**E-Health Telecommunication Systems and Networks, 2016, 5, 31-38**
Published Online June 2016 in SciRes. http://www.scirp.org/journal/etsn
http://dx.doi.org/10.4236/etsn.2016.52004

# Computational Nutrition: An Algorithm to Generate a Diet Plan to Meet Specific Nutritional Requirements

**Thomas Pikes, Robert Adams**

School of Computing and Engineering, Grand Valley State University,
Allendale, MI, USA
Email: adamsr@gvsu.edu

Received 5 April 2016; accepted 24 May 2016; published 27 May 2016

## Abstract

**Many methods have been proposed to generate meal plans, but most of them only consider proximates. However, the human body requires a combination of proximates and several macronutrients, micronutrients, vitamins, and minerals. Furthermore, the models designed to generate these meal plans do not take into account an individual's specific nutritional needs. These needs are often expressed as a combination of lower bound amount (LBA), ideal amount (IA), and upper bound amount (UBA) necessary for the human body to thrive. The aim of this project is to generate an algorithm to produce a list of food items that will meet specific nutritional requirements. With the proposed algorithm, each nutrient receives a score based on the amount of nutrient contained in the food list in relation to the LBA, IA, and UBA. These scores are aggregated to give the meal plan an overall score.**

## Keywords

**Computational Nutrition, Diet, Preventative, Nutrition Therapy**

## 1. Introduction

The United States has the highest health expenditures as a percentage of Gross Domestic Product (GDP) among developed countries [1]. The US also spends the most of any country on healthcare [1]. In 2006, health care cost for an overweight person was $1429 greater per person than spending for normal weight people. More than one third of all US adults over the age of 19 are obese [2]. This ranks the US among the top ten most obese countries in the world [3]. Given the high rate of obesity and the increased medical spending, reducing the obesity rate in the US will contribute to a significant reduction in health care spending. Adopting a healthy lifestyle can help prevent and reduce obesity [4].

Fortunately, people are becoming more aware of their heath and how simple things like diet and exercise can contribute to an overall healthy life. Furthermore, technology is helping more people become increasingly aware and engaged in maintaining a healthy life. As evidence, there are literally thousands of apps in the Apple App Store related to health and fitness [https://itunes.apple.com/us/genre/ios-health-fitness/id6013?mt=8], not to mention the thousands of website devoted to health. Wearable computing is also revolutionizing how people monitor and maintain their well-being.

Diet is one important aspect of health that people are increasingly taking notice of. There are dozens of mobile apps and website devoted to tracking calorie consumption, meal planning, grocery shopping, etc. Besides maintaining general health, diet is also known to be the "cheapest and most accessible medicine" [5] [6], offering alternatives to pharmaceutical drugs for the prevention and treatment of disorders and diseases. Traditionally, a registered dietitian could be hired to plan meals and food choices to help reach nutritional goals. However, this kind of specialized advice is cost-prohibitive for the average person. What is needed is an inexpensive tool to allow the average health-conscience consumer a way to make positive food choices tailored to a set of nutritional goals. This paper proposes a method that aids a person in creating customized food lists that meet specific nutritional requirements. In addition, the tool could be used as an aid to a registered dietitian in creating a diet plan aimed at treating patients with specific disorders, such as diabetes or food sensitivities.

## 2. Existing Methods

### 2.1. SSC3gd

The SSC3gd model is nutrient profiling method developed to rate foods from less healthy to more healthy. It ranks foods based on energy density, saturated fat, non-milk extrinsic sugars, sodium, calcium, iron, and n-3 polyunsaturated fatty acids [7]. SSC3gd is a very good model for ranking the nutritional value of foods. However, it does not take into account a person's specific nutritional requirements when assigns a food its SSC3gd [7].

### 2.2. EatThisMuch

There are several websites with the purpose of creating a tailored meal plan. These websites offer a fast way of creating a meal plan to meet certain types of diets such as paleo, vegetarian, or ketogenic type diet. However, one major shortcoming of these websites is that they do not take into account the various macronutrients, micronutrients, vitamins, and minerals that are essential to the human body's overall health.

