Supplementary Material

Supplemental Figure 1.

Nova Classification

The four levels of the NOVA processing classification of food processing is determined by the criteria listed in this handout. This handout is designed to help you classify foods according to their NOVA level based on the definitions created by Monteiro and colleagues.

Group 1: Unprocessed or Minimally Processed Foods

Unprocessed foods: originate from the edible parts of plants, animals, fungi, algae and water after separation from nature.

Minimally processed foods: Unprocessed foods altered by removal of inedible or unwanted parts, drying, crushing grinding, fractioning, roasting, toasting, boiling, pasteurization, refrigeration, freezing, placing in containers, vacuum packaging, non-alcoholic fermentation, and other methods that largely preserve the food matrix and do not add salt, sugar, oils or fats or other food substances to the original food. The main aim of these industrial processes is to extend the life of unprocessed foods, enabling their storage for longer use, and to make their preparation easier or more diverse.

Examples of food Level 1 unprocessed foods:

- Fresh, squeezed, chilled, frozen, or dried fruits and vegetables
- Grains
- Brown, parboiled, or white rice, corn cob or kernel, wheatberry or grain
- Legumes, beans, lentils, chickpeas
- Starchy roots and tubers
- Fungi
- Fresh or dried mushrooms
- Meat, poultry, fish, seafood
- Fresh, powdered, chilled, frozen without added ingredients such as stabilizers or preservatives
- Fresh or pasteurized fruit juice, not from concentrate
- No added sugar, sweeteners, or flavors

Group 2: Processed Culinary Ingredients

Ingredients derived from group 1 foods without altering the content of the food such as pressing milling and drying. These foods are often used in combination with additional foods to make meals palatable, diverse, nourishing, and enjoyable. Found in foods such as soups, broths, salads, breads, preserves, drinks, and desserts. To distinguish between group 1 and 2 ask “was this ingredient made from something else?” For example, butter was made from milk or olive oil from olives.

Group 2 examples:

- Vegetable oils crushed from seeds, nuts, or fruit
- Butter and lard obtained from milk and pork
- Sugar and molasses from cane or beet

Nova Classification

- Naturally obtained honey and syrup
- Vinegar
- Starches from corn and plants
- Salt mined or seawater
- Items consisting of two group two items such as salted butter
- Group 2 items with added vitamins or minerals such as iodized salt

Group 3: Processed Foods

Made by adding salt, oil, sugar, or other substances from group 2 to group 1 foods. Various preservation or cooking methods fall into this category including curing, smoking, canning, salting, and non-alcoholic fermentation to make foods like breads and cheeses. Processes and ingredients here aim to increase the durability of group 1 foods and make them more enjoyable by modifying or enhancing their sensory qualities. Processed foods often contain additives that prolong product duration, protect original properties or prevent proliferation of microorganisms (such as preservatives and antioxidants), but not additives with cosmetic functions.

Examples include:

- Canned or bottled vegetables and legumes in brine
- Salted or sugared nuts and seeds
- Fruits in syrup
- And dried or canned fish
- Breads, cheese, pastries, cakes, cookies (biscuits); sweet or savory snacks; cured meats; and ready-to-heat products such as burgers, and pre-prepared pies and pasta and pizza dishes (when these products are made exclusively from group 1 foods and salt, oil, sugar or other Nova group 2 ingredients and **do not contain classes of additives with cosmetic function**)

Group 4: Ultra-Processed Foods

Industrially manufactured food products made up of several ingredients (formulations) including sugar, oils, fats and salt (generally in combination and in higher amounts than in processed foods) and food substances of no or rare culinary use (such as high-fructose corn syrup, hydrogenated oils, modified starches and protein isolates). Group 1 foods are absent or represent a small proportion of the ingredients in the formulation. Processes enabling the manufacture of ultra-processed foods include industrial techniques such as extrusion, molding and pre-frying; application of additives including those whose function is to make the final product palatable or hyper-palatable such as flavors, colorants, non-sugar sweeteners and emulsifiers; and sophisticated packaging, usually with synthetic materials. Processes and ingredients here are designed to create highly profitable (low-cost ingredients, long shelf-life, emphatic branding), convenient, tasteful alternatives to all other Nova food groups and to freshly prepared dishes and meals.

