Modelling of the Leverhulme grant on NDL

Eva Viviani

February 2020

Contents

Distributions of interest 1

Exponential distribution 2

Zipfian and power law distributions 2

1) Strand 1 - experiment 2 2

Simulation of the Suffix condition - learning over time of the category 1 (“ma”): 4

Simulation of the Prefix condition - learning over time of the category 1 (“ma”) 6

2) Strand 1 - Experiment 3 7

Simulation of the Suffix condition - learning over time of the category 1 (“ma”): 9

Simulation of the Prefix condition - learning over time of the category 1 (“ma”): 10

3) Modeling with beta estimated from a real exponential distribution 12

This Notebook recreates the simulations reported in the grant (strand 1 experiment 2 and 3). Simulations were originally made by *Masa Vujovic*.

Set your local working directory. This should be (and is assumed to be in the rest of the code) the highest point in your local folder:

```
localGitDir <- 'C:/Users/eva_v/OneDrive/Documenti/UCL/For_sharing_EVA/Modelling/'
setwd(localGitDir);
```

Distributions of interest

In order to model our learning, we need to approximate the cues’ frequency distribution. In this notebook I consider three main distributions: a flat distribution–where all cues are equally distributed, an exponential distribution–where cues follow a costant average rate, and a zipfian distribution–that is a variant of a power law distribution where gamma is -1. I provide codes and formulas of all the three, but I focus on the exponential distribution since it’s the one considered in the grant.

- Importantly, Masa’s thesis was done on a flat distribution.

Exponential distribution

- The analytical formula is: f(k) = a*exp(-bk), where 'k' is the ranking position (literally how many steps we want in the distribution),'a' is the frequency of occurrence of the most frequent item and 'b' is the exponential decay, that is, the costant average rate at which the distribution unfolds.
- Hypothetically, one needs an hypothesis to define the value of b. By trials and errors, I found that Masa used a value of .69 for the grant, therefore I will stick to it.
- UPDATE: Michael rightly pointed out that a better solution is to use a real distribution and to derive the beta. This has been done in the point 3.
- When b is 0 every item is equiprobable, therefore in order to obtain a flat distribution simply assign b = 0.

```
exponentialFun <- function(k, a, b) {return(round(a*exp(-b*k)))}
```

Zipfian and power law distributions

- The analytical formula for a zipfian distribution is f(k) = c/k, where 'k' is the ranking position and 'c' has the same function of 'a' in the exponential distribution–frequency of occurrence of the most frequent item.
- Note that the zipfian distribution is a type of power law distribution where gamma is -1. Therefore the analytical formula in full of a power law is f(k) = c*k^gamma.

```
zipfianFun <- function(k, c) { return(round(c/k))}
powerFun <- function(k, c, g) { return(c*(k^g))}
```

1) Strand 1 - experiment 2

Masa created a dataframe containing 8 set of stimuli for the affix 'ge' and 'ma' with the following frequencies of occurrence.

```
myexp = read.csv(paste(localGitDir,"inputs/exponential_s1_1.csv", sep = ''));
summary(myexp);
```

```
##          Cues    Outcomes   Frequency
##  blue_d1_i1: 1   ge:8     Min.   :  17
##  blue_d1_i2: 1   ma:8     1st Qu.:  57
##  blue_d1_i3: 1            Median : 193
##  blue_d1_i4: 1            Mean   : 511
##  blue_d1_i5: 1            3rd Qu.: 641
##  blue_d1_i6: 1            Max.   :2049
##  (Other)   :10
```

Crucially, in this experiment we have type by token manipulation. LF-type items are presented high-frequently (2049 and 1025) compared to HF-type of items (17, 33, 65, 129, 257, 513). The distribution follows a costant exponential decay of .693. I am going to recreate the same distribution with our formula.

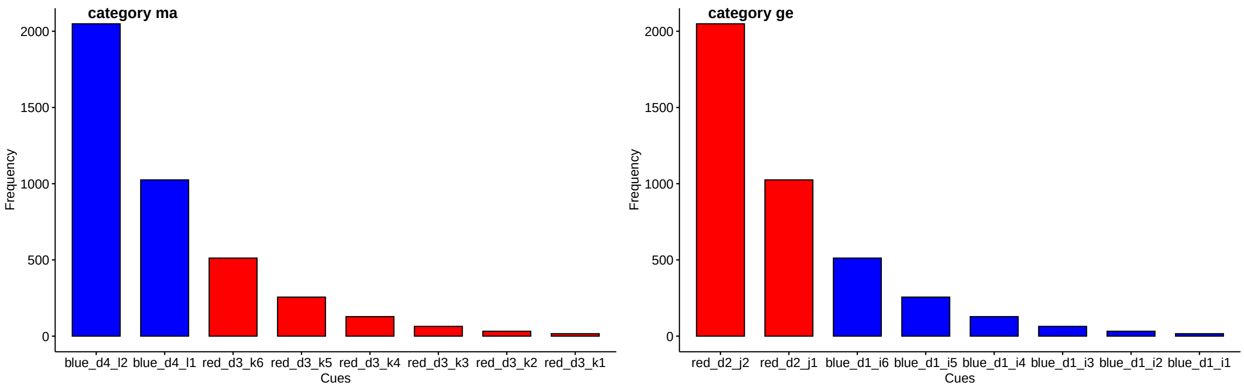
```r
k <- c(0:7)
a = 2049
b = .693

expDistribution <- NULL
for (i in 1:length(k)){
  expDistribution[i]<- exponentialFun(k[i], a, b)
}
expDistribution;
```

```
## [1] 2049 1025  512  256  128   64   32   16
```

```r
ma<-data.frame(Cues = myexp$Cues[9:16],
               Frequency = sort(expDistribution, decreasing =  F),
               Outcomes = 'ma')

ge<-data.frame(Cues = myexp$Cues[1:8],
               Frequency = sort(expDistribution, decreasing =  F),
               Outcomes = 'ge')

rbind(ma, ge)-> strand1expe2; rm(ge, ma, k, a, b, expDistribution); summary(strand1expe2);
```

```
##          Cues      Frequency      Outcomes
##  blue_d1_i1: 1   Min.   :  16.0   ma:8
##  blue_d1_i2: 1   1st Qu.:  56.0   ge:8
##  blue_d1_i3: 1   Median : 192.0
##  blue_d1_i4: 1   Mean   : 510.2
##  blue_d1_i5: 1   3rd Qu.: 640.2
##  blue_d1_i6: 1   Max.   :2049.0
##  (Other)   :10
```

```r
area.color <- c("blue", "blue", "red", "red", "red", "red", "red", "red");
mp<-ggbarplot(strand1expe2[strand1expe2$Outcomes=='ma',],
          x = "Cues", y = "Frequency",
          fill = area.color,
          order = c("blue_d4_l2", "blue_d4_l1",
                    "red_d3_k6", "red_d3_k5",
                    "red_d3_k4", "red_d3_k3",
                    "red_d3_k2", "red_d3_k1"));

area.color <- c("red", "red", "blue", "blue", "blue", "blue", "blue", "blue");
gp<-ggbarplot(strand1expe2[strand1expe2$Outcomes=='ge',],
          x = "Cues", y = "Frequency",
          fill = area.color,
          order = c("red_d2_j2", "red_d2_j1",
                    "blue_d1_i6", "blue_d1_i5",
                    "blue_d1_i4", "blue_d1_i3",
                    "blue_d1_i2", "blue_d1_i1"));

ggarrange(mp, gp,
          labels = c("category ma", "category ge"),
          ncol=2, nrow=1,
          hjust = -1,
          legend="right");
```



```
rm(mp,gp);
```

