Notes for Tuesday January 22, 1998

1 Sum of Squares and Test of Significance for One-Way ANOVA model

1.1 Initial Analysis and ANOVA table: cell means model

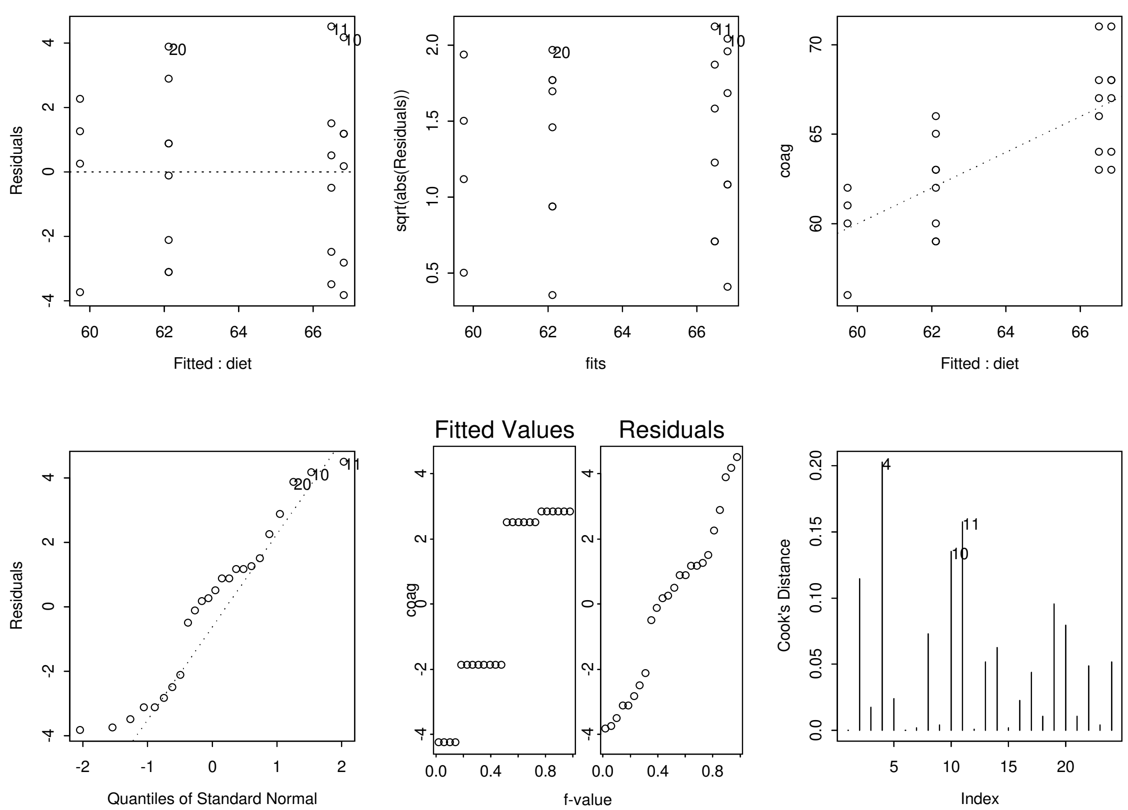
The following data is from Box, Hunter and Hunter (1978) and is also analyzed in Chapter 13 of the SPLUS Guide to Statistics. It gives blood coagulation times for each of four diets.

