Spring 5-9-2022

Heterogeneity of Gene Expression in Peanut-Allergic Cells May be Related to Differing Cell Subpopulations

Sophia R. Tekorius
Seattle Pacific University

HETEROGENEITY OF GENE EXPRESSION IN PEANUT-ALLERGIC CELLS MAY BE RELATED TO DIFFERING CELL SUBPOPULATIONS

by

SOPHIA TEKORIUS

FACULTY MENTOR:
DR. JENNY TENLEN

HONORS PROGRAM DIRECTOR:
DR. CHRISTINE CHANEY

A project submitted in partial fulfillment of the requirements
or the Bachelor of Arts degree in Honors Liberal Arts
Seattle Pacific University
2022

Presented at the SPU Honors Research Symposium
May 21st, 2022

Abstract

Peanut allergies can have severe and life-threatening complications. By understanding the biology of individuals struggling with this disease, better treatments can be developed. This project explores the possibility of two phenotypically distinct cell types within CD154+ cell types in patients with peanut allergy. Previous studies show that there is a T_h2 cell subpopulation associated with allergies, and within effector CD4+ memory populations two distinct phenotypes have been found. This project expands upon this research at the genetic level and through bioinformatic analyses, including the production of heatmaps, the project has found evidence for differing gene expression in two cell subpopulations. One group, CRTH2+ cell type, exhibits expression patterns similar to the T_h2 cell subpopulation while the other, CCR6+ cell type, expresses genes that are known to have a Th17-like or regulatory function. This is significant because very few of the genes associated with the CCR6+ cell type have been well studied, indicating a potential gap in research that may broaden our understanding of this disease.

Introduction

Peanut allergy affects approximately 1% of children under the age of 5, and that number has increased significantly in the last 15 years.[1-5] Peanuts are one of the four most common food allergens in both children and adults[1, 3] and cause the largest number of cases of severe anaphylaxis and death in the U.S.[2] The prevalence and severity of potential consequences of peanut allergy make this disease an important area of study.

Peanut allergic response can be highly variable, ranging from mild hives, to severe allergic response or death.[6] This wide range in response has led to this project exploring whether there may be patterns in gene expression that reflect allergic phenotypes, represented by RNASeq data, and a wide range of variables including peanut IgE, age, and more. In the search for a correlation of any kind, this project found evidence of a potential subset of two cell groups within the larger group of CD154+ peanut reactive cells.

T helper cell 2 (T_h2) and T helper cell 17 (Th17) play strong roles in the immune system, Th2 cytokines have been found in association with strong antibody and allergic responses while Th17 cells can cause inflammation and autoimmunity in the body[7,8]. Previous studies have shown that there is a cell subpopulation within the T_h2 cell population that is associated with allergies.[9] This cell subpopulation has been studied by the Wambre lab at Benaroya Research Institute (BRI) and two distinct phenotypes in the effector CD4+ memory populations have been found: the first being a better-studied allergic TH2A phenotype (CD27- CRTH2+ CCR4+ CCR6-), and the second being a Th17-like phenotype (CD27- CRTH2- CCR4+ CCR6+).[10] In the study put forth by this paper, a heatmap of gene expression for both overall peanut allergic cells, and a heatmap comparing two cell types, CCR6+ and CRTH2+, were produced.

Materials and Methods

Acquisition of Samples

Patients from three different clinical trials, P354, P350, P392, and P193, had blood samples taken. For projects P354, P350, and P193, patients underwent a double-blind, placebo-controlled food challenge (DBPCFC) to ensure that all were allergic to peanuts. For P392, a skin-prick test was used to check for peanut allergy. P43 were peanut allergic samples taken outside of a clinical trial; peanut allergy was confirmed with a DBPCFC. All cells were selected by the Wambre lab for being CD154+ in the presence of peanut protein in solution, ensuring that the cells selected were peanut-responsive.

RNASeq Data

After cells were selected, RNASeq data was collected. The data exhibited how often genes were transcribed into RNA, giving a general idea of the gene expression of each gene. This RNASeq data was in a file labeled "counts" as gene expression is given in the number of times that gene is counted when the RNA is being sequenced. These data were entered into a table in R Studio along with other information relevant to the patients the samples came from.

Organization of Data

All data in this project was assessed in RStudio[9], unless otherwise specified. To build the first heatmap, each data file was labeled with the project number, P193, P350, P392, or P354 and whether it was a "counts" or "design" data frame, indicating whether the data was the gene "counts" RNA Sequencing data, or was a "design" put together by Dr. Hannah DeBerg at BRI containing information on the sample such as patient's age, IgE levels, and other data. The data provided by P392 was quickly excluded from the first heatmap due to the project not using the gold-standard DBPCFC to confirm allergy.

The function "idkey" was then used to recode different variables for consistency across projects, ie. Peanut IgE and ige becoming peanut_IgE for all projects. From there, basic organization took place including setting colors and directories for graphing. Here, the mean and standard deviation of age and IgE levels for each project were calculated, as well as the number and percentage of baseline samples and gender.

Rename Variables for Consistency Across Projects

Each "counts" column was labeled with the associated library ID for future organization, which then allowed all "counts" data to be compiled into one folder vertically instead of

horizontally. Each of the "design" data frames were combined into one large folder. From there T1, V0, and V1 values in the Baseline column were renamed to “Baseline” as each value meant that they were the initial sample taken prior to the beginning of the clinical trials. Any results after the trials had begun are not relevant to the project. Also values called “Peanut peptide ON", "Arah1", "Arah2", "Arah3", “peanut” in the stimulation column were recoded to “peanut extract” indicating that all of these samples were stimulated with peanut extract as some had been stimulated with alder extract.

Quality Control

Samples without “baseline”, “peanut extract”, and cell type as “CD154+” were filtered out, as this project focused on untreated patients with peanut allergies that resulted in CD154+ cells, indicating an immune response. Quality control of the RNASeq data then took place, with an alignment cutoff for samples greater than 65%, a total reads cutoff of samples with greater than 500,000 reads, and a median CV cutoff of less than 1.1. This created a column called “qc_pass” which showed whether the data passed the cutoff in TRUE/FALSE format. Graphs plotted the samples based on total reads, percent aligned, and median CV coverage giving a visual representation of the quality control cutoff. The “qc_pass” column was then used to cutout samples with less quality, where qc_pass = TRUE in dplyr filter indicated that the sample was kept.[12]

Genes were then filtered down to only protein-coding genes with one duplicate removed by matching each library ID (libid) with the HGNC code from a preset file of HGNC protein names. The overall "counts" file was then reorganized so the first three columns were annotation information that organized each sample and all other columns were counts for each protein-coding gene. Counts were then filtered to only those that passed quality control and a function was built to filter out lowly expressed genes that excluded samples where counts per million was below the counts cutoff of one in at least 10% of the counts.

Examine Correlations

The function “tmm” built by Dr. Hannah DeBerg can be found in the code (S1, S2) and was used to normalize the "counts" data. This was used to prepare the data to remove batch effects via the function RemoveBatchEffect.[13] A principal component analysis (PCA) was run on the data, producing a data frame that showed several potential correlations.[15] PC1 and PC2 were plotted together shading the data points with a variety of variables, looking for patterns. The same was done with PC3 vs PC4. From there, the top genes that varied when peanut IgE, IgG4, the skin prick test baseline, maximum tolerated dose, CD154 frequency, CCR6 percentage, CD27 percentage, and CRTH2 levels also varied were collected using

VoomWithQualityWeights[13] and organized from highest to lowest variation. Each of these data frames were saved as Excel files.

To build the heatmap, all the "counts" and "design" files were combined into a data frame of even width and length. The variable "genes_for_heatmap" was used to select the genes examined in the heatmap with many of the top genes from the Excel files, as well as several genes that were hand-picked from PCA data as well as previous studies. HeatmapAnnotation[15] was then used to build several heatmaps with different genes and different variables shaded along the top.

Examining CCR6+ and CRTH2+ Potential Cell Groups

The second heatmap of this project examined potential cell subtypes and operated very similarly to the process to build the first. Key differences included adding data from P43 and P392 and not including data from P350. No batch effects were removed either and the genes used were specifically selected. A correlation map of genes that were differentially expressed based on cell type was also produced by STRING.[16]

Results

Examining Heterogeneity in Peanut-Allergic Cells

The mean age of patients for each trial changed considerably between each project, with P354 having a mean age of 3.22 +/- 0.0601 years and P350 having a mean age of 24.3 +/- 4.38 years of age (Table 1). This is important to note as it is known that allergies, including peanut, tend to change throughout a person's lifetime.[17] While P354 focused on pediatric allergies, the samples from each project were selected based on volunteer status and a DBPCFC to confirm peanut allergy. While P193 tends towards more female and P350 includes more male samples, no trend in sex was seen in later data analysis, though it was considered.

	AR101 (P193)	Astellas (P350)	IMPACT (P354)
Total Samples	125	328	184
Baseline Samples	59	79	79
Pass QC	59.4%	68.4%	100%
Sex	54.4% F / 45.6% M	40.5% F / 59.5% M	-
Age (mean +/- sd)	15.9 +/- 7.77	24.3 +/- 4.38	3.22 +/- 0.601

Peanut IgE (mean +/- sd)	127.8 +/- 203.6	27.6 +/- 29.19	208.6 +/- 347.7

Table 1: Overview of the makeup of each project used in the first part of this study. Sex is expressed in percentage of female / male. Age and Peanut IgE are expressed in mean +/- one standard deviation.

Quality control of the RNA Seq data took place with an alignment cutoff of samples greater than 65%, a total reads cutoff of samples with greater than 500,000 reads, and a median CV cutoff of less than 1.1. The majority of RNA sequencing samples passed quality control (Fig. 1, Table 1). This ensured the data used in the following analyses was of quality.

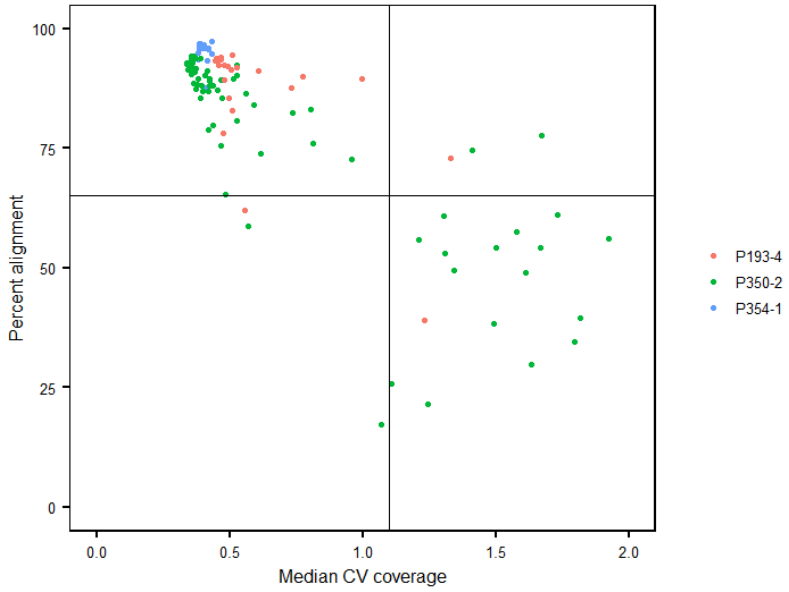

Figure 1: Visual representation of data points versus the quality control metrics. Samples with a median CV coverage less than 1.1 and greater than 65% coverage passed the quality control and are in the top left quadrant.

Batch effects were exposed by the PCA plot of PC1 vs PC2 (Fig. 2A). Each project tended to group together, however when batch effects were removed, some grouping remained consistent (Fig. 2B).

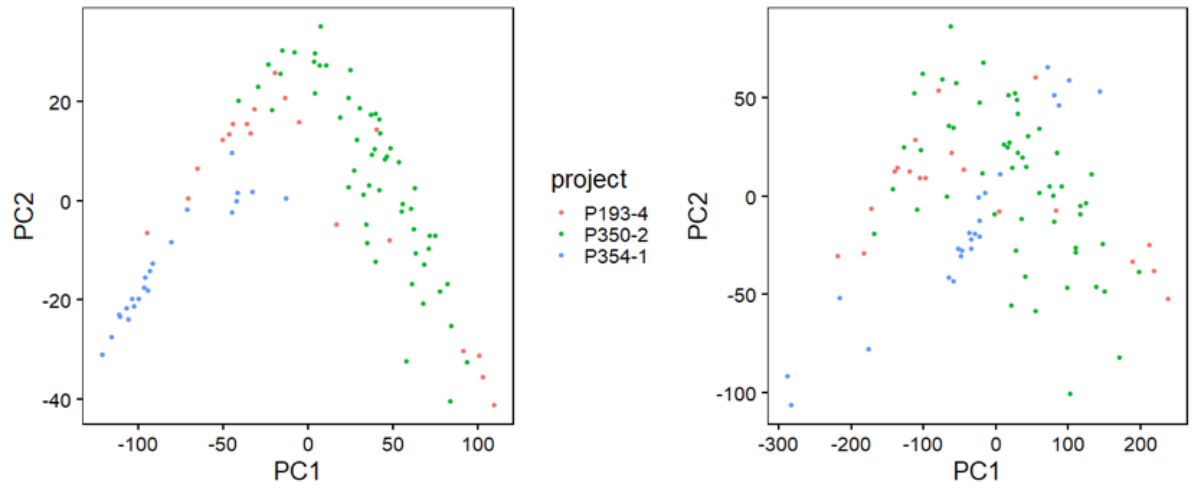

Figure 2: Figure 2A (left) shows a graph of PC1 versus PC2 provided by the PCA function prior to batch effects being moved. Figure 2B (right) shows the PCA plot after RemoveBatchEffect had been run on the data, indicating less of a batch effect seen when data is colored by project.

After examining potential correlations in the data, a heatmap was put together using a list of genes known to be involved in allergic response. This list was provided by Dr. Justine Calise from BRI. In this heatmap, a visual examination shows two groups of gene expression that appear to be separate from one another (Fig.3). Many of the genes in the bottom half of the list, such as RORC, IL-17A, and IL-17F are known Th17 markers.[9] Samples were colored by age and IgE levels as the PCA plot showed that when batch effects were removed, some grouping between projects was eliminated but not all, indicating that another factor may have been at play, because mean age differed drastically between projects, and studies have shown that allergies may change over time, the age variable was shown.[17] IgE was chosen due to the fact that when added to the heatmap it showed an almost inverse trend compared to age.

