* STATA.OUTPUT -- Chapter 15

```
. clear
. *normal approximation

. input n x
          n   x
  1. 50 15
  2. end

. gen q = x/n

. display "lower bound =", q-1.960*sqrt(q*(1-q)/n)
lower bound = .17297749

. display "upper bound =", q+1.960*sqrt(q*(1-q)/n)
upper bound = .42702253

. *exact confidence intervals

. cii 50 15, level(95)

    Variable |    Obs        Mean    Std. Err.       [95% Conf. Interval]
-------------+-------------------------------------------------------------
             |     50          .3    .0648074        .1786178    .4460823

. *exact confidence intervals

. cii 20183 0, level(90)

    Variable |    Obs        Mean    Std. Err.       [90% Conf. Interval]
-------------+-------------------------------------------------------------
             |  20183           0           0               0    .0001484*

(*) one-sided, 95% confidence interval


. clear
. *normal approximation

. input n x
          n x
  1. 50 2
  2. end

. gen q = x/n

. display "lower bound =", q-1.960*sqrt(q*(1-q)/n)
lower bound = -.01431711

. display "upper bound =", q+1.960*sqrt(q*(1-q)/n)
upper bound = .09431711
```

```
. *exact confidence intervals

. cii 50 2, level(95)

    Variable |   Obs         Mean    Std. Err.       [95% Conf. Interval]
-------------+----------------------------------------------------------
             |    50          .04    .0277128        .0048814    .1371376

. clear
. *normal approximation

. input n x
        n  x
  1. 50 25
  2. end

. gen q = x/n

. display "lower bound =", q-1.960*sqrt(q*(1-q)/n)
lower bound = .36140707

. display "upper bound =", q+1.960*sqrt(q*(1-q)/n)
upper bound = .63859293


. *exact confidence intervals

. cii 50 25, level(95)

    Variable |   Obs         Mean    Std. Err.       [95% Conf. Interval]
-------------+----------------------------------------------------------
             |    50           .5    .0707107         .355273     .644727
```

```
. clear
. *loess smooth curve:  dizygotic twins by year of birth
. input year mm ff  mf
          year   mm   ff   mf
   1.    1983 137 130  91
   2.    1984 183 181 146
   3.    1985 201 201 162
   4.    1986 371 367 276
   5.    1987 527 500 382
   6.    1988 596 647 530
   7.    1989 729 742 588
   8.    1990 258 260 224
   9.    1991 366 330 325
  10.    1992 330 326 318
  11.    1993 313 312 328
  12.    1994 326 329 271
  13.    1995 301 313 343
  14.    1996 298 316 326
  15.    1997 301 311 358
  16.    1998 145 160 167
  17.    1999 338 312 386
  18.    2000 362 336 400
  19.    2001 414 443 505
  20.    2002 431 405 547
  21.    2003 500 506 656
  22. end

. gen dz = 2*mf/(mm+mf+ff)
. lowess dz year, bwidth(2) gen(DZ) nograph

. list dz DZ

     +----------------------+
     |       dz          DZ |
     |----------------------|
  1. | .5083799   .53583547 |
  2. |  .572549   .54324262 |
  3. | .5744681   .55144949 |
  4. | .5443787   .56164545 |
  5. | .5422285   .57362456 |
  6. | .5978568    .5857708 |
  7. | .5711510   .59775471 |
  8. | .6037736   .61002194 |
  9. | .6366308   .62334455 |
 10. | .6529774    .6380150 |
     |----------------------|
     |----------------------|
 15. | .7381443   .70750394 |
 16. | .7076271   .72061809 |
 17. | .7451738   .73417052 |
 18. | .7285975   .74757361 |
 19. | .7415565   .75988048 |
 20. |  .791034   .77245559 |
 21. | .7894104   .78777085 |
     +----------------------+
```

```
. clear
. *sample size calculation of n for p0 = 0.1 and an odds ratio of or = 3

. input p0 or alpha beta

            p0 or alpha beta
          1. 0.1 3 0.95 0.10
          2. end

. gen p1 = p0*or/(1+p0*(or-1))
. gen zI = invnorm(alpha)
. gen zII = invnorm(1-beta)
. gen p = (p0+p1)/2
. gen s0 = sqrt(2*p*(1-p))
. gen s1 = sqrt(p0*(1-p0)+p1*(1-p1))
. gen n = (zI*s0+zII*s1)ˆ2/(p0-p1)ˆ2

. list p0 s0 p1 s1 n

     +--------------------------------------------+
     | p0         s0     p1         s1          n |
     |--------------------------------------------|
     | .1   .5373546    .25   .5267827   108.0171 |
     +--------------------------------------------+


.*power calculation

. sampsi 0.10 0.25, onesided nocontinuity

Estimated sample size for two-sample comparison of proportions

Test Ho: p1 = p2, where p1 is the proportion in population 1
and p2 is the proportion in population 2
Assumptions:
         alpha =   0.0500  (one-sided)
         power =   0.9000
            p1 =   0.1000
            p2 =   0.2500
         n2/n1 =   1.00
Estimated required sample sizes:
            n1 =      109
            n2 =      109
```

. *CALCULATIONS OF POWER AND SAMPLE SIZES FOR A VARIETY OF SITUATIONS

. sampsi 10 11, n1(50) sd1(7.07106) alpha(0.05) onesided

```
Estimated power for two-sample comparison of means

Test Ho: m1 = m2, where m1 is the mean in population 1
and m2 is the mean in population 2
Assumptions:
         alpha =   0.0500  (one-sided)
            m1 =       10
            m2 =       11
           sd1 =  7.07106
           sd2 =  7.07106
sample size n1 =       50
            n2 =       50
         n2/n1 =     1.00
Estimated power:
         power =   0.1742
```

. sampsi 10 14, n1(50) sd1(7.07106) alpha(0.05) onesided

```
Estimated power for two-sample comparison of means

Test Ho: m1 = m2, where m1 is the mean in population 1
and m2 is the mean in population 2
Assumptions:
         alpha =   0.0500  (one-sided)
            m1 =       10
            m2 =       14
           sd1 =  7.07106
           sd2 =  7.07106
sample size n1 =       50
            n2 =       50
         n2/n1 =     1.00
Estimated power:
         power =   0.8817
```

. sampsi 0.50 0.56, n(25) onesample onesided

```
Estimated power for one-sample comparison of proportion to hypothesized value

Test Ho: p = 0.5000, where p is the proportion in the population
Assumptions:
         alpha =   0.0500  (one-sided)
 alternative p =   0.5600
 sample size n =       25
Estimated power:
         power =   0.1463
```

. sampsi 0.50 0.68, n(25) onesample onesided

```
Estimated power for one-sample comparison of proportion to hypothesized value

Test Ho: p = 0.5000, where p is the proportion in the population
Assumptions:
         alpha =   0.0500  (one-sided)
 alternative p =   0.6800
 sample size n =       25
Estimated power:
         power =   0.5660
```

. sampsi 0.10 0.18, n1(50) onesided nocontinuity

```
Estimated power for two-sample comparison of proportions

Test Ho: p1 = p2, where p1 is the proportion in population 1
and p2 is the proportion in population 2
Assumptions:
         alpha =   0.0500  (one-sided)
            p1 =   0.1000
            p2 =   0.1800
sample size n1 =       50
            n2 =       50
         n2/n1 =     1.00
Estimated power:
         power =   0.3102
```

. sampsi 0.10 0.42, n1(50) onesided nocontinuity

```
Estimated power for two-sample comparison of proportions

Test Ho: p1 = p2, where p1 is the proportion in population 1
and p2 is the proportion in population 2
Assumptions:
         alpha =   0.0500  (one-sided)
            p1 =   0.1000
            p2 =   0.4200
sample size n1 =       50
            n2 =       50
         n2/n1 =     1.00
Estimated power:
         power =   0.9843
```

