Assignment 04/06/2016

A matter of taste: visualization of global variation in taste receptor genes

Course: Management of Large-Scale Omics Data [10U19a]
Professor: Prof.Dr. Jan Aerts

Student: Carlos de Lannoy (carlosvictor.delannoy@student.kuleuven.be)
StudentID: r0629998

1. Task description

As one of our classic senses, taste perception takes a highly significant place in many aspects of human life. From already pivotal beginnings, as a means for selection of required nutrient sources and the detection of spoiled or poisonous food [12], it has developed into a defining factor for cultures and humanity in general.

Classically, the human palate has been described as a combination of four basic flavors: sweet, bitter, salty and sour. This set was later extended to include umami (savory, evoked by several L-amino acids) [8]. Arguably, water, fat, kokumi (‘mouthfulness’, evoked by calcium-containing salts) and pungency (spiciness, evoked by capsaicin) are classifiable as flavors too, as specific taste receptors have been found or suggested for each [1]. Together with our other primary senses, these flavors constitute the wide range of culinary experiences we as a species explore on a daily basis.

The development of the human palate is a highly heterogeneous process, in which exposure to tastes in utero and in early life is thought to play a significant role [Nehring 2015]. As such, cultural influences help form the preference for certain tastes. However, it is thought that genetics also plays a major role in this process [1]. In many cases, variations in genes coding for taste receptors have been linked to variations in taste perception. As taste perception and subsequent dietary preference plays a major role in contemporary health issues (e.g. obesity, alcoholism [4,11], cardiovascular disease [5]) and development of novel food products, a better understanding of the geographical distribution of the genetic background of taste perception may provide insights that are of value to multiple industries.The aim of the work presented in this report is to visualize SNP data collected in the framework of the Human Genome Diversity Project, in a way that highlights likeness and differences in taste receptor genes from different continents, countries and populations (in this work, collectively referred to as population units).

2. Design

Three approaches towards achieving insight in the genetic basis of taste perception have been constructed and assessed.

2.1 World map visualization

At first glance, a world map with connections denoting genetic similarities (fig. 1) would seem a logical approach, since genetic variation data is available on geographically distinct groups. This approach also has the advantage of immediate recognizability of explainable patterns; if a certain combination of taste genes (e.g. those related to pungency) is shared by a population unit in a geographically distinct area and the preference for that combination in that area is known to the observer (e.g. popularity of extremely hot curries in the Indian province of Goa), then this relation can be identified easily. In the mock-up for this visualization, nodes represent countries for which data was available. The size of the node is dependent on the number of people available for that group. Through radio buttons, a viewer can switch between division based on continent, country or ethnic group. The user can switch connections involving a certain population on and off by clicking on the nodes. Every line connecting two nodes denotes a common SNP (for which the most common SNP per population unit is selected), while the color of the line denotes the taste group to which the gene belongs. The colors have been selected using the Colorbrewer online tool [3], to maximize discernibility to the human eye. To avoid cluttering of lines and allow finer examination of (groups of) genes associated with specific taste perceptions, a gene selection tool in the form of a Venn diagram is placed to the right of the picture. The user can either click on an outer circle of the diagram to select all genes for a certain flavor, or on smaller circles to select particular genes.

While the world map visualization does give a complete and recognizable overview of the data, it also suffers from inherent efficiency issues. Most importantly, two dimensions are now used to denote spatial information, while this is not the most important information to be displayed. Rather than geographical distribution, the graphic should primarily provide information on genetic (dis)similarities. Furthermore, using one line per shared SNP heavily skews the number of lines towards the bitter part of the spectrum, since many more genes are known to influence bitter perception than other flavors [1] which results in many more SNPs being available. Lastly, the geographical location of the smaller ethnic groups is often not precisely known or knowable, so giving them a place on the map would introduce some arbitrariness.

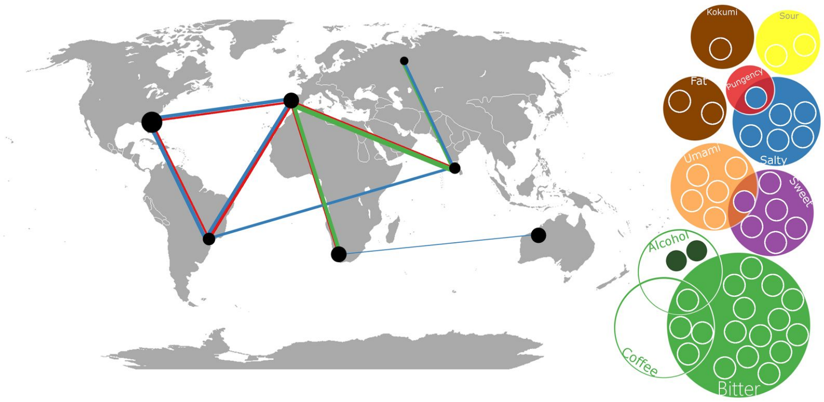


Fig. 1: world map visualization mockup. Lines denote matching SNPs. the selection tool on the right allows distinction between different tastes and genes.

