ffmanova.m
50-50 MANOVA for MATLAB version 2.0.

Øyvind Langsrud

The Matlab function ffmanova.m performs general linear modelling of several response variables (Y). Collinear and highly correlated response variables are handled. The X-factors can be categorical, continuous and composite continuous (several variables as one model term).

The function calculates:

- 50-50 MANOVA results.
- raw single response p-values.
- familywise adjusted and false discovery rate adjusted single response p-values by rotation testing.
- predictions, mean predictions and least squares means.
- standard deviations of those predictions.

This document illustrates the use of the function ffmanova.m. The first four sections describe four alternative ways of using the function. All the examples refer to the data (a,b,c,Y) given in Section 1.1.

1 results = ffmanova(modelFormula,stand,nSim)

1.1 Example

```
a = [0 1 0 1 0 1 0 1 0 1 0 1 0 1 0]';
b = [1 1 1 1 1 2 2 2 2 2 3 3 3 3 3]';
c = [1:15]';
Y = randn(15,20);

results =ffmanova('Y = a|b$|c#2 + c^3 - a^2',0,999);
```

```
  --- 50-50 MANOVA TestVersion --- 15 objects -- 20 responses:
  Source  DF        exVarSS nPC nBu exVarPC exVarBU    p-Value
  a        1       0.073912   1   0  0.437   0.437    0.527380
  b        2       0.140798   1   0  0.378   0.378    0.741638
  c        1       0.061413   1   0  0.453   0.453    0.554535
  a*b      2       0.091722   1   0  0.383   0.383    0.998189
  a*c      1       0.062868   1   0  0.421   0.421    0.808439
  b*c      2       0.109980   1   0  0.366   0.366    0.988167
  c^2      1       0.032898   1   0  0.458   0.458    0.944172
  c^3      1       0.086554   1   0  0.393   0.393    0.778939
  Error    3       0.268958 - STANDARDIZATION OFF ------------
```

1.2 Fields of the output structure (results)

```
 termNames: name of model terms (including "error").
   exVarSS: (Sum of SS for each response)/(Sum of total SS for each response).
        df: degrees of freedom - adjusted for other terms in model.
     df_om: degrees of freedom - adjusted for terms contained in actual term.
       nPC: number of principal components used for testing.
       nBU: number of principal components used as buffer components.
   exVarPC: variance explained by nPC components
   exVarBU: variance explained by (nPC+nBU) components
   pValues: 50-50 MANOVA p-values.
outputText: 50-50 MANOVA results as text.
      Yhat: Fitted values.
   YhatStd: Standard deviations of the fitted values.
      nSim: as input (-1 -> 0), but could have been changed interactively.
 pAdjusted: familywise adjusted p-values.
   pAdjFDR: false discovery rate adjusted p-values.
      pRaw: raw p-values.
      stat: Unvivariate t-statistics (df=1) or  F-statistics (df>1)
   newPred: Yhat's and YhatStd's according to Xnew
```

1.3 modelFormula

Main effect model – continuous variables

```
ffmanova('Y = a + b + c');
```

Main effect model. a,b and c are continuous.

Specifying categorical variables

A variable is treated as categorical if `$` is included at the end of the variable name (anywhere in a complex model formula).

```
ffmanova('Y = a + b$ + c');
```

`b` is categorical since `$`

```
b_ = {b};
ffmanova('Y = a + b_ + c');
```

`b_` is categorical since cell array

Four ways of specifying the same model

```
ffmanova('Y = a + b$ + c + a*b + a*c + b*c + c^2 + c^3');
ffmanova('Y = a|b$|c + c^2 + c^3 - a*b*c');
ffmanova('Y = a|b$|c@2 + c^2 + c^3');
ffmanova('Y = a|b$|c#2 + c^3 - a^2');
```

`@2` means interactions up to order 2
`#2` means all terms up to order 2

Eval used to interpret

Except that `=,+,-,|,@,#,*,^` are special symbols in the model formula, ffmanova uses eval to interpret the string. In particular

```
ffmanova('log(100+Y) = a + b==2 + 1./c') ;
```

is a valid expression. Note that + before = is interpreted by eval and == is always interpreted by eval.

1.4 stand

```
stand  - standardization of responses, = 0 (default) or 1
```

Standardization of responses has effect on the 50-50 MANOVA testing and the calculation of exVarSS.

1.5 nSim

```
nSim(1,#terms) - Number of rotation testing simulations.
     - nSim can be a single number -> equal nSim for all terms.
     - nSim =  0 -> pAdjusted and pAdjFDR are not calculated.
     - nSim = -1 (default) -> pRaw, pAdjusted and pAdjFDR are not calculated.
```

Example:

```
ffmanova('Y = a + b$ + c',1,999);
ffmanova('Y = a + b$ + c',1,0);
ffmanova('Y = a + b$ + c',1,[0 999 9999]);
```

2 results = ffmanova(X,Y,cova,model,xNames,stand,nSim)

Instead of specifying a model formula, ffmanova can be used with the five input arguments; X, Y, cova, model and xNames.

```
X{1,#Xvariables} - design information as cell array. Nonzero elements
      of cova indicate cells of X that are covariates. Design variables
      that uses multiple columns are allowed (e.g dummy coding).
           - Alternatively X can be an ordinary matrix where each
             column is a design variable.
Y(#observations,#responses) - matrix of response values.
        cova(1,#Xvariables) - covariate terms (see above)
  model(#terms,#Xvariables) - model matrix or order coded model or
                              text coded model (see below)
                     xNames - Names of x variables. Default: {'A' 'B' 'C'}
                     stand  - standardization of responses, = 0 (default) or 1
             nSim(1,#terms) - Number of rotation testing simulations.
```

