CaverDock - exploratory analysis

The preregistration

The preregistraion of this evaluation follows, some comments regarding more detailed explanation and argumentation for choices added:

1) Have any data been collected for this study already?

It's complicated. We have already collected some data but explain in Question 8 why readers may consider this a valid pre-registration nevertheless.

2) What's the main question being asked or hypothesis being tested in this study?

We hypothesize that a new search heuristic in CaverDock will decrease the maximal energy peak of the trajectories of ligands passing through the tunnel. This means that the average of the maximal energy peak over experiments will be lower in experiments done with the new search heuristic compared to experiments done with the old search heuristic. We expect to see this improvement in datasets with more complex tunnels (1BN7, 1ZNJ). More open and easier to pass tunnels (ACHE, SPL1) might not benefit from this improvement, or they do, but the effect of the improvement might not be large enough to observe with our sample size.

3) Describe the key dependent variable(s) specifying how they will be measured.

Each experiment has the maximal peak of its upper-bound trajectory's energy profile. We calculate its mean over experiments with the same dataset and heuristic.

4) How many and which conditions will participants be assigned to?

Experiments will be conducted with two versions of CaverDock. CaverDock 1.0.1 contains the old search heuristic, CaverDock 1.1 the new search heuristic. We have four datasets with proteins containing tunnels of various complexity: 1ZNJ, 1BN7, ACHE, SPL1. These datasets are independent. ACHE and SPL1 are "easier": their tunnels are more open, they do not include any major energy barrier than the ligand has to pass. 1BN7 and 1ZNJ are more complex: their tunnels are more difficult to pass through due to spatial or energy restrictions.

5) Specify exactly which analyses you will conduct to examine the main question/hypothesis.

For each dataset, Welch's t-test between experiments with the old and the new heuristic.

6) Describe exactly how outliers will be defined and handled, and your precise rule(s) for excluding observations.

We will exclude the experiments that failed to result in an upper-bound trajectory or were running five times longer that the average walltime for the given dataset and CaverDock version.

7) How many observations will be collected or what will determine sample size?

No need to justify decision, but be precise about exactly how the number will be determined. For each dataset and each version of CaverDock, 140 experiments. This sample size is determined using power analysis, where we estimate the effect size 0.3 based on our previous experiments.

#Load pwr package to easily calculate the statistical power

```
if(!require(pwr)){install.packages('pwr')}
```

```
## Loading required package: pwr
```

```
library(pwr)
#Disable scientific notation (1.05e10)
options(scipen=999)
```

We wanted a recommended statistical power of 80%, Type 1 error rate of 5% and we assume one-way directional effect (small to medium). #Calculate power formally by power analysis

```
samplesize<-pwr.t.test(d=-0.3, power=0.8, sig.level=0.05,type="two.sample",alternative="less")$n
samplesize
```

```
## [1] 138.0716
```

This resulted in recommended sample size of 138. We rounded it up to 140 and ran 140 experiments with the old heuristic (single bt+fw) and 140 experiments with a new heuristic (multiple bt+fw) for each datatset.

8) Anything else you would like to pre-register?

(e.g., secondary analyses, variables collected for exploratory purposes, unusual analyses planned?) We have two other, minor families of hypotheses we would like to test.

H2 The new search heuristic creates more stable trajectories, i.e. the standard deviation of maximal peaks of upper-bound trajectory over the experiments will be lower in experiments done with the new heuristic than in the experiments done with the old heuristic. We will analyze this for each dataset with Levene's test between experiments with the old and the new heuristic.

H3 The new search heuristic creates smoother trajectories, i.e. the mean of cumulative barriers of the trajectories (the sum of increases in the energy profile) will be lower in experiments done with the new search heuristic than in experiments done with the old search heuristic. We will analyze this for each dataset with Welch's t-test between experiments with the old and the new heuristic.