2.2 Radar chart visualization

Since comparison between different populations is key in this visualization, a radar chart (fig. 2) could be implemented to point out those differences in multiple groups of genes simultaneously. Each of the axes in this radar chart denotes a different taste group. The selection tool on the right of the graph allows the user to select a number of populations. For each pair of populations, a line is drawn on the radar chart, which is nearer to the outer border of the graph as the genes associated with the taste on that axis are more alike. The likeness is measured in an adapted version of the identity by state (IBS) genetic relation metric described by Lee [9]; $\mu = \frac{IBS2*}{IBS2* + IBS0}$, in which IBS2* and IBS0 denote the number of heterozygous matches and complete non-matches respectively. Specifically, the IBS metric is calculated using the number of IBS2* and IBS0 for each unique pair of individuals of each group, for the SNPs in requested taste receptor genes. The same is done for all SNPs and the score for the gene under consideration is then divided by the score for all SNPs. This has the added advantage over the regular IBS metric that a score of 1 now denotes a degree of identity that one would expect between the two groups involved. While a score of 1 does not imply that the relation does not result in similar taste perception, it is proposed that an unexpected lack or presence of IBS may lead to new insights. A less than average IBS score is displayed as a line closer to the center of the graph (where the score is 0), while an above average IBS is nearer to the outer rim.

Some issues with this visualization come to mind when the perceived surface area within the line is considered. While surface area generally will increase with an increase in IBS in SNPs for one or more taste groups, this relation is by no means constant. As such, the user should keep in mind that enclosed area does not necessarily mean anything. Furthermore, the order of the taste groups in the graph involves some arbitrariness; there is no reason why e.g. umami is between salty and sweet, nor is there a viable reason to place it in any other order. Lastly, this visualization only allows for one-versus-one comparisons in terms of population units. That is, multiple lines may be added to the graph, but every line will only represent a comparison of two groups.

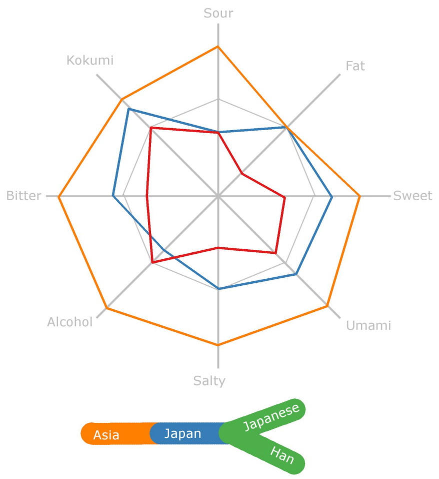


Fig. 2: Radar plot mock-up. Every line denotes a pair of population units, selected using the selection tool below. Clicking on the name of a population unit will give the user the choice to either go to a smaller level of granularity (if possible) or select a second name from the same level of granularity. It is not possible to select population units from different granularities. The orange line denotes a pair that is above averagely similar in their taste genes, while the red line denotes a pair that is remarkably dissimilar. Note that this mock-up does not contain actual data.

2.3 Radial chart visualization

Like the radar chart, the radial chart visualization (fig. 3) makes effective use of its two spatial dimensions. On a circle, nodes are placed representing user-selected population units. The size of the node denotes the IBS score calculated over all pairs within a population unit. The distance between the nodes in degrees is a measure of the IBS between them, calculated as described for the radar chart. The small circle between the main graphs displays the measure of IBS represented by the 180 degrees of the circle (i.e. the maximum distance two population units may be apart). Every time a population unit is added, this number is to be recalculated. Two radial charts are displayed simultaneously. In the right, a line between two nodes represents a likeness of 1 or higher. A thicker line denotes a higher IBS ratio for that gene or gene group. In the left, line thickness increases with decreasing IBS, while the thinnest line still displays a value of 1. The aim of this mechanism is to separate and emphasize any possible genetic background for population units being alike in taste preference (in the similarity graph) or unlike each other (in the dissimilarity graph).

In comparison to the world map and radar chart visualizations, the radial chart involves less arbitrarily made design choices; placement, order and size of the elements on the graphs all have a meaning. Furthermore, the graph is highly flexible, as a selection of more population units or more genes results in the rearrangement of the nodes on the circle and the addition of new lines respectively. However, the radar chart does allow for a clear view of both likeness and unlikeness in one graph. A similar attempt in a radial chart, does not allow such clarity; it is not immediately clear what line thickness denotes a IBS score ratio of 1 if both thicker and thinner lines are in the graph. This problem is mostly solved by displaying the dual charts. Ultimately, the described advantages are thought to justify the selection of the radial chart over the radar chart and map visualization.

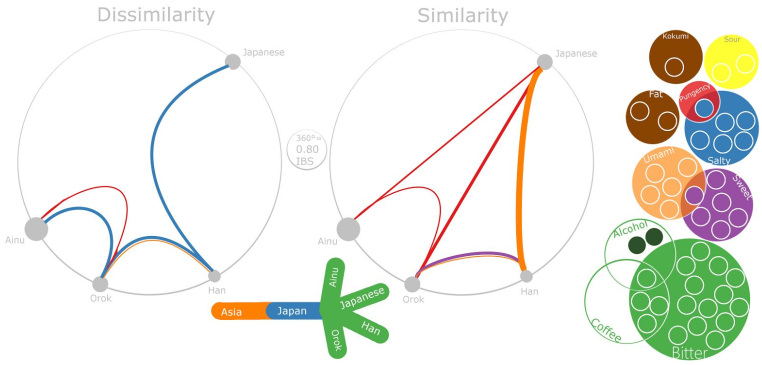


Fig 3: Radial chart mockup. Nodes denote population units. The distance between nodes denotes the genetic similarity between population units. A line between nodes denotes either an above average similarity for a taste gene (left graph) or a below average similarity (right). A measure of overall displayed genetic distance is found between the graphs. The selection tool on the right allows a selection of tastes and taste receptor genes, while the tool below the figure allows selection of population units.

