Displays for Statistics 5401/8401

Lecture 19

October 19, 2005

Christopher Bingham, Instructor

612-625-1024, kb@umn.edu
372 Ford Hall

Class Web Page

http://www.stat.umn.edu/~kb/classes/5401

All tests of multivariate linear hypotheses are derived from different ways of comparing **H** and **E**. A particularly important class of tests are based on $\mathbf{E}^{-1}\mathbf{H}$. Such tests have the following form:

- Reject H_0 when $\mathbf{E}^{-1}\mathbf{H}$ is "too large" compared to $(f_h/f_e)\mathbf{I}_p$,

or equivalently

- Reject H_0: when the "multivariate F" $(f_e/f_h)\mathbf{E}^{-1}\mathbf{H}$ is too large compared to $\mathbf{I}_p$

Here's a problem:

$\mathbf{E}^{-1}\mathbf{H}$ is a p by p matrix. What number or numbers measure how large it is?

- $\det(\mathbf{E}^{-1}\mathbf{H})$ is **not** a useful number because

$$\det(\mathbf{E}^{-1}\mathbf{H}) = \det(\mathbf{E}^{-1})\det(\mathbf{H}) = \det(\mathbf{H})/\det(\mathbf{E})$$

But when $f_h < p$, $\det(\mathbf{H}) = 0$, making $\det(\mathbf{E}^{-1}\mathbf{H}) = 0$ so this is *not* helpful.

What are helpful are measures computed from the *eigenvalues* of **H** relative to **E**, that is the *relative eigenvalues.*

See the handout for a fairly complete explanation.

Vocabulary

The relative eigenvalues of **H** relative to **E** are the ordinary eigenvalues of $\mathbf{E}^{-1}\mathbf{H}$

$$\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \ldots \geq \hat{\lambda}_p \geq 0$$

You can use relative eigenvalues to express and compute several standard test statistics for multivariate linear hypothesis .

The relative eigenvectors $\hat{\mathbf{u}}_1$, $\hat{\mathbf{u}}_2$, ..., $\hat{\mathbf{u}}_p$ of **H** relative to **E** are the ordinary eigen-vectors of $\mathbf{E}^{-1}\mathbf{H}$. They satisfy

$$\mathbf{E}^{-1}\mathbf{H}\hat{\mathbf{u}}_j = \hat{\lambda}_j\hat{\mathbf{u}}_j$$

The standard normalization, which I always assume, is $\hat{\mathbf{u}}_j'\mathbf{E}\hat{\mathbf{u}}_j = 1$.

These are all measure that *are* helpful:

- **Hotelling's generalized** T^2 (trace test) based on tr $\mathbf{E}^{-1}\mathbf{H} = \sum_i\hat{\lambda}_i$
- **Roy's maximum root test** based on $\hat{\lambda}_{max} = \hat{\lambda}_1$.
- **Likelihood ratio test** (Wilks' or Rao's test) based on $1/\det(\mathbf{I}_p + \mathbf{E}^{-1}\mathbf{H}) = 1/\Pi_i(1 + \hat{\lambda}_i)$ or $\log(\det(\mathbf{I}_p + \mathbf{E}^{-1}\mathbf{H})) = \sum_i\log(1 + \hat{\lambda}_i)$
- **Pillai's trace test** based on $\mathrm{tr}\,((\mathbf{H} + \mathbf{E})^{-1}\mathbf{H}) = \mathrm{tr}(\mathbf{I} + \mathbf{E}^{-1}\mathbf{H})^{-1}\mathbf{E}^{-1}\mathbf{H} =$
$$\sum_i\hat{\lambda}_i/(1 + \hat{\lambda}_i)$$

When p = 1, there is only one $\hat{\lambda}$ so they are functions of $\hat{\lambda}_1 = SS_h/SS_e = (f_h/f_e)F$. In particular $\hat{\lambda}_1/(1 + \hat{\lambda}_1) = SS_h/(SS_h+SS_e)$.