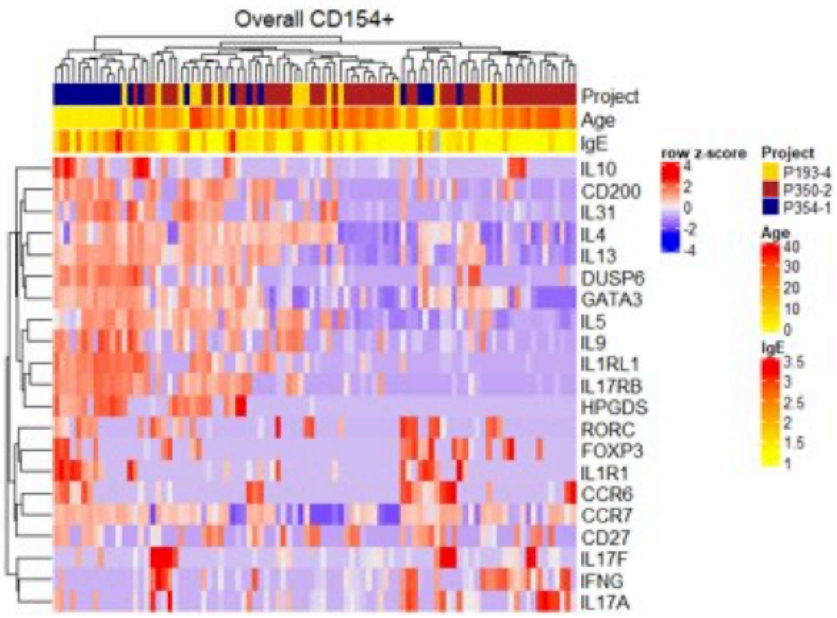

Figure 3: The heatmap from the overall CD154+ cell data set. Two general groups of gene expression can be seen in the top left corner and bottom right corners. Along the top, general patterns can be seen in age and peanut IgE which are colored from lowest in yellow, to highest in red.

To further explore the grouping seen in the heatmap, the percentage of CRTH2 and CCR6 proteins expressed on the surface of the cells from each sample were plotted against each other and colored by age, as age was shown to have a trend across the x-axis in the heatmap as well as the plot. When the plot was colored by IgE, no such trend occurred. Due to the fact that CRTH2 and CCR6 were part of what defined the two distinct phenotypes in the effector CD4+ memory populations found by the Wambre lab, ie. the TH2A phenotype (CD27- CRTH2+ CCR4+ CCR6-), and the Th17-like phenotype (CD27- CRTH2- CCR4+ CCR6+), and many of the genes in the bottom half of the heatmap were known Th17 markers, CRTH2 and CCR6 were selected for plotting and further study.[9,10] Here, the graph shows that individuals tended to have cells high in either CRTH2 expression or CCR6 expression, with those with more CCR6 tending to be younger (Fig. 4). This prompted the second heatmap of the project.

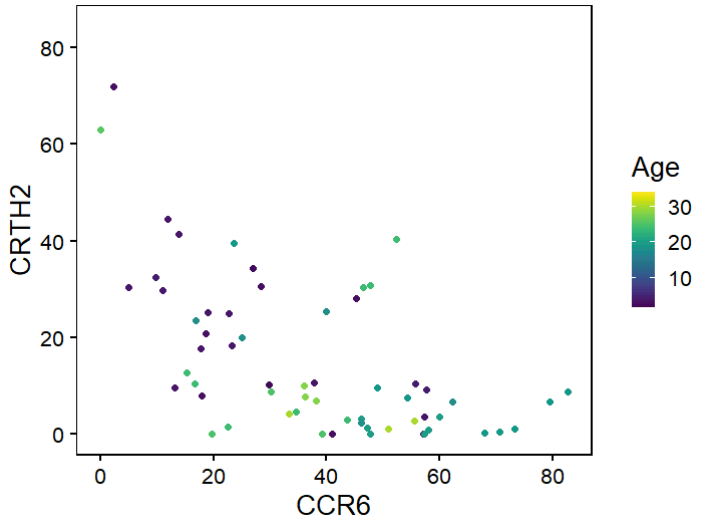

Figure 4: The quantity, in percentage, of CRTH2 and CCR6 expressed on the cell surfaces of CD154+ cells are plotted against each other and colored by age. This graph shows that older ages roughly tend to have less CRTH2 and more CCR6 while younger ages tend to have more CRTH2 and less CCR6 on the allergic cell surfaces.

Examining Differences between CCR6+ and CRTH2+ Cell Groups

The data from three clinical trials and one collection of peanut allergic samples done at BRI were used to explore the potential cell subgroups seen in the previous heatmap. While P392 had not been used in the earlier heatmap due to the lack of DBPCFC, the focus of this heatmap was more on sample size and less on collection quality. P350 was not used as none of the cell types included being CRTH2+ or CCR6+, only CD154+. The breakdown of patient and sample data can be seen in Table 2.

	AR101	IMPACT (P354)	Aravax (P392)	BRI Samples (P43)
Total Samples	125	184	307	25
Pre-Treatment Samples	59	79	86	25 (no treatment)
Pass QC	58.3%	100%	60%	94.4%
Age (mean +/- sd)	15.9 +/- 7.77	3.22 +/- 0.601	-	26.09 +/- 15.1
CCR6+	5	6	10	13
CRTH2+ Samples	7	12	0	5

Table 2: Overview of the makeup of the patients and the samples for each project used in the second part of this study. The BRI samples were not part of a clinical study and therefore received no treatment.

After quality control had taken place, a heatmap was built with colors for project and cell type along the top, and the list of the top 50 differentially expressed genes pulled from the PCA data. While there are areas of overlap, the two cell subgroups are clearly distinguished in gene expression (Fig 5). Highlighted genes indicate that they were on the list of genes known to play roles in allergic responses. Most of the known allergic related genes have high expression in the CRTH2+ subgroup. This is consistent with our prior knowledge of CRTH2+ cells being a subgroup within CD154+ allergic cells, and therefore having genes that are better studied. FOXP3 and IL1R1 are known to have functions that are either regulatory in nature or related to a Th17-like phenotype.[10]

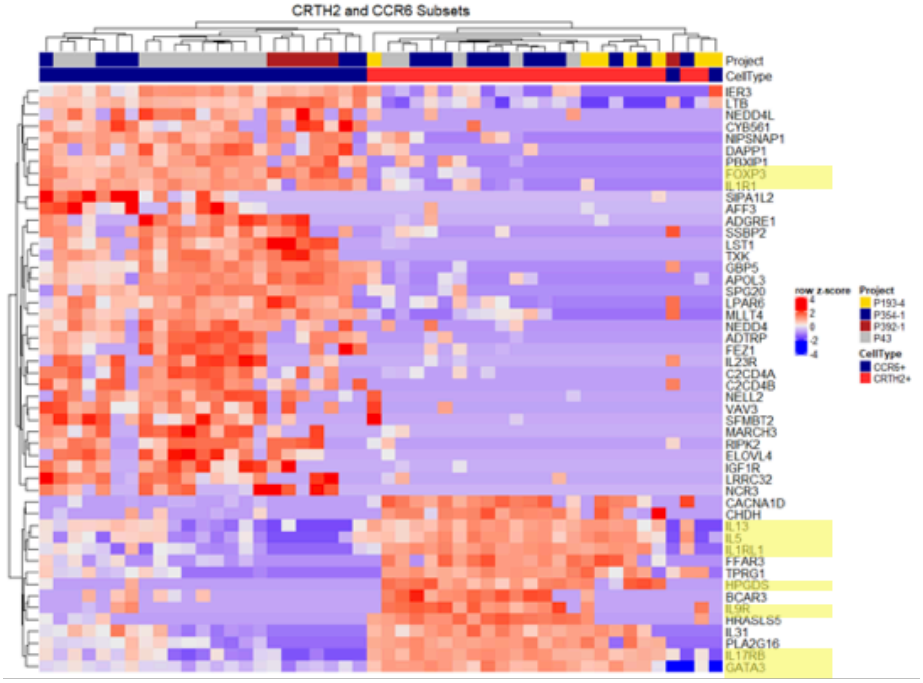

Figure 5: The heatmap exploring CRTH2+ and CCR6+ cell subsets. In this heatmap, the two cell types clearly separate from one another. Genes highlighted in yellow indicate genes that are known to play roles in allergic and immune responses.

The same 50 genes used in the heatmap were then submitted to STRING[18], a web-based database and program that draws networks indicating documented correlations between the two genes. In the produced figure, one main grouping of interactions can be seen (Fig. 6).

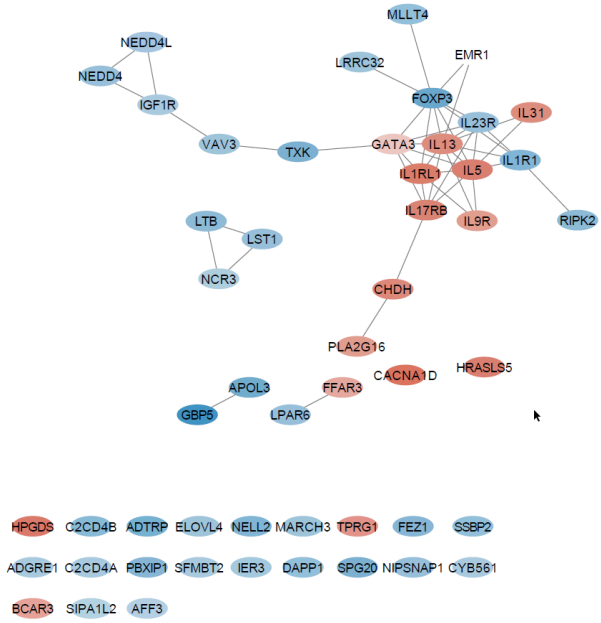

Figure 6: Network of the protein to protein interactions between the top 50 genes that differentiated by cell type. Here red or pink indicates genes associated with the CRTH2+ cell type, and blue indicates genes associated with the CCR6+ cell type.

Discussion

Both heatmaps support the concept that within CD4+ memory populations two subgroups exist: one with a CRTH2+ phenotype and TH2A function, and the other with a CCR6+ phenotype and regulatory function perhaps similar to Th17 cells. Many of the genes in the bottom half of the list in Figure 3, such as RORC, IL-17A, and IL-17F are known Th17 markers, suggesting that one of these cell types may have a Th17-like phenotype. Most known allergic related genes have high expression in the CRTH2+ subgroup in Figure 5. This is consistent with our prior knowledge of CRTH2+ cells being a subgroup within CD154+ allergic cells, and therefore having genes that are better studied. FOXP3 and IL1R1 are known to have functions that are either regulatory in nature or related to a Th17-like phenotype.[10] In the STRING plot, TH2A and regulatory responses are reflected in the interactions. IL23R is a gene known to be active in Th17+ cells which may indicate a correlation between Th17+ and CCR6+ cell types (Fig. 6).

Heterogeneity of common proteins associated with allergic disease has been found in CD154+ T cells from individuals with peanut allergies, which has made it difficult to identify an accurate biological measure of allergic disease.[9] That heterogeneity may be the result of differing cell populations, as studies have seen seemingly “paradoxical” results with a cell type high in one patient and low in the other while both exhibit intense allergic response.[20] By identifying and defining differing cell populations, a biological measure of atopic disorders that does not risk individual health, such as a DBPCFC, could be developed. TH2A has been

identified as a cell subgroup by the Wambre lab which may counterbalance a CCR6+ cell type in peanut allergic cells.[9] This project has added evidence to previous studies that there may be a regulatory / TH17 signature in CCR6+ peanut-reactive cells, and has added knowledge to previous studies that focused solely on CRTH2+ cells in other allergies such as asthma.[10,18]

Next steps should include a longitudinal study looking at CRTH2+ and CCR6+ populations over time in order to further examine a correlation between age and these two cell types as well as further study of CC6+ associated genes. Further study into how these two subgroups of peanut-allergic cells interact with both each other and the rest of the body could provide insight into how peanut allergies change over time, due to the rough correlation between age and these cell types. The existence of two genetically and phenotypically different cell types may indicate that food allergy is not consistent between individuals and may be much more nuanced. In addition, because the genes and proteins associated with the CCR6+ cell type are significantly less studied, exploration of these structures and functions could fill in gaps in our knowledge of peanut allergies. By increasing understanding we can hope to come closer to better treatments for those that struggle with this disease and better biological measurements of disease progression.

Acknowledgements

I would like to acknowledge the efforts of so many people who have made this project possible. First and foremost, Hannah DeBerg from Benaroya Research Institute, from the Bioinformatics and Software Engineering department, as well as, Charlie Quinn, Peter Linsley, Scott Presnell, Matt Dufort, Mario Rosasco, Alex Hu, Alex Walker, Adarsh Manjunath, Anna Bjork, and Robyn Meshulam. Special thanks to the Wambre Laboratory, including Justine Calise, Takeshi Yamamoto, Sugandikha Khosa, Maochang Liu, Anne Chaize, Blake Rust, Nahir Garabatos, and Kelly Aldridge. Vivian Gersuk, Quynh-Anh Nguyen, and Kimberly O'Brien from Systems Immunology were also key to the project and from Clinical Core: Gina Marchesini , Claire Mangan, and Thien-Son Nguyen. Mary Farrington and David Jeong from Virginia Mason made this project possible as well as the internship coordinators: Jessica Hamerman and Ruth Penn. For contributing samples acknowledgements go to, Aimmune therapeutics, Astellas pharma, Aravax, and ITN IMPACT team. Finally, from Seattle Pacific University I would like to thank Drs. Christine Chaney and Jenny Tenlen for guiding this project to completion, not to mention the many peers that reviewed this paper and provided brainstorming sessions.

References

1. Sampson HA. Update on food allergy. J Allergy Clin Immunol 2004; 113: 805–19.
2. Sicherer SH, Sampson HA. 9. Food allergy. J Allergy Clin Immunol 2006; 117 (Suppl 2): S470–S475.
3. Lee LA, Burks AW. Food allergies: prevalence, molecular characterization, and treatment/prevention strategies. Ann Rev Nutr 2006; 26: 539–65.
4. Sampson HA. Clinical practice. Peanut allergy. N Engl J Med 2002; 346: 1294–99.
5. Grundy J, Matthews S, Bateman B, Dean T, Arshad SH. Rising prevalence of allergy to peanut in children: data from 2 sequential cohorts. J Allergy Clin Immunol 2002; 110: 784–89.
6. Allen KJ, Remington BC, Baumert JL, Crevel RW, Houben GF, Brooke-Taylor S, Kruizinga AG, Taylor SL. Allergen reference doses for precautionary labeling (VITAL 2.0): clinical implications. Journal of allergy and clinical immunology. 2014 Jan 1;133(1):156-64.
7. Bettelli E, Korn T, Kuchroo VK. Th17: the third member of the effector T cell trilogy. Current opinion in immunology. 2007 Dec 1;19(6):652-7.
8. Mosmann TR, Sad S. The expanding universe of T-cell subsets: Th1, Th2 and more. Immunology today. 1996 Mar 1;17(3):138-46.
9. Wambre E, Bajzik V, DeLong JH, O'Brien K, Nguyen QA, Speake C, Gersuk VH, DeBerg HA, Whalen E, Ni C, Farrington M. A phenotypically and functionally distinct human TH2 cell subpopulation is associated with allergic disorders. Science translational medicine. 2017 Aug 2;9(401):eaam9171.
10. Bajzik V, Deberg H, Calise J, Farington M, Wambre E. Peanut-reactive T cell profiling reveals diversity in allergic patients and predicts response to therapy. World Allergy Organization Journal. 2020 Aug 1;13(8).
11. RStudio Team (2020). RStudio: Integrated Development for R. RStudio, PBC, Boston, MA URL http://www.rstudio.com/.
12. Wickham H, François R, Henry L, Müller K (2022). dplyr: A Grammar of Data Manipulation. https://dplyr.tidyverse.org, https://github.com/tidyverse/dplyr.
13. Ritchie ME, Phipson B, Wu D, Hu Y, Law CW, Shi W, Smyth GK (2015). "limma powers differential expression analyses for RNA-sequencing and microarray studies." Nucleic Acids Research, 43(7), e47. doi: 10.1093/nar/gkv007.
14. Venables, W. N. and B. D. Ripley (2002) Modern Applied Statistics with S, Springer-Verlag.
15. Gu Z, Eils R, Schlesner M (2016). "Complex heatmaps reveal patterns and correlations in multidimensional genomic data." Bioinformatics.