3. Implementation

3.1 Data collection

An initial list of relevant genes per taste was retrieved from Ensembl Biomart, by filtering assembly GRCh38.p5 on GO-terms related to perception of those specific tastes. Genes listed in Bachmanov et al. [1] and Beckett et al. [2] as important to taste perception but not included based on GO-terms were added manually. ASIC1 to 3, identified based on the initial GO-term search were manually removed from the list, due to a lack of evidence in literature. The assignment of the GO-term was possibly due to the role of these genes in acidity sensing, not (conscious) perception of sour tastes. SNP sequences and positions (as defined in human genome build 36.1) were retrieved from the Human Genome Diversity website [6]. A list of relevant gene positions in the correct build was provided by Jan Aerts. As TAS2R30, TAS2R31 and RTP5 were not found in this list, they were added manually from the UCSC genome database [13].

3.2 Data processing and visualization

SNP data was converted into matrices denoting IBS0 and IBS2* scores for every pair of individuals, using R (v 3.3.2, see appendix for script). The stringdist-package (v0.9.4.1) [14] was used to facilitate identification of

IBS0 and IBS2* cases. The code for the visualisation was written in P5.js editor (v2.1). Fig. 4 gives an impression of the layout. A screencast was made to demonstrate its interactive elements (see https://youtu.be/AyV0_UNjmS0 or available upon request from the author).

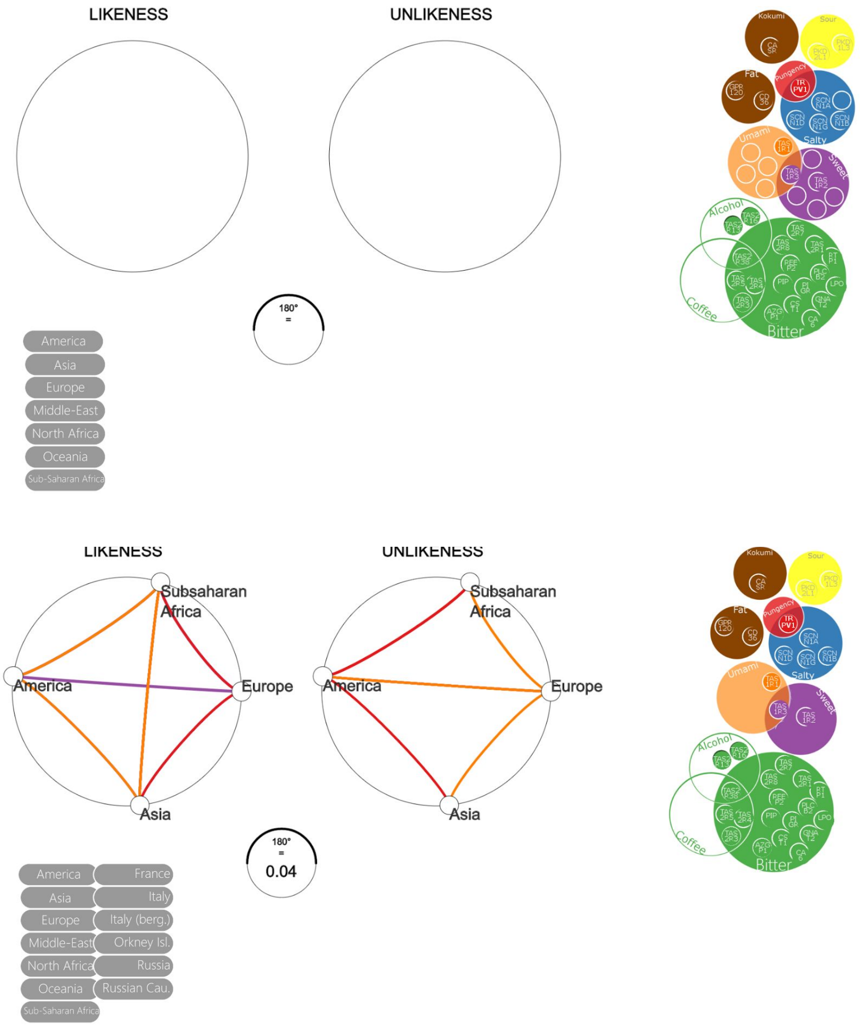


Fig. 4: Impressions of the visualisation layout. The user starts out with a choice of continents to display and is allowed to choose more finely grained population units when a continent is selected. The gene selection tool on the right allows the user to display certain combinations of genes in the graphs. Lines appear color coded in the same way as the selection tool.