Nova Classification

Some exceptions to this rule are fortification of flours with nutrients such as iron (commonly ferrous sulfate), folic acid, and thiamine mononitrate (thiamine). These nutrients added alone count as minimally processed foods and belong in group 1 according to NOVA criteria.

Ingredients that constitute ultra-processed foods if present in any ingredient list:

Varieties of sugars:

- Fructose, high fructose corn syrup, fruit juice concentrates, inverted sugars, maltodextrin, dextrose, lactose
- Modified oils
- Hydrogenated, esterified oils

Proteins

- Hydrolyzed proteins, soy protein isolate, gluten, casein, whey, mechanically separated meat

Additives

- Flavors, flavor enhancers, colors, emulsifiers, emulsifying salts, artificial sweeteners, thickeners, foaming, anti-caking, bulking, carbonating, gelling, and glazing agents

- This includes agents such as **pectin, citric acid, calcium chloride, added colors, natural or artificial flavors, soy lecithin, mono- and diglycerides, tricalcium biphosphate, gums (guar, locust bean, xanthan), silicon dioxide, carbon dioxide, cellulose, polyethylene glycol, phosphates, aluminum sulfate, calcium sulfate, potassium chloride, magnesium sulfate, polysorbates, calcium alginate, ammonium alginate, calcium lactate, konjac flour, mineral oil, gum Arabic, beeswax, carrageenan, carnauba wax, castor oil, PES, sorbates, sulfites, fatty acid esters of glycerol, gluconic delta-lactone**

Food examples:

Ready-to-consume products

- Carbonated soft drinks, sweet or savory packaged snacks, chocolate, candies, ice-cream, mass produced bread/buns, margarines and other spreads, cookies, biscuits, pastries, cakes, cake mixes, breakfast cereals with added sugar, cereal and energy bars, energy drinks, milk drinks (including chocolate milk), fruit yogurts, fruit drinks, instant sauces

Ready-to-heat products

- Pies, pasta dishes such as macaroni and cheese, pizza dishes, poultry and fish nuggets, sausages, burgers, hot dogs, other reconstituted meat products, powdered and packaged instant soups, noodles, desserts

Infant formulas, meal replacement drinks

Nova Classification

Fruits

- Banana chips with preservative agents
- Canned or packaged fruit in high fructose corn syrup or with any preservatives added

Vegetables

- Canned vegetables with additives (calcium chloride, EDTA, etc.)

Dairy

- Milk with additives (lactase, tocopherols, etc.)
- Yogurt w/ fruit, sugar, natural flavors, guar gum, any artificial sweeteners such as sucralose, acesulfame potassium, aspartame, saccharin, mannitol, sorbitol, xylitol, stevia
- Shredded cheeses w/ anti-caking agents added
- Soymilk/soy yogurt

Protein

- Canned fish w/ additives, frozen seasoned fish filets
- Peanut butter w/ hydrogenated oils and/or added sugar or sweeteners, flavored nuts/seeds

Grains

- Packaged breads, rolls, bagels, English muffins, tortillas, cereals, pasta with additives

Supplemental Table 1.

5 Ingredient Fresh Fruit Salad Recipe

Make this 5-ingredient fresh fruit salad with nutritious summer fruit in a matter of minutes for a lightened-up version of everyone's favorite side dish!