We have different distributions of cues within the stimuli that make them more or less predictive of the affix outcome. Specifically, for Category 1 (“ma”):

- 6 items are red and have discriminating feature 3 (d3) [75%]
- 2 items are blue and have discriminating feature 4 (d4) [25%]

While for Category 2 (“ge”):

- 6 items are blue and have discriminating feature 1 (d1) [75%]
- 2 items are red and have discriminating feature 2 (d2) [25%]

So within the same category (“ma” versus “ge”) we can group other two subcategories:

category “ma”:

- items blue + d4
- items red + d3

category “ge”:

- items blue + d1
- items red + d2

Certainly, in order to discriminate the objects a learner needs to discard the ambigous information, that is, the one that is not 100% associated to the right outcome. We can see that color indeed is not predictive of the categories ma and ge. We don’t have a strong prediction instead on the role of type by token frequency manipulation.

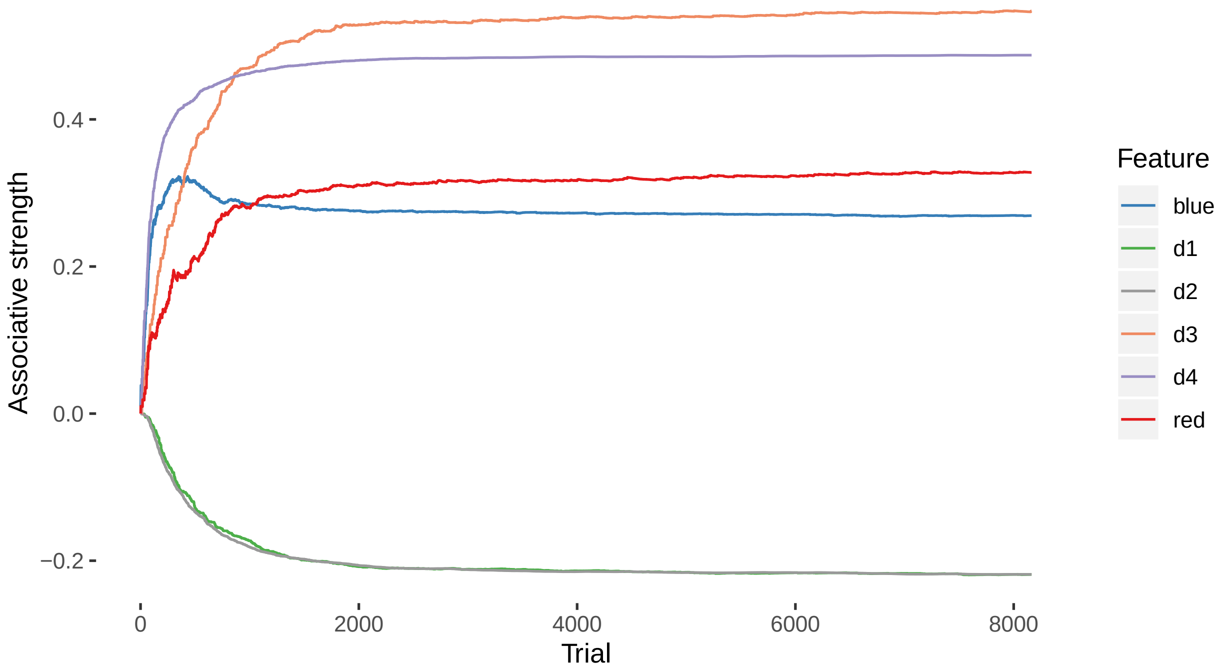
Simulation of the Suffix condition - learning over time of the category 1 (“ma”):

Now we can simulate how each cues is predictive of the category affix ‘ma’ in the case of the suffix. In other words, we want to see whether the model is able to predict the right affix/outcome, when provided the cues before the labels. Therefore, object labels will serve as cues and affixes will serve as outcomes:

```
traceOutcomes=c("ma");
traceCues=c("red", "blue", "d1", "d2", "d3", "d4");

mystims_suffix <- prepare_data(strand1expe2, condition = "suffix");

sal <- rep(0.1, length(traceCues));
exp_grant_suffix <- plot_line(mystims_suffix, saliency = sal, condition = "suffix");
```

```
## [1] "suffix condition"
## [1] 0.3277899
## [1] 0.2690579
## [1] -0.2186564
## [1] -0.2186336
## [1] 0.5472669
## [1] 0.4873436
```

```
exp_grant_suffix;
```



In the suffix condition, that is, when the cues are presented before the affix, the model is able to discriminate correctly the category "ma" for the two distinctive features, d4 and d3, while ignoring the colors which are not as predictive as the features.

Note that:

- d3 has higher associative strength than d4 and that is expected due to the difference in frequency of occurrence (75% vs 25%).
- d1 and d2 are negatively weighted, that is also expected because representative of the other category ("ge"), this means that the model has learned to negatively associate these two features to the category "ma".

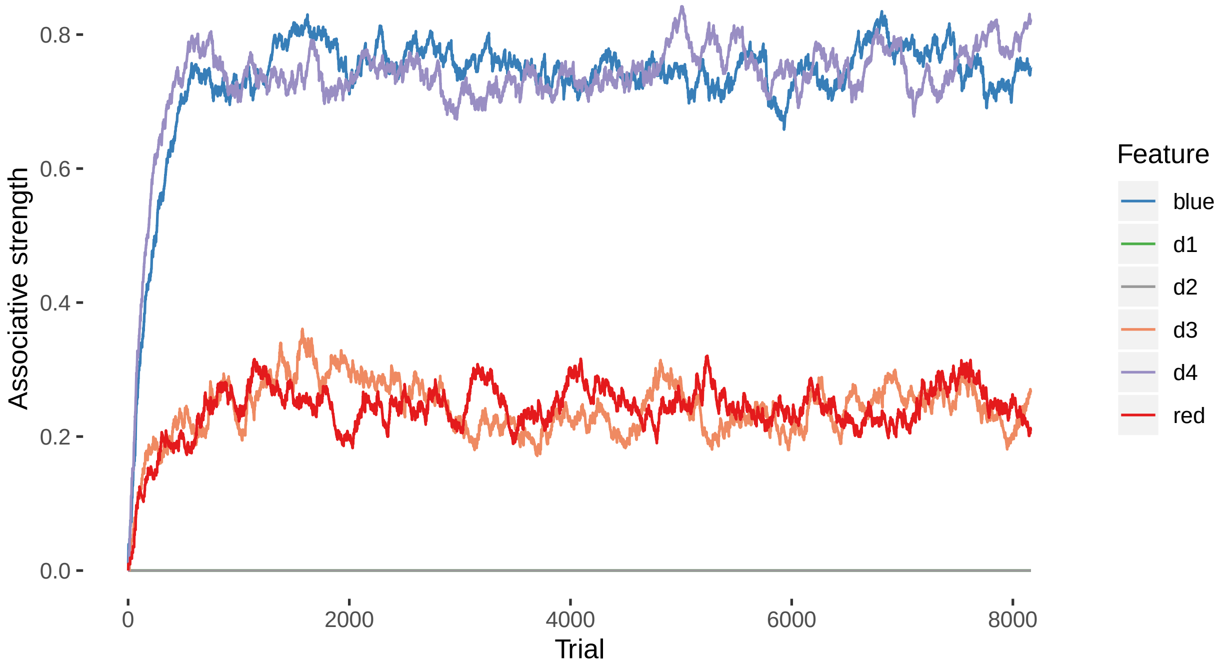
Simulation of the Prefix condition - learning over time of the category 1 ("ma")

In the prefix condition, the affix/outcome/label comes before its cues. We will therefore consider object labels as outcomes and affixes as cues.

```
traceCues=c("ma");
traceOutcomes=c("red", "blue", "d1", "d2", "d3", "d4");

mystims_prefix <- prepare_data(strand1expe2, condition = "prefix");

sal <- rep(0.1, length(traceOutcomes));
exp_grant_prefix <- plot_line(mystims_prefix, saliency = sal, condition = "prefix") ;
```

```
## [1] "prefix condition"
## [1] 0.2085332
## [1] 0.7518955
## [1] 0
## [1] 0
## [1] 0.2653066
## [1] 0.8231974
```

```
exp_grant_prefix;
```



In the prefix condition, that is, when the affix/outcome "ma" is presented before its discriminative cues, the type of learning is completely different. The model learns single instances (blue + d4, subcategory 1, and red + d3, subcategory 2).

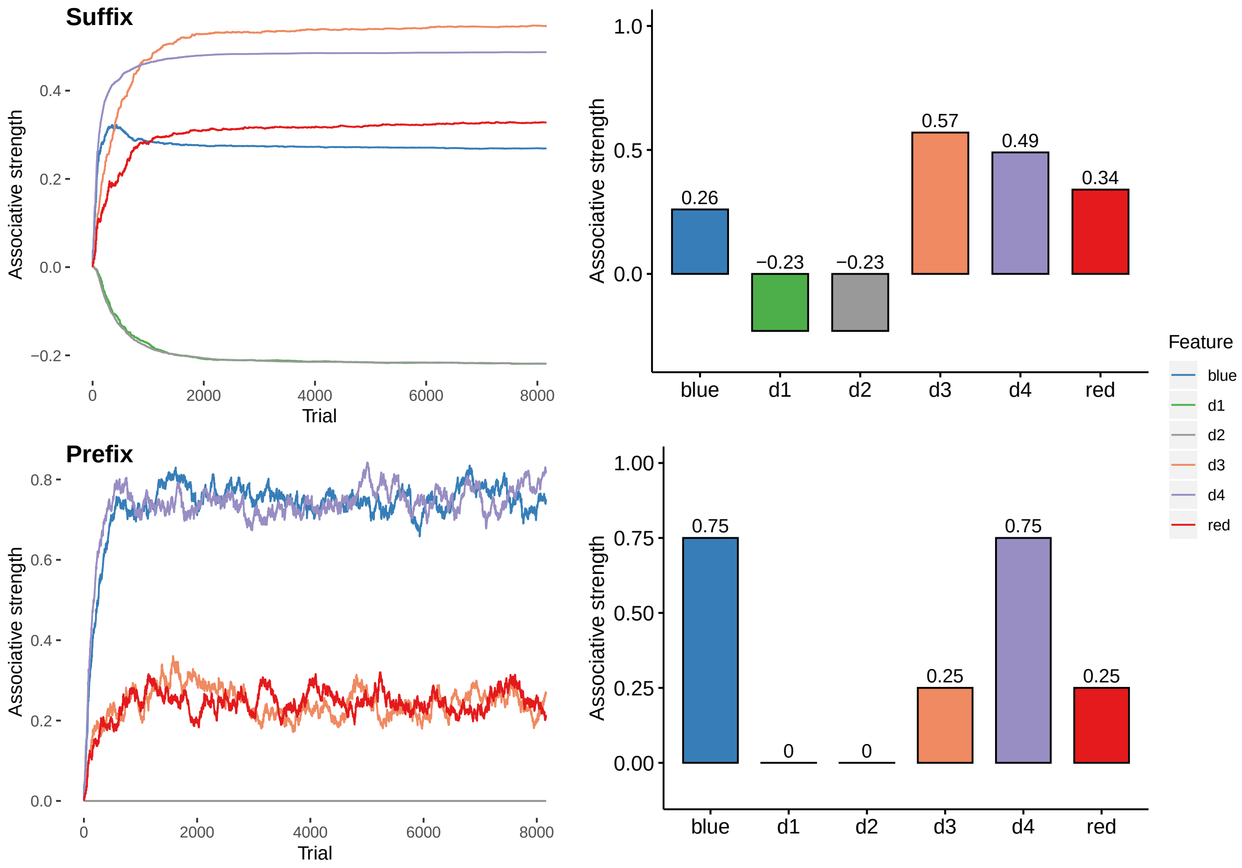
However, note that:

- blue + d4 are presented only 25% of the trials, instead here they show 75% of associative weight - that is because LF-type are presented very frequently compared to HF-type

- the model never learns that there are cues predictive of the other category “ge” (d1 and d2) that is evident because the associative strength never becomes negative for these two cues (they show a stationary value of 0). That is possible because in this learning there are never instances in which the label “ma” is wrongly associated with one of these cues, this is due to the type of learning: Seeing always first the label doesn’t allow the model to evaluate all the cues and weight each one of the based on the outcome.

Let’s compare the two different type of learnings:

```
ggarrange(exp_grant_suffix, exp_grant_suffixbar,
          exp_grant_prefix, exp_grant_prefixbar,
          labels = c("Suffix", " ", "Prefix", " "), ncol=2, nrow=2,
          legend="right",
          hjust = -1,
          common.legend = TRUE);
```



2) Strand 1 - Experiment 3

In this third experiment the HF-type of LF-type of manipulation is reversed. That is, the two HF items now have the highest frequency (2049, 1025, 513, 257, 129 and 65 trials), and the LF items are less frequent (33 and 17 trials).

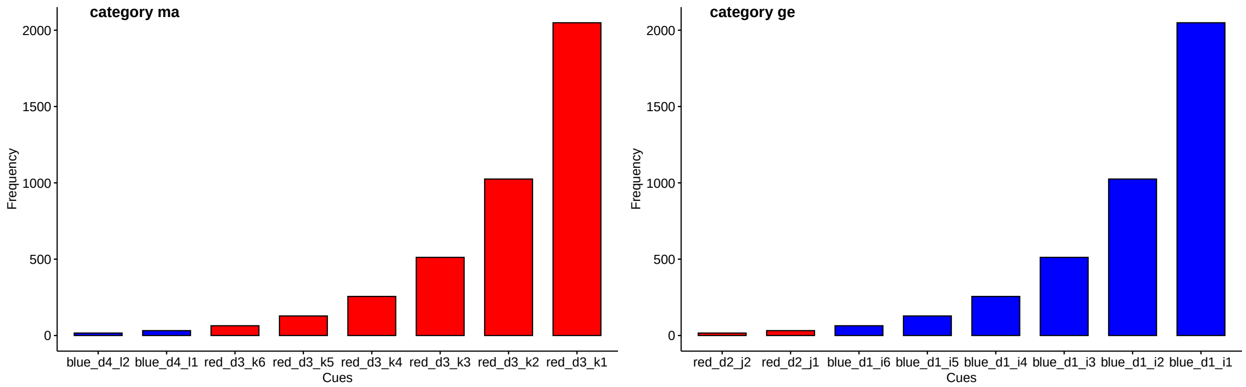
```
k <- c(0:7)
a = 2049
b = .693

expDistribution <- NULL
for (i in 1:length(k)){
  expDistribution[i]<- exponentialFun(k[i], a, b)
}
expDistribution;
```

```
## [1] 2049 1025  512  256  128   64   32   16
```

```
ma<-data.frame(Cues = myexp$Cues[9:16],
               Frequency = sort(expDistribution, decreasing =  T),
               Outcomes = 'ma')

ge<-data.frame(Cues = myexp$Cues[1:8],
               Frequency = sort(expDistribution, decreasing =  T),
               Outcomes = 'ge')

rbind(ma, ge)-> strand1expe3; rm(ge, ma, k, a, b, expDistribution); summary(strand1expe3);
```

```
##          Cues      Frequency      Outcomes
##  blue_d1_i1: 1   Min.   :  16.0   ma:8
##  blue_d1_i2: 1   1st Qu.:  56.0   ge:8
##  blue_d1_i3: 1   Median : 192.0
##  blue_d1_i4: 1   Mean   : 510.2
##  blue_d1_i5: 1   3rd Qu.: 640.2
##  blue_d1_i6: 1   Max.   :2049.0
##  (Other)   :10
```

```
area.color <- c("blue", "blue", "red", "red", "red", "red", "red", "red");
mp<-ggbarplot(strand1expe3[strand1expe3$Outcomes=='ma',],
          x = "Cues", y = "Frequency",
          fill = area.color,
          order = c("blue_d4_l2", "blue_d4_l1",
                    "red_d3_k6", "red_d3_k5",
                    "red_d3_k4", "red_d3_k3",
                    "red_d3_k2", "red_d3_k1"));

area.color <- c("red", "red", "blue", "blue", "blue", "blue", "blue", "blue");
gp<-ggbarplot(strand1expe3[strand1expe3$Outcomes=='ge',],
          x = "Cues", y = "Frequency",
          fill = area.color,
          order = c("red_d2_j2", "red_d2_j1",
                    "blue_d1_i6", "blue_d1_i5",
                    "blue_d1_i4", "blue_d1_i3",
                    "blue_d1_i2", "blue_d1_i1"));

ggarrange(mp, gp,
          labels = c("category ma", "category ge"),
          ncol=2, nrow=1,
          hjust = -1,
          legend="right");
```



```
rm(mp,gp);
```

We have again different distributions of cues within the stimuli that make them more or less predictive of the affix outcome. Specifically, for Category 1 ("ma"):

- 6 items are red and have discriminating feature 3 (d3) [75%]
- 2 items are blue and have discriminating feature 4 (d4) [25%]

While for category 2 ("ge"):