EatThisMuch.com is a feature rich system for creating a daily meal plan. It generates the meal plan based on the number of calories, number of meals, and a range of protein, carbohydrate, and fat targets. It provides a breakdown of the percentage of calories from proteins, carbohydrates, and fats for the generated list of meals. It provides a daily weight tracker to give the user feedback on the effect their diet is having on their weight. However, it does not provide the user with the opportunity to limit or increase the micronutrient makeup of the meals.

### 2.3. Custom Meal Planner

Custom Meal Planner.com is another website dedicated to generating a personal diet plan. It provides a different set of features than EatThisMuch.com. It generates its meals based on a given Basal Metabolic Rate (BMR) Model, in conjunction with a certain diet type, goal, and activeness. It does not appear to take into account specific nutritional requirements that a person may have.

## 3. Proposed Method

We propose a method that generates a customized meal plan to meet personalized nutritional goals. Our daily meal plans are based on the nutrients referenced in **Table 1**. All nutrients under consideration for this project are given a Lower Bound (LB). The LB is the absolute least amount of that nutrient that will be accepted in order

**Table 1.** Nutrient list.

| Nutrient | Unit |
|---|---|
| Calcium | mg |
| Phosphorus | mg |
| Magnesium | mg |
| Potassium | mg |
| Sodium | mg |
| Iron | mg |
| Manganese | mg |
| Copper | mg |
| Zinc | mg |
| Selenium | mcg |
| Vitamin A | IU |
| Vitamin C | mg |
| Vitamin E | mg |
| Vitamin K | mcg |
| Thiamin | mg |
| Roboflavin | mg |
| Niacin | mg |
| Pantothenic Acid | mg |
| Folate | mcg |

for the meal plan to be accepted. As explained later, a default LB is provided for all nutrients, and users are given the ability to customize the LB to account for deficiencies or excesses to provide better health outcomes. For example, a nursing mother will require a different set of nutrients than a pregnant mother. All nutrients are also given an Ideal Amount (IA). The IA is the amount of the nutrient that will result in the meal plan receiving a perfect score for the specific nutrient. It is possible for the LB and IA to be the same. Some nutrients could also have an Upper Bound (UB). The UB is the maximum amount that will be accepted beyond which the meal plan will be deemed unacceptable. The UB provides the user with the ability to limit undesirable qualities of a meal plan. For example, if someone is suffering from high blood pressure the UB of nutrients can be altered to levels that will contribute to the lowering of blood pressure.

To determine if a meal plan meets a given set of nutritional goals, we calculate a numerical score for a meal plan, called the MSCORE. The MSCORE is a summation of each nutrient's score (NSCORE). If we look at two MSCOREs generated with an identical set of nutritional goals, the larger MSCORE correlates to a meal plan that more closely fits the nutritional requirements.

An NSCORE is generated using one of a set of three linear functions. The selection of which of the three functions to use is based on the amount of nutrient currently in the meal plan ($X_{nut}$) in relation to the nutrient's UB, IA, and LB. If $X_{nut}$ is greater than the IA and less than the UB the first function below is used. If $X_{nut}$ is less than the IA and greater than the LB, the second function is used. Finally, if $X_{nut}$ is less than the LB the third function is used. Essentially, the three functions differ in the "weight" and "default score" used. As shown below, our method allows for three different weights and three different default values, allowing users to place a numerical emphasis on the LB, IA, or UB for a given nutrient.

