

Power Analyses for SEQLinkage

Wed Apr 2 10:45:41 CDT 2014 | Let's rock and roll!

Basics

We simulate and analyze two autosomal recessive genes GJB2 and SLC26A4 and two autosomal dominant genes MYO7A and MYH9. The program takes two genes source data as input.

As we call seqlink from inside of the program multiple times we'd like to speed up the execution by using ramdisk of 8000MB.

```
sudo mkdir -p /ramcache
sudo mount -t tmpfs ramfs -o size=8000m /ramcache
cd /ramcache
ln -s ~/SVN/SEQLinco/simulations/LinkagePowerCalc.py pcal.py
```

Hierarchy of simulation parameters

I want to draw one figure with 4 panels. Each panel is one mode of inheritance, namely A&B are recessive and compound recessive using recessive genes; C&D are dominant and compound dominant using recessive genes. On each panel, the X-axis is sample size and Y-axis is allelic heterogeneity. Powers are contour plots of two colors, one color for new method, one for old method. The only parameter not scanned here is --offsprings. For now I set it in between 3 to 8. It can be varied for additional simulations in supplemental. In this figure only one of the gene pair will be reported and the complementary figure can be found in supplemental.

Linkage analysis parameters are always -K 0.01 -W 0 -M 1. See seqlink -h for details.

Example command

```
python pcal.py -g MYH9.tsv MYO7A.tsv -m dominant -r 10 -n 20 \
--blueprint blueprint.txt --tempdir /ramcache \
--run-linkage -K 0.01 --moi AD -W 0 -M 1 --output-entries 0
```

Power calculation script

Script generator:

PYTHON

```
name = 'pcal.sh'
rep = 500
batch = 50
count = 0
gdict = {'recessive': 'SLC26A4.tsv GJB2.tsv', 'compound_recessive': 'SLC26A4.tsv GJB2.tsv',
         'dominant': 'MYO7A.tsv MYH9.tsv'}
mdict = {'recessive': 'AR', 'compound_recessive': 'AR', 'dominant': 'AD'}
fdict = {'': 'CHPResult.csv', '--single-markers': 'SNVResult.csv'}
folder = ''
```

```

with open(name, 'w') as f:
    for m in ['recessive', 'dominant', 'compound_recessive']:
        for a in [' ', '-a']:
            for p in [i * 0.01 for i in range(0, 110, 10)]:
                for n in range(5, 81, 1):
                    for method in [' ', '--single-markers']:
                        if count % batch == 0:
                            folder = 'B' + str(count / batch + 1)
                            mkdir = 'mkdir -p {}'.format(folder)
                        else:
                            mkdir = ''
                        count += 1
                        f.write("{}12}.pcal.py --debug -s 3 8 -g {} -m {} -n {} -p {} {}4} -r {}5} \"\
                            \"--ofile {}9} {}10} --blueprint blueprint.txt {}6} --jobs 1 \"\
                            \"--run-linkage -K 0.01 -W 0 -M 1 --moi {}7} --output-entries 0 \"\
                            \"> {}11}/{8}.{}9}\n\".\
                            format(gdict[m], m, n, p, 1-p, rep, method, mdict[m],
                                fdict[method], "RUN%s" % count, a, folder, mkdir))

```

Use command below to run the generated script on desktop:

```
cat pcal.sh | gw-parallel -j 7
```

Or on cluster:

pbs

```

#PBS -o info/log
#PBS -e info/err
#PBS -N SLsim
#PBS -t 0-4355
#PBS -l walltime=96:00:00
#
cd $PBS_O_WORKDIR
mkdir -p /mnt/ram/slsim/RUN$PBS_ARRAYID

rep=500
pNarray=$(seq 0 10 100)
MOI=( 'recessive' 'dominant' 'compound_recessive' ) #MOI in simulation
Mmap=( 'AR' 'AD' 'AR' ) #MOI inmlink
Gmap=( 'SLC26A4.tsv GJB2.tsv' 'MYO7A.tsv MYH9.tsv' 'SLC26A4.tsv GJB2.tsv' ) #gene pairs
Mtype=( ' ' '--single-marker' )
Fmap=( 'CHPResult.csv' 'SNVResult.csv' ) #result
aArray=( ' ' '-a' )

#compute parameters
N=$(bc <<< $PBS_ARRAYID/33*2+5) #family number
pN=$(bc <<< $PBS_ARRAYID/33*11)
p=$(bc <<< "1 - $pN/10")
q=$(bc <<< "1 - $p")
m=$(bc <<< $PBS_ARRAYID/33/11*3)
a=${aArray[$(bc <<< $PBS_ARRAYID/33/11/3*2)]}
t=$(bc <<< $PBS_ARRAYID/33/11/3/2)
echo "./LinkagePowerCalc.py --debug -s 3 8 -g ${Gmap[$m]} \
    -m ${MOI[$m]} -n $N -p $p $q -r $rep \
    --ofile RUN${PBS_ARRAYID} $a --blueprint blueprint.txt \
    ${Mtype[$t]} --jobs 1 \
    --run-linkage -K 0.01 -W 0 -M 1 --tempdir /mnt/ram/slsim/RUN$PBS_ARRAYID \
    --moi ${Mmap[$m]} --output-entries 0 \
    --quiet > result/RUN${PBS_ARRAYID}.${Fmap[$t]}"
./LinkagePowerCalc.py --debug -s 3 8 -g ${Gmap[$m]} \
    -m ${MOI[$m]} -n $N -p $p $q -r $rep \
    --ofile RUN${PBS_ARRAYID} $a --blueprint blueprint.txt \
    ${Mtype[$t]} --jobs 1 \
    --run-linkage -K 0.01 -W 0 -M 1 --tempdir /mnt/ram/slsim/RUN$PBS_ARRAYID \
    --moi ${Mmap[$m]} --output-entries 0 \
    --quiet > result/RUN${PBS_ARRAYID}.${Fmap[$t]}
exit 0

```

makefile

```
run:
    qsub SubmitJobs.pbs
clean:
    rm -rf RUN* info/* cache/
test:
    PBS_O_WORKDIR=. PBS_ARRAYID=6 bash SubmitJobs.pbs
.PHONY: run clean test
```

Power calculation result

Consolidate all result data into database PowerCalc

```
cat CHPResult.csv.* > CHPResult.csv
cat SNVResult.csv.* > SNVResult.csv
./pcal.py --print-header | sqlite PowerCalc -i CHPResult.csv --as CHP --header - -d", "
./pcal.py --print-header | sqlite PowerCalc -i SNVResult.csv --as SNV --header - -d", "
```

To view attributes of the database

```
sqlite PowerCalc -s
sqlite PowerCalc -s CHP
sqlite PowerCalc -s SNV
```

All output data and results are backed up to 040*.tar.gz for future reference. Now let's have a rough feeling on power comparison between the methods

BASH

```
for k in 0 1; do
    for i in recessive dominant compound_recessive; do
        for j in 1 2; do
            echo $i Gene$j, allelic heterogeneity $k
            echo CHP=$(sqlite PowerCalc "select avg(plod$j) from CHP where moi = '$i' and ahet = $k" \
            SNV=$(sqlite PowerCalc "select avg(plod$j) from SNV where moi = '$i' and ahet = $k" \
        done
    done
done
```

Results below are based on data from 0430.tar.gz:

OUTPUT

```
recessive Gene1, allelic heterogeneity 0
CHP=0.817241508458 SNV=0.795367095338
recessive Gene2, allelic heterogeneity 0
CHP=0.797376342463 SNV=0.790279430082
dominant Gene1, allelic heterogeneity 0
CHP=0.895763085399 SNV=0.858033057851
dominant Gene2, allelic heterogeneity 0
CHP=0.891829201102 SNV=0.811327823691
compound_recessive Gene1, allelic heterogeneity 0
CHP=0.815912978502 SNV=0.63308014474
compound_recessive Gene2, allelic heterogeneity 0
CHP=0.799027536692 SNV=0.628935586131
recessive Gene1, allelic heterogeneity 1
CHP=0.817835222056 SNV=0.522331617895
recessive Gene2, allelic heterogeneity 1
CHP=0.797724920312 SNV=0.659947881302
dominant Gene1, allelic heterogeneity 1
CHP=0.895184573003 SNV=0.830385674931
dominant Gene2, allelic heterogeneity 1
CHP=0.892539944904 SNV=0.762303030303
```

```
compound_recessive Gene1, allelic heterogeneity 1
CHP=0.816515557635 SNV=0.423349862259
compound_recessive Gene2, allelic heterogeneity 1
CHP=0.798936799341 SNV=0.565154320847
```