16. Szklarczyk D*, Gable AL*, Nastou KC, Lyon D, Kirsch R, Pyysalo S, Doncheva NT, Legeay M, Fang T, Bork P‡, Jensen LJ‡, von Mering C‡.The STRING database in 2021: customizable protein–protein networks, and functional characterization of user-uploaded gene/measurement sets. Nucleic Acids Res. 2021 Jan 8;49(D1):D605-12.
17. Venter C, Hasan Arshad S, Grundy J, Pereira B, Bernie Clayton C, Voigt K, Higgins B, Dean T. Time trends in the prevalence of peanut allergy: three cohorts of children from the same geographical location in the UK. Allergy. 2010 Jan;65(1):103-8.
18. Szklarczyk et al. Nucleic Acids Res. 2015 43(Database issue):D447-52, http://string-db.org
19. Palikhe NS, Laratta C, Nahirney D, Vethanayagam D, Bhutani M, Vliagoftis H, Cameron L. Elevated levels of circulating CD 4+ CRT h2+ T cells characterize severe asthma. Clinical & Experimental Allergy. 2016 Jun;46(6):825-36.
20. Wambre ER, Farrington M, Bajzik V, DeBerg HA, Ruddy M, DeVeaux M, Meier P, Robinson D, Cantor M, Huang C, Orengo JM. Clinical and immunological evaluation of cat-allergic asthmatics living with or without a cat. Clinical & Experimental Allergy. 2021 Dec;51(12):1624-33.

Appendix

Research as Service: Cultivating Care for Neighbor and Creation

This past summer I participated in an internship at Benaroya Research Institute under the guidance of Dr. Hannah DeBerg in the bioinformatics department. In this internship, I looked for connections between gene expression and peanut allergies. The research I did has since turned into my honor's project which provides a context for ideas surrounding research.

Now the title of this panel is Research as Service: Cultivating Care for Neighbor and Creation, and this title comes in response to the question asked every year I've been in the honors program: what does it mean to be human? As my parents can attest, when first asked for a connection between my research project and the meaning of being human, all I could come up with was "sometimes being human means being allergic to peanuts". And in part, that is true, to be human is to have things go wrong, but also to be human is to want to solve these issues. The meaning of humanity is not directly related to *what* the research is but is instead related to *why* we conduct the research.

We conduct research oftentimes to help people, food allergies affect about 32 million people in the United States, with peanuts being one of the four most common food allergens in both children and adults. Peanut allergies cause the largest number of cases of severe anaphylaxis and death in the U.S. However, the differences in response to peanuts can be drastic between individuals, with one person only experiencing mild hives and another going into severe anaphylactic shock. This heterogeneity led to this project, which is where I must introduce a little background for those of you in the audience with less experience in cell biology.

Many of you know that DNA helps guide our bodies but it does so through a process called transcription and translation. A gene from the DNA is copied inside the nucleus, this copy is called mRNA or messenger RNA. This copy then leaves the safety of the nucleus and is used in translation using tRNA, which converts the information in mRNA into a protein sequence. One way to think of this is that the DNA is the master blueprint, mRNA is a copy of the blueprint, and the tRNA starts placing pieces of wood together. Proteins are important because they do the majority of things in our bodies. This project used RNA Seq data which sequences the mRNA transcribed from the DNA. This gives us a general idea of what genes are being transcribed, which gives us an indication that the associated proteins are being produced as well. This is not a perfect correlation between gene and protein expression but they are close enough for this project.

The data for this project was collected in cell samples from patients in four different clinical trials, and one collection of samples stored at BRI. For most patients, a double-blind, placebo-controlled oral food challenge was used to confirm that they were allergic to patients, which is basically a fancy way of saying a doctor gave them some peanuts or peanut protein, and if they had an allergic reaction they were confirmed to have peanut allergies. One project used a skin prick test to verify peanut allergies which, while less dangerous, can have false positives. Therefore, the data was excluded during the first part of my project. Samples were then stimulated overnight with peanut proteins and T-cells that recognized these proteins upregulated a surface marker, CD154. By sorting CD154 positive T-cells we were able to select for peanut-

reactive cells only. In sequencing, RNA is chopped up into chunks, these chunks are then aligned to the human genome and each chunk is counted as a "read". The more reads, the more the gene / protein was expressed. This RNA Sequencing data was given to me at the start of my project by Hannah DeBerg, courtesy of the Genomics Core.

All of the data manipulation I did took place in R Studio, I began by performing quality control on the 637 samples I began with, removing any samples that were not at baseline, since any point after that could be changed due to the study being done. Some samples were not stimulated overnight, and some did not express CD154+, indicating that they were not peanut-reactive cells, these were removed from the study. Finally I did some quality control on the reads of the RNA Sequencing themselves and was left with 96 samples. This sounds fairly small after starting with 600+ but is actually a large sample size for a study like this.

From there, I performed a Principle Component Analysis to look for potential correlations between gene expression and a variety of variables attached to the samples I was given including sex, age, peanut IgE levels (which are associated with allergic response), skin prick test results, and many more. Overall, I saw a slight correlation with peanut IgE and a slightly stronger one with age.

After all the data preparation, I got to the fun part of the project (in my opinion), I built a heatmap. Now heatmaps of gene expression take the number of reads (more reads = more expression) and color the value of each gene, here red means it is expressed more and blue means it is expressed less. Now not only does it color code, but heatmaps sort each sample based on similar gene expression. To get the genes I wanted to use in the heatmap, I met with Justine at BRI who gave me a list of genes that were of interest in allergic response to the Wambre lab at BRI. I used this list and created a heatmap of gene expression for those 21 genes. As you can see, two vague groups can be seen, one in the top left, and the other in the bottom right. This was interesting as the genes in the top half were more related to one of two different cell types already being explored by the Wambre lab, called CRTH2+ and the bottom half related to the other cell type, CCR6+. In several papers published by the Wambre lab over the last couple of years, they have identified these two cell populations. In this heatmap, these groupings can be seen as well, with the genes relating to a CCR6+ cell type also being related to a Th17 cell type that was thought to be correlated.

Before building the second heatmap to explore the correlation further, I noticed there was a slight trend in age in the first heatmap. Because of this, I used R Studio to plot the quantity of the CRTH2 and CCR6 proteins on the cells, and colored the data by age. Here we see that older ages roughly tend to have less CRTH2 and more CCR6 while younger ages have more CRTH2 and less CCR6. Previous studies show that CRTH2 is a marker for TH2A, and that a TH2A phenotype is associated with an allergic response. However, those with medium to low TH2A cell levels can also be allergic. Usually those with low TH2A and allergies have high CCR6 instead, it is important to note that less is known about the role of CCR6+ cells in allergy. This plot indicates that these cell types could explain some heterogeneity in allergic response or how allergies may change throughout a person's life.

I then set up the data for the second heatmap slightly differently, three clinical trials and some general samples that BRI had on file were used. I began with 641 samples, which after filtering to baseline and peanut stimulation, I also sorted by whether the cells were CRTH2+ or

CCR6+ positive. I had 34 CCR6+ and 24 CRTH2+ samples prior to quality control of the RNA Sequencing itself. To increase the number of samples used, I included one of the clinical trials that used a skin prick test to test for peanut allergy, despite it not being the gold standard. In the end, I had 48 samples to use for my second heatmap, less than what was used for the first but still decent. From there, I used PCA to get a list of genes that differentiated the most between the two cell types. This produced a list of genes sorted by differentiation which I chose the top 50 of.

I used these genes in a heatmap and here we can see much more distinct grouping. There are a few samples that are a little unexpected, but overall you can see a trend between the two cell types. On this heatmap, I've highlighted the genes that were on the list of important proteins given to me by Justine. As expected, the CRTH2+ genes align with the CRTH2+ cell type. However, it is interesting that FOXP3 and IL1R1 are in the CCR6+ cell type as these are regulatory proteins and associated with a Th17 expression. These findings were consistent with previous studies in that CRTH2+ cells have been studied much more than CCR6+ cell types, and previous evidence suggests CCR6+ cells could have a regulatory function.

Overall, both heatmaps support the concept that within CD4+ memory populations two subgroups exist: one with a CRTH2+ phenotype and TH2A function, and the other with a CCR6+ phenotype and regulatory function perhaps similar to Th17 cells. Many of the genes in the bottom half of the first heatmap are known Th17 markers, suggesting that one of these cell types may have a Th17-like phenotype. Most known allergic related genes have high expression in the CRTH2+ subgroup which is consistent with our prior knowledge and therefore having genes that are better studied. FOXP3 and IL1R1 are known to have functions that are either regulatory in nature or related to a Th17-like phenotype.

Heterogeneity of common proteins associated with allergic disease has been found in CD154+ T cells from individuals with peanut allergies, which has made it difficult to identify an accurate biological measure of allergic disease.[9] That heterogeneity may be the result of differing cell populations, as studies have seen seemingly "paradoxical" results with a cell type high in one patient and low in the other while both exhibit intense allergic response.[20] By identifying and defining differing cell populations, a biological measure of allergies that does not risk individual health, such as a DBPCFC, could be developed. This project has added evidence to previous studies that there may be a regulatory / TH17 signature in CCR6+ peanut-reactive cells.

Next steps should include a longitudinal study looking at CRTH2+ and CCR6+ populations over time in order to further examine a correlation between age and these two cell types as well as further study of CCR6+ associated genes. The existence of two genetically and phenotypically different cell types may indicate that food allergy is not consistent between individuals but may in fact be much more nuanced, indicating that a more nuanced approach to this disease could be necessary to help those afflicted.

I have spent the last four years of my life getting a degree in cellular and molecular biology with two driving factors: 1) I am curious, and 2) I want to help people, and I believe these traits are inherently human traits that drive many of us into research.

To be human is to be curious, even the youngest of us asks, why is the sky blue? What's your favorite color? Why do apples fall out of trees? And this curiosity does not go away, we explore the depths of the sea and light years out into space. We search for a cure for cancer

and examine mesophotic macroalgal abundance in the Puget Sound. Some of these have practical applications and some have less obvious ones, but that does not stop us from always reaching beyond, and doing our research. During this project I felt that I was always looking for more, building a second heatmap, wishing to build my own study, I truly felt the call of knowing more. However, curiosity has not been my only drive.

While writing my honor's project paper I ran into an issue that I think resonates well with why we do our research. I asked Dr. Tenlen, one of my advisors, for some feedback on my paper, and one thing she noticed was that I did not communicate why this project mattered. Sure, I had done some interesting things that I thought were super cool, but the *why* was missing. I thought about it, and spent hours diving into research and I came up with a couple paragraphs that connected my research to the research of others, and how it could help us understand peanut allergies. By combining our knowledge with others, the importance of that knowledge begins to grow. The introductions and discussions of our papers focus on the <u>why</u> of our papers, and they often boil down to the idea that this research could help others in some way, shape, or form. In short, we look to understand our world around us so that we may help the people around us. While these ideas are surely influenced by my identity as a Christian, these ideas are not limited to a Christian ideology. Research in itself can absolutely be considered an act of service, and is but one way that we express our humanity.