. clear
. calculation p1 from odds ratio

```
. input p0 or
       p0    or
  1. 0.1 1.8
  2. 0.1 3.0
  3. 0.1 4.6
  4. end
```

. gen p1 = or*p0/(1+p0*(or-1))

. list or p0 p1

```
     +----------------------+
     |  or   p0         p1 |
     |----------------------|
  1. | 1.8   .1   .1666667 |
  2. |   3   .1        .25 |
  3. | 4.6   .1   .3382353 |
     +----------------------+
```

. sampsi 0.10 0.1667, n1(50) onesided nocontinuity

```
Estimated power for two-sample comparison of proportions

Test Ho: p1 = p2, where p1 is the proportion in population 1
and p2 is the proportion in population 2
Assumptions:
         alpha =   0.0500  (one-sided)
            p1 =   0.1000
            p2 =   0.1667
sample size n1 =       50
            n2 =       50
         n2/n1 =     1.00
Estimated power:
         power =   0.2524
```

. sampsi 0.10 0.25, n1(50) onesided nocontinuity

```
Estimated power for two-sample comparison of proportions

Test Ho: p1 = p2, where p1 is the proportion in population 1
and p2 is the proportion in population 2
Assumptions:
         alpha =   0.0500  (one-sided)
            p1 =   0.1000
            p2 =   0.2500
sample size n1 =       50
            n2 =       50
         n2/n1 =     1.00
Estimated power:
         power =   0.6314
```

```
. sampsi 0.10 0.3382, n1(50) onesided nocontinuity

Estimated power for two-sample comparison of proportions

Test Ho: p1 = p2, where p1 is the proportion in population 1
and p2 is the proportion in population 2
Assumptions:
         alpha =   0.0500  (one-sided)
            p1 =   0.1000
            p2 =   0.3382
sample size n1 =       50
            n2 =       50
         n2/n1 =     1.00
Estimated power:
         power =   0.9013
```

```
. clear

. * nchi2(df,ncp,x)
. * normal(x,m,s)
. * normal(z)

. input x df ncp
            x df ncp
  1.    2 2 4
  2.    4 2 4
  3.    6 2 4
  4.    8 2 4
  5.   10 2 4
  6.   14 2 4
  7.   18 2 4
  8. end

. *noncentral chi-square distribution and approximations

. gen p = nchi2(df,ncp,x)
. gen m = df+ncp
. gen t1 = (x-m)/sqrt(2*(m+ncp))
. gen p1 = normal(t1)
. gen f = (2*(m+ncp))/(9*mˆ2)
. gen t2 = ((x/m)ˆ(1/3)-(1-f))/sqrt(f)
. gen p2 = normal(t2)

. list x p1 p2 p

     +-------------------------------------+
     |  x         p1         p2          p |
     |-------------------------------------|
  1. |  2   .1855467   .1621291   .1825848 |
  2. |  4   .3273604   .3972866    .396499 |
  3. |  6         .5   .5981076   .5852894 |
  4. |  8   .6726395   .7432926   .7299606 |
  5. | 10   .8144533   .8402784   .8314311 |
  6. | 14   .9631808   .9408554   .9403706 |
  7. | 18   .9963548    .978744   .9808597 |
     +-------------------------------------+
```

```
. clear
. *binary variables

. input e n
       e   n
  1. 0.0   50
  2. 0.02  50
  3. 0.04  50
  4. 0.06  50
  5. 0.08  50
  6. 0.0  100
  7. 0.02 100
  8. 0.04 100
  9. 0.06 100
 10. 0.08 100
 11. end

. gen p1 = 1-nchi2(1,n*(e/(0.1*0.9))^2,3.841)
. gen p2 = 1-nchi2(1,n*(e/(0.2*0.8))^2,3.841)
. gen p3 = 1-nchi2(1,n*(e/(0.5*0.5))^2,3.841)
. gen phi = e/(0.2*0.8)

. list

     +---------------------------------------------------+
