Extending scPower to complex designs

Matthias Heinig, Katharina Schmid

14 June, 2021

Contents

Power estimations for the standard eQTL analysis and two-group DE analysis are directly implemented in scPower. Additionally, it is possible to estimate the power for more complex designs based on generalized linear models (GLMs) with a few small extensions.

For this, we follow the approach of Lyes et al, 2007 (see pubmed). The main steps for this are:

1) Simulate one example data set
2) Analyse simulated data set with GLMs and get test statistics
3) Plug-in the test statistics in the complex design power formula and combine it with the expression probability model

This vignette shows the approach for a three-group comparison (group0, group1, group2), which could for example represent different disease subtypes. We evaluate the required sample size assuming a balanced distribution between the groups.

```
library(scPower)
library(reshape2)
library(ggplot2)
```

Decide on experimental parameters

Corresponding to other evaluations with our tool, first a few parameters need to be chosen. Additionally to selecting the main experimental parameter (number samples, number cells per sample and read depth), we simulate in this example fold changes for 200 DE genes with different strengths between group0-group1 and group0-group2 using the respective functions from scPower. Instead of simulating, effect size from reference studies can be taken when available.

```
#Main experimental parameters
nSamples<-20
nCells<-1000
readDepth<-10000

ct<-"CD4 T cells"

#Standard significance threshold
sign_threshold<-0.05

#Simulate effect sizes
set.seed(1)
num_de_genes<-200
```

```
ranks<-uniform.ranks.interval(start=1,end=20000,numGenes=num_de_genes)
de_group1<-effectSize.DE.simulation(mean=1,sd=0.5,numGenes=num_de_genes)
de_group2<-effectSize.DE.simulation(mean=2,sd=0.5,numGenes=num_de_genes)

de_priors<-data.frame(ranks=ranks,
                      foldChange_group1=de_group1,
                      foldChange_group2=de_group2)
```

Simulate a three group single cell data set and analyse it to get the test statistics

In contrast to the standard DE/eQTL workflow, for more complex design choices we need to simulate once a data set with the design to get the test statistics of the generalized linear model for this setting. In this hybrid approach of analytic and simulation method, this step needs to be done once per number of cells and read depth, but is independent of the sample size.

We apply scPower to estimate the mean and dispersion values based on the read depth and the selected expression prior and scale them by the number of cells per person and cell type to get the parameters of the pseudbulk. The mean values for group 1 and group 2 are then shifted by the simulated fold change.

For the simulation itself, only the simulation of DE genes is necessary. The chosen sample size of the simulated data set is independent from the target sample size of the experiment, because the test statistics can be scaled by the sample size. We would in general recommend a large sample size for the simulation that will lead to more robust estimations for the test statistics.

To save memory, we simulate separately the counts for each DE gene and calculate immediatly the likelihood of the full and the null model based on a negative binomial regression. The full model assumes three different groups, while the null model assumes that all samples come from the same distribution. The log-likelihoods of the full and the reduced model are then combined to the non-centrality parameter.

In an alternative strategie, simulation-based tools such as powsimR or muscat could be used to simulate the count matrix. The approach with scPower has the advantage that it is a bit more flexible (simulting only the DE genes and starting from the pseudobulk), which can save memory during the simulation.

```
#Get single cell negative binomial parameter
nb_values<-scPower:::estimate.mean.dsp.values(readDepth,
                                read.umi.fit[read.umi.fit$type=="10X_PBMC_1",],
                                gamma.mixed.fits,ct,disp.fun.param)

#Scale it with the number of cells to get the pseudobulk NB parameters
nb_values$mean_pseudobulk<-nb_values$means*nCells
nb_values$dsp_pseudobulk<-nb_values$dsp/nCells

#Extract the DE genes
nb_values<-nb_values[order(nb_values$means,decreasing=TRUE),]

#Simulate a count matrix for the DE genes
de_base_values<-nb_values[de_priors$ranks,]

#Shift mean by fold change for the second and third group
de_base_values$means_pseudobulk_g1<-de_base_values$mean_pseudobulk *
  de_priors$foldChange_group1
de_base_values$means_pseudobulk_g2<-de_base_values$mean_pseudobulk *
  de_priors$foldChange_group2
```

```r
#Simulate a count matrix for DE genes
n_sim<-300
groups<-as.factor(rep(c(1,2,3),each=n_sim/3))
llikehoods<-NULL
for(i in 1:nrow(de_base_values)){
  mean_vector<-c(rep(de_base_values$mean_pseudobulk[i],n_sim/3),
                 rep(de_base_values$means_pseudobulk_g1[i],n_sim/3),
                 rep(de_base_values$means_pseudobulk_g2[i],n_sim/3))
  counts<-rnbinom(length(mean_vector),mu=mean_vector,
                  size=1/de_base_values$dsp_pseudobulk[i])

  #Estimate log likelihood of the full model
  full_model<-glm(counts ~ groups,family="gaussian")
  logLik_full<-logLik(full_model)

  #Estimate log likelihood of the reduced model
  reduced_model<-glm(counts ~ 1, family="gaussian")
  logLik_reduced<-logLik(reduced_model)

  llikehoods<-rbind(llikehoods,
                    data.frame(logLik_full,logLik_reduced))

}

#Estimate non-centrality parameter
llikehoods$non_cent_param<--2*(llikehoods$logLik_reduced - llikehoods$logLik_full)
```

