HW 1 Solutions

Wendy Liang

9/7/2020

7.2 Fake-data simulation and regression:

Simulate 100 data points from the linear model, y = a + bx + error, with a = 5, b = 7, the values of x being sampled at random from a uniform distribution on the range [0, 50], and errors that are normally distributed with mean 0 and standard deviation 3.

7.2a

Fit a regression line to these data and display the output.

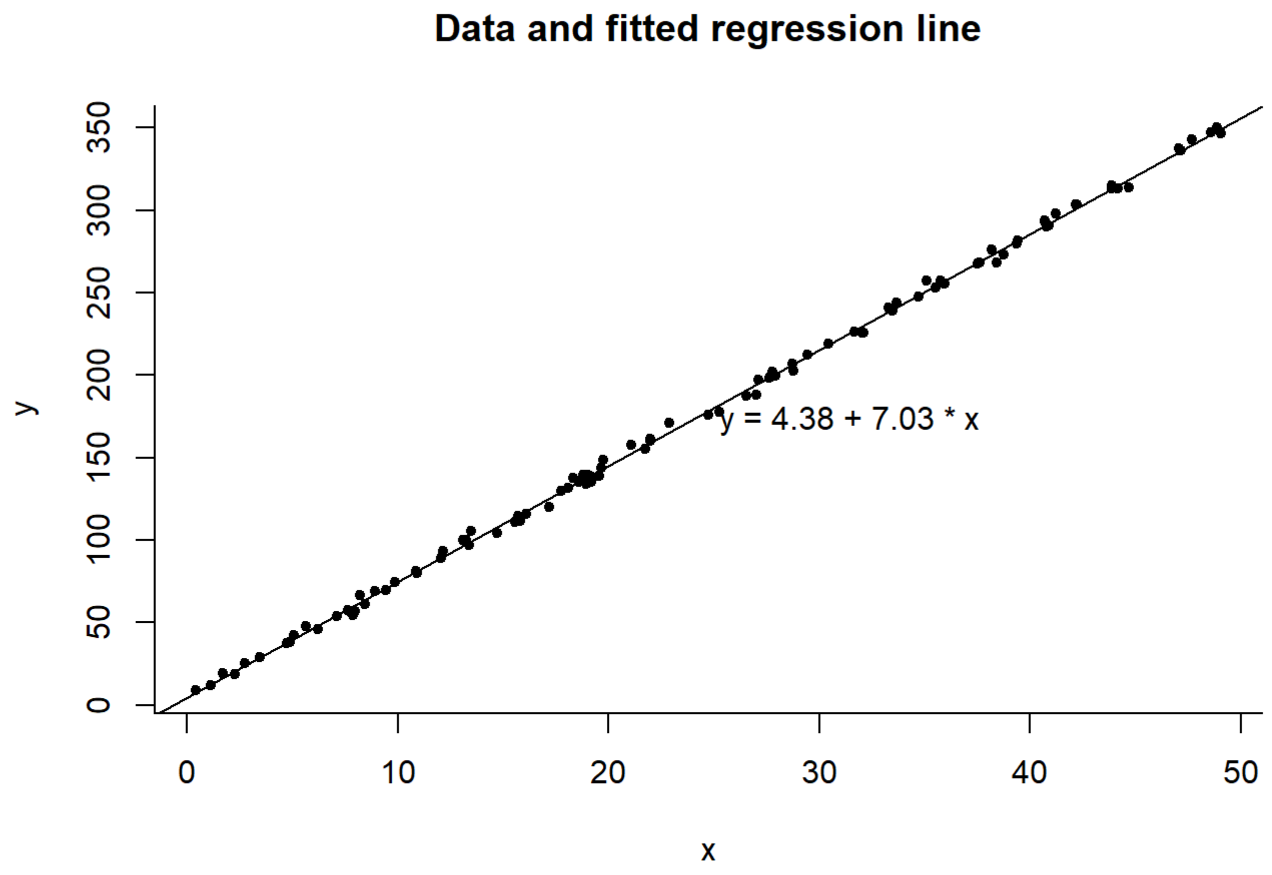
```r
#all the chp7 learn from Simplest examples
library(rstanarm)
set.seed(12)
x=runif(100,0,50)
y=5+7*x+rnorm(100,0,3)
fake=data.frame(x,y)
fit=stan_glm(y~x, data=fake, seed=12, refresh = 0)
print(fit)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y ~ x
##  observations: 100
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) 4.4    0.6
## x           7.0    0.0
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 2.9    0.2
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

7.2b

Graph a scatterplot of the data and the regression line.

```
plot(fake$x, fake$y, main="Data and fitted regression line", bty="l", pch=20,
     xlab = "x", ylab = "y")
a_hat = coef(fit)[1]
b_hat = coef(fit)[2]
abline(a_hat, b_hat)
x_bar = mean(fake$x)
text(x_bar, a_hat + b_hat*x_bar, paste("   y =", round(a_hat, 2), "+", round(b_hat, 2), "* x"),
adj=0)
```



7.2c

Use the text function in R to add the formula of the fitted line to the graph. **see the answer of 7.2b**

7.3 Fake-data simulation and fitting the wrong model:

Simulate 100 data points from the model, y = a + bx + cx^2 + error, with the values of x being sampled at random from a uniform distribution on the range [0, 50], errors that are normally distributed with mean 0 and standard deviation 3, and a, b, c chosen so that a scatterplot of the data shows a clear nonlinear curve.

7.3 a

Fit a regression line stan_glm(y ~ x) to these data and display the output.

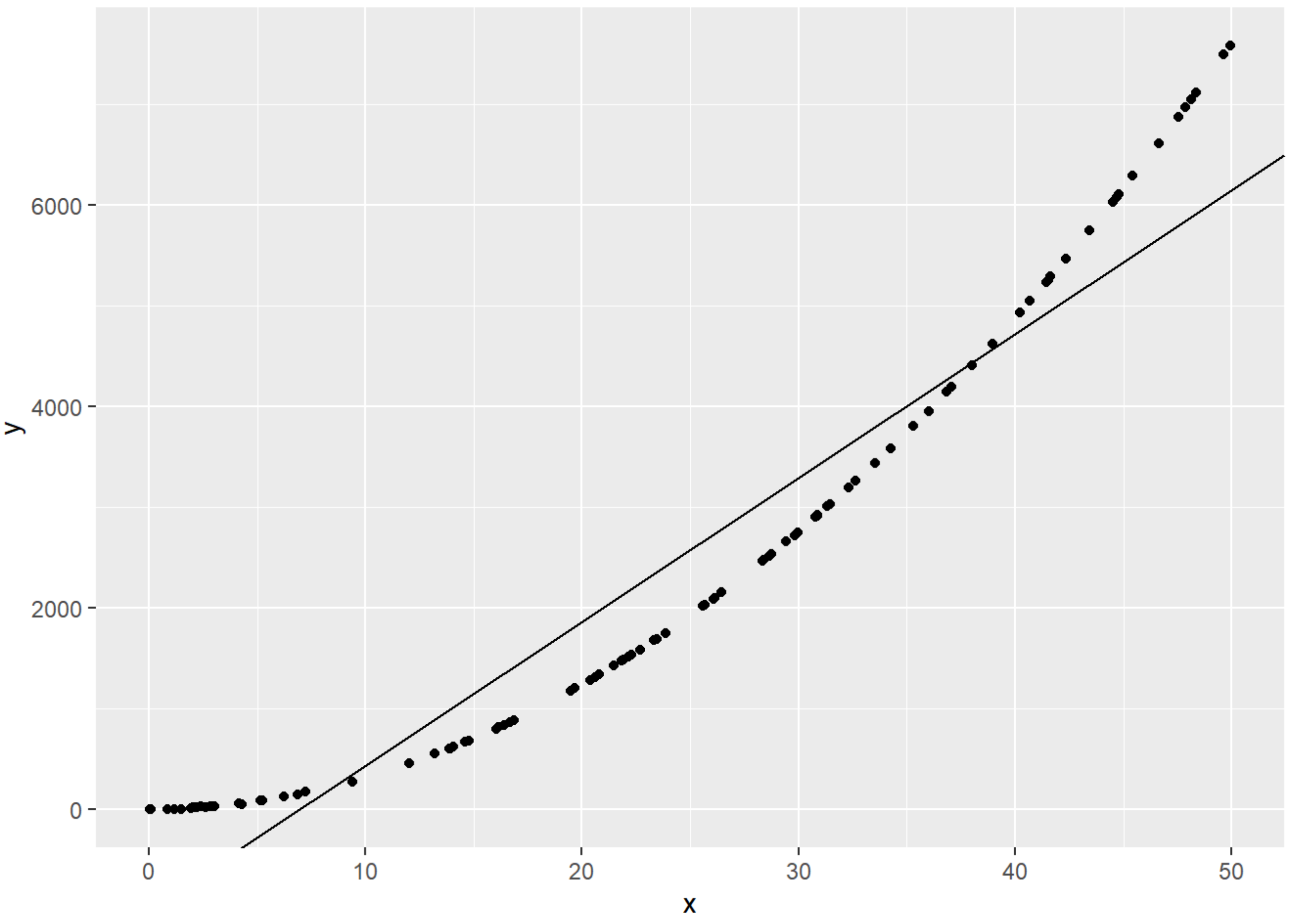
```
x=runif(100,0,50)
a=1
b=2
c=3
y=a+b*x+c*x^2+rnorm(100,0,3)
mydata7=data.frame(x,y)
fit7=stan_glm(y~x,data=mydata7,refresh=0)
print(fit7)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y ~ x
##  observations: 100
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) -995.9  118.9
## x            142.9    4.2
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 626.7   44.5
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