Supplemental

Supplemental 1: CD154_P193_P350_P354.R Code

```
library(dplyr)
library(ggplot2); theme_set(theme_bw(20) +
                        theme(panel.grid.major = element_blank(),
                            panel.grid.minor = element_blank(),
                            panel.border = element_rect(colour="black", fill=NA, size=1),
                            axis.text=element_text(colour="black"),
                            axis.ticks=element_line(colour="black"),
                            legend.key = element_blank(),
                            text = element_text(size=20),
                            strip.text.x = element_text(size = 10, margin = margin (b = 2, t = 2)),
                            strip.background = element_rect(fill="white", colour="black")))
library(knitr)
library(ggthemes)
library(ggbeeswarm)
library(viridis)
library(stringr)
library(readxl)
library(RColorBrewer)
library(plotly)
library(tidyr)
library(gtools)
#library(annotables)
#If annotables is needed, use the following lines of code to get it
#install.packages("devtools")
#devtools::install_github("stephenturner/annotables")
library(data.table)
library(edgeR) #from bioconductor
library(limma) #from bioconductor
library(ggrepel)
library(ComplexHeatmap)
library(inlmisc) #For colors
library(umap)
library(forcats)
library(RNAseQC)
library(circlize)

setwd("/Users/stekorius/Box/2021_Sophia")

load("data/P193_4AR101CountsAndDesign.Rdata")
P193_counts<- t(counts)
```

```
P193_design <- design
load("data/P350_2AstellasCountsAndDesign.Rdata")
P350_counts <- t(counts)
P350_design <- design
load("data/P354IMPACTCountsAndDesign.Rdata")
P354_counts <- t(counts)
P354_design <- design
load("data/P392_1Data.Rdata")
p392_anno<-anno
p392_counts<-counts
p392_metrics<-metrics
##Merge p392 data by libid
#p392_metrics$libid <- str_extract(p392_metrics$libid_fcid, "lib[0-9]+")
#p392_design <- full_join(p392_anno, p392_metrics, by = c("libid"))
#p392_counts <- t(p392_counts)
#colnames(p392_counts) <- str_extract(colnames(p392_counts), "lib[0-9]+")

#Set up directories for plotting
base_dir <- getwd()
data_dir <- file.path(base_dir, "data")
plots_dir <- file.path(base_dir, "plots")
tables_dir <- file.path(base_dir, "tables")

#set up color schemes
#project_colors <- c("P354-1" = "blue", "P350-2" = "firebrick", "P193-4" = "gold", "P392" =
"white" )

project_colors <- GetColors(4, scheme = "high-contrast") %>%  as.vector()
names(project_colors) <- c("P193-4",
                           "P350-2",
                           "P354-1")


#rename columns for consistency across files
idkey_P354 <- data.frame("original" =  c("Peanut IgE","Peanut IgG4", "Ara h 1 IgE", "Ara h 1
IgG4", "Ara h 2 IgE",
                                         "Ara h 2 IgG4", "Ara h 3 IgE", "Ara h 3 IgG4", "Ara h 6 IgE", "Ara h 6
IgG4",
                                         "dose", "spt", "studyVisit"),
                         "new" = c("peanut_IgE", "peanut_IgG4", "ara_h1_IgE", "ara_h1_IgG4",
"ara_h2_IgE",
                                   "ara_h2_IgG4", "ara_h3_IgE", "ara_h3_IgG4", "ara_h6_IgE",
"ara_h6_IgG4",
```

```
                          "mtd", "spt_baseline", "visit"))

colnames(P354_design) <- dplyr::recode(
  colnames(P354_design),
  !!!setNames(as.character(idkey_P354$new), idkey_P354$original))

#rename P193
idkey_P193 <- data.frame("original" =  c("ige","igg4","cd154_freq",
                                 "ccr6_pos", "cd27_pos", "th2a", "cell_sort"),
                         "new" = c("peanut_IgE", "peanut_IgG4", "CD154freq",
                                 "CCR6pct", "CD27pct","CRTH2pct", "cellType"))

colnames(P193_design) <- dplyr::recode(
  colnames(P193_design),
  !!!setNames(as.character(idkey_P193$new), idkey_P193$original))

#rename P350
idkey_P350 <- data.frame("original" =  c("Peanut IgE","Peanut IgG4", "Peanut IgG", "Ara h 1
IgE", "Ara h 1 IgG4", "Ara h 2 IgE",
                                 "Ara h 2 IgG4", "Ara h 3 IgE", "Ara h 3 IgG4", "ara_h1_IgG",
"ccr4_pct",
                                 "baselineMaxTolDose", "baselineSPT", "timePoint", "Ara h 2 IgG",
"cd154_freq",
                                 "ccr6_pct", "cd27_pct", "cd38_pct", "cd49b_pct", "crth2_pct",
                                 "cxcr3_pct", "lag3_pct", "tim3_pct", "Ara h 6 IgE", "Ara h 6 IgG4",
                                 "Ara h 1 IgG", "Ara h 3 IgG"),
                         "new" = c("peanut_IgE", "peanut_IgG4", "peanut_IgG", "ara_h1_IgE",
"ara_h1_IgG4", "ara_h2_IgE",
                                 "ara_h2_IgG4", "ara_h3_IgE", "ara_h3_IgG4", "ara_h1_IgG", "CCR4pct",
                                 "mtd", "spt_baseline", "visit", "ara_h2_IgG", "CD154freq",
                                 "CCR6pct", "CD27pct","CD38pct", "CD49bpct", "CRTH2pct",
                                 "CXCR3pct", "LAG3pct", "TIM3pct", "ara_h6_IgE", "ara_h6_IgG4",
                                 "ara_h1_IgG", "ara_h3_IgG"))

colnames(P350_design) <- dplyr::recode(
  colnames(P350_design),
  !!!setNames(as.character(idkey_P350$new), idkey_P350$original))

#stats for each project
#age
mean(P193_design$age, na.rm = TRUE)
project_stats$P193 <- sd(P193_design$age, na.rm = TRUE)
```

```
mean(P350_design$age, na.rm = TRUE)
sd(P350_design$age, na.rm = TRUE)

mean(P354_design$age, na.rm = TRUE)
sd(P354_design$age, na.rm = TRUE)

#baseline
table(P193_design$visit)
table(P350_design$visit)
table(P354_design$visit)

#IgE
mean(P193_design$peanut_IgE, na.rm = TRUE)
sd(P193_design$peanut_IgE, na.rm = TRUE)

mean(P350_design$peanut_IgE, na.rm = TRUE)
sd(P350_design$peanut_IgE, na.rm = TRUE)

mean(P354_design$peanut_IgE, na.rm = TRUE)
sd(P354_design$peanut_IgE, na.rm = TRUE)

#gender
table(P193_design$sex)
table(P350_design$sex)
table(P354_design$sex)


#Merge p193 data by libid
P193_design$libid <- str_extract(P193_design$libid_fcid, "lib[0-9]+")
P193_counts <- t(P193_counts)
colnames(P193_counts) <- str_extract(colnames(P193_counts), "lib[0-9]+")

#Merge p350 data by libid
P350_design$libid <- str_extract(P350_design$libid_fcid, "lib[0-9]+")
P350_counts <- t(P350_counts)
colnames(P350_counts) <- str_extract(colnames(P350_counts), "lib[0-9]+")

#Merge p354 data by libid
P354_design$libid <- str_extract(P354_design$libid_fcid, "lib[0-9]+")
P354_counts <- t(P354_counts)
colnames(P354_counts) <- str_extract(colnames(P354_counts), "lib[0-9]+")

#Merge count data
```

```
P193_counts <- as.data.frame(P193_counts)
P350_counts <- as.data.frame(P350_counts)
P354_counts <- as.data.frame(P354_counts)
counts <- bind_cols(P193_counts, P350_counts)
counts<- bind_cols(counts, P354_counts)

#combine data
P354_design$donorID <- as.character(P354_design$donorID)
design <- bind_rows(P354_design, P350_design)
design <- bind_rows(design, P193_design)

counts <- as.data.frame(t(counts))
design_counts <- bind_cols(design, counts)

#rename all baseline terms (T1, V0, V1) to 'Baseline'
design$visit <- recode(design$visit, "T1"="Baseline", "V0"="Baseline", "V1"="Baseline")
design_counts$visit <- recode(design_counts$visit, "T1"="Baseline", "V0"="Baseline",
"V1"="Baseline")

#rename all peanut stimulant types to 'peanut extract' ("Peanut peptide ON", "Arah1", "Arah2",
"Arah3", "peanut extract")
design$stimulation <- recode(design$stimulation, "Peanut peptide ON"="peanut extract",
"Peanut extract ON"="peanut extract",
                             "Arah1"="peanut extract", "Arah2"="peanut extract", "Arah3"="peanut
extract",
                             "peanut"="peanut extract")

#filter to baseline samples only
design$baseline<-design$visit == "Baseline"

design <- design %>%
  dplyr::filter(baseline ==TRUE)

#filter out alder and only 3 hours of stimulation with peanut
design$stimulation_peanut <- design$stimulation == "peanut extract"

design <- design %>%
  dplyr::filter(stimulation_peanut == TRUE)

#filter to CD154+ samples
design$CD154_true <- design$cellType == "CD154+"

design <- design %>%
```

```
  dplyr::filter(CD154_true ==TRUE)

#quality control
align_cut <- 65
total_reads_cut <- 0.5
median_cv_cut <-1.1

design$qc_pass <- design$fastq_total_reads > total_reads_cut*10^6 &
  design$pct_aligned > align_cut &
  design$median_cv_coverage < median_cv_cut

#graph of percent aligned cutoff
g_pct_aligned <- ggplot(data = design[design$fastq_total_reads > total_reads_cut*10^6, ]) +
  geom_point(aes(x=median_cv_coverage,
                 y=pct_aligned,
                 color=project),
             size=1.5) +
  labs(x="Median CV coverage",
       y = "Percent alignment",
       color="")+
  geom_hline(yintercept =  align_cut)+
  geom_vline(xintercept =  median_cv_cut)+
  ylim(c(0,100))+
  xlim(c(0,2))+
  theme(text = element_text(size=12))

print(g_pct_aligned)

#quality control filter
design_qc <- design %>%
  dplyr::filter(qc_pass ==TRUE)

counts <- t(counts)
counts <- as.data.frame(counts)
counts_qc <- counts[,colnames(counts) %in% design_qc$libid]

#What percent of libraries pass QC?
pct_pass_qc <- round(nrow(design_qc)/nrow(design) *100)
pct_pass_qc

#for each project?
table(design_qc$project)/table(design$project)*100
```

```
#Get protein coding genes with HGNC symbols
gene_key <- read.table(file.path(data_dir,"EnsembleToHGNC_GRCh38.txt"), header =
TRUE,sep = "\t",na.strings = "")
genes_hgnc <- gene_key[!is.na(gene_key$HGNC.symbol),]

counts_hgnc <- counts_qc[rownames(counts_qc) %in% genes_hgnc$Ensembl.Gene.ID,]

counts <- t(counts)
counts <- as.data.frame(counts)

genes_pc <- subset(genes_hgnc, genes_hgnc$Gene.type == "protein_coding") #21119
genes_pc <- genes_pc[!duplicated(genes_pc$Ensembl.Gene.ID),] #remove duplicated ensembl
genes #21117
counts_pc <- merge(genes_pc, counts_qc, by.x="Ensembl.Gene.ID", by.y ="row.names")
gene_key_pc <- counts_pc[,1:3] #First three columns contain annotation information
counts_pc <- counts_pc[,4:ncol(counts_pc),] #The remaining columns contain counts
information
rownames(counts_pc) <- gene_key_pc[,1]

#number of protein-coding genes
nrow(counts_pc)

#filter baseline
design_qc <- design_qc %>%
  dplyr::filter(qc_pass ==TRUE)

## filter lowly expressed genes

#Define a function to filter out lowly expressed genes
gene_filter <- function(counts_in,
                        per_cutoff = 0.1,
                        counts_cutoff = 1){
  #Keep genes with cpm of at least counts_cutoff in at least per_cutoff fraction of libraries
  #CPM normalize
  counts_cpm_norm <- as.data.frame(t(t(counts_in*10^6)/colSums(counts_in)))

  #Filter out lowly expressed genes
  keepRows <- rowSums((counts_cpm_norm) >= counts_cutoff) >=
per_cutoff*ncol(counts_cpm_norm)
  counts_filtered <- counts_in[keepRows,]
```

```
  return(counts_filtered)}

#Run function to filter lowly expressed genes
#counts_all_filtered <- gene_filter(counts_hgnc, 0.10) #all genes with HGNC symbols
counts_pc_filtered <- gene_filter(counts_pc, 0.10, 1)

normalize_counts <- function(counts_in, method){
  #normalize using tmm or deconvolution
  #tmm is good for bulk RNAseq
  #deconvolution is best for large datasets of single cell RNAseq
  #deconvolution is NOT recommended for smaller datasets (less than a few hundred cells)

  if(method == "decon"){
    #Normalize using the deconvolution algorithm
    decon_norm_factors <- computeSumFactors(as.matrix(counts_in))