★★★★★
5 from 1 vote

Prep Time	Total Time
20 mins	20 mins

Course: Appetizer, Dessert, Salad, Snack Cuisine: American
Diet: Low Calorie, Low Fat, Vegan Servings: 20 people
Calories: 75kcal Author: Sammi Ricke Cost: $15.00

Equipment

- knife
- large bowl
- cutting board

Ingredients

- 6 cups watermelon, peeled (1/2 watermelon)
- 4 cups cantaloupe, peeled
- 4 cups green grapes, washed
- 2 cups blueberries, washed
- 2 20 oz cans pineapple chunks (drained) or 3 cups fresh pineapple chunks

Instructions

1. Remove watermelon and cantaloupe rinds. Cut into bite-sized pieces. Add to a large bowl.
   6 cups watermelon, peeled (1/2 watermelon), 4 cups cantaloupe, peeled
2. Add in pineapple chunks, grapes, and blueberries. Gently stir until all the fruit is distributed evenly. Serve immediately or store in refrigerator.
   4 cups green grapes, washed, 2 cups blueberries, washed,
   2 20 oz cans pineapple chunks (drained) or 3 cups fresh pineapple chunks
3. (Optional) Use mini cookie cutters to create cute shapes from the melon to place on top of the finished salad!

Notes

Ingredient Tips - Feel free to substitute other fruits in this salad. It is a very customizable recipe.

Storage Tips - This recipe is best enjoyed right away while the fruit is still fresh. However, if you want to prepare in advance, you can prep all the fruit, and store them in separate airtight containers. Then, when you're ready to eat, combine the ingredients, and enjoy!

Nutrition

Serving: 1serving | Calories: 75kcal | Carbohydrates: 18g | Protein: 1g | Fat: 0g | Saturated Fat: 0g | Cholesterol: 0mg | Sodium: 1mg | Fiber: 1g | Sugar: 16g

Thank You! https://happihomemade.com/5-ingredient-fruit-salad/

5 Minute Strawberry Yogurt

Strawberry Yogurt is quick dessert or breakfast made with only 3 ingredients. Just chop the strawberries, mix with maple syrup and yogurt, and voila.

Prep Time	Total Time
5 mins	5 mins

Course: Dessert Cuisine: Ukrainian Servings: 1 serving
Calories: 182kcal Author: Olena Osipov

iFoodReal

★★★★★
5 from 12 votes

Ingredients

- 1 1/2 cups fresh or frozen ripe strawberries hulled
- 2 tbsp brown sugar or maple syrup
- ½ cup plain Greek or regular yogurt
- Whipped cream optional

Instructions

1. Place strawberries in a glass or serving dish and cut using kitchen shears until mushy. You can also use a food processor if making multiples (don't over process into complete mush) or mash berries with a muddler.
2. Add maple syrup and mix well. Add yogurt and stir gently to combine.
3. Serve immediately. You can adjust sweetness to taste, top with whipped cream, cacao nibs or chocolate chips. I like to keep this strawberry yogurt simple as good quality strawberries are SO good!

Notes

- It's best to use local strawberries in season because they are the sweetest and juiciest. "Cardboard" texture like from the grocery store are not good, you need the juices. Frozen strawberries work very well too, just thaw them for 15-20 minutes first.
- Use sour cream instead of yogurt for a richer but less healthy treat.

Nutrition

Serving: 1cup | Calories: 182 kcal | Carbohydrates: 34g | Protein: 12g |Fat: 1g | Saturated Fat: 1g | Cholesterol: 5mg | Sodium: 40 mg | Fiber: 4g | Sugar: 26g

5 Minute Strawberry Yogurt - https://ifoodreal.com/strawberry-yogurt-recipe/

Sign up here to **save** your favourite recipes!

Supplemental Table 2.

