{:title “Clojure - Basic Incanter” :layout :post :date “2017-1-29” :tags [“Clojure” “Code” “Guide”]}

Incanter is similiar to R which has many uses for statistical computing. Incanter is a clojure based version of R which runs on the JVM. It has many useful functions to help aid in making graphs and analyze data. We'll only be looking on making graphs in this post but in a later post we'll look at analyzing data using incanter. For now lets make some graphs! Go ahead make a new clojure project with leiningen. Add the following to your project.clj to include incanter into your dependencies.

```
:dependencies [[org.clojure/clojure "1.8.0"]
               [incanter "1.5.5"]]
```

Go ahead and run lein deps so that leiningen can get incanter as a dependency.

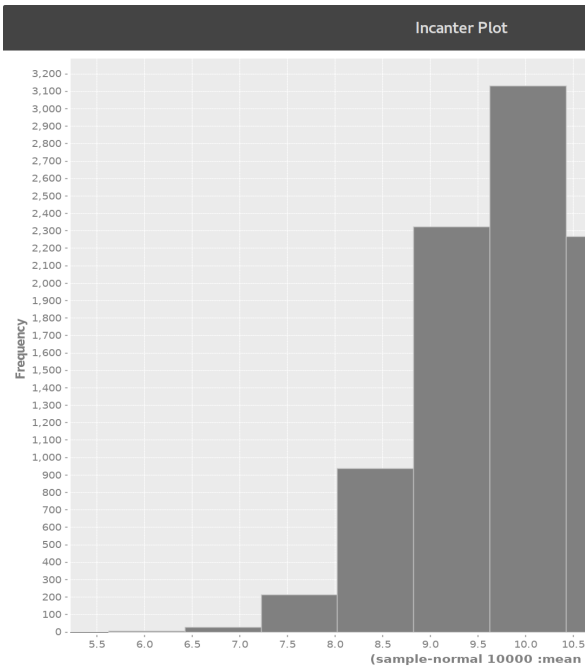
```
lein deps
```

After that we are ready to go and start editing the core.clj and first examining how to create a neat little histogram. So lets use the things we are going to need from incanter make your namespace as the following just replace the namespace to whatever project you created in my case I did lein new incantertut but yours will be different so rechange that part of what you did.

```
(ns incantertut.core
  (:use [incanter.charts :only [histogram scatter-plot pie-chart]]
        [incanter.core :only [view]]
        [incanter.stats :only [sample-normal]]))
```

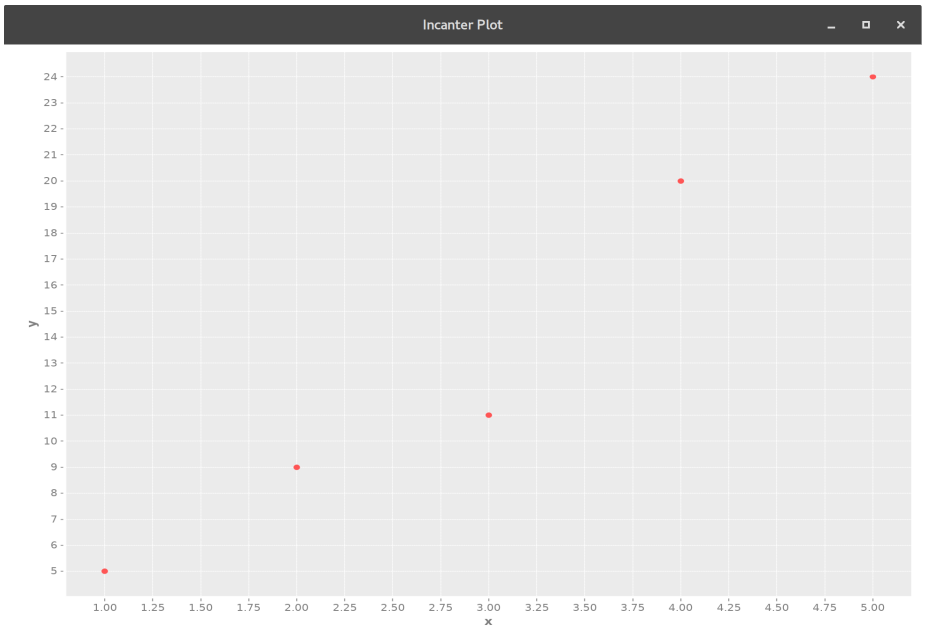
In this post we'll only be looking at histograms, scatter-plots and pie-charts. Next part is fun because it is literally one line to generate and view the graph and behold.

```
(view (histogram (sample-normal 10000 :mean 10)) :width 700 :height 700)
```