Graphically, the nutritional functions are displayed in **Figure 1**. The X axis is the amount of the nutrient present in the meal plan (in this case in milligrams). The Y axis is the corresponding NSCORE for this nutrient. The weights in the functions correspond to the slope of the line segments. A slope of (1) correlates to a weight of (1), a slope of (2) correlates to a weight of (2), etc. The default scores in the functions correspond to the intercepts

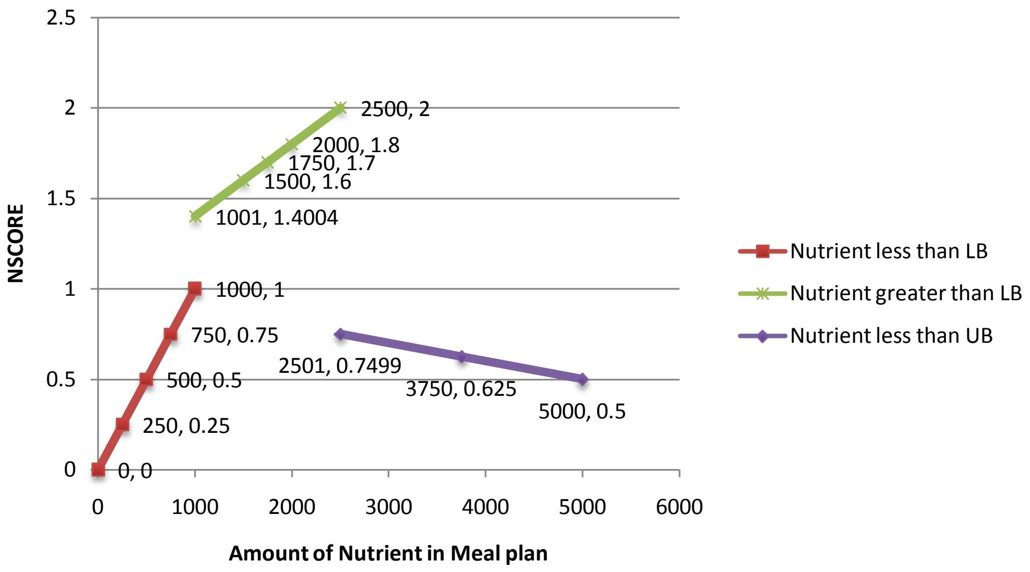

**Figure 1.** A graphical representation of a nutrient's score functions.

of the line segments. The line segment in red is the function that is used if the nutrient amount falls between 0 and LB. Notice that the slope is quite steep, indicating that the user feels it is important to get at least the LB for this nutrient. The green line segment is the function that is used when the nutrient amount falls between LB and IA. Note here that if the meal plan includes the IA (2500 mg), then the maximum NSCORE is generated (2). Finally, the purple segment is the function is used if the nutrient amount exceeds IA but is less than UB. Notice that the further away from IA, the more the NSCORE decreases.

$$\text{NSCORE} = f\left(x_{nut}\right) = \begin{cases} -\left(\text{weight}_{<ub}\left(\dfrac{x_{nut}}{ub}\right)\right) + \text{default}_{<ub}, & ub \geq x_{nut} > ia \\ \text{weight}_{>lb}\left(\dfrac{x_{nut}}{ia}\right) + \text{default}_{>lb}, & ia \geq x_{nut} > lb \\ \text{weight}_{<lb}\left(\dfrac{x_{nut}}{lb}\right) + \text{default}_{<lb}, & lb \geq x_{nut} \geq 0 \end{cases}$$

## 4. Effects of Customization

Our method provides three distinct ranges and functions to calculate an NSCORE in order to allow the meal plan to be customized. Numerically, it provides the user with the ability to apply a benefit or penalty to the NSCORE if the amount of a nutrient falls within a desirable or undesirable range. Customizing the weight and/or default values can be used to achieve specific nutritional outcomes. For example, users that are pregnant may require an increase or decrease in Folate. Users with high blood pressure may be sensitive to sodium.

To demonstrate this, consider calcium as defined in Table 2. This table shows that a user is interested in generating a meal plan that has at least 1000 mg of calcium (the LB), ideally 2500 mg (the IA), and no more than 5000 mg (the UB). Table 2 also shows that our method has generated a food list that results in 1164 mg of calcium (between the LB and IA). We will use information to show how altering the weights and defaults can affect the NCSORE (summarized in Table 3).

First, consider a control case with the weight and default values set to 1. This results in the NSCORE being the ratio of the amount of calcium to the IA (1164/2500) plus an additional point for the default value (1). This will result in an NSCORE of 1.466.