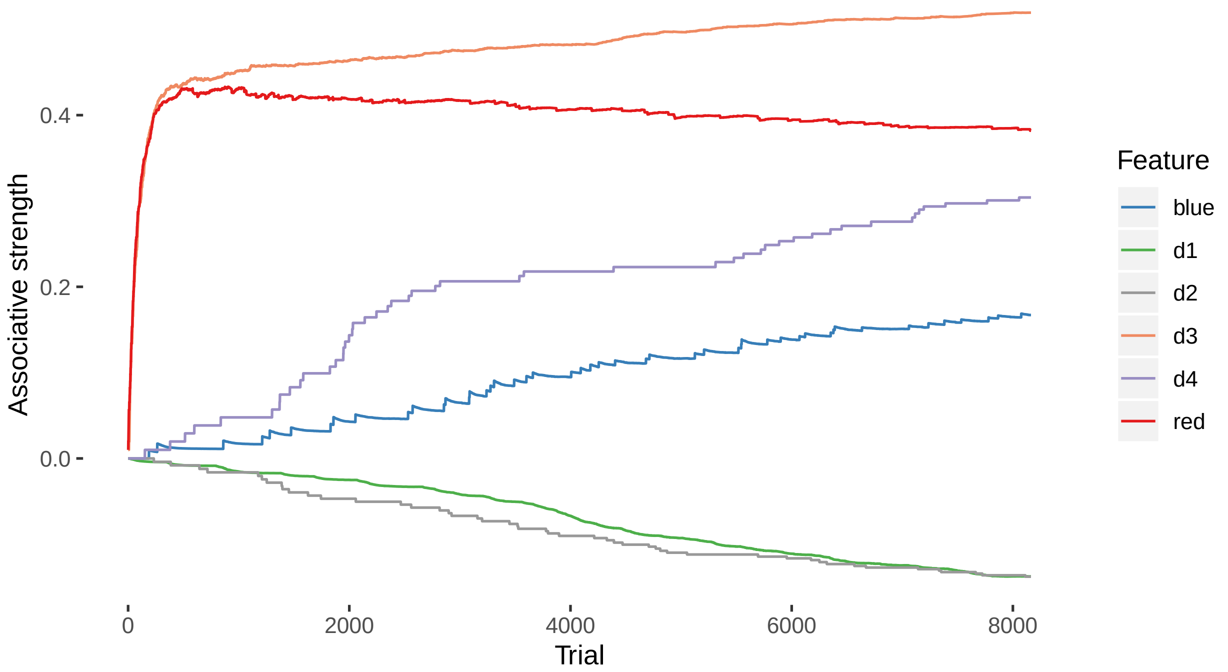
- 6 items are blue and have discriminating feature 1 (d1) [75%]
- 2 items are red and have discriminating feature 2 (d2) [25%]

Simulation of the Suffix condition - learning over time of the category 1 ("ma"):

```
traceOutcomes=c("ma");
traceCues=c("red", "blue", "d1", "d2", "d3", "d4");

mystims_suffix <- prepare_data(strand1expe3, condition = "suffix");

sal <- rep(0.1, length(traceCues));
exp_grant_suffix <- plot_line(mystims_suffix, saliency = sal, condition = "suffix");
```

```
## [1] "suffix condition"
## [1] 0.3819497
## [1] 0.1671258
## [1] -0.1374599
## [1] -0.1377419
## [1] 0.5194438
## [1] 0.3040645
```

```
exp_grant_suffix;
```



In the suffix condition, that is, when the cues preceed the label/affix, when the LF-type of cues is displayed very few times, the models is able again to predict that d3 (high frequent predictive cue) is correct, however is unable to weight down the color red, that is not as predictive as the other critical feature, d4. That is because d4 is displayed very few times.

Note that:

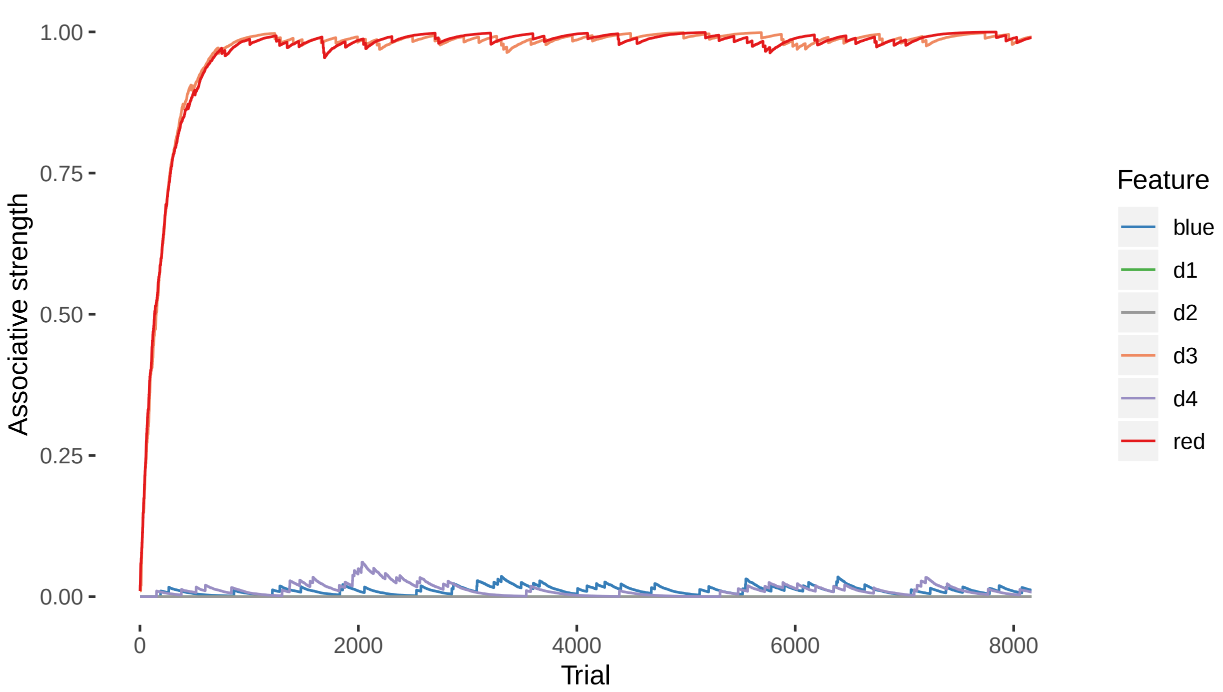
- the model is able anyway to weight negatively the features that are predictive of the other affix.

