Exercise 10.1 Full factorial design

(a) To calculate the effect of a factor, X, we use the following formula:

$$\text{Effect}(X) = \frac{\sum_{i\in\text{high level runs}} y_i - \sum_{i\in\text{low level runs}} y_i}{2^{N-1}}, \tag{1}$$

where N is the number of factors.

(i) For the phosphate concentration:

$$\text{Effect}(P) = \frac{(7+8+12+7) - (10+11+11+7)}{4} = -\frac{5}{4}.$$

(ii) For the sucrose concentration:

$$\text{Effect}(S) = \frac{(10+11+12+7) - (7+8+11+7)}{4} = \frac{7}{4}.$$

(iii) For the nitrate concentration:

$$\text{Effect}(N) = \frac{(8+11+12+7) - (7+10+11+7)}{4} = \frac{3}{4}.$$

(b) The extended design matrix is given in Table 1. We make use of Eq. (1) to calculate

P	S	N	PS	PN	SN	PSN	**Growth rate**
+	−	−	−	−	+	+	7
−	+	−	−	+	−	+	10
+	−	+	−	+	−	−	8
−	+	+	−	−	+	−	11
−	−	−	+	+	+	−	11
+	+	+	+	+	+	+	12
+	+	−	+	−	−	−	7
−	−	+	+	−	−	+	7

Table 1: *Experimental design matrix for the growth rate of the investigated bateria. The factors are the concentration of phosphate (P), the concentration of sucrose (S) and the concentration of nitrate (N).*

the influence:

(i) For the phosphate & sucrose concentrations:

$$\text{Effect}(PS) = \frac{(11+12+7+7)-(7+10+8+11)}{4} = \frac{1}{4}.$$

(ii) For the phosphate & nitrate concentrations:

$$\text{Effect}(PN) = \frac{(10+8+11+12)-(7+11+7+7)}{4} = \frac{9}{4}.$$

(iii) For the sucrose & nitrate concentrations:

$$\text{Effect}(SN) = \frac{(7+11+11+12)-(10+8+7+7)}{4} = \frac{9}{4}.$$

(iv) For the phosphate, sucrose & nitrate concentrations:

$$\text{Effect}(PSN) = \frac{(7+10+12+7)-(8+11+11+7)}{4} = -\frac{1}{4}.$$

(c) The growth rate seems to be increased by the factors S and N and the interactions PS, PN and SN.

(d) To create the least squares model, we convert "+" to 1 and "−" to −1 for Table 1. The two models are then (see Listing 1 for Python code for creating the models) as follows,

(i) Model 1:

$$y = a_0 + a_1 \times P + a_2 \times S + a_3 \times N, \tag{2}$$

with coefficients as given in Table 2.

Coefficient	a_0	a_1	a_2	a_3
Value	$\frac{73}{8}$	$-\frac{5}{8}$	$\frac{7}{8}$	$\frac{3}{8}$

Table 2: *Regression coefficients for model 1.*

(ii) Model 2:

$$y = a_0 + a_1 \times P + a_2 \times S + a_3 \times N + a_4 \times PS + a_5 \times PN + a_6 \times SN + a_7 \times PSN, \tag{3}$$

with coefficients as given in Table 3.

We first note that the coefficients have identical signs to their corresponding effects, for instance, both a_1 and a_7 are negative as the effects of P and PSN are. Thus, we again find that S, N, PS, PN and SN positively influences the growth rate.

Coefficient	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7
Value	$\frac{73}{8}$	$-\frac{5}{8}$	$\frac{7}{8}$	$\frac{3}{8}$	$\frac{1}{8}$	$\frac{9}{8}$	$\frac{9}{8}$	$-\frac{1}{8}$

Table 3: *Regression coefficients for model 2.*

Secondly, we note that a_0 is just the average response, $a_0 = \frac{1}{8}(7 + 10 + 8 + 11 + 11 + 12 + 7 + 7) = \frac{73}{8}$.