Now consider if the weight is doubled from 1 to 2 (graphically, this increases the slope of the line). By raising the weight we double the effect the ratio of calcium to IA. Now the calculation is 2 × (1164/2500) + 1. The final NSCORE for the second set of weights and defaults is 1.931. With the ability to adjust the weights and defaults

**Table 2.** Sample nutrient list.

| Nutrient | Unit | LB | IA | UB | Current Amount |
|---|---|---|---|---|---|
| Calcium | mg | 1000 | 2500 | 5000 | 1164 |

**Table 3.** Weight, Default, and NSCORE.

| Nutrient | Weight | | Default | | NSCORE | |
|---|---|---|---|---|---|---|
| | #1 | #2 | #1 | #2 | #1 | #2 |
| Calcium | 1 | 2 | 1 | 1 | 1.466 | 1.931 |

across three separate ranges the user will have fine grained control over the impact each nutrient range will have on the NSCORE and MSCORE.

## 5. Prototype Implementation

To demonstrate the utility of our method, we built a prototype implementation in Python Programming Language. The prototype uses the United States Department of Agricultures (USDA) National Nutrient Database (NND) for nutrient information of 8789 foods as of Standard Reference Release 28. For an overview of the algorithm see **Figure 2**. The prototype starts with an empty meal plan and incrementally adds foods in an attempt to increase the meal plan's MSCORE. The prototype operates by examining each nutrient in turn. For each nutrient the prototype selects a random food from a list of 100 foods that contain the highest amount of that nutrient (the NND can generate such a list). The food is added to the meal plan and a new MSCORE is calculated. If the food does not cause any $X_{nut}$ to exceed the $UB_{nut}$ the food is kept in the meal plan. The next nutrient in the list is selected and the process repeats. Foods are added until each nutrient exceeds its LB. The meal plan is complete when all nutrient minimum requirements.

Adding a food, whoever, may cause a nutrient to exceed its upper bound. When that happens the food is removed, and a new food is attempted. For example, adding bacon to the meal plan to increase phosphorus may cause sodium to increase beyond its UB. During this looping process our prototype keeps a record of the number of times a nutrient causes a food rejection. If a nutrient causes a food rejection four or more times we remove the food item from the meal plan with the highest amount of that nutrient. Continuing with the example above, once bacon is rejected, the algorithm may add peanuts, which may also increase sodium above its UB. Assume two more foods are added that cause sodium to exceed its UB. Our assumption is that the meal plan is already high in sodium, and the algorithm is having trouble adding foods that maintain sodium below its UB. Therefore, the algorithm removes the food currently in the meal plan that has the highest sodium content.

Below is a sample output from the prototype on its first loop through the nutrient list. Note line 12. While looking for a food to add calcium, the prototype selected a food that would have caused sodium to exceed its UB and was rejected. On line 22, while attempting to fit a food for vitamin E, sodium caused a food to be rejected for the fourth time. Because of this before we start the next loop the prototype will remove the food with the highest amount of sodium (line 24).

1) Looking for Vitamin C.
2) Adding Peppers, sweet, green, cooked, boiled, drained, without salt.
3) Looking for Vitamin B1 (thiamine).
4) Adding WORTHINGTON Dinner Roast, frozen, unprepared.
5) Looking for Vitamin B2 (riboflavin).
6) Adding Spaghetti with meat sauce, frozen entrée.
7) Looking for Vitamin B3 (niacin).
8) Adding Fast foods, submarine sandwich, turkey, roast beef and ham on white bread with lettuce and tomato.
9) Looking for Pantothenic Acid.
10) Adding Chicken, liver, all classes, cooked, pan-fried.
11) Looking for Calcium.
12) Exceeded Sodium (1).