We enter the data more or less as in the SPLUS manual,

```
402 > coag.times _ scan()
1: 62 63 68 56
5: 60 67 66 62
9: 63 71 71 60
13: 59 64 67 61
17: 65 68 63
20: 66 68 64
23: 63
24: 59
25:
402 > diet _ factor(c(rep(LETTERS[1:4],4),rep(LETTERS[1:3],2),c("A","A")))
402 > split(coag.times,diet)  # check that the factor labels are right
$A: [1] 62 60 63 59 65 66 63 59
$B: [1] 63 67 71 64 68 68
$C: [1] 68 66 71 67 63 64
$D: [1] 56 62 60 61
402 > sapply(split(coag.times,diet),mean)
      A        B    C     D
 62.125 66.83333 66.5 59.75
402 > coag _ data.frame(coag=coag.times,diet=diet)
```

Now we fit the model, look at some diagnostic plots, and consider the analysis of variance table. More details on the SPLUS parts of the problem can be found in Chapter 13 of the SPLUS Guide to Statistics.

```
402 > coag.aov _ aov(coag ~ diet, data=coag)
402 > par(mfrow=c(2,3))
402 > plot(coag.aov)
```



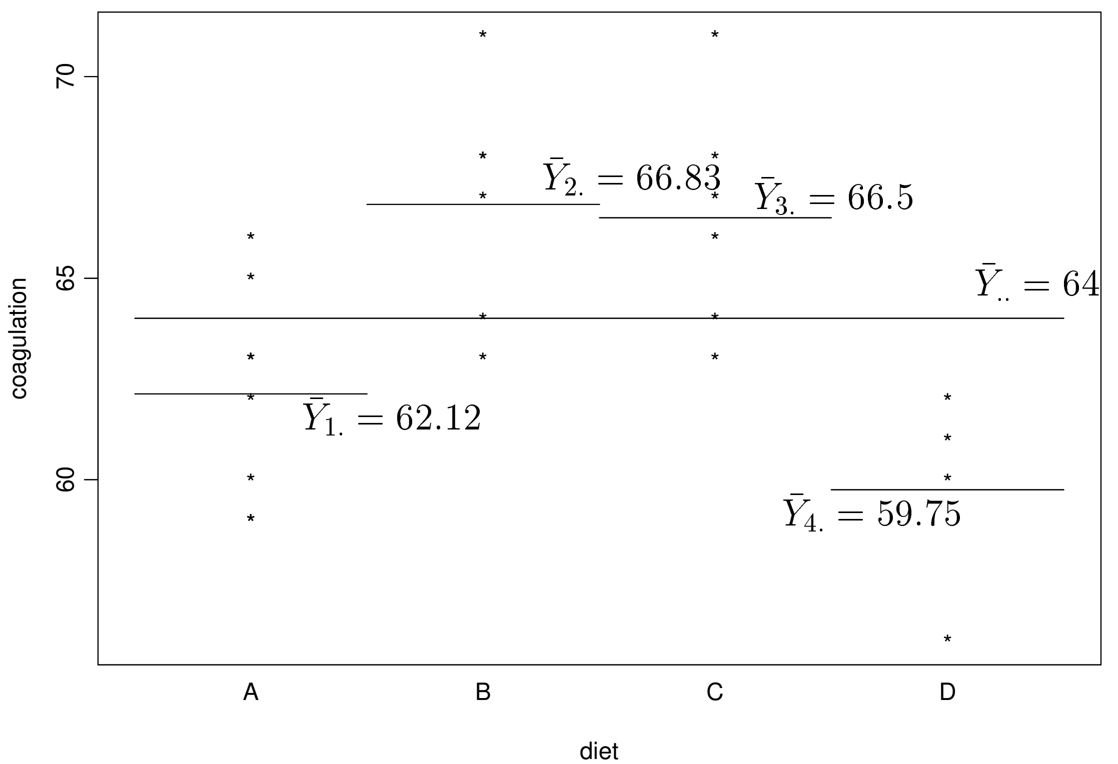

```
402 > model.tables(coag.aov,type="means")
Refitting model to allow projection
Tables of means
Grand mean

 64

 diet
        A     B     C     D
    62.12 66.83 66.5 59.75
```

```
rep  8.00  6.00  6.0  4.00
```