    counts_norm <- as.data.frame(t(t(counts_in)/decon_norm_factors))
  }

  if(method == "tmm"){
    #Normalize using the TMM algorithm
    dge <- DGEList(counts_in)
    dge <- calcNormFactors(dge)
    counts_norm <- cpm(dge, normalized.lib.sizes=TRUE)
  }

  return(counts_norm)

}

counts_pc_norm <- normalize_counts(counts_pc_filtered, "tmm")
#counts_all_norm <- normalize_counts(counts_all_filtered, "tmm")

#Write out normalized counts
#Add HGNC symbols to gene names
#log 2 transform

counts_out <- log2(counts_pc_norm+1)

write.csv(counts_out, file.path(tables_dir,"TMM_normalized_log2_transformed_counts.csv"),
          quote=FALSE, row.names = TRUE)

counts_pc_norm <- as.data.frame(counts_pc_norm)
idkey <- data.frame("original" =  c(as.character(gene_key_pc$Ensembl.Gene.ID)),
```

```
                "new" = c(as.character(gene_key_pc$HGNC.symbol)))

rownames(counts_pc_norm) <- dplyr::recode(
  rownames(counts_pc_norm),
  !!!setNames(as.character(idkey$new), as.character(idkey$original)))

#remove batch effects
counts_rbe <- removeBatchEffect(log2(counts_pc_norm+1),
                            batch = design_qc$project)

#Run PCA on the normalized log2 transformed counts data
#pca = prcomp(log2(as.data.frame(t(counts_out))+1), center=TRUE, scale=FALSE)
#screeplot(pca)

pca = prcomp((as.data.frame(t(counts_rbe))), center=TRUE, scale=FALSE)
screeplot(pca)

#Get PCA results and merge with sample information stored in metrics
pca_scores= as.data.frame(pca$x)

pdatscores <- merge(design_qc, pca_scores, by.x = "libid", by.y="row.names")

#look at potential correlations
pc_cors <- calc_PCcors(pca, design_qc, id_col = "libid") %>%
  t()

#remove highest age (age is 52 in row 78) for PC1vPC2 age graph ***
#pdatscores_age <- pdatscores[-c(55),]

#Make a plot of PC1 vs PC2. Color by
g_PC1vPC2 <- ggplot(data = pdatscores,
                aes(x=PC1,
                    y=PC2,
                    color=project)) +
  geom_point()
# + scale_color_viridis()

print(g_PC1vPC2)

pdf(file.path(plots_dir,
              "PC1vPC2_project.pdf"),
    height=4,
    width = 6)
```

```
print(g_PC1vPC2)

invisible(dev.off())

#Make a plot of PC3 vs PC4. Color by
g_PC3vPC4 <- ggplot(data = pdatscores,
                   aes(x=PC3,
                       y=PC4,
                       color=log10(peanut_IgE))) +
  geom_point()+
  scale_color_viridis()

print(g_PC3vPC4)

pdf(file.path(plots_dir,
              "PC1=3vPC4.pdf"),
    height=4,
    width = 6)

print(g_PC3vPC4)

invisible(dev.off())



#model gene expression, linear modeling for peanut_IgE
design_model <- design_qc %>%
  dplyr::filter(!is.na(peanut_IgE))

counts_model <- counts_pc_norm[,design_model$libid]

design_matrix <- model.matrix(~design_model$peanut_IgE)

#simplify columns names so contrasts are easier to make later
colnames(design_matrix) <- c("Intercept", "peanut_IgE")

#voom_design <- voom(counts_model, design= design_matrix, plot=TRUE, span=0.2)

vwts_design <- voomWithQualityWeights(counts_model, design= design_matrix, plot=FALSE,
span=0.2)

#Check for patterns of weights that vary with group
```

```
#design_qc$sampleWeight <-
vwts_design$targets$sample.weights[match(design_qc$libid,rownames(vwts_design$targets))]

#ggplot(design_qc,
#       aes(x = libid,
#           y = sampleWeight,
#           fill = project))+
#  geom_col()

# fit model)
vfit <-
  lmFit(vwts_design)

vfit_eB <- eBayes(vfit)

#Get top genes that vary with peanut_IgE
top_genes_peanut_IgE <- topTable(vfit_eB, coef = "peanut_IgE", sort.by = "P", number = Inf)

#save table as excel file
write.csv(top_genes_peanut_IgE, file.path(tables_dir,"top_genes_IgE.csv"),
          quote=FALSE, row.names = TRUE)



#model gene expression, linear modeling for peanut_IgG4
design_model <- design_qc %>%
  dplyr::filter(!is.na(peanut_IgG4))

counts_model <- counts_pc_norm[,design_model$libid]

design_matrix <- model.matrix(~design_model$peanut_IgG4)

#simplify columns names so contrasts are easier to make later
colnames(design_matrix) <- c("Intercept", "peanut_IgG4")

#voom_design <- voom(counts_model, design= design_matrix, plot=TRUE, span=0.2)

vwts_design <- voomWithQualityWeights(counts_model, design= design_matrix, plot=FALSE,
span=0.2)

#Check for patterns of weights that vary with group
#design_qc$sampleWeight <-
vwts_design$targets$sample.weights[match(design_qc$libid,rownames(vwts_design$targets))]
```

```
#ggplot(design_qc,
#       aes(x = libid,
#           y = sampleWeight,
#           fill = project))+
#  geom_col()

# fit model)
vfit <-
  lmFit(vwts_design)

vfit_eB <- eBayes(vfit)

#Get top genes that vary with peanut_IgG4
top_genes_peanut_IgG4 <- topTable(vfit_eB, coef = "peanut_IgG4", sort.by = "P", number =
Inf)

#save table as excel file
write.csv(top_genes_peanut_IgG4, file.path(tables_dir,"top_genes_IgG4.csv"),
          quote=FALSE, row.names = TRUE)



#model gene expression, linear modeling for spt_baseline
design_model <- design_qc %>%
  dplyr::filter(!is.na(spt_baseline))

counts_model <- counts_pc_norm[,design_model$libid]

design_matrix <- model.matrix(~design_model$spt_baseline)

#simplify columns names so contrasts are easier to make later
colnames(design_matrix) <- c("Intercept", "spt_baseline")

#voom_design <- voom(counts_model, design= design_matrix, plot=TRUE, span=0.2)

vwts_design <- voomWithQualityWeights(counts_model, design= design_matrix, plot=FALSE,
span=0.2)

#Check for patterns of weights that vary with group
#design_qc$sampleWeight <-
vwts_design$targets$sample.weights[match(design_qc$libid,rownames(vwts_design$targets))]
```

```
#ggplot(design_qc,
#       aes(x = libid,
#           y = sampleWeight,
#           fill = project))+
#  geom_col()

# fit model)
vfit <-
  lmFit(vwts_design)

vfit_eB <- eBayes(vfit)

#Get top genes that vary with spt_baseline
top_genes_spt_baseline <- topTable(vfit_eB, coef = "spt_baseline", sort.by = "P", number = Inf)

#save table as excel file
write.csv(top_genes_spt_baseline, file.path(tables_dir,"top_genes_spt_baseline.csv"),
          quote=FALSE, row.names = TRUE)



#model gene expression, linear modeling for mtd
design_model <- design_qc %>%
  dplyr::filter(!is.na(mtd))

counts_model <- counts_pc_norm[,design_model$libid]

design_matrix <- model.matrix(~design_model$mtd)

#simplify columns names so contrasts are easier to make later
colnames(design_matrix) <- c("Intercept", "mtd")

#voom_design <- voom(counts_model, design= design_matrix, plot=TRUE, span=0.2)

vwts_design <- voomWithQualityWeights(counts_model, design= design_matrix, plot=FALSE,
span=0.2)

#Check for patterns of weights that vary with group
#design_qc$sampleWeight <-
vwts_design$targets$sample.weights[match(design_qc$libid,rownames(vwts_design$targets))]

#ggplot(design_qc,
#       aes(x = libid,
```

```
#           y = sampleWeight,
#           fill = project))+
#  geom_col()

# fit model)
vfit <-
  lmFit(vwts_design)

vfit_eB <- eBayes(vfit)

#Get top genes that vary with mtd
top_genes_mtd <- topTable(vfit_eB, coef = "mtd", sort.by = "P", number = Inf)

#save table as excel file
write.csv(top_genes_mtd, file.path(tables_dir,"top_genes_mtd.csv"),
          quote=FALSE, row.names = TRUE)



#model gene expression, linear modeling for CD154freq
design_model <- design_qc %>%
  dplyr::filter(!is.na(CD154freq))

counts_model <- counts_pc_norm[,design_model$libid]

design_matrix <- model.matrix(~design_model$CD154freq)

#simplify columns names so contrasts are easier to make later
colnames(design_matrix) <- c("Intercept", "CD154freq")

#voom_design <- voom(counts_model, design= design_matrix, plot=TRUE, span=0.2)

vwts_design <- voomWithQualityWeights(counts_model, design= design_matrix, plot=FALSE,
span=0.2)

#Check for patterns of weights that vary with group
#design_qc$sampleWeight <-
vwts_design$targets$sample.weights[match(design_qc$libid,rownames(vwts_design$targets))]

#ggplot(design_qc,
#       aes(x = libid,
#           y = sampleWeight,
#           fill = project))+
```

```
#  geom_col()

# fit model)
vfit <-
  lmFit(vwts_design)

vfit_eB <- eBayes(vfit)

#Get top genes that vary with CD154freq
top_genes_CD154freq <- topTable(vfit_eB, coef = "CD154freq", sort.by = "P", number = Inf)

#save table as excel file
write.csv(top_genes_CD154freq, file.path(tables_dir,"top_genes_CD154freq.csv"),
          quote=FALSE, row.names = TRUE)




#model gene expression, linear modeling for CCR6pct
design_model <- design_qc %>%
  dplyr::filter(!is.na(CCR6pct))

counts_model <- counts_pc_norm[,design_model$libid]

design_matrix <- model.matrix(~design_model$CCR6pct)

#simplify columns names so contrasts are easier to make later
colnames(design_matrix) <- c("Intercept", "CCR6pct")

#voom_design <- voom(counts_model, design= design_matrix, plot=TRUE, span=0.2)

vwts_design <- voomWithQualityWeights(counts_model, design= design_matrix, plot=FALSE,
span=0.2)

#Check for patterns of weights that vary with group
#design_qc$sampleWeight <-
vwts_design$targets$sample.weights[match(design_qc$libid,rownames(vwts_design$targets))]

#ggplot(design_qc,
#       aes(x = libid,
#           y = sampleWeight,
#           fill = project))+
#  geom_col()
```

```
# fit model)
vfit <-
  lmFit(vwts_design)

vfit_eB <- eBayes(vfit)

#Get top genes that vary with CCRpct
top_genes_CCR6pct <- topTable(vfit_eB, coef = "CCR6pct", sort.by = "P", number = Inf)

#save table as excel file
write.csv(top_genes_CCR6pct, file.path(tables_dir,"top_genes_CCR6pct.csv"),
          quote=FALSE, row.names = TRUE)



#model gene expression, linear modeling for CD27pct
design_model <- design_qc %>%
  dplyr::filter(!is.na(CD27pct))

counts_model <- counts_pc_norm[,design_model$libid]

design_matrix <- model.matrix(~design_model$CD27pct)

#simplify columns names so contrasts are easier to make later
colnames(design_matrix) <- c("Intercept", "CD27pct")

#voom_design <- voom(counts_model, design= design_matrix, plot=TRUE, span=0.2)

vwts_design <- voomWithQualityWeights(counts_model, design= design_matrix, plot=FALSE,
span=0.2)

#Check for patterns of weights that vary with group
#design_qc$sampleWeight <-
vwts_design$targets$sample.weights[match(design_qc$libid,rownames(vwts_design$targets))]

#ggplot(design_qc,
#       aes(x = libid,
#           y = sampleWeight,
#           fill = project))+
#  geom_col()

# fit model)
```

```
vfit <-
  lmFit(vwts_design)

vfit_eB <- eBayes(vfit)

#Get top genes that vary with CD27pct
top_genes_CD27pct <- topTable(vfit_eB, coef = "CD27pct", sort.by = "P", number = Inf)

#save table as excel file
write.csv(top_genes_CD27pct, file.path(tables_dir,"top_genes_CD27pct.csv"),
          quote=FALSE, row.names = TRUE)



#model gene expression, linear modeling for CRTH2
design_model <- design_qc %>%
  dplyr::filter(!is.na(CRTH2pct))

counts_model <- counts_pc_norm[,design_model$libid]

design_matrix <- model.matrix(~design_model$CRTH2pct)

#simplify columns names so contrasts are easier to make later
colnames(design_matrix) <- c("Intercept", "CRTH2pct")

#voom_design <- voom(counts_model, design= design_matrix, plot=TRUE, span=0.2)

vwts_design <- voomWithQualityWeights(counts_model, design= design_matrix, plot=FALSE,
span=0.2)

#Check for patterns of weights that vary with group
#design_qc$sampleWeight <-
