SortTimes

Samyak Ahuja

August 3, 2018

Complexity for different Sorting Algorithms.

Helper Functions

Replicator

```
replicator <- function(func){
  ele <- seq(from = 0, to = 10000, by = 250)
  ele <- ele[-1]
  timeElapsed <- c()
  for(n in ele){
    timeElapsed <- c(timeElapsed, system.time(replicate(10, func(sample(x = 1:100, size = n, replace = 
  }
  return (data.frame(timeElapsed,ele))
}
```

Plotter

```
plotter <- function(df){
  ggplot(df, aes(timeElapsed, ele, color = timeElapsed)) +
    geom_point(shape = 16, size = 5, show.legend = FALSE, alpha = 0.6) +
    stat_smooth(method="loess", formula=y~x) +
    theme_minimal() +
    scale_color_gradient(low = "#32aeff", high = "#f2aeff")
}
```

Insertion Sort

Sorting Algorithm

```
insertionSort <- function(vec){
  n <- length(vec)
  for(i in 2:n){
    val <- vec[i]
    pos <- which.max(vec[1:i] > val) #returns index of first occurence of TRUE
    if(pos == 1){
      if(val < vec[1]){
        vec <- c(val, vec[-i])
      }
    }
    else{
      vec <- vec[-i]
```

```
      vec <- c(vec[1:(pos-1)], val, vec[pos:(n-1)])
    }
  }
  return (vec)
}
```

Proof of concept

```
insertionSort(c(1,2,99,-21,2,23,1))