When $p > 1$ and $f_h > 1$ they are all different.

Hotelling's generalized T^2

$$T_0^2 = f_e \sum_i \hat{\lambda}_i = f_e tr(\mathbf{E}^{-1}\mathbf{H}) = tr(\mathbf{S}^{-1}\mathbf{H})$$

where $\mathbf{S} = (1/f_e)\mathbf{E} = \hat{\mathbf{\Sigma}}$.

When H_0 is true, <u>in large samples</u>, T_0^2 is approximately χ_f^2, where $f = f_h p$.

$f = f_h p$ is the total number of *scalar* parameters (or linear combinations of *scalar* parameters) under test. There are f_h for each of p dimensions.

- 1-way MANOVA with g groups
  $$f_h = g-1,\ f = (g-1)p$$
- Testing two-way interaction in MANOVA, with
  $$f_h = (a-1)(b-1),\ f = (a-1)(b-1)p$$
  where a and b are the numbers of levels in the two factors.
- Testing $\boldsymbol{\beta}_1 = \boldsymbol{\beta}_2 = 0$, $f_h = 2$ and $f = 2p$.

You get a match to χ_f^2 if you replace f_e by $m_2 \equiv f_e - p - 1$, so the usual form of this test is

$$T = (f_e - p - 1)tr(\mathbf{E}^{-1}\mathbf{H}) = (1 - (p+1)/f_e)T_0^2.$$

Note that the $1 - (p+1)/f_e \to 1$ as $f_e \to \infty$, so with large enough f_e using m_2 makes little difference.

For an example I created some artificial data with g = 4 groups and p = 4 variables. You can download the data from

www.stat.umn.edu/~kb/classes/5401/datafiles.html

```
Cmd> data <- read("manovadata.txt","data")
data           50     5 LABELS
) Artificial one-way MANOVA data with p = 4 variables and
) g = 4 groups
) n_1 = 3, n_2 = 11, n_3 = 16, n_4 = 10
) Col. 1: Factor group with levels 1, 2, 3, 4
) Col. 2: Response Y1
) Col. 3: Response Y2
) Col. 4: Response Y3
) Col. 5: Response Y4
Read from file "TP1:Stat5401:Stat5401F05:Data:manovadata.txt"
Cmd> addmacrofile("") # get new version of mulvar.mac
Cmd> group <- factor(data[,1])
Cmd> Y <- data[,-1] # 50 by 4 matrix of response variables
Cmd> p <- ncols(Y) # number of dimensions
```

```
Cmd> manova("Y=group")
Model used is Y=group
WARNING: summaries are sequential
                  SS and SP Matrices
                 DF
CONSTANT          1
            Y1          Y2          Y3          Y4
Y1       48157       47709       47960       47828
Y2       47709       47265       47514       47383
Y3       47960       47514       47765       47633
Y4       47828       47383       47633       47501
group             3
            Y1          Y2          Y3          Y4
Y1      4.9987      5.3629      5.6136      3.7318
Y2      5.3629      7.2951      9.5705       7.042  = H
Y3      5.6136      9.5705      15.473      12.081
Y4      3.7318       7.042      12.081       9.578
ERROR1           46
            Y1          Y2          Y3          Y4
Y1      42.715      6.0666      1.1321      -8.952
Y2      6.0666      42.181     -5.4086      7.4368  = E
Y3      1.1321     -5.4086      51.907      2.5298
Y4      -8.952      7.4368      2.5298      44.303
```

Extract **H** and **E** and compute univariate F-statistics, their Bonferronized P-values and $\mathbf{E}^{-1}\mathbf{H}$.

```
Cmd> h <- matrix(SS[2,,]); fh <- DF[2]

Cmd> e <- matrix(SS[3,,]); fe <- DF[3]

Cmd> fstats <- (fe/fh)*diag(h)/diag(e); fstats
(1)      1.7944       2.6519       4.5706       3.315

Cmd> p*cumF(fstats,fh,fe,upper:T) # Bonferronized P-values
(1)     0.64589     0.23903     0.027837     0.11202
```