We are controlling Type 1 errors for all three families of hypotheses with FDR and Benjamini–Hochberg procedure. If the old and the new heuristic differ, the improvement might express in three ways, hence three independent families of hypotheses. For each family of hypotheses, we will have four p-values (one test per dataset). We set controlled Type 1 error rate in FDR to 1/4 for all families of hypotheses.

Experiments for these analyses have already run during the summer when the computational resources were available. We have not accessed these data to this date.

Evaluation

We have experiments done for four datasets and for two versions of CaverDock on gcn.fi.muni.cz:~/CD-exploratory-analysis

```
if(!require(pwr)){install.packages(c('astsa','lawstat','pracma'))}
library(astsa)
library(lawstat)
library(pracma)
```

```
##
## Attaching package: 'pracma'

## The following object is masked from 'package:lawstat':
##
##     cd
```

```
mysummary <- function(v) {
  ms <- matrix(nrow=1, ncol=3, dimnames = list(c(), c("mean","sd", "median")))
  ms[1,"mean"] <- mean(v)
  ms[1,"sd"] <- sd(v)
  ms[1,"median"] <- median(v)
  ms
}
```

Get the energy profiles

We ran energyprofile.py for each experiment, generating energy.dat.

```
for j in {1..140}; do cd $j; cd-energyprofile -d tun-ex.dsd -t run-ub.pdbqt -s 0 > energy.dat;
cd ..; done;
```

Handle outliers and exclude

All experiments ended successfully within the limited walltime.

H1 lower mean of maximal peaks

We found the maximal value in each energy.dat and put it in max-ub.dat

```
for i in {1..140}; do cd $i; awk 'BEGIN{getline;max = -100000;}
{if (max < $3){max=$3}}END{print max}' energy.dat;  cd ..; done > max-ub.dat
```

1BN7

```
old_table <- read.table('./CD-1.0.1/1BN7/max-ub.dat')
new_table <- read.table('./CD-1.1/1BN7/max-ub.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##            mean         sd median
## [1,] -0.6435714 0.06141112   -0.6
```

```
mysummary(y)
```

```
##            mean         sd median
## [1,] -0.6128571 0.07571516   -0.6
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] 0.4455533
```

```
z<-t.test(y, x, alternative="less") #perform the t-test
p_1bn7_max_ub<-z$p.value #get the p-value and store it
p_1bn7_max_ub
```

```
## [1] 0.999882
```

1ZNJ

```
old_table <- read.table('./CD-1.0.1/1ZNJ/max-ub.dat')
new_table <- read.table('./CD-1.1/1ZNJ/max-ub.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##             mean         sd median
## [1,] -0.2328571 0.07722047   -0.2
```

```
mysummary(y)
```

```
##             mean         sd median
## [1,] -0.2571429 0.08148165   -0.3
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] -0.3059438
```

```
z<-t.test(y, x, alternative="less") #perform the t-test
p_1znj_max_ub<-z$p.value #get the p-value and store it
p_1znj_max_ub
```

```
## [1] 0.00550337
```

ACHE

```
old_table <- read.table('./CD-1.0.1/ACHE/max-ub.dat')
new_table <- read.table('./CD-1.1/ACHE/max-ub.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##           mean        sd median
## [1,] -1.442143 0.2145995   -1.5
```

```
mysummary(y)
```

```
##           mean        sd median
## [1,] -1.507143 0.2016632   -1.6
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] -0.3121521
```

```
z<-t.test(y, x, alternative="less") #perform the t-test
p_ache_max_ub<-z$p.value #get the p-value and store it
p_ache_max_ub
```

```
## [1] 0.004750992
```

SPL1

```
old_table <- read.table('./CD-1.0.1/SPL1/max-ub.dat')
new_table <- read.table('./CD-1.1/SPL1/max-ub.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##           mean        sd median
## [1,] -3.048571 0.3765588   -3.1
```

```
mysummary(y)
```

```
##           mean        sd median
## [1,] -3.313571 0.1551099   -3.3
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] -0.9202288
```

```
z<-t.test(y, x, alternative="less") #perform the t-test
p_spl1_max_ub<-z$p.value #get the p-value and store it
p_spl1_max_ub
```

```
## [1] 0.0000000000004005263
```

