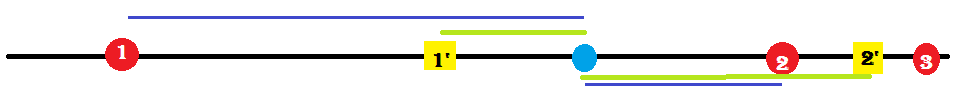
If there are two **sorted** arrays: a (length n) and b (length m), how do we get the index of b such that the element of b with i-th index is most closed to to the i-th element in a, i.e.

Let's consider a real line with several numbers like this:



The problem is get the index of red ball that is most closed to the blue ball in the sense of L2 distance. But indeed, , hence we could average (take mean of) the adjacent red balls to do the same thing, i.e. 1’ is the average of 1 and 2, 2’ is the average of 2 and 3. We call the i-th yellow cubic the represent element of (i,i+1) red ball. Then the final result is number of represent element before the blue ball adds a shift 1. In R the code is:

# author: yifan yang yifan.yang AT UKY DoT EdU

# license: GPL 3

# O((n+m)log(n+m))

MeanMatch **<-** **function(**x,y**){**

y2 **=** c**((**y**[-**1**]+**y**[-**length**(**y**)])/**2**)** # KEY, **x y should be sorted**

full **=** cbind**(**

c**(**x,y2**)**, # combined

c**(**

rep**(**0,length**(**x**))**, # the lable must be 0/1

rep**(**1,length**(**y2**))**

**))**

full**=**full**[**order**(**full**[**,1**])**,**]** # data.table has another neat way to do this

cumsum**(**full**[**,2**])** **->** inx

inx**[**which**(**full**[**,2**]==**0**)]** **+**1 **->** result

return**(**result**)**;

**}**

## simple example

x**=**sort**(**abs**(**rnorm**(**10**)))**

y**=**sort**(**runif**(**5**))**

u **=** y

## A plain for-loop version O(n\*m)

**for** **(**ii **in** 1**:**length**(**x**))** which.min**(**abs**(**x**[**ii**]-**y**))** **->** u**[**ii**]**

MeanMatch**(**x,y**)** **->** u2

plot**(**0,xlab**=**'',ylab**=**'',xlim**=**range**(**c**(**x,y**))**,ylim**=**c**(**0,.5**)**,type**=**'n',xaxt**=**'n',yaxt**=**'n'**)**

text**(** x**=**x, y **=**.13, label**=**1**:**length**(**x**)**, col**=**'cornflowerblue'**)**

points**(**x**=**x,y**=**rep**(**.1,length**(**x**))**,col**=**'cornflowerblue',pch**=**'+',cex**=**2**)**

points**(**x**=**y,y**=**rep**(**1,length**(**y**))**,col**=**'chocolate2',cex**=**2,type**=**'h'**)**

text**(** x**=**y, y **=**.25, label**=**1**:**length**(**y**)**, col**=**2**)**

abline**(**h**=**0.1**)**

sum**(**abs**(**u**-**u2**))**

