Methods for Using NHANES for Exposure Estimates

Heather R. Campbell
University of Kentucky, hrca227@g.uky.edu

Robert A. Lodder
University of Kentucky, r.lodder@uky.edu

Methods for Using NHANES for Exposure Estimates

Heather R. Campbell and Robert A. Lodder*

Department of Pharmaceutical Sciences, University of Kentucky, Lexington, KY, USA

*Author to whom correspondence should be addressed. Email: Lodder@g.uky.edu

Abstract

The National Health and Nutrition Examination Survey (NHANES) is a scheme of studies with the objective of measuring the health and nutritional status of adults and children in the United States. The survey is unique in that it has a clinical component (NCT00005154), and combines interviews and physical examinations into its data sets. NHANES is an important program of the National Center for Health Statistics (NCHS). NCHS is a unit of the Centers for Disease Control and Prevention (CDC) and has the responsibility for producing vital and health statistics for the United States. Data from the NHANES studies can be used to produce exposure estimates of the population to foods, food ingredients, additives, dietary supplements, and in some years of the surveys, other items like allergens, prescription medications, endotoxins, cigarettes and alcohol. This paper is about methods for using NHANES to establish exposure estimates for food ingredients and additives. Sample Octave/Matlab code is provided, but other software can be used with the NHANES databases as well.

Introduction

Data-driven computational science (DDCS) has found a plethora of applications in drug design. Molecular data are frequently employed to come up with new drug molecules (Mouchlis, 2020)(Alshehri, 2020). Engineering process simulations help steer the development of the Chemistry, Manufacturing, and Controls (CMC) section of Investigational New Drug (IND) applications submitted to the US Food and Drug Administration (FDA)(Moreno-Sader, 2020)(Diab, 2020). Computer simulations can also guide the design of preclinical trials (Van Norman, 2020) and human clinical trials as well as substitute for some as virtual clinical trials ().

The Centers for Disease Control (CDC) National Health and Nutrition Examination Survey (NHANES) is a program of studies designed to assess the health and nutritional status of adults and children in the United States (NCT00005154 and NCHS IRB/ERB Protocol Number #2011-17). The survey is unique in that it combines interviews and physical examinations which includes laboratory results. The NHANES database can be mined to determine the average and 90th percentile exposures to a food additive or ingredient, and early human formulation testing conducted at levels beneath those to which the US population is ordinarily exposed through food, dietary supplements, and occasionally other sources, depending upon the years of the NHANES survey (different surveys often include ad hoc measurements of environmental exposure, medical history, etc.)(Campbell, 2020;Lodder, 2017).

Food Consumption Survey Data

Survey Description

The data from the National Health and Nutrition Examination Surveys are available free online for public use. De-identified data are openly available, while detailed data that might identify participants are only available to cleared research personnel. NHANES are conducted as a continuous, annual survey, and are published online in 2-year cycles. In each cycle, approximately 10,000 people across the U.S. complete the health examination part of the survey. Any choice of consecutive years of data collection forms a nationally representative sample of the U.S. population. It is well established that the duration of a dietary survey changes the estimated consumption of individual consumers and that short-term surveys, such as the typical 1-day dietary survey, overestimate consumption over longer time periods (Gregory et al., 1995). In NHANES, two 24-hour dietary recalls are delivered and these data are employed to generate exposure estimates for intake analyses.

The NHANES provides the most suitable data for evaluating food-use and food-consumption trends in the United States, containing two years of data on persons chosen via stratified multistage probability sampling of the civilian non-institutionalized population of the United States. NHANES survey data are assembled from persons and households using 24-hour dietary recalls delivered on 2 non-consecutive days (Day 1 and Day 2) throughout all four seasons of the year. Day 1 data are obtained in-person in the Mobile Examination Center (MEC), and Day 2 data are obtained by telephone in the following 3 to 10 days, on different days of the week, to attain the targeted level of statistical independence. The data are collected by first choosing Primary Sampling Units (PSUs), which are counties throughout the United States. Small counties are combined in PSUs to reach a minimum population size. These PSUs are divided into segments and households are selected within each segment. One or more persons within a household are interviewed. Fifteen PSUs are visited each year. For example, in the 2009-2010 NHANES, there were 13,272 persons selected; of these 10,253 were considered respondents to the MEC examination and data collection. 9,754 of the MEC respondents gave complete dietary intakes for Day 1 and of those supplying the Day 1 data, 8,405 supplied complete dietary intakes for Day 2. These remain typical numbers for the subsequent biannual NHANES.

In addition to assimilating information on the varieties and quantities of foods being consumed, the NHANES surveys gather socioeconomic, physiological, and demographic information from individual persons in the survey, such as sex, age, height and weight, and other variables helpful in characterizing consumption. The incorporation of this information allows a deeper evaluation of food intake based on consumption by specific population groups within the total population.

Sample weights are built into NHANES surveys to counteract the possible under-representation of intakes from specific population groups as a result of sample variability due to survey design, differential non-response rates, or other aspects, such as problems in the sampling frame (CDC, 2006; USDA, 2012).

Methods

Consumption data from the dietary records of subjects, detailing food items consumed by each subject surveyed, can be collated by computer in R, Python, SAS, Octave, Matlab or other languages and used to generate estimates of the intake of an ingredient by the U.S. population. Sample code for doing this calculation in Octave or Matlab appears in the Appendix. Estimates for the daily intake of the ingredient represent projected 2-day averages for each individual from Day 1 and Day 2 records of NHANES data. These average amounts form the consumption distribution from which mean and percentile intake estimates are constructed. Mean and percentile estimates are generated using sample weights to produce intake estimates for the total population of the United States. “All-user” intake refers to the estimated consumption of the specified ingredient by those individuals consuming food products containing the targeted ingredient. Individuals are considered users if they ingest 1 or more food products containing the target ingredient on either Day 1 or Day 2 of the survey.