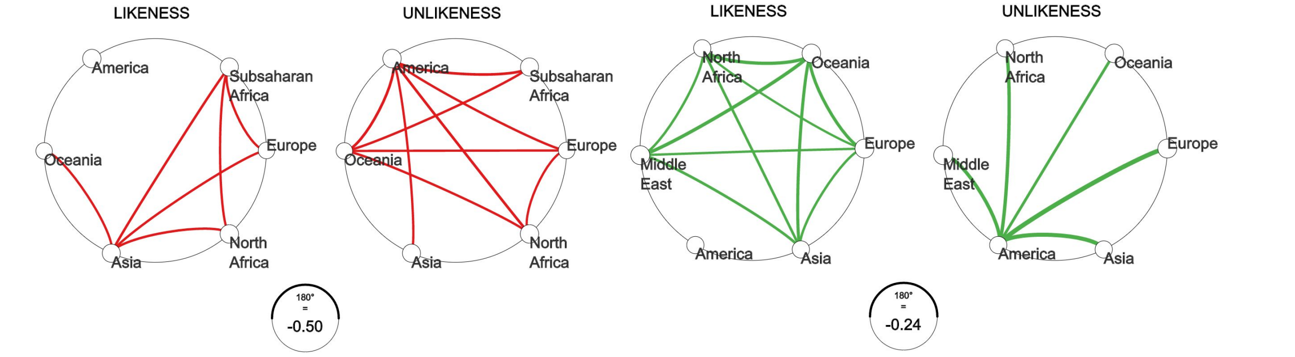
4. Insights

Preliminary analysis of genetic likenesses between different populations on the continent level pointed out several interesting relations, of which some are described here.

A strikingly low relationship was found between the American population and the other population groups for the TAS2R38 gene (Fig 5). This gene has been associated with the ability to perceive 6-N-Propylthiouracil (PROP) and similar substances as bitter. Since some vegetables contain PROP-like substances and an added bitter taste of PROP usually does not encourage a person to eat those vegetables, the ability to taste PROP has been proposed to have a slight negative influence on vegetable consumption [10], but for the same reason also to cause a decreased susceptibility to alcoholism [4,11]. The fact that the American population unit appears to be distinct from other populations may denote either an increased or decreased PROP-tasting ability with respect to other populations. Indeed, previous research by Woodling et al. implied that a haplotype linked to increased PTC-tasting is more prevalent in the American population than in European, African and Asian ones [16]. However, it should be noted that Woodling et al. considered a North-American population, while the sample in this work consists solely of South-American people. Furthermore, the amount of usable SNP data (i.e. IBS0 and IBS2* cases) for this gene in the American population was low. For a more robust inference and any relation to dietary choices, a study focussed on this particular population and gene would be required.

Another interesting pattern emerges for the GPR120 gene. Dysfunction of the GPR120 lipid sensor has been linked to obesity [7]. The graph for this gene shows a likeness between the American, Oceanian and European population units, but likeness with Asian and Subsaharan African units is below average. It can be hypothesized that the variants found in European descendants functions differently and that this may result in a tendency to consume more fatty foods. Although the epidemic nature of obesity in Western society in particular is likely a result of many factors, the possibly population-wide role of GPR120 in this may be further assessed in future research.

Lastly, the TRPV1 gene showed a distinct pattern for Americans as well. Specifically, the likeness was below average compared to all other population units. The Asian population, in contrast, showed above average likeness to all except American populations. Wang et al. [15] noted that Hispanics reacted different from other ethnic groups upon injections with capsaicin, the compound that binds the TRPV1 receptor. Possibly, this effect is reflected in the distinct nature of this group found in the graphs.



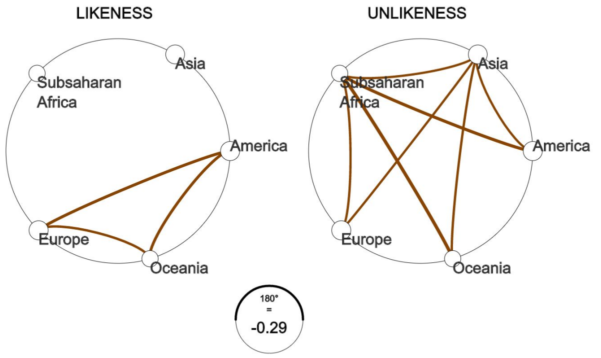


Fig. 5: radial charts for several population subunits on the continent level. The graphs give an easily interpretable visualization of the difference displayed by the American population in the TAS2R38 gene (green) and TRPV1 gene (red), and the separation of Oceanian, European and American populations from other populations for the GPR120 gene (brown).

5. Citations

[1] Bachmanov, Alexander, Natalia Bosak, Cailu Lin, Ichiro Matsumoto, Makoto Ohmoto, Danielle Reed, and Theodore Nelson. "Genetics of Taste Receptors." *CPD Current Pharmaceutical Design* 20.16 (2014): 2669-683. Print.

[2] Beckett, Emma L., Charlotte Martin, Zoe Yates, Martin Veysey, Konsta Duesing, and Mark Lucock. "Bitter Taste Genetics – the Relationship to Tasting, Liking, Consumption and Health." *Food Funct.* 5.12 (2014): 3040-054. Print.