The cell means $Y_{i.}$ and the grand mean $Y_{..}$ are illustrated in the figure below; the cell means $\bar{Y}_{i.}$ are the estimates of the μ_i's in the *cell means model*

$$Y_{ij} = \mu_i + \epsilon_{ij}, \quad i = 1, \ldots, \#cells, \quad J = 1, \ldots, n_i$$

where k is the number of cells, and there are n_i observations in the i^{th} cell. As distributed $N(0, \sigma^2)$



Now recall that we can write the deviation of a single observation from the grand mean as

$$(Y_{ij} - Y_{..}) = (Y_{ij} - Y_{i.}) + (Y_{i.} - Y_{..})$$

If we square and sum these terms, the magic of orthogonal sums of squares tells us

$$\sum_i \sum_j (Y_{ij} - Y_{..})^2 = \sum_i \sum_j (Y_{ij} - Y_{i.})^2 + \sum_i n_i \cdot (Y_{i.} - Y_{..})^2$$

with degrees of freedom $n - 1$, $n - k$ and $k - 1$ respectively. This gives rise to the ANOVA table

Source	*SS*	*df*	*MS*
Factor	SSTR=$\sum_i n_i \cdot (Y_{i.} - Y_{..})^2$	$k-1$	MSTR=$SSTR/(k-1)$
Error	SSE=$\sum_i \sum_j (Y_{ij} - Y_{i.})^2$	$n-k$	MSE=$SSE/(n-k)$
Total	SST=$\sum_i \sum_j (Y_{ij} - Y_{..})^2$	$n-1$	MST=$SST/(n-1)$

Here is SPLUS's ANOVA table for this ANOVA model.

```
402 > anova(coag.aov)     # some output omitted below
Terms added sequentially (first to last)
          Df Sum of Sq  Mean Sq  F Value       Pr(F)
     diet  3  186.0417 62.01389 8.055931 0.001028171
Residuals 20  153.9583  7.69792
402 >  186.0417 /( 186.0417 +153.9583)  # R^2
[1] 0.5471815
```

The F statistic for testing whether the factor explains the variation in Y is

$$F^* = \frac{MSTR}{MSE} = \frac{(\text{between SS})/(k-1)}{(\text{within SS})/(n-k)}$$

Under the null hypothesis

$$H_0 : \mu_1 = \mu_2 = \cdots = \mu_k$$

F^* is distributed as $F(k-1, n-k)$. Large values of F^* argue in favor of

$$H_a : \text{not all } \mu_i \text{ are equal}$$

1.2 Contrasts

It is useful to get standard errors with the estimates of the coefficients of the ANOVA model. For the *cell means model* this is easy. The model says

$$Y_{ij} \sim N(\mu_i, \sigma^2), \text{ independent } \forall i, j$$

and hence

$$\bar{Y}_{i,} \sim N(\mu_i, \sigma^2/n_i), \text{ independent } \forall i$$

The independence makes it easy to compare pairs of cell means also:

$$\bar{Y}_{1.} - \bar{Y}_{2.} \sim N(\mu_1 - \mu_2, \sigma^2 \cdot [1/n_1 + 1/n_2])$$

and similarly with interesting linear combinations:

$$\sum_i w_i \bar{Y}_{i.} \sim N\left(\sum_i w_i \mu_i,\ \sigma^2 \sum_i w_i^2/n_i\right) \tag{1}$$

When we replace σ^2 with the usual regression estimator MSE, of course, all the $N(0,1)$'s get replaced by t_{n-k}'s.

In the blood coagulation data it is interesting to think about whether $\mu_2 = \mu_3$, and perhaps whether $\mu_1 = \mu_4$. $(1-\alpha) \times 100\%$ confidence intervals for the differences in cell means are:

$$\begin{array}{rrl} \mu_2 - \mu_3 : & & 66.83 - 66.5 \pm t_{24-4,1-\alpha/2} \cdot \sqrt{7.70(1/6 + 1/6)} \\ & \text{or} & 0.33 \pm t_{20,1-\alpha/2} \cdot 1.60 \\ \mu_1 - \mu_3 : & & 62.12 - 59.75 \pm t_{24-4,1-\alpha/2} \cdot \sqrt{7.70(1/8 + 1/4)} \\ & \text{or} & 2.42 \pm t_{20,1-\alpha/2} \cdot 1.70 \end{array}$$

Both intervals contain zero; inverting these intervals to make hypothesis tests we would not have any evidence to reject the null hypotheses of equality.

