StatR 101: Fall 2012
Homework 5 - Solutions
Eli Gurarie

1. **Summarizing summary statistics**

(a) *Create a function called* ***`SummaryStatsA(x,y)`*** *which takes two vectors and computes the following summary statistics: sample mean and standard deviation of x and y, correlation coefficient r, intercept and slope parameters a and b, the three sums of squares (SS_{total}, SS_{model} and $SS_{residual}$), and the coefficient of determination r^2.*

```
SummaryStatsA <- function(x, y){
  n <- length(x)
  x.bar <- sum(x)/n
  y.bar <- sum(y)/n
  x.sd <- sqrt(sum((x-x.bar)^2)/(n-1))
  y.sd <- sqrt(sum((y-y.bar)^2)/(n-1))
  r <- sum((x-x.bar)*(y-y.bar))/x.sd/y.sd/(n-1)
  b.hat <- r*y.sd/x.sd
  a.hat <- y.bar - b.hat*x.bar

  residual <- y - (a.hat + b.hat*x)
  SStotal <- sum( (y-y.bar)^2 )
  SSerror <- sum( residual^2 )
  SSmodel <- SStotal-SSerror
  r2 <- SSmodel/SStotal

  return(list(x.bar = x.bar, y.bar = y.bar, x.sd = x.sd, y.sd = y.sd,
           r = r, a = a, b = b,
           SStotal = SStotal, SSerror = SSerror, SSmodel = SSmodel,
           r2 = r2))
}
```

(b) *Test the output of this function by plotting and drawing a regression line against any paired set of data you like.*

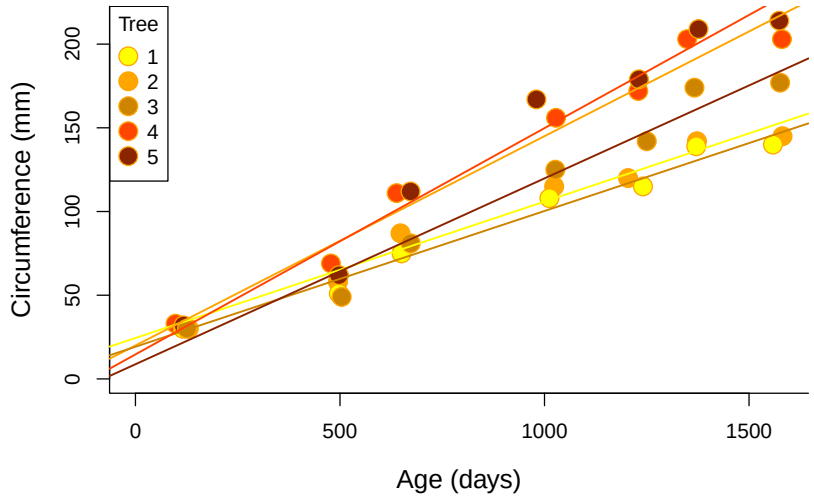
I chose to look at the `Orange` data in the R base package, which measures circumference of an orange tree against age. The data were collected on five trees. The code is below:

```
oranges <- c("yellow", "orange", "orange3", "orangered1", "orangered4")
plot(jitter(Orange$age), Orange$circumference, main="Orange Tree Growth",
    xlab="Age (days)", ylab="Circumference (mm)",
    ylim=c(0,max(Orange$circum)), xlim=c(0,max(Orange$age)),
```

```
    bg = oranges[Orange$Tree], pch=21, col="orange", cex=2)
legend("topleft", title="Tree", legend=1:5,
       pt.bg=oranges, col="orange", pch=21, pt.cex=2)

T1.stats <- SummaryStatsA(Orange$age[Orange$Tree == 1], Orange$circum[Orange$Tree == 1])
T2.stats <- SummaryStatsA(Orange$age[Orange$Tree == 2], Orange$circum[Orange$Tree == 2])
T3.stats <- SummaryStatsA(Orange$age[Orange$Tree == 3], Orange$circum[Orange$Tree == 3])
T4.stats <- SummaryStatsA(Orange$age[Orange$Tree == 4], Orange$circum[Orange$Tree == 4])
T5.stats <- SummaryStatsA(Orange$age[Orange$Tree == 5], Orange$circum[Orange$Tree == 5])

abline(T1.stats$a, T1.stats$b, col=oranges[1], lwd=1.5)
abline(T2.stats$a, T2.stats$b, col=oranges[2], lwd=1.5)
abline(T3.stats$a, T3.stats$b, col=oranges[3], lwd=1.5)
abline(T4.stats$a, T4.stats$b, col=oranges[4], lwd=1.5)
abline(T5.stats$a, T5.stats$b, col=oranges[5], lwd=1.5)
```