Food Usage

In publications, the individual food uses are generally summarized in an appendix or in supplementary material because the list of food codes can be so long. Food codes representative of each approved use are chosen from the Food and Nutrition Database for Dietary Studies (FNDDS) for the corresponding biennial NHANES survey. In FNDDS, the primary (usually generic) description of a given food is assigned a unique 8-digit food code (CDC, 2006; USDA, 2012).

The sample program code in the Appendix is designed to calculate mean Expected Daily Intake (EDI) and 90th percentile EDI for a food ingredient called ellagic acid (Dickerson, 2018). Ellagic acid is a natural phenol antioxidant found in many fruits and vegetables. The antiproliferative and antioxidant properties of ellagic acid have elicited research into its potential health benefits.

The consumption patterns in Figures 1-4 reveal a common trend in such EDI studies: on a pure mass basis (grams consumed), adults are the largest consumers of most ingredients and additives, but on a weight-adjusted basis (grams consumed per gram of body weight), children are often the largest consumers.

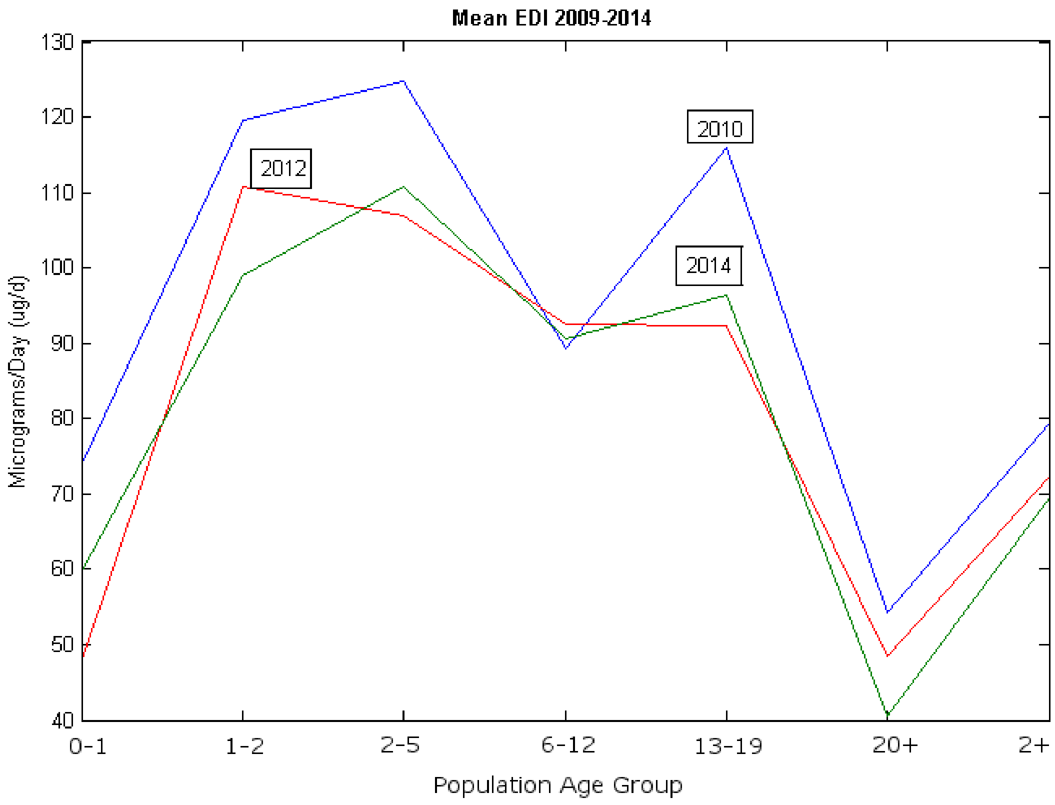

Figure 1. Children consume more ellagic acid on average than adults. Baby foods are often made from ingredients that contain large amounts of ellagic acid. The blue line shows data from the 2009–2010 NHANES, the red line data from the 2011–2012 NHANES, and the green line data from the 2013–2014 NHANES.

Another feature of EDI consumption curves is that they are usually similar from one biennial NHANES survey to the next (Figures 1-4 show minor variation from 2010-2014). Nevertheless, trends are sometimes seen, and a gradual reduction in consumption of fruits and vegetables will appear as a slow reduction in ellagic acid consumption in EDI curves drawn from successive NHANES.

