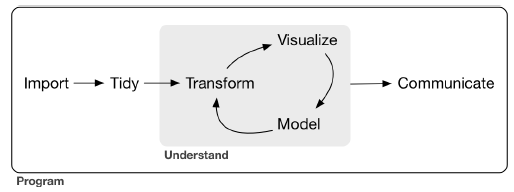
***Book: Hardley wickham R for data science. Import tidy transform visualize and model data***

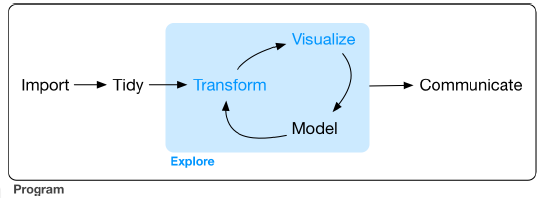


Tidying your data means storing it in a consistent form that matches the semantics of the dataset with the way it is stored.

Together, tidying and transforming are called wrangling, because getting your data in a form that’s natural to work with often feels like a fight!

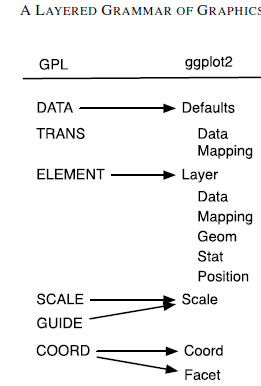
The tidyverse is a coherent system of packages for data manipulation, exploration and visualization that share a common design philosophy

Part 1 explore



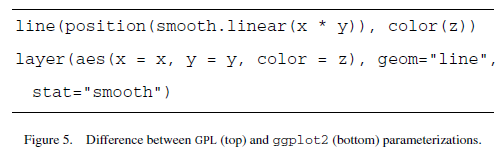
***Chapter 1 : Data visualization with ggplot2***

A layered grammar of graphics



The layer component is particularly important as it determines the physical representation of the data, with the combination of stat and geom defining many familiar named graphics: the scatterplot, histogram, contourplot, and so on. In practice, many plots have (at least) three layers: the data, context for the data, and a statistical summary of the data.

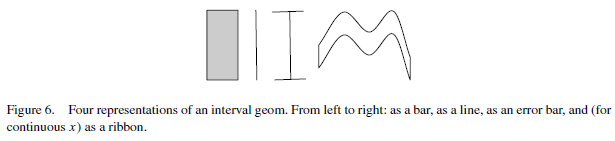
Layers



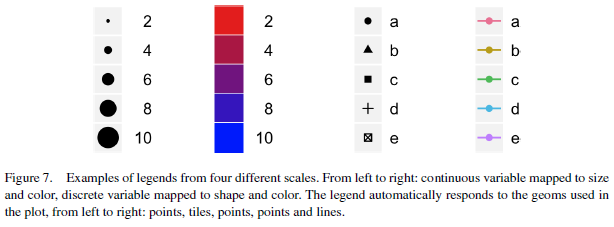
* Data and aesthetic mapping
* A statistical transformation (stat) 🡪 summarizing in some manner
* A geometric object (geom) 🡪 type of plot that you create
* A position adjustment

Classify geoms by their dimensionality:

* 0d: point, text
* 1d: path, line
* 2d: polygon, interval

They can be rendered in different ways 🡪 

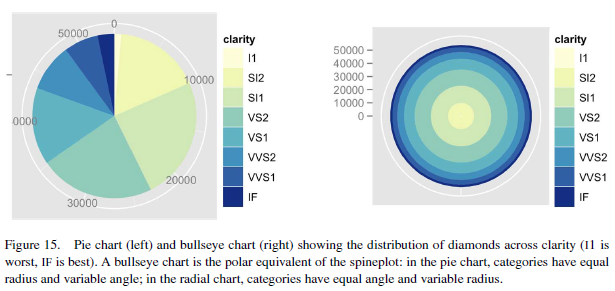
Scales



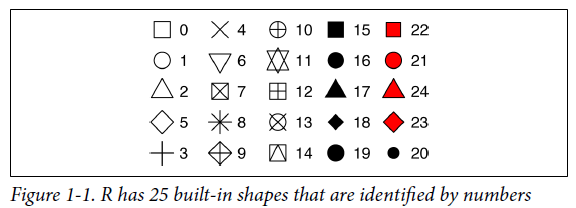
Faceting makes it easy to create small multiples of different subsets of an entire dataset. This is a powerful tool when investigating whether patterns are the same or different across conditions. The faceting specification describes which variables should be used to split up the data, and how they should be arranged.

Polar coordinates

For figure 15 🡪 ggplot(diamonds,aes(x = "", fill=clarity)) + geom\_bar(width = 1) + coord\_polar (theta="y"),



Shapes points



Facets are very useful for categorical variables

A geom is the geometrical object that a plot uses to represent data.

The algorithm used to calculate new values for a graph is called a stat.

***Chapter 3: Data transformation with dplyr***

Tibbles are data frames, but slightly tweaked to work better in the tidyverse

• int stands for integers.

• dbl stands for doubles, or real numbers.

• chr stands for character vectors, or strings.

• dttm stands for date-times (a date + a time).

Dplyr basics:

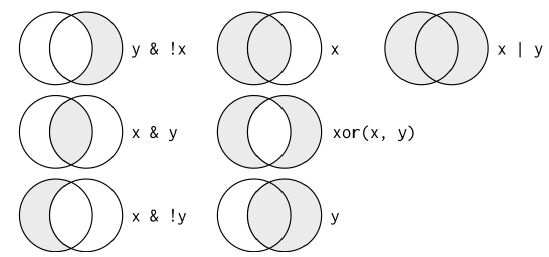
• Pick observations by their values (filter()).

• Reorder the rows (arrange()).

• Pick variables by their names (select()).

• Create new variables with functions of existing variables (mutate()).

• Collapse many values down to a single summary (summarize()).



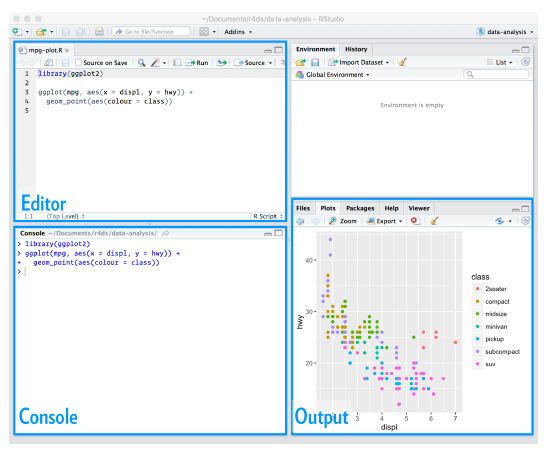
• starts\_with("abc") matches names that begin with “abc”.

• ends\_with("xyz") matches names that end with “xyz”.

• contains("ijk") matches names that contain “ijk”.

• matches("(.)\\1") selects variables that match a regular expression. This one matches any variables that contain repeated characters.

• num\_range("x", 1:3) matches x1, x2, and x3.



NAs (“not availables”).

***Chapter 5 exploratory data analysis***

Exploratory data analysis, or EDA for short.

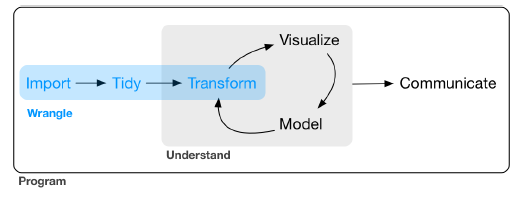
Variation is the tendency of the values of a variable to change from measurement to measurement.

To examine the distribution of a categorical variable, use a bar chart. To examine the distribution of a continuous variable, use a histogram.

If you wish to overlay multiple histograms in the same plot, I recommend using geom\_freqpoly() instead of geom\_histogram().

If variation describes the behavior within a variable, covariation describes the behavior between variables. Covariation is the tendency for the values of two or more variables to vary together in a related way.

***Part 2 wrangle***



***Chapter 7 tibbles with tibble***

Tibbles are data frames, but they tweak some older behaviors to make life a little easier.

If you’re already familiar with data.frame(), note that tibble() does much less: it never changes the type of the inputs (e.g., it never converts strings to factors!), it never changes the names of variables, and it never creates row names.

It’s possible for a tibble to have column names that are not valid R variable names, aka nonsyntactic names.

***Chapter 9 tidy data with tidyr***

In this chapter, you will learn a consistent way to organize your data in R, an organization called tidy data.

There are three interrelated rules which make a dataset tidy:

1. Each variable must have its own column.

2. Each observation must have its own row.

3. Each value must have its own cell.

Surprisingly, a value can be missing in one of two possible ways:

• Explicitly, i.e., flagged with NA.

• Implicitly, i.e., simply not present in the data.

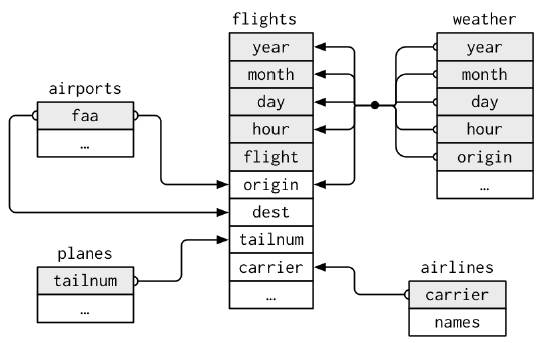
***Chapter 10 relational data with dplyr***

• Mutating joins, which add new variables to one data frame from matching observations in another.

• Filtering joins, which filter observations from one data frame based on whether or not they match an observation in the other table.

• Set operations, which treat observations as if they were set elements.

Relational database management system (or RDBMS), SQL



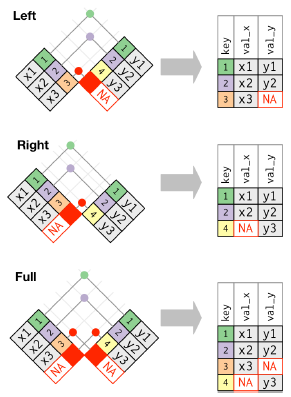
The variables used to connect each pair of tables are called keys. A key is a variable (or set of variables) that uniquely identifies an observation.

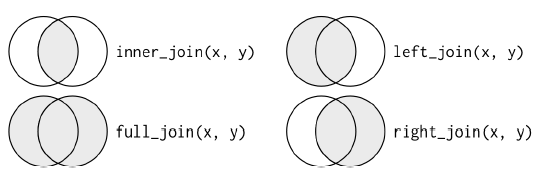
There are two types of keys:

• A primary key uniquely identifies an observation in its own table. For example, planes$tailnum is a primary key because it uniquely identifies each plane in the planes table.

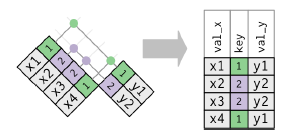
• A foreign key uniquely identifies an observation in another table. For example, flights$tailnum is a foreign key because it appears in the flights table where it matches each flight to a unique plane.

