Help

```
apropos()
?
??
example()
```

Basic Calculations

Basic calculation works like a calculator.

```
# basic ops: + - * / ^ ( )
log(); exp(); sqrt()
log10(); abs(); choose()
```

Formula Interface

The following syntax (often with some parts omitted) is used for graphical summaries, numerical summaries, and inference procedures.

```
goal(y ~ x | z, data=...,
                groups=...)
```

For plots:

- `y`: is y-axis variable
- `x`: is x-axis variable
- `z`: conditioning variable (separate panels)
- `groups`: conditioning variable (overlaid graphs)

For other things:

'`y ~ x | z`' can usually be read '`y` is modeled by (or depends on) `x` differently for each `z`'.

See the sampler for examples.

Numerical Summaries

These functions have a formula interface to match plotting.

```
favstats()   # mosaic
tally()      # mosaic
mean()       # mosaic augmented
median()     # mosaic augmented
sd()         # mosaic augmented
var()        # mosaic augmented
diffmean()   # mosaic
```

```
quantile()   # mosaic augmented
prop()       # mosaic
perc()       # mosaic
rank()
IQR()        # mosaic augmented
min(); max() # mosaic augmented
```

Graphics (mostly lattice)

```
bwplot()
xyplot()
histogram() # mosaic augmented
densityplot()
freqpolygon() # mosaic
qqmath()
makeFun()   # mosaic
plotFun()   # mosaic
```

```
ladd()      # mosaic
dotPlot()   # mosaic
bargraph()  # mosaic
xqqmath()   # mosaic
```

```
mplot(data=HELPrct, 'scatter')
mplot(data=HELPrct, 'boxplot')
mplot(data=HELPrct, 'histogram')
```

Randomization/Simulation

```
rflip()     # mosaic
do()        # mosaic
sample()    # mosaic augmented
resample()  # with replacement
shuffle()   # mosaic
rbinom()
rnorm()     # etc, if needed
```

Distributions

```
pbinom(); pnorm();
xpnorm()    # mosaic augmented
pchisq(); pt()
qbinom(); qnorm();
qchisq(); qt()
plotDist()  # mosaic
```

Inference

```
t.test()      # mosaic augmented
binom.test()  # mosaic augmented
prop.test()   # mosaic augmented
xchisq.test() # mosaic
fisher.test()
pval()        # mosaic
model <- lm() # linear models
summary(model)
coef(model)
confint(model) # mosaic augmented
anova(model)
makeFun(model) # mosaic
resid(model); fitted(model)
mplot(model)   # mosaic
```

```
mplot(TukeyHSD(model))
model <- glm() # logistic reg.
```

Data

```
require(mosaicData)  # load package
read.file()    # mosaic
nrow(); ncol(); dim()
summary()
str()
names()
head(); tail()
with()
factor()
```

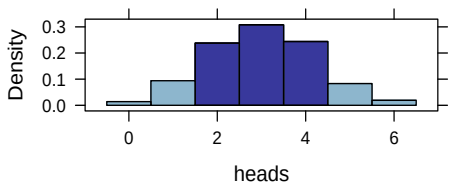
```
ntiles()      # mosaic
cut()
c()
cbind(); rbind()
colnames()
rownames()
relevel()
reorder()
```

```
rep()
seq()
sort()
rank()
```

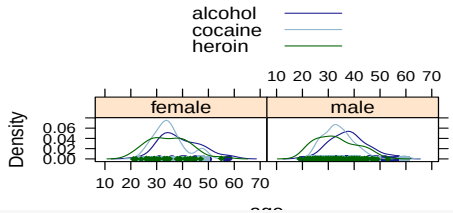
Data Transformation

```
select()      # dplyr
mutate()      # dplyr
filter()      # dplyr
arrange()     # dplyr
summarise()   # dplyr
group_by()    # dplyr
left_join()   # dplyr
inner_join()  # dplyr
merge()
```

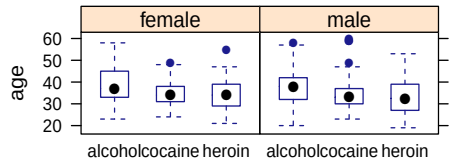
```
rflip(6)


Flipping 6 coins [ Prob(Heads) = 0.5 ] ...

T H T H H T

Number of Heads: 3 [Proportion Heads: 0.5]

do(2) * rflip(6)

Using 'parallel'.  Set seed with set.rseed().

  n heads tails      prop
1 6     4     2 0.6666667
2 6     4     2 0.6666667

coins <- do(1000) * rflip(6)

Using 'parallel'.  Set seed with set.rseed().

tally(~heads, data = coins)


   0   1   2   3   4   5   6
  14  94 238 308 244  83  19
tally(~heads, data = coins, format = "perc")


   0    1    2    3    4    5    6
 1.4  9.4 23.8 30.8 24.4  8.3  1.9

tally(~(heads >= 5 | heads <= 1), data = coins)


 TRUE FALSE
  210   790
histogram(~heads, data = coins, width = 1,
    groups = (heads >= 5 | heads <= 1))
```



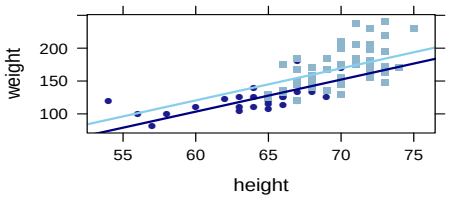

```
require(mosaicData)  # this package contains data se
tally(sex ~ substance, data = HELPrct)

        substance
sex      alcohol cocaine heroin
  female      36      41     30
  male       141     111     94

mean(age ~ sex, data = HELPrct)

  female     male
36.25234 35.46821

diffmean(age ~ sex, data = HELPrct)

  diffmean
-0.7841284

favstats(age ~ sex, data = HELPrct)

     sex min Q1 median   Q3 max
1 female  21 31     35 40.5  58
2   male  19 30     35 40.0  60
      mean       sd   n missing
1 36.25234 7.584858 107       0
2 35.46821 7.750110 346       0
densityplot(~age | sex, groups = substance,
    data = HELPrct, auto.key = TRUE)
```




```
bwplot(age ~ substance | sex, data = HELPrct)
```




```
pval(binom.test(~sex, data = HELPrct))

     p.value
1.931901e-30

confint(t.test(~age, data = HELPrct))

mean of x     lower     upper     level
 35.65342  34.94150  36.36534   0.95000
model <- lm(weight ~ height + gender,
            data=Heightweight)
wt <- makeFun(model)
wt( height=72, gender="male")

       1
179.0859

xyplot(weight ~ height, groups=gender,
       data=Heightweight)
plotFun(wt(h,gender="male") ~ h,
        add=TRUE, col="skyblue")
plotFun(wt(h,gender="female") ~ h,
        add=TRUE, col="navy")
```




```
plotDist("chisq", df = 4)
```