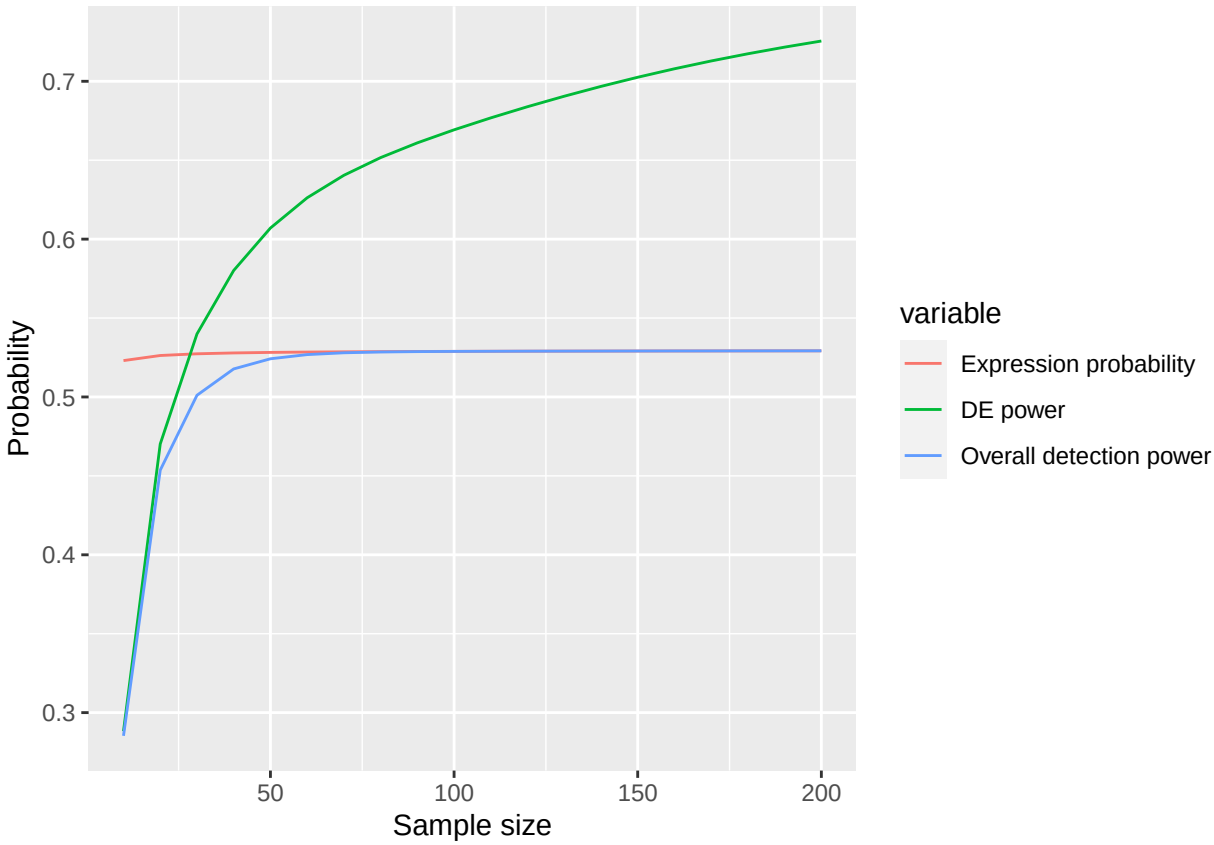
Plug-in the test statistics in the complex design power formular and combine it with the expression probability model

Equivalent to our standard scPower pipeline, we calculate the overall detection power as a combination of the expression probability and the power. For this reason, we estimate now first the expression probability and perform multiple testing correction of the significance threshold based on the estimated number of expressed genes, here using the Bonferroni method.

The complex power itself depends on the non-centrality parameter, the multiple testing corrected significance threshold, the degrees of freedom (= #parameters of the full model - #parameters of the reduced model) and the effective sample size. The effective sample size is the ratio of the target sample size of the new data set and the sample size of the simulated data set. The power is calculated separately for each DE gene and the combined with the respective probability.

```r
#' Calculate the power for a complex design
#'
#' @param v non centrality parameter: -2*(ll_reduced-ll_full)
#' @param h Degrees of freedom
#' @param alpha multiple testing corrected significance threshold
#' @param effsize effective sample size: #samples (new data set) / #samples (simulation data)
#'
#' @return power of the complex design
power.complex<-function(v,h,alpha,effssize=1){
  if(is.nan(v)){
    return(0)
  }
```

```
  return(pchisq(qchisq(1-alpha,df=h),df=h,ncp=effssize*v,lower.tail=FALSE))
}

res<-NULL
for(nSamples in seq(10,200,by=10)){


  #Get expression probability using scPower
  exp_prob<-scPower::estimate.exp.prob.param(nSamples,readDepth,nCells,
                                             read.umi.fit[read.umi.fit$type=="10X_PBMC_1",],
                                             gamma.mixed.fits,ct,disp.fun.param)
  num_exp_genes<-sum(exp_prob)

  #Adjust significance threshold
  mt_threshold<-sign_threshold/num_exp_genes

  power<-sapply(llikehoods$non_cent_param,power.complex,h=2,alpha=mt_threshold,
         effssize=nSamples/n_sim)

  #Filter for the expression probability of the DE genes
  exp_prob<-sort(exp_prob,decreasing = TRUE)
  exp_prob<-exp_prob[de_priors$ranks]

  #Calculate the overall detection power
  overall_power<-exp_prob*power

  res<-rbind(res,
             data.frame(nSamples,
                        exp_prob=mean(exp_prob),
                        power=mean(power),
                        overall_power=mean(overall_power)))
}

res_plot<-melt(res,id.vars="nSamples")
ggplot(res_plot,aes(x=nSamples,y=value,color=variable))+
  geom_line()+ylab("Probability")+xlab("Sample size")+
  scale_color_discrete(labels=c("Expression probability","DE power","Overall detection power"))
```


0.7
0.6
0.5
0.4
0.3
Probability
50
100
150
200
Sample size
variable
Expression probability
DE power
Overall detection power