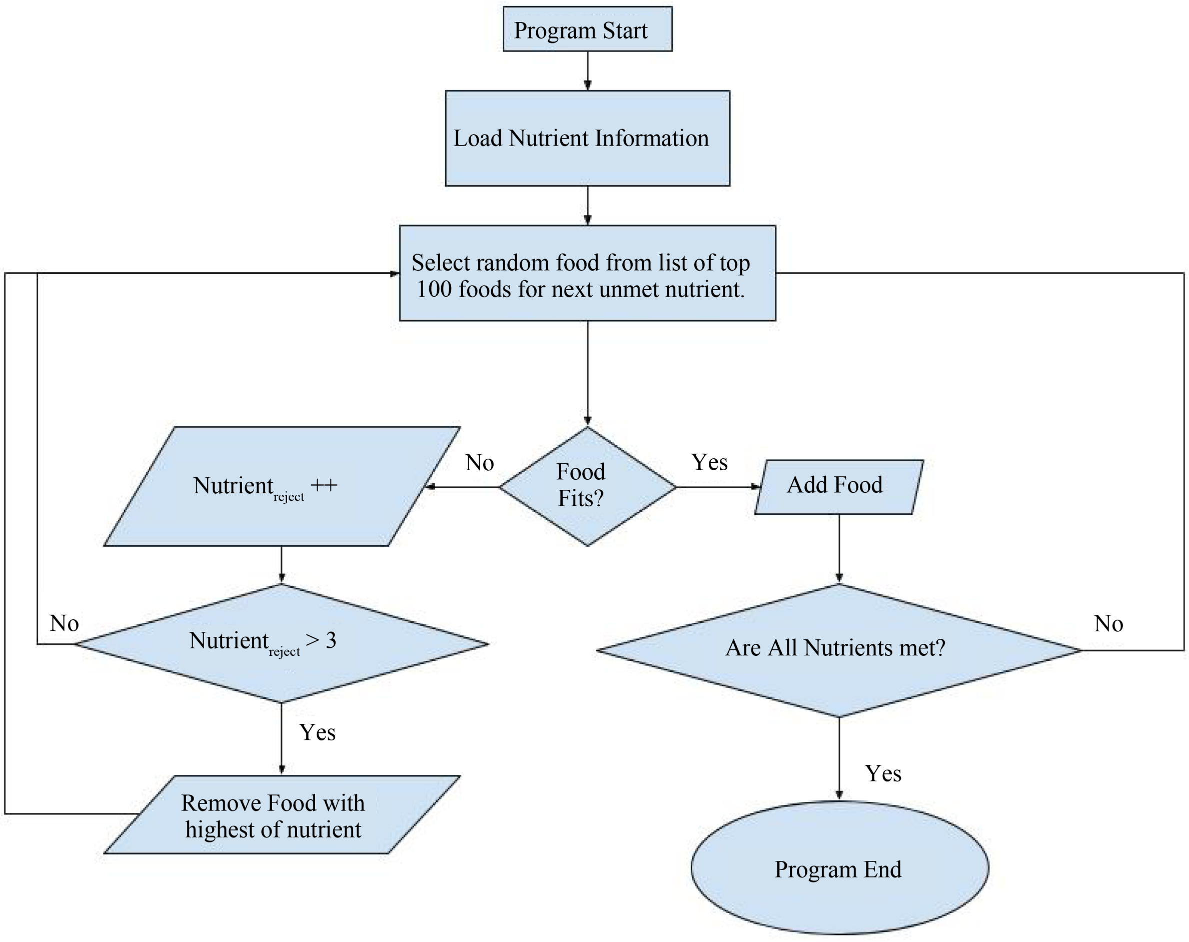
Program Start
Load Nutrient Information
Select random food from list of top 100 foods for next unmet nutrient.
Food Fits?
No
Yes
Nutrient$_{reject}$ ++
Add Food
Nutrient$_{reject}$ > 3
No
Yes
Are All Nutrients met?
No
Yes
Remove Food with highest of nutrient
Program End