There are two things to note about this example: one is nomenclature and the other is a warning.

- **Contrasts:** The parameters (and their estimates) $\mu_i - \mu_j$ are called *contrasts*: they contrast the values of two different cell means.

  As ANOVA and linear models developed, the meaning of the word *contrast* has also expanded: nowadays, a *contrast* is **any** linear combination of the cell means $L = \sum_i w_i \mu_i$ where the coefficients sum to zero, $\sum_i w_i = 0$.

  Equation (1) lets us estimate and test hypotheses about any contrast L.

  Sets of contrasts are sometimes used to reparameterize an ANOVA model to make it easier to work with numerically (that is the point of the "Helmert contrasts" that you may have read about in the SPLUS Guide to Statistics). In cases like this, the parameters or "effects" of the reparametrized model are also contrasts. This is confusing, but that's the way the nomenclature works.

- The analysis we illustrated with the contrasts $\mu_2 - \mu_3$ and $\mu_1 - \mu_4$ was flawed in two ways:

  1. **Capitalizing on chance:** We looked at the data (sample cell means) to decide which contrasts to consider. This is bad news, since the distribution theory that leads to the $N(0,1)$ and t_{n-k} distributions above depend on *not* having looked at the data yet for their validity. (Nevertheless this is a common *exploratory* device. Any results obtained this way would need to be confirmed in an independent replication of the experiment, however.)
  2. **Multiple Comparisons:** We didn't make any adjustment for the fact that we calculated two confidence intervals (or performed two hypothesis tests) instead of one. This is the *multiple inference* or *multiple comparisons* problem. Just because each interval individually has confidence 95% (say), it does not mean that both intervals together have this confidence: Indeed, letting

     $$\begin{aligned} A &= \{\mu_2 - \mu_3 \text{ is in its CI}\};\ \textit{and} \\ B &= \{\mu_1 - \mu_4 \text{ is in its CI}\}, \end{aligned}$$

     we see from

     $$P[A \cap B] = P[A] + P[B] - P[A \cup B]$$

     that it would be quite unusual if A, B *and* $A \cap B$ all had probability 0.95 (say).

1.3 Multiple Comparisons

If we are contemplating making m inferences $A_i, \ldots, A_m$, then the argument above extends to show that

$$P[\cap_i A_i] = \sum_i P[A_i] - \sum_i \sum_j P[A_i \cup A_j] + \sum_i \sum_j \sum_k P[A_i \cap A_j \cap A_k] \pm \cdots \quad (2)$$

The way to adjust inferences for multiple comparisons is to consider all of the inferences one is likely to make, and then use equation (2)—or exploit the structure of the regression/ANOVA model to get around (2)—to compute what the confidence level for each A_i should be so that $P[\cap_i A_i]$ is at least $(1-\alpha)\times 100\%$.

For linear models in general, and ANOVA models in particular, there are three common ways of doing this:

- In the *Bonferroni* method, the A_i's are whatever particular CI's that you are interested in. It is easy to see, by induction from (2) for two events, that
$$P[\{\cap_{i=1}^m A_i\}^c] \equiv P[\cup_{i=1}^m A_i^c] \leq \sum_{i=1}^m P[A_i^c]$$
and so
$$P[\cap_i A_i] = 1 - P[\cup_i A_i^c] \geq 1 - \sum_i (1 - P[A_i])$$
This suggests that if we want $P[\cap_i A_i] > 1-\alpha$, we should take $P[A_i] = 1-\alpha/m$.

