# BIOS560R - Function Case Study

Pittard
Biostatistics and Bioinformatics

October 16, 2013

## 1 Compute the median of a vector

Computing the median of a vector involves finding the “middle” value in a vector. To do this you have to first determine if a vector is of even length or odd length. Based on that answer you will need to apply a different formula. As an example consider the vector below:

```
> exampodd =   c(3,6,9,1,10)

> sort(exampodd)
[1]  1  3  6  9 10
```

So what element is in the “middle” ? The third element whose value is 6. How do you arrive at this number ? Simple. Divide the length of the vector in half and use one of the numeric functions (round or ceiling) to get the middle element number and then use that to index into the vector. So in our case we have “3” as the middle element so exampo[3] is equal to 6. Does this match what the built in R function returns ?

```
> median(exampodd)  # Yes it does
[1] 6
```

But what about the case wherein the length of the vector is even ? How do we find its median ?

```
> exampeven = c(11,9,4,7)
> exampeven
[1] 11  9  4  7
> sort(exampeven)
[1]  4  7  9 11
> median(exampeven)
[1] 8
```

Here we sort the vector, then divide its length in half to find the “middle two values” after which we take their average. As before we’ll need to use one or more of the numeric functions to access the correct elements. Here, the middle two elements are 7 and 9 so we take their mean to get a value of 8.

## 2 Write a function

Write a function called mymedian that accepts a vector of general length and computes its median using the above logic. Your function will first determine if the length of the vector is odd or even and will handle the cases accordingly as described above within each branch of your if statement. Your function will return a single value. In addition to the example vectors above here are some test data so you can determine if your function is working correctly. Here is a big hint. Your “psuedo-code” will like this:

```
if the length of the vector is odd
   sort the vector
   divide its length by 2 to find the precise middle element number index
     (use numeric functions to help)
   use this to index into the sorted vector to get the value
else
   sort the vector
   divide its length by 2 to find the two middle element numbers
   use these to index into the sorted vector to get the values and take
    their mean
```

## 3 Let’s write the first part of the function

So let’s deal with the first branch where the vector length is odd. How do we sort the vector ? Easy. use the sort command.

```
> exampodd =   c(3,6,9,1,10)
> mys = sort(exampodd)
> mys
[1]  1  3  6  9 10
>
```

So how do we find the “middle” value in this vector ? Well, one way might be to divide its length by two which should give us something close to the middle element.

```
> index = length(mys)/2
> index
[1] 2.5
```

Well that’s close. I mean we already know that the middle element is the third element of the sorted vector. So how do we coax 2.5 to be 3 ?

```
> round(index)  # Doesn't work. It rounds down
[1] 2

> ceiling(index)  # Aha !
[1] 3
```

So here we have a working approach. To find the median value we could do:

```
> mys = sort(exampodd)
> index = ceiling(length(mys)/2)
> mymedian = mys[index]
> mymedian
[1] 6
```

So let's embed this logic into a function:

```
mymedian <- function(somevec) {

# Input: A vector of some length
# Output: A single value representing the median

   if (length(somevec) %% 2 != 0) {  # If the length is odd
      mys = sort(somevec)
      index = ceiling(length(mys)/2)
      mymedian = mys[index]

   } else {     # The length of the input vector is even

     # Put logic here to handle even length vectors

   }
   return(mymedian)
}

> mymedian(exampodd)
[1] 6
```