7.3b

Graph a scatterplot of the data and the regression line. This is the best-fit linear regression. What does ??? best-fit??? mean in this context?

```
ggplot(mydata7,aes(x,y))+geom_point()+geom_abline(intercept=-995.9,slope=142.9)
```



“best fit” is defined as minimizing the sum of squared errors

7.6 Formulating comparisons as regression models:

Take the election forecasting model and simplify it by creating a binary predictor defined as x = 0 if income growth is less than 2% and x = 1 if income growth is more than 2%.

```
library(readr)
mydata=read.table("C:/Users/dell/Documents/ROS/ElectionsEconomy/data/hibbs.dat",header=T)
mydata$growth[mydata$growth<2]=0
mydata$growth[mydata$growth>2]=1

x_0=mydata$growth[mydata$growth==0]
x_1=mydata$growth[mydata$growth==1]

y_0=mydata$vote[mydata$growth==0]
y_1=mydata$vote[mydata$growth==1]

n_0=length(x_0)
n_1=length(x_1)
```

7.6a

Compute the difference in incumbent party???s vote share on average, comparing those two groups of elections, and determine the standard error for this difference.

```
diff = mean(y_1) - mean(y_0)
se_0 = sd(y_0)/sqrt(n_0)
se_1 = sd(y_1)/sqrt(n_1)
se = sqrt(se_0^2 + se_1^2)
print(diff)
```

```
## [1] 5.5075
```

7.6b

Regress incumbent party???s vote share on the binary predictor of income growth and check that the resulting estimate and standard error are the same as above.

```
y = c(y_0, y_1)
x = c(x_0,x_1)
fake = data.frame(y, x)
fit_com = stan_glm(y ~ x, data = fake, refresh = 0,
                  prior_intercept = NULL, prior = NULL, prior_aux = NULL)
print(fit_com)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y ~ x
##  observations: 16
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) 49.2    1.9
## x            5.6    2.6
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 5.3    1.0
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

8.8 Comparing lm and stan_glm:

Use simulated data to compare least squares estimation to default Bayesian regression:

8.8a

Simulate 100 data points from the model, y = 2 + 3x + error, with predictors x drawn from a uniform distribution from 0 to 20, and with independent errors drawn from the normal distribution with mean 0 and standard deviation 5. Fit the regression of y on x data using lm and stan_glm (using its default settings) and check that the two programs give nearly identical results.

```
library(rstanarm)
set.seed(12)
x=runif(100,0,20)
y=2+3*x+rnorm(100,0,5)
fake8=data.frame(x,y)
fit1=lm(y~x,data=fake)
fit2=stan_glm(y~x,data=fake,refresh=0)
print(fit1)
```

```
##
## Call:
## lm(formula = y ~ x, data = fake)
##
## Coefficients:
## (Intercept)            x
##      49.301        5.508
```

```
print(fit2)
```

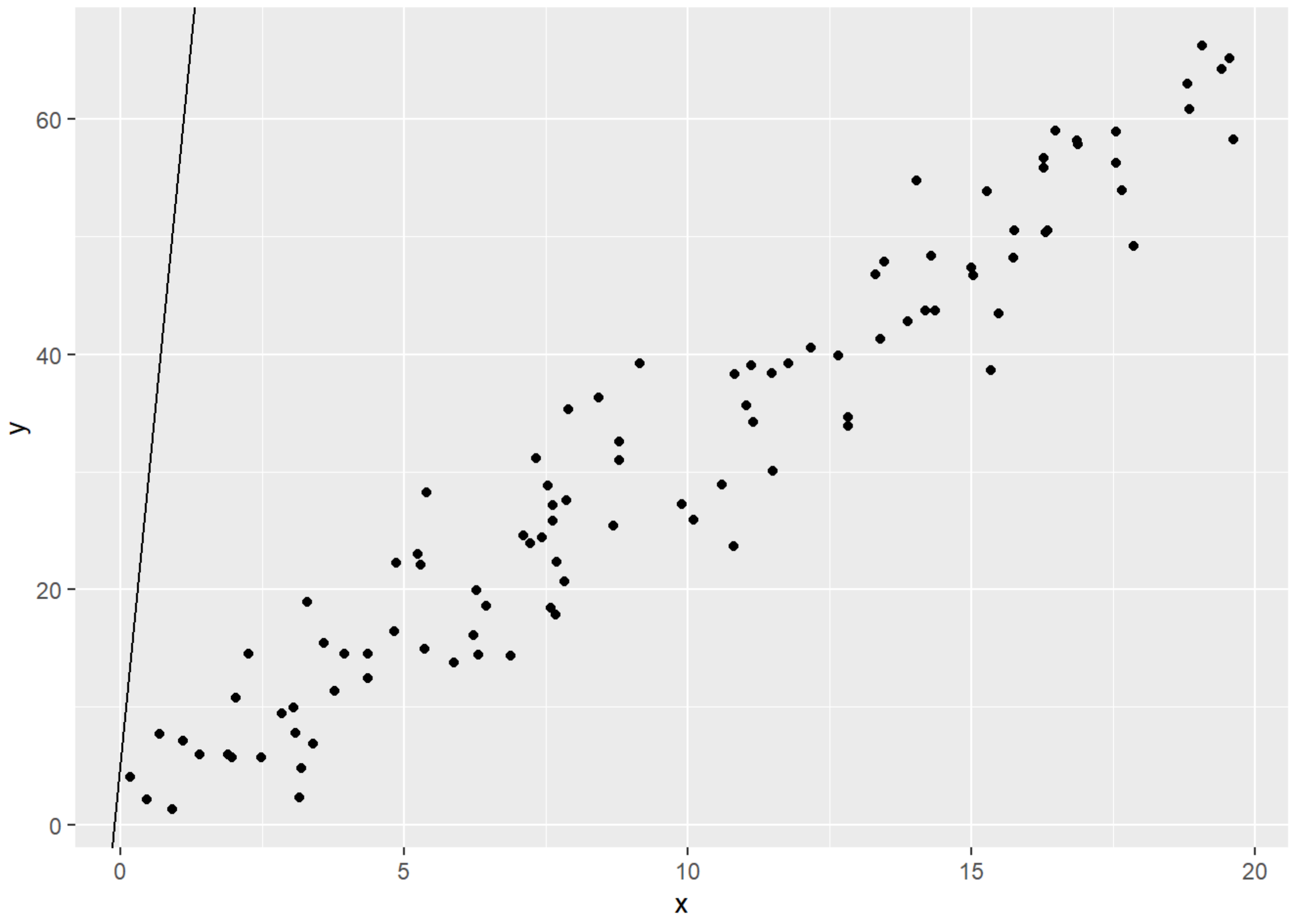
```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y ~ x
##  observations: 16
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) 49.4    1.8
## x            5.4    2.6
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 5.2    1.0
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

8.8b

Plot the simulated data and the two fitted regression lines.

```
library(ggplot2)
a_hat1=coef(fit1)[1]
b_hat1=coef(fit1)[2]
a_hat2=coef(fit1)[1]
b_hat2=coef(fit1)[2]
ggplot(fake8,aes(x,y))+geom_point()+geom_abline(intercept=b_hat1,slope=a_hat1)+geom_abline(inte
rcept=b_hat2,slope=a_hat2,aes(colour=Orange))
```

```
## Warning: geom_abline(): Ignoring `mapping` because `slope` and/or `intercept`
## were provided.
```

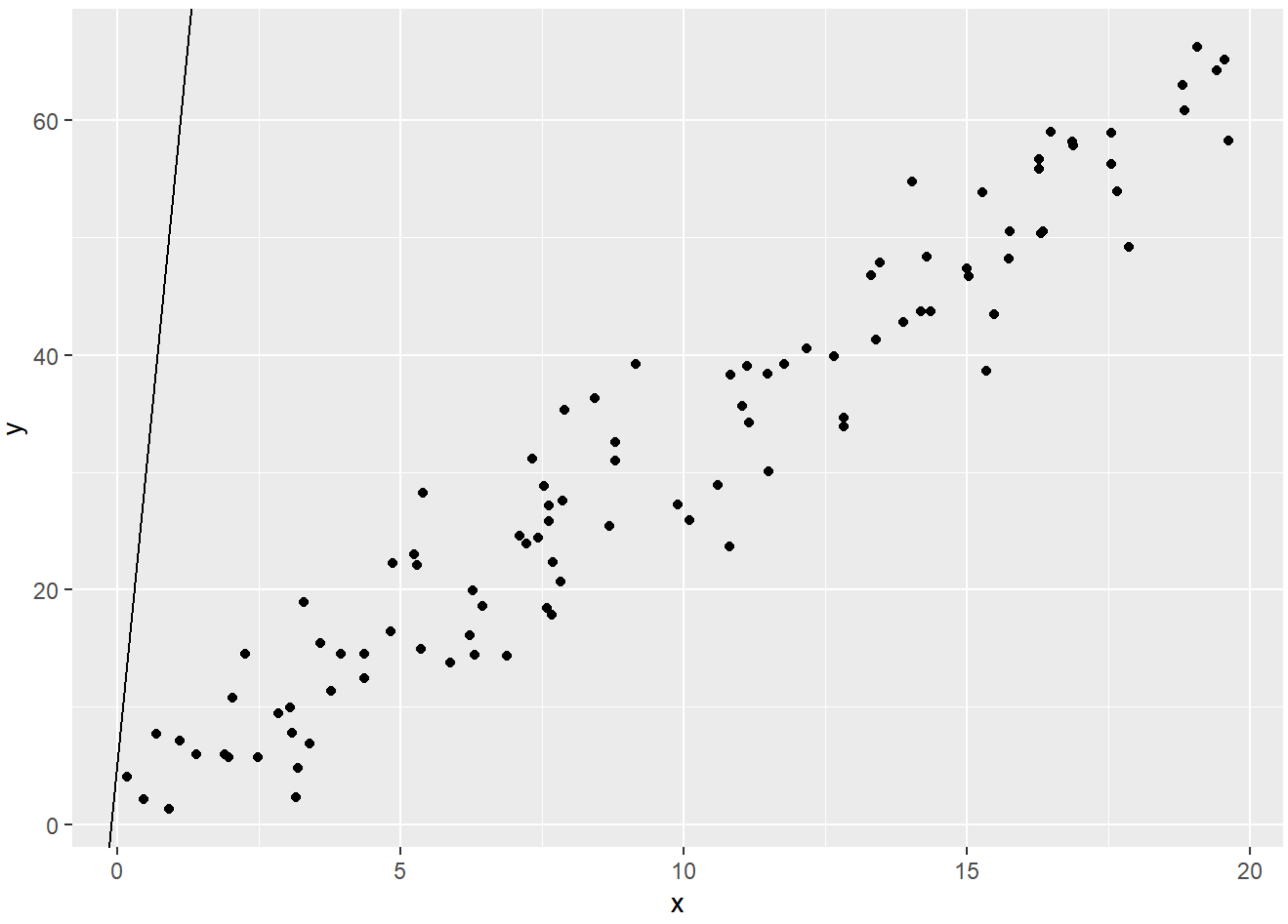


8.8c

Repeat the two steps above, but try to create conditions for your simulation so that lm and stan_glm give much different results.

```
##
## Call:
## lm(formula = y ~ x, data = fake)
##
## Coefficients:
## (Intercept)            x
##      49.301        5.508
## stan_glm
##  family:       gaussian [identity]
##  formula:      y ~ x
##  observations: 16
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) 49.4    1.8
## x            5.4    2.6
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 5.2    1.0
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
## Warning: geom_abline(): Ignoring `mapping` because `slope` and/or `intercept`
## were provided.
```



10.1 Regression with interactions:

Simulate 100 data points from the model, y = b0 + b1 x + b2 z + b3 xz + error, with a continuous predictor x and a binary predictor z, coefficients b = c(1, 2, -1, -2), and errors drawn independently from a normal distribution with mean 0 and standard deviation 3, as follows. For each data point i, first draw zi, equally likely to take on the values 0 and 1. Then draw xi from a normal distribution with mean zi and standard deviation 1. Then draw the error from its normal distribution and compute yi.