- In *Scheffé's* method, the A_i's are CI's for all possible contrasts $L = \sum_i w_i\mu_i$ with $\sum_i w_i = 0$. The essential idea is that for the contrast L, (1) tells us that a confidence interval will be of the form
$$\sum_i w_i \bar{Y}_{i.} \pm \text{cutoff}(\alpha)\cdot\sqrt{MSE \cdot \sum_i w_i^2/n_i}$$
For a single CI, we would use the upper $\alpha/2$ tail cutoff for a t_{n-k} distribution,
$$\text{cutoff}(\alpha) = t_{n-k,1-\alpha/2}$$
Scheffé's remarkable result is that if you replace this with the square root of a scaled upper-α tail cutoff for an F distribution,
$$\text{cutoff}(\alpha) = \sqrt{(k-1)F(1-\alpha,k-1,n-k)}$$
all resulting intervals for *every possible* contrast L will have confidence $(1-\alpha)\times 100\%$.

- In the *Tukey*, or *Tukey-Kramer* method, the A_i's are CI's for all pairwise comparisons $\mu_i - \mu_{i'}$. The idea is essentially the same as Scheffé, except that
  - It only applies to the family of contrasts $L = \mu_i - \mu_j$; and

- the cutoff we use is a scaled upper α tail cutoff of the "studentized range" distribution q,

$$\text{cutoff}(\alpha) = \frac{1}{\sqrt{2}} q(1-\alpha, k, n-k)$$

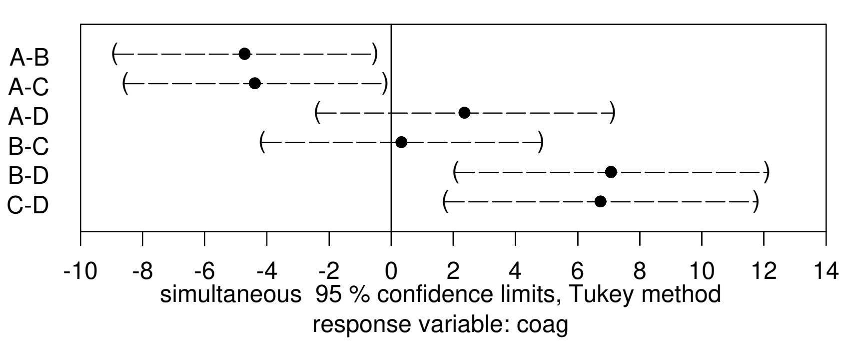
(The distribution q is the distribution of the ratio w/s where w is the range $Y_{max} - Y_{min}$ of a random sample of size k from $N(\mu, \sigma^2)$, and s^2 is a $n-k$ degrees of freedom estimate of σ^2, independent of w).

Naturally one wants to choose the method that leads to the narrowest intervals, but also has a defensible confidence statement. The following guidlines more or less follow Neter, Wasserman and Kutner (1990, p. 589).

1. The Bonferroni method usually gives the best (narrowest) intervals, if only a few comparisons or contrasts will be looked at.
2. Scheffé's method is better than Bonferroni only when the number of confidence intervals is much larger than the number of factor levels.
3. For looking at all pairwise comparisons $\mu_i - \mu_{i'}$, Tukey's method is better (leads to narrower intervals) than Bonferroni.
4. Sometimes, as in our example of contrasts above, you want to snoop through a lot of CI's to see what comparisons or contrasts look "interesting".
   - Clearly Scheffé's and Tukey's methods give the best protection against this sort of "data snooping", since you do not have to specify in advance how many or which contrasts you will be looking at.
   - The Bonferroni method is not well suited to data snooping, since you have to specify in advance how many CI's m you will calculate to make the Bonferroni correction α/m.
5. In any given problem, we are allowed by the theory to try all three methods (if appropriate) and select the method that gives the narrowest confidence intervals. This is valid since the confidence intervals generated by each method for a particular contrast are nested within one another (so the confidence of the narrowest interval is a lower bound on the confidence of wider intervals).