```

```
## [1] -21   1   1   2   2  23  99
```

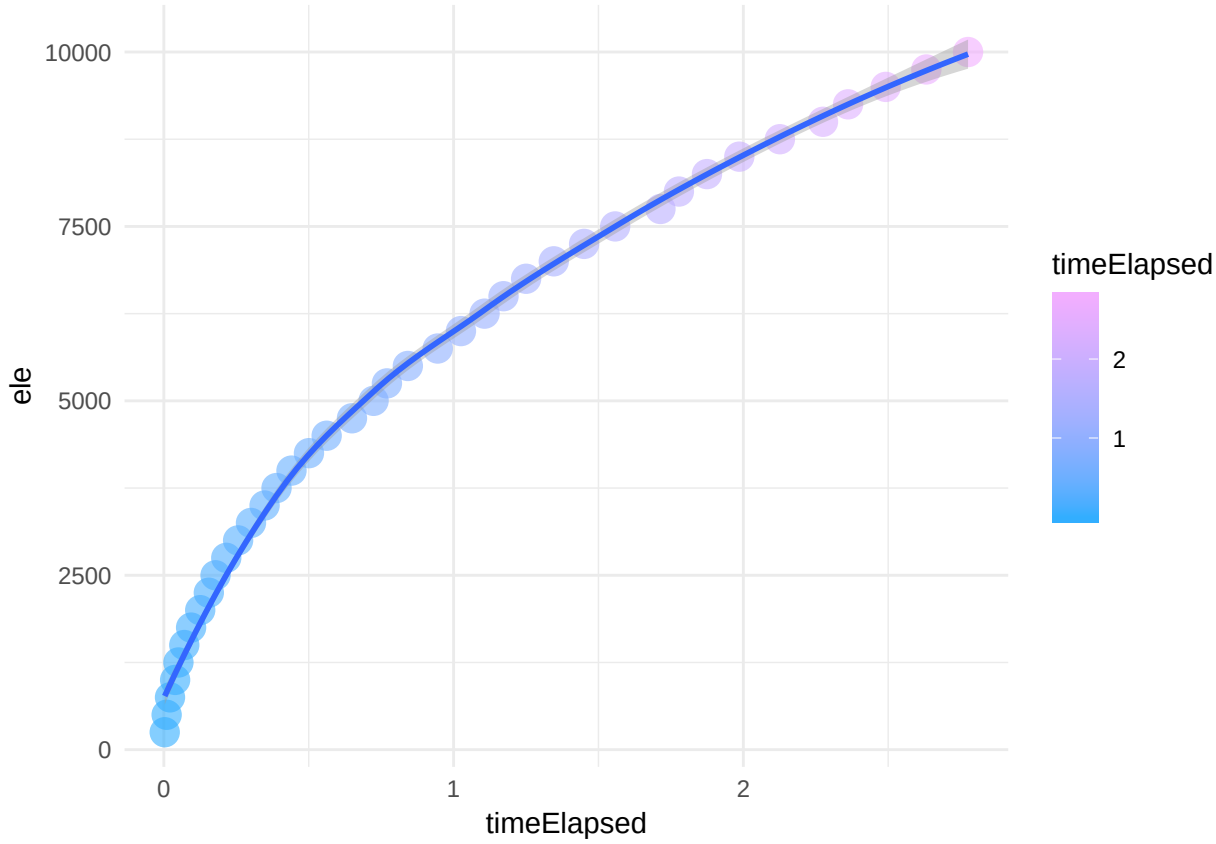
RunTime and Plot

```
isdf <- replicator(insertionSort)
isdf
```

```
##    timeElapsed   ele
## 1       0.0030   250
## 2       0.0096   500
## 3       0.0212   750
## 4       0.0389  1000
## 5       0.0503  1250
## 6       0.0705  1500
## 7       0.0940  1750
## 8       0.1255  2000
## 9       0.1552  2250
## 10      0.1784  2500
## 11      0.2154  2750
## 12      0.2563  3000
## 13      0.3007  3250
## 14      0.3477  3500
## 15      0.3889  3750
## 16      0.4405  4000
## 17      0.5008  4250
## 18      0.5619  4500
## 19      0.6493  4750
## 20      0.7238  5000
## 21      0.7699  5250
## 22      0.8422  5500
## 23      0.9451  5750
## 24      1.0254  6000
## 25      1.1069  6250
## 26      1.1723  6500
## 27      1.2503  6750
## 28      1.3458  7000
## 29      1.4505  7250
## 30      1.5577  7500
## 31      1.7140  7750
## 32      1.7777  8000
## 33      1.8746  8250
## 34      1.9861  8500
```

```
## 35      2.1262  8750
## 36      2.2749  9000
## 37      2.3615  9250
## 38      2.4914  9500
## 39      2.6317  9750
## 40      2.7756 10000
```

```r
plotter(isdf)
```