1 P-value < .05 so <u>reject</u> H_0 at 5% level

```
Cmd> e_inv_h <- solve(e,h); e_inv_h # E^(-1)H
            Y1          Y2          Y3          Y4
Y1     0.11565     0.13005     0.14767     0.10174
Y2     0.11045      0.1524     0.20021     0.14761
Y3     0.11311     0.19017     0.30346     0.23624
Y4    0.082606     0.14879      0.2516     0.19848
```


Now find eigenvalues. Not this way:

```
Cmd> eigvals <- eigenvals(e_inv_h) # doesn't work
ERROR: 1st argument to eigenvals() must be symmetric REAL matrix
```

You *can't* use **eigen()** or **eigenvals()** since they work only with symmetric matrices and $\mathbf{E}^{-1}\mathbf{H}$ is not symmetric. Use **releigen()** and **releigenvals()** instead.

```
Cmd> eigs <- releigen(h,e); eigs
component: values          eigs$values or eigs[1]
(1)     0.69325    0.073323   0.0034282  1.4755e-16
component: vectors         eigs$vectors or eigs[2]
             (1)         (2)         (3)         (4)
Y1    0.051256    -0.10812     0.10376   -0.017947
Y2    0.064869   -0.053316    -0.13646  -0.0099296
Y3    0.091838    0.038661  -0.0010638    0.098878
Y4    0.074768    0.057023    0.061523    -0.11043

Cmd> eigvals <- eigs$values;

Cmd> lambdamax <- eigvals[1]; lambdamax
(1)     0.69325

Cmd> trace(e_inv_h) # sum of diagonals of E^(-1)H
(1)        0.77

Cmd> sum(eigvals) # same as trace(e_inv_h)
(1)        0.77

Cmd> t0sq <- fe*trace(e_inv_h); t0sq # Hotelling's T_0^2
(1)       35.42
```

t0sq = T_0^2 tests

$$H_0\text{: } \boldsymbol{\mu}_1 = \boldsymbol{\mu}_2 = \boldsymbol{\mu}_3 = \boldsymbol{\mu}_4$$

which is the same as

$$H_0\text{: } \boldsymbol{\alpha}_1 = \boldsymbol{\alpha}_2 = \boldsymbol{\alpha}_3 = \boldsymbol{\alpha}_4 = \mathbf{0}$$

```
Cmd> cumchi(t0sq,p*fh,upper:T) # chi-squared(12) P-value
(1) 0.00040136      Very small P-value => Reject H_0
```

Compute the modified value which has the more accurate χ^2 approximation.

```
Cmd> m2 <- fe - p - 1; m2 # optimal replacement for fe = 147
(1)        41
Cmd> m2*trace(e_inv_h) # Improved Trace test statistic
(1)      31.57

Cmd> cumchi(m2*trace(e_inv_h),p*fh,upper:T)#chi-sq(12) P-val
(1)   0.0016117      Better  large  sample  P-value
```

cumtrace() in the new version of **Mulvar.mac** uses an asymptotic series to find a yet more accurate P-value.

```
Cmd> cumtrace(trace(e_inv_h),fh,fe,p,upper:T)
(1)   0.0047263
```

Note this is about 10 times larger than crudest P-value from $f_e \mathrm{tr}\mathbf{E}^{-1}\mathbf{H}$ and about 3 times larger than the "better" large sample P-value from $m_2 \mathrm{tr}\mathbf{E}^{-1}\mathbf{H}$.