vwts_design$targets$sample.weights[match(design_qc$libid,rownames(vwts_design$targets))]

#ggplot(design_qc,
#       aes(x = libid,
#           y = sampleWeight,
#           fill = project))+
#  geom_col()

# fit model)
vfit <-
  lmFit(vwts_design)
```

```
vfit_eB <- eBayes(vfit)

#Get top genes that vary with CRTH2
top_genes_CRTH2pct <- topTable(vfit_eB, coef = "CRTH2pct", sort.by = "P", number = Inf)

#save table as excel file
write.csv(top_genes_CRTH2pct, file.path(tables_dir,"top_genes_CRTH2pct.csv"),
          quote=FALSE, row.names = TRUE)



#look at specific genes and variables of interest
sel_genes <- c("IL9", "IL17RB", "GEMIN5")
sel_counts <- counts_pc_norm[sel_genes,]
sel_counts <- t(sel_counts) #Rotate so genes are columns
sel_counts <- as.data.frame(sel_counts) #make a data frame
sel_counts <- log2(sel_counts+1) #Log transform counts
sel_counts$libid <- rownames(sel_counts) #make libids a column

#Merge, take log2(counts+1)
sel_design_counts <- left_join(design_qc, sel_counts, by = "libid")

sel_design_counts$log_peanut_IgE <- log10(sel_design_counts$peanut_IgE)
sel_design_counts$log_peanut_IgG4 <- log10(sel_design_counts$peanut_IgG4)

#sel_vars <- c("mtd",
#              "spt_baseline")
sel_vars <- c("log_peanut_IgE",
              "log_peanut_IgG4")

#Make longer in terms of selected clinical variables
design_counts_long_var <- sel_design_counts %>%
  pivot_longer(cols = sel_vars,
               names_to = "clinical_var",
               values_to = "clinical_value")

#Make longer in terms of selected genes
design_counts_long_gene <- design_counts_long_var %>%
  pivot_longer(cols = sel_genes,
               names_to = "gene",
               values_to = "counts")
```

```
#plot variables by genes
ggplot(design_counts_long_gene,
       aes(x = clinical_value,
           y = counts)) +
#  scale_color_viridis()+
  geom_point()+
#  xlim(0,15)+
  facet_grid(gene~clinical_var,
             scales = "free")

#heatmap with genes from IgE
genes_for_hmap <- rownames(top_genes_peanut_IgE)
genes_for_hmap <- genes_for_hmap[1:20]

#genes_for_hmap <- c("IL9", "IL17RB", "OSBPL3", "SLA", "IL2RA", "TAGAP", "WDR36",
"IL9R", "CCR4", "IL1RL1",
#                    "IL5", "IL4", "IL13", "PRKCH", "GEMIN5", "OSBPL9", "HGS", "KANSL2",
"MYBBP1A", "RPL7")

#genes_for_hmap <- c("IL5", "IL4", "IL1RL1", "IL17RB", "IL9")


counts_for_hmap <- log2(counts_pc_norm[genes_for_hmap, design_qc$libid]+1)
scaled_counts <- t(scale(t(counts_for_hmap)))

design_qc$log_peanut_IgE <- log10(design_qc$peanut_IgE+1)
design_qc$log_peanut_IgG4 <- log10(design_qc$peanut_IgG4+1)
design_qc$log_CRTH2pct <- log10(design_qc$CRTH2pct +1)

project_colors <- c("P354-1" = "darkblue", "P350-2" = "firebrick", "P193-4" = "gold1")

var_range <- range(design_qc$log_peanut_IgG4, na.rm = TRUE)
var2_range <- range(design_qc$log_peanut_IgE, na.rm = TRUE)
var3_range <- range(log10(design_qc$mtd+1), na.rm = TRUE)
var4_range <- range(design_qc$spt, na.rm = TRUE)
var5_range <- range(log10(design_qc$CD154freq), na.rm = TRUE)
var6_range <- range(design_qc$CCR6pct, na.rm = TRUE)
var7_range <- range(design_qc$CD27pct, na.rm = TRUE)
var8_range <- range(design_qc$CRTH2pct, na.rm = TRUE)
var9_range <- range(design_qc$log_CRTH2pct, na.rm = TRUE)

top_anno <- HeatmapAnnotation(Project = design_qc$project,
                              Age = design_qc$age,
```

```
                    IgG4 = design_qc$log_peanut_IgG4,
                    IgE = design_qc$log_peanut_IgE,
#                     mtd = log10(design_qc$mtd+1),
#                     Spt = design_qc$spt,
#                     CD154freq = log10(design_qc$CD154freq),
                    CCR6pct = design_qc$CCR6pct,
                    CD27pct = design_qc$CD27pct,
#                     CRTH2pct = design_qc$CRTH2pct,
                    CRTH2log = design_qc$log_CRTH2pct,
                    #na_col = "black",
                    col = list(Project = project_colors,
                               Age = colorRamp2(c(0, 35), c("yellow", "red")),
                               IgG4 = colorRamp2(c(0, 1), c("yellow", "red")),
                               IgE = colorRamp2(var2_range, c("yellow", "red")),
#                                mtd = colorRamp2(var3_range, c("yellow", "red")),
#                                Spt = colorRamp2(var4_range, c("yellow", "red")),
#                                CD154freq = colorRamp2(var5_range, c("yellow", "red")),
                               CCR6pct = colorRamp2(var6_range, c("yellow", "red")),
                               CD27pct = colorRamp2(var7_range, c("yellow", "red")),
#                                CRTH2pct = colorRamp2(var8_range, c("yellow", "red")),
                               CRTH2log = colorRamp2(var9_range, c("yellow", "red"))))

#bottom_anno <- HeatmapAnnotation(IgE = design_qc$log_peanut_IgE,
#                         col = list(IgE = colorRamp2(var2_range, c("yellow", "purple"))))

Heatmap(scaled_counts,
        name = "row z-score",
        top_annotation = top_anno,
#       bottom_annotation = bottom_anno,
        show_column_names = FALSE,
        show_row_names = TRUE,
        column_title = "Top Genes from IgE")



#genes for Hmap from CD27pct
#vector with names of genes for heat map
genes_for_hmap <- rownames(top_genes_CD27pct)
genes_for_hmap <- genes_for_hmap[1:20]

design_qc$log_CRTH2pct <- log10(design_qc$CRTH2pct +1)

project_colors <- c("P354-1" = "darkblue", "P350-2" = "firebrick", "P193-4" = "gold1")
```

```
counts_for_hmap <- log2(counts_pc_norm[genes_for_hmap, design_qc$libid]+1)
scaled_counts <- t(scale(t(counts_for_hmap)))

top_anno <- HeatmapAnnotation(Project = design_qc$project,
                        Age = design_qc$age,
                        IgE = design_qc$log_peanut_IgE,
                        CCR6pct = design_qc$CCR6pct,
                        CD27pct = design_qc$CD27pct,
                        CRTH2pct = design_qc$CRTH2pct,
                        col = list(Project = project_colors,
                                Age = colorRamp2(c(0, 35), c("yellow", "red")),
                                IgE = colorRamp2(var2_range, c("yellow", "red")),
                                CCR6pct = colorRamp2(var6_range, c("yellow", "red")),
                                CD27pct = colorRamp2(var7_range, c("yellow", "red")),
                                CRTH2pct = colorRamp2(var8_range, c("yellow", "red"))
                                ))

Heatmap(scaled_counts,
      name = "row z-score",
      top_annotation = top_anno,
      show_column_names = FALSE,
      show_row_names = TRUE,
      column_title = "Top Genes from CD27pct")




#heatmap with genes from CD154
genes_for_hmap <- rownames(top_genes_CCR6pct)
genes_for_hmap <- genes_for_hmap[1:20]

#genes_for_hmap <- c("IL9", "IL17RB", "OSBPL3", "SLA", "IL2RA", "TAGAP", "WDR36",
"IL9R", "CCR4", "IL1RL1",
#                 "IL5", "IL4", "IL13", "PRKCH", "GEMIN5", "OSBPL9", "HGS", "KANSL2",
"MYBBP1A", "RPL7")

#genes_for_hmap <- c("IL5", "IL4", "IL1RL1", "IL17RB", "IL9")


counts_for_hmap <- log2(counts_pc_norm[genes_for_hmap, design_qc$libid]+1)
scaled_counts <- t(scale(t(counts_for_hmap)))
```

```
top_anno <- HeatmapAnnotation(Project = design_qc$project,
                        Age = design_qc$age,
                        IgG4 = design_qc$log_peanut_IgG4,
                        IgE = design_qc$log_peanut_IgE,
                        mtd = log10(design_qc$mtd+1),
                        Spt = design_qc$spt,
#                         CD154freq = log10(design_qc$CD154freq),
                        CCR6pct = design_qc$CCR6pct,
                        CD27pct = design_qc$CD27pct,
                        CRTH2pct = design_qc$CRTH2pct,
                        #na_col = "black",
                        col = list(Project = project_colors,
                                Age = colorRamp2(c(0, 35), c("yellow", "red")),
                                IgG4 = colorRamp2(var_range, c("yellow", "red")),
                                IgE = colorRamp2(var2_range, c("yellow", "red")),
                                mtd = colorRamp2(var3_range, c("yellow", "red")),
                                Spt = colorRamp2(var4_range, c("yellow", "red")),
#                                 CD154freq = colorRamp2(var5_range, c("yellow", "red")),
                                CCR6pct = colorRamp2(var6_range, c("yellow", "red")),
                                CD27pct = colorRamp2(var7_range, c("yellow", "red")),
                                CRTH2pct = colorRamp2(var8_range, c("yellow", "red"))))

#bottom_anno <- HeatmapAnnotation(IgE = design_qc$log_peanut_IgE,
#                             col = list(IgE = colorRamp2(var2_range, c("yellow", "purple"))))

Heatmap(scaled_counts,
      name = "row z-score",
      top_annotation = top_anno,
#       bottom_annotation = bottom_anno,
      show_column_names = FALSE,
      show_row_names = TRUE,
      column_title = "Top Genes from CCR6pct")



#heatmap with genes of interest
genes_for_hmap <- c("CCR7", "RORC", "CD27", "IL17RB", "IL1RL1", "HPGDS", "CD200",
"CCR6", "FOXP3", "IL31",
               "GATA3", "IFNG", "IL17A", "IL17F", "IL10", "IL1R1", "IL4", "IL5", "IL9",
"IL13",
               "DUSP6")
```

```
counts_for_hmap <- log2(counts_pc_norm[genes_for_hmap, design_qc$libid]+1)
scaled_counts <- t(scale(t(counts_for_hmap)))

top_anno <- HeatmapAnnotation(Project = design_qc$project,
                        Age = design_qc$age,
                        #IgG4 = design_qc$log_peanut_IgG4,
                        IgE = design_qc$log_peanut_IgE,
#                         mtd = log10(design_qc$mtd+1),
#                         CD154freq = log10(design_qc$CD154freq),
                        #CCR6pct = design_qc$CCR6pct,
                        #CD27pct = design_qc$CD27pct,
                        #CRTH2pct = design_qc$CRTH2pct,
                        #na_col = "black",
                        col = list(Project = project_colors,
                                Age = colorRamp2(c(0, 35), c("yellow", "red")),
                                IgG4 = colorRamp2(c(0, 1), c("yellow", "red")),
                                IgE = colorRamp2(var2_range, c("yellow", "red")),
#                                 mtd = colorRamp2(var3_range, c("yellow", "red")),
#                                 CD154freq = colorRamp2(var5_range, c("yellow", "red")),
                                CCR6pct = colorRamp2(var6_range, c("yellow", "red")),
                                CD27pct = colorRamp2(var7_range, c("yellow", "red"))
                                #CRTH2pct = colorRamp2(var8_range, c("yellow", "red"))
                                ))

Heatmap(scaled_counts,
       name = "row z-score",
       top_annotation = top_anno,
       show_column_names = FALSE,
       show_row_names = TRUE,
       column_title = "Overall CD154+")

#design_qc$IgG4_limit<-design_qc$peanut_IgG4 < 5

#design_qc_IgG4 <- design_qc %>%
#  dplyr::filter(IgG4_limit ==TRUE)

g_CRTH2_CCR6 <- ggplot(data = design_qc,
                 aes(x=CCR6pct,
                     y=CRTH2pct,
                     color=peanut_IgE)) +
  labs(x="CCR6",
       y = "CRTH2",
       color="IgE")+
```

```
  geom_point(size=2) +
  scale_color_viridis()

print(g_CRTH2_CCR6)
```