10.1a

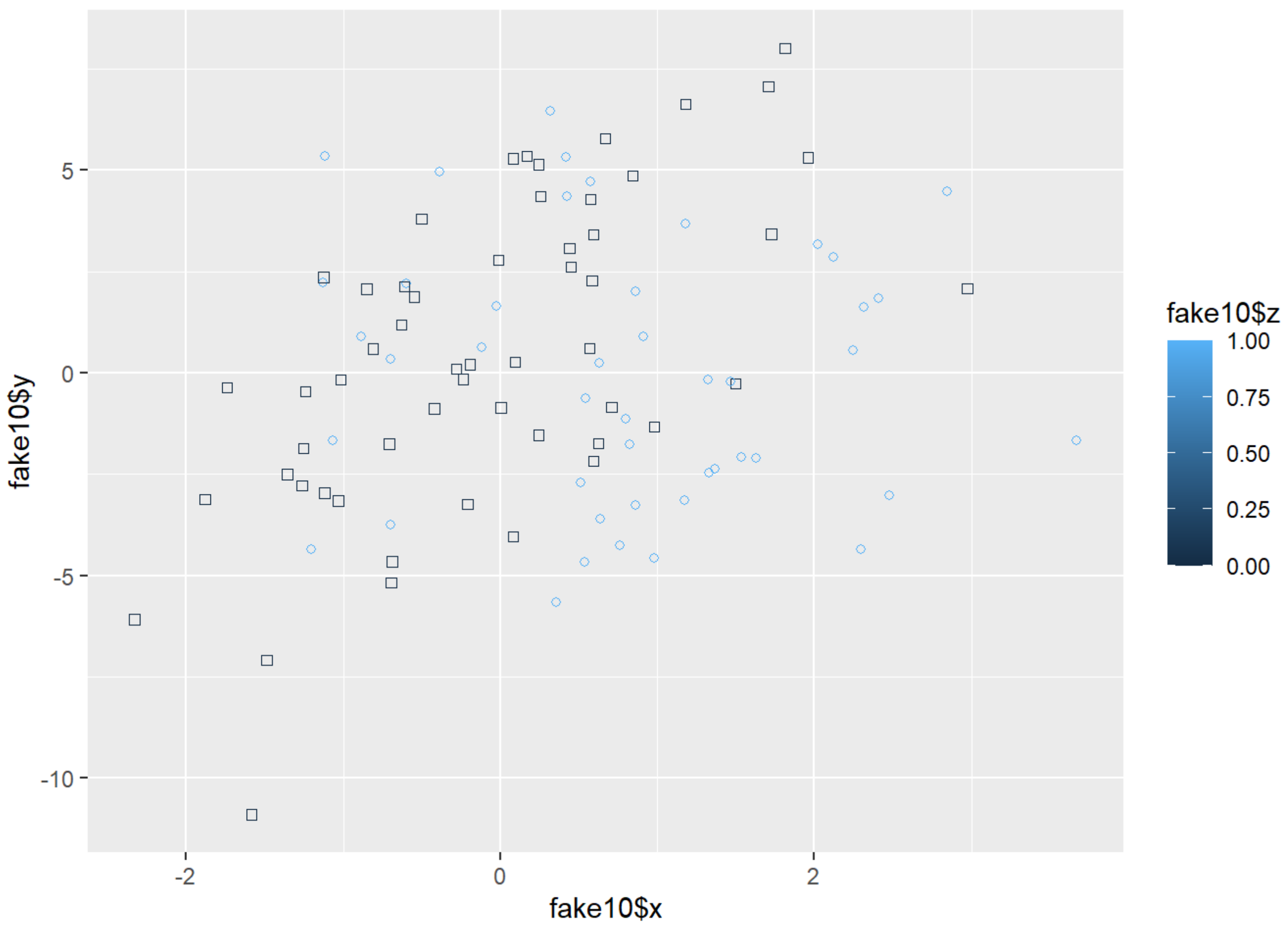
Display your simulated data as a graph of y vs. x, using dots and circles for the points with z = 0 and 1, respectively.

```
z=rep(0,100)
for(i in 1:100){
z[i]=round((runif(1,0,1)))
x[i]=rnorm(1,z[i],1)
y[i]=1+2*x[i]-z[i]-2*x[i]*z[i]+rnorm(1,0,3)
}
fake10=data.frame(x,y,z)
ggplot(fake10,aes(fake10$x,fake10$y,colour=fake10$z))+geom_point(shape=fake10$z)
```

```
## Warning: Use of `fake10$x` is discouraged. Use `x` instead.
```

```
## Warning: Use of `fake10$y` is discouraged. Use `y` instead.
```

```
## Warning: Use of `fake10$z` is discouraged. Use `z` instead.
```