Likelihood Ratio test (Wilks or Rao)

When errors are $N_p(\mathbf{0},\boldsymbol{\Sigma})$, the <u>likelihood ratio statistic</u> to test H_0 vs H_1 is

$$\lambda = \det(\mathbf{E}^{-1}(\mathbf{E} + \mathbf{H}))^{-N/2} = (\Lambda^*)^{N/2}$$

$$\Lambda^* \equiv 1/\det(\mathbf{I}_p + \mathbf{E}^{-1}\mathbf{H}) = \det(\mathbf{E})/\det(\mathbf{H}+\mathbf{E})$$

Then

$$-2 \log \lambda = N \log \det(\mathbf{I}_p + \mathbf{E}^{-1}\mathbf{H}) = -N \log\Lambda^*$$

$$= N \sum_{1\le j\le p} \log(1 + \hat{\lambda}_j)$$

The theory of LR tests says

$$-2 \log \lambda = N \log \det(\mathbf{I}_p + \mathbf{E}^{-1}\mathbf{H})$$

should be approximately χ_f^2 in large samples when H_0 is true, where $f = f_h p$.

This is **Wilks' or Rao's test**.

Note $f = f_h p$ is the same as for Hotelling's trace test.

A better multiplier of $-\log\Lambda$ than N

$$m_1 \equiv f_e - (p - f_h + 1)/2,$$

so the standard form for the likelihood ratio test statistic is

$$(f_e - (p - f_h + 1)/2)\log \det(\mathbf{I}_p + \mathbf{E}^{-1}\mathbf{H})$$
$$= (f_e - (p-f+1)/2)\,(\log(\det(\mathbf{H}+\mathbf{E})/\det(\mathbf{E}))$$
$$= (f_e - (p-f+1)/2)\,\sum_i \log(1 + \hat{\lambda}_i)$$
$$\tilde{=}\ \chi_{f_h p}^{\ 2}$$

There are other approximations described in the handout on MANOVA tests and implemented in macro **cumwilks()**..

```
Cmd> N <- nrows(Y) # sample size
Cmd> I_p <- dmat(p,1) # identity matrix
Cmd> N * log(det(I_p + e_inv_h))
(1)      30.041
Cmd> N*sum(log(1 + eigvals))
(1)      30.041
Cmd> m1 <- (fe - (p - fh + 1)/2); m1
(1)          45
Cmd> wilks <- m1*log(det(I_p + e_inv_h)); wilks
(1)      27.037
Cmd> m1*sum(log(1 + eigvals)) # from eigenvalues
(1)      27.037
Cmd> cumchi(wilks,fh*p,upper:T) # approximate P-value
(1)   0.0076322
```


cumwilks() computes a more accurate P-value based on an F-statistic computed from a power of Λ^*. See the handout for details.

```
Cmd> cumwilks(det(e)/det(h+e),fh,fe,p)
(1)   0.0077151
Cmd> cumwilks(1/prod(1+eigvals),fh,fe,p)
(1)   0.0077151
```

This is very close to the P-value 0.00763 computed from $\chi_f^{\ 2}$.

H_0 may be rejected at the 1% level of significance.

Important facts

- p by p matrix **H** has rank

$$s \equiv \min(f_h, p).$$

  $f_h = 1 \Rightarrow s = 1$.

  $p = 1 \Rightarrow s = 1$.

- There are only s non-zero relative eigenvalues of **H** relative to **E**.
- When $p > f_h$, $s = f_h$ and

$$\hat{\lambda}_{f_h+1} = \hat{\lambda}_{f_h+2} = \ldots = \hat{\lambda}_p = 0.$$

- When $p > f_h$, **H** is singular.
- A *relative* eigenvalue $\hat{\lambda}$ satisfies

$$\mathbf{H}\hat{\mathbf{u}} = \hat{\lambda}\mathbf{E}\hat{\mathbf{u}} \text{ for some vector } \hat{\mathbf{u}}.$$

  which implies

$$\mathbf{E}^{-1}\mathbf{H}\hat{\mathbf{u}} = \hat{\lambda}\hat{\mathbf{u}}$$

- $\hat{\mathbf{u}}$ is a *relative eigenvector* of **H** relative to **E** with relative eigenvalues $\hat{\lambda}$.