Supplemental 2: CCR6_CRTH2_subset.R Code:

```
library(dplyr)
library(ggplot2); theme_set(theme_bw(20) +
                        theme(panel.grid.major = element_blank(),
                            panel.grid.minor = element_blank(),
                            panel.border = element_rect(colour="black", fill=NA, size=1),
                            axis.text=element_text(colour="black"),
                            axis.ticks=element_line(colour="black"),
                            legend.key = element_blank(),
                            text = element_text(size=20),
                            strip.text.x = element_text(size = 10, margin = margin (b = 2, t = 2)),
                            strip.background = element_rect(fill="white", colour="black")))
library(knitr)
library(ggthemes)
library(ggbeeswarm)
library(viridis)
library(stringr)
library(readxl)
library(RColorBrewer)
library(plotly)
library(tidyr)
library(gtools)
#library(annotables)
#If annotables is needed, use the following lines of code to get it
#install.packages("devtools")
#devtools::install_github("stephenturner/annotables")
library(data.table)
library(edgeR) #from bioconductor
library(limma) #from bioconductor
library(ggrepel)
library(ComplexHeatmap)
library(inlmisc) #For colors
library(umap)
library(forcats)
library(RNAseQC)
library(circlize)

setwd("/Users/stekorius/Box/2021_Sophia")

load("data/P193_4AR101CountsAndDesign.Rdata")
P193_counts<- t(counts)
P193_design <- design
load("data/P354IMPACTCountsAndDesign.Rdata")
P354_counts <- t(counts)
P354_design <- design
```

```
load("data/P43BulkCountsAndDesign.Rdata")
P43_counts <- t(counts)
P43_design <- design
P43_design$project <- "P43"
P43_design$visit <- "Baseline"
load("data/P392_1Data.Rdata")
P392_anno<-anno
P392_counts<-counts
P392_metrics<-metrics
P392_design <- bind_cols(P392_anno, P392_metrics)

#Set up directories for plotting
base_dir <- getwd()
data_dir <- file.path(base_dir, "data")
plots_dir <- file.path(base_dir, "plots")
tables_dir <- file.path(base_dir, "tables")

#set up color schemes
project_colors <- c("P354-1" = "blue", "P43" = "firebrick", "P193-4" = "gold", "P392-1" = "white")

#rename columns for consistency across files
idkey_P354 <- data.frame("original" =  c("Peanut IgE","Peanut IgG4", "Ara h 1 IgE", "Ara h 1 IgG4",
"Ara h 2 IgE",
                                         "Ara h 2 IgG4", "Ara h 3 IgE", "Ara h 3 IgG4", "Ara h 6 IgE", "Ara h 6 IgG4",
                                         "dose", "spt", "studyVisit"),
                         "new" = c("peanut_IgE", "peanut_IgG4", "ara_h1_IgE", "ara_h1_IgG4", "ara_h2_IgE",
                                   "ara_h2_IgG4", "ara_h3_IgE", "ara_h3_IgG4", "ara_h6_IgE", "ara_h6_IgG4",
                                   "mtd", "spt_baseline", "visit"))

colnames(P354_design) <- dplyr::recode(
  colnames(P354_design),
  !!!setNames(as.character(idkey_P354$new), idkey_P354$original))

#rename P193
idkey_P193 <- data.frame("original" =  c("ige","igg4","cd154_freq",
                                         "ccr6_pos", "cd27_pos", "th2a", "cell_sort"),
                         "new" = c("peanut_IgE", "peanut_IgG4", "CD154freq",
                                   "CCR6pct", "CD27pct","CRTH2pct", "cellType"))

colnames(P193_design) <- dplyr::recode(
  colnames(P193_design),
  !!!setNames(as.character(idkey_P193$new), idkey_P193$original))

#rename P43
idkey_P43 <- data.frame("original" =  c("cell_type", "treatment.x", "mapped_reads_w_dups",
"MEDIAN_CV_COVERAGE"),
                        "new" = c("cellType", "stimulation", "pct_aligned", "median_cv_coverage"))

colnames(P43_design) <- dplyr::recode(
  colnames(P43_design),
  !!!setNames(as.character(idkey_P43$new), idkey_P43$original))

P43_design$pct_aligned <- P43_design$pct_aligned*100
```

```
#rename P392
idkey_P392 <- data.frame("original" =  c("timePoint"),
                         "new" = c("visit"))

colnames(P392_design) <- dplyr::recode(
  colnames(P392_design),
  !!!setNames(as.character(idkey_P392$new), idkey_P392$original))

#Merge p193 data by libid
P193_design$libid <- str_extract(P193_design$libid_fcid, "lib[0-9]+")
P193_counts <- t(P193_counts)
colnames(P193_counts) <- str_extract(colnames(P193_counts), "lib[0-9]+")
colnames(P392_design) <- dplyr::recode(colnames(P392_design),
                                       !!!setNames("libid", "libid...23"))

#Merge p354 data by libid
P354_design$libid <- str_extract(P354_design$libid_fcid, "lib[0-9]+")
P354_counts <- t(P354_counts)
colnames(P354_counts) <- str_extract(colnames(P354_counts), "lib[0-9]+")

#Merge p43
P43_counts <- t(P43_counts)
colnames(P43_counts) <- str_extract(colnames(P43_counts), "lib[0-9]+")


#Merge p392 data by libid
P392_design$libid <- str_extract(P392_design$libid_fcid, "lib[0-9]+")
P392_counts <- t(P392_counts)
colnames(P392_counts) <- str_extract(colnames(P392_counts), "lib[0-9]+")
colnames(P392_design) <- dplyr::recode(colnames(P392_design),
  !!!setNames("libid", "libid...23"))
colnames(P392_design) <- dplyr::recode(colnames(P392_design),
                                       !!!setNames("x.libid", "libid...41"))

#Merge count data
P193_counts <- as.data.frame(P193_counts)
P354_counts <- as.data.frame(P354_counts)
P43_counts <- as.data.frame(P43_counts)
P392_counts <- as.data.frame(P392_counts)
counts <- bind_cols(P193_counts, P354_counts)
counts <- bind_cols(counts, P392_counts)
P43_counts$EnsembleID <- rownames(P43_counts)
counts$EnsembleID <- rownames(counts)

counts <- left_join(counts, P43_counts, by = "EnsembleID")
rownames(counts) <- counts$EnsembleID

#combine data
P354_design$donorID <- as.character(P354_design$donorID)
design <- bind_rows(P354_design, P193_design)
design <- bind_rows(design, P43_design)
design$dateCreated <- as.character(design$dateCreated)
design$dateUpdated <- as.character(design$dateUpdated)
design$dateReceived <- as.character(design$dateReceived)
```

```
design <- bind_rows(design, P392_design)

#rename all baseline terms (T1, V0, V1) to 'Baseline'
design$visit <- recode(design$visit, "V0"="Baseline", "V1"="Baseline", "Day1"="Baseline")

#rename all peanut stimulant types to 'peanut extract' ("Peanut peptide ON", "Arah1", "Arah2", "Arah3",
"peanut extract")
design$stimulation <- recode(design$stimulation, "Peanut peptide ON"="peanut extract", "Peanut Peptide
ON"="peanut extract",
                             "Peanut extract ON"="peanut extract", "nArah1nArah2"="peanut extract")

#rename all CCR6+ cell types
design$cellType <- recode(design$cellType, "CD154+CCR4+CCR6+"="CCR6+",
"CD154+CCR6+"="CCR6+", "CCR6+ST2-"="CCR6+")

#rename all CRTH2+ cell types
design$cellType <- recode(design$cellType, "CD154+CRTH2+"="CRTH2+")

#filter to baseline samples only
design$baseline<-design$visit == "Baseline"
design <- design %>%
  dplyr::filter(baseline ==TRUE)

#filter out alder and only 3 hours of stimulation with peanut
design$stimulation_peanut <- design$stimulation == "peanut extract"

design <- design %>%
  dplyr::filter(stimulation_peanut == TRUE)

#filter to CCR6+ samples
design$CCR6_true <- design$cellType == "CCR6+"

design_ccr6 <- design %>%
  dplyr::filter(CCR6_true ==TRUE)

#filter to CRTH2+ samples
design$CRTH2_true <- design$cellType == "CRTH2+"

design_crth2 <- design %>%
  dplyr::filter(CRTH2_true ==TRUE)

#combine CRTH2+ and CCR6+
design_CRTH2_CCR6 <- bind_rows(design_crth2, design_ccr6)
counts_CRTH2_CCR6 <- counts

#quality control
align_cut <- 65
total_reads_cut <- 0.5
median_cv_cut <-0.9

design_CRTH2_CCR6$qc_pass <- design_CRTH2_CCR6$fastq_total_reads > total_reads_cut*10^6 &
  design_CRTH2_CCR6$pct_aligned > align_cut &
  design_CRTH2_CCR6$median_cv_coverage < median_cv_cut
```

```
#quality control filter
design_qc_CRTH2_CCR6 <- design_CRTH2_CCR6 %>%
  dplyr::filter(qc_pass ==TRUE)

idkey <- data.frame("original" =  c("libid...27"),
                    "new" = c("libid"))

colnames(design_qc_CRTH2_CCR6) <- dplyr::recode(
  colnames(design_qc_CRTH2_CCR6),
  !!!setNames(as.character(idkey$new), idkey$original))

counts_CRTH2_CCR6 <- as.data.frame(counts_CRTH2_CCR6)
counts_qc_CRTH2_CCR6 <- counts_CRTH2_CCR6[,colnames(counts_CRTH2_CCR6) %in%
design_qc_CRTH2_CCR6$libid]

#What percent of libraries pass QC?
pct_pass_qc_CRTH2_CCR6 <- round(nrow(design_qc_CRTH2_CCR6)/nrow(design_CRTH2_CCR6)
*100)

#Get protein coding genes with HGNC symbols
gene_key <- read.table(file.path(data_dir,"EnsembleToHGNC_GRCh38.txt"), header = TRUE,sep =
"\t",na.strings = "")
genes_hgnc <- gene_key[!is.na(gene_key$HGNC.symbol),]

counts_hgnc <- counts_qc_CRTH2_CCR6[rownames(counts_qc_CRTH2_CCR6) %in%
genes_hgnc$Ensembl.Gene.ID,]

counts_CRTH2_CCR6 <- t(counts_CRTH2_CCR6)
counts_CRTH2_CCR6 <- as.data.frame(counts_CRTH2_CCR6)

genes_pc <- subset(genes_hgnc, genes_hgnc$Gene.type == "protein_coding") #21119
genes_pc <- genes_pc[!duplicated(genes_pc$Ensembl.Gene.ID),] #remove duplicated ensembl genes
#21117
counts_pc <- merge(genes_pc, counts_qc_CRTH2_CCR6, by.x="Ensembl.Gene.ID", by.y ="row.names")
gene_key_pc <- counts_pc[,1:3] #First three columns contain annotation information
counts_pc <- counts_pc[,4:ncol(counts_pc),] #The remaining columns contain counts information
rownames(counts_pc) <- gene_key_pc[,1]

design_qc_CRTH2_CCR6 <- design_qc_CRTH2_CCR6 %>%
  dplyr::filter(qc_pass ==TRUE)

## filter lowly expressed genes
counts_pc <- head(counts_pc, n = 19102)

#Define a function to filter out lowly expressed genes
gene_filter_CRTH2_CCR6 <- function(counts_in,
                          per_cutoff = 0.1,
                          counts_cutoff = 1){
  #Keep genes with cpm of at least counts_cutoff in at least per_cutoff fraction of libraries
  #CPM normalize
  counts_cpm_norm_CRTH2_CCR6 <- as.data.frame(t(t(counts_in*10^6)/colSums(counts_in)))

  #Filter out lowly expressed genes
```

```
  keepRows <- rowSums((counts_cpm_norm_CRTH2_CCR6) >= counts_cutoff) >=
per_cutoff*ncol(counts_cpm_norm_CRTH2_CCR6)
  counts_filtered_CRTH2_CCR6 <- counts_in[keepRows,]

  return(counts_filtered_CRTH2_CCR6)}

counts_pc_filtered_CRTH2_CCR6 <- gene_filter_CRTH2_CCR6(counts_pc, 0.10, 1)

normalize_counts <- function(counts_in, method){
  #normalize using tmm or deconvolution
  #tmm is good for bulk RNAseq
  #deconvolution is best for large datasets of single cell RNAseq
  #deconvolution is NOT recommended for smaller datasets (less than a few hundred cells)

  if(method == "decon"){
    #Normalize using the deconvolution algorithm
    decon_norm_factors <- computeSumFactors(as.matrix(counts_in))
    counts_norm_CRTH2_CCR6 <- as.data.frame(t(t(counts_in)/decon_norm_factors))
  }

  if(method == "tmm"){
    #Normalize using the TMM algorithm
    dge <- DGEList(counts_in)
    dge <- calcNormFactors(dge)
    counts_norm_CRTH2_CCR6 <- cpm(dge, normalized.lib.sizes=TRUE)
  }

  return(counts_norm_CRTH2_CCR6)

}

counts_pc_norm_CRTH2_CCR6 <- normalize_counts(counts_pc_filtered_CRTH2_CCR6, "tmm")
#counts_all_norm <- normalize_counts(counts_all_filtered, "tmm")

#Write out normalized counts
#Add HGNC symbols to gene names
#log 2 transform

counts_out_CRTH2_CCR6 <- log2(counts_pc_norm_CRTH2_CCR6+1)

write.csv(counts_out_CRTH2_CCR6,
file.path(tables_dir,"TMM_normalized_log2_transformed_CRTH2_CCR6_counts.csv"),
          quote=FALSE, row.names = TRUE)

counts_pc_norm_CRTH2_CCR6 <- as.data.frame(counts_pc_norm_CRTH2_CCR6)
idkey <- data.frame("original" =  c(as.character(gene_key_pc$Ensembl.Gene.ID)),
                    "new" = c(as.character(gene_key_pc$HGNC.symbol)))

counts_pc_norm_CRTH2_CCR6 <- counts_pc_norm_CRTH2_CCR6[!
row.names(counts_pc_norm_CRTH2_CCR6) %in% c("ENSG00000280987"),]

rownames(counts_pc_norm_CRTH2_CCR6) <- dplyr::recode(
  rownames(counts_pc_norm_CRTH2_CCR6),
  !!!setNames(as.character(idkey$new), as.character(idkey$original)))
```

```
counts_pc_norm_CRTH2_CCR6$HGNC <- rownames(counts_pc_norm_CRTH2_CCR6)



#remove batch effects
#counts_rbe_CRTH2_CCR6 <- removeBatchEffect(log2(counts_pc_norm_CRTH2_CCR6+1),
#                           batch = design_qc_CRTH2_CCR6$project)

#Run PCA on the normalized log2 transformed counts data
pca_CRTH2_CCR6 = prcomp(log2(as.data.frame(t(counts_out_CRTH2_CCR6))+1), center=TRUE,
scale=FALSE)
screeplot(pca_CRTH2_CCR6)

pca_CRTH2_CCR6 = prcomp((as.data.frame(t(counts_pc_norm_CRTH2_CCR6))), center=TRUE,
scale=FALSE)
screeplot(pca_CRTH2_CCR6)

#Get PCA results and merge with sample information stored in metrics
pca_scores_CRTH2_CCR6= as.data.frame(pca_CRTH2_CCR6$x)

pdatscores_CRTH2_CCR6 <- merge(design_qc_CRTH2_CCR6, pca_scores_CRTH2_CCR6, by.x =
"libid", by.y="row.names")

#look at potential correlations
pc_cors_CRTH2_CCR6 <- calc_PCcors(pca_CRTH2_CCR6, design_qc_CRTH2_CCR6, id_col =
"libid") %>% t()

#remove highest age (age is 52 in row 78) for PC1vPC2 age graph ***
#pdatscores_age <- pdatscores[-c(55),]

#Make a plot of PC1 vs PC2. Color by
g_PC1vPC2_CRTH2_CCR6 <- ggplot(data = pdatscores_CRTH2_CCR6,
                aes(x=PC1,
                    y=PC2,
                    color=project)) +
  geom_point()
# + scale_color_viridis()

print(g_PC1vPC2_CRTH2_CCR6)

#model gene expression, linear modeling for peanut_IgE
design_model_CRTH2_CCR6 <- design_qc_CRTH2_CCR6 %>%
  dplyr::filter(!is.na(cellType))

counts_model_CRTH2_CCR6 <- counts_pc_norm_CRTH2_CCR6[,design_model_CRTH2_CCR6$libid]

design_matrix_CRTH2_CCR6 <- model.matrix(~design_model_CRTH2_CCR6$cellType)

#simplify columns names so contrasts are easier to make later
colnames(design_matrix_CRTH2_CCR6) <- c("Intercept", "cellType")

#voom_design <- voom(counts_model, design= design_matrix, plot=TRUE, span=0.2)
```

```
vwts_design_CRTH2_CCR6 <- voomWithQualityWeights(counts_model_CRTH2_CCR6, design=
design_matrix_CRTH2_CCR6, plot=FALSE, span=0.2)

#Check for patterns of weights that vary with group
#design_qc$sampleWeight <-
vwts_design$targets$sample.weights[match(design_qc$libid,rownames(vwts_design$targets))]

#ggplot(design_qc,
#       aes(x = libid,
#           y = sampleWeight,
#           fill = project))+
#  geom_col()

# fit model)
vfit <-
  lmFit(vwts_design_CRTH2_CCR6)

vfit_eB <- eBayes(vfit)

#Get top genes that vary with peanut_IgE
top_genes_cellType <- topTable(vfit_eB, coef = "cellType", sort.by = "P", number = Inf)
top_genes_cellType$gene <- rownames(top_genes_cellType)

write.csv(top_genes_cellType, file.path(tables_dir,"top_genes_CellType.csv"),
          quote=FALSE, row.names = TRUE)

#heatmap of CCR6+
genes_for_hmap <- c("CCR7", "RORC", "CD27", "IL17RB", "IL1RL1", "HPGDS", "CD200", "CCR6",
"FOXP3", "IL31",
                    "GATA3", "IL10", "IL1R1", "IL4", "IL5", "IL9", "IL13", "CD200R1",
                    "DUSP6")

sig_genes <- top_genes_cellType[top_genes_cellType$adj.P.Val < 0.000001,]
genes_for_hmap <- rownames(sig_genes)

genes_for_hmap <- rownames(top_genes_cellType)
genes_for_hmap <- genes_for_hmap[1:50]

counts_for_hmap_CRTH2_CCR6 <- log2(counts_pc_norm_CRTH2_CCR6[genes_for_hmap,
design_qc_CRTH2_CCR6$libid]+1)
scaled_counts_CRTH2_CCR6 <- t(scale(t(counts_for_hmap_CRTH2_CCR6)))

