Food Balance Sheets

Wheat

For this example, we'll first consider the full process for creating a food balance sheet for wheat. We start off with an empty table:

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	0	0	0	0	0	0	0	0	0	0	0
Wheat flour	0	0	0	0	0	0	0	0	0	0	0
Bulgur	0	0	0	0	0	0	0	0	0	0	0
Breakfast cereals	0	0	0	0	0	0	0	0	0	0	0
Wheat starch	0	0	0	0	0	0	0	0	0	0	0
Wheat bran	0	0	0	0	0	0	0	0	0	0	0

Production

For production data, we first fill in the table with any available official figures. To impute production, we must also consider yield and area harvested data as yield is defined as production divided by area harvested (and thus with any two elements the third is uniquely defined). Suppose we have the following official data:

Name	Area Harvested	Yield	Production
Wheat	0	0	0
Wheat flour	0	0	0

In this case, the production value is only known for wheat flour (it is missing for wheat), and for wheat we are also missing the yield value. The first step in the imputation process is to impute the yield, using the previously described production imputation methodology.

ADD SOMETHING HERE SHOWING THE IMPUTATION EXAMPLE!!!

Name	Area Harvested	Yield	Production
Wheat	0	0	0
Wheat flour	0	0	0

Now, we have enough information to compute the production data:

Name	Area Harvested	Yield	Production
Wheat	0	0	0
Wheat flour	0	0	0

Next, we fill in the table with our production values. Production is only imputed for primary products (and occassionally official figures are provided for processed products, as is the case here). So, in this case, no additional values are filled in outside of wheat and flour.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	**54420000**	0	0	0	0	0	0	0	0	0	0
Wheat flour	**18650000**	0	0	0	0	0	0	0	0	0	0
Bulgur	**-**	0	0	0	0	0	0	0	0	0	0
Breakfast cereals	**-**	0	0	0	0	0	0	0	0	0	0
Wheat starch	**-**	0	0	0	0	0	0	0	0	0	0
Wheat bran	**-**	0	0	0	0	0	0	0	0	0	0

Trade

For this example, we take the country totals of all imports and exports and insert into this table.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	54420000	**1999100**	**32790000**	0	0	0	0	0	0	0	0
Wheat flour	18650000	**341500**	**572800**	0	0	0	0	0	0	0	0
Bulgur	-	**182500**	**580000**	0	0	0	0	0	0	0	0
Breakfast cereals	-	**312500**	**217300**	0	0	0	0	0	0	0	0
Wheat starch	-	**624900**	**224500**	0	0	0	0	0	0	0	0
Wheat bran	-	**258900**	**2343700**	0	0	0	0	0	0	0	0

Stock Changes

We now estimate the stock changes. Note that for most products, we assume that countries do not hold stocks. Generally, stocks will only be held for primary level products, and not even all of these products. The numbers below represent the estimated stock changes (by the stock imputation methodology described previously) for the example country we're considering.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	54420000	1999100	32790000	**-230600**	0	0	0	0	0	0	0
Wheat flour	18650000	341500	572800	**-**	0	0	0	0	0	0	0
Bulgur	-	182500	580000	**-**	0	0	0	0	0	0	0
Breakfast cereals	-	312500	217300	**-**	0	0	0	0	0	0	0
Wheat starch	-	624900	224500	**-**	0	0	0	0	0	0	0
Wheat bran	-	258900	2343700	**-**	0	0	0	0	0	0	0

Food

The allocation to food, on the other hand, can potentially be considered at any processing level, although some commodities (such as wheat) are assumed to not be eaten as such. We impute food consumption numbers for the example country and update the SUA table below.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	54420000	1999100	32790000	-230600	**-**	0	0	0	0	0	0
Wheat flour	18650000	341500	572800	-	**18540000**	0	0	0	0	0	0
Bulgur	-	182500	580000	-	**3700**	0	0	0	0	0	0
Breakfast cereals	-	312500	217300	-	**98100**	0	0	0	0	0	0
Wheat starch	-	624900	224500	-	**-**	0	0	0	0	0	0
Wheat bran	-	258900	2343700	-	**-**	0	0	0	0	0	0

Feed

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	54420000	1999100	32790000	-230600	-	0	**4898000**	0	0	0	0
Wheat flour	18650000	341500	572800	-	18540000	0	**-**	0	0	0	0
Bulgur	-	182500	580000	-	3700	0	**-**	0	0	0	0
Breakfast cereals	-	312500	217300	-	98100	0	**-**	0	0	0	0
Wheat starch	-	624900	224500	-	-	0	**-**	0	0	0	0
Wheat bran	-	258900	2343700	-	-	0	**3355500**	0	0	0	0