**Figure 2.** Graphical representation of the algorithm used to generate a food list.

13) Looking for Vitamin K.
14) Exceeded Sodium (2).
15) Looking for Magnesium.
16) Adding Peanuts, Virginia, oil-roasted, without salt.
17) Looking for Potassium.
18) Adding Plums, dried (prunes), uncooked.
19) Looking for Zinc.
20) Exceeded Sodium (3).
21) Looking for Vitamin E.
22) Exceeded Sodium (4).
23) The score is: 5.52796417989.
24) Removing a food for Sodium.
25) Looking for Vitamin C.
26) Adding Orange juice, raw.
27) Looking for Calcium.
28) Adding CARRABBA'S ITALIAN GRILL, cheese ravioli with marinara sauce.
29) Looking for Vitamin K.
30) Exceeded Sodium (1).
31) Looking for Magnesium.
32) Exceeded Sodium (2).
33) Looking for Potassium.

34) Adding Apricots, dehydrated (low-moisture), sulfured, stewed.
35) Looking for Zinc.
36) Exceeded Sodium (3).
37) Looking for Vitamin E.
38) Adding Oil, sunflower, linoleic (less than 60%).
39) This report is for date: 02/08/16. The score is: 7.55021904762.

Once the method has added enough foods to bring each nutrient over its LB, the program is stopped and the user is presented with the final food list, each NSCORE, an MSCORE, and nutrient breakdown. Below is a sample of the final output.

1) Elapsed time (ms): 329500.334978.
2) This report is for date: 02/20/16. The score is: 19.7325233333.
3) Nutrient List.
4) Histidine = N/A DRV (2.068 g/0).
5) Vitamin C = 40.7% of Ideal (609.9 mg/1.2198).
6) Vitamin B1 (thiamine) = 84.4% of Ideal (2.111 mg/2.5332).
7) Vitamin B2 (riboflavin) = 174.6% of Ideal (4.015 mg/0.330833333333).
8) Vitamin B3 (niacin) = 69.4% of Ideal (20.81 mg/2.081).
9) Pantothenic Acid = 54.2% of Ideal (10.837 mg/1.62555).
10) Folate = 197.8% of Ideal (989.0 ug/0.6044).
11) Calcium = 107.8% of Ideal (2694.0 mg/3.6896).
12) Vitamin K = 24.8% of Ideal (124.1 ug/0.7446).
13) Iron = 73.7% of Ideal (22.12 mg/2.212).
14) Magnesium = 87.2% of Ideal (698.0 mg/2.6175).
15) Phosphorus = 246.1% of Ideal (2461.0 mg/0.0156).
16) Potassium = 153.1% of Ideal (6890.0 mg/−3.92142857143).
17) Sodium = 136.3% of Ideal (1363.0 mg/0.0264285714286).
18) Zinc = 107.1% of Ideal (18.2 mg/0.636).
19) Copper = 49.6% of Ideal (2.978 mg/1.489).
20) Fluoride = 0.0% of Ideal (0 ug/0).
21) Manganese = 178.7% of Ideal (5.36 mg/0.464).
22) Selenium = 72.0% of Ideal (72.0 ug/2.16).
23) Vitamin A = 111.1% of Ideal (8334.0 IU/0.66664).
24) Vitamin E = 154.1% of Ideal (46.22 mg/0.5378).
25) Vitamin D = 0.4% of Ideal (8.0 IU/0).
26) Tryptophan = N/A DRV (0.904 g/0).
27) Threonine = N/A DRV (3.2 g/0).
28) Isoleucine = N/A DRV (3.487 g/0).
29) Leucine = N/A DRV (6.107 g/0).
30) Lysine = N/A DRV (4.907 g/0).
31) Methionine = N/A DRV (0.914 g/0).
32) Phenylalanine = N/A DRV (1.002 g/0).
33) Valine = N/A DRV (3.732 g/0).
34) Food List.
35) Beverages, MONSTER energy drink, low carb-Beverages.
36) Broccoli, frozen, chopped, unprepared-Vegetables and Vegetable Products.
37) Beans, white, mature seeds, raw-Legumes and Legume Products.
38) Desserts, mousse, chocolate, prepared-from-recipe-Sweets.
39) Strawberries, frozen, sweetened, sliced-Fruits and Fruit Juices.
40) Peppers, sweet, red, raw-Vegetables and Vegetable Products.
41) Lupins, mature seeds, raw-Legumes and Legume Products.
42) CAMPBELL’S, V8 Vegetable Juice, Essential Antioxidants V8-Vegetables and Vegetable Products.
43) Pepper, banana, raw-Vegetables and Vegetable Products.
44) Peppers, hot chili, green, raw-Vegetables and Vegetable Products.