Define $\hat{z}_j = \hat{\mathbf{u}}_j'\mathbf{y}$ where $\hat{\mathbf{u}}_j$ = j^{th} relative eigenvector.

$\hat{z}_j$ is the j^{th} <u>MANOVA canonical variable</u> associated with hypothesis H_0.

- $\hat{z}_j = \sum_{1\leq\ell\leq p}\hat{u}_{\ell j}y_j$ is a linear combination of the response variables $y_1, y_2, \ldots y_p$ with coefficients from the relative eigenvector $\hat{\mathbf{u}}_j$.
- $\hat{\mathbf{u}}_j'\mathbf{H}\hat{\mathbf{u}}_j = \hat{\lambda}_j = SS_h(\hat{z}_j)$ = ANOVA hypothesis SS computed from $\hat{z}_j$ as if it were a new response variable.
- $\hat{\mathbf{u}}_j'\mathbf{E}\hat{\mathbf{u}}_j = 1 = SS_e(\hat{z}_j)$ = ANOVA error SS computed from $\hat{z}_j$

Example continued

```
Cmd> u_1 <- eigs$vectors[,1]; z1 <- Y %*% u_1 # Canonical var
Cmd> u_2 <- eigs$vectors[,2]; z2 <- Y %*% u_2
Cmd> u_3 <- eigs$vectors[,3]; z3 <- Y %*% u_3
Cmd> anova("z1 = group", silent:T); SS
    CONSTANT       group      ERROR1
      3809.1     0.69325           1
Cmd> anova("z2 = group", silent:T); SS
    CONSTANT       group      ERROR1
      208.54    0.073323           1
Cmd> anova("z3 = group", silent:T); SS
    CONSTANT       group      ERROR1
      39.422   0.0034282           1
Cmd> eigvals
(1)     0.69325    0.073323   0.0034282  1.4755e-16
```

Note:

- SS_h in the analyses of the $\hat{z}_j$'s match the relative eigenvalues in **eigvals**.
- SS_e are all 1

Do MANOVA computations on matrix of all 4 canonical variables:

```
Cmd> z <- Y %*% eigs$vectors; list(z) # all 4 canvar
z               REAL    50   4     (labels)
Cmd> manova("z = group",silent:T)
```

- Diagonal elements $\hat{\mathbf{u}}_j'\mathbf{H}\hat{\mathbf{u}}_j = \hat{\lambda}_j$
- Off diagonal elements $\hat{\mathbf{u}}_j'\mathbf{H}\hat{\mathbf{u}}_k = 0$, $j \neq k$

```
Cmd> round(SS[2,,],12) # H for Z to 12 decimals
                   (1)         (2)         (3)         (4)
group (1)      0.69325           0           0           0
      (2)            0    0.073323           0           0
      (3)            0           0   0.0034282           0
      (4)            0           0           0           0
```

(**round(SS[2,,],12)** suppresses small numbers like 1.242e-16 which are really zeros in disguise.)

- Diagonal elements $\hat{\mathbf{u}}_j'\mathbf{E}\hat{\mathbf{u}}_j = 1$
- Off diagonal elements $\hat{\mathbf{u}}_j'\mathbf{E}\hat{\mathbf{u}}_k = 0$, $j \neq k$

```
Cmd> round(SS[3,,],12) # E for z to 12 decimals
                   (1)         (2)         (3)         (4)
ERROR1 (1)           1           0           0           0
       (2)           0           1           0           0
       (3)           0           0           1           0
       (4)           0           0           0           1
```

The canonical correlations have 0 within group correlation.

An important fact is

- $\hat{\lambda}_1 = SS_h(\hat{z}_j)/SS_e(\hat{z}_j)$
  $= \max_{\mathbf{u}} SS_h(\mathbf{u}'\mathbf{y})/SS_e(\mathbf{u}'\mathbf{y})$
- $(f_h/f_e)\hat{\lambda}_1 = F_{max}$ = largest possible F-statistic computed from any linear combination $y_u = \mathbf{u}'\mathbf{y}$.