In SPLUS there is a special function `multicomp()` that handles the details of multiple comparisons. Here are some examples of its use with the `coag` dataset.

```
402 > coag.mca _ multicomp(coag.aov,focus="diet")
402 > coag.mca

95 % simultaneous confidence intervals for specified
linear combinations, by the Tukey method

critical point: 2.7987
response variable: coag

intervals excluding 0 are flagged by '****'

    Estimate Std.Error Lower Bound Upper Bound
A-B   -4.710      1.50       -8.90      -0.515 ****
A-C   -4.380      1.50       -8.57      -0.181 ****
A-D    2.370      1.70       -2.38       7.130
B-C    0.333      1.60       -4.15       4.820
B-D    7.080      1.79        2.07      12.100 ****
C-D    6.750      1.79        1.74      11.800 ****

402 > plot(coag.mca)
```

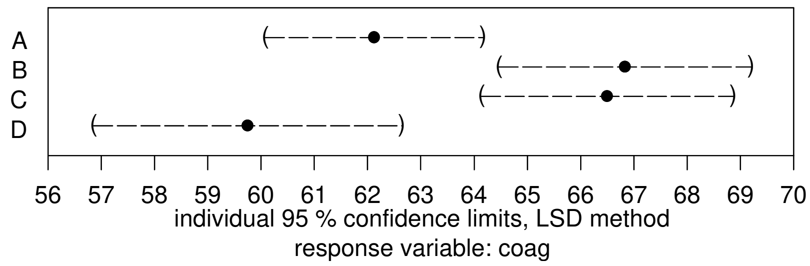


The `multicomp()` procedure does exactly what is indicated in item #5 above: it tries several methods (including the three mentioned above) of doing multiple comparisons, and then reports to us the best (narrowest intervals) method. You can force it to try a few more computer-intensive methods by saying `method="best"`, or you can force it to use a particular method by specifying the method. Some method choices include:

- `method="bon"` selects the Bonferroni method.
- `method="scheffe"` selects Scheffé's method.
- `method="tukey"` selects the Tukey-Kramer method.
- `method="best.fast"` selects among several fast methods (including Bonferroni, Scheffé' and Tukey-Kramer); this is the default.
- `method="best"` also tries some computationally intensive methods in addition to the fast methods, and selects the best (narrowest intervals) result.

Finally, `multicomp()` can be used to generate unadjusted CI's for the cell means as well:

```
402 > multicomp(coag.aov,focus="diet",comparisons="none",
+ method="lsd",error.type="cwe",plot=T)

95 % non-simultaneous confidence intervals for specified
linear combinations, by the Fisher LSD method

critical point: 2.086
response variable: coag

intervals excluding 0 are flagged by '****'

  Estimate Std.Error Lower Bound Upper Bound
A     62.1     0.981        60.1        64.2 ****
B     66.8     1.130        64.5        69.2 ****
C     66.5     1.130        64.1        68.9 ****
D     59.8     1.390        56.9        62.6 ****
```



Similarly, you could get `multicomp()` to give the "uncorrected" confidence intervals we calculated above when contrasts were introduced, by dropping the "comparisons="none"" parameter.

```
402 > multicomp(coag.aov,focus="diet",method="lsd",error.type="cwe",plot=T)
```

`method="lsd"` stands for Fisher's method of least significant differences, which is precisely the unadjusted t intervals we first calculated. Since it has R. A. Fisher's name attached to it, lots of nonstatisticians use it (try searching for "`+Fisher +"least significant difference"`" in Alta-Vista!); however this method does *not* protect against the degradation of confidence levels in multiple CI's, and it is not much better than the 68%-95%-99% eyeball rule from Statistics 36-201.