[3] "COLORBREWER 2.0." *ColorBrewer: Color Advice for Maps*. (colorbrewer2.org) Web. 30 May 2016.

[4] Duffy, V., J. Peterson, and L. Bartoshuk. "Associations between Taste Genetics, Oral Sensation and Alcohol Intake." *Physiology & Behavior* 82.2-3 (2004): 435-45. Print.

[5] Duffy, Valerie B. "Associations between Oral Sensation, Dietary Behaviors and Risk of Cardiovascular Disease (CVD)." *Appetite* 43.1 (2004): 5-9. Print.

[6] "Human Genome Diversity Project." *Stanford University*. Web. 31 May 2016.

[7] Ichimura, Atsuhiko, Akira Hirasawa, Odile Poulain-Godefroy, Amélie Bonnefond, Takafumi Hara, Loïc Yengo, Ikuo Kimura, Audrey Leloire, Ning Liu, Keiko Iida, Hélène Choquet, Philippe Besnard, Cécile Lecoeur, Sidonie Vivequin, Kumiko Ayukawa, Masato Takeuchi, Kentaro Ozawa, Maithé Tauber, Claudio Maffeis, Anita Morandi, Raffaella Buzzetti, Paul Elliott, Anneli Pouta, Marjo-Riitta Jarvelin, Antje Körner, Wieland Kiess, Marie Pigeyre, Roberto Caiazzo, Wim Van Hul, Luc Van Gaal, Fritz Horber, Beverley Balkau, Claire Lévy-Marchal, Konstantinos Rouskas, Anastasia Kouvatsi, Johannes Hebebrand, Anke Hinney, Andre Scherag, François Pattou, David Meyre, Taka-Aki Koshimizu, Isabelle Wolowczuk, Gozoh Tsujimoto, and Philippe Froguel. "Dysfunction of Lipid Sensor GPR120 Leads to Obesity in Both Mouse and Human." *Nature* 483.7389 (2012): 350-54. Print.

[8] Ikeda, K. "New Seasonings." *Chemical Senses* 27.9 (2002): 847-49. Print. (reprint of 1909 article)

[9] Lee, W.-C. "Testing the Genetic Relation Between Two Individuals Using a Panel of Frequency-unknown Single Nucleotide Polymorphisms." *Annals of Human Genetics Ann Human Genet* 67.6 (2003): 618-19. Print.

[10] Oftedal, Katherine Nolen, and Beverly J. Tepper. "Influence of the PROP Bitter Taste Phenotype and Eating Attitudes on Energy Intake and Weight Status in Pre-adolescents: A 6-year Follow-up Study." *Physiology & Behavior* 118 (2013): 103-11. Print.

[11] Pelchat, Marcia Levin, and Sarah Danowski. "A Possible Genetic Association between PROP-tasting and Alcoholism." *Physiology & Behavior* 51.6 (1992): 1261-266. Print.

[12] Reed, Danielle Renee, and Antti Knaapila. "Genetics of Taste and Smell." *Progress in Molecular Biology and Translational Science Genes and Obesity* (2010): 213-40. Print.

[13] "UCSC Genome Browser Home." *UCSC Genome Browser Home*. Web. 31 May 2016.

[14] Van der Loo, Mark PJ. "The stringdist package for approximate string matching." *The R* (2014).

[15] Wang, H., A.d.p. Papoiu, R.c. Coghill, T. Patel, N. Wang, and G. Yosipovitch. "Ethnic Differences in Pain, Itch and Thermal Detection in Response to Topical Capsaicin: African Americans Display a Notably Limited Hyperalgesia and Neurogenic Inflammation." *British Journal of Dermatology* 162.5 (2010): 1023-029. Print.

[16] Wooding, Stephen, Un-Kyung Kim, Michael J. Bamshad, Jennifer Larsen, Lynn B. Jorde, and Dennis Drayna. "Natural Selection and Molecular Evolution in PTC, a Bitter-Taste Receptor Gene." *The American Journal of Human Genetics* 74.4 (2004): 637-46. Print.

