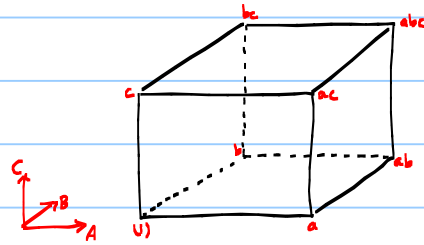
handout 16-1

For a 2^3 factorial design, write ABC and BCA in Yates notation (ie. (1), a, b, ab, c, ...), and show that ABC = BCA. Hint ABC is defined as $\frac{1}{2}[(\text{Avg } AB \mid C=+) - (\text{Avg } AB \mid C=-)]$.



$$ABC = \frac{1}{2}\left[(\text{Avg } AB \mid C=+) - (\text{Avg } AB \mid C=-)\right]$$

$$= \frac{1}{2}\left[\frac{(abc+c)-(bc+ac)}{2n} - \frac{(ab+(1))-(b+a)}{2n}\right]$$

$$= \frac{1}{4n}\left[(abc+c)-(bc+ac) - (ab+(1)) + (b+a)\right]$$

$$BCA = \frac{1}{2}\left[(\text{Avg } BC \mid A=+) - (\text{Avg } BC \mid A=-)\right]$$

$$= \frac{1}{2}\left[\frac{(abc+a)-(ac+ab)}{2n} - \frac{(bc+(1))-(b+c)}{2n}\right]$$

$$= \frac{1}{4n}\left[(abc+a) - (ac+ab) - (bc+(1)) + (b+c)\right]$$

$$\boxed{ABC = BCA}$$

hw-lect 16-2

We have seen that for 2^2, The ANOVA decomposition for The model $Y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \epsilon_{ijk}$ gives expressions for SS_A (and SS_B, SS_{AB}) which is proportional to $(\text{Contrast}_A)^2$

Show that for 2^3, SS_A as defined by $SS_A = \frac{1}{bcn}\sum_{i=1}^{a} Y_{i\cdots}^2 - \frac{1}{abcn} Y_{\cdots\cdot}^2$ (See eq. 5.28, p. 206) is also proportional to the square of a contrast. What is that contrast (Hint: look at +/- Table)?

$$SS_A = \frac{1}{bcn}\sum_i^{a} Y_{i\cdots}^2 - \frac{1}{abcn} Y_{\cdots\cdot}^2 \qquad a, b, c, = 1, 2$$

$$= \frac{1}{bcn}\left[Y_{1\cdots}^2 + Y_{2\cdots}^2 - \frac{1}{2} Y_{\cdots\cdot}^2 \right] = \frac{1}{bcn}\left[Y_{1\cdots}^2 + Y_{2\cdots}^2 - \frac{1}{2}(Y_{1\cdots} + Y_{2\cdots})^2 \right]$$

$$= \frac{1}{bcn}\left[Y_{1\cdots}^2 + Y_{2\cdots}^2 - \frac{1}{2} Y_{1\cdots}^2 - \frac{1}{2} Y_{2\cdots}^2 - Y_{1\cdots} Y_{2\cdots} \right]$$

$$= \frac{1}{bcn}\left[\frac{1}{2} Y_{1\cdots}^2 + \frac{1}{2} Y_{2\cdots}^2 - Y_{1\cdots} Y_{2\cdots} \right] = \frac{1}{bcn}\frac{1}{2}\left(Y_{2\cdots} - Y_{1\cdots} \right)^2$$

$$= \frac{1}{8n}\left[(a + ab + ac + abc) - ((1) + b + c + bc) \right]^2$$

$$= \frac{1}{8n}\left[-(1) + a - b + ab - c + ac - bc + abc \right]^2$$

$$= \frac{1}{2^3 n}(\text{Contrast}_A)^2$$

Based on The 2^3 +/- Table, The quantity inside The bracket is The contrast for A The effect.

hw-lect16-3 For a 2^4 design, write the ABC effect in Yates notation.

(1) start with the Design Matrix, with A changing fastest, ...

(2) multiply the A, B, C, D columns.

(3) identify the Yates sums.

	A	B	C	D	ABCD
(1)	−	−	−	−	+
a	+	−	−	−	−
b	−	+	−	−	+
ab	+	+	−	−	+
c	−	−	+	−	−
ac	+	−	+	−	+
bc	−	+	+	−	+
abc	+	+	+	−	−
d	−	−	−	+	−
ad	+	−	−	+	+
bd	−	+	−	+	+
abd	+	+	−	+	−
cd	−	−	+	+	+
acd	+	−	+	+	−
bcd	−	+	+	+	−
abcd	+	+	+	+	+

(4)
$$ABCD = \frac{1}{2^4 n}\Big[(1) - a + b + ab - c + ac + bc - abc - d + ad + bd - abd + cd - acd - bcd + abcd\Big]$$

There are $(2^4 n)$ terms in the [ ].