Supplemental Figure 2.

```
# Weekly shopping simulation for meal cost analysis
# Last updated: June 2025

# Setup ----
library(readxl)
library(ggplot2)
library(dplyr)
library(zoo)

# Load the data
data <- read_excel("/Users/Christopher/ Supplementary_Table_2.xlsx")

# Data prep ----
# Calculate weekly container usage percentages
data$`1HH_Container_Percent_Weekly` <-
round(data$`1HH_Serving_Price_Total` / data$Container_Price, 2)
data$`4HH_Container_Percent_Weekly` <-
round(data$`4HH_Serving_Price_Total` / data$Container_Price, 2)

# How many weeks to finish a container
data$`1HH_Weeks_To_Consume_Container` <- round(data$Container_Price /
data$`1HH_Serving_Price_Total`, 2)
data$`4HH_Weeks_To_Consume_Container` <- round(data$Container_Price /
data$`4HH_Serving_Price_Total`, 2)

# Convert to days
data$`1HH_Days_To_Consume_Container` <-
data$`1HH_Weeks_To_Consume_Container` * 7
data$`4HH_Days_To_Consume_Container` <-
data$`4HH_Weeks_To_Consume_Container` * 7


# Main simulation function ----
simulate_shopping <- function(menu_data, household_size = 4) {
  weeks <- 52
  costs <- rep(0, weeks) #sets each week to 0
  inventory <- list()
  waste_log <- data.frame()

  # Outer loop, week by week purchase checks
  for(w in 1:weeks) {
    week_start <- (w - 1) * 7 + 1
    week_end <- min(w * 7, 365)

    week_cost <- 0

    # 1st Loop - Counts food item rows and creates sequence
```

```
    for(i in 1:nrow(menu_data)) {
      item <- menu_data[i,]

      # Skip if data is incomplete (for prices)
      if(anyNA(c(item$Container_Price, item$Shelf_stability_all_opened,
                 item$Shelf_stability_unopened,
item$`4HH_Days_To_Consume_Container`))) {
        next
      }

      item_name <- paste0("item_", i)
      price <- item$Container_Price
      shelf_opened <- item$Shelf_stability_all_opened
      shelf_unopened <- item$Shelf_stability_unopened
      days_to_finish <- item$`4HH_Days_To_Consume_Container`

      # Figure out if it's a single-serve item
      single_serve <- FALSE
      if("Food_Item" %in% names(item) && "Menu" %in% names(item) &&
item$Menu == "CONV") {
        patterns <- c("wrap", "sandwich", "pizza", "burrito", "meal",
"bowl", "kit")
        single_serve <- any(grepl(paste(patterns, collapse="|"),
                                  item$Food_Item, ignore.case=TRUE))
      }

      # Check for spoiled items
      if(item_name %in% names(inventory)) {
        good_containers <- list()

        for(container in inventory[[item_name]]) { #"loops through all
containers
          if(container$expires < week_start && container$amount > 0) {
#checks if expired and if food is left {
            # Log the waste
            waste_log <- rbind(waste_log, data.frame(
              week = w, #When it happened
              item = item_name,
              opened = container$opened, #was it opened
              amount = container$amount,
              value = container$amount * price
            ))
          } else if(container$amount > 0) {
            good_containers[[length(good_containers) + 1]] <- container
          } #Add to good container list if not expired and has food in it
        }
        inventory[[item_name]] <- good_containers #updates inventory
removing expired items
      }

      # Figure out what we need (item_name %in% names(inventory) prevents
crashing
      if(single_serve) {
        need <- household_size
```

```
        have <- 0               #Start assuming we have nothing
        if(item_name %in% names(inventory) &&
length(inventory[[item_name]]) > 0)
          #Loop through each container
          {
          have <- sum(sapply(inventory[[item_name]], function(x)
x$amount))
        }         #Check current inventory and change "have" if we do
        buy <- max(0, need - have) #Mark to buy what's missing
      } else {
