Research Assistant Work

JuMP:

Introduction - Read

Read lightly over "Getting started with sets and indexing"

- Sets are often used to represent some variable in a sample with multiple observations

- Unordered sets can be used for representing non-numeric vectors, such as Strings. They are most commonly initialized through a standard vector. Other methods include using a dictionary (each key is associated with some value), and using Julia's Set functionality, which allows you to iterate through elements of a vector/dictionary and assign the result to a new set. This method will also remove duplicate elements

- Numbered sets are used for numeric variables. These are often initialized using ranges to automatically create a vector that includes each element within the range. You can use the syntax [start:step:stop] to initialize a vector that goes from start to stop in steps the size of "step".

- You can also use Tuples as sets through [(tuple\_name)] or iterating through tuple combinations. For multi-dimensional sets, you can construct containers using x[tuple\_1,tuple\_2], where tuple\_1 is the first dimension and tuple\_2 is the second dimension.

- Set operations include setdiff(x,y) which returns all elements that are not common between x and y. Another operation is union(x,y) which combines the elements of x and y vectors. Length(x) returns the number of elements in a vector.

- You can test for set membership using the "in" keyword, and combine with "for" and "if" keywords in expressions to filter and create new sets.

- For compound sets, where each element includes a function(s) of other elements, such as distance between cities, we can use a matrix to represent the set. We can set model objectives using some function of the variables. We can also use containers as indices by converting tuples to index values.

- Data conversion to a more intuitive form is useful.

Tutorial - Linear Programming

- Some (linear) optimization function according to linear constraints.

- Mixed-integer Linear Programs (MILPs) enforce some level of integrality where variables take discrete values.

- Knapsack problem: Maximize some value (c) associated with each state of i (objective function can be considered nutritional value, e.g.) according to some constraint (cost) on the number of i's that can be used through the variable w (cost per item). x is an indicator function for whether each i is activated (purchased). Can be written as a function of the cost and value vectors multiplied by a transposed "x" vector, with the summed elements of the resulting vectors corresponding to the objective and constraint functions. x is to be determined by the model here as the optimal set of decisions x for each i.

- Syntax: @variable to determine decision variables, @constraint to determine constraint function (cost), @objective to determine objective function. Use function optimize!(model) to solve program and solution\_summary(model) to summarize the results. @assert sets important variables for the model to run. Bin means a binary variable.

- Once you have created your model, it is a good idea to build a function around it that takes vectors/variables as inputs and computes the results, returning the decision variables outputted by the solver.

Time: 4 Hours

Week 2: