map of the data:

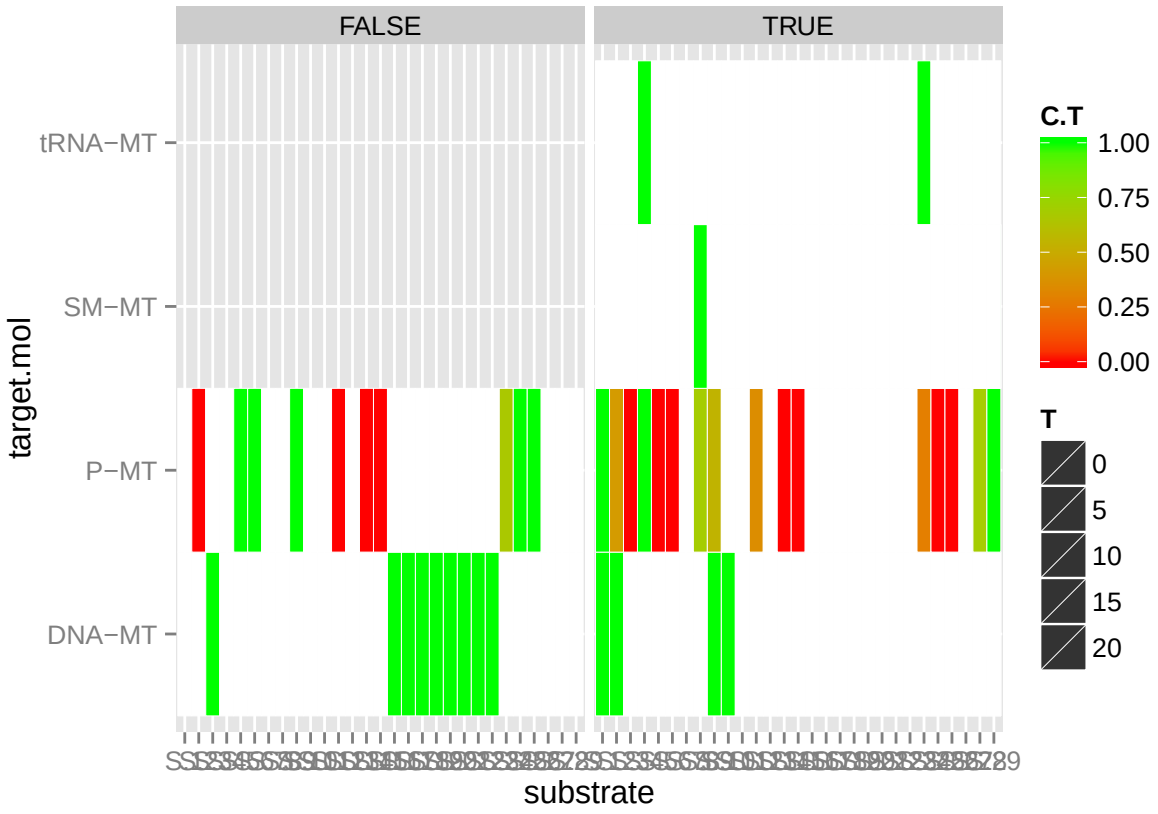
```
p <- ggplot(daf.mol, aes(target.mol, substrate)) +
  geom_tile(aes(fill=C.T, alpha=T, group=substrate), colour="white") +
  scale_fill_gradient(low = "red", high = "green", na.value="white") +
  scale_alpha_continuous(range = c(1,1)) +
  facet_grid(~wt)

base_size <- 9
p + theme_grey(base_size = base_size) +
  labs(x = "", y= "") +
  scale_x_discrete(expand=c(0,0)) +
  theme(legend.position = "none", axis.ticks = element_blank(), panel.background = element_blank())
```

PLot heat-map of the substrates vs. target.mol (heat = times tested):

```
p <- ggplot(daf, aes(target.mol, substrate)) +
  geom_tile(aes(fill=C.T, alpha=T, group=substrate), colour="white") +
  scale_fill_gradient(low = "red", high = "green", na.value="white") +
  scale_alpha_continuous(range = c(1,1)) +
  facet_grid(~wt)

base_size <- 9
p + theme_grey(base_size = base_size) +
  labs(x = "", y= "") +
  scale_x_discrete(expand=c(0,0)) +
  theme(legend.position = "none", axis.ticks = element_blank(), panel.background = element_blank())
```

Radial plot of times a substrate was tested vs. substrate. The actual times the substrates were converted is indicated by the lines.

```
daf <- daf[order(daf$type),]

newlev <- merge(data.frame(substrate=unique(daf$substrate)), (daf[,c(3,5)]), by.x = "substrate", by.y =
  unique

newlev <- newlev[order(newlev$type),]

daf$substrate <- factor(daf$substrate, levels = as.character(newlev$substrate))
nodes <- data.frame(gpmin=c(0, cumsum(summary(newlev$type)[-length(summary(newlev$type))])+0.5), gpmax=
nodes[nrow(nodes),ncol(nodes)] <- nodes[nrow(nodes),ncol(nodes)]+1
nodes$type <- levels(newlev$type)

size <- daf %>% group_by(type) %>%
  summarise(size = sum(T))

nodes %<>% mutate(mid = gpmin + (gpmax-gpmin)/2)
nodes$size <- scales::rescale(size$size,to = c(5,30))


p + coord_flip()
```




```
#p +  geom_segment(data=tmp, aes(x=substrate, xend=substrate, y=0, yend=C), color="black")
# p + geom_line(data=tmp, aes(group=group, x=substrate, y=C), color="black")
rm(tmp)
```

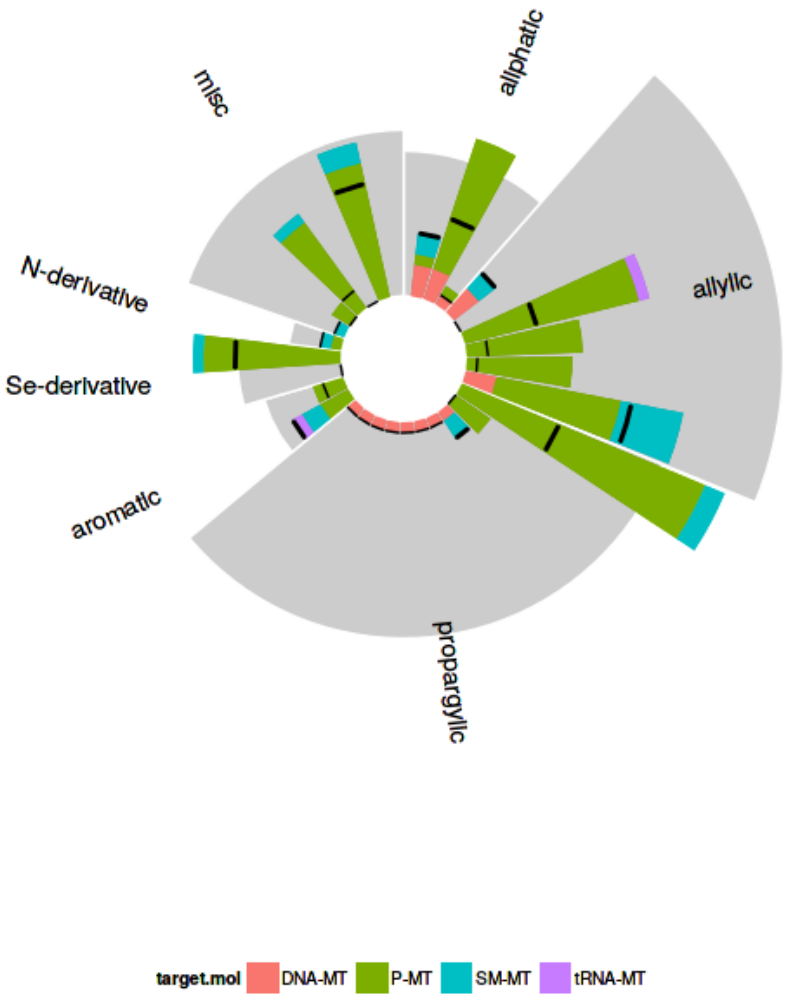
A heat map for different target molecules:

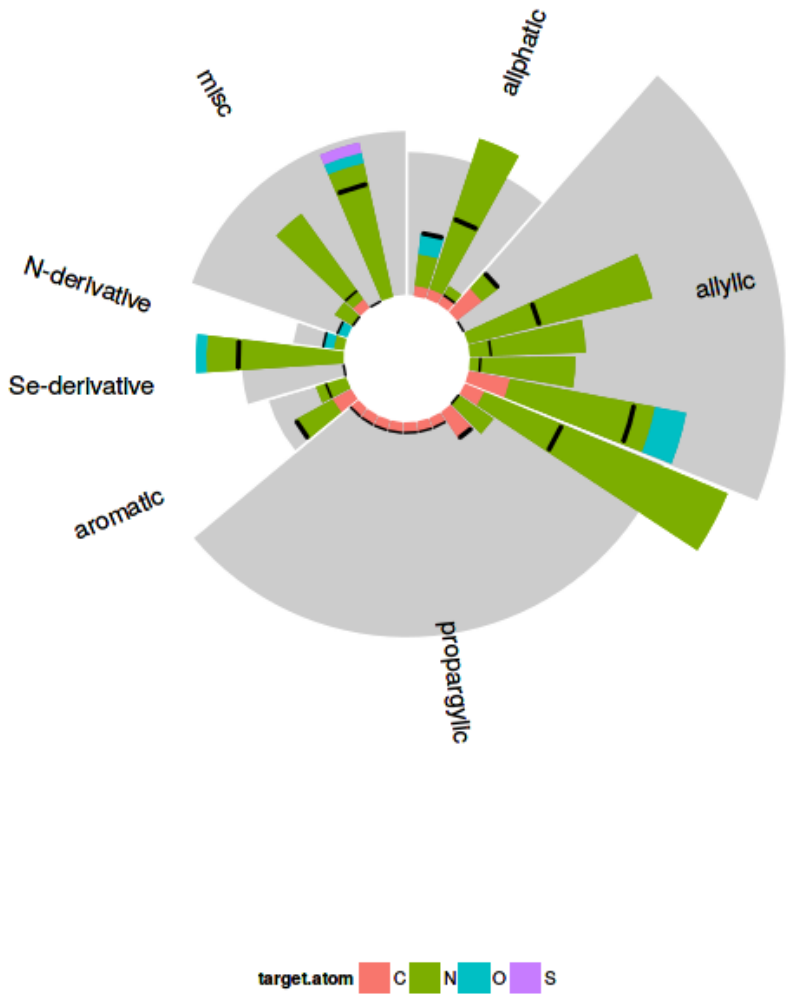
```
p <- ggplot(daf.mol, aes(target.mol, substrate)) +
  geom_point(aes(color=C.T, size=T)) +
  scale_color_gradient(low = "red", high = "green", na.value="white") +
  scale_size_continuous(range = c(5,20), na.value=0) +
  facet_grid(~wt)
```