The second half of the code could have been easily replaced with a short loop:

```
for(i in 1:5){
  myStats <- SummaryStatsA(Orange$age[Orange$Tree==i], Orange$circum[Orange$Tree==i])
  abline(myStats$a, myStats$b, col=oranges[i], lwd=1.5)
}
```



Note that a linear regression is not typically the best way to model growth curves but are often *sigmoidal* or S-shaped (see: http://en.wikipedia.org/wiki/Sigmoid_curve). However, even in this case the regressions are useful for showing, for example, that trees 4 and 5 have the highest growth rate and 1 and 2 have the lowest.

(c) *Create a second function called* ***SummaryStatsB(x,y)*** *which produces the exact same output but uses various R shortcuts such as* `sd()`, `cor()`, *and most crucially,* `lm()`. *Confirm that this function gives the same results as* ***SummaryStatsA()***.

```
SummaryStatsB <- function(x, y){
  r <- cor(x,y)
  a <- as.numeric(mylm$coef[1])
  b <- as.numeric(mylm$coef[2])
  SStotal <- sum( (y-mean(y))^2 )
  SSerror <- sum(mylm$resid^2)
  SSmodel <- SStotal-SSerror
  return(list(x.bar = x.bar, y.bar = y.bar, x.sd = x.sd, y.sd = y.sd,
           r = r, a = a, b = b,
           SStotal = SStotal, SSerror = SSerror, SSmodel = SSmodel,
           r2 = r2)
}
```

and a loop to create some comparative output:

```
for(i in 1:5)
{
  myResults <- cbind(SummaryStatsA(Orange$age[Orange$Tree == i], Orange$circum[Orange$Tree == i]),
                     SummaryStatsB(Orange$age[Orange$Tree == i], Orange$circum[Orange$Tree == i]))
  if(i == 1) Results <- myResults
  else Results <- data.frame(Results, myResults)
}
names(Results) <- paste("Tree", rep(1:5, each=2), rep(c("A","B")))
```

And the output:

Statistic	Tree 1 A	Tree 1 B	Tree 2 A	Tree 2 B	Tree 3 A	Tree 3 B	Tree 4 A	Tree 4 B	Tree 5 A	Tree 5 B
x.bar	922.14	922.14	922.14	922.14	922.14	922.14	922.14	922.14	922.14	922.14
y.bar	99.57	99.57	135.29	135.29	94.00	94.00	139.29	139.29	111.14	111.14
x.sd	523.63	523.63	523.63	523.63	523.63	523.63	523.63	523.63	523.63	523.63
y.sd	43.29	43.29	66.32	66.32	42.98	42.98	71.90	71.90	58.86	58.86
r	0.99	0.99	0.99	0.99	0.99	0.99	0.98	0.98	0.99	0.99
a	24.44	24.44	19.96	19.96	19.20	19.20	14.64	14.64	8.76	8.76
b	0.08	0.08	0.13	0.13	0.08	0.08	0.14	0.14	0.11	0.11
SStotal	11245.71	11245.71	26393.43	26393.43	11084.00	11084.00	31015.43	31015.43	20786.86	20786.86
SSerror	324.48	324.48	662.88	662.88	260.55	260.55	956.41	956.41	506.67	506.67
SSmodel	10921.23	10921.23	25730.55	25730.55	10823.45	10823.45	30059.02	30059.02	20280.19	20280.19
r2	0.97	0.97	0.97	0.97	0.98	0.98	0.97	0.97	0.98	0.98

Note how consistently high the r^2 is for these data.