Thirdly, we see that for all coefficients $a_1, \ldots, a_7$ we have that the regression coefficient is $\frac{1}{2}$ of the corresponding effect. We have for instance that $a_2 = \frac{7}{8} = \frac{1}{2} \times \frac{7}{4} = \frac{\text{Effect}(S)}{2}$. Let see if we can understand this by calculating the effect of S,

$$\text{Effect}(S) = \tfrac{1}{4}\left((y_2 + y_4 + y_6 + y_7) - (y_1 + y_3 + y_5 + y_8)\right),$$

using Eq. (3) for y_i with the corresponding values for P, S and N from Table 1. It is a bit tedious to write this out, so let us rewrite Eq. (3) as $y = \mathbf{a}^\top \mathbf{x}$, where $\mathbf{a}^\top = [a_0, a_1, \ldots, a_7]$, and $\mathbf{x}^\top = [1, P, S, \ldots, PSN]$. The effect of S is then,

$$\begin{aligned}\text{Effect}(S) &= \tfrac{1}{4}\mathbf{a}^\top\left((\mathbf{x}_2 + \mathbf{x}_4 + \mathbf{x}_6 + \mathbf{x}_7) - (\mathbf{x}_1 + \mathbf{x}_3 + \mathbf{x}_5 + \mathbf{x}_8)\right)\\ &= \tfrac{1}{4}\mathbf{a}^\top[0, 0, 8, 0, 0, 0, 0, 0]^\top\\ &= \tfrac{8}{4}a_2 = 2a_2,\end{aligned}$$

and similarly for the other coefficients. That is, the effects are twice the regression coefficients. More generally, if the "high" value of S is s_+ and the "low" is s_-, then we get (by the same approach),

$$\text{Effect}(S) = \tfrac{1}{4}\mathbf{a}^\top[0, 0, 4(s_+ - s_-), 0, 0, 0, 0, 0]^\top = (s_+ - s_-)a_2.$$

Thus, the general relation between a regression coefficient and the corresponding effect is,

$$a_i = \frac{\text{Effect}(i)}{i_+ - i_-},$$

where i is the factor, a_i the corresponding regression coefficient and i_+ is the value for which i is considered "high" and i_- is the value for which i is considered "low".

In conclusion, the regression coefficients contain the same information as the effects we have previously calculated.

Exercise 10.2 Fractional factorial design

(a) With 6 factors in a two-level design, the number of experiments would be $2^6 = 64$.

(b) When we carry out a full factorial design, we do enough experiments so that we can determine the effect of all factors. When we do a fractional factorial design, we do too few experiments to uniquely determine all the effects. When we estimate effects, we then find that the expressions for some of the effects are identical and there is no way that we can distinguish between them. The extreme case would be if we only carried out one experiment, then we would not have any information on how the different variables influences out measured quantity.

In the present case, we need 64 experiments (see point (a)) but we can only do 16. We therefore know that some of our factors will be confounded with each other.

Luckily, we have some choice and we can, for instance, make it so that the main effects are confounded with higher-order interactions. This is useful as we often can neglect higher-order interactions compared to the main effects.

(c) A defining contrast is a relation that can be used to deduce what variables are confounded. In this case, we decided on two generators which can be used to generate the defining contrasts,

- $E = ABC \implies E^2 = 1 = ABCE$.
- $F = BCD \implies F^2 = 1 = BCDF$.

In addition, we have that,

$$1 \times 1 = 1 = ABCE \times BCDF = ADEF.$$

Thus, we have 3 defining constrasts here:

- $1 = ABCE$.
- $1 = BCDF$.
- $1 = ADEF$.

(d) Here, we have made the choices $E = ABC$ and $F = BCD$. This means that we confound the two main effects E and F with 3-factor interactions. Let check the confounding in full detail for the main effects (see also the summary in Table 4):

- For A we have:
  (i) $A = A \times 1 = A \times ABCE = BCE$,
  (ii) $A = A \times 1 = A \times BCDF = ABCDF$,
  (iii) $A = A \times 1 = A \times ADEF = DEF$,

  and we see that A is confounded with BCE, DEF and $ABCDF$.
- For B we have:
  (i) $B = B \times 1 = B \times ABCE = ACE$,

(ii) $B = B \times 1 = B \times BCDF = CDF$,
(iii) $B = B \times 1 = B \times ADEF = ABDEF$,

and we see that B is confounded with ACE, CDF and $ABDEF$.