Merge Sort

Sorting Algorithm

```r
mergeSort <- function(vec){

  mergeTwo <- function(left,right){
    res <- c()
    while(length(left) > 0 && length(right) > 0){
      if(left[1] <= right[1]){
        res <- c(res,left[1])
        left <- left[-1]
      }else{
        res <- c(res,right[1])
        right <- right[-1]
      }
    }
```

```
    if(length(left) > 0) res <- c(res,left)
    if(length(right) > 0) res <- c(res,right)
    return (res)
  }

  n <- length(vec)
  if(n <= 1) return (vec)
  else{
    middle <- length(vec) / 2
    left <- vec[1:floor(middle)]
    right <- vec[floor(middle + 1):n]
    left <- mergeSort(left)
    right <- mergeSort(right)
    if(left[length(left)] <= right[1]){
      return (c(left,right))
    }else{
      return (mergeTwo(left,right))
    }
  }
}
```

Proof of Concept

```
mergeSort(c(12,-22,13,2,-33,2))
```

```
## [1] -33 -22   2   2  12  13
```

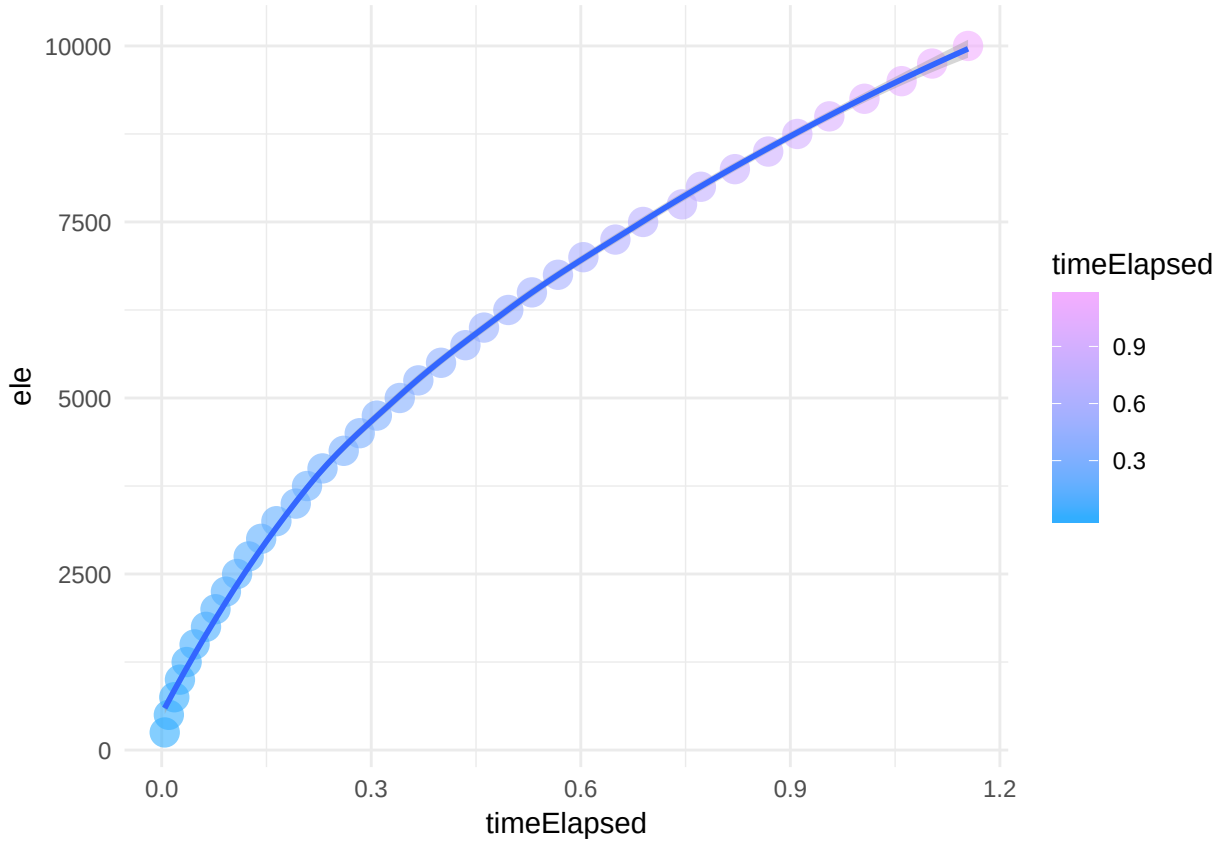
RunTime and Plot

```
msdf <- replicator(mergeSort)
msdf
```

```
##    timeElapsed   ele
## 1       0.0043   250
## 2       0.0102   500
## 3       0.0180   750
## 4       0.0261  1000
## 5       0.0357  1250
## 6       0.0471  1500
## 7       0.0634  1750
## 8       0.0770  2000
## 9       0.0919  2250
## 10      0.1079  2500
## 11      0.1245  2750
## 12      0.1424  3000
## 13      0.1642  3250
## 14      0.1919  3500
## 15      0.2080  3750
## 16      0.2302  4000
## 17      0.2606  4250
## 18      0.2835  4500
## 19      0.3080  4750
```

```
## 20      0.3408  5000
## 21      0.3674  5250
## 22      0.3999  5500
## 23      0.4348  5750
## 24      0.4614  6000
## 25      0.4962  6250
## 26      0.5300  6500
## 27      0.5673  6750
## 28      0.6037  7000
## 29      0.6495  7250
## 30      0.6892  7500
## 31      0.7449  7750
## 32      0.7719  8000
## 33      0.8206  8250
## 34      0.8683  8500
## 35      0.9100  8750
## 36      0.9558  9000
## 37      1.0061  9250
## 38      1.0593  9500
## 39      1.1030  9750
## 40      1.1544 10000
```

```
plotter(msdf)
```



Quick Sort

Sorting Algorithm

```
quickSort <- function(vec){
  if(length(vec) > 1){
    pivot <- median(vec)
    return (c(quickSort(vec[vec < pivot]), vec[vec == pivot], quickSort(vec[vec > pivot])))
  }else{
    return (vec)
  }
}
```

Proof of Concept

```
quickSort(c(12,-22,13,2,-33,2))
```

```
## [1] -33 -22   2   2  12  13
```