        daily_use <- 1 / days_to_finish
        week_need <- daily_use * 7
        have <- 0
        # Check if we have bought the item and then do we currently havec
containers
        if(item_name %in% names(inventory) &&
length(inventory[[item_name]]) > 0)
          #Loop through each container
          {
          have <- sum(sapply(inventory[[item_name]], function(x)
x$amount))
        }
        buy <- ceiling(max(0, week_need - have)) #max prevents purchasing
negative amounts and ceiling rounds up
      }

      # Buy what we need
      if(buy > 0) {
        week_cost <- week_cost + (buy * price)
        #Loop
        if(!(item_name %in% names(inventory))) {
          inventory[[item_name]] <- list()
        }

        #Adds items to inventory and picks location to add +1
        for(j in 1:buy) {
          inventory[[item_name]][[length(inventory[[item_name]]) + 1]] <-
list(
            amount = 1, #Full container
            expires = week_start + shelf_unopened - 1, #Calculate
expiration
            opened = FALSE
          )
        }
      }
    }

    costs[w] <- week_cost #store cost of this week

    # Daily consumption - Loop 2
    for(day in week_start:week_end)
      #Go through food items
      {
      for(i in 1:nrow(menu_data)) {
```

```
        item <- menu_data[i,]
        #Skip missing items
        if(anyNA(c(item$Container_Price, item$Shelf_stability_all_opened,
                   item$Shelf_stability_unopened,
item$`4HH_Days_To_Consume_Container`))) {
          next
        }

        item_name <- paste0("item_", i)
        #Skip items not in inventory or if used up
        if(!(item_name %in% names(inventory)) ||
length(inventory[[item_name]]) == 0) {
          next
        }

        # Check if single serve
        single_serve <- FALSE
        if("Food_Item" %in% names(item)) {
          patterns <- c("wrap", "sandwich", "pizza", "burrito", "meal",
"bowl")
          single_serve <- any(grepl(paste(patterns, collapse="|"),
                                    item$Food_Item, ignore.case=TRUE))
        }

        if(single_serve && day == week_start) {
          # Use single-serve items on first day of week
          used <- 0
          for(j in seq_along(inventory[[item_name]])) {
            if(used >= household_size) break

            container <- inventory[[item_name]][[j]]
            if(container$expires >= day && container$amount > 0) {
              container$amount <- 0
              container$opened <- TRUE
              inventory[[item_name]][[j]] <- container
              used <- used + 1
            }
          }
        } else if(!single_serve) {

           # Regular items
          daily_use <- 1 / item$`4HH_Days_To_Consume_Container`
          needed <- daily_use

          #Loop through 1 container at a time
          for(j in seq_along(inventory[[item_name]])) {
            if(needed <= 0) break

            #creates variable for seperate container of each food item
            container <- inventory[[item_name]][[j]]
            if(container$expires >= day && container$amount > 0) {

              # open containers if needed and adds new shelf stability
              if(!container$opened){
```

```
                container$opened <- TRUE
                container$expires <- day + item$Shelf_stability_all_opened
- 1
              }

              #consume from container (smaller amount of needed or have)
              use <- min(needed, container$amount)
              container$amount <- container$amount - use
              needed <- needed - use

              #save update container info
              inventory[[item_name]][[j]] <- container
            }
          }
        }

        # Remove empty containers
        inventory[[item_name]] <- Filter(function(x) x$amount > 0,
inventory[[item_name]])
      }
    }
  }