1.4 Grand Mean plus Effects model

Last time we also talked about the "grand mean plus effects" parameterization,

$$Y_{ij} = \mu + \alpha_i + \epsilon_{ij}, \quad i = 1, \ldots, k \quad j = 1, \ldots, n_i$$

The relationship of this model to the cell means model

$$Y_{ij} = \mu_i + \epsilon_{ij}$$

is that μ is supposed to be an average of the $\mu's$, and that $\alpha_i = \mu_i - \mu$.

The grand mean plus effects model has $k+1$ parameters, μ, α_1, ..., α_k and only k cell means to estimate, so some sort of linear constraint is needed to make the model work. There are several different ways to impose a "sum to zero" constraint, each corresponding to a different "grand mean":

- **Unweighted Mean.** If we take $\mu = \sum_i \mu_i / k \equiv \mu_.$, it follows that

$$\sum_i \alpha_i = \sum_i (\mu_i - \mu) = \sum_i \mu_i - \mu_. = 0$$

- **Sample Size Weighted Mean.** If we take $\mu = \sum_i n_i \mu_i / n$, then it follows that

$$\sum_i n_i \alpha_i = \sum_i (n_i \mu_i - n_i \mu) = \sum_i n_i \mu_i - n\mu = \sum_i n_i \mu_i - \sum_i n_i \mu_i = 0$$

- **A-priori Weighted Mean.** If there are some external weights w_i such that $\sum_i w_i = 1$ and we define $\mu = \sum_i w_i \mu_i$, it similarly follows that

$$\sum_i w_i \alpha_i = 0.$$

Each method is appropriate for some problems. Probably the two most common are the first two. Note that under the null hypothesis that all the cell means are the same,

$$\sum_i w_i \bar{Y}_{i.}$$

is an unbiased estimator of the grand mean $\mu^* = E[Y_{ij}]$. It can be shown that if $w_i = n_i / n$ (sample size weights) this estimator has the least possible variance (this is a nice exercise in multivariate differential calculus).

It's easy to see that the `model.tables()` function gives you the sample-size weighted effects when you specify `type="effects"`:

```
402 > model.tables(coag.aov,type="effects")
Refitting model to allow projection
Tables of effects

 diet
        A     B   C     D
    -1.875 2.833 2.5 -4.25
rep  8.000 6.000 6.0  4.00
402 > sum(scan())
```

```
1:   -1.875 2.833 2.5 -4.25
5:
[1] -0.792
402 > sum(scan()*scan())
1:   -1.875 2.833 2.5 -4.25
5:
1:  8.000 6.000 6.0  4.00
5:
[1] -0.002
```

One can fit the unweighted effects model directly by telling SPLUS what kind of contrasts to use:

```
402 > coag.sum.aov _ aov(coag ~ diet,data=coag,contrasts=list(diet=contr.sum))
402 > coefficients(coag.sum.aov)
 (Intercept)     diet1   diet2    diet3
    63.80208 -1.677083 3.03125 2.697917
402 > co _ .Last.value
402 > c(co[2:4],-sum(co[2:4]))
     diet1   diet2    diet3
 -1.677083 3.03125 2.697917 -4.052083
402 > co[1] + c(co[2:4],-sum(co[2:4]))
  diet1    diet2 diet3
 62.125 66.83333  66.5 59.75
402 > model.tables(coag.aov,type="means")$tables$diet
Refitting model to allow projection
      A        B    C     D
 62.125 66.83333 66.5 59.75
```