Simulation of the Prefix condition - learning over time of the category 1 (“ma”):

```
traceCues=c("ma");
traceOutcomes=c("red", "blue", "d1", "d2", "d3", "d4");

mystims_prefix <- prepare_data(strand1expe3, condition = "prefix");

sal <- rep(0.1, length(traceOutcomes));
exp_grant_prefix <- plot_line(mystims_prefix, saliency = sal, condition = "prefix") ;
```

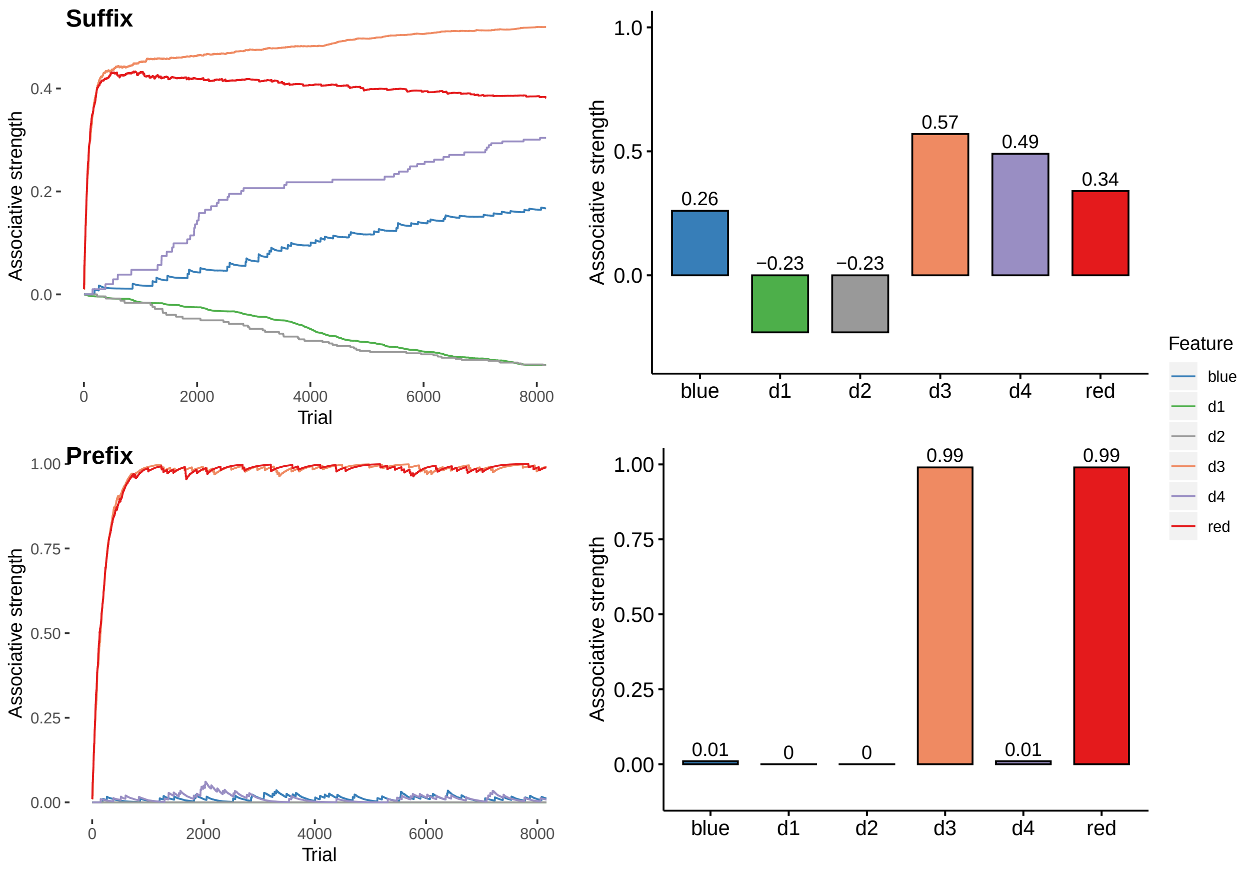
```
## [1] "prefix condition"
## [1] 0.9897707
## [1] 0.01120786
## [1] 0
## [1] 0
## [1] 0.9910913
## [1] 0.007866687
```

```
exp_grant_prefix;
```



In the prefix type of learning, that is, label comes before cues, the model is able to learn only one set of subgroup, while the other apparently is never learned because is presented very few times.

```
ggarrange(exp_grant_suffix, exp_grant_suffixbar,
          exp_grant_prefix, exp_grant_prefixbar,
          labels = c("Suffix", " ",
                     "Prefix", " "), ncol=2, nrow=2,
          legend="right",
          hjust = -1,
          common.legend = TRUE);
```



3) Modeling with beta estimated from a real exponential distribution

Load an existing dataset - I have downloaded the COCA English dataset used by Michael *(https://www.wordfrequency.info/free.asp?s=y)* and I have focused on kinship terms.

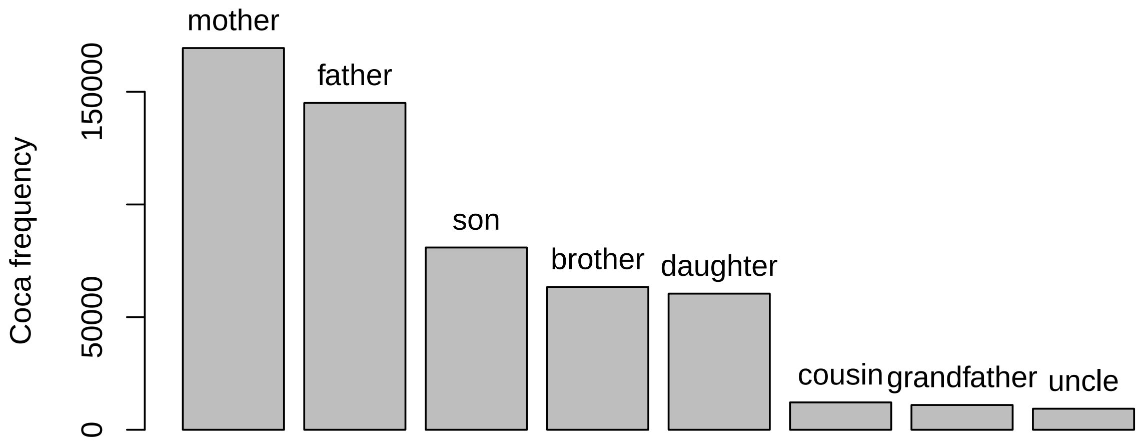
```
coca<-read.csv("C:/Users/eva_v/OneDrive/Documenti/UCL/For_sharing_EVA/Modelling/cocaKinshipFrequency.cs
coca$Word <- gsub("[[:space:]]", "", coca$Word); #remove white spaces
```