Losses

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	54420000	1999100	32790000	-230600	-	0	4898000	0	0	0	**560300**
Wheat flour	18650000	341500	572800	-	18540000	0	-	0	0	0	**-**
Bulgur	-	182500	580000	-	3700	0	-	0	0	0	**-**
Breakfast cereals	-	312500	217300	-	98100	0	-	0	0	0	**-**
Wheat starch	-	624900	224500	-	-	0	-	0	0	0	**-**
Wheat bran	-	258900	2343700	-	-	0	3355500	0	0	0	**-**

Seed

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	54420000	1999100	32790000	-230600	-	0	4898000	**1904200**	0	0	560300
Wheat flour	18650000	341500	572800	-	18540000	0	-	**-**	0	0	-
Bulgur	-	182500	580000	-	3700	0	-	**-**	0	0	-
Breakfast cereals	-	312500	217300	-	98100	0	-	**-**	0	0	-
Wheat starch	-	624900	224500	-	-	0	-	**-**	0	0	-
Wheat bran	-	258900	2343700	-	-	0	3355500	**-**	0	0	-

Industrial Utilization

For most commodities, industrial utilization will be zero. This element can be important when considering commodities related to biofuels and vegetable oils, but for wheat it is irrelevant.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	54420000	1999100	32790000	-230600	-	0	4898000	1904200	0	**-**	560300
Wheat flour	18650000	341500	572800	-	18540000	0	-	-	0	**-**	-
Bulgur	-	182500	580000	-	3700	0	-	-	0	**-**	-
Breakfast cereals	-	312500	217300	-	98100	0	-	-	0	**-**	-
Wheat starch	-	624900	224500	-	-	0	-	-	0	**-**	-
Wheat bran	-	258900	2343700	-	-	0	3355500	-	0	**-**	-

