# The Olympic Diet Problem

Emma Tarmey, 2940 4045

May 2021

**Abstract**

When an athlete trains for the Olympic games, they will need to consume far more food and drink than people who are not Olympic athletes. This is because they expend significant amounts energy in their training regiments and require far more nutrients to continually repair their muscles. However, there remain several questions. What types of foods does an Olympic athlete eat? How much more does an Olympic athlete eat compared to a non-Olympian? How much does such a diet cost? To explore these questions in more detail, we directly compare the requirements of Olympian and non-Olympian diets. From here, we use a newly updated formulation of "The Diet Problem" to explain how much more an Olympic athlete eats in terms of how much such a diet would cost. From here, we determine which aspects of an Olympian's diet are incurring the greatest cost and draw conclusions about the realistic livability of our diet-plan.

# Contents

# Chapter 1

# Introduction

## 1.1 Introducing the Diet Problem

The diet problem is an optimisation problem first proposed by George J. Stigler in the Journal of Farm Economics (Stigler, 1945). In particular, it is a linear programming problem for determining the minimum cost of feeding one person subject to a number of minimum nutritional requirements.

In his original paper, titled "The Cost of Subsistence", Stigler considers 9 nutritional requirements and 80 common foods, with their associated costs. He then uses a heuristic technique to determine a diet composed of these foods of minimal total cost, which satisfies all nutritional requirements.

The nine nutritional requirements Stigler uses as constraints and modern equivalents to these requirements are as seen in Table 1.1 We have sourced modern RDA values sources from the NHS (*Reference Intakes Explained*, 2020) and UK government dietary recommendations (*Government Dietary Recommendations*, 2016). Please note that Ascorbic acid is the form of vitamin C found in common foods. (*Vitamins and Minerals - Vitamin C*, 2020)

The nutritional values and costs of common foods available are recorded in detail, as part of two Tables, each with thirteen columns where each row entry corresponded to one common food. From here, a solution is obtained by a repeated procedure of by means of formulating composite "meta-foods" and then eliminating "redundant" foods to reduce the Table size. Composite "meta-foods" are formed by taking the mean average of the nutritional content and cost of two or more foods. A food is considered "redundant" if it's nutritional benefits per unit cost are all (or almost all) less than or equal to that of another food. As such a food would never be part of an optimal diet, it may be safely

Table 1.1: Stigler's Nutritional Requirements

(a) Stigler's Requirements (1945)

| Nutrient | Amount |
|---|---|
| Calories | 3000 kcal |
| Protein | 70 g |
| Calcium | 800 mg |
| Iron | 12 mg |
| Vitamin A | 5000 IU |
| Thiamine | 1.8 mg |
| Riboflavin | 2.7 mg |
| Niacin | 18 mg |
| Ascorbic Acid | 75 mg |

(b) Modern Male Adult RDA (2016)

| Nutrient | Amount |
|---|---|
| Calories | 2500 kcal |
| Protein | 55.5 g |
| Calcium | 700 mg |
| Iron | 8 mg |
| Vitamin A | 700 $\mu$g |
| Thiamine | 1.0 mg |
| Riboflavin | 1.3 mg |
| Niacin equivalent | 16.5 mg |
| Vitamin C | 40 mg |

discarded as a means of reducing the size of the problem. This allowed the total number of foods being examined to reduce from 77 to 14.

Stigler then trials a handful of the 510 possible combinations of the remaining items to find a solution of lowest total cost. It is determined that, while the solution was not fully optimal, the further reductions in cost by trialling additional combinations would be minimal. This in turn became an example of a "heuristic" solution, one which is not optimal but exists as an approximate solution where methods for finding an optimal solution did not yet exist.

The resultant solution, as seen in Table 1.2, became known as the "Stigler Diet". It is of course not a viable diet for a healthy person. Despite this, it stands as an excellent example of an optimisation problem through which we may examine existing diets.

Table 1.2: Optimal Diets of Stigler's Diet Problem

(a) The Stigler Diet (1939)

| Commodity | Quantity | Cost |
|---|---|---|
| Wheat Flour | 370 lbs | $13.33 |
| Evaporated Milk | 57 cans | $3.84 |
| Cabbage | 111 lbs | $4.11 |
| Spinach | 23 lbs | $1.85 |
| Dried Navy Beans | 285 lbs | $16.80 |
| Total Cost | - | $39.93 |

(b) The Stigler Diet (1945)

| Commodity | Quantity | Cost |
|---|---|---|
| Wheat Flour | 535 lbs | $34.53 |
| Cabbage | 107 lbs | $5.23 |
| Spinach | 13 lbs | $1.56 |
| Pancake Flour | 134 lbs | $13.08 |
| Pork Liver | 25 lbs | $5.48 |
| Total Cost | - | $59.88 |

## 1.2 Solving the Diet Problem

In 1947, Dantzig solved the diet problem by means of formulating it as a linear programming problem. Dantzig then used the Simplex algorithm, which he had developed, to solve the resulting linear programming problem. This in turn enabled an solution to be obtained directly, with proven optimality. (Dantzig, 1990) In particular, the linear program (LP) of the Diet Problem can be codified as follows (*The Diet Problem*, 2020):

$$
\begin{array}{lll}
\text{minimize} & \displaystyle\sum_{i=1}^{n} c_i x_i & \\
\text{subject to} & \displaystyle\sum_{i=1}^{n} a_{ij} x_i \geq N_{min_j}, & j = 1, ..., m \\
 & \displaystyle\sum_{i=1}^{n} a_{ij} x_i \leq N_{max_j}, & j = 1, ..., m \\
 & x_i \geq F_{min_i}, & i = 1, ..., n \\
 & x_i \leq F_{max_i}, & i = 1, ..., n
\end{array}
$$

where:

$F$ = Set of all $n$ foods, $i = 1, ..., n$
$N$ = Set of all $m$ nutrients, $j = 1, ..., m$
$a_{ij}$ = Amount of nutrient $j$ per unit of food $i$
$c_i$ = Cost per unit of food $i$
$x_i$ = Amount of food $i$ contained in solution diet
$F_{min_i}$ = Minimum requirement of food $i$
$F_{max_i}$ = Maximum allowance of food $i$
$N_{min_j}$ = Minimum requirement of nutrient $j$
$N_{max_j}$ = Maximum allowance of nutrient $j$

The Simplex method works by finding the optimal solution to a given LP in canonical form using row operations, which are performed on a "tableau" form of the canonical LP. In particular, the canonical form of an LP is:

$$\begin{array}{ll} \text{maximise} & c^T x \\ \text{subject to} & Ax \leq b \\ & x \geq 0 \end{array}$$

where:

$$\begin{aligned} c &= \text{Column vector of costs} \\ x &= \text{Column vector of costs} \\ A &= \text{Matrix of coefficients of constraints} \\ b &= \text{Column vectors of constraint right-hand values} \end{aligned}$$

This allowed Dantzig to identify a solution directly - a diet whose cost is minimal to proven optimality (Dantzig, 1991). Though this diet too is unlivable (Dantzig, 1990), it became a foundational example of optimisation problems in linear programming.

Though this form of the diet problem has already been solved to proven optimality, we may use the techniques established in this solution method to solve a wider variety of problems. In particular, we are extending the diet problem to accommodate a wider and more varied set of constraints. We additionally use a variable time-frame, an entirely new data-set and a set of comparisons to form new solutions and new results.

From here, we endeavour to form an optimal diet-plan of minimal total cost, accommodating the dietary needs of an Olympic athlete. To do so, we must first determine what the needs of an Olympic athlete are and which foods may plausibly fulfill these needs.

# Chapter 2

# The Diet of an Olympian

## 2.1 The dietary requirements of an Olympic Athlete

For the purposes of more direct comparison to other diet-plans and greater usability as a general diet plan, we are overhauling the constraints of Stigler's Diet Problem formulation. In particular, we are replacing the 9 nutritive requirements with 8 more modern dietary guidelines:

- Calories
- Fat
- Saturates
- Carbohydrates
- Sugar
- Protein
- Salt
- Fibre

This not only ensures compatibility with a wide variety of available data, but also enables us use our model solutions as a more practical guide for modern living. This is because the above guidelines are much more readily assessed from supermarket websites and food item packaging.

For our model, we are looking at the dietary requirements of Olympians and non-Olympians in order to compare them. Though, there also exist differences between males and females which need to be accounted for. As a result, we have four total categories of people, whose dietary requirements must be examined separately:

- Male Olympian
- Female Olympian
- Male Non-Olympian
- Female Non-Olympian

We source our data for the dietary requirements of non-Olympians from Table 2 of the UK government guidelines published in 2019 (*Government Dietary Recommendations*, 2016). We source our data for the dietary requirements of Olympic athletes various sources, centrally from Table 2 of (Grandjean, 1997). Our new dietary requirements are as seen in Table 2.1 and Table 2.2. Please see appendix A.2 for the full calculations.

Now, the recommended daily intake allowance is defined as the amount that a person should intake on a given day, meaning a healthy individual should attempt to be as close to 100% as possible in each category. To ensure that our Olympians never encounter any kind of deficiency during their training, we will treat our values as lower bounds.

Table 2.1: Olympic Dietary Requirements

(a) Olympic Male RDA

| Requirement | Amount |
|---|---|
| Calories | 15540 |
| Fat | 628.51 g |
| Saturates | 200.86 g |
| Carbohydrates | 1825.95 g |
| Sugar | 40.0 g |
| Protein | 644.91 g |
| Salt | 6.0 g |
| Fibre | 35.0 g |

(b) Olympic Female RDA

| Requirement | Amount |
|---|---|
| Calories | 10861 |
| Fat | 500.81 g |
| Saturates | 154.1 g |
| Carbohydrates | 1140.41 g |
| Sugar | 40.0 g |
| Protein | 448.02 g |
| Salt | 6.0 g |
| Fibre | 35.0 g |

Table 2.2: Non-Olympic Dietary Requirements

(a) Non-Olympic Male RDA

| Requirement | Amount |
|---|---|
| Calories | 2500 |
| Fat | 97.0 g |
| Saturates | 31.0 g |
| Carbohydrates | 333.0 g |
| Sugar | 33.0 g |
| Protein | 55.5 g |
| Salt | 6.0 g |
| Fibre | 30.0 g |

(b) Non-Olympic Female RDA

| Requirement | Amount |
|---|---|
| Calories | 2000 |
| Fat | 78.0 g |
| Saturates | 24.0 g |
| Carbohydrates | 267.0 g |
| Sugar | 27.0 g |
| Protein | 45.0 g |
| Salt | 6.0 g |
| Fibre | 30.0 g |

Now, as we can see, the dietary requirements are significantly higher for Olympic athletes than for non-Olympian individuals. Moreover, there is a disparity between the requirements of males and females in both categories. The following bar chart illustrates the differences between the four sets of requirements more clearly.

Please note that, in Figures 2.1, 2.2 and 2.3, the "Calories" requirements has been scaled down by a factor of 100 to more easily illustrate the differences between our four categories. All other requirements are measured in grams and not scaled.

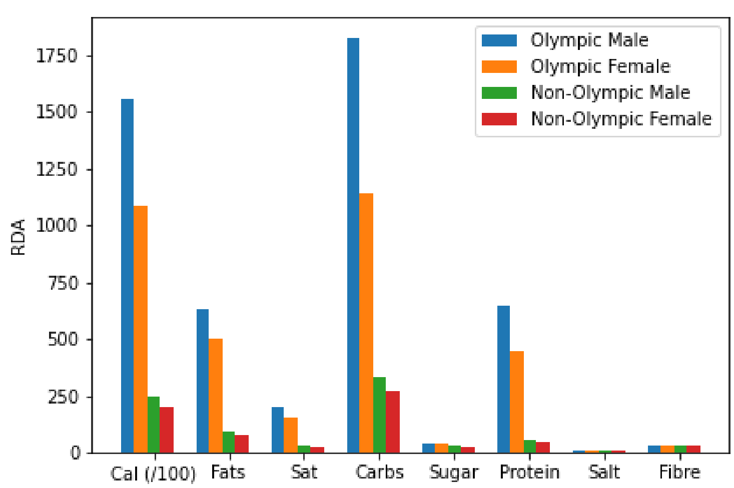


Figure 2.1: Recommended Daily Allowances

The above bar-chart illustrates clearly two key aspects of our requirements: the disparity between Male and Female intake requirements, and the disparity between Olympian and Non-Olympian intake requirements.

Exploring the disparity between Olympic and non-Olympic requirements more closely gives the following:

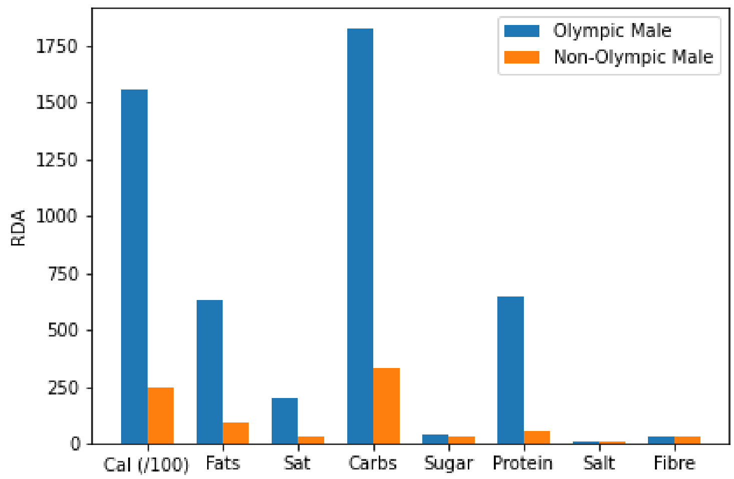


Figure 2.2: Male RDA

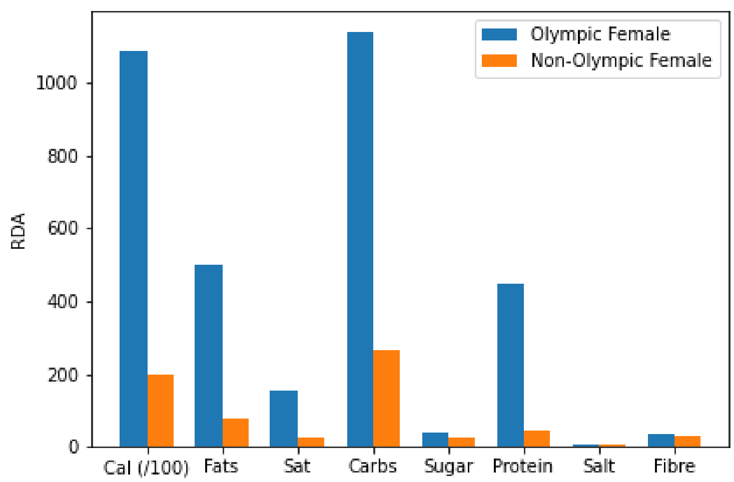


Figure 2.3: Female RDA

The above graphs illustrate that, although we have significant differences between the male and female categories, the relationship between Olympic and non-Olympic requirements is the same within both sexes. Seeing as both Male and Female experience disparities of roughly the same size in each category, we can expect the results of our diet plan to mirror this relationship.

Consequently, we compute four optimal diets and compare the characteristics of each to fully explore our problem.

## 2.2 The intake of an Olympic Athlete

### 2.2.1 Determining the Items

To make our diet plan more realistically livable than the Stigler diet, we begin from a position of considering existing Olympic diets. To do so, we consult Tables 1, 2, 3, 4, 5 and 6 of (Pelly et al., 2011) and Tables 1 and 4 of (Grivetti and Applegate, 1997).

We then identify all foods and drinks mentioned as being already part of an existing Olympic diet. We then collect all such items and list them together. This has resulted in a data-set containing 35 food items and 7 drinks items, totalling 42 items which may appear in our resultant diet plan.

Please note that, for editorial reasons, we have chosen to exclude the drink "Powerade" on account of being unavailable in UK supermarkets.

### 2.2.2 Determining the Costs

To determine the costs of sourcing such items, we firstly look to (*Which? reveals the cheapest supermarket of the year*, 2021). In particular, this source gives us an list of UK supermarkets, ordered by the total cost of a number of key items:

- Lidl
- Aldi
- Asda
- Tesco
- Morrisons
- Sainsbury's
- Ocado
- Waitrose

From here, we begin an iterative procedure. For each item, we check availability at the top-most supermarket. If available online from here, we record the price. If not, we move to the next supermarket in the list. Repeating until we find appropriately minimal costs for all items in our data-set. This approach is to ensure that we always source the lowest cost of a given item, subject only to availability in the UK.

### 2.2.3 Determining the Nutritional Benefits

To determine the dietary benefits given by each item, we centrally consult (*Eat This Much*, 2021), a website which maintains a vast database of foods / drinks and their associated dietary benefits. In particular, dietary information is provided for a given portion of some food, with an associated weight. From here, the nutritional benefits of consuming one such portion of the given food are given. We may then record the relevant information for each of our 8 parameters for each of our 42 foods.

To account for the difference in the sizes of items available in supermarkets and the portion sizes given by (*Eat This Much*, 2021), we take into consideration the weights of both. This allows us to find the proportionality between these two weights. From here, we create a “price per portion” field in our data, giving the effective value of a given item’s cost. This then allows us to simply total the cost of an integer number of portions of a given item in our solution.

## 2.3 Olympic Data Set

The resultant data-set from our above investigation is as follows in Tables 2.3 and 2.4.

Table 2.3: Dietary Information of our Food and Drinks items

| Name | Calories | Fat | Saturates | Carbohydrates | Sugar | Protein | Salt | Fibre |
|---|---|---|---|---|---|---|---|---|
| Anchovies (canned in oil) | 8.4 | 0.4g | 0.1g | 0.0g | 0.0g | 1.2g | 146.7mg | 0.0g |
| Apples | 95 | 0.3g | 0.1g | 25.1g | 18.9g | 0.5g | 1.8mg | 4.4g |
| Asparagus | 3.2 | 0.0g | 0.0g | 0.6g | 0.3g | 0.4g | 0.3mg | 0.3g |
| Beef | 78.8 | 6.4g | 2.6g | 0.0g | 0.0g | 5.0g | 16.7mg | 0.0g |
| Baguettes | | | | | | | | |
| Beans (baked, tinned) | 265.7 | 1.0g | 0.3g | 51.8g | 19.7g | 12.1g | 2.5mg | 13.9g |
| Beets | 35.3 | 0.1g | 0.0g | 7.8g | 5.5g | 1.3g | 64mg | 2.3g |
| Bread-Loaves | 35.3 | 0.1g | 0.0g | 7.8g | 5.5g | 1.3g | 64mg | 2.3g |
| Bread-Rolls | | | | | | | | |
| Cabbage | 227 | 0.9g | 0.3g | 52.7g | 29.1g | 11.6g | 163.4mg | 22.7g |
| Cauliflower | 26.8 | 0.3g | 0.1g | 5.3g | 2.0g | 2.1g | 32.1mg | 2.1g |
| Cereal (corn flakes) | 100 | 0.1g | 0.0g | 23.5g | 2.7g | 2.1g | 204.1mg | 0.9g |
| Cheese (cheddar) | 115.1 | 9.6g | 5.5g | 0.4g | 0.1g | 6.8g | 182.6mg | 0.0g |
| Eggs | 71.5 | 4.8g | 1.6g | 0.4g | 0.2g | 6.3g | 71.0mg | 0.0g |
| Garlic | 4.5 | 0.0g | 0.0g | 1.0g | 0.0g | 0.2g | 0.5mg | 0.1g |
| Glucose Supplements | 15 | 0.0g | 0.0g | 0.0g | 3.7g | 0.0g | 0.0mg | 0.0g |
| Grape-Nuts Cereal | 419.7 | 2.4g | 0.5g | 92.6g | 10.2g | 12.5g | 545.1mg | 13.9g |
| Honey | 63.8 | 0.0g | 0.0g | 17.3g | 17.2g | 0.1g | 0.0mg | 0.0g |
| Ice Cream | 273.2 | 14.5g | 9.0g | 31.2g | 28.0g | 4.6g | 105.6mg | 0.9g |
| Lemons | 24.4 | 0.3g | 0.0g | 7.8g | 2.1g | 0.9g | 1.7mg | 2.4g |
| Lettuce | 5.4 | 0.1g | 0.0g | 1.0g | 0.3g | 0.5g | 10.1mg | 0.5g |
| Nuts (peanuts, salted) | 163.9 | 13.8g | 1.8g | 5.6g | 0.0g | 7.3g | 122.8mg | 2.5g |
| Onions (red, medium) | 44 | 0.1g | 0.0g | 10.3g | 4.7g | 1.2g | 4.4mg | 1.9g |
| Parsley | 1.4 | 0.0g | 0.0g | 0.2g | 0.0g | 0.1g | 2.1mg | 0.1g |
| Pasta (penne) | 37.1 | 0.3g | 0.0g | 7.1g | 0.0g | 1.5g | 1.7mg | 0.0g |
| Peaches | 58.5 | 0.4g | 0.0g | 14.3g | 12.6g | 1.4g | 0.0mg | 2.3g |
| Peas | 103.2 | 0.5g | 0.1g | 18.3g | 6.7g | 7.0g | 144.mg | 6.0g |
| Pizza (cheese and tomato) | 250 | 7.1g | 3.3g | 30.0g | 3.6g | 12.0g | 0.0mg | 2.3g |
| Potatoes | 284.1 | 0.3g | 0.1g | 64.5g | 2.9g | 7.5g | 22.1mg | 8.1g |
| Poultry (chicken breast) | 141.6 | 3.1g | 0.7g | 0.0g | 0.0g | 26.6g | 53.1mg | 0.0g |
| Rice (long grain) | 684.5 | 5.4g | 1.1g | 142.9g | 1.6g | 14.7g | 13.0mg | 6.5g |
| Spaghetti | 105.2 | 0.4g | 0.1g | 21.2g | 0.8g | 3.7g | 1.7mg | 0.9g |
| Spinach | 6.9 | 0.1g | 0.0g | 1.1g | 0.1g | 0.9g | 23.7mg | 0.7g |
| Strawberries | 46.1 | 0.4g | 0.0g | 11.1g | 7.0g | 1.0g | 1.4mg | 2.9g |
| Sugar (granulated) | 15.5 | 0.0g | 0.0g | 4.0g | 4.0g | 0.0g | 0.0mg | 0.0g |
| Coffee | 10.6 | 0.0g | 0.0g | 2.3g | 0.0g | 0.4g | 1.1mg | 0.0g |
| Milk-Dairy (Dairy, whole) | 163.8 | 9.4g | 5.8g | 11.9g | 0.0g | 8.4g | 125.4mg | 0.0g |
| Milk-Soy (Soy, unsweetened) | 85 | 5.3g | 1.0g | 1.0g | 0.5g | 8.5g | 60mg | 0.0g |
| Soft-Drinks (diet cola) | 10 | 0.2g | 0.0g | 1.5g | 0.0g | 0.6g | 40mg | 0.0g |
| Water (still, bottled) | 0 | 0.0g | 0.0g | 0.0g | 0.0g | 0.0g | 0.0mg | 0.0g |
| Whiskey (40% ABV) | 104.3 | 0.0g | 0.0g | 0.0g | 0.0g | 0.0g | 0.0mg | 0.0g |
| Wine (Sauvignon Blanc) | 144.8 | 0.0g | 0.0g | 3.7g | 3.7g | 0.1g | 0.0mg | 0.0g |