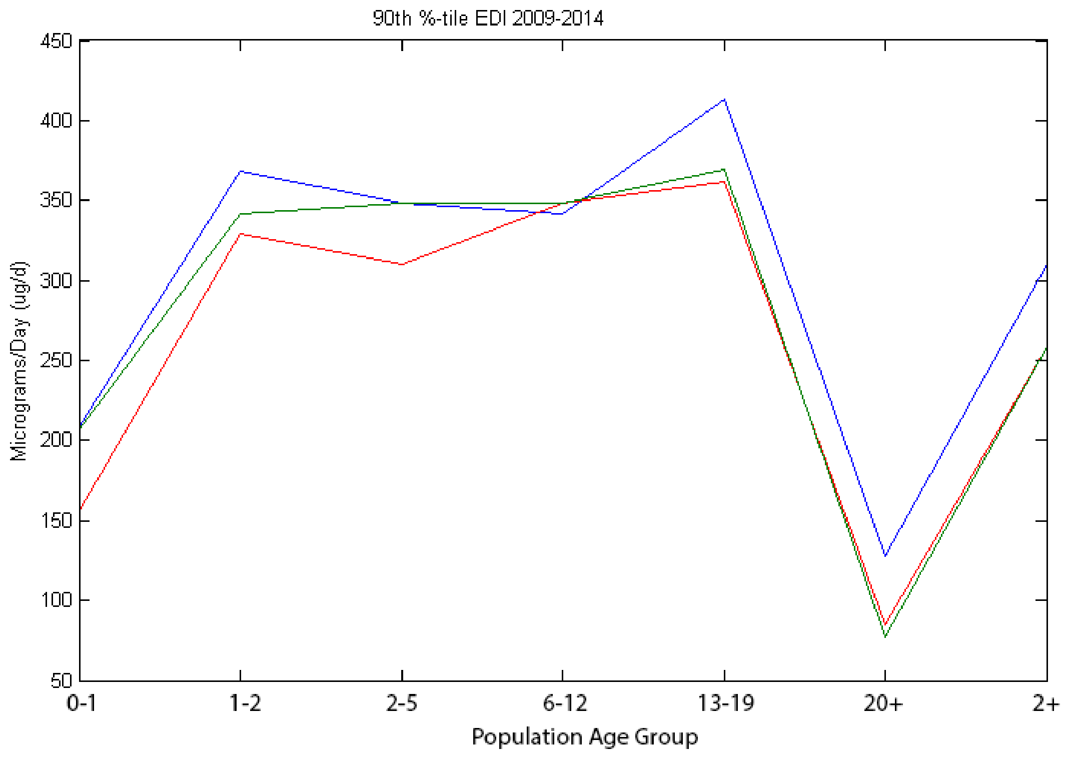

Figure 2. Teenagers contribute the highest peak consumption among the 90th percentile consumers of EA. The blue line shows data from the 2009–2010 NHANES, the red line data from the 2011–2012 NHANES, and the green line data from the 2013–2014 NHANES.

To determine whether a long term trend in ellagic acid consumption really exists more NHANES surveys must be analyzed, of course. A model built on 6-10 successive surveys, with analysis of the residuals, would be more convincing.

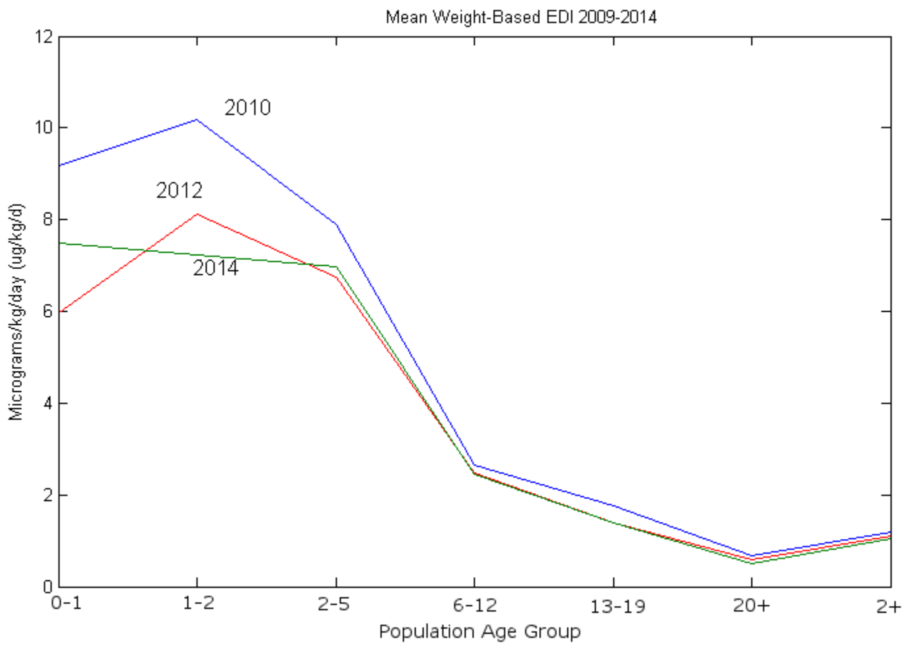

Figure 3. When ellagic acid exposure is calculated on a per kilogram of body weight basis, toddlers aged 1 to 2 years are exposed to the most ellagic acid on average. The blue line shows data from the 2009–2010 NHANES, the red line data from the 2011–2012 NHANES, and the green line data from the 2013–2014 NHANES.