False discovery error rate control

```
p <- c(p_1bn7_max_ub, p_1znj_max_ub, p_ache_max_ub, p_spl1_max_ub)
index_max_accepted_p_value <- FDR(p, qlevel = 0.25)
threshold_p_value <- p[index_max_accepted_p_value]
p <= threshold_p_value
```

```
## [1] FALSE  TRUE  TRUE  TRUE
```

```
#exploratory part, what if we have stricter FDR error rate?
index_max_accepted_p_value <- FDR(p, qlevel = 0.05)
threshold_p_value <- p[index_max_accepted_p_value]
p <= threshold_p_value
```

```
## [1] FALSE  TRUE  TRUE  TRUE
```

H2 lower standard deviation of maximal peak

1BN7

```
old_table <- read.table('./CD-1.0.1/1BN7/max-ub.dat')
new_table <- read.table('./CD-1.1/1BN7/max-ub.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##            mean         sd median
## [1,] -0.6435714 0.06141112   -0.6
```

```
mysummary(y)
```

```
##            mean         sd median
## [1,] -0.6128571 0.07571516   -0.6
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] 0.4455533
```

```
all <- c(x,y)
groups <- as.factor(c(rep(1, times=length(x)), rep(2, times=length(y))))
z<-levene.test(all, groups, location="mean", trim.alpha = 0) #perform the test
p_1bn7_std_max_ub<-z$p.value #get the p-value and store it
p_1bn7_std_max_ub
```

```
## [1] 0.2327259
```

1ZNJ

```
old_table <- read.table('./CD-1.0.1/1ZNJ/max-ub.dat')
new_table <- read.table('./CD-1.1/1ZNJ/max-ub.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##            mean         sd median
## [1,] -0.2328571 0.07722047   -0.2
```

```
mysummary(y)
```

```
##            mean         sd median
## [1,] -0.2571429 0.08148165   -0.3
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] -0.3059438
```

```
all <- c(x,y)
groups <- as.factor(c(rep(1, times=length(x)), rep(2, times=length(y))))
z<-levene.test(all, groups, location="mean", trim.alpha = 0) #perform the test
```

```
p_1znj_std_max_ub<-z$p.value #get the p-value and store it
p_1znj_std_max_ub
```

```
## [1] 0.9124113
```

ACHE

```
old_table <- read.table('./CD-1.0.1/ACHE/max-ub.dat')
new_table <- read.table('./CD-1.1/ACHE/max-ub.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##           mean        sd median
## [1,] -1.442143 0.2145995   -1.5
```

```
mysummary(y)
```

```
##           mean        sd median
## [1,] -1.507143 0.2016632   -1.6
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] -0.3121521
```

```
all <- c(x,y)
groups <- as.factor(c(rep(1, times=length(x)), rep(2, times=length(y))))
z<-levene.test(all, groups, location="mean", trim.alpha = 0) #perform the test
p_ache_std_max_ub<-z$p.value #get the p-value and store it
p_ache_std_max_ub
```

```
## [1] 0.2983247
```

SPL1

```
old_table <- read.table('./CD-1.0.1/SPL1/max-ub.dat')
new_table <- read.table('./CD-1.1/SPL1/max-ub.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##           mean        sd median
## [1,] -3.048571 0.3765588   -3.1
```

```
mysummary(y)
```

```
##           mean        sd median
## [1,] -3.313571 0.1551099   -3.3
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] -0.9202288
```

```
all <- c(x,y)
groups <- as.factor(c(rep(1, times=length(x)), rep(2, times=length(y))))
z<-levene.test(all, groups, location="mean", trim.alpha = 0) #perform the test
p_spl1_std_max_ub<-z$p.value #get the p-value and store it
p_spl1_std_max_ub
```

```
## [1] 0.000000000002027978
```

