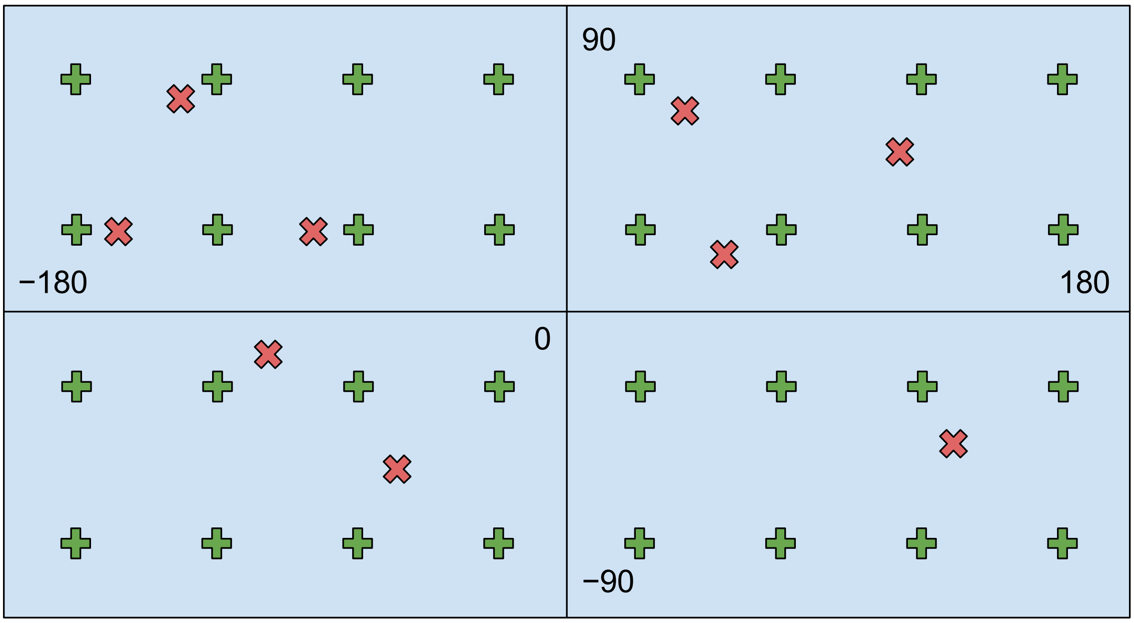
Milestone overview

One of the primary goals of this project is to be able to visualize the evolution of the climate. If you tried to visualize the temperatures of different years in the previous milestone, you might have noticed that it is actually quite hard to really measure how the temperatures have evolved since 1975.

That's why we propose to visualize the *deviations* of the temperatures over the years, rather than just the temperatures themselves. The goal of this milestone is to compute such deviations. You will have to complete the file **Manipulation.scala**.

Computing deviations means comparing a value to a previous value, which serves as a reference, or a "normal" temperature. You will first compute the average temperatures between 1975 and 1990, all over the world. This will constitute your reference temperatures. You will then compare the yearly average temperatures, for each year between 1991 and 2015, to the normals.

In order to make things faster, you will first spatially interpolate your scattered data into a regular grid:



The above figure illustrates the grid points (in green) and the actual data points (in red). You will have to guess the temperature at the green locations based on the known temperatures at the red locations.

Once you will have such a grid for each year, you will easily be able to compute average (coordinate wise) over years and deviations.

You can monitor your progress by submitting your work at any time during the development of this milestone. Your submission token and the list of your graded submissions is available on this page.

Grid generation

You will have to Implement the following method:

```
def makeGrid(temperatures: Iterable[(Location, Double)]): (Int, Int) => Double
```

It takes as parameter the temperatures associated with their location and returns the corresponding grid. You are free to internally represent the grid as you want (e.g. using a class Grid), but to interoperate with the grading system you will have to convert it to a function of type (Int, Int) => Double. This function is expected to return the temperature at the given latitude and longitude parameters. The latitude can be any integer between -89 and 90, and the longitude can be any integer between -180 and 179. Note that the order of the parameters matters: the first one is the latitude and the second one is the longitude. The top-left corner has coordinates (90, -180), and the bottom-right corner has coordinates (-89, 179).

Average and deviation computation

You will have to implement the following two methods:

```
def average(temperaturess: Iterable[Iterable[(Location, Double)]]): (Int, Int) => Double
```

This method takes a sequence of temperature data over several years (each “temperature data” for one year being a sequence of pairs of temperature and location), and returns a grid containing the average temperature at each location.

```
def deviation(
  temperatures: Iterable[(Location, Double)],
  normals: (Int, Int) => Double
): (Int, Int) => Double
```

This method takes temperature data and a grid containing normal temperatures, and returns a grid containing temperature deviations compared to normals.

Mark as completed