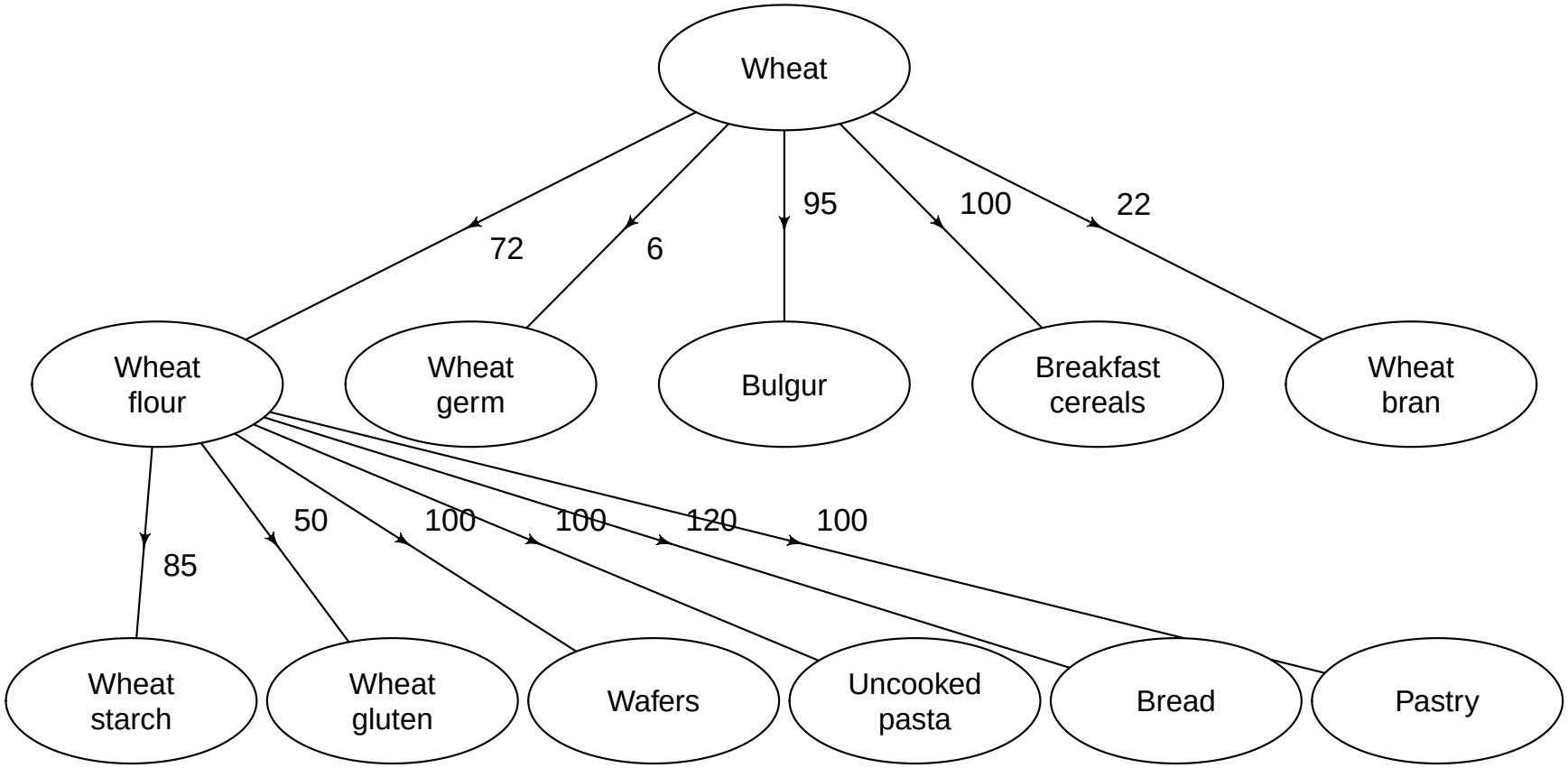
Tourist Consumption

The tourist consumption estimation approach uses tourist data from the WTO as well as last year's consumption patterns to estimate the impact of tourism on local consumption. Note that tourist consumption can be negative; as an extreme example consider a case where many nationals travel abroad but no tourists enter. In this case, the country will have a negative "tourist consumption" because more calories will be consumed abroad than locally.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	54420000	1999100	32790000	-230600	-	0	4898000	1904200	**65**	-	560300
Wheat flour	18650000	341500	572800	-	18540000	0	-	-	**-29200**	-	-
Bulgur	-	182500	580000	-	3700	0	-	-	**-**	-	-
Breakfast cereals	-	312500	217300	-	98100	0	-	-	**-**	-	-
Wheat starch	-	624900	224500	-	-	0	-	-	**-**	-	-
Wheat bran	-	258900	2343700	-	-	0	3355500	-	**-**	-	-

Standardization and Balancing

Now, suppose we have the following commodity tree:



We first start with the pre-standardized table:

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	54420000	1999100	32790000	-230600	-	0	4898000	1904200	65	-	560300
Wheat flour	18650000	341500	572800	-	18540000	0	-	-	-29200	-	-
Bulgur	-	182500	580000	-	3700	0	-	-	-	-	-
Breakfast cereals	-	312500	217300	-	98100	0	-	-	-	-	-
Wheat starch	-	624900	224500	-	-	0	-	-	-	-	-
Wheat bran	-	258900	2343700	-	-	0	3355500	-	-	-	-

We then compute the required "production" of each of the processed products to satisfy any deficits due to exports or consumption (note that we can allow production to be zero if supply exceeds utilization).

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	54420000	1999100	32790000	-230600	-	0	4898000	1904200	65	-	560300
Wheat flour	18650000	341500	572800	-	18540000	0	-	-	-29200	-	-
Bulgur	**401200**	182500	580000	-	3700	0	-	-	-	-	-
Breakfast cereals	**2900**	312500	217300	-	98100	0	-	-	-	-	-
Wheat starch	**0**	624900	224500	-	-	0	-	-	-	-	-
Wheat bran	**5440300**	258900	2343700	-	-	0	3355500	-	-	-	-

Since wheat starch is produced from wheat flour, we would first need to ensure the wheat flour "food to processing" can cover any deficits of wheat starch. However, since wheat starch imports exceed exports plus food, we don't have to worry about this requirement. Instead, we can just standardize all the first processed level products back to food to processing of wheat.

Name	Production (processed)	SD(Production)	Wheat Equivalent	SD(Wheat Equivalent)
Wheat flour	18650000	0	25910000	0
Bulgur	401200	880	422300	930
Breakfast cereals	2900	1500	2900	1500
Wheat bran	5440300	167800	24730000	762600

Now, we wish to compute the distribution for the "food to processing" element for wheat. The main requirement is in the wheat flour and bran, and it should be noted that the 26 million kilogram requirement for wheat flour will automatically be satisfied if the 35 million kilogram requirement for wheat bran is satisfied (as they are produced together). Thus, the food to processing element for wheat has a mean of 35 million kilograms (the sum of the last three) and a standard deviation of 2.55 million kilograms (the square-root of the sum of the squares of the last three standard deviations). Thus, we now have the following table:

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	54420000	1999100	32790000	-230600	-	**26330000**	4898000	1904200	65	-	560300
Wheat flour	18650000	341500	572800	-	18540000	0	-	-	-29200	-	-
Bulgur	401200	182500	580000	-	3700	0	-	-	-	-	-
Breakfast cereals	2900	312500	217300	-	98100	0	-	-	-	-	-
Wheat starch	0	624900	224500	-	-	0	-	-	-	-	-
Wheat bran	5440300	258900	2343700	-	-	0	3355500	-	-	-	-

Now, we must balance this table. To do this, we need to extract the computed standard deviations of each element. The table below shows the expected value and estimated standard deviation for each of the elements for wheat:

```
## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"
```

```
## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in `[.data.table`(printMean, is.na(Value), `:=`(Value, 0)):
## Coerced 'double' RHS to 'character' to match the column's type; may have
## truncated precision. Either change the target column to 'double' first (by
## creating a new 'double' vector length 21 (nrows of entire table) and
## assign that; i.e. 'replace' column), or coerce RHS to 'character' (e.g.
## 1L, NA_[real|integer]_, as.*, etc) to make your intent clear and for
## speed. Or, set the column type correctly up front when you create the
## table and stick to it, please.

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"
```

```
## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"
```

```
## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"
```

Variable	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Waste	Seed	Industrial	Tourist
Mean	54420000	1999100	32790000	-230600	0	26330000	4898000	560300	1904200	0	65
Standard Dev.	544188	0	0	89854	NA	1749	244900	56031	1129	NA	7

Note that in this case, the standard deviation for food for processing is very small because the flour production is an official figure (and this is the main use of wheat). Thus, the "food for processing" element is not adjusted much.