False discovery error rate control

```
library(astsa)
p <- c(p_1bn7_std_max_ub, p_1znj_std_max_ub, p_ache_std_max_ub, p_spl1_std_max_ub)
index_max_accepted_p_value <- FDR(p, qlevel = 0.25)
threshold_p_value <- p[index_max_accepted_p_value]
p <= threshold_p_value
```

```
## [1] FALSE FALSE FALSE  TRUE
```

```
#exploratory part, what if we have stricter FDR error rate?
index_max_accepted_p_value <- FDR(p, qlevel = 0.05)
threshold_p_value <- p[index_max_accepted_p_value]
p <= threshold_p_value
```

```
## [1] FALSE FALSE FALSE  TRUE
```

H3 lower cummulative barrier

We added up the cummulative energy barrier and put it in cum-eb.dat

```
for i in {1..140}; do cd $i; awk 'BEGIN{getline;sum=0;previous=$3;}
{if (0 < $3 - previous){sum = sum + ($3-previous)}; previous=$3}END{print sum}' energy.dat;
cd ..; done > cum-eb.dat
```

1BN7

```
old_table <- read.table('./CD-1.0.1/1BN7/cum-eb.dat')
new_table <- read.table('./CD-1.1/1BN7/cum-eb.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##          mean      sd median
## [1,] 3.490714 0.30153    3.4
```

```
mysummary(y)
```

```
##          mean        sd median
## [1,] 3.533571 0.2997867    3.5
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] 0.1425437
```

```
z<-t.test(y, x, alternative="less") #perform the t-test
p_1bn7_cum_eb<-z$p.value #get the p-value and store it
p_1bn7_cum_eb
```

```
## [1] 0.8829797
```

1ZNJ

```
old_table <- read.table('./CD-1.0.1/1ZNJ/cum-eb.dat')
new_table <- read.table('./CD-1.1/1ZNJ/cum-eb.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##          mean        sd median
## [1,] 4.961429 0.1638501      5
```

```
mysummary(y)
```

```
##          mean        sd median
## [1,] 5.003571 0.1436476      5
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] 0.2735122
```

```
z<-t.test(y, x, alternative="less") #perform the t-test
p_1znj_cum_eb<-z$p.value #get the p-value and store it
p_1znj_cum_eb
```

```
## [1] 0.9885602
```

ACHE

```
old_table <- read.table('./CD-1.0.1/ACHE/cum-eb.dat')
new_table <- read.table('./CD-1.1/ACHE/cum-eb.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##          mean       sd median
## [1,] 4.106429 0.492565    4.1
```

```
mysummary(y)
```

```
##      mean        sd median
## [1,] 4.02 0.4295221      4
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] -0.1870263
```

```
z<-t.test(y, x, alternative="less") #perform the t-test
p_ache_cum_eb<-z$p.value #get the p-value and store it
p_ache_cum_eb
```

```
## [1] 0.05939721
```

SPL1

```
old_table <- read.table('./CD-1.0.1/SPL1/cum-eb.dat')
new_table <- read.table('./CD-1.1/SPL1/cum-eb.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##          mean        sd median
## [1,] 2.853571 0.4846434    2.8
```

```
mysummary(y)
```

```
##          mean        sd median
## [1,] 3.058571 0.4605261    3.1
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] 0.4336435
```

```
z<-t.test(y, x, alternative="less") #perform the t-test
p_spl1_cum_eb<-z$p.value #get the p-value and store it
p_spl1_cum_eb
```

```
## [1] 0.9998301
```

False discovery error rate control

```
p <- c(p_1bn7_cum_eb, p_1znj_cum_eb, p_ache_cum_eb, p_spl1_cum_eb)
index_max_accepted_p_value <- FDR(p, qlevel = 0.25)
threshold_p_value <- p[index_max_accepted_p_value]
p <= threshold_p_value
```

```
## [1] FALSE FALSE  TRUE FALSE
```

```
#exploratory part, what if we have stricter FDR error rate?
index_max_accepted_p_value <- FDR(p, qlevel = 0.05)
threshold_p_value <- p[index_max_accepted_p_value]
p <= threshold_p_value
```

```
## logical(0)
```

Further exploratory analysis, not pre-registered

Is the cummulative barrier significantnly higher for the new heuristic?