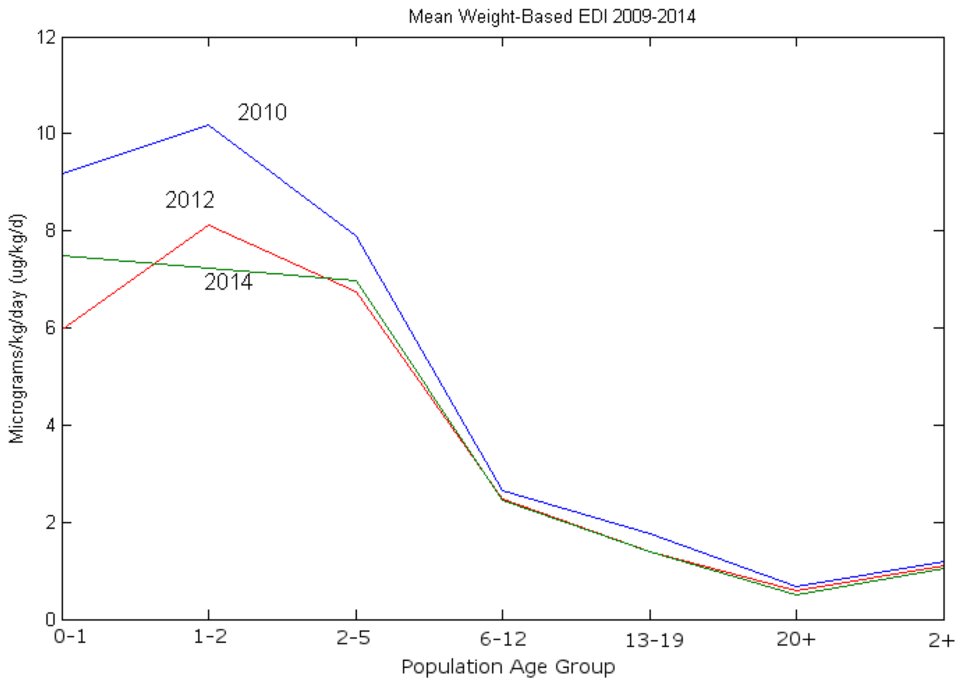

Figure 4. When EA exposure is calculated on a per kilogram of body weight basis for the 90th percentile consumers, toddlers aged 1 to 2 years are again exposed to the most EA. The blue line shows data from the 2009–2010 NHANES, the red line data from the 2011–2012 NHANES, and the green line data from the 2013–2014 NHANES.

Acknowledgements

The project described was supported by NSF ACI-1053575 allocation number BIO170011 and the National Center for Research Resources and the National Center for Advancing Translational Sciences, National Institutes of Health, through Grant UL1TR001998. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

References

Alshehri, Abdulelah S., Rafiqul Gani, and Fengqi You. "Deep Learning and Knowledge-Based Methods for Computer Aided Molecular Design--Toward a Unified Approach: State-of-the-Art and Future Directions." arXiv preprint arXiv:2005.08968 (2020).https://arxiv.org/ftp/arxiv/papers/2005/2005.08968.pdf

Campbell, Heather R., Cecil, Regan P. and Lodder, Robert A. Population Data-Driven Formulation of a COVID-19 Therapeutic, MEDRXIV/2020/161547

CDC 2006. Analytical and Reporting Guidelines: The National Health and Nutrition Examination Survey (NHANES). National Center for Health Statistics, Centers for Disease Control and Prevention; Hyattsville, Maryland. Available from: http://www.cdc.gov/nchs/data/nhanes/nhanes_03_04/nhanes_analytic_guidelines_dec_2005.pdf

Diab, S. (2020). Computational modelling of separation processes for green continuous pharmaceutical manufacturing. Edinburgh Research Archive.

Dickerson, C., Ensor, M., & Lodder, R. A. (2018, June). Establishing EDI for a Clinical Trial of a Treatment for Chikungunya. In International Conference on Computational Science (pp. 773-782). Springer, Cham.

Gregory, J. R., Collins, D. L., Davies, P. S. W., Hughes, J. M., & Clarke, P. C. (1995). National Diet and Nutrition Survey: children aged 1.5 to 4.5 years. HMSO Publications Centre.

Lodder, Robert A. (2017) "Data-Driven Design of an Ebola Therapeutic." Procedia Computer Science 108 : 1612. https://www.sciencedirect.com/science/article/pii/S1877050917306944 or click here for the PDF

Moreno-Sader, K., Meramo-Hurtado, S. I., & González-Delgado, A. D. (2020). Environmental sustainability analysis of chitosan microbeads production for pharmaceutical applications via computer-aided simulation, WAR and TRACI assessments. Sustainable Chemistry and Pharmacy, 15, 100212.

Mouchlis, Varnavas D., et al. "Computer-Aided Drug Design of β-Secretase, γ-Secretase and Anti-Tau Inhibitors for the Discovery of Novel Alzheimer's Therapeutics." International Journal of Molecular Sciences 21.3 (2020): 703. https://www.mdpi.com/1422-0067/21/3/703/pdf

USDA 2012. What We Eat In America (WWEIA), NHANES: overview. Available from: http://www.ars.usda.gov/Services/docs.htm?docid=13793#release

Van Norman, G. A. (2020). Limitations of Animal Studies for Predicting Toxicity in Clinical Trials: Part 2: Potential Alternatives to the Use of Animals in Preclinical Trials. JACC: Basic to Translational Science, 5(4), 387-397.
https://www.sciencedirect.com/science/article/pii/S2452302X20301261

Appendix

The programs are written in Matlab code that will execute in GNU Octave.
GNU Octave can be downloaded free of charge at https://www.gnu.org/software/octave/