Appendix: data preparation R-script

```
####################################
# Retrieve relavant gene positions #
####################################

# Load relevant gene names per taste from biomart folder in
# dataframe allGenes
bitterGenes = read.table("biomart\\bitter_genes.txt", header=F)
bitterGenes = cbind(bitterGenes, rep("bitter",dim(bitterGenes)[2]))
names(bitterGenes)[2]="taste"
sweetGenes = read.table("biomart\\sweet_genes.txt", header=F)
sweetGenes = cbind(sweetGenes, rep("sweet",dim(sweetGenes)[2]))
names(sweetGenes)[2]="taste"
sourGenes = read.table("biomart\\sour_genes.txt", header=F)
sourGenes = cbind(sourGenes, rep("sour",dim(sourGenes)[2]))
names(sourGenes)[2]="taste"
umamiGenes = read.table("biomart\\umami_genes.txt", header=F)
umamiGenes = cbind(umamiGenes, rep("umami",dim(umamiGenes)[2]))
names(umamiGenes)[2]="taste"
saltyGenes = read.table("biomart\\salty_genes.txt", header=F)
saltyGenes = cbind(saltyGenes, rep("salty",dim(saltyGenes)[2]))
names(saltyGenes)[2]="taste"
kokumiGenes = read.table("biomart\\kokumi_genes.txt", header=F)
kokumiGenes = cbind(kokumiGenes, rep("kokumi",dim(kokumiGenes)[2]))
names(kokumiGenes)[2]="taste"
alcoholGenes = read.table("biomart\\alcohol_genes.txt", header=F)
alcoholGenes = cbind(alcoholGenes, rep("alcohol",dim(alcoholGenes)[2]))
names(alcoholGenes)[2]="taste"
fatGenes = read.table("biomart\\fat_genes.txt", header=F)
fatGenes = cbind(fatGenes, rep("fat",dim(fatGenes)[2]))
names(fatGenes)[2]="taste"
coffeeGenes = read.table("biomart\\coffee_genes.txt", header=F)
coffeeGenes = cbind(coffeeGenes, rep("coffee",dim(coffeeGenes)[2]))
names(coffeeGenes)[2]="taste"
geneTable = rbind(bitterGenes,sweetGenes, sourGenes, umamiGenes, saltyGenes,
   kokumiGenes, alcoholGenes, fatGenes)
names(geneTable)[1]="gene"

# prepare allGenes dataframe for adding chromosome and start/stop basepair
# info from gene positions file
nGenes = dim(geneTable)[1]
geneTableAdd = data.frame(chr=as.character(rep(NaN,nGenes)),
   start=as.numeric(rep(NaN,nGenes)),end=as.numeric(rep(NaN,nGenes)),stringsAsFactors =
   F)
geneTable = cbind(geneTable, geneTableAdd)

# Load file with all gene positions
genePositions = read.table("gene_positions.tsv",header=T, sep="\t",quote =
   "",stringsAsFactors=F)

# Find chromosome name, gene start and end bp number in gene positions file and
# add to geneTable. If multiple entries exist, use the widest values. If none exist,
   leave NaN.
```

```
for (i in 1:nGenes){
  currentGP = genePositions[genePositions[,7]==geneTable[i,1],]
  if(dim(currentGP)[1]==1)
    geneTable[i,3:5] = currentGP[2:4]
  else if(dim(currentGP)[1]>1 & all(currentGP[,2]==currentGP[1,2])){
    geneTable[i,3:5] = c(currentGP[1,2],min(currentGP[,3]),max(currentGP[,4]))
  }
}

# Two genes are left NA due to multiple genes having the same name in the gene
   positions file.
# They are filled in separately.
geneTable[geneTable[,1]=="PIP",3:5] =
   genePositions[genePositions[,1]=="uc003wcf.1",2:4]
geneTable[geneTable[,1]=="TAS1R3",3:5] =
   genePositions[genePositions[,1]=="uc001aep.1",2:4]

# Convert chromosome names to contain only number/symbol
chrVector = unlist(strsplit(geneTable[,3], "[^0-9]+"))
geneTable[,3] = chrVector[chrVector!=""]


# Generate raw list of SNPs
conmap = file('HGDP_Map.txt')
open(conmap)

# Select SNPs in relevant genes
file.create("relevantSNPs.txt")
write(c("SNP","gene","taste","chr","pos"),ncolumns=5, sep="\t",
   file="relevantSNPs.txt")
for (i in 1:660918){
  currentSNP = scan(file = conmap, what="character",n=3)
  currentGenes =
   geneTable[geneTable[3]==currentSNP[2]&geneTable[4]<=currentSNP[3]&geneTable[5]>=curr
   entSNP[3], ]
  if(length(currentGenes[,1])!=0){
    for(j in 1:length(currentGenes[,1])){
      entry =
   c(currentSNP[1],as.character(currentGenes[j,1]),as.character(currentGenes[j,2]),as.c
   haracter(currentGenes[j,3]), currentSNP[3])
      write(entry,file="relevantSNPs.txt",ncolumns=5,sep="\t",append=TRUE)
    }
  }
}

######################
# SNP data selection #
######################
relSNPs = read.table("relevantSNPs.txt", sep="\t", header =T)


# Extract sample information for relevant SNPs, save in
   HGDP_FinalReport_Forward_Relevant.txt
snpmap = file('HGDP_FinalReport_Forward.txt')
```

```
open(snpmap)
saveFile = "HGDP_FinalReport_Forward_Relevant.txt"
write(as.character(c("SNP",scan(file=snpmap,what="Character", n=1043))),file =
   saveFile, ncolumns=1044,sep="\t")

currentSNP = scan(file=snpmap, what ="Character",n=1,sep="\t")
for (i in 1:660918){
  if (any(currentSNP==relSNPs[,1])){