Table 2.4: Cost Information of all Food and Drinks items

| Name | Cost | Unit (grams) | Serving (grams) | Cost per serving |
|---|---|---|---|---|
| Anchovies (canned in oil) | 0.49 | 50 | 4 | 0.0392 |
| Apples | 0.54 | 745 | 149 | 0.108 |
| Asparagus | 1.69 | 250 | 16 | 0.10816 |
| Beef | 4.45 | 700 | 28.35 | 0.180225 |
| Baguettes | 0.42 | 300 | 150 | 0.21 |
| Beans (baked, tinned) | 0.22 | 420 | 253 | 0.13252381 |
| Beets | 0.49 | 500 | 82 | 0.08036 |
| Bread-Loaves | 0.36 | 800 | 28.6 | 0.01287 |
| Bread-Rolls | 0.45 | 378 | 63 | 0.075 |
| Cabbage | 0.43 | 800 | 908 | 0.48805 |
| Cauliflower | 0.79 | 265 | 107 | 0.318981132 |
| Cereal (corn flakes) | 0.5 | 500 | 28 | 0.028 |
| Cheese (cheddar) | 1.19 | 200 | 28.35 | 0.1686825 |
| Eggs | 1.18 | 810 | 54 | 0.078666667 |
| Garlic | 0.69 | 9 | 3 | 0.23 |
| Glucose Supplements | 0.89 | 40 | 4 | 0.089 |
| Grape-Nuts Cereal | 2.98 | 580 | 115 | 0.590862069 |
| Honey | 0.85 | 340 | 21 | 0.0525 |
| Ice Cream | 1.29 | 1270 | 132 | 0.13407874 |
| Lemons | 0.69 | 336 | 84 | 0.1725 |
| Lettuce | 0.43 | 324 | 36 | 0.047777778 |
| Nuts (peanuts, salted) | 0.46 | 200 | 28.35 | 0.065205 |
| Onions (red, medium) | 0.47 | 330 | 110 | 0.156666667 |
| Parsley | 0.57 | 30 | 3.8 | 0.0722 |
| Pasta (penne) | 0.29 | 500 | 28.35 | 0.016443 |
| Peaches | 1.29 | 600 | 150 | 0.3225 |
| Peas | 0.55 | 907 | 134 | 0.081256891 |
| Pizza (cheese and tomato) | 0.67 | 314 | 100 | 0.213375796 |
| Potatoes | 0.49 | 1000 | 369 | 0.18081 |
| Poultry (chicken breast) | 1.69 | 300 | 118 | 0.664733333 |
| Rice (long grain) | 0.85 | 1000 | 185 | 0.15725 |
| Spaghetti | 0.2 | 500 | 28.35 | 0.01134 |
| Spinach | 1.79 | 450 | 30 | 0.119333333 |
| Strawberries | 1.39 | 500 | 144 | 0.40032 |
| Sugar (granulated) | 0.65 | 1000 | 4 | 0.0026 |
| Coffee | 1.39 | 200 | 3 | 0.02085 |
| Milk-Dairy (Dairy, whole) | 1.09 | 2270 | 250 | 0.120044053 |
| Milk-Soy (Soy, unsweetened) | 0.55 | 1000 | 250 | 0.1375 |
| Soft-Drinks (diet cola) | 0.17 | 2000 | 500 | 0.0425 |
| Water (still, bottled) | 0.17 | 2000 | 250 | 0.02125 |
| Whiskey (40% ABV) | 10.79 | 700 | 45 | 0.693642857 |
| Wine (Sauvignon Blanc) | 3.99 | 750 | 175 | 0.931 |

# Chapter 3

# The Olympic Diet Formulation

## 3.1 Formulating our Model

In formulating our new model, we start by overhauling the LP model proposed by Dantzig in 1947. The central aim of our new formulation is to form a livable diet plan that would realistically satisfy the needs of an Olympian or non-Olympian, given their varying needs.

To this end, we make three central changes:

- The change of constraints from nutritional requirements to more modern daily intake allowances
- The variable constraints for training and non-training days for an Olympic athlete, allowing for direct comparison with non-Olympic diets
- The extended time-frame of the diet plan from 1 day to t days, allowing variable training regiments within the time-frame

In addition, we are formulating four separate diet plans in order to be able to compare them, as this will allow a deeper exploration of the question of how much more an Olympic athlete eats compared to a non-Olympian. In our solution in particular, we will be extending the time-frame of the diet-plan to one week. Finally, we are utilising an entirely new data-set to form our optimal diet-plan solution.

Our new LP formulation is as follows:

$$\begin{aligned}
\text{minimize} \quad & \sum_{d=1}^{t}\sum_{i=1}^{n} c_i x_{d,i} \\
\text{subject to} \quad & \sum_{i=1}^{n} a_{ij} x_{d,i} \geq D_{min_j}(d), \quad d = 1, ..., t \ \ \& \ \ j = 1, ..., m \\
& x_{d,i} \geq 0, \qquad\qquad\qquad\quad d = 1, ..., t \ \ \& \ \ j = 1, ..., m
\end{aligned}$$

where:

| | |
|---|---|
| $F$ | = set of all $n$ foods |
| $N$ | = set of all $m$ dietary requirements |
| $D$ | = set of all $t$ days in our time-frame |
| $c_i$ | = cost per portion of food $i$ |
| $x_{d,i}$ | = number of portions of food $i$ on day $d$ in our diet plan |
| $a_{i,j}$ | = amount of dietary requirement $j$ satisfied by food $i$ |
| $D_{min_j}(d)$ | = minimum RDA for dietary requirement $j$ on day $d$ |
| $train_d$ | = whether or not day $d$ is a training day |
| $i$ | = iterator variable for all $n$ foods |
| $j$ | = iterator variable for all $m$ dietary requirements |
| $d$ | = iterator variable for all $t$ days in our time-frame |

In our case, we have from our data:

$$\begin{aligned}
m &= 42 \qquad \text{(foods and drinks)} \\
n &= 8 \qquad\qquad \text{(constraints)} \\
t &= 7 \qquad \text{(days in time-frame)}
\end{aligned}$$

For our solution in particular, we will be assuming that an Olympic athlete's training regiment consists of a variable regiment comprised of "training days" in which they require their full intake, and "rest days" in which they only require a reduced intake equal to that of a non-Olympian. Towards this end, we have the function $D_{min_j}(d)$ which is defined in the following way:

$$D_{min_j}(d) = \begin{cases} \text{Olympic RDA for requirement } j \text{ if } d \text{ is a training day} \\ \text{Non-Olympic RDA for requirement } j \text{ if } d \text{ is not a training day} \end{cases}$$

For modelling purposes, we will then place our four categories of person into the following regiments:

Table 3.1: Training Regiments for Diet Plan Models

(a) Olympians

| Day | Training |
|---|---|
| Monday | True |
| Tuesday | True |
| Wednesday | True |
| Thursday | True |
| Friday | True |
| Saturday | True |
| Sunday | False |

(b) Non-Olympians

| Day | Training |
|---|---|
| Monday | False |
| Tuesday | False |
| Wednesday | False |
| Thursday | False |
| Friday | False |
| Saturday | False |
| Sunday | False |

The distinction between male and female RDA will then be handled at the data input level, where constraint values are given to the program. This simplifies the model by allowing it to focus purely on whether a given day is a training day or not, while allowing us to compute the four optimal diet plans by means of varying the RDA constraints and training regiments within our data. With the above in mind, we then implement our new LP formulation in Pyomo.

Please note that, for computational simplicity, a slightly altered model program was used to find the non-Olympian diet-plans. In particular, for this model, we define:

$$D_{min_j}(d) = \text{Non-Olympic RDA for requirement } j \text{ always}$$

Please see appendices C.1 and C.2 for the full code.

## 3.2 Computing Our Optimal Diets

For our Pyomo implementation, the code we produce was inspired by the Pyomo Gallery implementation of Dantzig's 1947 Diet Problem Formulation (*A collection of Pyomo examples*, 2020). The model implementation itself was created using the Spyder Scientific Python environment. The translation of our above data-set into a Pyomo-compatible data file was accomplished using a Jupyter Notebook. The GLPK solver was then used to find the optimal diet plan solution for all four categories of people, the execution of which was managed using a Jupyter notebook. Please see appendices B.1, B.2, B.3 and B.4 for the full command line output.