- For C we have:
  (i) $C = C \times 1 = C \times ABCE = ABE$,
  (ii) $C = C \times 1 = C \times BCDF = BDF$,
  (iii) $C = C \times 1 = C \times ADEF = ACDEF$,

  and we see that C is confounded with ABE, BDF and $ACDEF$.
- For D we have:
  (i) $D = D \times 1 = D \times ABCE = ABCDE$,
  (ii) $D = D \times 1 = D \times BCDF = BCF$,
  (iii) $D = D \times 1 = D \times ADEF = AEF$,

  and we see that D is confounded with AEF, BCF and $ABCDE$.
- For E we have:
  (i) $E = E \times 1 = E \times ABCE = ABC$,
  (ii) $E = E \times 1 = E \times BCDF = BCDEF$,
  (iii) $E = E \times 1 = E \times ADEF = ADF$,

  and we see that E is confounded with ABC, ADF and $BCDEF$.
- For F we have:
  (i) $F = F \times 1 = F \times ABCE = ABCEF$,
  (ii) $F = F \times 1 = F \times BCDF = BCD$,
  (iii) $F = F \times 1 = F \times ADEF = ADE$,

  and we see that F is confounded with ADE, BCD and $ABCEF$.

Based on this, we conclude that *none* of the main effects are confounded with 2-factor interactions.

(e) The resolution is identical to the length of the shortest defining contrast. From point (c) we see that the shortest defining contrast has a length of 4. This means that our design is a 2^{6-2}_{IV} fractional factorial design.

We also know that given the resolution R, a p-order factor will be confounded with other factors of order $R - p$ or higher. For the main effects, $p = 1$, and we have then that these effects will be confounded with factors of order $R - p = 4 - 1 = 3$ or higher. This is exactly what we found in point (d). Thus, we could have answered the previous question without all the calculations just by considering the resolution.

(f) The design matrix for the 6 main effects (see Listing 2 for Python code for generating the matrix) is given in Table 5.

Main effect	**Confounded with**
A	BCE, DEF & $ABCDF$
B	ACE, CDF & $ABDEF$
C	ABE, BDF & $ACDEF$
D	AEF, BCF & $ABCDE$
E	ABC, ADF & $BCDEF$
F	ADE, BCD & $ABCEF$

Table 4: *Confounding between variables in the 2^{6-2}_{IV} fractional factorial design.*

A	B	C	D	E	F
$-$	$-$	$-$	$-$	$-$	$-$
$-$	$-$	$-$	$+$	$-$	$+$
$-$	$-$	$+$	$-$	$+$	$+$
$-$	$-$	$+$	$+$	$+$	$-$
$-$	$+$	$-$	$-$	$+$	$+$
$-$	$+$	$-$	$+$	$+$	$-$
$-$	$+$	$+$	$-$	$-$	$-$
$-$	$+$	$+$	$+$	$-$	$+$
$+$	$-$	$-$	$-$	$+$	$-$
$+$	$-$	$-$	$+$	$+$	$+$
$+$	$-$	$+$	$-$	$-$	$+$
$+$	$-$	$+$	$+$	$-$	$-$
$+$	$+$	$-$	$-$	$-$	$+$
$+$	$+$	$-$	$+$	$-$	$-$
$+$	$+$	$+$	$-$	$+$	$-$
$+$	$+$	$+$	$+$	$+$	$+$

Table 5: *Experimental design matrix for the 2^{6-2}_{IV} fractional factorial design.*

(g) With 8 experiments, we get even less information and we expect the confounding to change. This is a 2^{6-3} fractional design. In this case we can determine 3 of the main effects fully ($2^3 = 2^{6-3}$) and for the remaining 3 we have to select generators. Let us say that we select A, B and C as the main effects we determine fully. Then, we have to choose generators for D, E and F by combining A, B and C. We can make 3

combinations* using 2 of the three variables, for instance,

(i) $D = AB$,

(ii) $E = AC$,

(iii) $F = BC$.

The defining contrasts will then be,

(i) $D = AB \implies D^2 = 1 = ABD$,

(ii) $E = AC \implies E^2 = 1 = ACE$,

(iii) $F = BC \implies F^2 = 1 = BCF$,

(iv) $1 = 1 \times 1 = ABD \times ACE = BCDE$,

(v) $1 = 1 \times 1 = ABD \times BCF = ACDF$,

(vi) $1 = 1 \times 1 = ACE \times BCF = ACDF$,

(vii) $1 = 1 \times 1 \times 1 = ABD \times ACE \times BCF = DEF$.

We see from this that the shortest defining contrast has a length of 3 (i.e. the resolution is $R = 3$) and we have a 2^{6-3}_{III} fractional factorial design. We know then that the main effects ($p = 1$) will be confounded with factors of order $R - p = 3 - 1 = 2$ or higher. So we have that our main effects will be confounded with 2-factor interactions.

In general, we want our main effects to not be confounded with interactions lower than 3-factor interactions. We would then advise against doing only 8 experiments, and keep to the original plan with 16 experiments to be able to better assess the main effects.