2. **Anscombe's Quartet**

(a) *Load the data into* `R` *(by typing "*`data(anscombe)`*" directly into R) and (without plotting the data) use either of your summary statistics functions from problem 1 to obtain and tabulate the summary statistics for all four data sets.*

```
data(anscombe)
attach(anscombe)
cbind(xy1 = SummaryStatsA(x1,y1),
      xy2 = SummaryStatsA(x2,y2),
      xy3 = SummaryStatsA(x3,y3),
      xy4 = SummaryStatsA(x4,y4))
```

	xy1	xy2	xy3	xy4
x.bar	9.00	9.00	9.00	9.00
y.bar	7.50	7.50	7.50	7.50
x.sd	3.32	3.32	3.32	3.32
y.sd	2.03	2.03	2.03	2.03
r	0.82	0.82	0.82	0.82
a	3.00	3.00	3.00	3.00
b	0.50	0.50	0.50	0.50
SStotal	41.27	41.28	41.23	41.23
SSerror	13.76	13.78	13.76	13.74
SSmodel	27.51	27.50	27.47	27.49
r2	0.67	0.67	0.67	0.67

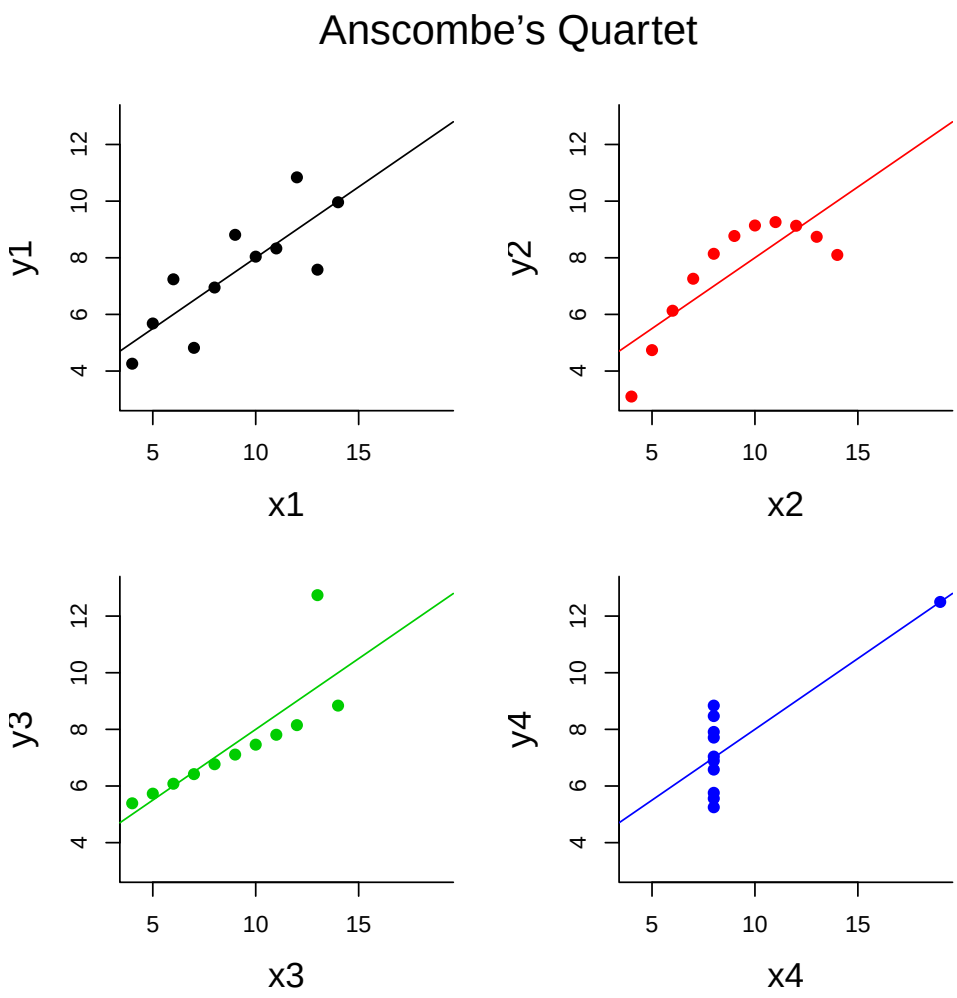
Clearly, most of these statistics are identical for all four paired datasets.