EDI.m code

```
function [zarray] = EDI(serving_sizes,dose,age,day1,day2,mass)
%% This function solves for Expected Daily Intakes (EDIs) using the NHANES and FNDDS
databases.
% copyright R. Lodder March 10, 2016, revised
% The programs searchnzero, searchfoods.m, and searchsizes.m must all be in
% the working directory path.
% serving sizes = in rows to target_codes, 1st col.=food codes, 2nd
%               col. = serving sizes in g
% dose is a scalar variable specified by the client
% age, day1, day2, and mass are all in the “NHANES basic startup workspace.mat” workspace
% the columns of age are 1-SubjectID, 2-age in months
% the columns of day1 and day2 are 1-SubjectID, 2-foodcode, 3-mass consumed (g)
% the columns of mass are 1-SubjectID, 2-body mass in kg
%
% zarray returns the mean mass of all ages just to put something into the output variable and
avoid an error
%
% Start with all patient sequence numbers in age groups
% Time to make food code tables for each group
%
% loop through Day1 and check each row to see if the SEQN in the first column is in the set of
SEQN for each age group
% if so, leave the row as is. if not, then replace the row with all zeros and eliminate the zeros
later
% get all foods consumed by each age group
target_codes = serving_sizes(:,1); % target_codes is a vector food codes of length = number of
rows in serving_sizes
day1(isnan(day1))=0;      % eliminate NaNs in grams consumed and replace with zeros
day2(isnan(day2))=0;
% find the infant groups
ind_age0_1 = find((age(:,2)>0)&(age(:,2)<=13));   % create index locations for age groups, ages
are given in months, not years
age0_1=age(ind_age0_1,1);
disp('Number of infants aged 0-1')
size(ind_age0_1)
```

```
ind_age1_2 = find((age(:,2)>14)&(age(:,2)<=24));
age1_2=age(ind_age1_2,1);
disp('Number of toddlers aged 1-2')
size(ind_age1_2)
ind_age2_5 = find((age(:,2)>23)&(age(:,2)<=61));
age2_5=age(ind_age2_5,1);
disp('Number of subjects aged 2-5')
size(age2_5)
ind_age6_12 = find((age(:,2)>62)&(age(:,2)<=145));
age6_12=age(ind_age6_12,1);
disp('Number of subjects aged 6-12')
size(age6_12)
ind_age13_19 = find((age(:,2)>146)&(age(:,2)<=229));
age13_19=age(ind_age13_19,1);
disp('Number of subjects aged 13-19')
size(age13_19)
ind_age20_up = find((age(:,2)>229));
age20_up=age(ind_age20_up,1);
disp('Number of subjects aged 20 and up')
size(ind_age20_up)
% find all ages > 2 years
ind_age2up = find((age(:,2)>24)&(age(:,2)<2000));
age2up=age(ind_age2up,1);
disp('Total number of subjects aged 2 and up')
size(age2_5)+size(age6_12)+size(age13_19)+size(ind_age20_up)   % was size(ind_age2up)
% Day 1, delete entries not in specified age group
day1_age0_1 = searchnzero(age0_1,day1);
day1_age1_2 = searchnzero(age1_2,day1);
day1_age2_5 = searchnzero(age2_5,day1);
day1_age6_12 = searchnzero(age6_12,day1);
day1_age13_19 = searchnzero(age13_19,day1);
day1_age20_up = searchnzero(age20_up,day1);
day1_age2up = searchnzero(age2up,day1);
% Day 2
day2_age0_1 = searchnzero(age0_1,day2);
day2_age1_2 = searchnzero(age1_2,day2);
day2_age2_5 = searchnzero(age2_5,day2);
day2_age6_12 = searchnzero(age6_12,day2);
day2_age13_19 = searchnzero(age13_19,day2);
day2_age20_up = searchnzero(age20_up,day2);
day2_age2up = searchnzero(age2up,day2);
```

```
% throw away all food codes for each age's dietary data that are not in the targeted food code
list
%
day1_fc_age0_1 = searchfoods(day1_age0_1,target_codes);
day1_fc_age1_2 = searchfoods(day1_age1_2,target_codes);
day1_fc_age2_5 = searchfoods(day1_age2_5,target_codes);
day1_fc_age6_12 = searchfoods(day1_age6_12,target_codes);
day1_fc_age13_19 = searchfoods(day1_age13_19,target_codes);
day1_fc_age20_up = searchfoods(day1_age20_up,target_codes);
day1_fc_age2up = searchfoods(day1_age2up,target_codes);
%
day2_fc_age0_1 = searchfoods(day2_age0_1,target_codes);
day2_fc_age1_2 = searchfoods(day2_age1_2,target_codes);
day2_fc_age2_5 = searchfoods(day2_age2_5,target_codes);
day2_fc_age6_12 = searchfoods(day2_age6_12,target_codes);
day2_fc_age13_19 = searchfoods(day2_age13_19,target_codes);
day2_fc_age20_up = searchfoods(day2_age20_up,target_codes);
day2_fc_age2up = searchfoods(day2_age2up,target_codes);
%
disp('Number of subjects eating target food on Day 1 aged 0-1')
day1_fc_age0_1u = unique(day1_fc_age0_1(:,1));   % just count unique users
size(day1_fc_age0_1u)
disp('Number of subjects eating target food on Day 1 aged 1-2')
day1_fc_age1_2u = unique(day1_fc_age1_2(:,1));
size(day1_fc_age1_2u)
disp('Number of subjects eating target food on Day 1 aged 2-5')
day1_fc_age2_5u = unique(day1_fc_age2_5(:,1));
size(day1_fc_age2_5u)
disp('Number of subjects eating target food on Day 1 aged 6-12')
day1_fc_age6_12u = unique(day1_fc_age6_12(:,1));
size(day1_fc_age6_12u)
disp('Number of subjects eating target food on Day 1 aged 13-19')
day1_fc_age13_19u = unique(day1_fc_age13_19(:,1));
size(day1_fc_age13_19u)
disp('Number of subjects eating target food on Day 1 aged 20 and up')