I select only 8 of them in order to have the same k as proposed in the grant:

```
kinship <- c('mother', 'father', 'son',
             'brother', 'daughter', 'cousin',
             'grandfather', 'uncle');

target<- subset(coca, coca$Word %in% kinship);
target <- droplevels(target);
```

Let's plot their raw frequency distribution:

```
p<-barplot(sort(target$Frequency, decreasing = TRUE),
           ylab = 'Coca frequency',
           xlba = 'kinship terms')
text(p, sort(target$Frequency, decreasing = TRUE),
     labels=kinship,
     xpd=NA,
     pos = 3) #
```



```
rm(p)
```

Let’s find our exponential decay for this distribution. I’m following Michael advice and using a fit.

```
k <- c(0:7) #our steps
exponential.model <- lm(log(target$Frequency)~ k);
```

Then I exctract alpha and beta from the model:

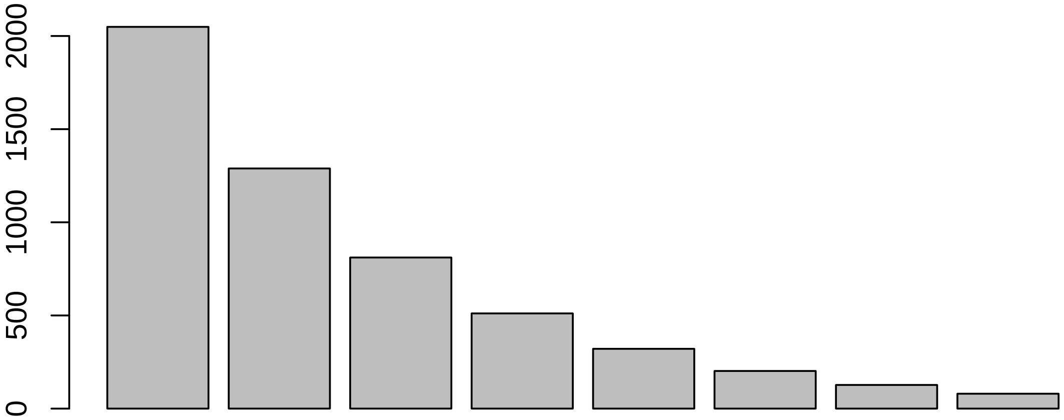
```
alpha <- exp(coef(exponential.model)[1]);
beta <- coef(exponential.model)[2];
```

Plot of the distribution with the beta obtained.

```
k <- c(0:7);
a = 2049;
b = -beta; #beta is negative, just switching sign for simplicity

expDistribution <- NULL
for (i in 1:length(k)){
  expDistribution[i]<- exponentialFun(k[i], a, b)
}
expDistribution;
```

```
## [1] 2049 1289  811  511  321  202  127   80
```

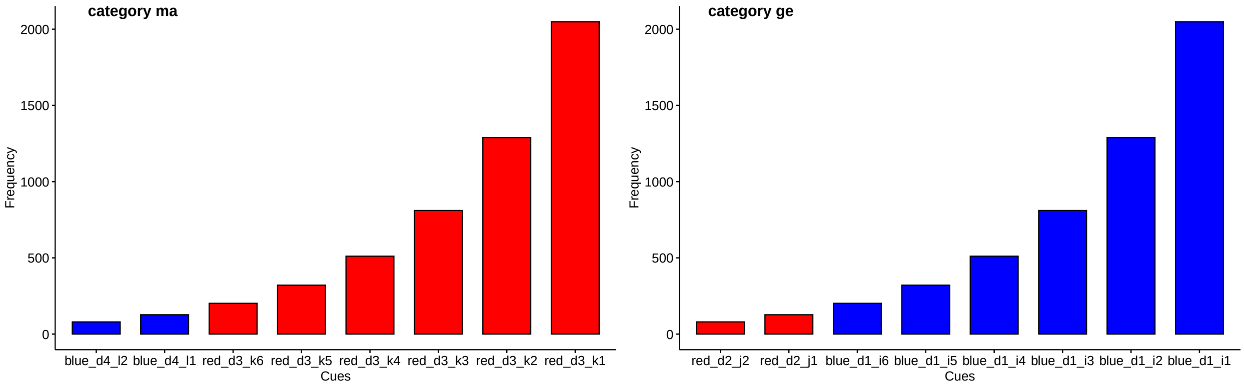
```
barplot(sort(expDistribution, decreasing = TRUE));
```



```
ma<-data.frame(Cues = myexp$Cues[9:16],
               Frequency = sort(expDistribution, decreasing =  T),
               Outcomes = 'ma')

ge<-data.frame(Cues = myexp$Cues[1:8],
               Frequency = sort(expDistribution, decreasing =  T),
               Outcomes = 'ge')

rbind(ma, ge)-> strand1expereal; rm(ge, ma, k, a, b, expDistribution); summary(strand1expereal);
```

```
##          Cues      Frequency      Outcomes
##  blue_d1_i1: 1   Min.   :  80.0   ma:8
##  blue_d1_i2: 1   1st Qu.: 183.2   ge:8
##  blue_d1_i3: 1   Median : 416.0
##  blue_d1_i4: 1   Mean   : 673.8
##  blue_d1_i5: 1   3rd Qu.: 930.5
##  blue_d1_i6: 1   Max.   :2049.0
##  (Other)   :10
```

```
area.color <- c("blue", "blue", "red", "red", "red", "red", "red", "red");
mp<-ggbarplot(strand1expereal[strand1expereal$Outcomes=='ma',],
          x = "Cues", y = "Frequency",
          fill = area.color,
          order = c("blue_d4_l2", "blue_d4_l1",
                    "red_d3_k6", "red_d3_k5",
                    "red_d3_k4", "red_d3_k3",
                    "red_d3_k2", "red_d3_k1"));

area.color <- c("red", "red", "blue", "blue", "blue", "blue", "blue", "blue");
gp<-ggbarplot(strand1expereal[strand1expereal$Outcomes=='ge',],
          x = "Cues", y = "Frequency",
          fill = area.color,
          order = c("red_d2_j2", "red_d2_j1",
                    "blue_d1_i6", "blue_d1_i5",
                    "blue_d1_i4", "blue_d1_i3",
                    "blue_d1_i2", "blue_d1_i1"));

ggarrange(mp, gp,
          labels = c("category ma", "category ge"),
          ncol=2, nrow=1,
          hjust = -1,
          legend="right");
```



```
rm(mp,gp);
```

```
traceOutcomes=c("ma");
traceCues=c("red", "blue", "d1", "d2", "d3", "d4");

mystims_suffix <- prepare_data(strand1expereal, condition = "suffix");

sal <- rep(0.1, length(traceCues));
exp_grant_suffix <- plot_line(mystims_suffix, saliency = sal, condition = "suffix");
```

```
## [1] "suffix condition"
## [1] 0.3416632
## [1] 0.2594778
## [1] -0.221186
## [1] -0.2205222
## [1] 0.5626474
## [1] 0.4795123
```

```
traceCues=c("ma");
traceOutcomes=c("red", "blue", "d1", "d2", "d3", "d4");
```