    write(c(currentSNP, scan(file=snpmap, what ="Character",n=1043)),file = saveFile,
   ncolumns=1044,append=TRUE,sep="\t")
    currentSNP = scan(file=snpmap, what ="Character",n=1,sep="\t")
  }
  else
    currentSNP = scan(file=snpmap, what ="Character", n=1,skip=1)
}

###############################
#Identity by state calculation#
###############################

# Note: code snippet below demonstrates that only AG, AC, TC and TG exist as
   heterozygotes
# this fact will be used in this part of the process.
het.test = c()
snpmap = file('HGDP_FinalReport_Forward.txt')
open(snpmap)
diffTable.names = scan(file=snpmap, what ="Character", n=1043)
for(i in 1:660918){het.test = unique(c(het.test, (unique(scan(file=snpmap, what
   ="Character", n=1044,quiet = T)[-1]))))}
#-------------------------------------------------------------------------------------

library(data.table)
library(stringdist)
gc()
snpmap = file('HGDP_FinalReport_Forward.txt')
open(snpmap)
diffTable.names = scan(file=snpmap, what ="Character", n=1043)
het = c("AG","AC","TC","TG")

# pre-allocate memory
curSNP = vector(mode="character", length = 1043)
diffVector.IBS0 = rep(0,(1043^2-1043)/2)
diffVector.IBS2h = rep(0,(1043^2-1043)/2)
v1 = rep(0,(1043^2-1043)/2)
v2 = rep(0,(1043^2-1043)/2)
v3 = rep(0,(1043^2-1043)/2)
tm = matrix(0,1043,1043)
index.het = c()
index.all = c()

# Define at which moments to save (at ~every 5%)
savepoint = seq(from=0,to=660918,by=floor(660918/20))[-1]
```

```
for (i in 1:660918){
  # Store SNP sequences in vector (without SNP name)
  curSNP = scan(file=snpmap, what ="Character", n=1044,quiet = T)[-1]

  # get indices of heterozygotes and all SNPs (without no-calls)
  index.het = which(curSNP %in% het)
  index.all = which(curSNP != "--")

  # calculate number of IBS2* cases, by taking subset including
  # only homozygotes and finding those cases in which string matches are
  # exact.
  if (length(index.het)>1){
    curSNP.dist = stringdistmatrix(curSNP[index.het])==0
    v1 = rep(index.het, rev(1:length(index.het)-1))
    tm = matrix(index.het,length(index.het),length(index.het))
    v2 = tm[lower.tri(tm)]
    v3 = 1043*v1-(v1^2+v1)/2 -(1043-v2)
    diffVector.IBS2h[v3] = diffVector.IBS2h[v3] + curSNP.dist
  }

  # calculate number of IBS0 cases, by looking at all SNPs (excluding no-calls)
  # and finding those instances that differ in both places (thus excluding IBS1 cases).
  curSNP.dist = stringdistmatrix(curSNP[index.all])==2

  v1 = rep(index.all, rev(1:length(index.all)-1))
  tm = matrix(index.all,length(index.all),length(index.all))
  v2 = tm[lower.tri(tm)]
  v3 = 1043*v1-(v1^2+v1)/2 -(1043-v2)
  diffVector.IBS0[v3] = diffVector.IBS0[v3] + curSNP.dist

  if (i %in% savepoint){
    print(c(as.character(i/660918*100),as.character(Sys.time())))
    write(diffVector.IBS2h,file="diffVector_intermediate_IBS2h.txt")
    write(diffVector.IBS0,file="diffVector_intermediate_IBS0.txt")
    write(c(i, as.character(Sys.time())), file =
   "diffVectors_progressCounter.txt",append=T)
  }
}

close(snpmap)

# Reshape IBS0 and IBS2h vectors into dist objects and save
reshape.IBS2h = dist(rep(1,1043))
reshape.IBS2h[1:543403] = diffVector.IBS2h
reshape.IBS2h.mat = as.matrix(reshape.IBS2h)
colnames(reshape.IBS2h.mat) = diffTable.names; rownames(reshape.IBS2h.mat) =
   diffTable.names
write.table(reshape.IBS2h.mat,file="diffTable_IBS2h.txt")

reshape.IBS0 = dist(rep(1,1043))
reshape.IBS0[1:543403] = diffVector.IBS0
reshape.IBS0.mat = as.matrix(reshape.IBS0)
colnames(reshape.IBS0.mat) = diffTable.names; rownames(reshape.IBS0.mat) =
   diffTable.names
```

```
write.table(reshape.IBS0.mat,file="diffTable_IBS0.txt")

# Calculate conditional concordance probability as described by Lee et al. (2003)
# and save
reshape.ratio = reshape.IBS2h/(reshape.IBS0 + reshape.IBS2h)
reshape.ratio.mat = as.matrix(reshape.ratio)
colnames(reshape.ratio.mat) = diffTable.names; rownames(reshape.ratio.mat) =
   diffTable.names
write.table(reshape.ratio.mat,file="diffTable_ratio.txt")

#####################################################
#calculate average IBS statistic per population unit#
#####################################################

# Load IBS0 and IBS2h info for all SNPs
overallIBS0 = read.table("diffTable_IBS0.txt",header=T,row.names=1)
overallIBS2h = read.table("diffTable_IBS2h.txt",header=T,row.names=1)

# Load IBS0 and IBS2h info for taste genes (have been stored differently, 1 gene per
   column)
tasteIBS0 = read.table("diffTable_IBS0_tasteGenes.txt",header=T)
tasteIBS2h = read.table("diffTable_IBS2h_tasteGenes.txt",header=T)