MODEL CODING:

The model can be specified in three different ways; a model matrix, an order coded model or text coded model.

```
- order coded model:
      model{1,#Xvariables} specifys maximum order of the factors
- text coded model:
       'linear'    is equivalent to { 1 1 1 ..... 1}
       'quadratic' is equivalent to { 2 2 2 ..... 2}
       'cubic'     is equivalent to { 3 3 3 ..... 3}
- model matrix example: X = {A B C}
     model = [1 0 0; 0 1 0 ; 0 0 1; 2 0 0; 1 1 0; 1 0 1; 0 1 1; 3 0 0]
        ->   Constant + A + B + C + A^2 + A*B + A*C + B*C + A^3
     Constant term is automatically included. But create constant term
     manually ([0 0 0; ...]) to obtain constant term output.
- default model is the identity matrix -> main factor model
```

Example:

```
results = ffmanova({a b c},Y,[1 0 1],{1 2 3},{'a' 'b' 'c'},0,999);
results = ffmanova({a b c},Y,[1 0 1],'quadratic',{'a' 'b' 'c'},0,999);
```

Also see examples in the sections below.

3 model = ffmanova([],[],cova,model,xNames)

When X or Y is empty the model matrix is returned and printet (with matlab code).

Example:

```
model = ffmanova([],[],[1 0 1],{1 2 3},{'a' 'b' 'c'});
```

```
m = [ 1  0  0 ; ... %    1: a
      0  1  0 ; ... %    2: b
      0  0  1 ; ... %    3: c
      0  1  1 ; ... %    4: b*c
      0  0  2 ; ... %    5: c^2
      0  0  3 ];    %    6: c^3
```

4 [X,Y,cova,model,xNames] = ffmanova(modelFormula)

When five output arguments are specified, ffmanova returns X, Y, cova, model and xNames (see Section 2 above). Note that model is returned as a model matrix.

Examples:

```
[X,Y,cova,model,xNames] = ffmanova('Y = a|b$|c#2 + c^3 - a^2');

results = ffmanova(X,Y,cova,model,xNames,0,999);
results = ffmanova(X,Y,cova,'quadratic',xNames,0,999);
results = ffmanova(X,Y,cova,{1 2 3},xNames,0,999);
```

5 results = ffmanova(...,stand,nSim,xNew)

This section describes the use of the additional argument, xNew, which are used for making predictions, mean predictions or least-squares-means. xNew is a column-vector cell-array where each cell is a new X for prediction calculations. Predictions with corresponding standard deviations are returned as results.newPred.

```
results = ffmanova(X,Y,cova,model,xNames,stand,nSim,xNew);
```

or

```
results = ffmanova(modelFormula,stand,nSim,xNew);
```

Example:

```
Xnew1 = X;
Xnew2 = {[0 1 0 1 0 1]',[1:3 1:3]',[5 5 5 5 5 5]'};
Xnew3 = {[0 1/2 1]',[],[]};
Xnew4 = {[],[1:3]',[]};
Xnew5 = {[0 1 0 1 0 1]',[1:3 1:3]',[]};
Xnew5 = {1.5,[],3};
XNEW = {Xnew1;Xnew2;Xnew3;Xnew4;Xnew5};
results = ffmanova('Y = a|b$|c#2 + c^3 - a^2',0,0,XNEW);
```

- Predictions corresponding to Xnew1 are returned as `results.newPred{1}.Yhat`. Since, in this case, `Xnew1=X`, we have that `results.newPred{1}.Yhat = results.Yhat`.
- `results.newPred{2}.Yhat` contains predictions for level combinations of `a` and `b` when `c=5`.
- `results.newPred{3}.Yhat` contains mean predictions for three values of `a`.
- `results.newPred{4}.Yhat` contains least-squares-means for the three levels of `b`.
- `results.newPred{5}.Yhat` contains mean predictions for level combinations of `a` and `b`.
- `results.newPred{6}.Yhat` contains predictions for `a=1,5` and `c=3` that are adjusted for the effect of `b`.

Standard deviations corresponding to ...`Yhat` are returned as ...`YhatStd`. E.g: `results.newPred{5}.YhatStd`.

6 Coding of categorical variables

If *Statistics Toolbox* is available (the function grp2idx), categorical variables can be coded as a character matrix (each row representing a group name), or a cell array of strings stored as a column vector. Using such coding, the `$`-symbol in the model formula is not needed.

Example:

```
B = {'b1' 'b1' 'b1' 'b1' 'b1' 'b2' 'b2' 'b2' 'b2' 'b2' 'b3' 'b3' 'b3' 'b3' 'b3'}';
results = ffmanova('Y=a|B|c#2+c^3-a^2',0,0,{{[],{'b1' 'b2' 'b3'}',[]}});
```

```
results.newPred{1}.Yhat
```

contains least-squares-means for the three levels of `b`

7 Octave compatibility

The ffmanova function is made to be compatible with *GNU Octave, version 2.1.64* (`www.octave.org`).

- As mentioned in the previous section, categorical variables cannot be character coded when the function grp2idx (Statistics Toolbox) is not available.
- Unfortunately, the program runs much slower in Octave. Remember to set `ignore_function_time_stamp = 'all'`. Otherwise, the program can be extremely slow.
- When performing rotation testing in Matlab, a dialog box gives information about the simulations and it is possible to stop or pause the simulations. This dialog box will not appear in Octave.