Once you’ve identified the primary keys in your tables, it’s good practice to verify that they do indeed uniquely identify each observation. One way to do that is to count() the primary keys and look for entries where n is greater than one

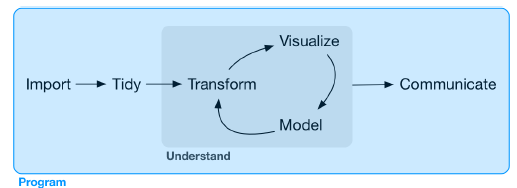




Duplicated keys



***Part 3 program***

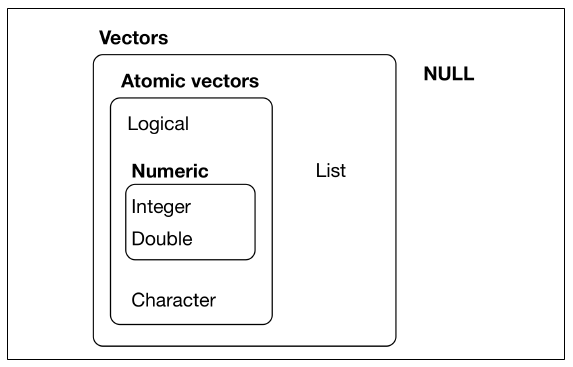


***Chapter 16 vectors***

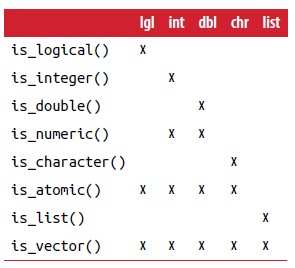
There are two types of vectors:

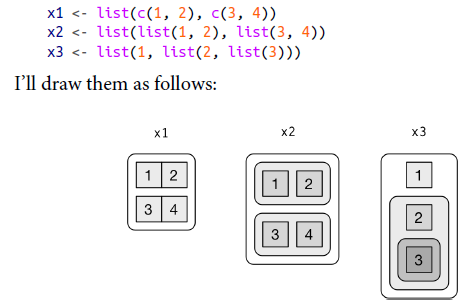
• Atomic vectors, of which there are six types: logical, integer, double, character, complex, and raw. Integer and double vectors are collectively known as numeric vectors.

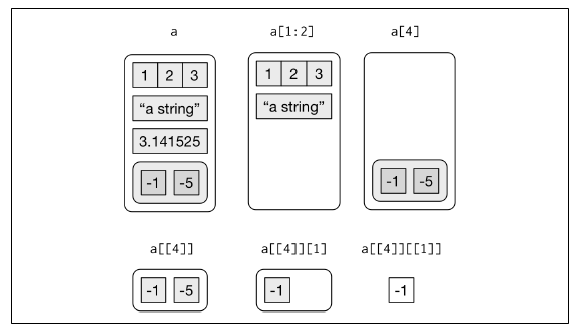
• Lists, which are sometimes called recursive vectors because lists can contain other lists.



The chief difference between atomic vectors and lists is that atomic vectors are homogeneous, while lists can be heterogeneous.







***Part IV model***

If you are serious about doing a confirmatory analysis, one approach is to split your data into three pieces before you begin the analysis:

• 60% of your data goes into a training (or exploration) set. You’re allowed to do anything you like with this data: visualize it and fit tons of models to it.

• 20% goes into a query set. You can use this data to compare models or visualizations by hand, but you’re not allowed to use it as part of an automated process.

• 20% is held back for a test set. You can only use this data ONCE, to test your final model.

The goal of a model is to provide a simple low-dimensional summary of a dataset.

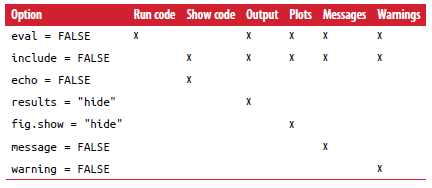
The predictions tell you the pattern that the model has captured, and the residuals tell you what the model has missed. That the average of the residual will always be 0.

***Part V communicate***

R Markdown Cheat Sheet: available in the RStudio IDE under Help → Cheatsheets → R Markdown Cheat Sheet.

<http://rstudio.com/cheatsheets>.





Ggplot page 441…..