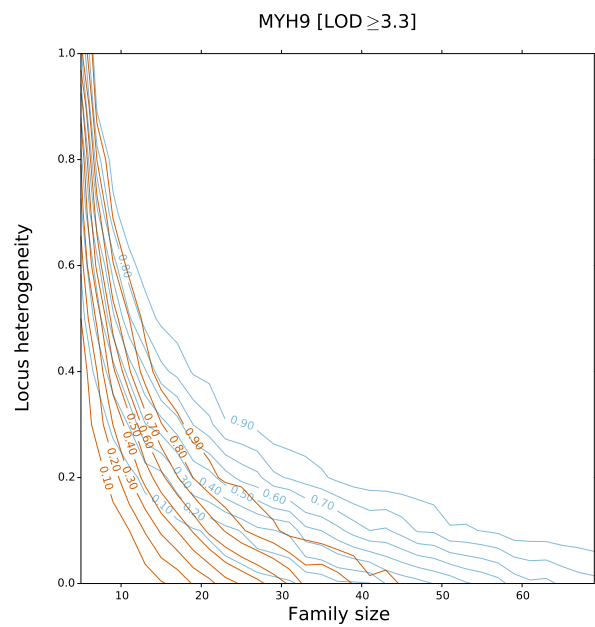
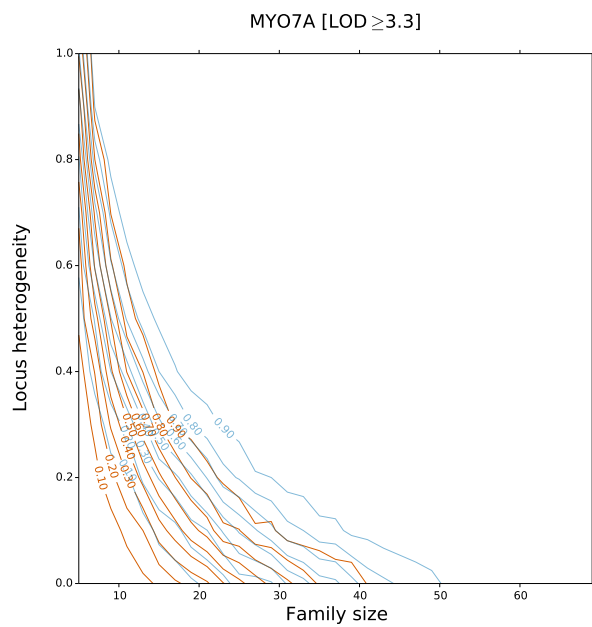
Thu Apr 3 14:51:32 CDT 2014 | A sleepy afternoon with voltaren all over my hands ...

Power plot for LOD

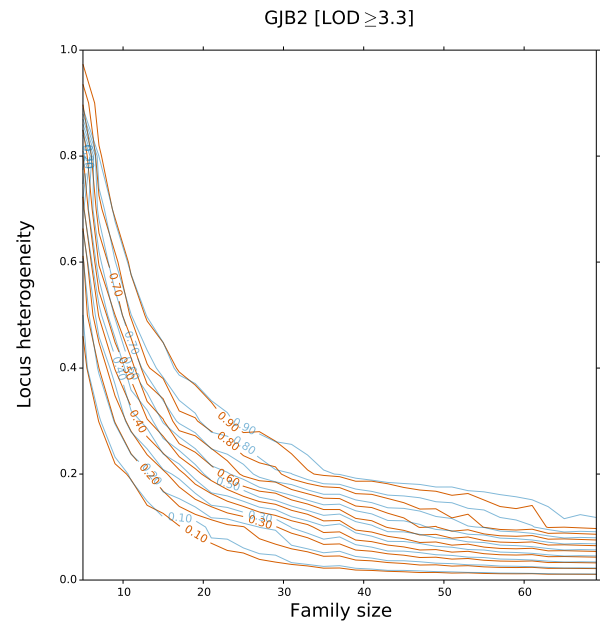
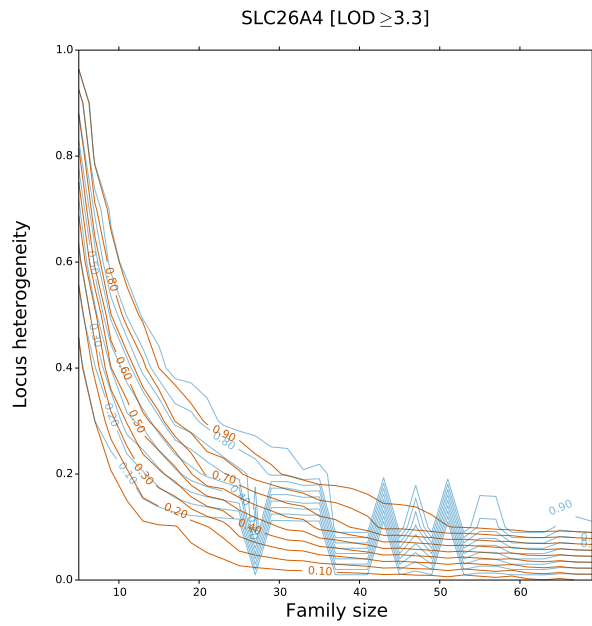
Script see PowerContour .py under this folder. Graphs below are based on data from 0430.tar.gz:

Without allelic heterogeneity

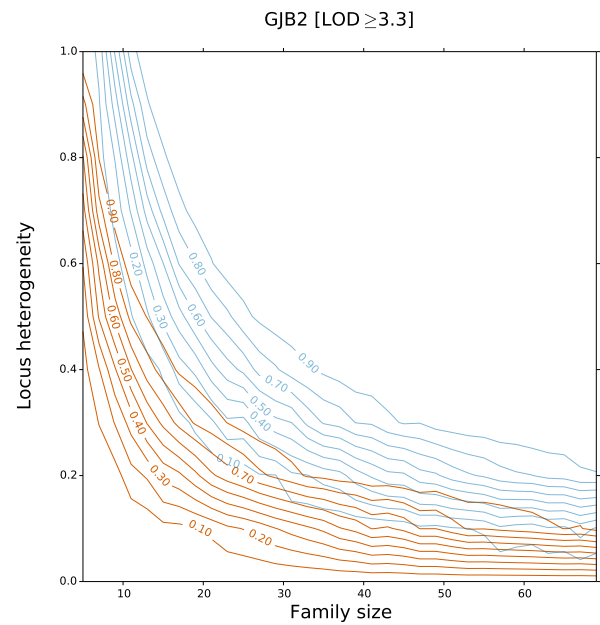
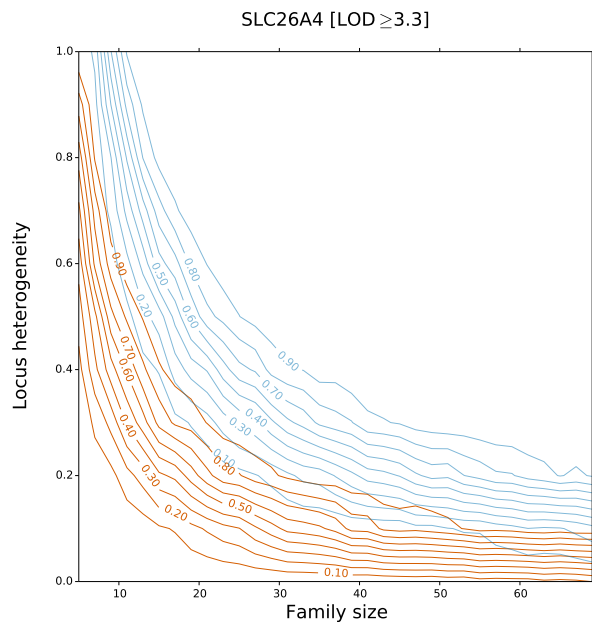
Dominant



Recessive

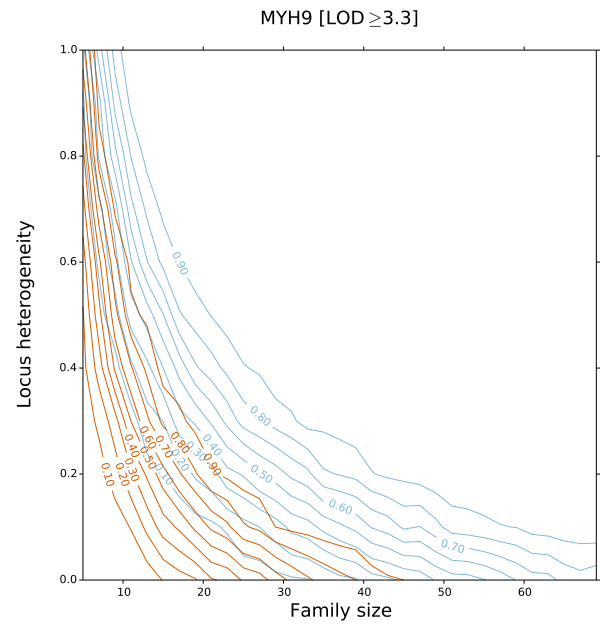
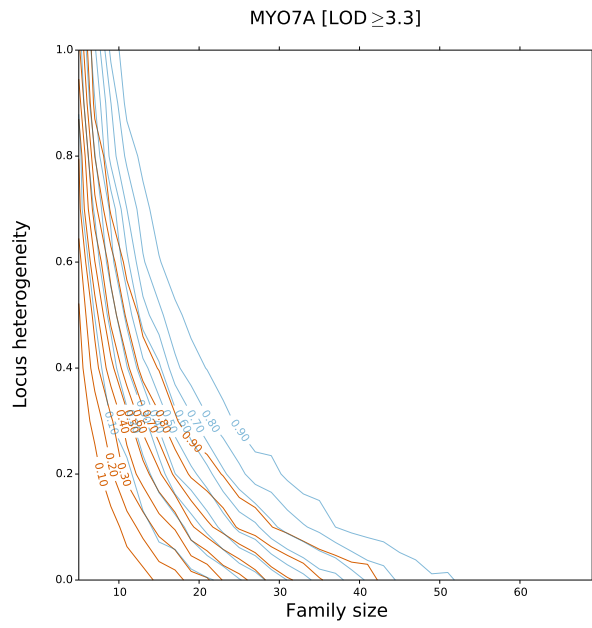


