Supplemental Methods

Isaacs et al., Consumer Product Chemical Weight Fractions from Publicly-Available Ingredient Lists

Curation of Product Ingredient Lists. All available HTML or PDF files for products were identified and downloaded from the source sites, and saved as text using tools provided in the R statistical language (R, 2016) packages 'httr', 'rcurl', and 'XML'. Files were converted to text for analysis. Custom scripts were generated for each data source depending on the structure of the source's product webpage(s). Product names and ingredients were collated into a standardized computable form using custom scripts employing standard R text and character parsing tools. The full reported name of each ingredient was retained along with its numeric rank in the reported list. All of the files collected from Unilever contained the string "The list below displays ingredients in descending order, with those present in highest quantities coming first" and thus all Unilever lists were assumed to follow Case 1 rules. A number of products from Procter and Gamble had colorants specifically listed at the end of the list. These products were noted and handled as described in the main text.

Products that had two individual ingredient lists associated with two or more components (e.g. a dishwasher gel-pack product with a film and liquid component) were removed. In total, ingredient lists that could be handled with the current model were collected for 1123 consumer products, resulting in 24,228 individual ingredient observations. The median number of reported ingredients per product was 21 (maximum 53). The ingredient names used by the manufacturers followed no standard nomenclature (the ACI communication initiative guidelines suggest a mix of naming conventions, including common names). The ingredient list dataset contained 1293 unique names. These names included chemical names (e.g.,"1,2-hexanediol", "2-bromo-2-nitropropane-1,3-diol"), botanical ingredients (e.g., "castor oil", "coconut / jojoba butter"), common names (e.g., "water", "corn starch"), and generic designations (e.g., "fluorescent brightener", "fragrance"). Obvious synonyms (e.g., "water", "aqua") were mapped to harmonized names. Chemical Abstract Service Registration Numbers (CASRN) for ingredient names were obtained where possible from the MSDS database (i.e. in the case where both a name and a CASRN had been co-reported). CASRN could be obtained for 218 ingredient names. While these ingredients represent less than 1 percent of the ingredients in the 1123 products, the chemicals accounted for 60 percent of the 24,228 ingredient observations. The median number of ingredients per product which matched to a CASRN was 13. This was higher than the median number of ingredient per product in the MSDS data (median=6) (Goldsmith et al. 2014).

Categorization of Consumer Products. Each product was categorized as belonging to one of more than 200 product use codes (PUCs.) The PUCs are based on those used in the SHEDS-HT model (Isaacs et al. 2014), but these PUCs were updated to account for new products types added to the MSDS database, refined category definitions, and formulation differences (e.g., sprays vs. gels). The PUCs all belong to one of nine high-level categories (e.g., "Personal Care", "Inside the Home") used by the National Library of Medicine's Household Products Database (NLM 2016). An initial automated categorization was performed by searching for keywords (e.g., "shampoo", "fabric softener") within the product names using custom R and SAS (SAS Institute,

Cary NC) scripts, followed by hand-curation of each individual product category assignment to check for inaccuracies, and one-by-one coding of products with names that could not be categorized using the automated methods. The products represented 81 unique categories (Table S1). A large number of products (N=762, 68%) were associated with personal care categories. The remaining products were associated with general household or cleaning products.

Assignment of Ingredients to Chemical Functions. Where possible, the ingredients were also mapped to chemical functions. Chemical function is a qualitative description of the role or "purpose" that a chemical serves in the formulation. EPA has developed a dataset of chemical function (e.g. "emulsifier", "colorant") information collected from publicly available industry, government, and retailer data sources (Isaacs et al. 2016; Phillips et al. 2016). This dataset (called the Functional Use Database or FUse) contains information for over 14,000 chemicals with Chemical Abstract Service Numbers (CASRNs). In FUse, the compounds are categorized by original reported functional uses and by a set of harmonized function definitions developed for use in exposure modeling and alternatives assessments (Isaacs et al. 2016; Phillips et al. 2016). Harmonized functions for ingredients with CASRN were obtained from FUse where possible. Of the 218 ingredient names matched to a CASRN, 191 could be mapped to chemical function using the FUse database, comprising 19 unique harmonized chemical functions. Functions having the largest number of ingredients were "Viscosity-Controlling/Emulsion Stabilizers/Binding Agents" (25 ingredients) and "Surfactants/Cleansers/Emulsifiers" (19 ingredients). The functions associated with the largest number of ingredient observations were "Perfumes" (2768 occurrences in products) and "Solvents" (1919).