That is, for *any* vector **u** defining a linear combination of the variables in **y**, in a univariate ANOVA of $y_u = \mathbf{u}'\mathbf{y}$, the ANOVA F-statistic must satisfy

$$F = (SS_h(\mathbf{u}'\mathbf{y})/f_h)/(SS_e(\mathbf{u}'\mathbf{y})/f_e) \leq f_e\hat{\lambda}_1/f_h$$

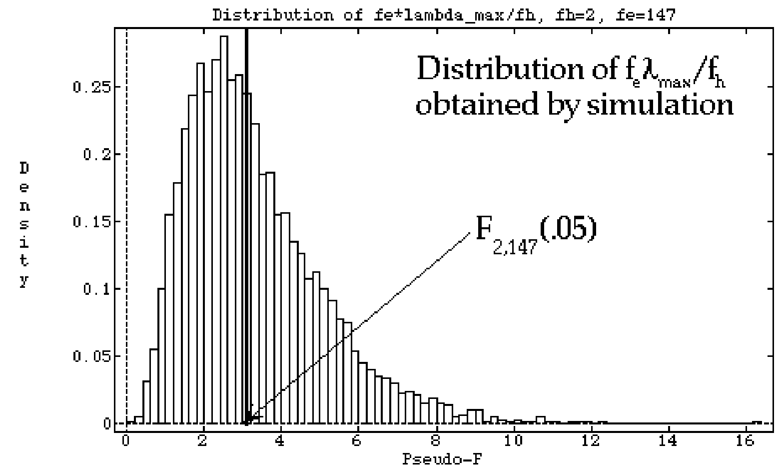
This suggests that the "pseudo-F-statistic" $f_e\hat{\lambda}_1/f_h$ or even just $\hat{\lambda}_1$ might be a good candidate for a statistic to test H_0.

Warning: When p > 1, $(f_h/f_e)\hat{\lambda}_1$ does *not* have a F-distribution.

I did a small simulation of the null distribution (distribution when H_0 is true) of $(f_h/f_e)\hat{\lambda}_1$ that shows this clearly.

`lambda_max` is a vector of $\hat{\lambda}_1$'s computed from M = 5,000 simulated samples .

```
Cmd> hist(fe*lambda_max/fh,vector(0,.2),xlab:"Pseudo-F",\
     title:"Distribution of fe*lambda_max/fh, fh=2, fe=147")
Cmd> addlines(rep(invF(.05,fh,fe,upper:T),2),vector(0,.3),\
            thickness:2)
```



$F_{2,147}(.05)$ is closer to the median of simulated values than to the upper 5% point.

Roy's maximum root test

Reject H_0 when $\hat{\lambda}_1 = \hat{\lambda}_{max}$ is "large"

I found estimates of $\hat{\lambda}_{max}(.10)$, $\hat{\lambda}_{max}(.05)$ and $\hat{\lambda}_{max}(.01)$ from the 5000 simulated values in **lambda_max**.

```
Cmd> lambda_max[round(vector(.90,.95,.99)*M)]
(1)     0.076562     0.090821      0.12154
```

Actually **Roy** proposed the canonical correlation form of the statistic

$$\hat{\theta}_1 = \hat{\theta}_{max}$$

where $\hat{\theta}_j = \hat{\lambda}_j/(1 + \hat{\lambda}_j)$, $j = 1, \ldots, p$

```
Cmd> theta_max <- lambda_max/(1 + lambda_max)
Cmd> theta_max[round(vector(.90,.95,.99)*M)] # critical vals
(1)     0.071117     0.083259      0.10837   10%,  5%,  1%
```

These last are estimated critical values for $\hat{\theta}_{max}$.

This approach by simulation is always available with the right software.