(b) *Plot all four data sets (on a single plot), with regression lines.*

```
par(mfrow=c(2,2), mar=c(4,4,2,2), oma= c(0,0,2,0))
plot(x1,y1, col=1, pch=19, xlim=c(4,19), ylim=c(3,13)); abline(lm(y1~x1), col=1)
plot(x2,y2, col=2, pch=19, xlim=c(4,19), ylim=c(3,13)); abline(lm(y2~x2), col=2)
plot(x3,y3, col=3, pch=19, xlim=c(4,19), ylim=c(3,13)); abline(lm(y3~x3), col=3)
plot(x4,y4, col=4, pch=19, xlim=c(4,19), ylim=c(3,13)); abline(lm(y4~x4), col=4)
mtext("Anscombes Quartet", outer = TRUE, cex=1.5)
```

Or with a simple loop:

```
for(i in 1:4)
{
  x <- anscombe[,i]
  y <- anscombe[,i+4]
  plot(x, y , col=1, pch=19, xlim=c(4,19), ylim=c(3,13))
}
# etc...
```



(c) *What do you think Dr. Anscombe's message was when he generated (with, I imagine, considerable effort) these sets of numbers?*

The regression lines and, in fact, all summary statistics are identical but for four different reasons. The first is more or less a "standard" set of data for a linear regression: the residuals are more or less normally distributed with a uniform variance. The second is very clearly a deterministic curve, likely quadratic. The third is very linear, with a much lower slope than the regression line, and one outlier with enough influence to raise the slope significantly and lower the correlation from 1 to 0.816. Finally, the last one consists of 10 distributed measurements at one x value, and a single high x and y extreme observation. The clearest message in these data is that it is vital to plot data before modeling or analyzing it: even a cursory visual analysis is sufficient to identify the main patterns. A secondary important point is to illustrate the influence of an inappropriate model (e.g. linear on a parabola in number 2) and of outliers (numbers 3 and 4) on regression coefficients.

Bonus Problem: *Fit a quadratic function to the four data sets, plot the fitted line, and report the coefficients and the r^2 value (in a table).*