day1_fc_age20_upu = unique(day1_fc_age20_up(:,1));
size(day1_fc_age20_upu)
disp('Number of subjects eating target food on Day 1 aged 2 and up')
day1_fc_age2upu = unique(day1_fc_age2up(:,1));
% was size(day1_fc_age2upu)
size(day1_fc_age2_5u)+size(day1_fc_age6_12u)+size(day1_fc_age13_19u)+size(day1_fc_age
20_upu)
```

```
%
disp('Number of subjects eating target food on Day 2 aged 0-1')
day2_fc_age0_1u = unique(day2_fc_age0_1(:,1));   % just count unique users
size(day2_fc_age0_1u)
disp('Number of subjects eating target food on Day 2 aged 1-2')
day2_fc_age1_2u = unique(day2_fc_age1_2(:,1));
size(day2_fc_age1_2u)
disp('Number of subjects eating target food on Day 2 aged 2-5')
day2_fc_age2_5u = unique(day2_fc_age2_5(:,1));
size(day2_fc_age2_5u)
disp('Number of subjects eating target food on Day 2 aged 6-12')
day2_fc_age6_12u = unique(day2_fc_age6_12(:,1));
size(day2_fc_age6_12u)
disp('Number of subjects eating target food on Day 2 aged 13-19')
day2_fc_age13_19u = unique(day2_fc_age13_19(:,1));
size(day2_fc_age13_19u)
disp('Number of subjects eating target food on Day 2 aged 20 and up')
day2_fc_age20_upu = unique(day2_fc_age20_up(:,1));
size(day2_fc_age20_upu)
disp('Number of subjects eating target food on Day 2 aged 2 and up')
day2_fc_age2upu = unique(day2_fc_age2up(:,1));
% was size(day2_fc_age2upu)
size(day2_fc_age2_5u)+size(day2_fc_age6_12u)+size(day2_fc_age13_19u)+size(day2_fc_age
20_upu)
% scale food consumption of each food in grams to number of servings
% now do Day 1
serv_day1_age0_1 = searchsizes(day1_fc_age0_1,serving_sizes);
serv_day1_age1_2 = searchsizes(day1_fc_age1_2,serving_sizes);
serv_day1_age2_5 = searchsizes(day1_fc_age2_5,serving_sizes);
serv_day1_age6_12 = searchsizes(day1_fc_age6_12,serving_sizes);
serv_day1_age13_19 = searchsizes(day1_fc_age13_19,serving_sizes);
serv_day1_age20_up = searchsizes(day1_fc_age20_up,serving_sizes);
serv_day1_age2up = searchsizes(day1_fc_age2up,serving_sizes);
% now do Day 2
serv_day2_age0_1 = searchsizes(day2_fc_age0_1,serving_sizes);
serv_day2_age1_2 = searchsizes(day2_fc_age1_2,serving_sizes);
serv_day2_age2_5 = searchsizes(day2_fc_age2_5,serving_sizes);
serv_day2_age6_12 = searchsizes(day2_fc_age6_12,serving_sizes);
serv_day2_age13_19 = searchsizes(day2_fc_age13_19,serving_sizes);
serv_day2_age20_up = searchsizes(day2_fc_age20_up,serving_sizes);
serv_day2_age2up = searchsizes(day2_fc_age2up,serving_sizes);
```

```
% now calculate mean Expected Daily Intakes (EDIs) using the mass of additive that the client
wants to add to each food.
% Day 1
disp('Mean Expected Daily Intakes (EDIs)')
mean0_1day1 = mean(serv_day1_age0_1(:,4))*dose; % mg EDI
mean1_2day1 = mean(serv_day1_age1_2(:,4))*dose; % mg EDI
mean2_5day1 = mean(serv_day1_age2_5(:,4))*dose; % mg EDI
mean6_12day1 = mean(serv_day1_age6_12(:,4))*dose; % mg EDI
mean13_19day1 = mean(serv_day1_age13_19(:,4))*dose; % mg EDI
mean20_upday1 = mean(serv_day1_age20_up(:,4))*dose; % mg EDI
mean2upday1 = mean(serv_day1_age2up(:,4))*dose; % mg EDI
% now do day 2
mean0_1day2 = mean(serv_day2_age0_1(:,4))*dose; % mg EDI
mean1_2day2 = mean(serv_day2_age1_2(:,4))*dose; % mg EDI
mean2_5day2 = mean(serv_day2_age2_5(:,4))*dose; % mg EDI
mean6_12day2 = mean(serv_day2_age6_12(:,4))*dose; % mg EDI
mean13_19day2 = mean(serv_day2_age13_19(:,4))*dose; % mg EDI
mean20_upday2 = mean(serv_day2_age20_up(:,4))*dose; % mg EDI
mean2upday2 = mean(serv_day2_age2up(:,4))*dose; % mg EDI
% to make histograms of exposure data here, add a breakpoint and use
% hist().  For example, copy and paste
% bins=round(sqrt(size(serv_day2_age2up(:,4))))
% bins=bins(1)
% hist(serv_day2_age2up(:,4),bins)
%
% average day 1 and day 2
disp('Mean EDI ages 0 to 1')
medi0_1=(mean0_1day1+mean0_1day2)/2
disp('Mean EDI ages 1-2')
medi1_2=(mean1_2day1+mean1_2day2)/2
disp('Mean EDI ages 2-5')
medi2_5=(mean2_5day1+mean2_5day2)/2
disp('Mean EDI ages 6-12')
medi6_12=(mean6_12day1+mean6_12day2)/2
disp('Mean EDI ages 13-19')
medi13_19=(mean13_19day1+mean13_19day2)/2
disp('Mean EDI ages 20 and up')
medi20up=(mean20_upday1+mean20_upday2)/2
disp('Mean EDI ages 2 and up')
medi2up=(mean2upday1+mean2upday2)/2
% for 90th percentile calculation of Day 1, produces sorted doses in each age group
sort_day1_age0_1 = sort(serv_day1_age0_1(:,4))*dose;
```

```
sort_day1_age1_2 = sort(serv_day1_age1_2(:,4))*dose;
sort_day1_age2_5 = sort(serv_day1_age2_5(:,4))*dose;
sort_day1_age6_12 = sort(serv_day1_age6_12(:,4))*dose;
sort_day1_age13_19 = sort(serv_day1_age13_19(:,4))*dose;
sort_day1_age20_up = sort(serv_day1_age20_up(:,4))*dose;
sort_day1_age2up = sort(serv_day1_age2up(:,4))*dose;
%
sort_day1_age0_1_d1 = sort_day1_age0_1(round(.9*size(sort_day1_age0_1)));
sort_day1_age1_2_d1 = sort_day1_age1_2(round(.9*size(sort_day1_age1_2)));
sort_day1_age2_5_d1 = sort_day1_age2_5(round(.9*size(sort_day1_age2_5))); % 90th
percentile
sort_day1_age6_12_d1 = sort_day1_age6_12(round(.9*size(sort_day1_age6_12)));