Compound recessive

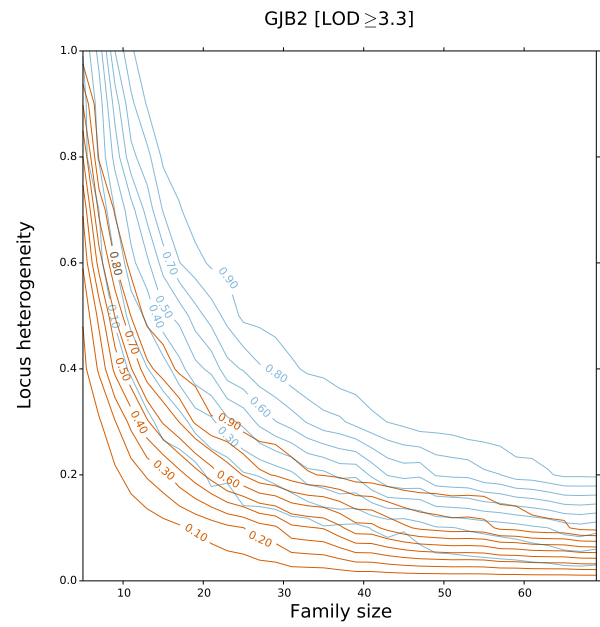
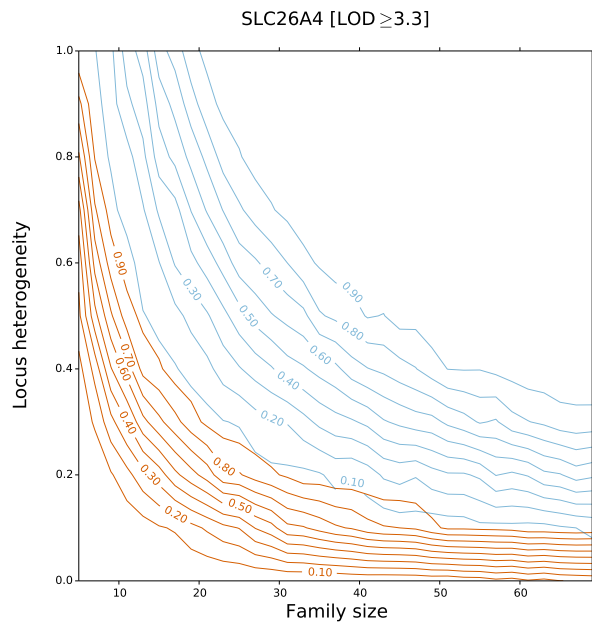


With allelic heterogeneity

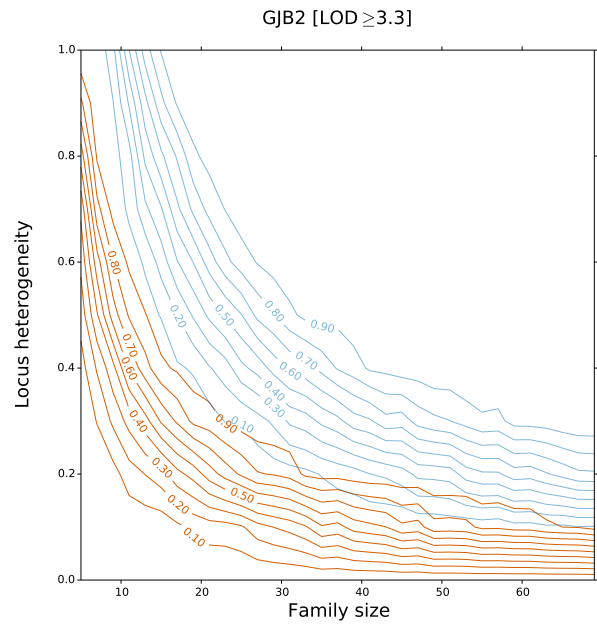
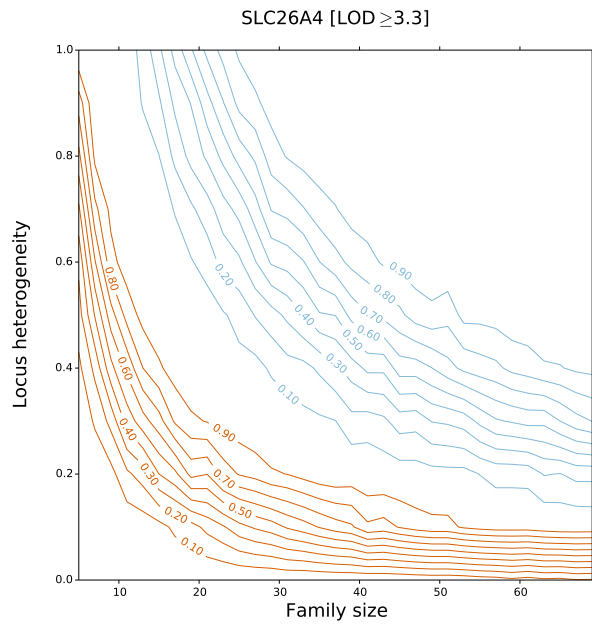
Dominant