design_qc_CRTH2_CCR6$log_peanut_IgE <- log10(design_qc_CRTH2_CCR6$peanut_IgE+1)
design_qc_CRTH2_CCR6$log_peanut_IgG4 <- log10(design_qc_CRTH2_CCR6$peanut_IgG4+1)
design_qc_CRTH2_CCR6$log_CRTH2_CCR6pct <- log10(design_qc_CRTH2_CCR6$CRTH2pct +1)


project_colors <- c("P354-1" = "darkblue", "P392-1" = "firebrick", "P193-4" = "gold1", "P43"="gray")

top_anno <- HeatmapAnnotation(Project = design_qc_CRTH2_CCR6$project,
                              CellType = design_qc_CRTH2_CCR6$cellType,
                              na_col = "black",
                              col = list(Project = project_colors,
                                         CellType = c("CCR6+" = "darkblue", "CRTH2+" = "firebrick1")))
```

```
CCR6_CRTH2_heatmap <- Heatmap(scaled_counts_CRTH2_CCR6,
                    name = "row z-score",
                    top_annotation = top_anno,
                    show_column_names = FALSE,
                    show_row_names = TRUE,
                    column_title = "CRTH2 and CCR6 Subsets")

print(CCR6_CRTH2_heatmap)

pdf(file.path(plots_dir,
           "CCR6_CRTH2_top50_heatmap.pdf"),
   height=8,
   width = 12)

print(CCR6_CRTH2_heatmap)

invisible(dev.off())

load("data/P193_4AR101CountsAndDesign.Rdata")
P193_counts<- t(counts)
P193_design <- design
load("data/P354IMPACTCountsAndDesign.Rdata")
P354_counts <- t(counts)
P354_design <- design

#Set up directories for plotting
base_dir <- getwd()
data_dir <- file.path(base_dir, "data")
plots_dir <- file.path(base_dir, "plots")
tables_dir <- file.path(base_dir, "tables")

#set up color schemes
project_colors <- c("P354-1" = "blue", "P193-4" = "gold")

#rename columns for consistency across files
idkey_P354 <- data.frame("original" =  c("Peanut IgE","Peanut IgG4", "Ara h 1 IgE", "Ara h 1 IgG4",
"Ara h 2 IgE",
                                  "Ara h 2 IgG4", "Ara h 3 IgE", "Ara h 3 IgG4", "Ara h 6 IgE", "Ara h 6 IgG4",
                                  "dose", "spt", "studyVisit"),
                         "new" = c("peanut_IgE", "peanut_IgG4", "ara_h1_IgE", "ara_h1_IgG4", "ara_h2_IgE",
                                 "ara_h2_IgG4", "ara_h3_IgE", "ara_h3_IgG4", "ara_h6_IgE", "ara_h6_IgG4",
                                 "mtd", "spt_baseline", "visit"))

colnames(P354_design) <- dplyr::recode(
  colnames(P354_design),
  !!!setNames(as.character(idkey_P354$new), idkey_P354$original))

#rename P193
idkey_P193 <- data.frame("original" =  c("ige","igg4","cd154_freq",
                                  "ccr6_pos", "cd27_pos", "th2a", "cell_sort"),
                         "new" = c("peanut_IgE", "peanut_IgG4", "CD154freq",
                                 "CCR6pct", "CD27pct","CRTH2pct", "cellType"))
```

```
colnames(P193_design) <- dplyr::recode(
  colnames(P193_design),
  !!!setNames(as.character(idkey_P193$new), idkey_P193$original))

#Merge p193 data by libid
P193_design$libid <- str_extract(P193_design$libid_fcid, "lib[0-9]+")
P193_counts <- t(P193_counts)
colnames(P193_counts) <- str_extract(colnames(P193_counts), "lib[0-9]+")

#Merge p354 data by libid
P354_design$libid <- str_extract(P354_design$libid_fcid, "lib[0-9]+")
P354_counts <- t(P354_counts)
colnames(P354_counts) <- str_extract(colnames(P354_counts), "lib[0-9]+")

#Merge count data
P193_counts <- as.data.frame(P193_counts)
P354_counts <- as.data.frame(P354_counts)
counts <- bind_cols(P193_counts, P354_counts)

#combine data
P354_design$donorID <- as.character(P354_design$donorID)
design <- bind_rows(P354_design, P193_design)

#rename all baseline terms (T1, V0, V1) to 'Baseline'
design$visit <- recode(design$visit, "V0"="Baseline", "V1"="Baseline")

#rename all peanut stimulant types to 'peanut extract' ("Peanut peptide ON", "Arah1", "Arah2", "Arah3",
"peanut extract")
design$stimulation <- recode(design$stimulation, "Peanut peptide ON"="peanut extract", "Peanut Peptide
ON"="peanut extract",
                             "Peanut extract ON"="peanut extract")

#rename all CCR6+ cell types
design$cellType <- recode(design$cellType, "CD154+CCR4+CCR6+"="CCR6+",
"CD154+CCR6+"="CCR6+")

#rename all CRTH2+ cell types
design$cellType <- recode(design$cellType, "CD154+CRTH2+"="CRTH2+")

#filter to baseline samples only
design$baseline<-design$visit == "Baseline"
design <- design %>%
  dplyr::filter(baseline ==TRUE)

#filter out alder and only 3 hours of stimulation with peanut
design$stimulation_peanut <- design$stimulation == "peanut extract"

design <- design %>%
  dplyr::filter(stimulation_peanut == TRUE)

#filter to CCR6+ samples
design$CCR6_true <- design$cellType == "CCR6+"

design_ccr6 <- design %>%
```

```
  dplyr::filter(CCR6_true ==TRUE)

#filter to CRTH2+ samples
design$CRTH2_true <- design$cellType == "CRTH2+"

design_crth2 <- design %>%
  dplyr::filter(CRTH2_true ==TRUE)

#combine CRTH2+ and CCR6+
design_CRTH2_CCR6 <- bind_rows(design_crth2, design_ccr6)
counts_CRTH2_CCR6 <- counts

#quality control
align_cut <- 65
total_reads_cut <- 0.5
median_cv_cut <-0.9

design_CRTH2_CCR6$qc_pass <- design_CRTH2_CCR6$fastq_total_reads > total_reads_cut*10^6 &
  design_CRTH2_CCR6$pct_aligned > align_cut &
  design_CRTH2_CCR6$median_cv_coverage < median_cv_cut

#quality control filter
design_qc_CRTH2_CCR6 <- design_CRTH2_CCR6 %>%
  dplyr::filter(qc_pass ==TRUE)

counts_CRTH2_CCR6 <- as.data.frame(counts_CRTH2_CCR6)
counts_qc_CRTH2_CCR6 <- counts_CRTH2_CCR6[,colnames(counts_CRTH2_CCR6) %in%
design_qc_CRTH2_CCR6$libid]

#What percent of libraries pass QC?
pct_pass_qc_CRTH2_CCR6 <- round(nrow(design_qc_CRTH2_CCR6)/nrow(design_CRTH2_CCR6)
*100)

#Get protein coding genes with HGNC symbols
gene_key <- read.table(file.path(data_dir,"EnsembleToHGNC_GRCh38.txt"), header = TRUE,sep =
"\t",na.strings = "")
genes_hgnc <- gene_key[!is.na(gene_key$HGNC.symbol),]

counts_hgnc <- counts_qc_CRTH2_CCR6[rownames(counts_qc_CRTH2_CCR6) %in%
genes_hgnc$Ensembl.Gene.ID,]

counts_CRTH2_CCR6 <- t(counts_CRTH2_CCR6)
counts_CRTH2_CCR6 <- as.data.frame(counts_CRTH2_CCR6)

genes_pc <- subset(genes_hgnc, genes_hgnc$Gene.type == "protein_coding") #21119
genes_pc <- genes_pc[!duplicated(genes_pc$Ensembl.Gene.ID),] #remove duplicated ensembl genes
#21117
counts_pc <- merge(genes_pc, counts_qc_CRTH2_CCR6, by.x="Ensembl.Gene.ID", by.y ="row.names")
gene_key_pc <- counts_pc[,1:3] #First three columns contain annotation information
counts_pc <- counts_pc[,4:ncol(counts_pc),] #The remaining columns contain counts information
rownames(counts_pc) <- gene_key_pc[,1]

design_qc_CRTH2_CCR6 <- design_qc_CRTH2_CCR6 %>%
  dplyr::filter(qc_pass ==TRUE)
```

```
## filter lowly expressed genes

#Define a function to filter out lowly expressed genes
gene_filter_CRTH2_CCR6 <- function(counts_in,
                                  per_cutoff = 0.1,
                                  counts_cutoff = 1){
  #Keep genes with cpm of at least counts_cutoff in at least per_cutoff fraction of libraries
  #CPM normalize
  counts_cpm_norm_CRTH2_CCR6 <- as.data.frame(t(t(counts_in*10^6)/colSums(counts_in)))

  #Filter out lowly expressed genes
  keepRows <- rowSums((counts_cpm_norm_CRTH2_CCR6) >= counts_cutoff) >=
per_cutoff*ncol(counts_cpm_norm_CRTH2_CCR6)
  counts_filtered_CRTH2_CCR6 <- counts_in[keepRows,]

  return(counts_filtered_CRTH2_CCR6)}

counts_pc_filtered_CRTH2_CCR6 <- gene_filter_CRTH2_CCR6(counts_pc, 0.10, 1)

normalize_counts <- function(counts_in, method){
  #normalize using tmm or deconvolution
  #tmm is good for bulk RNAseq
  #deconvolution is best for large datasets of single cell RNAseq
  #deconvolution is NOT recommended for smaller datasets (less than a few hundred cells)

  if(method == "decon"){
    #Normalize using the deconvolution algorithm
    decon_norm_factors <- computeSumFactors(as.matrix(counts_in))
    counts_norm_CRTH2_CCR6 <- as.data.frame(t(t(counts_in)/decon_norm_factors))
  }

  if(method == "tmm"){
    #Normalize using the TMM algorithm
    dge <- DGEList(counts_in)
    dge <- calcNormFactors(dge)
    counts_norm_CRTH2_CCR6 <- cpm(dge, normalized.lib.sizes=TRUE)
  }

  return(counts_norm_CRTH2_CCR6)

}

counts_pc_norm_CRTH2_CCR6 <- normalize_counts(counts_pc_filtered_CRTH2_CCR6, "tmm")
#counts_all_norm <- normalize_counts(counts_all_filtered, "tmm")

#Write out normalized counts
#Add HGNC symbols to gene names
#log 2 transform

counts_out_CRTH2_CCR6 <- log2(counts_pc_norm_CRTH2_CCR6+1)

write.csv(counts_out_CRTH2_CCR6,
file.path(tables_dir,"TMM_normalized_log2_transformed_CRTH2_CCR6_counts.csv"),
```

```
        quote=FALSE, row.names = TRUE)

counts_pc_norm_CRTH2_CCR6 <- as.data.frame(counts_pc_norm_CRTH2_CCR6)
idkey <- data.frame("original" =  c(as.character(gene_key_pc$Ensembl.Gene.ID)),
                "new" = c(as.character(gene_key_pc$HGNC.symbol)))

counts_pc_norm_CRTH2_CCR6 <- counts_pc_norm_CRTH2_CCR6[!
row.names(counts_pc_norm_CRTH2_CCR6) %in% c("ENSG00000280987"),]

rownames(counts_pc_norm_CRTH2_CCR6) <- dplyr::recode(
  rownames(counts_pc_norm_CRTH2_CCR6),
  !!!setNames(as.character(idkey$new), as.character(idkey$original)))

counts_pc_norm_CRTH2_CCR6$HGNC <- rownames(counts_pc_norm_CRTH2_CCR6)



#remove batch effects
#counts_rbe_CRTH2_CCR6 <- removeBatchEffect(log2(counts_pc_norm_CRTH2_CCR6+1),
#                           batch = design_qc_CRTH2_CCR6$project)

#Run PCA on the normalized log2 transformed counts data
pca_CRTH2_CCR6 = prcomp(log2(as.data.frame(t(counts_out_CRTH2_CCR6))+1), center=TRUE,
scale=FALSE)
screeplot(pca_CRTH2_CCR6)

pca_CRTH2_CCR6 = prcomp((as.data.frame(t(counts_pc_norm_CRTH2_CCR6))), center=TRUE,
scale=FALSE)
screeplot(pca_CRTH2_CCR6)

#Get PCA results and merge with sample information stored in metrics
pca_scores_CRTH2_CCR6= as.data.frame(pca_CRTH2_CCR6$x)

pdatscores_CRTH2_CCR6 <- merge(design_qc_CRTH2_CCR6, pca_scores_CRTH2_CCR6, by.x =
"libid", by.y="row.names")

#look at potential correlations
pc_cors_CRTH2_CCR6 <- calc_PCcors(pca_CRTH2_CCR6, design_qc_CRTH2_CCR6, id_col =
"libid") %>%
  t()

#remove highest age (age is 52 in row 78) for PC1vPC2 age graph ***
#pdatscores_age <- pdatscores[-c(55),]

#Make a plot of PC1 vs PC2. Color by
g_PC1vPC2_CRTH2_CCR6 <- ggplot(data = pdatscores_CRTH2_CCR6,
                           aes(x=PC1,
                               y=PC2,
                               color=project)) +
  geom_point()
# + scale_color_viridis()

print(g_PC1vPC2_CRTH2_CCR6)
```

```
#model gene expression, linear modeling for peanut_IgE
design_model_CRTH2_CCR6 <- design_qc_CRTH2_CCR6 %>%
  dplyr::filter(!is.na(cellType))

counts_model_CRTH2_CCR6 <- counts_pc_norm_CRTH2_CCR6[,design_model_CRTH2_CCR6$libid]

design_matrix_CRTH2_CCR6 <- model.matrix(~design_model_CRTH2_CCR6$cellType)

#simplify columns names so contrasts are easier to make later
colnames(design_matrix_CRTH2_CCR6) <- c("Intercept", "cellType")

#voom_design <- voom(counts_model, design= design_matrix, plot=TRUE, span=0.2)

vwts_design_CRTH2_CCR6 <- voomWithQualityWeights(counts_model_CRTH2_CCR6, design=
design_matrix_CRTH2_CCR6, plot=FALSE, span=0.2)

#Check for patterns of weights that vary with group
#design_qc$sampleWeight <-
vwts_design$targets$sample.weights[match(design_qc$libid,rownames(vwts_design$targets))]

#ggplot(design_qc,
#       aes(x = libid,
#           y = sampleWeight,
#           fill = project))+
#  geom_col()

# fit model)
vfit <-
  lmFit(vwts_design_CRTH2_CCR6)

vfit_eB <- eBayes(vfit)

#Get top genes that vary with peanut_IgE
top_genes_CRTH2_CCR6 <- topTable(vfit_eB, coef = "cellType", sort.by = "P", number = Inf)
top_genes_CRTH2_CCR6$gene <- rownames(top_genes_CRTH2_CCR6)



#heatmap of CCR6+
genes_for_hmap <- c("CCR7", "RORC", "CD27", "IL17RB", "IL1RL1", "HPGDS", "CD200", "CCR6",
"FOXP3", "IL31",
                    "GATA3", "IL10", "IL1R1", "IL4", "IL5", "IL9", "IL13", "CD200R1",
                    "DUSP6")

sig_genes <- top_genes_CRTH2_CCR6[top_genes_CRTH2_CCR6$adj.P.Val < 0.01, ]
genes_for_hmap <- rownames(sig_genes)

counts_for_hmap_CRTH2_CCR6 <- log2(counts_pc_norm_CRTH2_CCR6[genes_for_hmap,
design_qc_CRTH2_CCR6$libid]+1)
scaled_counts_CRTH2_CCR6 <- t(scale(t(counts_for_hmap_CRTH2_CCR6)))

design_qc_CRTH2_CCR6$log_peanut_IgE <- log10(design_qc_CRTH2_CCR6$peanut_IgE+1)
design_qc_CRTH2_CCR6$log_peanut_IgG4 <- log10(design_qc_CRTH2_CCR6$peanut_IgG4+1)
design_qc_CRTH2_CCR6$log_CRTH2_CCR6pct <- log10(design_qc_CRTH2_CCR6$CRTH2pct +1)
```

```
project_colors <- c("P354-1" = "darkblue", "P350-2" = "firebrick", "P193-4" = "gold1")

var_range <- range(design_qc_CRTH2_CCR6$log_peanut_IgG4, na.rm = TRUE)
var2_range <- range(design_qc_CRTH2_CCR6$log_peanut_IgE, na.rm = TRUE)
var3_range <- range(log10(design_qc_CRTH2_CCR6$mtd+1), na.rm = TRUE)
var4_range <- range(design_qc_CRTH2_CCR6$spt, na.rm = TRUE)
var5_range <- range(log10(design_qc_CRTH2_CCR6$CD154freq), na.rm = TRUE)
var6_range <- range(design_qc_CRTH2_CCR6$CCR6pct, na.rm = TRUE)
var7_range <- range(design_qc_CRTH2_CCR6$CD27pct, na.rm = TRUE)
var8_range <- range(design_qc_CRTH2_CCR6$CRTH2pct, na.rm = TRUE)
var9_range <- range(design_qc_CRTH2_CCR6$log_CRTH2pct, na.rm = TRUE)

top_anno <- HeatmapAnnotation(Project = design_qc_CRTH2_CCR6$project,
                        Age = design_qc_CRTH2_CCR6$age,
                        #IgG4 = design_qc_CRTH2_CCR6$log_peanut_IgG4,
                        IgE = design_qc_CRTH2_CCR6$log_peanut_IgE,
                        #mtd = log10(design_qc_CRTH2_CCR6$mtd+1),
                        #CD154freq = log10(design_qc_CRTH2_CCR6$CD154freq),
                        #CCR6pct = design_qc_CRTH2_CCR6$CCR6pct,
                        #CD27pct = design_qc_CRTH2_CCR6$CD27pct,
                        #CRTH2pct = design_qc_CRTH2_CCR6$CRTH2pct,
                        CellType = design_qc_CRTH2_CCR6$cellType,
                        #na_col = "black",
                        col = list(Project = project_colors,
                                Age = colorRamp2(c(0,20), c("yellow", "red")),
                                #IgG4 = colorRamp2(c(0, 1), c("yellow", "red")),
                                IgE = colorRamp2(var2_range, c("yellow", "red")),
                                #mtd = colorRamp2(var3_range, c("yellow", "red")),
                                #CD154freq = colorRamp2(var5_range, c("yellow", "red")),
                                #CCR6pct = colorRamp2(var6_range, c("yellow", "red")),
                                #CD27pct = colorRamp2(var7_range, c("yellow", "red")),
                                #CRTH2pct = colorRamp2(var8_range, c("yellow", "red")),
                                CellType = c("CCR6+" = "darkblue", "CRTH2+" = "firebrick1")))

#bottom_anno <- HeatmapAnnotation(CellType = design_qc_CRTH2_CCR6$cellType,
#                        col = list( CellType = c("CCR6+" = "darkblue", "CRTH2+" = "firebrick1")))

Heatmap(scaled_counts_CRTH2_CCR6,
        name = "row z-score",
        top_annotation = top_anno,
        #       bottom_annotation = bottom_anno,
        show_column_names = FALSE,
        show_row_names = TRUE,
        column_title = "2 Group CRTH2 and CCR6 Subsets")
```