Code for Generating Weight Fraction Predictions

The R code given below provides the full R code for simulating WF based on ingredient list. The code contains a function that takes as arguments the number of ingredients, minimum WF for a reported ingredient, unreported fraction, number of Monte Carlo samples, the shape of the assumed distribution of WF between bounds, and the reporting case. In addition, this code includes an example of calling the function for a MC run of 10000 samples for a product with 10 ingredients.

```
##############################################################################################
# Function for calculating weight fraction in terms of number of ingredients, minimum
WF for a reported ingredient (Fm),
# total unreported fraction (Fu), and number of repetitions
# shape = 1(uniform), 2 (symmetric triangle), 3 (high-weighted triangle)
# type  = 1(all ingredients reported in order), 2(ingredients under WF=0.01 reported
in random order, i.e. FDA rules)

wfractions <- function(N_ingredients, Fm, Fu, N_reps, shape, type) {
  weightf <- matrix(, nrow = N_reps, ncol = N_ingredients)

  #update variable names to fit code
  C_L<-Fu
  L_report<-Fm
```

```
  for (rep in 1:N_reps) {
    for (rank in 1:N_ingredients) {


      if (rank == 1) {
        L_upper <- 1  - C_L - L_report*(N_ingredients-1)
        L_lower <- (1 - C_L) / (N_ingredients)
        if (shape==1) { weightf[rep,rank] <- L_lower + (L_upper-L_lower)*runif(1)}
        if (shape==2) { weightf[rep,rank] <- L_lower + (L_upper-L_lower)*rtriang(1, m
in=0, mode=0.5, max=1)}
        if (shape==3) { weightf[rep,rank] <- L_lower + (L_upper-L_lower)*rtriang(1, m
in=0, mode=1, max=1)}
      }
      else {
        sum <- 0
        n <- rank-1

        for (i in 1:n) {
          sum <- sum + weightf[rep,i]
        }
        #cat("rank: ",rep)
        L_upper <- min(1. - sum - C_L - L_report*(N_ingredients-rank), weightf[rep,ra
nk-1])
        L_lower <- max((1. - sum - C_L) / (N_ingredients - rank+1), L_report)
        if (shape==1) { weightf[rep,rank] <- L_lower + (L_upper-L_lower)*runif(1)}
        if (shape==2) { weightf[rep,rank] <- L_lower + (L_upper-L_lower)*rtriang(1, m
in=0, mode=0.5, max=1)}
        if (shape==3) { weightf[rep,rank] <- L_lower + (L_upper-L_lower)*rtriang(1, m
in=0, mode=1, max=1)}
      }
    }

    #Shuffle the <1% (WF=0.01) for this rep if type 2 (thresholds)
    if (type==2) {
      k<<-weightf[rep,]
      indexit<<-which(k<=.01)
      lessthans<<-weightf[rep,indexit]
      indexit<<-which(k>.01)
      greaterthans<<-weightf[rep,indexit]
      shuffledlessthans<-sample(lessthans) #sample function just randomizes the vecto
r
      #replace the shuffled values back into the weight fraction vector
      weightf[rep,]<-c(greaterthans,shuffledlessthans)
    }
  }
  return(weightf)
}
################################################################################
#Example of running model for 10000 MC samples for a product with "ningred" ingredien
ts, with Fu=0.01, Fm=1e-9, shape=uniform, type=2 (FDA reporting rules)
#The data table "thisrun" contains the results for each sample
#The data table "thisrun_results" contains the resulting percentiles of the run
```

```r
ningred<-10
means<- vector("numeric",ningred)
P95s <- vector("numeric",ningred)
P05s <- vector("numeric",ningred)
P50s <- vector("numeric",ningred)
thisrun<-wfractions(ningred, 1e-9, .01, 10000,1,2)
#demonstrate the predictions sum to 1-Fu
sums<-as.vector(rowSums(thisrun))

for (j in 1:ningred) {
  means[j]<-mean(thisrun[,j])
  P05s[j]<-quantile(thisrun[,j], c(.05))
  P50s[j]<-quantile(thisrun[,j], c(.50))
  P95s[j]<-quantile(thisrun[,j], c(.95))
}
thisrun_results<-cbind(means,P05s,P50s,P95s)
```