Variable	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Waste	Seed	Industrial	Tourist
Mean	62350000	1999100	32790000	-446800	0	26330000	3292200	476300	1904200	0	65
Standard Dev.	544188	0	0	89854	NA	1749	244900	56031	1129	NA	7

Now, when balancing, we find that food for processing is adjusted down slightly. This adjustment to food of wheat implies that the production of children commodities must also be updated (and hence their food values as well).

Name	Production (processed)	SD(Production)	Wheat Equivalent	SD(Wheat Equivalent)	Adjustment

Name	Production (processed)	SD(Production)	Wheat Equivalent	SD(Wheat Equivalent)	Adjustment
Wheat flour	18650000	0	25910000	0	0
Bulgur	401200	880	422300	930	0
Breakfast cereals	2900	1500	2900	1500	0
Wheat bran	5440300	167800	24730000	762600	-80

We can now update the production numbers for each of the first level primary elements. Note that in the process of creating flour, we also create bran and germ. The amount of bran and germ created, in this case, is determined by the amount of flour we need to create (as that was our most stringent requirement). Thus, we have:

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	62350000	1999100	32790000	-446800	0	26330000	3292200	1904200	65	0	476300
Wheat flour	18650000	341500	572800	-	18540000	0	-	-	-29200	-	-
Wheat germ	**1554300**	**0**	**0**	**0**	**0**	**0**	**1554300**	**0**	**0**	**0**	**0**
Bulgur	**401200**	182500	580000	0	**3700**	0	0	0	0	0	0
Breakfast cereals	**3000**	312500	217300	0	**98200**	0	0	0	0	0	0
Wheat starch	0	624900	224500	-	-	0	-	-	-	-	-
Wheat bran	**5699300**	258900	2343700	0	0	0	**3614500**	0	0	0	0

Our food balance sheet is nearly completed, except that some commodities haven't been handled yet. In particular, wheat starch had imports exceeding exports and so we have not balanced that commodity yet; also, wheat flour has official production and so we haven't modified that commodity either. These unbalanced elements must be updated, and since the production is already fixed (either because it's an official figure or because it's 0) the balancing is very straight-forward: the uncertainty will be entirely allocated to food (or, in general, to either food or feed).

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	62350000	1999100	32790000	-446800	0	26330000	3292200	1904200	65	0	476300
Wheat flour	18650000	341500	572800	-	**18450000**	0	-	-	-29200	-	-
Wheat germ	1554300	0	0	0	0	0	1554300	0	0	0	0
Bulgur	401200	182500	580000	0	3700	0	0	0	0	0	0
Breakfast cereals	3000	312500	217300	0	98200	0	0	0	0	0	0
Wheat starch	0	624900	224500	-	**-**	0	-	-	-	-	-
Wheat bran	5699300	258900	2343700	0	0	0	3614500	0	0	0	0

Now, the final step is aggregating this full table back into primary equivalent. For most elements, this is trivial: for example, the final stock change for wheat will simply be the current stock change because there is no stock change for processed products. However, there are three elements that must be handled differently: imports, exports, and food. Note that the final value for wheat equivalent production is simply the current value for wheat production: this is because "production" of flour (or any other processed product) isn't really production in the sense that the flour is acquired from a different commodity (whereas production of wheat is truly a production as it is not derived from anything else). Also, food processing will not be standardized as it is more of an accounting variable that specifies how much of a commodity at one level should be processed into a different commodity.

To standardize trade and food, we can simply aggregate the trade and food of the children commodities up into their primary equivalent by dividing by the extraction rate. We add these primary equivalents to the current value of trade/food of wheat, and we have our final, primary equivalent trade/food of wheat. Also, feed is not standardized back into wheat equivalent as it is accounted for **???**.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Wheat	62350000	3999200	34780000	-446800	25730000	26330000	3292200	1904200	-40500	0	476300

We can also compute calories, fats, and proteins at this point. First, we apply a calorie/fat/protein content factor to each individual element:

Name	Quantity	Energy	Protein	Fat
Wheat	0.000	1420.937	12.3400	1.86500
Wheat flour	18449983.700	1472.172	11.0475	1.33875
Wheat germ	0.000	NA	NA	NA
Bulgur	3706.173	NA	NA	NA
Breakfast cereals	98189.422	NA	NA	NA
Wheat starch	NA	NA	NA	NA
Wheat bran	0.000	NA	NA	NA