sort_day1_age13_19_d1 = sort_day1_age13_19(round(.9*size(sort_day1_age13_19)));
sort_day1_age20_up_d1 = sort_day1_age20_up(round(.9*size(sort_day1_age20_up)));
sort_day1_age2up_d1 = sort_day1_age2up(round(.9*size(sort_day1_age2up)));
%
% Day 2 for 90th percentile calculation, produces sorted doses in each age group
sort_day2_age0_1 = sort(serv_day2_age0_1(:,4))*dose;
sort_day2_age1_2 = sort(serv_day2_age1_2(:,4))*dose;
sort_day2_age2_5 = sort(serv_day2_age2_5(:,4))*dose;
sort_day2_age6_12 = sort(serv_day2_age6_12(:,4))*dose;
sort_day2_age13_19 = sort(serv_day2_age13_19(:,4))*dose;
sort_day2_age20_up = sort(serv_day2_age20_up(:,4))*dose;
sort_day2_age2up = sort(serv_day2_age2up(:,4))*dose;
%
sort_day2_age0_1_d2 = sort_day2_age0_1(round(.9*size(sort_day2_age0_1)));
sort_day2_age1_2_d2 = sort_day2_age1_2(round(.9*size(sort_day2_age1_2)));
sort_day2_age2_5_d2 = sort_day2_age2_5(round(.9*size(sort_day2_age2_5))); % 90th
percentile
sort_day2_age6_12_d2 = sort_day2_age6_12(round(.9*size(sort_day2_age6_12)));
sort_day2_age13_19_d2 = sort_day2_age13_19(round(.9*size(sort_day2_age13_19)));
sort_day2_age20_up_d2 = sort_day2_age20_up(round(.9*size(sort_day2_age20_up)));
sort_day2_age2up_d2 = sort_day2_age2up(round(.9*size(sort_day2_age2up)));
%
% calculate means of day 1 and day 2 90th percentiles
disp('Mean 90th percentiles')
M90_age0_1 = (sort_day1_age0_1_d1 + sort_day2_age0_1_d2)/2
M90_age1_2 = (sort_day1_age1_2_d1 + sort_day2_age1_2_d2)/2
M90_age2_5 = (sort_day1_age2_5_d1 + sort_day2_age2_5_d2)/2
M90_age6_12 = (sort_day1_age6_12_d1 +  sort_day2_age6_12_d2)/2
M90_age13_19 = (sort_day1_age13_19_d1 + sort_day2_age13_19_d2)/2
M90_age20_up = (sort_day1_age20_up_d1 + sort_day2_age20_up_d2)/2
```

```
M90_age2up = (sort_day1_age2up_d1 + sort_day2_age2up_d2)/2
% now do body mass calculations
% find masses of each age group
mass0_1 = searchnzero(age0_1,mass);
mass1_2 = searchnzero(age1_2,mass);
mass2_5 = searchnzero(age2_5,mass);
mass6_12 = searchnzero(age6_12,mass);
mass13_19 = searchnzero(age13_19,mass);
mass20_up = searchnzero(age20_up,mass);
mass2up = searchnzero(age2up,mass);
% calculate the mean masses of each age group
disp('Mean mass of each age group')
age_group_masses =
[mean(mass0_1(:,2)),mean(mass1_2(:,2)),mean(mass2_5(:,2)),mean(mass6_12(:,2)),mean(ma
ss13_19(:,2)),mean(mass20_up(:,2)),mean(mass2up(:,2))]
allmass_ages=[mean(mass0_1(:,2));mean(mass1_2(:,2));mass2_5(:,2);mass6_12(:,2);mass13_
19(:,2);mass20_up(:,2);mean(mass2up(:,2))];
disp('Total number of subjects in all mass/age groups')
size(allmass_ages)
disp('Mean weight of all mass/age groups')
mean(allmass_ages)
disp('Percent Users')
disp('age 0-1')
(size(day1_fc_age0_1u)/size(ind_age0_1))*100
disp('age 1-2')
(size(day1_fc_age1_2u)/size(ind_age1_2))*100
disp('age 2-5')
(size(day1_fc_age2_5u)/size(ind_age2_5))*100
disp('age 6-12')
(size(day1_fc_age6_12u)/size(ind_age6_12))*100
disp('age 13-19')
(size(day1_fc_age13_19u)/size(ind_age13_19))*100
disp('age 20+')
(size(day1_fc_age20_upu)/size(ind_age20_up))*100
disp('age 2+')
(size(day1_fc_age2_5u)+size(day1_fc_age6_12u)+size(day1_fc_age13_19u)+size(day1_fc_ag
e20_upu))/size(ind_age2up)*100
% now adjust the mean EDI and 90th %-tile for weight
disp('Weight-based Mean EDIs')
disp('age 0-1')
medi0_1/mean(mass0_1(:,2))
disp('age 1-2')
```

```
medi1_2/mean(mass1_2(:,2))
disp('age 2-5')
medi2_5/mean(mass2_5(:,2))
disp('age 6-12')
medi6_12/mean(mass6_12(:,2))
disp('age 13-19')
medi13_19/mean(mass13_19(:,2))
disp('age 20+')
medi20up/mean(mass20_up(:,2))
disp('age 2+')
medi2up/mean(mass2up(:,2))
%
disp('Weight-based 90th Percentile EDIs')
disp('age 0-1')
M90_age0_1/mean(mass0_1(:,2))
disp('age 1-2')
M90_age1_2/mean(mass1_2(:,2))
disp('age 2-5')
M90_age2_5/mean(mass2_5(:,2))
disp('age 6-12')
M90_age6_12/mean(mass6_12(:,2))
disp('age 13-19')
M90_age13_19/mean(mass13_19(:,2))
disp('age 20+')
M90_age20_up/mean(mass20_up(:,2))
disp('age 2+')
M90_age2up/mean(mass2up(:,2))
% At this point it would be good to print out the food codes and serving
% sizes used to produce these data.  Also the dose of proposed additive.
disp('Dose of Proposed Additive')
dose
disp('Food Codes and Serving Sizes Used to Produce These Data')
serving_sizes
zarray = mean(allmass_ages);     % just to put something into the output variable and avoid an
error
end

function zarray = searchnzero(age_group,dietary_data)
% This function takes the SEQN numbers for an age group for an EDI
% calculation in NHANES and zeros out rows not in that age group
zarray = dietary_data;       % initialize output array
c=size(dietary_data(:,1));   % setting counter maximum
```

```
c=c(1);                   % first number is loop size needed
for i=1:c
   if any(dietary_data(i,1) == age_group);  % compare subject SEQNs
      a=1;                % do nothing
   else
      zarray(i,:) = 0;
   end
end
keep=find(zarray(:,1) ~= 0);
zarray = zarray(keep,:);
```