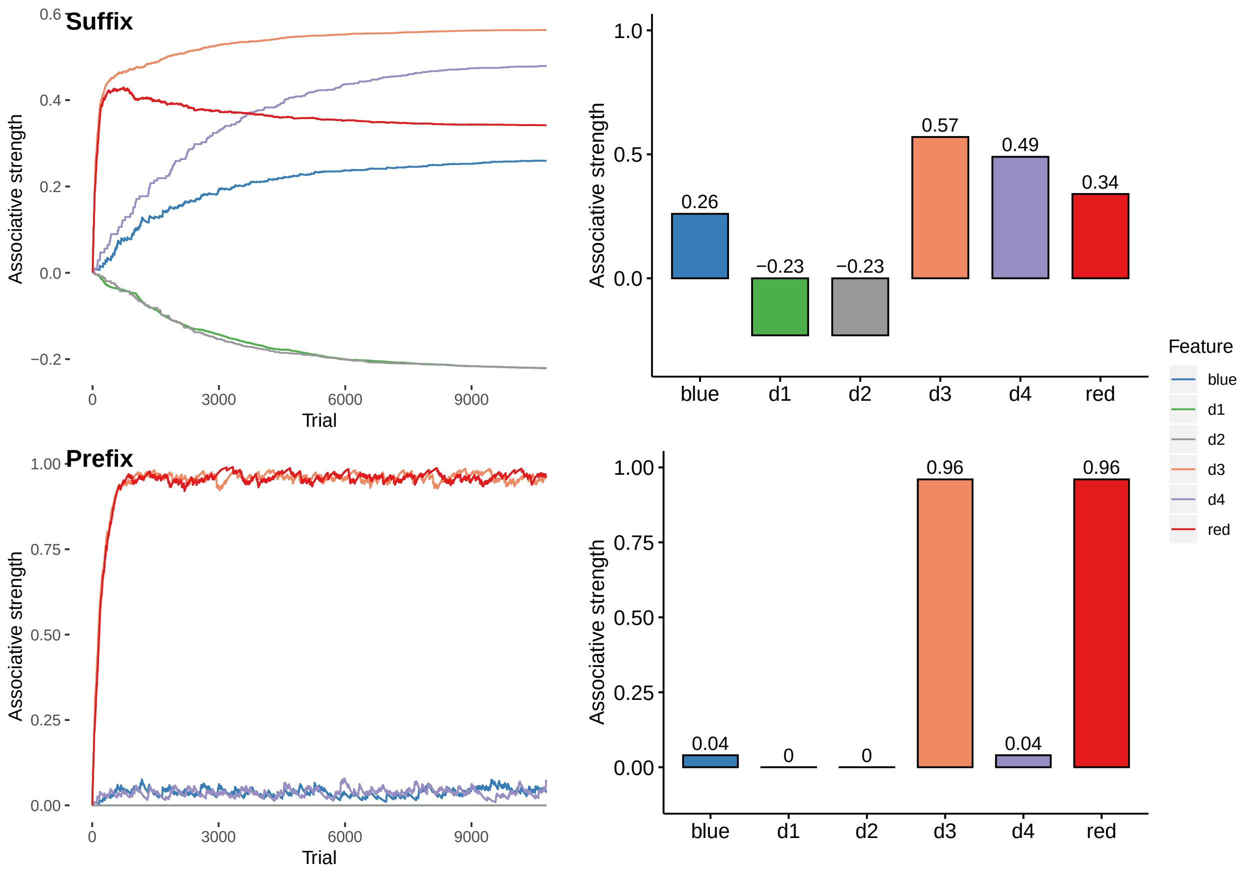
```
mystims_prefix <- prepare_data(strand1expereal, condition = "prefix");

sal <- rep(0.1, length(traceOutcomes));
exp_grant_prefix <- plot_line(mystims_prefix, saliency = sal, condition = "prefix") ;
```

```
## [1] "prefix condition"
## [1] 0.9647335
## [1] 0.03833567
## [1] 0
## [1] 0
## [1] 0.9668059
## [1] 0.0692797
```

```
exp_grant_prefixbar <- plot.bar(mystims_prefix, saliency = sal, condition = "prefix") ;
```

```
## [1] "prefix condition"
## [1] 0.9777695
## [1] 0.03610743
## [1] 0
## [1] 0
## [1] 0.9631007
## [1] 0.07107223
```

```
ggarrange(exp_grant_suffix, exp_grant_suffixbar,
          exp_grant_prefix, exp_grant_prefixbar,
          labels = c("Suffix", " ",
                     "Prefix", " "), ncol=2, nrow=2,
          legend="right",
          hjust = -1,
          common.legend = TRUE);
```


Suffix
Prefix
Associative strength
Trial
0.6
0.4
0.2
0.0
−0.2
1.00
0.75
0.50
0.25
0.00
0
3000
6000
9000
1.0
0.5
0.0
0.26
−0.23
−0.23
0.57
0.49
0.34
0.04
0
0
0.96
0.04
0.96
blue
d1
d2
d3
d4
red
Feature
blue
d1
d2
d3
d4
red