Standardization is trivial: all the commodities here are purely additive, so the standardized calories/fats/proteins are simply the sum of the total calories/fats/proteins for each element:

Energy (millions)	Protein (millions)	Fat (millions)
27161.56	203.83	24.7

Sugar

Now, let's consider the full process for creating a food balance sheet for sugar. We start off with an empty table:

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar Beet	0	0	0	0	0	0	0	0	0	0	0
Sugar Cane	0	0	0	0	0	0	0	0	0	0	0
Sugar and Syrups nes	0	0	0	0	0	0	0	0	0	0	0
Beet sugar	0	0	0	0	0	0	0	0	0	0	0
Refined sugar	0	0	0	0	0	0	0	0	0	0	0
Molasses	0	0	0	0	0	0	0	0	0	0	0

Production

For production data, we first fill in the table with any available official figures. In this case, the production value is known for all the primary products and thus no imputation is done. We also have production data for some of the processed commodities:

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar Beet	**26210000**	0	0	0	0	0	0	0	0	0	0
Sugar Cane	**26510000**	0	0	0	0	0	0	0	0	0	0
Sugar and Syrups nes	**-**	0	0	0	0	0	0	0	0	0	0
Beet sugar	**4561000**	0	0	0	0	0	0	0	0	0	0
Refined sugar	**-**	0	0	0	0	0	0	0	0	0	0
Molasses	**2075000**	0	0	0	0	0	0	0	0	0	0

Trade

For the next example, we'll show how the imputation, mirroring and balancing works. In this case, we just take the country totals and insert into this table.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar Beet	26210000	**194500**	**300**	0	0	0	0	0	0	0	0
Sugar Cane	26510000	**9700**	**860**	0	0	0	0	0	0	0	0
Sugar and Syrups nes	-	**265400**	**96200**	0	0	0	0	0	0	0	0
Beet sugar	4561000	**10**	**194800**	0	0	0	0	0	0	0	0
Refined sugar	-	**1275200**	**111200**	0	0	0	0	0	0	0	0
Molasses	2075000	**464200**	**236500**	0	0	0	0	0	0	0	0

Stock Changes

We now estimate the stock changes. Note that for most products, we assume that countries do not hold stocks. Generally, stocks will only be held for primary level products, and not even all of these products. The numbers below represent the estimated stock changes (by the stock imputation methodology described previously) for the example country we're considering.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar Beet	26210000	194500	300	**-**	0	0	0	0	0	0	0
Sugar Cane	26510000	9700	860	**-**	0	0	0	0	0	0	0
Sugar and Syrups nes	-	265400	96200	**-**	0	0	0	0	0	0	0
Beet sugar	4561000	10	194800	**-**	0	0	0	0	0	0	0
Refined sugar	-	1275200	111200	**79500**	0	0	0	0	0	0	0
Molasses	2075000	464200	236500	**-**	0	0	0	0	0	0	0

Food

The allocation to food, on the other hand, can potentially be considered at any processing level, although some commodities (such as wheat) are assumed to not be eaten as such. We impute food consumption numbers for the example country and update the FBS table below.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar Beet	26210000	194500	300	-	-	0	0	0	0	0	0
Sugar Cane	26510000	9700	860	-	-	0	0	0	0	0	0
Sugar and Syrups nes	-	265400	96200	-	**23000**	0	0	0	0	0	0
Beet sugar	4561000	10	194800	-	-	0	0	0	0	0	0
Refined sugar	-	1275200	111200	79500	**8800000**	0	0	0	0	0	0
Molasses	2075000	464200	236500	-	-	0	0	0	0	0	0

Feed

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar Beet	26210000	194500	300	-	-	0	**-**	0	0	0	0
Sugar Cane	26510000	9700	860	-	-	0	**-**	0	0	0	0
Sugar and Syrups nes	-	265400	96200	-	23000	0	**-**	0	0	0	0
Beet sugar	4561000	10	194800	-	-	0	**-**	0	0	0	0
Refined sugar	-	1275200	111200	79500	8800000	0	**-**	0	0	0	0

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Molasses	2075000	464200	236500	-	-	0	**9023600**	0	0	0	0

Losses

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar Beet	26210000	194500	300	-	-	0	-	0	0	0	**205500**
Sugar Cane	26510000	9700	860	-	-	0	-	0	0	0	**213300**
Sugar and Syrups nes	-	265400	96200	-	23000	0	-	0	0	0	**-**
Beet sugar	4561000	10	194800	-	-	0	-	0	0	0	**-**
Refined sugar	-	1275200	111200	79500	8800000	0	-	0	0	0	**-**
Molasses	2075000	464200	236500	-	-	0	9023600	0	0	0	**-**