Towards the goal of understanding our solution in more detail, we compute the optimal diets for one training day and one rest day separately from our overall solution. The Optimal Diets for one day under the given conditions with the given data-set are as follows:

Table 3.2: The Optimal Diet for 1 Training day

(a) Male

| Item | Portions | Cost |
|---|---|---|
| Ice Cream | 16 | £2.15 |
| Nuts | 26 | £1.70 |
| Spaghetti | 104 | £1.18 |
| Total Cost | - | £5.03 |

(b) Female

| Item | Portions | Cost |
|---|---|---|
| Ice Cream | 12 | £1.61 |
| Nuts | 22 | £1.43 |
| Spaghetti | 65 | £0.73 |
| Total Cost | - | £3.77 |

Table 3.3: The Optimal Diet for 1 Non-Training day

(a) Male

| Item | Portions | Cost |
|---|---|---|
| Bread-Loaved | 4 | £0.05 |
| Ice Cream | 2 | £0.27 |
| Milk-Dairy | 1 | £0.12 |
| Nuts | 4 | £0.26 |
| Spaghetti | 10 | £0.11 |
| Total Cost | - | £0.81 |

(b) Female

| Item | Portions | Cost |
|---|---|---|
| Bread-Loaved | 6 | £0.08 |
| Ice Cream | 2 | £0.27 |
| Nuts | 4 | £0.26 |
| Spaghetti | 6 | £0.07 |
| Sugar | 3 | £0.01 |
| Total Cost | - | £0.69 |

The optimal diets for seven days, given our training regiment, are as follows:

Table 3.4: The Optimal Olympic Diet for 7 days

(a) Male Olympian

| Item | Portions | Cost |
|---|---|---|
| Ice Cream | 98 | £13.14 |
| Nuts | 160 | £10.43 |
| Spaghetti | 634 | £7.16 |
| Bread-Loaved | 4 | £0.05 |
| Milk-Dairy | 1 | £0.12 |
| Total Cost | - | £30.90 |

(b) Female Olympian

| Item | Portions | Cost |
|---|---|---|
| Ice Cream | 74 | £9.92 |
| Nuts | 136 | £8.87 |
| Spaghetti | 396 | £4.47 |
| Bread-Loaved | 6 | £0.08 |
| Sugar | 3 | £0.01 |
| Total Cost | - | £23.35 |

Table 3.5: The Optimal Non-Olympic Diet for 7 days

(a) Male Non-Olympian

| Item | Portions | Cost |
|---|---|---|
| Bread-Loaved | 28 | £0.36 |
| Ice Cream | 14 | £1.88 |
| Milk-Dairy | 7 | £0.84 |
| Nuts | 28 | £1.83 |
| Spaghetti | 70 | £0.79 |
| Total Cost | - | £5.70 |

(b) Female Non-Olympian

| Item | Portions | Cost |
|---|---|---|
| Bread-Loaved | 42 | £0.54 |
| Ice Cream | 14 | £1.88 |
| Nuts | 28 | £1.83 |
| Spaghetti | 42 | £0.47 |
| Sugar | 21 | £0.05 |
| Total Cost | - | £4.77 |

## 3.3 Comparing optimal diet plan solutions

### 3.3.1 Comparing costs

The total costs for our four diet-plan solutions are visualised in Figure 3.1.

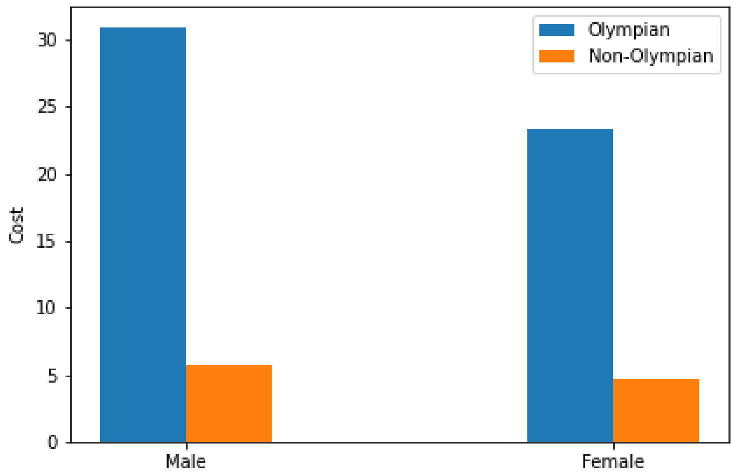


Figure 3.1: Total Costs

For our male diets, the Olympic Diet costs 5.42 times more in total. For our female diets, the Olympic Diet costs 4.90 times more. This is a very significant increase in both cases, clearly. But we also have a small disparity between the ratios for each sex. Male Olympians tend towards increasing their intake by a greater proportion (an increase of 0.52 times the equivalent Non-Olympian diet) than female Olympians. From here, we explore the cause of these increased costs.

### 3.3.2 Determining the source of our cost differences

To examine the above in more detail, we begin by considering what proportions of the diet plan's cost are spent on which foods. To this end, we calculate the percentage that each item contributes to the total cost. Please see appendix A.2 for the full calculations. We visualise these proportions as follows:

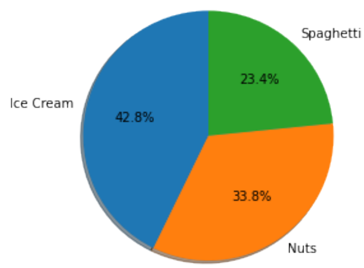


Figure 3.2: Cost Proportions of Male Training Day Diet

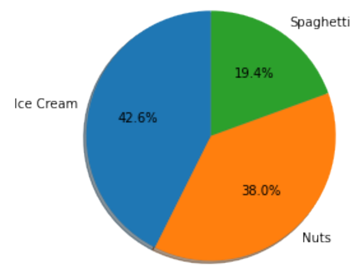


Figure 3.3: Cost Proportions of Female Training Day Diet

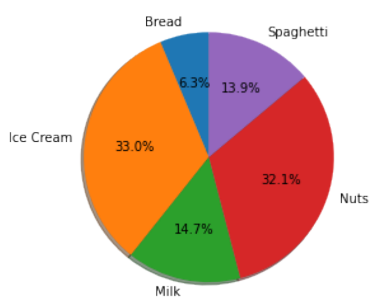


Figure 3.4: Cost Proportions of Male Rest Day Diet

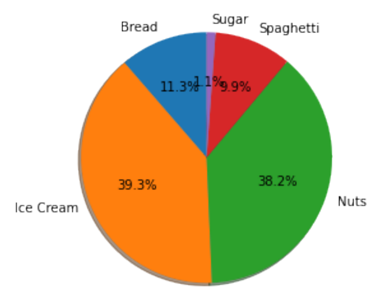


Figure 3.5: Cost Proportions of Female Rest Day Diet

Given our Olympic training regiment of six training days and one rest day within our seven day regiment, it is clear that the training day diet plan will be the source of any additional cost in the Olympian diet when compared to the non-Olympian diet. This is because, on rest days, our Olympians and non-Olympians will have the same dietary requirements. As a result, any additional cost incurred by the Olympic diet must have been caused by the presence of individually costly training-days.

During a training day, there are three items that an Olympian of either sex would intake when following our diet plan: Spaghetti, Nuts and Ice Cream. These foods have satisfied the eight dietary requirements given. From here we may determine which requirement was most strict upon the feasibility of our diet-plan solution.

We may do so by examining the surplus amount of each requirement in our diet-plan. This is because we seek to minimise the total cost of our diet-plan and, as a result, the surplus for each requirement should be as close to zero as possible in an optimal solution. Naturally, the requirements with the least surplus in our solution must have been the most difficult requirements to fulfill. In contrast, the requirements with the greatest surplus must have been the easiest requirement to fulfill. Due to the parallels between the male and female Olympian diets, we will examine the male diet-plan only, whose characteristics are largely very similar to the female diet-plan as per figures 3.2 and 3.3.

Table 3.6: Characteristics of the Training Day Diet

| Item | Portions | Calories | Fat (g) | Saturates (g) | Carbohydrates (g) | Sugar (g) | Protein (g) | Salt (mg) | Fibre (g) |
|---|---|---|---|---|---|---|---|---|---|
| Ice Cream | 16 | 4371.2 | 232 | 144 | 499.2 | 448 | 73.6 | 1689.6 | 14.4 |
| Nuts (peanuts, salted) | 26 | 4261.4 | 358.8 | 46.8 | 145.6 | 0 | 189.8 | 3192.8 | 65 |
| Spaghetti | 104 | 10940.8 | 41.6 | 10.4 | 2204.8 | 83.2 | 384.8 | 176.8 | 93.6 |
| Minimum Requirement | - | 15540 | 628.51 | 200.86 | 1825.95 | 40 | 644.91 | 6 | 35 |
| Surplus | - | 4033.4 | 3.89 | 0.34 | 1023.65 | 491.2 | 3.29 | 5053.2 | 138 |

From Table 3.6, we can gather that the requirements for Fats, Saturated Fats and Protein were the most difficult to fulfill in our diet-plan solution. We may also gather that Calories, Carbohydrates, Sugar and Salt were all abundant and thus did not impose on our solution's feasibility nearly as much. The requirement for Fibre has fallen in the middle, having a reasonable surplus but not being nearly as abundant as other requirements.

This is an interesting result because, given the shape of Figure 2.1, one would have expected the constraints for calories and carbohydrates to have been far more restrictive on our solution due to their being significantly larger than other constraints by a large margin. From this we may infer that, though consuming sufficiently many calories and carbohydrates is important, calories and carbohydrates are abundant in the foods available, and as such are not difficult requirements to fulfill. Moreover, the cost of edible calories and carbohydrates in our data-set must have been proportionally lower than edible amounts of the other dietary requirements, as we are seeking to minimise the total cost.

As a result, we may claim that the increased requirements for Fats, Saturated Fats and Protein are centrally responsible for the vastly increased total cost of the Olympian diet-plan for both sexes.

# Chapter 4

# Conclusion

## 4.1 Realistic Livability

The one week diet-plans we have constructed are certainly much more realistically livable than the Stigler diet. In particular, it is formed entirely of edible foods as opposed to raw ingredients. Moreover, we can be sure that the strict dietary requirements are certainly met by our diet. In this sense, we have succeeded in solving our more modern formulation of the Diet Problem.

Despite this, the diet plans do not appear to be any more or less realistically livable than other conventional "fad diets", on account of containing similarly few foods to select from. It seems unreasonable to expect even a very disciplined athlete to follow a diet of so few foods for such an extended period of time as people tend towards "using more realistic and practical methods" when dieting (*Fad Diets and Obesity — Part IV: Low-Carbohydrate vs. Low-Fat Diets*, 2005).

## 4.2 Expanding upon and improving our model

To improve our model, we may implement a "variety" constraint, which would prevent foods from appearing within our diet-plan if they appeared in the previous days' diet-plan. This would make our diet-plan more realistically livable by providing a wider selection of foods in the final solution, which would in turn make the diet easier to follow.

We may make additional findings by using a time-frame beyond one week, extending to several years. We may additionally use our existing training day modelling to allow for a varying week-to-week training regiment within our extended time-frame. This would enable a more in-depth modelling of an athlete's life-style, as we could then account for dietary changes across the seasons and the presence of non-training holidays.

As a final extension, we may consider the training regiments of professional bodybuilders as opposed to those of Olympic athletes. This is due to the regiments of bodybuilders generally favouring a strategy of two phases: the off-season "bulking" phase and on-season "cutting" phase (*Macronutrient Considerations for the Sport of Bodybuilding*, 2004). This would lead to interesting modelling constraints as the dietary requirements would vary not only between training and rest days for a given week, but between off-season and on-season weeks within a given year.

## 4.3 Conclusions about the diets of Olympic Athletes

It is clear from our investigation that not only do Olympic athletes intake far more than non-Olympians, but that the cost of an Olympian's diet would need to be many times that of a non-Olympian's to accommodate their vastly higher requirements. We centrally conclude that this cost is incurred in particular due to the vastly increased demand for Fats, Saturated Fats and Protein within an Olympian's diet. Moreover, we have successfully computed a series of livable, though frugal and tiresome, diet plans which minimise the cost of feeding one person subject to more advanced, modern daily recommendations for one's intake. In that sense, we have solved a new version of the diet problem to proven optimality, by utilising the techniques constructed by Stigler and Dantzig in the early 20$^{\text{th}}$ century.

# Bibliography

*A collection of Pyomo examples* (2020). [Accessed March 21st, 2021].
**URL:** *https://github.com/Pyomo/PyomoGallery*

Baur, J. (2011), 'The truth about sugar for athletes'. [Accessed March 21st, 2021].
**URL:** *https://www.stack.com/a/the-truth-about-sugar-for-athletes#:~:text=Staying%20within%20the%2040%2Dto,increases%20their%20daily%20sugar%20intake.*

Dantzig, G. B. (1990), 'The diet problem', *Interfaces* **20**(4), 43–47. [Accessed January 2nd, 2021].
**URL:** *http://www.jstor.org/stable/25061369*

Dantzig, G. B. (1991), *Linear Programming and Extensions*, Princeton University Press. [Accessed January 2nd, 2021].
**URL:** *http://www.jstor.org/stable/j.ctt1cx3tvg*

*Eat This Much* (2021). [Accessed March 21st, 2021].
**URL:** *https://www.eatthismuch.com*

*Fad Diets and Obesity — Part IV: Low-Carbohydrate vs. Low-Fat Diets* (2005), *Urologic Nursing Journal* . [Accessed May 10th, 2021].
**URL:** *https://www.suna.org/download/members/unjarticles/2005/05feb/67.pdf*

*Government Dietary Recommendations* (2016). [Accessed March 21st, 2021].
**URL:** *https://www.gov.uk/government/publications/the-eatwell-guide*

Grandjean, A. C. (1997), 'Diets of Elite Athletes: Has the Discipline of Sports Nutrition Made an Impact?', *The Journal of Nutrition* **127**(5), 874S–877S. [Accessed January 10th, 2021].
**URL:** *https://doi.org/10.1093/jn/127.5.874S*

Grivetti, L. E. and Applegate, E. A. (1997), 'From Olympia to Atlanta: A Cultural-Historical Perspective on Diet and Athletic Training', *The Journal*

*of Nutrition* **127**(5), 860S–868S. [Accessed January 10th, 2021].
**URL:** *https://doi.org/10.1093/jn/127.5.860S*

*How many calories are in one gram of fat, carbohydrate, or protein?* (2016). [Accessed March 21st, 2021].
**URL:** *https://www.nal.usda.gov/fnic/how-many-calories-are-one-gram-fat-carbohydrate-or-protein#:~:text=Nutrition%20Education%20Materials-,How%20many%20calories%20are%20in%20one%20gram%20of%20fat%2C%20carbohydrate,Facts%20Panel%20on%20food%20packages.*

Kasparek, L. (2018), 'Sodium 101 for athletes'. [Accessed March 21st, 2021].
**URL:** *https://news.sanfordhealth.org/healthy-living/sodium-101-for-athletes/*

*Macronutrient Considerations for the Sport of Bodybuilding* (2004), *Sports Medicine* . [Accessed May 10th, 2021].
**URL:** *https://link.springer.com/article/10.2165/00007256-200434050-00004*

Pelly, F. E., O'Connor, H. T., Denyer, G. S. and Caterson, I. D. (2011), 'Evolution of food provision to athletes at the summer Olympic Games', *Nutrition Reviews* **69**(6), 321–332. [Accessed January 10th, 2021].
**URL:** *https://doi.org/10.1111/j.1753-4887.2011.00396.x*

*Reference Intakes Explained* (2020). [Accessed January 12th, 2021].
**URL:** *https://www.nhs.uk/live-well/eat-well/what-are-reference-intakes-on-food-labels/*

Stigler, G. J. (1945), 'The cost of subsistence', *Journal of Farm Economics* **27**(2), 303–314. [Accessed January 2nd, 2021].
**URL:** *https://www.jstor.org/stable/1231810?seq=1*

*The Diet Problem* (2020). [Accessed January 14th, 2021].
**URL:** *https://neos-guide.org/content/diet-problem*

Thomas, T. (n.d.), 'Fiber intake guidelines for endurance athletes'. [Accessed March 21st, 2021].
**URL:** *https://www.trainingpeaks.com/blog/fiber-intake-guidelines-for-endurance-athletes/#:~:text=Fiber%20for%20Athletes,ingest%20on%20a%20daily%20basis.*

*Vitamins and Minerals - Vitamin C* (2020). [Accessed May 12th, 2021].
**URL:** *https://www.nhs.uk/conditions/vitamins-and-minerals/vitamin-c/*

*Which? reveals the cheapest supermarket of the year* (2021). [Accessed March 21st, 2021].
**URL:** *https://www.which.co.uk/news/2021/01/which-reveals-the-cheapest-supermarket-of-the-year/*

# Appendix A

# Additional Calculations

## A.1 Finding Olympic Dietary Requirements

We source our data for the dietary requirements of Olympic athletes various sources, centrally from Table 2 of (Grandjean, 1997).

For the male average, we aggregate across the intakes of teams from the 1952 Helsinki Games. In particular: China, Finland and The United States. To this end, we will make an unweighted mean average of the given values. We will also be separating the values by sex, where our given sources take their averages across both sexes.

The data for the intake requirements give us explicit values for the number of calories (kcal) within their intake. We are also given the percentages of which these calories are given in protein, fat and carbohydrates. From here, we may find explicit values for all four categories by converting with the following formulas:

- Protein (g) $= \frac{\text{Calories in Protein}}{4}$
- Fats (g) $= \frac{\text{Calories in Fats}}{9}$
- Carbohydrates (g) $= \frac{\text{Calories in Carbohydrates}}{4}$

The above is sourced from (*How many calories are in one gram of fat, carbohydrate, or protein?*, 2016).

With the above information, we obtain the following values:

Table A.1: Male Olympian Dietary Requirements

| Requirement | Mean Average | Resultant Total |
|---|---|---|
| Calories | $\frac{14586+12688+19347}{3} = 15540$ | 15540 |
| Protein | $\frac{20+15+15}{3} = 16.6\%$ | $\frac{15540\times0.166}{4} = 644.91$ |
| Fat | $\frac{43+30.5+36}{3} = 36.4\%$ | $\frac{15540\times0.364}{9} = 628.51$ |
| Carbohydrates | $\frac{38+55+48}{3} = 47.0\%$ | $\frac{15540\times0.470}{4} = 1825.95$ |

We also know to maintain the ratio of fats to saturated-fats from the standard RDA guidelines.

$$\implies \text{Saturated Fats} = \frac{31}{97} \times 628.51 = 200.86$$

For the female average, we aggregate across the intakes of teams from the 1952 Helsinki Games. In particular: China, Czekoslovakia and The United States. To this end, we proceed as before.

Table A.2: Female Olympian Dietary Requirements

| Requirement | Mean Average | Resultant Total |
|---|---|---|
| Calories | $\frac{14420+8157+10005}{3} = 10861$ | 10861 |
| Protein | $\frac{21+15+14.5}{3} = 16.5\%$ | $\frac{10861\times0.165}{4} = 448.02$ |
| Fat | $\frac{45.5+46+34}{3} = 41.5\%$ | $\frac{10861\times0.415}{9} = 500.81$ |
| Carbohydrates | $\frac{38.5+39+49}{3} = 42.0\%$ | $\frac{10861\times0.420}{4} = 1140.41$ |

$$\implies \text{Saturated Fats} = \frac{24}{78} \times 500.81 = 154.10$$

We also have sourced information from (Baur, 2011) to know that the sugar intake for an Olympic athlete of either sex should be between 40 and 60 grams per day.

The recommended daily intake for salt in athletes remains heavily disputed (Kasparek, 2018), as such we will choose to match the non-Olympic RDA for these requirements for both sexes.

The recommended daily intake for fibre athletes is given in (Thomas, n.d.) as between 20 and 35 grams, however, more recent guidance from the UK government given in (*Government Dietary Recommendations*, 2016) would indicate that ordinary people would also require roughly this amount of fibre. As a result, though Olympic athletes will be modelled as requiring more fibre than a non-Olympian, this increase will be less significant in our formulation.

## A.2 Calculating Proportions of Cost Contribution

Table A.3: Cost Proportions of Male Training Day Diet

| | Food | Number of Portions | Cost per Portion | Cost in Diet | Proportion |
|---|---|---|---|---|---|
| Training Day Male | Ice-Cream | 16 | 0.1341 | 2.1456 | 42.77512 |
| | Nuts | 26 | 0.0652 | 1.6952 | 33.79585 |
| | Spaghetti | 104 | 0.0113 | 1.1752 | 23.42903 |
| | Bread-Loaved | 0 | 0.0129 | 0 | 0 |
| | Milk-Dairy | 0 | 0.12 | 0 | 0 |
| Total | | | | 5.016 | 100 |

Table A.4: Cost Proportions of Female Training Day Diet

| | Food | Number of Portions | Cost per Portion | Cost in Diet | Proportion |
|---|---|---|---|---|---|
| Training Day Female | Ice-Cream | 12 | 0.1341 | 1.6092 | 42.59284 |
| | Nuts | 22 | 0.0652 | 1.4344 | 37.96617 |
| | Spaghetti | 65 | 0.0113 | 0.7345 | 19.44099 |
| | Bread-Loaved | 0 | 0.0129 | 0 | 0 |
| | Sugar | 0 | 0.0026 | 0 | 0 |
| Total | | | | 3.7781 | 100 |

Table A.5: Cost Proportions of Male Rest Day Diet

| | Food | Number of Portions | Cost per Portion | Cost in Diet | Proportion |
|---|---|---|---|---|---|
| Rest Day Male | Bread-Loaved | 4 | 0.0129 | 0.0516 | 6.342183 |
| | Ice-Cream | 2 | 0.1341 | 0.2682 | 32.9646 |
| | Milk-Dairy | 1 | 0.12 | 0.12 | 14.74926 |
| | Nuts | 4 | 0.0652 | 0.2608 | 32.05506 |
| | Spaghetti | 10 | 0.0113 | 0.113 | 13.88889 |
| Total | | | | 0.8136 | 100 |

Table A.6: Cost Proportions of Female Rest Day Diet

| | Food | Number of Portions | Cost per Portion | Cost in Diet | Proportion |
|---|---|---|---|---|---|
| Rest Day Female | Bread-Loaves | 6 | 0.0129 | 0.0774 | 11.34897 |
| | Ice-Cream | 2 | 0.1341 | 0.2682 | 39.32551 |
| | Nuts | 4 | 0.0652 | 0.2608 | 38.24047 |
| | Spaghetti | 6 | 0.0113 | 0.0678 | 9.941349 |
| | Sugar | 3 | 0.0026 | 0.0078 | 1.143695 |
| Total | | | | 0.682 | 100 |

# Appendix B

# Command Line Output

## B.1 Pyomo Output for Olympic Male Diet Plan

The following command line output may be obtained by running the command:

```
pyomo solve --solver=glpk <model code> <data file>
```

```
# ==========================================================
# = Solver Results                                         =
# ==========================================================
# ----------------------------------------------------------
#   Problem Information
# ----------------------------------------------------------
Problem:
- Name: unknown
  Lower bound: 30.9336005882965
  Upper bound: 30.9336005882965
  Number of objectives: 1
  Number of constraints: 57
  Number of variables: 295
  Number of nonzeros: 1779
  Sense: minimize
# ----------------------------------------------------------
#   Solver Information
# ----------------------------------------------------------
Solver:
- Status: ok
```

```
  Termination condition: optimal
  Statistics:
    Branch and bound:
      Number of bounded subproblems: 12353
      Number of created subproblems: 12353
  Error rc: 0
  Time: 7.4002485275268555
# ----------------------------------------------------------
#   Solution Information
# ----------------------------------------------------------
Solution:
- number of solutions: 1
  number of solutions displayed: 1
- Gap: 0.0
  Status: optimal
  Message: None
  Objective:
    cost:
      Value: 30.93360058829651
  Variable:
    x[Friday,Ice-Cream]:
      Value: 16
    x[Friday,Nuts]:
      Value: 26
    x[Friday,Spaghetti]:
      Value: 104
    x[Monday,Ice-Cream]:
      Value: 16
    x[Monday,Nuts]:
      Value: 26
    x[Monday,Spaghetti]:
      Value: 104
    x[Saturday,Ice-Cream]:
      Value: 16
    x[Saturday,Nuts]:
      Value: 26
    x[Saturday,Spaghetti]:
      Value: 104
    x[Sunday,Bread-Loaves]:
      Value: 4
    x[Sunday,Ice-Cream]:
      Value: 2
    x[Sunday,Milk-Dairy]:
      Value: 1
    x[Sunday,Nuts]:
      Value: 4
```

```
  x[Sunday,Spaghetti]:
    Value: 10
  x[Thursday,Ice-Cream]:
    Value: 16
  x[Thursday,Nuts]:
    Value: 26
  x[Thursday,Spaghetti]:
    Value: 104
  x[Tuesday,Ice-Cream]:
    Value: 16
  x[Tuesday,Nuts]:
    Value: 26
  x[Tuesday,Spaghetti]:
    Value: 104
  x[Wednesday,Ice-Cream]:
    Value: 16
  x[Wednesday,Nuts]:
    Value: 26
  x[Wednesday,Spaghetti]:
    Value: 104
Constraint: No values
```

## B.2 Pyomo Output for Olympic Female Diet Plan

The following command line output may be obtained by running the command:

```
pyomo solve --solver=glpk <model code> <data file>
```

```
# ==========================================================
# = Solver Results                                         =
# ==========================================================
# ----------------------------------------------------------
#   Problem Information
# ----------------------------------------------------------
Problem:
- Name: unknown
  Lower bound: 23.3653667716535
  Upper bound: 23.3653667716535
  Number of objectives: 1
  Number of constraints: 57
  Number of variables: 295
  Number of nonzeros: 1779
  Sense: minimize
# ----------------------------------------------------------
#   Solver Information
# ----------------------------------------------------------
Solver:
- Status: ok
  Termination condition: optimal
  Statistics:
    Branch and bound:
      Number of bounded subproblems: 2213
      Number of created subproblems: 2213
  Error rc: 0
  Time: 2.346888542175293
# ----------------------------------------------------------
#   Solution Information
# ----------------------------------------------------------
Solution:
- number of solutions: 1
  number of solutions displayed: 1
- Gap: 0.0
  Status: optimal
  Message: None
  Objective:
    cost:
```

```
    Value: 23.365366771653544
Variable:
  x[Friday,Ice-Cream]:
    Value: 12
  x[Friday,Nuts]:
    Value: 22
  x[Friday,Spaghetti]:
    Value: 65
  x[Monday,Ice-Cream]:
    Value: 12
  x[Monday,Nuts]:
    Value: 22
  x[Monday,Spaghetti]:
    Value: 65
  x[Saturday,Ice-Cream]:
    Value: 12
  x[Saturday,Nuts]:
    Value: 22
  x[Saturday,Spaghetti]:
    Value: 65
  x[Sunday,Bread-Loaves]:
    Value: 6
  x[Sunday,Ice-Cream]:
    Value: 2
  x[Sunday,Nuts]:
    Value: 4
  x[Sunday,Spaghetti]:
    Value: 6
  x[Sunday,Sugar]:
    Value: 3
  x[Thursday,Ice-Cream]:
    Value: 12
  x[Thursday,Nuts]:
    Value: 22
  x[Thursday,Spaghetti]:
    Value: 65
  x[Tuesday,Ice-Cream]:
    Value: 12
  x[Tuesday,Nuts]:
    Value: 22
  x[Tuesday,Spaghetti]:
    Value: 65
  x[Wednesday,Ice-Cream]:
    Value: 12
  x[Wednesday,Nuts]:
    Value: 22
```

```
  x[Wednesday,Spaghetti]:
    Value: 65
Constraint: No values
```

## B.3 Pyomo Output for Non-Olympic Male Diet Plan

The following command line output may be obtained by running the command:

```
pyomo solve --solver=glpk <model code> <data file>
```

```
# ==========================================================
# = Solver Results                                         =
# ==========================================================
# ----------------------------------------------------------
#   Problem Information
# ----------------------------------------------------------
Problem:
- Name: unknown
  Lower bound: 5.69731073224878
  Upper bound: 5.69731073224878
  Number of objectives: 1
  Number of constraints: 9
  Number of variables: 43
  Number of nonzeros: 255
  Sense: minimize
# ----------------------------------------------------------
#   Solver Information
# ----------------------------------------------------------
Solver:
- Status: ok
  Termination condition: optimal
  Statistics:
    Branch and bound:
      Number of bounded subproblems: 9
      Number of created subproblems: 9
  Error rc: 0
  Time: 0.02491307258605957
# ----------------------------------------------------------
#   Solution Information
# ----------------------------------------------------------
Solution:
- number of solutions: 1
  number of solutions displayed: 1
- Gap: 0.0
  Status: optimal
  Message: None
  Objective:
```

```
  cost:
    Value: 5.697310732248778
Variable:
  x[Bread-Loaves]:
    Value: 4
  x[Ice-Cream]:
    Value: 2
  x[Milk-Dairy]:
    Value: 1
  x[Nuts]:
    Value: 4
  x[Spaghetti]:
    Value: 10
Constraint: No values
```

## B.4 Pyomo Output for Non-Olympic Female Diet Plan

The following command line output may be obtained by running the command:

```
pyomo solve --solver=glpk <model code> <data file>
```

```
# ==========================================================
# = Solver Results                                         =
# ==========================================================
# ----------------------------------------------------------
#   Problem Information
# ----------------------------------------------------------
Problem:
- Name: unknown
  Lower bound: 4.77426236220472
  Upper bound: 4.77426236220472
  Number of objectives: 1
  Number of constraints: 9
  Number of variables: 43
  Number of nonzeros: 255
  Sense: minimize
# ----------------------------------------------------------
#   Solver Information
# ----------------------------------------------------------
Solver:
- Status: ok
  Termination condition: optimal
  Statistics:
    Branch and bound:
      Number of bounded subproblems: 11
      Number of created subproblems: 11
  Error rc: 0
  Time: 0.025993585586547855
# ----------------------------------------------------------
#   Solution Information
# ----------------------------------------------------------
Solution:
- number of solutions: 1
  number of solutions displayed: 1
- Gap: 0.0
  Status: optimal
  Message: None
  Objective:
```

```
  cost:
    Value: 4.774262362204725
Variable:
  x[Bread-Loaves]:
    Value: 6
  x[Ice-Cream]:
    Value: 2
  x[Nuts]:
    Value: 4
  x[Spaghetti]:
    Value: 6
  x[Sugar]:
    Value: 3
Constraint: No values
```

# Appendix C

# Python Code

## C.1 Pyomo Code for Olympic Diet Model

```
# %load model_split_nutrients.py

"""
Created on Wed Mar 31 18:49:19 2021

@author: Emma Tarmey
"""

import logging
import pandas as pd
import numpy as np
from pyomo.environ import *


model = AbstractModel()
logging.getLogger('pyomo.core').setLevel(logging.ERROR)


# Food and Nutrient data sets
model.Foods     = Set()
model.Nutrients = Set()
model.Days      = Set()


# Model parameters
```

```
model.costs               = Param(model.Foods, within = PositiveReals)

model.training            = Param(model.Days, within = Boolean)

model.amountCalories      = Param(model.Foods, within = NonNegativeReals)
model.amountFat           = Param(model.Foods, within = NonNegativeReals)
model.amountSaturates     = Param(model.Foods, within = NonNegativeReals)
model.amountCarbohydrates = Param(model.Foods, within = NonNegativeReals)
model.amountSugar         = Param(model.Foods, within = NonNegativeReals)
model.amountProtein       = Param(model.Foods, within = NonNegativeReals)
model.amountSalt          = Param(model.Foods, within = NonNegativeReals)
model.amountFibre         = Param(model.Foods, within = NonNegativeReals)

 # Minimum nutrient requirements when not training
model.minCalories      = Param(within = NonNegativeReals, default = 0.0)
model.minFat           = Param(within = NonNegativeReals, default = 0.0)
model.minSaturates     = Param(within = NonNegativeReals, default = 0.0)
model.minCarbohydrates = Param(within = NonNegativeReals, default = 0.0)
model.minSugar         = Param(within = NonNegativeReals, default = 0.0)
model.minProtein       = Param(within = NonNegativeReals, default = 0.0)
model.minSalt          = Param(within = NonNegativeReals, default = 0.0)
model.minFibre         = Param(within = NonNegativeReals, default = 0.0)

 # Minimum nutrient requirements when training
model.minCaloriesTrain      = Param(within = NonNegativeReals, \
                                    default = 0.0)
model.minFatTrain           = Param(within = NonNegativeReals, \
                                    default = 0.0)
model.minSaturatesTrain     = Param(within = NonNegativeReals, \
                                    default = 0.0)
model.minCarbohydratesTrain = Param(within = NonNegativeReals, \
                                    default = 0.0)
model.minSugarTrain         = Param(within = NonNegativeReals, \
                                    default = 0.0)
model.minProteinTrain       = Param(within = NonNegativeReals, \
                                    default = 0.0)
model.minSaltTrain          = Param(within = NonNegativeReals, \
                                    default = 0.0)
model.minFibreTrain         = Param(within = NonNegativeReals, \
                                    default = 0.0)

 # Maximum nutrient requirements when not training
model.maxCalories      = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxFat           = Param(within = NonNegativeReals, \
                               default = float("inf"))
```

```
model.maxSaturates     = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxCarbohydrates = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxSugar         = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxProtein       = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxSalt          = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxFibre         = Param(within = NonNegativeReals, \
                               default = float("inf"))

 # Maximum nutrient requirements when training
model.maxCaloriesTrain      = Param(within = NonNegativeReals, \
                                    default = float("inf"))
model.maxFatTrain           = Param(within = NonNegativeReals, \
                                    default = float("inf"))
model.maxSaturatesTrain     = Param(within = NonNegativeReals, \
                                    default = float("inf"))
model.maxCarbohydratesTrain = Param(within = NonNegativeReals, \
                                    default = float("inf"))
model.maxSugarTrain         = Param(within = NonNegativeReals, \
                                    default = float("inf"))
model.maxProteinTrain       = Param(within = NonNegativeReals, \
                                    default = float("inf"))
model.maxSaltTrain          = Param(within = NonNegativeReals, \
                                    default = float("inf"))
model.maxFibreTrain         = Param(within = NonNegativeReals, \
                                    default = float("inf"))


# Model Variable
# Measures the amount of a given food to eat on a given day
model.x = Var(model.Days, model.Foods, within = NonNegativeIntegers)


# Objective function
# the model seeks to minimise the value of this function
def objective_function(model):
    total = sum( sum( (model.costs[i] * model.x[d, i]) \
                     for i in model.Foods) for d in model.Days )
    return total


# Nutrient constraints
```

```python
def calories_constraint(model, d):
    # If today is a training day, raise our requirements appropriately
    if ( model.training[d] ):
        minimum = model.minCaloriesTrain
        maximum = model.maxCaloriesTrain
    else:
        minimum = model.minCalories
        maximum = model.maxCalories

    total = sum(model.amountCalories[i] * model.x[d, i] \
                for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def fat_constraint(model, d):
    # If today is a training day, raise our requirements appropriately
    if ( model.training[d] ):
        minimum = model.minFatTrain
        maximum = model.maxFatTrain
    else:
        minimum = model.minFat
        maximum = model.maxFat

    total = sum(model.amountFat[i] * model.x[d, i] \
                for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def saturates_constraint(model, d):
    # If today is a training day, raise our requirements appropriately
    if ( model.training[d] ):
        minimum = model.minSaturatesTrain
        maximum = model.maxSaturatesTrain
    else:
        minimum   = model.minSaturates
        maximum = model.maxSaturates

    total = sum(model.amountSaturates[i] * model.x[d, i] \
                for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def carbohydrates_constraint(model, d):
    # If today is a training day, raise our requirements appropriately
    if ( model.training[d] ):
        minimum = model.minCarbohydratesTrain
```

```
        maximum = model.maxCarbohydratesTrain
    else:
        minimum = model.minCarbohydrates
        maximum = model.maxCarbohydrates

    total = sum(model.amountCarbohydrates[i] * model.x[d, i] \
                for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def sugar_constraint(model, d):
    # If today is a training day, raise our requirements appropriately
    if ( model.training[d] ):
        minimum = model.minSugarTrain
        maximum = model.maxSugarTrain
    else:
        minimum = model.minSugar
        maximum = model.maxSugar

    total = sum(model.amountSugar[i] * model.x[d, i] \
                for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def protein_constraint(model, d):
    # If today is a training day, raise our requirements appropriately
    if ( model.training[d] ):
        minimum = model.minProteinTrain
        maximum = model.maxProteinTrain
    else:
        minimum = model.minProtein
        maximum = model.maxProtein

    total = sum(model.amountProtein[i] * model.x[d, i] \
                for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def salt_constraint(model, d):
    # If today is a training day, raise our requirements appropriately
    if ( model.training[d] ):
        minimum = model.minSaltTrain
        maximum = model.maxSaltTrain
    else:
        minimum = model.minSalt
        maximum = model.maxSaltTrain
```

```python
    total = sum(model.amountSalt[i] * model.x[d, i] \
                for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def fibre_constraint(model, d):
    # If today is a training day, raise our requirements appropriately
    if ( model.training[d] ):
        minimum = model.minFibreTrain
        maximum = model.maxFibreTrain
    else:
        minimum = model.minFibre
        maximum = model.maxFibre

    total = sum(model.amountFibre[i] * model.x[d, i] \
                for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied


# Attach all above methods to the pyomo model object
model.cost = Objective(rule = objective_function)

model.calories      = Constraint(model.Days, rule = calories_constraint)
model.fat           = Constraint(model.Days, rule = fat_constraint)
model.saturates     = Constraint(model.Days, \
                                 rule = saturates_constraint)
model.carbohydrates = Constraint(model.Days, \
                                 rule = carbohydrates_constraint)

model.sugar         = Constraint(model.Days, rule = sugar_constraint)
model.protein       = Constraint(model.Days, rule = protein_constraint)
model.salt          = Constraint(model.Days, rule = salt_constraint)
model.fibre         = Constraint(model.Days, rule = fibre_constraint)
```

## C.2 Pyomo Code for Non-Olympic Diet Model

```
# %load model_split_nutrients_non-olympian.py

"""
Created on Wed Mar 31 18:49:19 2021

@author: Emma Tarmey
"""

import logging
import pandas as pd
import numpy as np
from pyomo.environ import *


model = AbstractModel()
logging.getLogger('pyomo.core').setLevel(logging.ERROR)


# Food and Nutrient data sets
model.Foods     = Set()
model.Nutrients = Set()
model.Days      = Set()


# Model parameters
model.costs                = Param(model.Foods, within = PositiveReals)

model.amountCalories       = Param(model.Foods, within = NonNegativeReals)
model.amountFat            = Param(model.Foods, within = NonNegativeReals)
model.amountSaturates      = Param(model.Foods, within = NonNegativeReals)
model.amountCarbohydrates = Param(model.Foods, within = NonNegativeReals)
model.amountSugar          = Param(model.Foods, within = NonNegativeReals)
model.amountProtein        = Param(model.Foods, within = NonNegativeReals)
model.amountSalt           = Param(model.Foods, within = NonNegativeReals)
model.amountFibre          = Param(model.Foods, within = NonNegativeReals)

 # Minimum nutrient requirements
model.minCalories      = Param(within = NonNegativeReals, default = 0.0)
model.minFat           = Param(within = NonNegativeReals, default = 0.0)
model.minSaturates     = Param(within = NonNegativeReals, default = 0.0)
model.minCarbohydrates = Param(within = NonNegativeReals, default = 0.0)
model.minSugar         = Param(within = NonNegativeReals, default = 0.0)
model.minProtein       = Param(within = NonNegativeReals, default = 0.0)
```

```
model.minSalt          = Param(within = NonNegativeReals, default = 0.0)
model.minFibre         = Param(within = NonNegativeReals, default = 0.0)

 # Minimum nutrient requirements
model.maxCalories      = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxFat           = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxSaturates     = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxCarbohydrates = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxSugar         = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxProtein       = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxSalt          = Param(within = NonNegativeReals, \
                               default = float("inf"))
model.maxFibre         = Param(within = NonNegativeReals, \
                               default = float("inf"))


# Model Variables
# Measures the amount of a given food to eat on a given day
model.x = Var(model.Foods, within = NonNegativeIntegers)


# Objective function
# The model seeks to minimise the value of this function
def objective_function(model):
    return (len(model.Days) * sum( (model.costs[i] * model.x[i]) \
                                  for i in model.Foods ) )


# Nutrient constraints
def calories_constraint(model):
    minimum   = model.minCalories
    maximum   = model.maxCalories
    total = sum(model.amountCalories[i] * model.x[i] for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def fat_constraint(model):
    minimum   = model.minFat
    maximum   = model.maxFat
    total = sum(model.amountFat[i] * model.x[i] for i in model.Foods)
```

```
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def saturates_constraint(model):
    minimum   = model.minSaturates
    maximum   = model.maxSaturates
    total = sum(model.amountSaturates[i] * model.x[i] for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def carbohydrates_constraint(model):
    minimum   = model.minCarbohydrates
    maximum   = model.maxCarbohydrates
    total = sum(model.amountCarbohydrates[i] * model.x[i] for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def sugar_constraint(model):
    minimum   = model.minSugar
    maximum   = model.maxSugar
    total = sum(model.amountSugar[i] * model.x[i] for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def protein_constraint(model):
    minimum   = model.minProtein
    maximum   = model.maxProtein
    total = sum(model.amountProtein[i] * model.x[i] for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def salt_constraint(model):
    minimum   = model.minSalt
    maximum   = model.maxSalt
    total = sum(model.amountSalt[i] * model.x[i] for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied

def fibre_constraint(model):
    minimum   = model.minFibre
    maximum   = model.maxFibre
    total = sum(model.amountFibre[i] * model.x[i] for i in model.Foods)
    satisfied = (total >= minimum) and (total <= maximum)
    return satisfied
```

```python
# Attach all above methods to the pyomo model object
model.cost = Objective(rule = objective_function)

model.calories      = Constraint(rule = calories_constraint)
model.fat           = Constraint(rule = fat_constraint)
model.saturates     = Constraint(rule = saturates_constraint)
model.carbohydrates = Constraint(rule = carbohydrates_constraint)

model.sugar         = Constraint(rule = sugar_constraint)
model.protein       = Constraint(rule = protein_constraint)
model.salt          = Constraint(rule = salt_constraint)
model.fibre         = Constraint(rule = fibre_constraint)
```

## C.3 Pyomo Data File for Olympic Female Diet

The given command line output may be obtained by running the command:

```
pyomo solve --solver=glpk <model code> <data file>
```

Notably, the data files for each of the four categories of person are very similar. By editing the appropriate constraints within the data file and using the appropriate model code, you may obtain the command line output for each other category similarly. To avoid clutter, only the data file for the female Olympian diet is given.

```
data;


set Foods :=
Anchovies
Apples
Asparagus
Beef
Baguettes
Beans
Beets
Bread-Loaves
Bread-Rolls
Cabbage
Cauliflower
Cereal
Cheese
Eggs
Garlic
Glucose
Grape-Nuts
Honey
Ice-Cream
Lemons
Lettuce
Nuts
Onions
Parsley
Pasta
Peaches
Peas
```

```
Pizza
Potatoes
Poultry
Rice
Spaghetti
Spinach
Strawberries
Sugar
Coffee
Milk-Dairy
Milk-Soy
Soft-Drinks
Water
Whiskey
Wine
;


set Nutrients :=
Calories
Fat
Saturates
Carbohydrates
Sugar
Protein
Salt
Fibre
;


set Days :=
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
;


param training :=
Monday True
Tuesday True
Wednesday True
Thursday True
```

```
Friday True
Saturday True
Sunday False
;


param costs :=
Anchovies 0.0392
Apples 0.10800000000000001
Asparagus 0.10815999999999999
Beef 0.18022500000000002
Baguettes 0.21
Beans 0.13252380952380952
Beets 0.08036
Bread-Loaves 0.012870000000000001
Bread-Rolls 0.075
Cabbage 0.48805
Cauliflower 0.31898113207547174
Cereal 0.028
Cheese 0.1686825
Eggs 0.07866666666666666
Garlic 0.22999999999999998
Glucose 0.08900000000000001
Grape-Nuts 0.5908620689655173
Honey 0.0525
Ice-Cream 0.1340787401574803
Lemons 0.1725
Lettuce 0.04777777777777777
Nuts 0.06520500000000001
Onions 0.15666666666666665
Parsley 0.07219999999999999
Pasta 0.016443
Peaches 0.3225
Peas 0.08125689084895259
Pizza 0.21337579617834396
Potatoes 0.18081
Poultry 0.6647333333333333
Rice 0.15725
Spaghetti 0.011340000000000001
Spinach 0.11933333333333333
Strawberries 0.40031999999999995
Sugar 0.0026000000000000003
Coffee 0.020849999999999997
Milk-Dairy 0.12004405286343613
Milk-Soy 0.1375
Soft-Drinks 0.0425
```

```
Water 0.02125
Whiskey 0.693642857142857
Wine 0.931
;


param amountCalories :=
Anchovies 8.4
Apples 95.0
Asparagus 3.2
Beef 78.8
Baguettes 335.0
Beans 265.7
Beets 35.3
Bread-Loaves 35.3
Bread-Rolls 161.0
Cabbage 227.0
Cauliflower 26.8
Cereal 100.0
Cheese 115.1
Eggs 71.5
Garlic 4.5
Glucose 15.0
Grape-Nuts 419.7
Honey 63.8
Ice-Cream 273.2
Lemons 24.4
Lettuce 5.4
Nuts 163.9
Onions 44.0
Parsley 1.4
Pasta 37.1
Peaches 58.5
Peas 103.2
Pizza 250.0
Potatoes 284.1
Poultry 141.6
Rice 684.5
Spaghetti 105.2
Spinach 6.9
Strawberries 46.1
Sugar 15.5
Coffee 10.6
Milk-Dairy 163.8
Milk-Soy 85.0
Soft-Drinks 10.0
```

```
Water 0.0
Whiskey 104.3
Wine 144.8
;


param amountFat :=
Anchovies 0.4
Apples 0.3
Asparagus 0.0
Beef 6.4
Baguettes 1.2
Beans 1.0
Beets 0.1
Bread-Loaves 0.1
Bread-Rolls 0.8
Cabbage 0.9
Cauliflower 0.3
Cereal 0.1
Cheese 9.6
Eggs 4.8
Garlic 0.0
Glucose 0.0
Grape-Nuts 2.4
Honey 0.0
Ice-Cream 14.5
Lemons 0.3
Lettuce 0.1
Nuts 13.8
Onions 0.1
Parsley 0.0
Pasta 0.3
Peaches 0.4
Peas 0.5
Pizza 7.1
Potatoes 0.3
Poultry 3.1
Rice 5.4
Spaghetti 0.4
Spinach 0.1
Strawberries 0.4
Sugar 0.0
Coffee 0.0
Milk-Dairy 9.4
Milk-Soy 5.3
Soft-Drinks 0.2
```

```
Water 0.0
Whiskey 0.0
Wine 0.0
;


param amountSaturates :=
Anchovies 0.1
Apples 0.1
Asparagus 0.0
Beef 2.6
Baguettes 0.0
Beans 0.3
Beets 0.0
Bread-Loaves 0.0
Bread-Rolls 0.2
Cabbage 0.3
Cauliflower 0.1
Cereal 0.0
Cheese 5.5
Eggs 1.6
Garlic 0.0
Glucose 0.0
Grape-Nuts 0.5
Honey 0.0
Ice-Cream 9.0
Lemons 0.0
Lettuce 0.0
Nuts 1.8
Onions 0.0
Parsley 0.0
Pasta 0.0
Peaches 0.0
Peas 0.1
Pizza 3.3
Potatoes 0.1
Poultry 0.7
Rice 1.1
Spaghetti 0.1
Spinach 0.0
Strawberries 0.0
Sugar 0.0
Coffee 0.0
Milk-Dairy 5.8
Milk-Soy 1.0
Soft-Drinks 0.0
```

```
Water 0.0
Whiskey 0.0
Wine 0.0
;


param amountCarbohydrates :=
Anchovies 0.0
Apples 25.1
Asparagus 0.6
Beef 0.0
Baguettes 68.8
Beans 51.8
Beets 7.8
Bread-Loaves 7.8
Bread-Rolls 32.4
Cabbage 52.7
Cauliflower 5.3
Cereal 23.5
Cheese 0.4
Eggs 0.4
Garlic 1.0
Glucose 0.0
Grape-Nuts 92.6
Honey 17.3
Ice-Cream 31.2
Lemons 7.8
Lettuce 1.0
Nuts 5.6
Onions 10.3
Parsley 0.2
Pasta 7.1
Peaches 14.3
Peas 18.3
Pizza 30.0
Potatoes 64.5
Poultry 0.0
Rice 142.9
Spaghetti 21.2
Spinach 1.1
Strawberries 11.1
Sugar 4.0
Coffee 2.3
Milk-Dairy 11.9
Milk-Soy 1.0
Soft-Drinks 1.5
```

```
Water 0.0
Whiskey 0.0
Wine 3.7
;


param amountSugar :=
Anchovies 0.0
Apples 18.9
Asparagus 0.3
Beef 0.0
Baguettes 0.0
Beans 19.7
Beets 5.5
Bread-Loaves 5.5
Bread-Rolls 1.8
Cabbage 29.1
Cauliflower 2.0
Cereal 2.7
Cheese 0.1
Eggs 0.2
Garlic 0.0
Glucose 3.7
Grape-Nuts 10.2
Honey 17.2
Ice-Cream 28.0
Lemons 2.1
Lettuce 0.3
Nuts 0.0
Onions 4.7
Parsley 0.0
Pasta 0.0
Peaches 12.6
Peas 6.7
Pizza 3.6
Potatoes 2.9
Poultry 0.0
Rice 1.6
Spaghetti 0.8
Spinach 0.1
Strawberries 7.0
Sugar 4.0
Coffee 0.0
Milk-Dairy 0.0
Milk-Soy 0.5
Soft-Drinks 0.0
```

```
Water 0.0
Whiskey 0.0
Wine 3.7
;


param amountProtein :=
Anchovies 1.2
Apples 0.5
Asparagus 0.4
Beef 5.0
Baguettes 10.0
Beans 12.1
Beets 1.3
Bread-Loaves 1.3
Bread-Rolls 6.2
Cabbage 11.6
Cauliflower 2.1
Cereal 2.1
Cheese 6.8
Eggs 6.3
Garlic 0.2
Glucose 0.0
Grape-Nuts 12.5
Honey 0.1
Ice-Cream 4.6
Lemons 0.9
Lettuce 0.5
Nuts 7.3
Onions 1.2
Parsley 0.1
Pasta 1.5
Peaches 1.4
Peas 7.0
Pizza 12.0
Potatoes 7.5
Poultry 26.6
Rice 14.7
Spaghetti 3.7
Spinach 0.9
Strawberries 1.0
Sugar 0.0
Coffee 0.4
Milk-Dairy 8.4
Milk-Soy 8.5
Soft-Drinks 0.6
```

```
Water 0.0
Whiskey 0.0
Wine 0.1
;


param amountSalt :=
Anchovies 146.7
Apples 1.8
Asparagus 0.3
Beef 16.7
Baguettes 0.0
Beans 2.5
Beets 64
Bread-Loaves 64
Bread-Rolls 0.0
Cabbage 163.4
Cauliflower 32.1
Cereal 204.1
Cheese 182.6
Eggs 71.0
Garlic 0.5
Glucose 0.0
Grape-Nuts 545.1
Honey 0.0
Ice-Cream 105.6
Lemons 1.7
Lettuce 10.1
Nuts 122.8
Onions 4.4
Parsley 2.1
Pasta 1.7
Peaches 0.0
Peas 144.
Pizza 0.0
Potatoes 22.1
Poultry 53.1
Rice 13.0
Spaghetti 1.7
Spinach 23.7
Strawberries 1.4
Sugar 0.0
Coffee 1.1
Milk-Dairy 125.4
Milk-Soy 60
Soft-Drinks 40
```

```
Water 0.0
Whiskey 0.0
Wine 0.0
;


param amountFibre :=
Anchovies 0.0
Apples 4.4
Asparagus 0.3
Beef 0.0
Baguettes 2.5
Beans 13.9
Beets 2.3
Bread-Loaves 2.3
Bread-Rolls 1.5
Cabbage 22.7
Cauliflower 2.1
Cereal 0.9
Cheese 0.0
Eggs 0.0
Garlic 0.1
Glucose 0.0
Grape-Nuts 13.9
Honey 0.0
Ice-Cream 0.9
Lemons 2.4
Lettuce 0.5
Nuts 2.5
Onions 1.9
Parsley 0.1
Pasta 0.0
Peaches 2.3
Peas 6.0
Pizza 2.3
Potatoes 8.1
Poultry 0.0
Rice 6.5
Spaghetti 0.9
Spinach 0.7
Strawberries 2.9
Sugar 0.0
Coffee 0.0
Milk-Dairy 0.0
Milk-Soy 0.0
Soft-Drinks 0.0
```

```
Water 0.0
Whiskey 0.0
Wine 0.0
;


param minCaloriesTrain      := 10861 ;
param minFatTrain           := 500.81 ;
param minSaturatesTrain     := 154.1 ;
param minCarbohydratesTrain := 1140.41 ;
param minSugarTrain         := 40.0 ;
param minProteinTrain       := 448.02 ;
param minSaltTrain          := 6.0 ;
param minFibreTrain         := 35.0 ;


param minCalories      := 2000 ;
param minFat           := 78.0 ;
param minSaturates     := 24.0 ;
param minCarbohydrates := 267.0 ;
param minSugar         := 27.0 ;
param minProtein       := 45.0 ;
param minSalt          := 6.0 ;
param minFibre         := 30.0 ;


end;
```