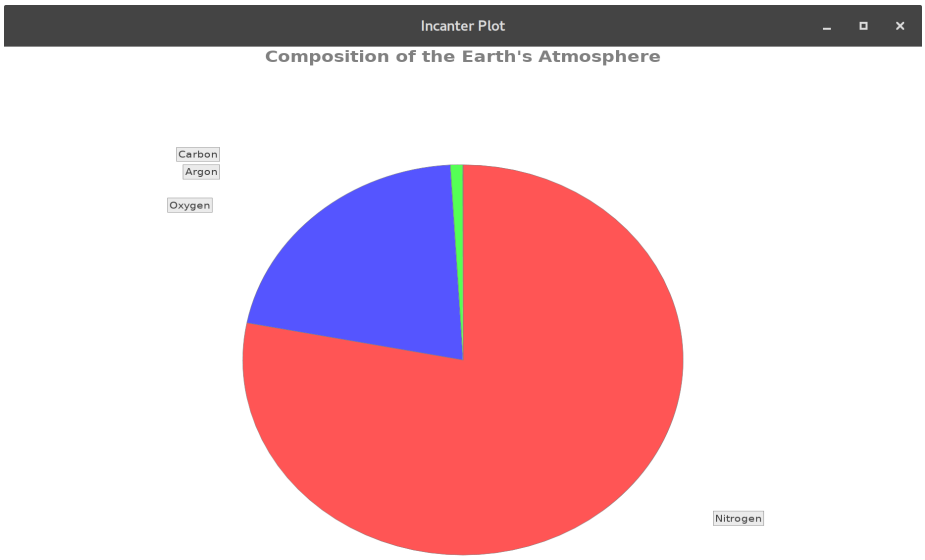
After running that you should see the following graph pop up.
The above one-liner creates a histogram. The sample-normal function creates a normal gaussian distribution with 10,000 points. You can select the mean you want for the distribution in this case it was set to 10 by default it is 1. You can also set the :sd which is the standard deviation. The view function takes other parameters such as a width and height so you can play around with those changing the starting height/width of the histogram. Note that the view function can do more than just graphs. It can view urls and images as well! Next lets see how to make a scatter-plot which is pretty much a one liner as well and exactly the same as making a histogram expect we use scatter-plot instead and define some variables before hand.

```
(def x [1 2 3 4 5])
(def y [5 9 11 20 24])

(view (scatter-plot x y))
```



x and y defines the data for the x axis and the y variable contains the data for the y axis. Note that the x and y variables are of type persistentVector. Notice that the data set is not perfectly linear but it looks like you could draw a line of best fit or a regression line which I'll go over on how to do that in clojure later. As well as find values such as the R^2 value telling us how well the data fits that line of best fit. Now for the file graph we'll make a pie graph. Piechart that we are about to make will include the elements making up earth's atmosphere and for that we'll define two variables called elements and percentages. Both persistent vectors however elements will consist of strings that will contain the name of the elements and percentages will contain the percentage of that element in the atmosphere. The main function here to create the pie-chart is called pie-chart which will be taking two arguments the elements and the percentages as well as you can add an additional parameter adding a :title as a string value such as "Composition of the Earth's Atmosphere".

```
(def elements ["Nitrogen" "Oxygen" "Argon" "Carbon"])
(def percentages [78 21 0.9 0.03])

(view (pie-chart elements percentages :title "Composition of the Earth's Atmosphere"))
```



There are more variations of graphs avaliable in incanter but I just went through some of the more basic ones that people tend to use so there you go now you can make histograms, scatter-plots, and pie charts. All in a few lines of clojure code.