If I haven't made a mistake, this should

(ABC | high D) − (ABC | low D).

hw_lect16-4

# We have developed simple formulas that allow us to compute effects and the corresponding SS values (and, therefore F) from the contrast of each effect:
effect = (1/(n*2^(k-1))) * contrast
SS = (1/(n*2^k)) * contrast^2
We have also developed a formula for doing t-tests on each effect, using t = (effect - 0) / sqrt(MSE/ (n*2^(k-2))) with df = N - p, where N is the total number of runs (i.e. n*2^k) and p is the no. of params/effects in the model from which MSE is computed, i.e., 2^k.

This way we can do tests "by hand," but to reproduce those results in R is a bit tricky, because the estimates of the effects in lm() depend on the specification of the "contrast" argument in lm(). The contrast in our formulas is, e.g., mean(y[A==high]) - mean(y[A==low]). In lm(), the contrast specification "contr.helmert" leads to parameter estimates which are the effects divided by 2. In this hw, you'll confirm all that, and more. For the data in problem 6.15,
a) Develop a full model, and generate the ANOVA table. Store the MSE value forlater use. Which parameters/effects are significant?

```
   rm(list=ls(all=TRUE))
   library(AlgDesign)          # for gen.factorial()

   k = 4                        # number of factors.
   n = 2                        # number of replicates.

   design = gen.factorial(c(rep(2,k),n),varNames=c("A","B","C","D","R"), factors="all")
   attach(design)
   y = c(
      7.037, 14.707, 11.635, 17.273, 10.403, 4.368, 9.360, 13.440, 8.561, 16.867, 13.876, 19.824, 11.846, 6.125, 11.190, 15.653,   #
1st replicate
      6.376, 15.219, 12.089, 17.815, 10.151, 4.098, 9.253, 12.923, 8.951, 17.052, 13.658, 19.639, 12.337, 5.904, 10.935, 15.053)   #
2nd replicate

   lm.1 = lm(y~A*B*C*D)
   summary.aov(lm.1)

#             Df Sum Sq Mean Sq  F value   Pr(>F)
# A            1  72.91   72.91  898.339 1.74e-15 ***
# B            1 126.46  126.46 1558.172  < 2e-16 ***
# C            1 103.46  103.46 1274.822  < 2e-16 ***
# D            1  30.66   30.66  377.802 1.49e-12 ***
# A:B          1  29.93   29.93  368.739 1.79e-12 ***
# A:C          1 128.50  128.50 1583.256  < 2e-16 ***
# B:C          1   0.07    0.07    0.908    0.355
# A:D          1   0.05    0.05    0.577    0.459
# B:D          1   0.02    0.02    0.220    0.645
# C:D          1   0.05    0.05    0.583    0.456
# A:B:C        1  78.75   78.75  970.325 9.49e-16 ***
# A:B:D        1   0.08    0.08    0.947    0.345
# A:C:D        1   0.00    0.00    0.036    0.852
# B:C:D        1   0.01    0.01    0.125    0.728
# A:B:C:D      1   0.00    0.00    0.020    0.890
# Residuals   16   1.30    0.08

  MSE = summary.aov(lm.1)[[1]][16,3]                  # 0.08115962
# The significant factor effects are A, B, C, D, AB, AC, and ABC.
```

b) Using the following pseudo-code, develop a full model and generate estimates of the parameters, and the corresponding results from t-tests. Which parameters are significant? As mentioned above, and from example 6.2 (in class), these parameter estimates are actually the effects divided by 2. Report the values of all the effects.