  list(costs = costs, waste = waste_log)
}


# Run the analysis ----
run_analysis <- function() {
  # Menu colors
  colors <- c(UPF = "#bbbbbb", MPF = "#36ebab", CONV = "#8853b0")

  # Split by menu type
  upf <- filter(data, Menu == "UPF")
  mpf <- filter(data, Menu == "MPF")
  conv <- filter(data, Menu == "CONV")

  # Run sims
  cat("Running UPF simulation...\n")
  upf_res <- simulate_shopping(upf, 4)

  cat("Running MPF simulation...\n")
  mpf_res <- simulate_shopping(mpf, 4)

  cat("Running CONV simulation...\n")
  conv_res <- simulate_shopping(conv, 4)

  # Prep data for plots
  plot_data <- data.frame(
    week = rep(1:52, 3),
    cost = c(upf_res$costs, mpf_res$costs, conv_res$costs),
    total = c(cumsum(upf_res$costs), cumsum(mpf_res$costs),
cumsum(conv_res$costs)),
    menu = rep(c("UPF", "MPF", "CONV"), each = 52)
```

```
  )

  # Weekly costs plot
  p1 <- ggplot(plot_data, aes(week, cost, color = menu)) +
    geom_line(size = 1.2) +
    scale_color_manual(values = colors) +
    scale_y_continuous(labels = scales::dollar) +
    scale_x_continuous(breaks = seq(0, 52, by=4)) +
    labs(x = "Week of Year", y = "Weekly Cost") +
    theme_minimal() +
    theme(
      legend.position = "bottom",
      legend.title = element_blank(),
      axis.title.x = element_text(size= 14, face = "bold"),
      axis.title.y = element_text(size= 14, face = "bold"),
      axis.text.x = element_text(size = 12, face = "bold"),
      axis.text.y = element_text(size = 12, face = "bold"),
      legend.text = element_text(hjust = 0.5, size = 12, face = "bold"),
      panel.grid.major = element_blank(),  # Remove major grid lines
      panel.grid.minor = element_blank()   # Remove minor grid lines
    )

  print(p1)

  # Annual totals
  totals <- plot_data %>%
    filter(week == 52) %>%
    select(menu, total)

  p2 <- ggplot(totals, aes(reorder(menu, -total), total, fill = menu)) +
    geom_col(width = 0.6) +
    geom_text(aes(label = scales::dollar(total)), vjust = -0.5) +
    scale_fill_manual(values = colors) +
    scale_y_continuous(labels = scales::dollar, expand = c(0, 0, 0.1, 0))
+
    labs(
      title = "Annual Food Costs (4-Person Household)",
      x = "",
      y = "Annual Cost"
    ) +
    theme_minimal() +
    theme(legend.position = "none")

  print(p2)

  # Print summary
  cat("\n--- ANNUAL COSTS ---\n")
  for(i in 1:nrow(totals)) {
    cat(sprintf("%s: %s\n", totals$menu[i],
scales::dollar(totals$total[i])))
  }

  # Cost differences
  cat("\n--- COST DIFFERENCES ---\n")
```

```
  upf_total <- totals$total[totals$menu == "UPF"]
  mpf_total <- totals$total[totals$menu == "MPF"]
  conv_total <- totals$total[totals$menu == "CONV"]

  cat(sprintf("UPF vs MPF: %s (%.1f%% difference)\n",
              scales::dollar(upf_total - mpf_total),
              (upf_total - mpf_total) / mpf_total * 100))

  cat(sprintf("UPF vs CONV: %s (%.1f%% difference)\n",
              scales::dollar(upf_total - conv_total),
              (upf_total - conv_total) / conv_total * 100))

  cat(sprintf("MPF vs CONV: %s (%.1f%% difference)\n",
              scales::dollar(mpf_total - conv_total),
              (mpf_total - conv_total) / conv_total * 100))

  # Waste analysis
  waste_data <- rbind(
    cbind(upf_res$waste, menu = "UPF"),
    cbind(mpf_res$waste, menu = "MPF"),
    cbind(conv_res$waste, menu = "CONV")
  )

  if(nrow(waste_data) > 0) {
    waste_summary <- waste_data %>%
      group_by(menu) %>%
      summarise(total_waste = sum(value))

    cat("\n--- FOOD WASTE ---\n")
    for(i in 1:nrow(waste_summary)) {
      cat(sprintf("%s: %s\n",
                  waste_summary$menu[i],
                  scales::dollar(waste_summary$total_waste[i])))
    }
  }

  invisible(list(
    weekly = plot_data,
    totals = totals,
    waste = waste_data
  ))
}

# Run it
results <- run_analysis()
```