1BN7

```
old_table <- read.table('./CD-1.0.1/1BN7/cum-eb.dat')
new_table <- read.table('./CD-1.1/1BN7/cum-eb.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##          mean      sd median
## [1,] 3.490714 0.30153    3.4
```

```
mysummary(y)
```

```
##          mean        sd median
## [1,] 3.533571 0.2997867    3.5
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] 0.1425437
```

```
z<-t.test(y, x, alternative="greater") #perform the t-test
p_1bn7_gcum_eb<-z$p.value #get the p-value and store it
p_1bn7_gcum_eb
```

```
## [1] 0.1170203
```

1ZNJ

```
old_table <- read.table('./CD-1.0.1/1ZNJ/cum-eb.dat')
new_table <- read.table('./CD-1.1/1ZNJ/cum-eb.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##          mean        sd median
## [1,] 4.961429 0.1638501      5
```

```
mysummary(y)
```

```
##          mean        sd median
## [1,] 5.003571 0.1436476      5
```

```
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] 0.2735122
```

```
z<-t.test(y, x, alternative="greater") #perform the t-test
p_1znj_gcum_eb<-z$p.value #get the p-value and store it
p_1znj_gcum_eb
```

```
## [1] 0.01143977
```

ACHE

```r
old_table <- read.table('./CD-1.0.1/ACHE/cum-eb.dat')
new_table <- read.table('./CD-1.1/ACHE/cum-eb.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##          mean       sd median
## [1,] 4.106429 0.492565    4.1
```

```r
mysummary(y)
```

```
##      mean        sd median
## [1,] 4.02 0.4295221      4
```

```r
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] -0.1870263
```

```r
z<-t.test(y, x, alternative="greater") #perform the t-test
p_ache_gcum_eb<-z$p.value #get the p-value and store it
p_ache_gcum_eb
```

```
## [1] 0.9406028
```

SPL1

```r
old_table <- read.table('./CD-1.0.1/SPL1/cum-eb.dat')
new_table <- read.table('./CD-1.1/SPL1/cum-eb.dat')
x <- old_table$V1
y <- new_table$V1
mysummary(x)
```

```
##          mean        sd median
## [1,] 2.853571 0.4846434    2.8
```

```r
mysummary(y)
```

```
##          mean        sd median
## [1,] 3.058571 0.4605261    3.1
```

```r
sd <- sqrt((sd(x)^2 + sd(y)^2)/2)
d <- (mean(y)-mean(x))/sd
d
```

```
## [1] 0.4336435
```

```r
z<-t.test(y, x, alternative="greater") #perform the t-test
p_spl1_gcum_eb<-z$p.value #get the p-value and store it
p_spl1_gcum_eb
```

```
## [1] 0.00016989
```

False discovery error rate control

```
p <- c(p_1bn7_gcum_eb, p_1znj_gcum_eb, p_ache_gcum_eb, p_spl1_gcum_eb)
index_max_accepted_p_value <- FDR(p, qlevel = 0.25)
threshold_p_value <- p[index_max_accepted_p_value]
p <= threshold_p_value
```

```
## [1]  TRUE  TRUE FALSE  TRUE
```

```
#exploratory part, what if we have stricter FDR error rate?
index_max_accepted_p_value <- FDR(p, qlevel = 0.05)
threshold_p_value <- p[index_max_accepted_p_value]
p <= threshold_p_value
```

```
## [1] FALSE  TRUE FALSE  TRUE
```