2

```
#   contr = as.character("contr.helmert")
#   lm( formula , contrasts = list(A=contr,B=contr,C=contr,D=contr)) # for 4 factors

  contr = as.character("contr.helmert")
  lm.2 = lm(y~A*B*C*D, contrasts = list(A=contr,B=contr,C=contr,D=contr))
  summary.lm(lm.2)
#              Estimate Std. Error t value Pr(>|t|)
# (Intercept) 11.988062   0.050361 238.042  < 2e-16 ***
# A1           1.509438   0.050361  29.972 1.74e-15 ***
# B1           1.987938   0.050361  39.474  < 2e-16 ***
# C1          -1.798125   0.050361 -35.705  < 2e-16 ***
# D1           0.978875   0.050361  19.437 1.49e-12 ***
# A1:B1        0.967062   0.050361  19.203 1.79e-12 ***
# A1:C1       -2.003875   0.050361 -39.790  < 2e-16 ***
# B1:C1        0.048000   0.050361   0.953    0.355
# A1:D1        0.038250   0.050361   0.760    0.459
# B1:D1        0.023625   0.050361   0.469    0.645
# C1:D1       -0.038438   0.050361  -0.763    0.456
# A1:B1:C1     1.568750   0.050361  31.150 9.49e-16 ***
# A1:B1:D1     0.049000   0.050361   0.973    0.345
# A1:C1:D1     0.009563   0.050361   0.190    0.852
# B1:C1:D1     0.017813   0.050361   0.354    0.728
# A1:B1:C1:D1  0.007062   0.050361   0.140    0.890

# The significant parameters are the same as from the F-test.  The effects are twice these parameter estimates:

  2*(lm.2$coefficients)[-1]
#       A1         B1         C1         D1       A1:B1      A1:C1
#   3.018875   3.975875  -3.596250   1.957750   1.934125  -4.007750
#     B1:C1      A1:D1      B1:D1      C1:D1    A1:B1:C1   A1:B1:D1
#   0.096000   0.076500   0.047250  -0.076875   3.137500   0.098000
#   A1:C1:D1   B1:C1:D1 A1:B1:C1:D1
#   0.019125   0.035625   0.014125

# The std errors are also the std errors of the effects divided by 2, so the  t-ratios returned
# by R are the same as the t-ratios as computed by our formulas. See next part.
```

So far, we have SS, F, t, and p-values, for each effect, all by R. Now, we'll confirm these results by hand. To that end, it is convenient if the levels of the factors are numerical, and -1 and +1. That can be done using the gen.factorial() as above, but without "factors=all" .

Now, suppose we want to compute all the contrasts. One way is to compute the y-totals, i.e., (1) a, b, ..., this way:

```
 one = sum(y[A==-1 & B==-1 & C==-1 & D==-1])
 a =   sum(y[A==+1 & B==-1 & C==-1 & D==-1]), etc.
```

and then compute the contrasts fom these y-totals. But there are too many, i.e. 2^4 = 16. Instead, there is a faster way of getting the contrasts which uses the special order of the effects and the magical properties of the +/- table. Make sure the values in the vector y are in their natural/Yate's order, and that the replication is the slowest changing factor. Here is the hint for this better way: the contrast of A is nothing more than sum(A*y).

3

c) Based on the above "hint," deduce how all of the other contrasts can be computed. Again, print them so the grader can see them.

```
   design = gen.factorial(c(rep(2,k),n),varNames=c("A","B","C","D","R"))
   attach(design)
   all.contrasts = c(sum(A*y), sum(B*y), sum(C*y), sum(D*y), sum(A*B*y), sum(A*C*y), sum(A*D*y),   sum(B*C*y), sum
(B*D*y), sum(C*D*y), sum(A*B*C*y), sum(A*B*D*y), sum(A*C*D*y), sum(B*C*D*y), sum(A*B*C*D*y) )

#  48.302  63.614 -57.540  31.324  30.946 -64.124   1.224   1.536   0.756
#   -1.230  50.200   1.568   0.306   0.570   0.226
```

d) From the above contrasts, compute all the effects, and perform a t-test on each. Use the MSE value saved in part b. Be careful and clear on what df you use for the tests. Are th effects, the t-ratios, and the p-values equal to the "by R" results in part b?

```
   all.effects = all.contrasts/(n*2^(k-1))
   all.t = (all.effects -0)/sqrt( MSE/ (n*2^(k-2) ))
   all.pv = 2*pt(abs(all.t),(n-1)*(2^k),lower.tail=F)    # df = N - p = n*2^k - 2^k

   cbind(all.effects, all.t, all.pv)

#       all.effects       all.t       all.pv
# [1,]    3.018875  29.9723026 1.740225e-15
# [2,]    3.975875  39.4736876 2.247046e-17
# [3,]   -3.596250 -35.7046559 1.100354e-16
# [4,]    1.957750  19.4371332 1.485334e-12
# [5,]    1.934125  19.2025770 1.790240e-12
# [6,]   -4.007750 -39.7901522 1.979949e-17
# [7,]    0.076500   0.7595151 4.585895e-01
# [8,]    0.096000   0.9531170 3.547091e-01
# [9,]    0.047250   0.4691123 6.453180e-01
# [10,]   -0.076875  -0.7632382 4.564284e-01
# [11,]    3.137500  31.1500474 9.485296e-16
# [12,]    0.098000   0.9729736 3.450474e-01
# [13,]    0.019125   0.1898788 8.517922e-01
# [14,]    0.035625   0.3536958 7.281845e-01
# [15,]    0.014125   0.1402373 8.902228e-01

# Same as in part b.
```