6.1 Power and sample size calculations

Many simple settings lend themselves to analytic power calculations, where closed form solutions are available. Other situations may require an empirical calculation, where repeated simulation is undertaken.

6.1.1 Analytic power calculation

It is straightforward to find power or sample size (given a desired power) for two sample comparisons of either continuous or categorical outcomes. We show simple examples for comparing means and proportions in two groups and supply additional information on analytic power calculation available for more complex methods.

SAS

```
/* find sample size for two-sample t-test */
proc power;
   twosamplemeans groupmeans=(0 0.5) stddev=1 power=0.9 ntotal=.;
run;
```

```
/* find power for two-sample t-test */
proc power;
   twosamplemeans groupmeans=(0 0.5) stddev=1 power=. ntotal=200;
run;
```

The latter call generates the following output:

```
The POWER Procedure
Two-sample t Test for Mean Difference
     Fixed Scenario Elements
Distribution                Normal
Method                       Exact
Group 1 Mean                     0
Group 2 Mean                   0.5
Standard Deviation               1
Total Sample Size              200
Number of Sides                  2
Null Difference                  0
Alpha                         0.05
Group 1 Weight                   1
Group 2 Weight                   1

Computed Power
Power 0.940
```

```
/* find sample size for two-sample test of proportions */
proc power;
   twosamplefreq test=pchi ntotal=. groupproportions=(.1 .2) power=0.9;
run;
```

```
/* find power for two-sample test of proportions */
proc power;
   twosamplefreq test=pchi ntotal=200 groupproportions=(.1 .2) power=.;
run;
```

Note: The `power` procedure also allows power calculations for the Wilcoxon rank-sum test, the log-rank and related tests for censored data, paired tests of means and proportions, correlations, and for ANOVA and linear and logistic regression. The syntax is similar with the desired output of power, total sample size, effect size, alpha level, or variance listed with a missing value (a period after the equals sign).

R

```
# find sample size for two-sample t-test
power.t.test(delta=0.5, power=0.9)
```

```
# find power for two-sample t-test
power.t.test(delta=0.5, n=100)
```

The latter call generates the following output:

```
     Two-sample t test power calculation
              n = 100
          delta = 0.5
             sd = 1
      sig.level = 0.05
          power = 0.9404272
    alternative = two.sided
 NOTE: n is number in *each* group
```

```
# find sample size for two-sample test of proportions
power.prop.test(p1=.1, p2=.2, power=.9)
```

```
# find power for two-sample test of proportions
power.prop.test(p1=.1, p2=.2, n=100)
```

Note: The `power.t.test()` function requires exactly four of the five arguments (sample size in each group, power, difference between groups, standard deviation, and significance level) to be specified. Default values exist for `sd=1` and `sig.level=0.05`. Other power calculation functions can be found in the `pwr` package.

6.1.2 Simulation-based power calculations

In some settings, analytic power calculations may not be readily available. A straightforward alternative is to estimate power empirically, simulating data from the proposed design under given assumptions regarding the alternative.

We consider a study of children clustered within families. Each family has 3 children; in some families all 3 children have an exposure of interest, while in others just 1 child is exposed. In the simulation, we assume that the outcome is multivariate normal with higher mean for those with the exposure, and 0 for those without. A compound symmetry correlation is assumed, with equal variances at all times. We assess the power to detect an exposure effect where the intended analysis uses a random intercept model (4.2.2) to account for the clustering within families.

With this simple covariance structure it is trivial to generate correlated errors directly, as in the SAS code below; an alternative which could be used with more complex structures in SAS would be `proc simnorm` (1.10.6).

SAS

```
data simpower1;
   effect = 0.35;  /* effect size */
   corr = 0.4;     /* desired correlation */
   covar = (corr)/(1 - corr);  /* implied covariance given variance = 1*/
   numsim = 1000;  /* number of datasets to simulate */
   numfams = 100;  /* number of families in each dataset */
   numkids = 3;    /* each family */
   do simnum = 1 to numsim;   /* make a new dataset for each simnum */
      do famid = 1 to numfams; /* make numfams families in each dataset */
         inducecorr = normal(42)* sqrt(covar);
                    /* this is the mechanism to achieve the desired
                            correlation between kids within family */
         do kidnum = 1 to numkids;   /* generate each kid */
            exposed = ((kidnum eq 1) or (famid le numfams/2)) ;
                             /* assign kid to be exposed */
            x = (exposed * effect) +
                (inducecorr + normal(0))/sqrt(1 + covar);
            output;
         end;
      end;
   end;
run;
```

In the code above, the integer provided as an argument in the initial use of the `normal` function sets the seed used for all calls to the pseudo-random number generator, so that the results can be exactly replicated, if necessary (see section 1.10.9.) Next, we run the desired model on each of the simulated datasets, using the `by` statement (A.6.2) and saving the estimated fixed effects parameters using the `ODS` system (A.7).

```
ods select none;
ods output solutionf=simres;
proc mixed data=simpower1 order=data;
by simnum;
   class exposed famid;
   model x = exposed / solution;
   random int / subject=famid;
run;
ods select all;
```

Finally, we process the resulting output dataset to generate an indicator of rejecting the null hypothesis of no exposure effect.

```
data powerout;
set simres;
   where exposed eq 1;
   reject=(probt lt 0.05);
run;
```

Note: The proportion of rejections shown in the results of `proc freq` is the empirical estimate of power.

```
proc freq data=powerout;
   tables reject / binomial (level='1');
run;
```

```
The FREQ Procedure
                                   Cumulative    Cumulative
reject    Frequency     Percent     Frequency      Percent
-----------------------------------------------------------
     0          153       15.30           153        15.30
     1          847       84.70          1000       100.00
```

The `binomial` option to `proc freq` provides asymptotic and exact CI for this estimated power:

```
Proportion                  0.8470
ASE                         0.0114
95% Lower Conf Limit        0.8247
95% Upper Conf Limit        0.8693

Exact Conf Limits
95% Lower Conf Limit        0.8232
95% Upper Conf Limit        0.8688
```

In R, we specify the correlation matrix directly, and simulate the multivariate normal.

R

```
library(MASS)
library(nlme)
# initialize parameters and building blocks
effect <- 0.35    # effect size
corr <- 0.4       # intrafamilial correlation
numsim <- 1000
n1fam <- 50       # families with 3 exposed
n2fam <- 50       # families with 1 exposed and 2 unexposed
vmat <- matrix(c  # 3x3 compound symmetry correlation
   (1,    corr, corr,
    corr, 1   , corr,
    corr, corr, 1    ), 3, 3)

# 1 1 1 ... 1 0 0 0 ... 0
x <- c(rep(1, n1fam), rep(1, n1fam), rep(1, n1fam),
       rep(1, n2fam), rep(0, n2fam), rep(0, n2fam))
# 1 2 ... n1fam 1 2 ... n1fam ...
id <- c(1:n1fam, 1:n1fam, 1:n1fam,
   (n1fam+1:n2fam), (n1fam+1:n2fam), (n1fam+1:n2fam))
power <- rep(0, numsim) # initialize vector for results
```

The concatenate function (`c()`) is used to glue together the appropriate elements of the design matrices and underlying correlation structure.

```
for (i in 1:numsim) {
   cat(i," ")
   # all three exposed
   grp1 <- mvrnorm(n1fam, c(effect, effect, effect), vmat)

   # only first exposed
   grp2 <- mvrnorm(n2fam, c(effect, 0,      0),      vmat)

   # concatenate the output vector
   y <- c(grp1[,1], grp1[,2], grp1[,3],
          grp2[,1], grp2[,2], grp2[,3])

   group <- groupedData(y ~ x | id)   # specify dependence structure
   res <- lme(group, random = ~ 1)    # fit random intercept model
   pval <- summary(res)$tTable[2,5]   # grab results for main parameter
   power[i] <- pval<=0.05             # is it statistically significant?
}

cat("\nEmpirical power for effect size of ", effect,
   " is ", round(sum(power)/numsim,3), ".\n", sep="")
cat("95% confidence interval is",
   round(prop.test(sum(power), numsim)$conf.int, 3), "\n")
```

This yields the following estimate.

```
Empirical power for effect size of 0.35 is 0.855.
95% confidence interval is 0.831 0.876
```