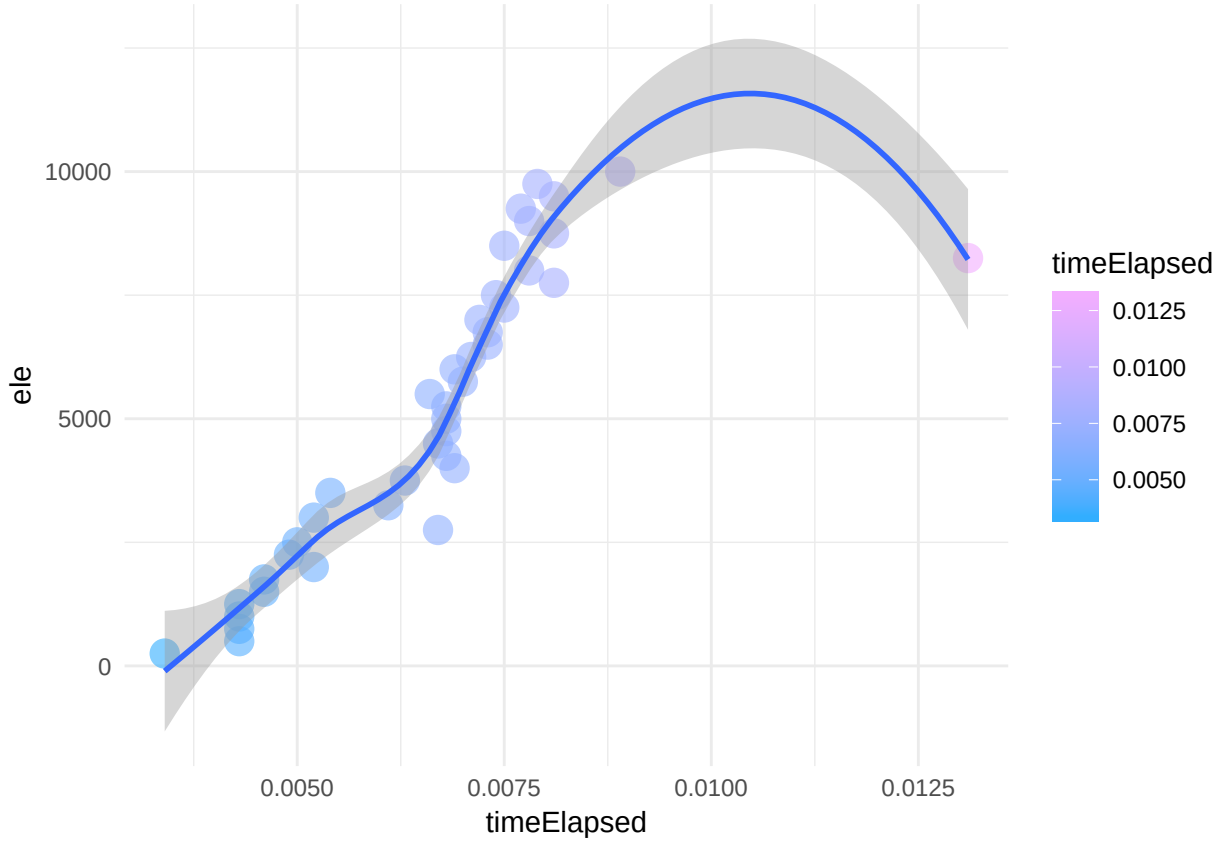
RunTime and Plot

```
qsdf <- replicator(quickSort)
qsdf
```

```
##    timeElapsed   ele
## 1       0.0034   250
## 2       0.0043   500
## 3       0.0043   750
## 4       0.0043  1000
## 5       0.0043  1250
## 6       0.0046  1500
## 7       0.0046  1750
## 8       0.0052  2000
## 9       0.0049  2250
## 10      0.0050  2500
## 11      0.0067  2750
## 12      0.0052  3000
## 13      0.0061  3250
## 14      0.0054  3500
## 15      0.0063  3750
## 16      0.0069  4000
## 17      0.0068  4250
## 18      0.0067  4500
## 19      0.0068  4750
## 20      0.0068  5000
## 21      0.0068  5250
## 22      0.0066  5500
## 23      0.0070  5750
## 24      0.0069  6000
## 25      0.0071  6250
## 26      0.0073  6500
```

```
## 27      0.0073  6750
## 28      0.0072  7000
## 29      0.0075  7250
## 30      0.0074  7500
## 31      0.0081  7750
## 32      0.0078  8000
## 33      0.0131  8250
## 34      0.0075  8500
## 35      0.0081  8750
## 36      0.0078  9000
## 37      0.0077  9250
## 38      0.0081  9500
## 39      0.0079  9750
## 40      0.0089 10000
```

```
plotter(qsdf)
```



Combined Plots

```
df <- data.frame(is = isdf[[1]], ms = msdf[[1]], qs = qsdf[[1]], ele = msdf[[2]])
df <- melt(df, id.vars = "ele")
ggplot(df, aes(value, ele, col = variable)) +
  geom_point(shape = 16, size = 2, alpha = 0.6) +
  stat_smooth(method="loess", formula=y~x) +
  stat_poly_eq(parse=T, aes(label = ..eq.label..), formula=y~x) +
  labs(subtitle = "Size vs Time",
```

```
       y = "Number of Elements",
       x = "Time (in seconds)",
       title = "Combined Scatter Plot")
```