Recessive



Compound recessive

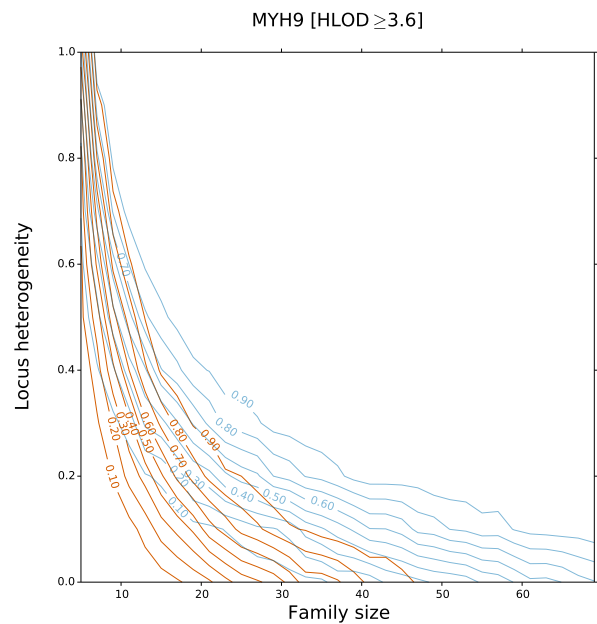
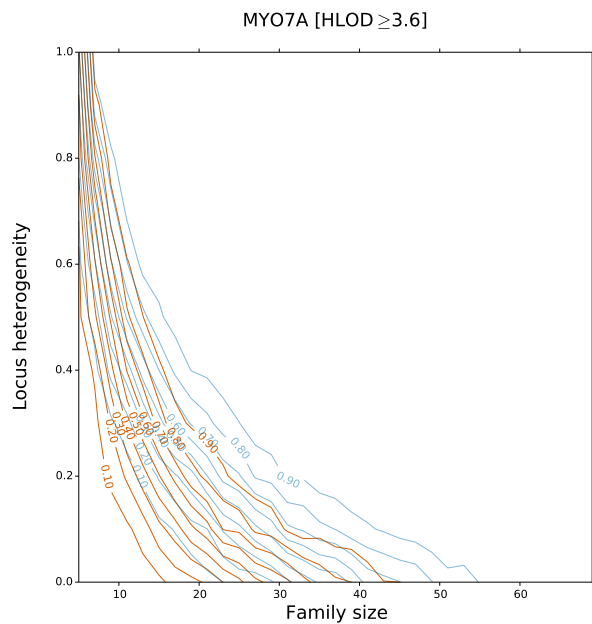


Power plot for HLOD

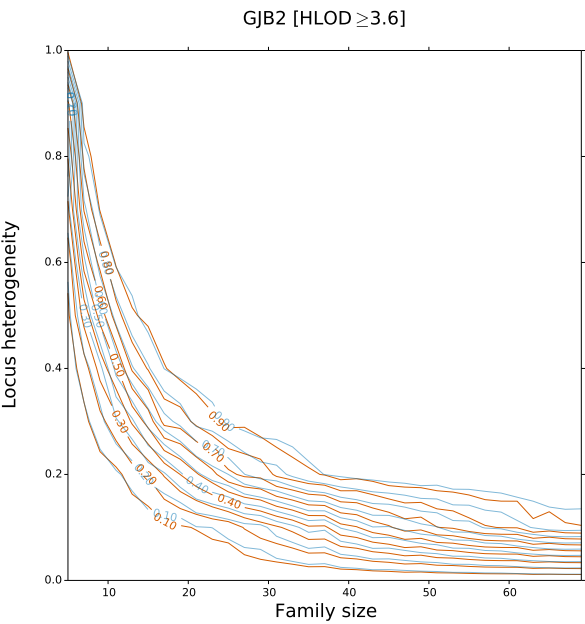
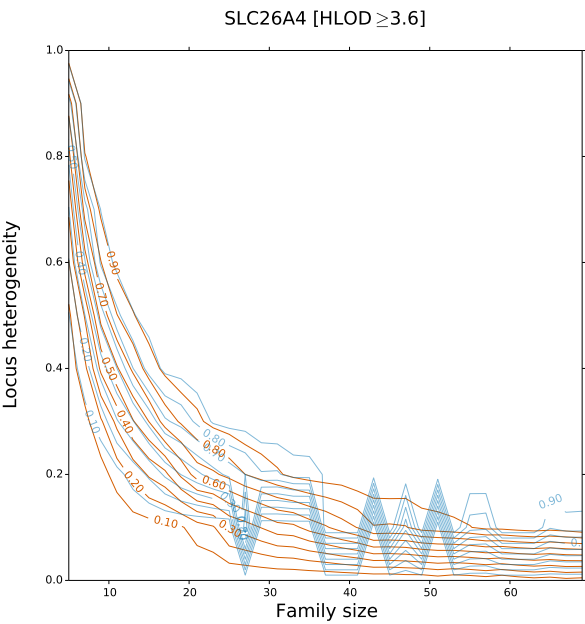
Script see PowerContour .py under this folder. Graphs below are based on data from 0430 .tar .gz:

Without allelic heterogeneity

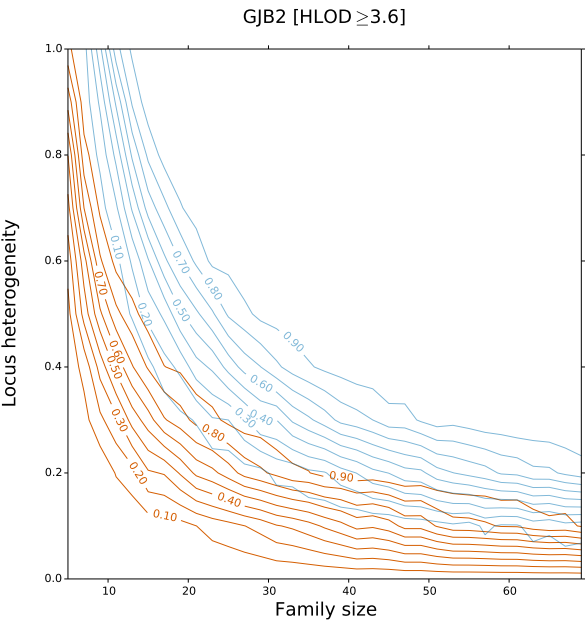
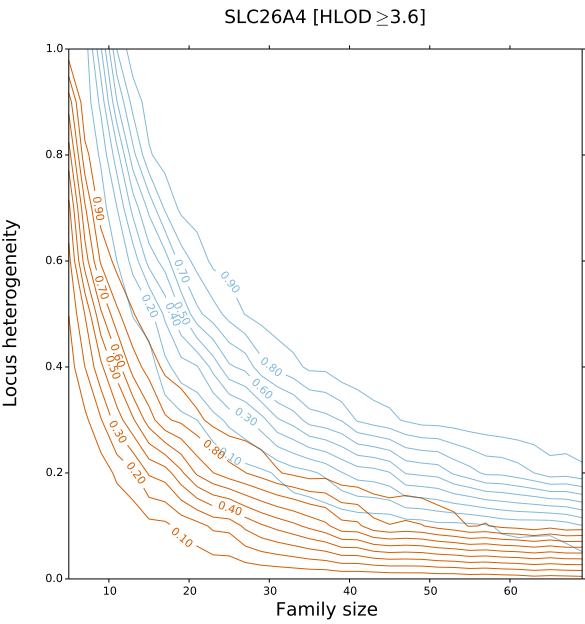
Dominant