Seed

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar Beet	26210000	194500	300	-	-	0	-	**-**	0	0	205500
Sugar Cane	26510000	9700	860	-	-	0	-	**1572200**	0	0	213300
Sugar and Syrups nes	-	265400	96200	-	23000	0	-	**-**	0	0	-
Beet sugar	4561000	10	194800	-	-	0	-	**-**	0	0	-
Refined sugar	-	1275200	111200	79500	8800000	0	-	**-**	0	0	-
Molasses	2075000	464200	236500	-	-	0	9023600	**-**	0	0	-

Industrial Utilization

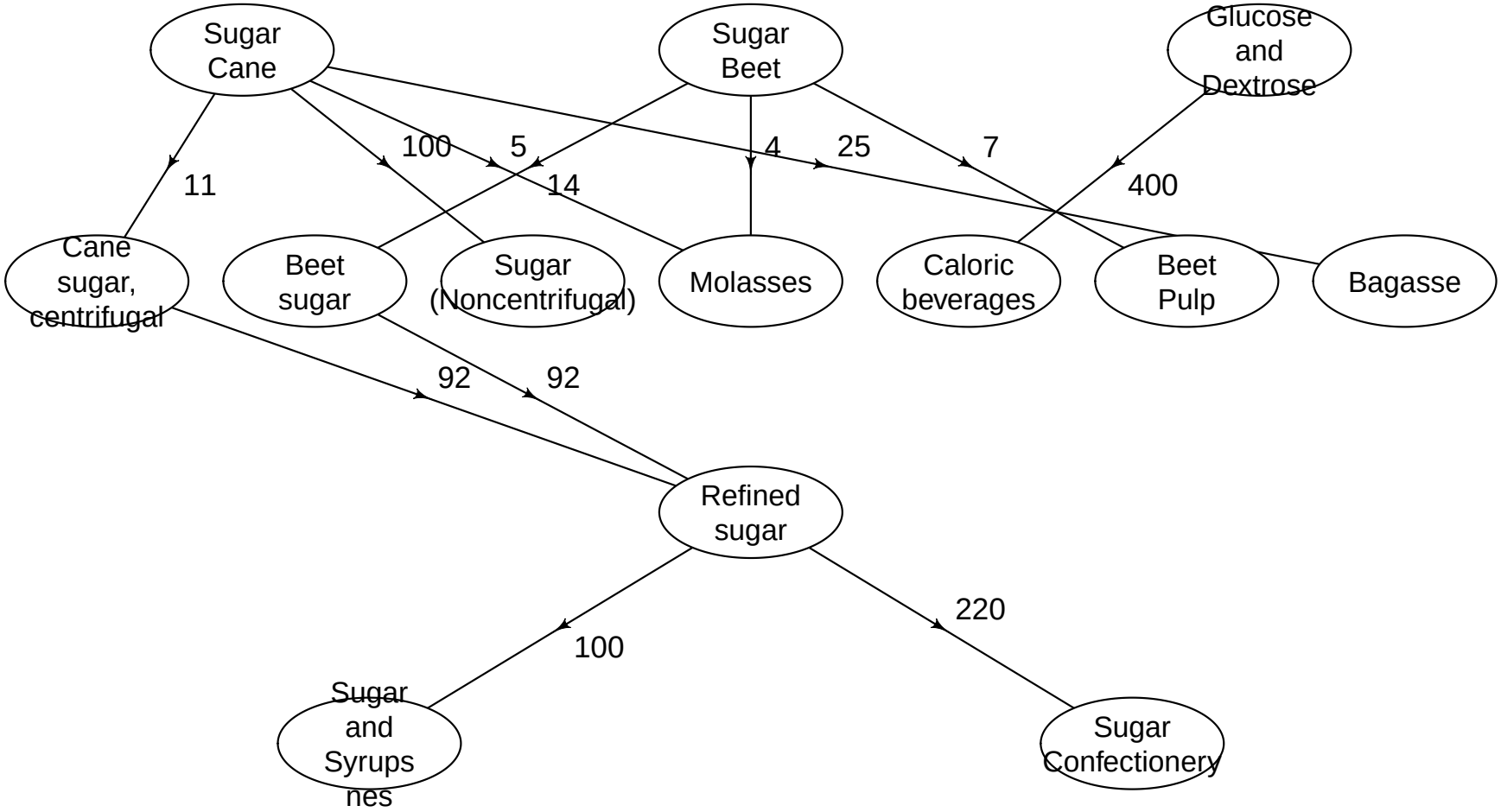
Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar Beet	26210000	194500	300	-	-	0	-	-	0	**-**	205500
Sugar Cane	26510000	9700	860	-	-	0	-	1572200	0	**-**	213300
Sugar and Syrups nes	-	265400	96200	-	23000	0	-	-	0	**-**	-
Beet sugar	4561000	10	194800	-	-	0	-	-	0	**-**	-
Refined sugar	-	1275200	111200	79500	8800000	0	-	-	0	**-**	-
Molasses	2075000	464200	236500	-	-	0	9023600	-	0	**-**	-