The following chart shows the total times each substrate has been tested, together with the total number of timnes this subsbstrate was converted (black line). The height of the grey segments corresponds to the frequency a substrate from this group was used.

```
tmp <- daf %>% group_by(substrate, name, type) %>%
  summarise(C = sum(C), group="A")

polarLabs <- function(pos = nodes$mid, max = 30){
  tmp <- pos/max
  le <- (-270 - 360 * tmp[which(tmp < 0.5)])
  #le <- le-(le%/%180)*180

  ri <- (-270 - 360 * tmp[which(tmp >= 0.5)])+180
  #ri <- ri-(ri%/%180)*180
  return(c(le,ri))
```

```
  }

nodes$angles <- polarLabs()

p <- daf %>% group_by(substrate, target.mol, name, type) %>%
  summarise(T = sum(T), C = sum(C)) %>%
  ungroup %>%
  ggplot(data=.) +
  geom_bar(stat="identity", aes(x = substrate, fill=target.mol, y=T)) +
  scale_y_continuous(expand=c(0.2, 0))  +
  theme(panel.background = element_blank(),
        axis.text = element_blank(),
        axis.title = element_blank(),
        axis.ticks = element_blank(),
        legend.position = "bottom") +
  geom_rect(data=nodes, aes(xmin=gpmin, xmax = gpmax, ymin=0, ymax=size), fill="black", color="white", a
  #  geom_segment(data=nodes, aes(x=gpmin, xend = gpmin, y=0, yend=30), color="black") +
  geom_bar(stat="identity", aes(x = substrate, fill=target.mol, y=T))  +
  geom_crossbar(data=tmp, aes(x=substrate, ymin=C, ymax=C, y=C), width=0.8, color="black") +
  # geom_segment(aes(x=substrate, y=0, yend=30, xend=),)
  geom_text(data=nodes, aes(x = mid, y=25, label=type, angle=angles))
p + coord_polar()
```


aliphatic
allylic
propargylic
aromatic
Se-derivative
N-derivative
misc
target.mol
DNA-MT
P-MT
SM-MT
tRNA-MT



and one for the different target atoms:

log-scaling:

```
p <- ggplot(daf.atom, aes(target.atom, substrate)) +
  geom_point(aes(color=C.T, size=log10(T))) +
  scale_color_gradient(low = "red", high = "green", na.value="white") +
  #scale_size_manual(values=c(0,4,6)) +
  scale_size_continuous(range = c(0,20), na.value=0) +
  facet_grid(~wt)

base_size <- 9
p + theme_grey(base_size = base_size) +
  labs(x = "", y= "") +
  scale_x_discrete(expand=c(0.1,0.1)) +
```

```
  theme(legend.position = "none", axis.ticks = element_blank(), panel.background = element_blank())
```

normal scaling:

```
p <- ggplot(daf.atom, aes(target.atom, substrate)) +
  geom_point(aes(color=C.T, size=T)) +
  scale_color_gradient(low = "red", high = "green", na.value="white") +
  #scale_size_manual(values=c(0,4,6)) +
  scale_size_continuous(range = c(5,20), na.value=0) +
  facet_grid(~wt)

base_size <- 9
p + theme_grey(base_size = base_size) +
  labs(x = "", y= "") +
  scale_x_discrete(expand=c(0.1,0.1)) +
  theme(legend.position = "none", axis.ticks = element_blank(), panel.background = element_blank())
```

```
daf %>% group_by(substrate, target.mol) %>%
  summarise(T = sum(T), C = sum(C)) %>%
  ungroup %>% gather(key, count, C:T) %>%
  ggplot(data=., aes(x = substrate, fill=target.mol)) +
  geom_bar(stat="identity", aes(y=count)) +
  coord_polar() +
  scale_y_continuous(expand=c(0.2, 0)) +
  facet_grid(~key)


ggplot(daf.mol, aes(x = substrate)) +
  geom_bar(stat="identity", aes(y=T, fill = C.T)) +
  coord_polar() + facet_grid(wt~target.mol) +
  scale_fill_gradient(low = "red", high = "green", na.value="white") +
  scale_y_continuous(expand=c(0.2, 0))

ggplot(daf.atom, aes(x = substrate)) +
  geom_bar(stat="identity", aes(y=T, fill = C.T)) +
  coord_flip() + facet_wrap(~target.atom, nrow = 2) +
  scale_fill_gradient(low = "red", high = "green", na.value="white") +
  scale_y_continuous(expand=c(0.2, 0))
```