Recessive

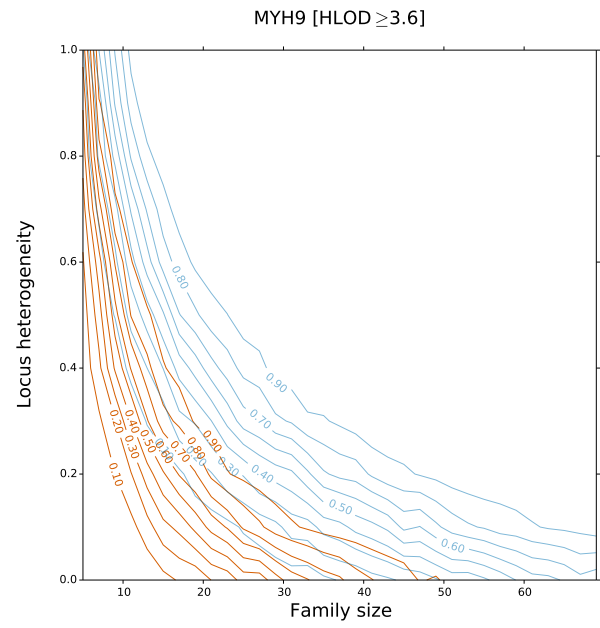
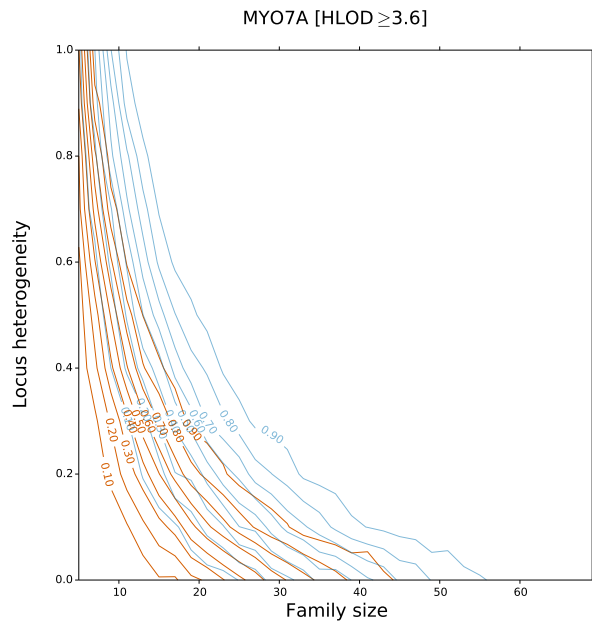


Compound recessive

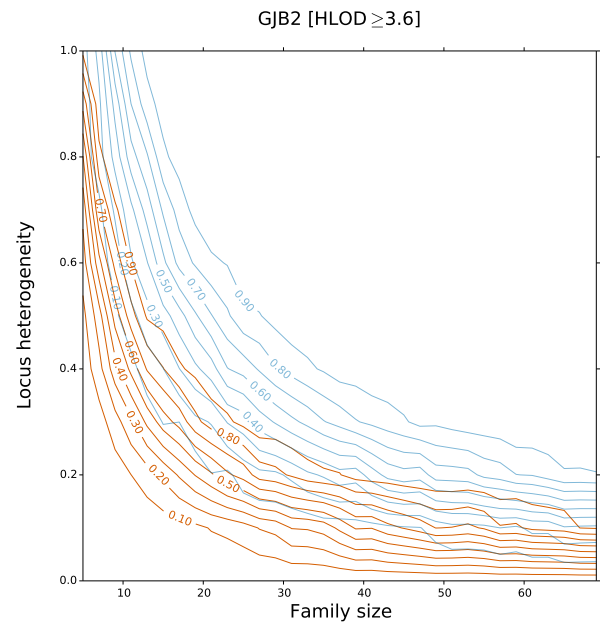
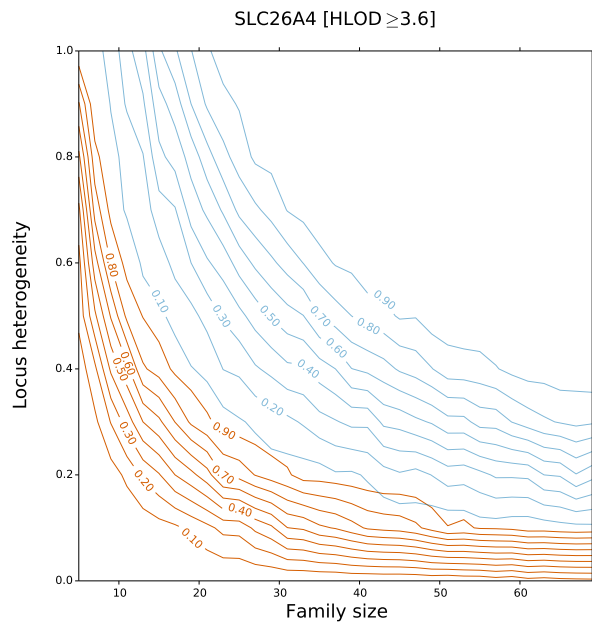


With allelic heterogeneity

Dominant



Recessive



Compound recessive

