**PROBLEM 1.**

```
#standardizing data X
standardizeX = function(X)
{
  mX = matrix(colMeans(X), nr=dim(X)[1], nc=dim(X)[2], byrow=T)
  X0 = X - mX
  sd = sqrt(colMeans(X0^2))
  X0 = X0%*%diag(1/sd)
  output = list(X0=X0, diag=sd)
  return(output)
}

#subgradient descent for optimizing linear MAE
sub_gd = function(X,y,eta)
{
  n = dim(X)[1]
  d = dim(X)[2]
  standX = standardizeX(X)
  X0 = cbind(rep(1,n), standX$X0)

  b0 = numeric(d+1)
  res0 = y
  b1 = rep(1, d+1)
  iter = 0

  repeat{
    if(max(abs(b1-b0)) < 10^(-5) | iter > 20000) break

    b1 = b0
    subgrad = sign(res0)
    grad = -t(X0) %*% subgrad/n
    b0 = b0 - eta*grad
    res0 = y - X0%*%b0
    iter = iter + 1
  }

  b0[-1] = b0[-1]/standX$diag
  b0[1] = b0[1] - sum(colMeans(X)*b0[-1])
  output = list(beta=b0, res=res0, iter=iter)
  return(output)

}
```

**PROBLEM 2.**

```
df = read.csv("education.csv")
X = data.matrix(df[,4:6])
y = data.matrix(df[,7])
#subgradient descent for MAE
maefit <- sub_gd(X,y,1)
#MSE
lsefit <- lm(y~X)
```

MAE estimator output:

```
$beta
             [,1]
[1,] -356.75802116
[2,]    0.02870102
[3,]    0.06467542
[4,]    0.96283273
```

MSE estimator output:

```
Coefficients:
(Intercept)          XX1          XX2          XX3
 -5.566e+02   -4.269e-03    7.239e-02    1.552e+00
```

In general, MAE tends to select the smaller values (in absolute value). For example, $\beta_0^*$ for MAE is |-356.76| < |-556.6| for MSE.

**PROBLEM 3.**

**Part 1:**

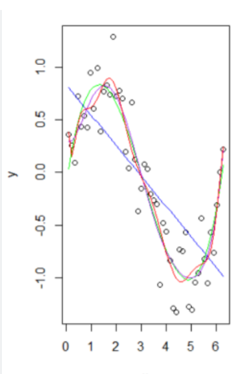
```r
build_poly <- function(x,d)
{
  X = matrix(nrow=length(x), ncol=d)
  for(j in 1:d)
  {
    X[,j] = x^(j)
  }
  X = cbind(1, X)
  return(X)
}
```

**Part 2:**

```r
df = read.csv("polynomial.csv")
x = data.matrix(df[,1])
y = data.matrix(df[,2])

#lse for d = 1,3,7,12
X1 <- build_poly(x,1)
coef1 <- lm(y~X1)$coefficients
coef1 <- coef1[-2]

X3 <- build_poly(x,3)
coef3 <- lm(y~X3)$coefficients
coef3 <- coef3[-2]

X7 <- build_poly(x,7)
coef7 <- lm(y~X7)$coefficients
coef7 <- coef7[-2]

X12 <- build_poly(x,12)
coef12 <- lm(y~X12)$coefficients
coef12 <- coef12[-2]

plot(x,y)
lines(sort(x), fitted(lm(y~X1))[order(x)], col='blue')
lines(sort(x), fitted(lm(y~X3))[order(x)], col='green')
lines(sort(x), fitted(lm(y~X7))[order(x)], col='purple')
lines(sort(x), fitted(lm(y~X12))[order(x)], col='red')
```

Estimated coefficients for degrees 1, 3, 7, and 12:

```
> coef1
(Intercept)         X12
  0.8343183  -0.2897662
> coef3
(Intercept)         X32         X33         X34
-0.11521169  1.58466469 -0.76372992  0.08215316
> coef7
  (Intercept)           X72           X73           X74           X75           X76
 0.3223564673 -0.4059901868  1.6430356488 -1.1213179058  0.2724796750 -0.0225108752
          X77           X78
-0.0007799992  0.0001539027
> coef12
  (Intercept)          X122          X123          X124          X125          X126
 1.321786e+00 -1.498618e+01  6.901167e+01 -1.475850e+02  1.786164e+02 -1.337016e+02
         X127          X128          X129         X1210         X1211         X1212
 6.510101e+01 -2.118972e+01  4.645912e+00 -6.770489e-01  6.281960e-02 -3.356027e-03
        X1213
 7.851603e-05
```



**Part 3:**

```r
#PART 3
yhat1 = X1%*%coef1
MSE1 = sum((y-yhat1)^2) / 50
RMSE1 = sqrt(2*MSE1)

yhat3 = X3%*%coef3
MSE3 = sum((y-yhat3)^2) / 50
RMSE3 = sqrt(2*MSE3)

yhat7 = X7%*%coef7
MSE7 = sum((y-yhat7)^2) / 50
RMSE7 = sqrt(2*MSE7)

yhat12 = X12%*%coef12
MSE12 = sum((y-yhat12)^2) / 50
RMSE12 = sqrt(2*MSE12)
```

RMSE output:

```
> RMSE1
[1] 0.6673336
> RMSE3
[1] 0.3656913
> RMSE7
[1] 0.3530707
> RMSE12
[1] 0.339871
```

From degree 1 to degree 3, we see a major decrease in the RMSE. However, from then on, any increase in degree does not correlate to a significant decrease in the RMSE. Because simpler models are valued over complex models due to overfitting issues, I believe that the model with degree 3 is the best fit for this data.