     |   e     n          p1          p2          p3    phi |
     |---------------------------------------------------|
  1. |   0    50    .0500137    .0500137    .0500137      0 |
  2. | .02    50    .3490304    .1432012    .0874106   .125 |
  3. | .04    50    .8815658    .4239364    .2046971    .25 |
  4. | .06    50    .9970582    .7554716    .3964836   .375 |
  5. | .08    50    .9999924     .942451     .619027     .5 |
  6. |   0   100    .0500137    .0500137    .0500137      0 |
  7. | .02   100    .6034983    .2395636     .125947   .125 |
  8. | .04   100    .9935151    .7054583    .3596663    .25 |
  9. | .06   100    .9999987    .9632853    .6700934   .375 |
 10. | .08   100           1    .9988177    .8925407     .5 |
     +---------------------------------------------------+
```

```
. clear
. *first example
. *four categories -- constant versus linear trends

. input p0 p1
          p0   p1
  1.  0.25 0.10
  2.  0.25 0.20
  3.  0.25 0.30
  4.  0.25 0.40
  5. end

. gen ncp0 = 50*sum((p0-p1)^2/p0)
. gen c = invnchi2(3,0,.95)
. gen p = 1-nchi2(3,ncp0,c)

. list ncp0 p in 4

     +-----------------+
     | ncp0          p |
     |-----------------|
     |   10   .7610631 |
     +-----------------+


. *second example
. clear
. *four categories -- linear versus threshold trends

. input p0 p1
         p0   p1
  1.  0.15 0.10
  2.  0.25 0.20
  3.  0.30 0.30
  4.  0.30 0.40
  5. end

. gen ncp = 100*sum((p0-p1)^2/p0)
. gen c = invnchi2(3,0,.95)
. gen p = 1-nchi2(3,ncp,c)

. list ncp p in 4

     +----------------+
     | ncp          p |
     |----------------|
     |   6   .5180786 |
     +----------------+
```

```
. *SAMPLE SIZE CALCULATION -- chi-square statistic

. drop ncp p
. gen ncp = 125*sum((p0-p1)ˆ2/p0)
. gen p = 1-nchi2(3,ncp,c)

. list ncp c p in 4


     +---------------------------+
     | ncp          c          p |
     |---------------------------|
     | 7.5   7.814728   .6227618 |
     +---------------------------+

. drop ncp p
. gen ncp = 150*sum((p0-p1)ˆ2/p0)
. gen p = 1-nchi2(3,ncp,c)

. list ncp c p in 4

     +---------------------------+
     | ncp          c          p |
     |---------------------------|
     | 9.0   7.814728   .7112536 |
     +---------------------------+

. drop ncp p
. gen ncp = 180*sum((p0-p1)ˆ2/p0)
. gen p = 1-nchi2(3,ncp,c)

. list ncp c p in 4

     +----------------------------+
     |  ncp          c          p |
     |----------------------------|
     | 10.8   7.814728   .7958511 |
     +----------------------------+

. drop ncp p
. gen ncp = 181*sum((p0-p1)ˆ2/p0)
. gen p = 1-nchi2(3,ncp,c)

. list ncp c p in 4

     +-----------------------------+
     |   ncp          c          p |
     |-----------------------------|
     | 10.86   7.814728   .7982866 |
     +-----------------------------+
```

```
. drop ncp p
. gen ncp = 182*sum((p0-p1)ˆ2/p0)
. gen p = 1-nchi2(3,ncp,c)

. list ncp c p in 4

     +------------------------------+
     |   ncp          c           p |
     |------------------------------|
     | 10.92   7.814728    .8006985 |
     +------------------------------+
```