Tourist Consumption

The tourist consumption estimation approach uses tourist data from the WTO as well as last year's consumption patterns to estimate the impact of tourism on local consumption. Note that tourist consumption can be negative; as an extreme example consider a case where many nationals travel abroad but no tourists enter. In this case, the country will have a negative "tourist consumption" because more calories will be assumed abroad than locally.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar Beet	26210000	194500	300	-	-	0	-	-	**0**	-	205500
Sugar Cane	26510000	9700	860	-	-	0	-	1572200	**70**	-	213300
Sugar and Syrups nes	-	265400	96200	-	23000	0	-	-	**-**	-	-
Beet sugar	4561000	10	194800	-	-	0	-	-	**-**	-	-
Refined sugar	-	1275200	111200	79500	8800000	0	-	-	**-3000**	-	-
Molasses	2075000	464200	236500	-	-	0	9023600	-	**0**	-	-
### Standardization and	Balancing										

Now, suppose we have the following commodity tree:



We first start with the pre-standardized table:

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar Beet	26210000	194500	300	-	-	0	-	-	0	-	205500
Sugar Cane	26510000	9700	860	-	-	0	-	1572200	70	-	213300
Sugar and Syrups nes	-	265400	96200	-	23000	0	-	-	-	-	-
Beet sugar	4561000	10	194800	-	-	0	-	-	-	-	-
Refined sugar	-	1275200	111200	79500	8800000	0	-	-	-3000	-	-
Molasses	2075000	464200	236500	-	-	0	9023600	-	0	-	-

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss

The processing for sugar works slightly differently than most commodities. Sugar cane and sugar beet are, in almost every case, converted into cane sugar or beet sugar (i.e. they are not eaten as such nor are they processed into other products). Thus, in this case, rather than standardizing values back to sugar cane and sugar beet, we instead assume all sugar cane and sugar beet is first converted into the corresponding sugar and we perform the balances at this level.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar and Syrups nes	-	265400	96200	-	23000	0	-	-	-	-	-
Cane sugar, centrifugal	**2720800**	-	-	-	-	-	-	-	-	-	-
Beet sugar	4561000	10	194800	-	-	0	-	-	-	-	-
Refined sugar	-	1275200	111200	79500	8800000	0	-	-	-3000	-	-
Molasses	2075000	464200	236500	-	-	0	9023600	-	0	-	-
Beet Pulp	**1834200**	-	-	-	-	-	-	-	-	-	-
Bagasse	**6183700**	-	-	-	-	-	-	-	-	-	-

The next step in this process is to balance the processed commodities by creating production values. These production values will require an amount of food processing from the parent commodities. We must start this process at the bottom of the tree, in this case considering "Sugar and Syrups nes" and "Sugar Confectionary" and going up to "Refined Sugar".

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar and Syrups nes	**0**	265400	96200	-	23000	0	-	-	-	-	-
Cane sugar, centrifugal	2720800	-	-	-	-	-	-	-	-	-	-
Beet sugar	4561000	10	194800	-	-	0	-	-	-	-	-
Refined sugar	-	1275200	111200	79500	8800000	**0**	-	-	-3000	-	-
Molasses	2075000	464200	236500	-	-	0	9023600	-	0	-	-
Beet Pulp	1834200	-	-	-	-	-	-	-	-	-	-
Bagasse	6183700	-	-	-	-	-	-	-	-	-	-

Next, we move up the tree to the balancing of refined sugar and the food processing required in the beet and cane sugar elements. We require refined sugar production in order to balance refined sugar, and thus we must create this production from the parent(s) of refined sugar. Looking at the commodity tree, we see that refined sugar can be created from both beet and cane sugar. We will allocate production of refined sugar from these parent commodities according to their availabilities.

Name	Availability	Percent
Beet sugar	4366203	61.6%
Cane sugar, centrifugal	2720837	40.8%

Thus, we allocate according to the parent availabilities:

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar and Syrups nes	0	265400	96200	-	23000	0	-	-	-	-	-
Cane sugar, centrifugal	2720800	-	-	-	-	**3328100**	-	-	-	-	-
Beet sugar	4561000	10	194800	-	-	**5055000**	-	-	-	-	-
Refined sugar	**7712500**	1275200	111200	79500	8800000	0	-	-	-3000	-	-
Molasses	2075000	464200	236500	-	-	0	9023600	-	0	-	-
Beet Pulp	1834200	-	-	-	-	-	-	-	-	-	-
Bagasse	6183700	-	-	-	-	-	-	-	-	-	-

Now, we must balance the primary products in this table (i.e. sugar cane and sugar beet). To do this, we need to extract the computed standard deviations of each element. The table below shows the expected value and estimated standard deviation for sugar beet (top) and sugar cane (bottom):

```
## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"
```

```
## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in `[.data.table`(printMean, is.na(Value), `:=`(Value, 0)):
## Coerced 'double' RHS to 'character' to match the column's type; may have
## truncated precision. Either change the target column to 'double' first (by
## creating a new 'double' vector length 21 (nrows of entire table) and
## assign that; i.e. 'replace' column), or coerce RHS to 'character' (e.g.
## 1L, NA_[real|integer]_, as.*, etc) to make your intent clear and for
## speed. Or, set the column type correctly up front when you create the
## table and stick to it, please.

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"
```

```
## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"

## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"
```