It is also possible to use `summary.lm` to look at the individual coefficients and their standard errors.

```
402 > summary.lm(coag.sum.aov)

Call: aov(formula = coag ~ diet, data = coag, contrasts = list(diet = contr.sum))
Residuals:
    Min     1Q Median    3Q Max
```

```
 -3.833 -2.583  0.375 1.312 4.5

Coefficients:
                Value Std. Error   t value  Pr(>|t|)
(Intercept)   63.8021    0.5838   109.2923    0.0000
      diet1   -1.6771    0.9066    -1.8499    0.0792
      diet2    3.0313    0.9911     3.0585    0.0062
      diet3    2.6979    0.9911     2.7221    0.0131

Residual standard error: 2.775 on 20 degrees of freedom
Multiple R-Squared: 0.5472
F-statistic: 8.056 on 3 and 20 degrees of freedom, the p-value is 0.001028

Correlation of Coefficients:
      (Intercept)   diet1   diet2
diet1 -0.1894
diet2 -0.0346     -0.2454
diet3 -0.0346     -0.2454 -0.3061
402 > #
402 > # now use the information in the summary to construct a
402 > # point estimate and se for the effect of the fourth diet ("D")
402 > #
402 > diet4 _ c(0,-1,-1,-1)
402 > diet4 %*% coefficients(coag.sum.aov)
          [,1]
[1,] -4.052083
402 > diet4 %*% summary.lm(coag.sum.aov)$cov.unscaled %*% diet4
          [,1]
[1,] 0.1692708
402 > sqrt(.Last.value)
          [,1]
[1,] 0.4114254
```

One can do something similar with the cell means model:

```
402 > coag.tx.aov _ aov(coag ~ diet,data=coag,contrasts=list(diet=contr.treatment))
402 > summary.lm(coag.tx.aov)

Call: aov(formula = coag ~ diet, data = coag, contrasts = list(diet = contr.treatmen
t))
Residuals:
    Min     1Q Median    3Q Max
 -3.833 -2.583  0.375 1.313 4.5

Coefficients:
               Value Std. Error  t value Pr(>|t|)
(Intercept)  62.1250   0.9809    63.3322   0.0000
      dietB   4.7083   1.4984     3.1422   0.0051
      dietC   4.3750   1.4984     2.9198   0.0085
      dietD  -2.3750   1.6990    -1.3979   0.1775

Residual standard error: 2.775 on 20 degrees of freedom
Multiple R-Squared: 0.5472
F-statistic: 8.056 on 3 and 20 degrees of freedom, the p-value is 0.001028

Correlation of Coefficients:
      (Intercept)   dietB   dietC
dietB -0.6547
dietC -0.6547      0.4286
dietD -0.5774      0.3780  0.3780
```

Now SPLUS has parametrized the model so that the intercept is the cell mean for Diet A, and the other parameters are the offsets from Diet A to Diets B, C, and D.

SPLUS has several kinds of built-in "contrast" options, that are used to reparametrize an ANOVA model. They are:

- `contr.treatment` produces coefficients that are are just the differences between level 1 and levels 2 through k of the factor.
- `contr.sum` produces coefficients satisfying the constraint that their sum is zero (the traditional analysis of variance parametrization).

- `contr.helmert` contrasts are equivalent to taking differences between level $i+1$ and the average of levels $[1:i]$, in the unweighted effects model. So for example if μ is the grand mean and α_i are the unweighted effects for level i (so all the α's sum to zero), the parameters β_i for the Helmert contrasts are related to the effects α_i in the unweighted effects model by

$$\begin{aligned} \mu &= (Intercept) \\ \beta_1 &= -\alpha_1 + \alpha_2 \\ \beta_2 &= -(\alpha_1 + \alpha_2) + 2\alpha_3 \\ \beta_3 &= -(\alpha_1 + \alpha_2 + \alpha_3) + 3\alpha_4 \\ \text{etc.} & \end{aligned}$$

  [to see this, compare `model.matrix(coag.sum.aov)[1:4,]` with `model.matrix(coag.aov)[1:4,]` above!]

- `contr.poly` creates orthogonal polynomials of degree 1, 2, etc., and is most useful for quantitative, rather than qualitative, factors.

In `aov()`, `contr.helmert` is the "default" for factors, and `contr.poly` is the default for continuous regressors. Both of these parameterizations produce a model matrix X with orthogonal columns, so that the estimated parameters are uncorrelated, making it is easy to convert from these into some other parametrization.