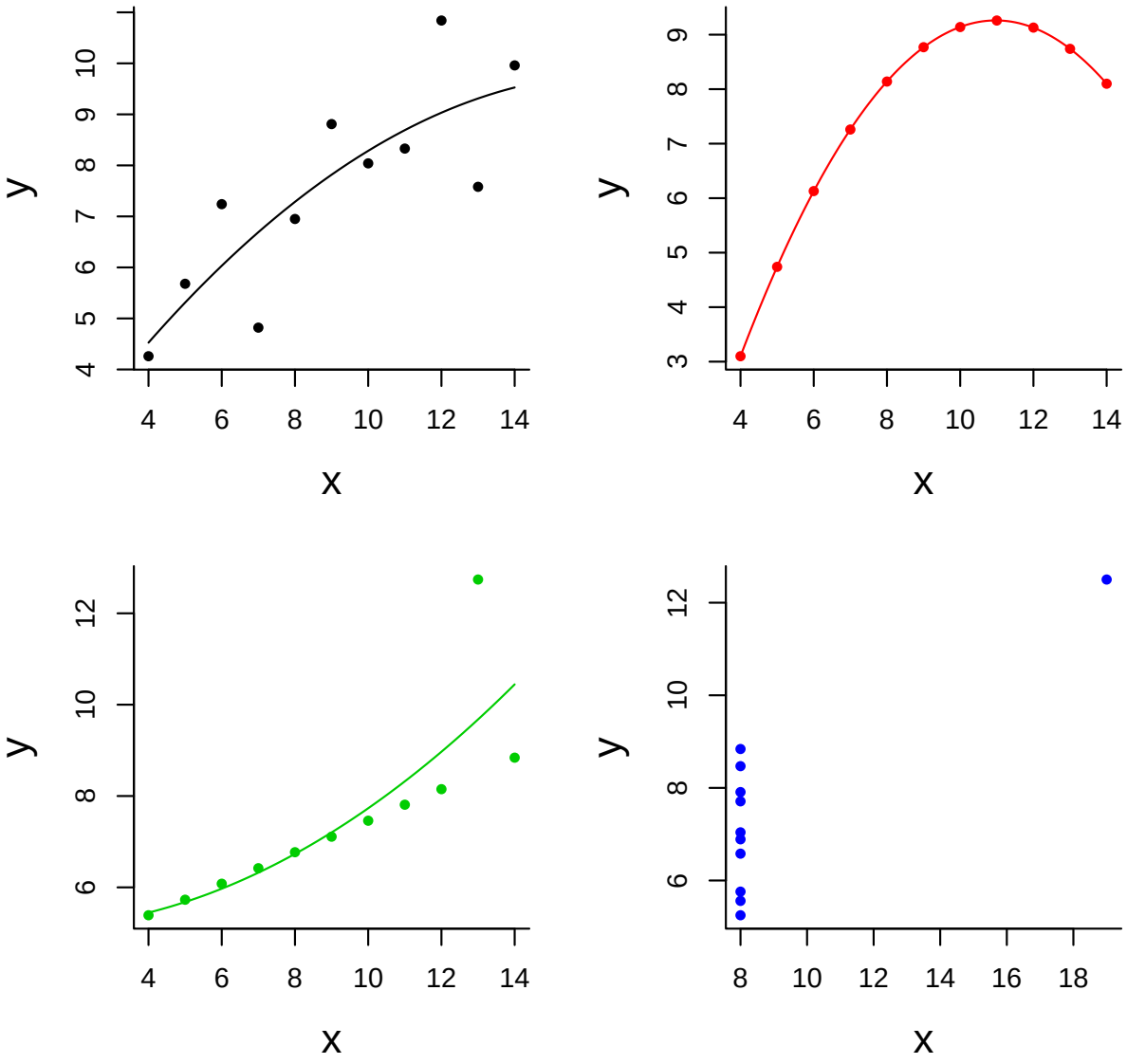
The following loop will perform the estimation and plotting for all four datasets:

```
for(i in 1:4)
{
  x <- anscombe[,i]
  y <- anscombe[,i+4]
  x2 <- x^2
  mylm <- lm(y ~ x+x2 )
  y.hat <- mylm$coef[1] + mylm$coef[2] * x + mylm$coef[3] * x^2
  r2 <- 1 - sum((y - y.hat)^2) / sum((y - mean(y)^2))

  stats <- data.frame(beta0 = mylm$coef[1], beta1 = mylm$coef[2], beta2 = mylm$coef[3], r2 = r
  if(i == 1)
    Results <- stats
  else
    Results <- rbind(Results, stats)

  plot(x, y, col=i, pch=20,  xlab="x", ylab="y")
   curve(mod$coef[1] + mod$coef[2] * x + mod$coef[3] * x^2, add=TRUE, col=i)
}
```

The results are:

	β_0	β_1	β_2	r^2
Data 1	0.76	-6.00	5.11	0.69
Data 2	1.07	2.78	-0.04	1.00
Data 3	-0.03	-0.13	0.03	0.68
Data 4	1.02	1.00	1.02	NA

Figure on the following page. Note that the second dataset is, indeed, an exact parabola as evidenced by the $r^2 = 1$, and that it is impossible to fit a parabola to the last curve because there are only two x-values. Any curve that goes through the mean of the cluster on the left and the actual point on the right will have the same sum of squares, and therefore the curve can not be minimized.


y
x
4
5
6
7
8
9
10
4
6
8
10
12
14
y
x
3
4
5
6
7
8
9
4
6
8
10
12
14
y
x
6
8
10
12
4
6
8
10
12
14
y
x
6
8
10
12
8
10
12
14
16
18