The design matrix for our selection of generators is given in Table 6. Note that you might get different design matrices, depending on your choice of generators.

*The general formula for the number of combinations when we have n objects and select r of them is $n!/(r!(n-r)!)$.

A	B	C	D	E	F
$-$	$-$	$-$	$+$	$+$	$+$
$-$	$-$	$+$	$+$	$-$	$-$
$-$	$+$	$-$	$-$	$+$	$-$
$-$	$+$	$+$	$-$	$-$	$+$
$+$	$-$	$-$	$-$	$-$	$+$
$+$	$-$	$+$	$-$	$+$	$-$
$+$	$+$	$-$	$+$	$-$	$-$
$+$	$+$	$+$	$+$	$+$	$+$

Table 6: *Experimental design matrix for the* 2^{6-3}_{III} *fractional factorial design.*

Python solutions

```
import numpy as np
from sklearn.linear_model import LinearRegression


def effect_column(y, design, column_id):
    """Calculate the effect for a given column."""
    print('Column:', column_id)
    values = design[:, column_id]
    pos = np.where(values == 1)[0]
    neg = np.where(values == -1)[0]
    print('\tPositive terms (index):', pos)
    print('\tNegative terms (index):', neg)
    y_pos = sum(y[pos])
    print('\tPositive terms:', ' + '.join([str(i) for i in y[pos]]))
    y_neg = sum(y[neg])
    print('\tNegative terms:', ' + '.join([str(i) for i in y[neg]]))
    print('\tPositive - negative:', y_pos - y_neg)
    return (y_pos - y_neg) / len(pos)


def main():
    y = np.array([7, 10, 8, 11, 11, 12, 7, 7])

    P = np.array([1, -1, 1, -1, -1, 1, 1, -1])
    S = np.array([-1, 1, -1, 1, -1, 1, 1, -1])
    N = np.array([-1, -1, 1, 1, -1, 1, -1, 1])
    PS = P * S
    PN = P * N
    SN = S * N
    PSN = P * S * N

    design = np.zeros((8, 7))
    for i, val in enumerate((P, S, N, PS, PN, SN, PSN)):
        design[:, i] = val

    print('Design matrix:')
    print(design)

    variables = ['P', 'S', 'N', 'PS', 'PN', 'SN', 'PSN']

    effect = [effect_column(y, design, i) for i in range(7)]

    print('Effects:')
    for i, j in zip(variables, effect):
        print('\t{}: {}'.format(i, j))

    X1 = design[:, :3]
```

```
    model1 = LinearRegression()
    model1.fit(X1, y)
    print(model1.intercept_)
    print(model1.coef_)

    X2 = design
    model2 = LinearRegression()
    model2.fit(X2, y)
    print(model2.intercept_)
    print(model2.coef_)


if __name__ == '__main__':
    main()
```

Listing 1: *Python code for the full fractional design.*

```
from itertools import product
import numpy as np


def design_matrix_16():
    # Generate for A, B, C, D first:
    ABCD = np.array(list(product([-1, 1], repeat=4)))
    # Use the generators for getting E and F
    # E = ABC:
    E = ABCD[:, 0] * ABCD[:, 1] * ABCD[:, 2]
    # F = BCD:
    F = ABCD[:, 1] * ABCD[:, 2] * ABCD[:, 3]
    design = np.zeros((16, 6), dtype=np.int64)
    for i in range(4):
        design[:, i] = ABCD[:, i]
    design[:, 4] = E
    design[:, 5] = F
    return design


def design_matrix_8():
    # Generate for A, B, C first:
    ABC = np.array(list(product([-1, 1], repeat=3)))
    # Use the generators for getting E, E and F
    # D = AB:
    D = ABC[:, 0] * ABC[:, 1]
    # E = AC:
    E = ABC[:, 0] * ABC[:, 2]
    # F = BC:
    F = ABC[:, 1] * ABC[:, 2]
    design = np.zeros((8, 6), dtype=np.int64)
    for i in range(3):
        design[:, i] = ABC[:, i]
    design[:, 3] = D
```

```
    design[:, 4] = E
    design[:, 5] = F
    return design


def print_design(design):
    for row in design:
        txt = ['$+$' if i > 0 else '$-$' for i in row]
        txt = ' & '.join(txt)
        print(r'{} \\'.format(txt))


print_design(design_matrix_16())
print()
print_design(design_matrix_8())
```

Listing 2: *Python code for the fractional factorial design.*