Searchfoods.m a function called by EDI.m

```
function zarray = searchfoods(age_diet,targ_codes)
% This function takes the food groups (targ_codes) targeted for an EDI
% calculation in NHANES and zeros out rows (foods) in a dietary age group
% (age_diet) that are not in the targeted food code list
zarray = age_diet;       % initialize output array
c=size(age_diet(:,2));   % setting counter maximum
c=c(1);                  % first number is loop size needed
for i=1:c
   if any(age_diet(i,2) == targ_codes);  % compare subject SEQNs
      a=1;               % do nothing
   else
      zarray(i,:) = 0;
   end
end
keep=find(zarray(:,2) ~= 0);   % keep foods in the food code list
zarray = zarray(keep,:);
```

Searchfoodcodes.m a function called by EDI.m

```
function [foodnums,longfooddesc] = searchfoodcodes(targ_codes)
% This function takes a 1-D list food codes (targ_codes) targeted for an EDI
% calculation in NHANES and zeros out rows (foods) in the food code list
% (food codes.xls, must be in path) that are not in the targeted food code list
[num,txt,raw] = xlsread('food codes.xls');   % read in food codes
            % column 1=numbers, 2=TEXT CAPS, 3=long text
c=size(num);              % setting counter maximum
c=c(1);                   % first number is loop size needed
zarray = txt(:,3);        % fill output array with long food descriptors
```

```
for i=1:c                  % loop over all food codes
   if any(num(i,1) == targ_codes);  % compare subject SEQNs to target codes
      a=1;                 % if match, do nothing, leave line there
   else
      zarray(i,1) = cellstr('X0');     % else zero it
   end
end
unkeep=find(strcmp(zarray(:,1),'X0'));   % delete these foods in the food code list
tu=transpose(unkeep);                  % vectors must be the same type, row vector
k=zeros(1,c);                        % create indices for set of all food codes
k=k+(1:c);                          % indices are now a row vector
keep=setdiff(k,tu);     % keep the set of foods we are not deleting
zarray = zarray(keep,:);      % by deleting lines with x zeros
longfooddesc = zarray;        % return the long food descriptors
foodnums = num(keep);         % return the corresponding food codes
```

Searchsizes.m a function called by EDI.m

```
function zarray = searchsizes(age_diet_fc,serving_sizes)
% This function takes the food groups (targ_codes) targeted for an EDI
% calculation in NHANES and zeros out rows (foods) in a dietary age group
% (age_diet) that are not in the targeted food code list
zarray = age_diet_fc;        % initialize output array
c=size(age_diet_fc(:,2));    % setting counter maximum
c=c(1);                    % first number is loop size needed
for i=1:c
   if any(age_diet_fc(i,2) == serving_sizes(:,1));  % compare subject SEQNs
       [l,loc_fc]=ismember(age_diet_fc(i,2),serving_sizes(:,1));  % find serving size
       zarray(i,4)=zarray(i,3)/serving_sizes(loc_fc,2);  % scale g to servings
   end
end
```