# Read population info
pop = read.table("HGDP-CEPH-ID_populations_tab.txt", header = T, row.names=1, sep =
   "\t")

# Some subjects in the populations information tab are not in the SNP data. They may
   cause
# problems later on, so the are removed.
pop = pop[which(row.names(pop) %in% row.names(overallIBS0)),]

continentNames = c(as.character(unique(pop$Region)))
countryNames = c(as.character(unique(pop$Geographic.origin)))
allNames = c(continentNames, countryNames)

# Create p5all data frame. On the diagonal, IBS within group is stored.
# In both upper and lower triangle, the IBS statistic between groups is stored.
p5all = matrix(data = NA, nrow=length(allNames),ncol = length(allNames))
p5all = data.frame(p5all, row.names = allNames)
colnames(p5all) = allNames

# Calculate overall average IBS compared per continent
for (i in 1: length(continentNames)){
  # get subjects from i-th continent
  sni = rownames(pop[pop$Region==continentNames[i],])
  for(j in i:length(continentNames)){
    # get subjects from j-th continent, where j>=i
    snj = rownames(pop[pop$Region==continentNames[j],])
    # calculate IBS statistic between i and j
    IBSstat = as.matrix(overallIBS2h[sni,snj]/(overallIBS0[sni,snj] +
   overallIBS2h[sni,snj]))
    # Take mean of all elements. Those on the main diagonal of overallIBS0 and
   overallIBS2h
    # conveniently turn into NAs (as they are filled with 0s, result of the way they
   were
    # constructed). These are omitted. No other 0 values are in the dataset (this has
   been
    # checked).
    p5all[continentNames[i],continentNames[j]] = mean(IBSstat,na.rm=T)
    p5all[continentNames[j],continentNames[i]] = mean(IBSstat,na.rm=T)
  }
}

# Calculate overall average IBS compared per country
for (i in 1: length(countryNames)){
  sni = rownames(pop[pop$Geographic.origin==countryNames[i],])
  for(j in i:length(countryNames)){
    snj = rownames(pop[pop$Geographic.origin==countryNames[j],])
    IBSstat = as.matrix(overallIBS2h[sni,snj]/(overallIBS0[sni,snj] +
   overallIBS2h[sni,snj]))
    p5all[countryNames[i],countryNames[j]] = mean(IBSstat,na.rm=T)
    p5all[countryNames[j],countryNames[i]] = mean(IBSstat,na.rm=T)
  }
}

# save files in the directory from which P5 pulls the data
pathname = "D:/Google Drive/Semester2/MLO Management of Large-scale Omics data/MLO
   project/P5_vizualization/taste_visualization/"
filename.p5all = paste(pathname,"p5all.csv", sep="")
write.csv(p5all, filename.p5all, row.names=F)
filename.allNames = paste(pathname,"p5popNamesIndex.csv", sep="")
write.csv(allNames,filename.allNames)


# Per taste gene calculation

geneNames = colnames(tasteIBS0)


# Calculate IBS statistic per gene
for (k in 1:length(geneNames)){
  p5taste = matrix(data = NA, nrow=length(allNames),ncol = length(allNames))
  p5taste = data.frame(p5taste, row.names = allNames)
  colnames(p5taste) = allNames


  # create templates to reshape the vectorized data into a dist object (safest way to
  # recreate the original data format)
  reshape.IBS2h = dist(rep(1,1043))
  reshape.IBS0 = dist(rep(1,1043))
  # reshape info for one gene into a dist matrix
  reshape.IBS2h[1:543403] = tasteIBS2h[,k]
  reshape.IBS2h = data.frame(as.matrix(reshape.IBS2h),row.names=rownames(overallIBS0))
  colnames(reshape.IBS2h) = rownames(overallIBS0)

  reshape.IBS0[1:543403] = tasteIBS0[,k]
  reshape.IBS0 = data.frame(as.matrix(reshape.IBS0),row.names=rownames(overallIBS0))
  colnames(reshape.IBS0) = rownames(overallIBS0)

  # Calculate overall average IBS compared per continent
  for(i in 1: length(continentNames)){
  sni  = rownames(pop[pop$Region==continentNames[i],])
  for(j in i:length(continentNames)){
      snj = rownames(pop[pop$Region==continentNames[j],])
      IBSstat = as.matrix(reshape.IBS2h[sni,snj]/(reshape.IBS0[sni,snj] +
   reshape.IBS2h[sni,snj]))
      p5taste[continentNames[i],continentNames[j]] = mean(IBSstat, na.rm=T)
      p5taste[continentNames[j],continentNames[i]] = mean(IBSstat, na.rm=T)
    }
  }


  # Calculate overall average IBS compared per country
  for (i in 1: length(countryNames)){
    sni = rownames(pop[pop$Geographic.origin==countryNames[i],])
    for(j in i:length(countryNames)){
      snj = rownames(pop[pop$Geographic.origin==countryNames[j],])
      IBSstat = as.matrix(reshape.IBS2h[sni,snj]/(reshape.IBS0[sni,snj] +
   reshape.IBS2h[sni,snj]))
      p5taste[countryNames[i],countryNames[j]] = mean(IBSstat, na.rm=T)
      p5taste[countryNames[j],countryNames[i]] = mean(IBSstat, na.rm=T)

      savename = paste(pathname,"p5gene_",geneNames[k],".csv",sep="")
      write.csv(p5taste,savename,row.names=F)
      }
  }
}
```