Variable	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Waste	Seed	Industrial	Tourist
Mean	4561000	10	194800	0	0	5164700	0	0	0	0	0
Standard Dev.	0	0	0	NA	NA	44387	NA	NA	NA	NA	NA

```
## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"
```

```
## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"
```

```
## Warning in `[.data.table`(printMean, is.na(Value), `:=`(Value, 0)):
## Coerced 'double' RHS to 'character' to match the column's type; may have
## truncated precision. Either change the target column to 'double' first (by
## creating a new 'double' vector length 11 (nrows of entire table) and
## assign that; i.e. 'replace' column), or coerce RHS to 'character' (e.g.
## 1L, NA_[real|integer]_, as.*, etc) to make your intent clear and for
## speed. Or, set the column type correctly up front when you create the
## table and stick to it, please.
```

```
## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"
```

```
## Warning in formatC(x, big.mark = ",", format = "s"): coercing argument to
## "character" for format="s"
```

Variable	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Waste	Seed	Industrial	Tourist
Mean	2720800	0	0	0	0	3218400	0	0	0	0	0
Standard Dev.	17453	NA	NA	NA	NA	44387	NA	NA	NA	NA	NA

After balancing the above tables, we're left with the following values. Note that only food processing is the element that receives most of the adjustment because it has a substantially higher variability.

Variable	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Waste	Seed	Industrial	Tourist
Mean	4561000	10	194800	0	0	4366200	0	0	0	0	0

Variable	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Waste	Seed	Industrial	Tourist
Standard Dev.	0	0	0	0	0	44387	0	0	0	0	0

Variable	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Waste	Seed	Industrial	Tourist
Mean	2787500	0	0	0	0	2787500	0	0	0	0	0
Standard Dev.	17453	0	0	0	0	44387	0	0	0	0	0

We must now process the changes in the current elements down the tree into the other elements.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar and Syrups nes	0	265400	96200	-	23000	0	-	-	-	-	-
Cane sugar, centrifugal	2787500	-	-	-	-	2787500	-	-	-	-	-
Beet sugar	4561000	10	194800	0	0	4366200	0	0	0	0	0
Refined sugar	6581400	1275200	111200	-1051500	8800000	0	0	0	-3100	0	0
Molasses	2075000	464200	236500	-	-	0	9023600	-	0	-	-
Beet Pulp	1834200	-	-	-	-	-	-	-	-	-	-
Bagasse	6183700	-	-	-	-	-	-	-	-	-	-

Lastly, some elements have not yet been updated in this process. To ensure a full balance of the SUA, we should go through and balance those rows as well.

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Sugar and Syrups nes	0	265400	96200	-	169200	0	-	-	-	-	-
Cane sugar, centrifugal	2787500	-	-	-	-	2787500	-	-	-	-	-
Beet sugar	4561000	10	194800	0	0	4366200	0	0	0	0	0
Refined sugar	6581400	1275200	111200	-1051500	8800000	0	0	0	-3100	0	0
Molasses	2075000	464200	236500	-	-	0	9023600	-	0	-	-
Beet Pulp	1834200	-	-	-	-	-	-	-	-	-	-
Bagasse	6183700	-	-	-	-	-	-	-	-	-	-

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Cane sugar, centrifugal	2787500	1674600	225400	-1142900	9749100	2787500	0	0	-3400	0	0

Name	Production	Imports	Exports	StockChange	Food	Food Processing	Feed	Seed	Tourist	Industrial	Loss
Beet sugar	4561000	1674600	420200	-1142900	9749100	4366200	0	0	-3400	0	0
Molasses	2075000	464200	236500	0	0	0	9023600	0	0	0	0
Beet Pulp	1834200	-	-	-	-	-	-	-	-	-	-
Bagasse	6183700	-	-	-	-	-	-	-	-	-	-

We can also compute calories, fats, and proteins at this point. First, we apply a calorie/fat/protein content factor to each individual element:

Name	Quantity	Energy	Protein	Fat
Sugar and Syrups nes	169162.4	NA	NA	NA
Beet sugar	0.0	1699.49	0	0.0
Refined sugar	8800000.0	1699.49	0	0.0
Molasses	NA	1274.11	0	0.1
Undenatured ethyl alcohol (>80%)	NA	NA	NA	NA
Undenatured ethyl alcohol (<=80%)	0.0	NA	NA	NA
Caloric beverages	238504.8	NA	NA	NA

Standardization is trivial: all the commodities here are purely additive, so the standardized calories/fats/proteins are simply the sum of the total calories/fats/proteins for each element:

Energy (millions)	Protein (millions)	Fat (millions)
14955.51	0	0