45) Whey, acid, dried-Dairy and Egg Products.
46) Oil, sunflower, high oleic (70% and over)-Fats and Oils.
47) Mushrooms, white, cooked, boiled, drained, without salt-Vegetables and Vegetable Products.

## 6. Evaluation

We have run the prototype implementation dozens of times using both the baseline configuration as well as custom weights and default values. The prototype never failed to produce a valid meal plan. The random nature of the food choices has effects, however. First, meal plans are seldom the same. It is theoretically possible to produce the same meal plan twice, but selecting from 100 foods for each of 19 nutrients makes it highly unlikely. The random nature of the food choices combined with the large number of food choices makes achieving a perfect score effectively impossible.

Second, some meal plans have some interesting food combinations. For example, one meal plan included a Red Bull and a Twinkie, yet still met the nutritional goals. A model such as SSC3gd could be used to pare down the food list prior to it being passed to our method. All else being equal, we tend to err on the side of variety.

Finally, because of the way the prototype generates a meal plan, achieving a valid meal plan takes more comparisons as the difference between the LB and UB shrinks. The algorithm will require even more comparisons as more nutrients have smaller ranges between their LB and UB. This is due to the randomized food choices. The prototype would need to select foods based, in part, on their effect on the MSCORE. We erred on the side of variety and decided to stay away from making the food selection too smart because this could potentially lead to the same foods always being selected, and the same meal plan being generated.

## 7. Future Improvements

The prototype exists only as a command line app. We would like to build a fully functional website using the core scoring algorithm implemented in the method. The website will have a pantry feature that would allow users to track the foods that they currently have in their pantry/house. We would use this information, along with the food list for generated meal plans, to provide the users with a shopping list. The website would also present users with nutritional goal templates, an example being "Post race recovery". The template would recommend the LB, IA, and UB, and would insert the nutrient defaults and weights. The website would provide the user with recipes, as opposed to a simple food list. The recipes would be user submitted, along with the selecting what foods from the USDA database to use.

## References

[1] Aetna (2008) The Facts About Rising Health Case Costs. http://www.aetna.com/health-reform-connection/aetnas-vision/facts-about-costs.htm

[2] Ogden, C.L., Carroll, M.D., Kit, B.K. and Flegal, K.M. (2014) Prevalence of Childhood and Adult Obesity in the United States. *The Journal of American Medical Association*, **311**, 806-814. http://dx.doi.org/10.1001/jama.2014.732

[3] World Health Organization (2015) Global Databose on Body Mass Index. http://apps.who.int/bmi/index.jsp

[4] National Heart, Lung, and Blood Institute (2012) How Can Overweight and Obesity Be Prevented? http://www.nhlbi.nih.gov/health/health-topics/topics/obe/prevention

[5] Finkelstein, E.A., Trogdon, J.G., Cohen, J.W. and Dietz, W. (2009) Annual Medical Spending Attributable to Obesity: Payer-and Service-Specific Estimates. *Health Affairs*, **28**, 822-831. http://dx.doi.org/10.1377/hlthaff.28.5.w822

[6] Tippett, K. (2009) Transcript for Mark Hyman, James Gordon, and Penny George—The Evolution of Medicine. http://www.onbeing.org/program/mark-hyman-james-gordon-and-penny-george-the-evolution-of-medicine/transcript/8187#main_content

[7] Rayner, M., Scarborough, P., Stockley, L. and Boxer, A. (2005) Nutrient Profiles; Further Refinement and Testing of Model SSCg3d. Final Report 2005. http://www.food.gov.uk/multimedia/pdfs/npreportsept05.pdf