10.1b

Fit a regression predicting y from x and z with no interaction. Make a graph with the data and two parallel lines showing the fitted model.

```
fit10_1 = stan_glm(y~x+z,data=fake10,refresh=0)
print(fit10_1)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y ~ x + z
##  observations: 100
##  predictors:   3
## ------
##             Median MAD_SD
## (Intercept)  0.6    0.5  
## x            1.0    0.3  
## z           -1.5    0.7  
## 
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 3.4    0.2   
## 
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

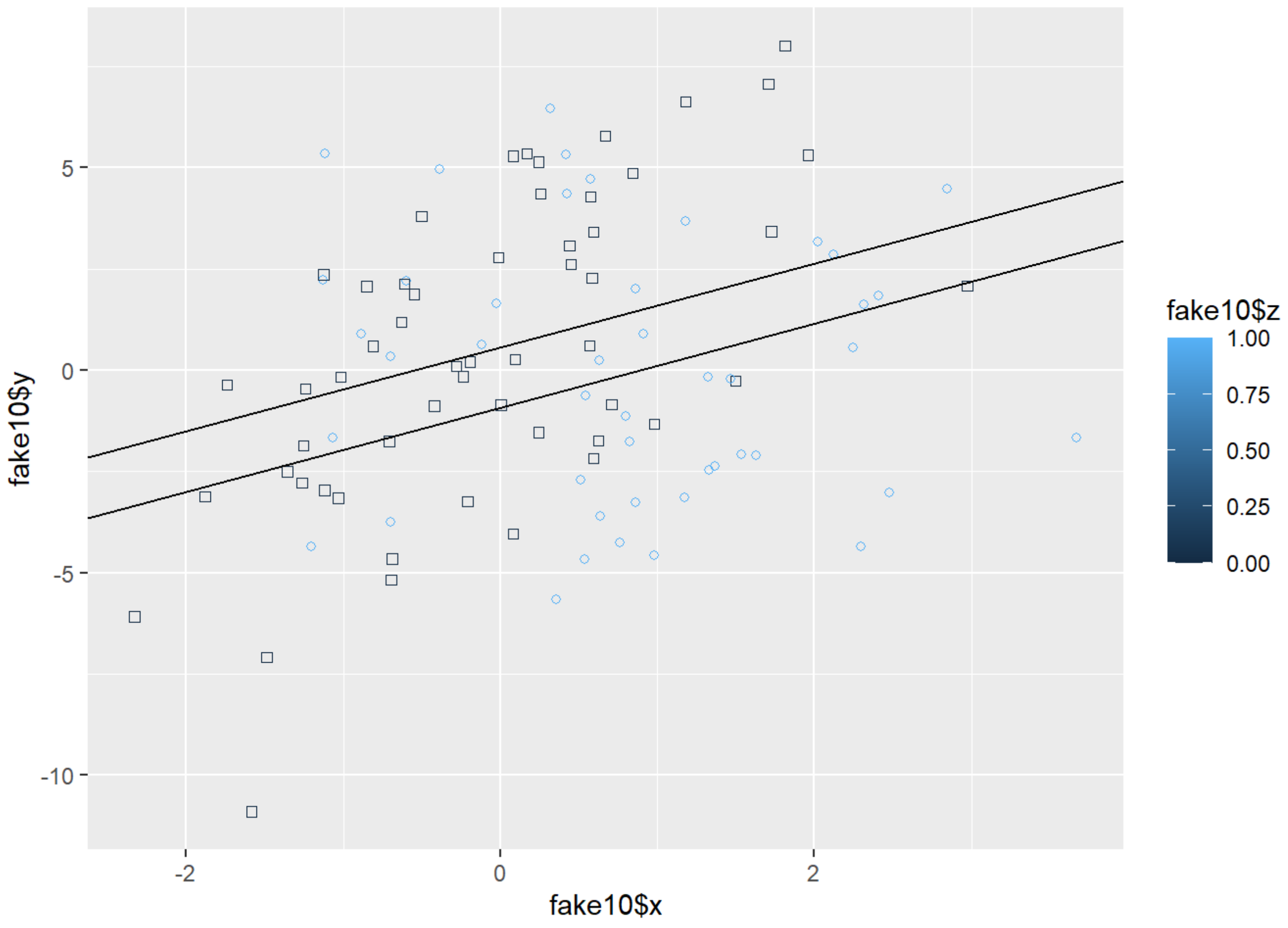
```
#y=0.4+1.1x-1.9z
intercept_0=coef(fit10_1)[1]
intercept_1=coef(fit10_1)[1]+coef(fit10_1)[3]
slope=coef(fit10_1)[2]
ggplot(fake10,aes(fake10$x,fake10$y,colour=fake10$z))+geom_point(shape=fake10$z)+geom_abline(in
tercept=intercept_0,slope=slope)+geom_abline(intercept=intercept_1,slope=slope,aes(colour="#000
099"))
```

```
## Warning: geom_abline(): Ignoring `mapping` because `slope` and/or `intercept`
## were provided.
```

```
## Warning: Use of `fake10$x` is discouraged. Use `x` instead.
```

```
## Warning: Use of `fake10$y` is discouraged. Use `y` instead.
```

```
## Warning: Use of `fake10$z` is discouraged. Use `z` instead.
```



10.1c

Fit a regression predicting y from x, z, and their interaction. Make a graph with the data and two lines showing the fitted model.

```
fit10_2 = stan_glm(y~x+z+x:z,data=fake10,refresh=0)
print(fit10_2)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y ~ x + z + x:z
##  observations: 100
##  predictors:   4
## ------
##             Median MAD_SD
## (Intercept)  0.7    0.4
## x            2.2    0.4
## z           -0.5    0.7
## x:z         -2.4    0.6
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 3.2    0.2
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

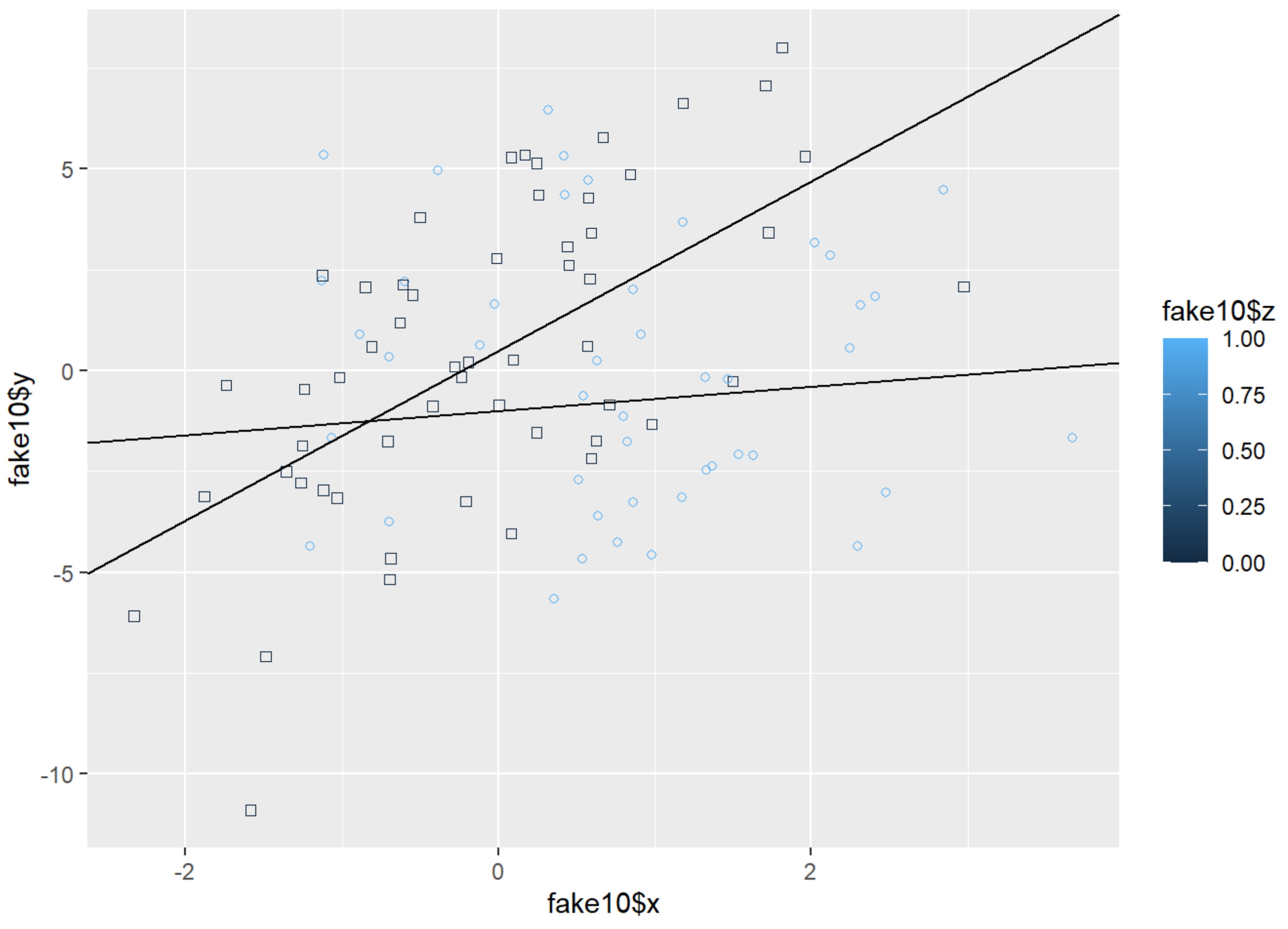
```
#y=0.5+2.1x-1.5z-1.8xz
ggplot(fake10,aes(fake10$x,fake10$y,colour=fake10$z))+geom_point(shape=fake10$z)+geom_abline(in
tercept=0.5,slope=2.1)+geom_abline(intercept=-1,slope=0.3,aes(colour="#000099"))
```

```
## Warning: geom_abline(): Ignoring `mapping` because `slope` and/or `intercept`
## were provided.
```

```
## Warning: Use of `fake10$x` is discouraged. Use `x` instead.
```

```
## Warning: Use of `fake10$y` is discouraged. Use `y` instead.
```

```
## Warning: Use of `fake10$z` is discouraged. Use `z` instead.
```



10.2 Regression with interactions:

Here is the output from a fitted linear regression of outcome y on pre-treatment predictor x, treatment indicator z, and their interaction:

10.2a

Write the equation of the estimated regression line of y on x for the treatment group and the control group, and the equation of the estimated regression line of y on x for the control group. $y = 1.2 + 1.6x + 2.7z + 0.7xz$ **for control group, z=0,we have** $y = 1.2 + 1.6x$ **for treatment group,z=1, we have** $y = 3.9 + 2.3x$

10.2b

Graph with pen on paper the two regression lines, assuming the values of x fall in the range (0, 10). On this graph also include a scatterplot of data (using open circles for treated units and dots for controls) that are consistent with the fitted model.

```
set.seed(10)
x=runif(100,0,10)
z=sample(c(0,1),100,prob=c(0.5,0.5),replace=TRUE)
y=1.2+1.6*x+2.7*z+0.7*x*z+rnorm(100,0,1)
data=data.frame(x,y,z)
fit=stan_glm(y~x+z+x:z,data=data,refresh=0)
print(fit)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y ~ x + z + x:z
##  observations: 100
##  predictors:   4
## ------
##             Median MAD_SD
## (Intercept) 0.9    0.3
## x           1.7    0.1
## z           3.0    0.4
## x:z         0.6    0.1
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 1.0    0.1
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

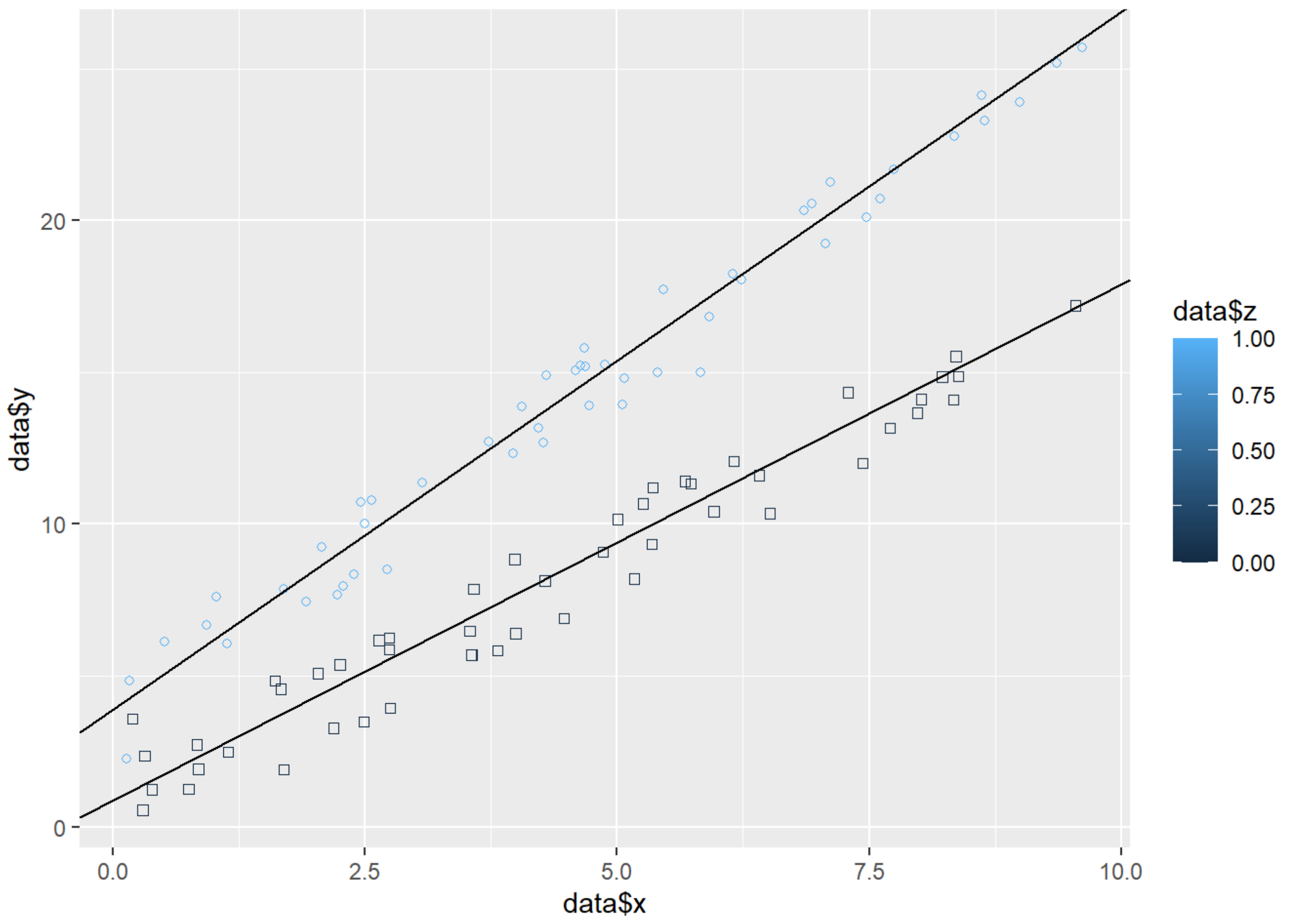
```
ggplot(data,aes(data$x,data$y,colour=data$z))+geom_point(shape=data$z)+geom_abline(intercept=0.
9,slope=1.7)+geom_abline(intercept=3.9,slope=2.3,aes(colour="#000099"))
```

```
## Warning: geom_abline(): Ignoring `mapping` because `slope` and/or `intercept`
## were provided.
```

```
## Warning: Use of `data$x` is discouraged. Use `x` instead.
```

```
## Warning: Use of `data$y` is discouraged. Use `y` instead.
```

```
## Warning: Use of `data$z` is discouraged. Use `z` instead.
```



10.5 Regression modeling and prediction:

The folder KidIQ contains a subset of the children and mother data discussed earlier in the chapter. You have access to children???s test scores at age 3, mother???s education, and the mother???s age at the time she gave birth for a sample of 400 children.

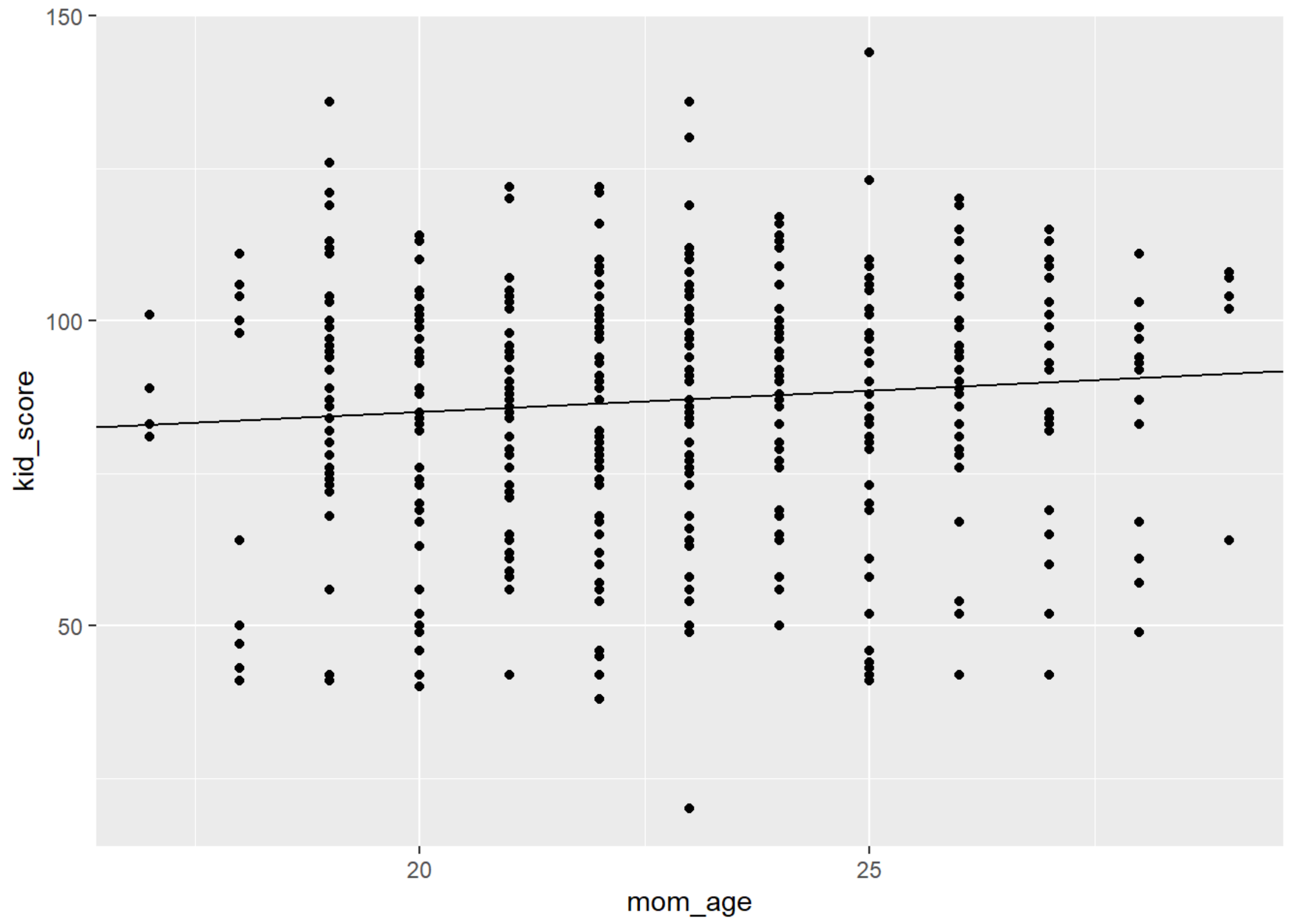
10.5a

Fit a regression of child test scores on mother???s age, display the data and fitted model, check assumptions, and interpret the slope coefficient. Based on this analysis, when do you recommend mothers should give birth? What are you assuming in making this recommendation?

```
Kidiq=read.csv("C:/Users/dell/Documents/ROS/KidIQ/data/kidiq.csv")
fit=stan_glm(kid_score~mom_age,data=Kidiq,refresh=0)
print(fit)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      kid_score ~ mom_age
##  observations: 434
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) 71.1    8.1
## mom_age      0.7    0.4
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 20.3    0.7
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
ggplot(kidiq,aes(mom_age,kid_score))+geom_point()+geom_abline(intercept=71.1,slope=0.7)
```



```
#y=0.7x+71.1
```

(1)coef of mom age: the difference of children score average value is 0.7 between two mom group with 1 year-old difference. (2)according to the scatterplot, I recommend mother in 24 year-old give birth since the mean value of childre score is highest. My assumption is there’s no other impact factor of children score.

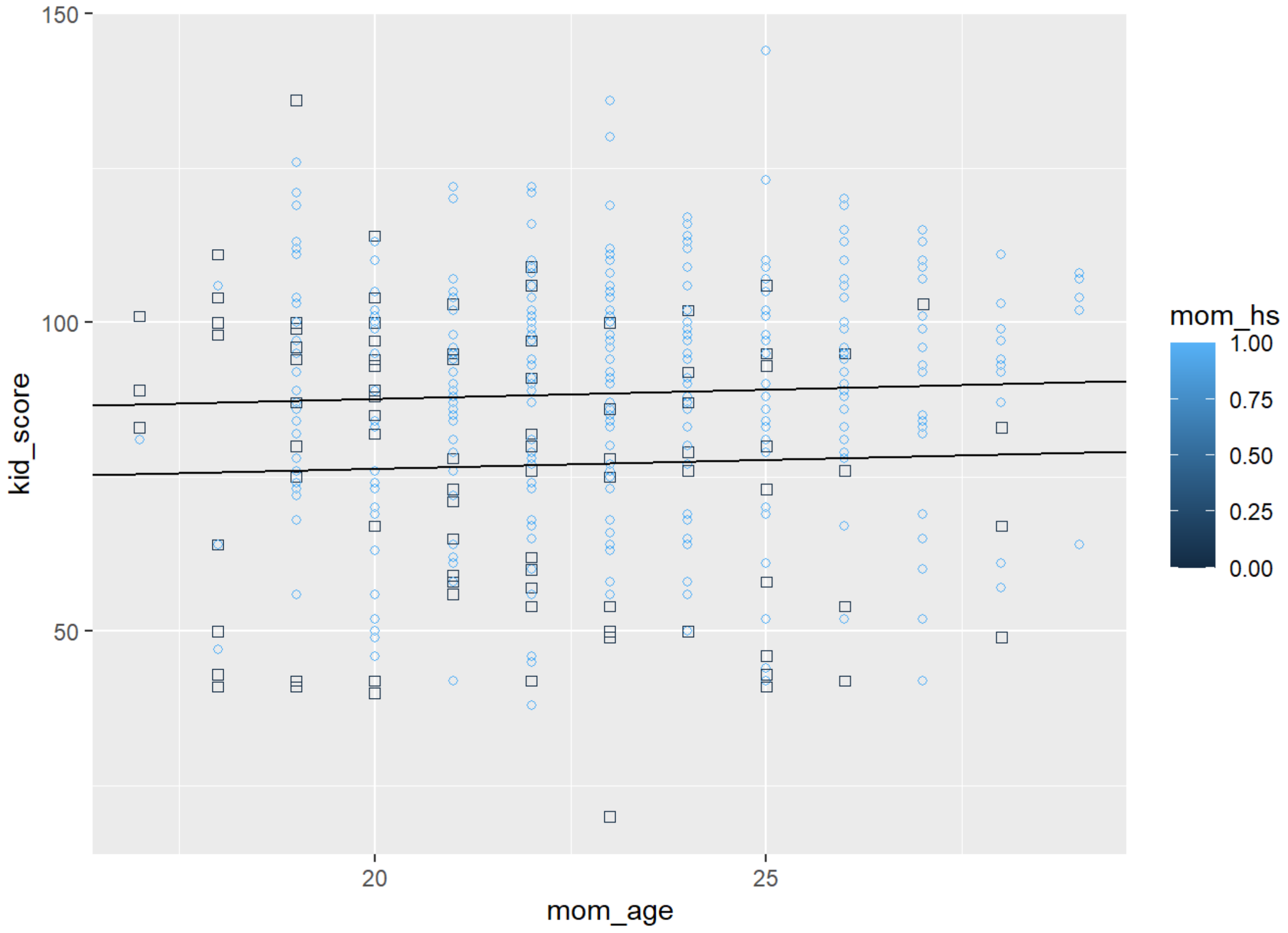
10.5b

Repeat this for a regression that further includes mother???s education, interpreting both slope coefficients in this model. Have your conclusions about the timing of birth changed?

```
fit=stan_glm(kid_score~mom_age+mom_hs,data=Kidiq,refresh=0)
print(fit)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      kid_score ~ mom_age + mom_hs
##  observations: 434
##  predictors:   3
## ------
##             Median MAD_SD
## (Intercept) 70.4    8.3
## mom_age      0.3    0.4
## mom_hs      11.3    2.5
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 19.9    0.7
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
ggplot(kidiq,aes(mom_age,kid_score,colour=mom_hs))+geom_point(shape=Kidiq$mom_hs)+geom_abline(i
ntercept=70.3,slope=0.3)+geom_abline(intercept=81.6,slope=0.3)
```



```
#y=70.3+0.3age+11.3hs
```

The intercept represents the predicted test scores for children whose mom didn’t complete high school and give birth at 0 year-old. But it’s not meaningful The coef of mom_hs means the difference between the predicted test scores for children whose mothers did not complete high school and give birth at 0, and children whose mothers did complete high school andgive birth at 0. The coefficient of mom_age can be thought of as the comparison of mean test scores across children whose mothers did not complete high school, but whose mothers differ by 1 year in give birth age. T I think the recommend give birth age should be discussed by group. Whether mother complete high school will impact the best age.

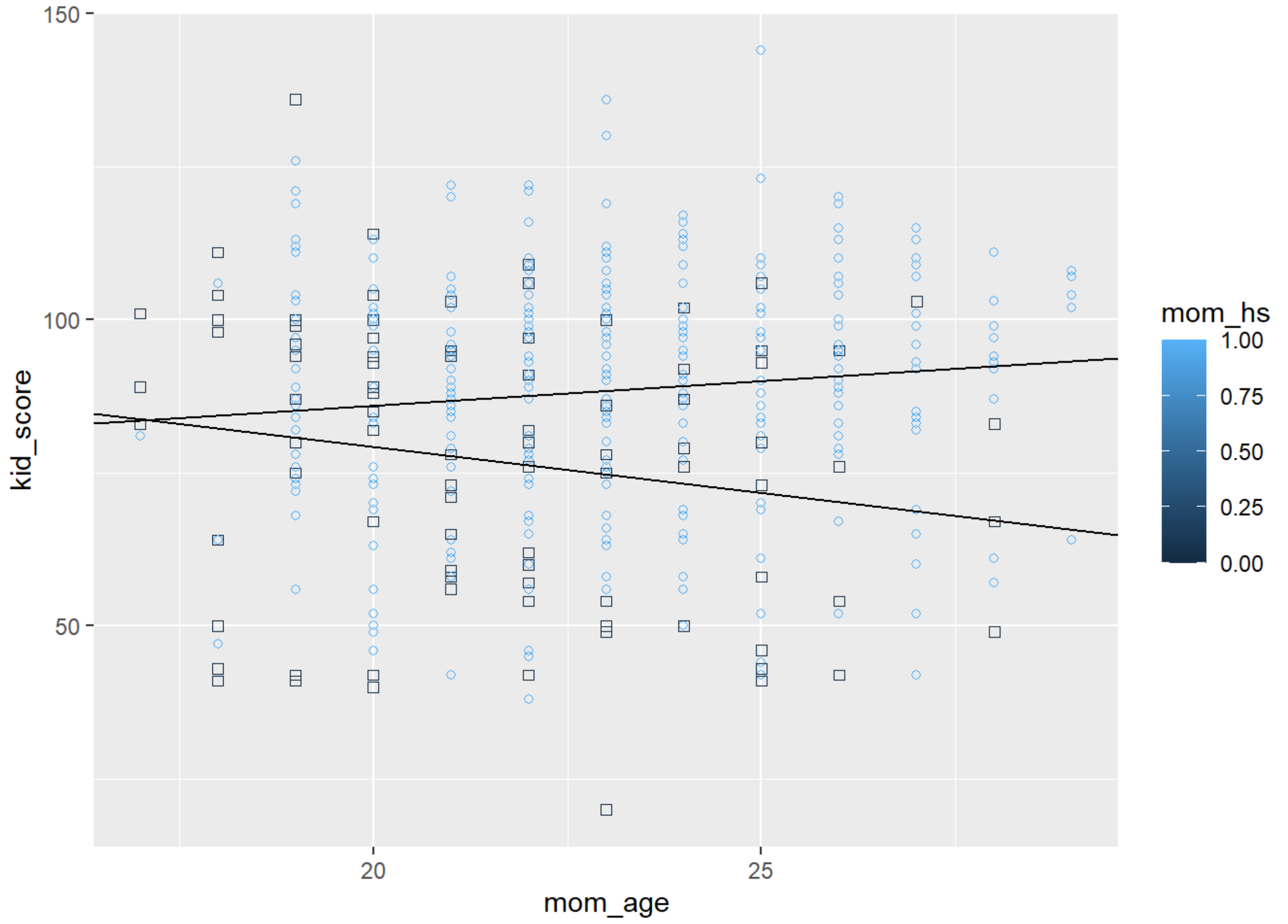
10.5c

Now create an indicator variable reflecting whether the mother has completed high school or not. Consider interactions between high school completion and mother???s age. Also create a plot that shows the separate regression lines for each high school completion status group.

```
fit=stan_glm(kid_score~mom_age+mom_hs+mom_age:mom_hs,data=Kidiq,refresh=0)
print(fit)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      kid_score ~ mom_age + mom_hs + mom_age:mom_hs
##  observations: 434
##  predictors:   4
## ------
##                Median MAD_SD
## (Intercept)    109.3   16.6 
## mom_age         -1.5    0.8 
## mom_hs         -39.6   17.9 
## mom_age:mom_hs   2.3    0.8 
## 
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 19.7    0.7  
## 
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

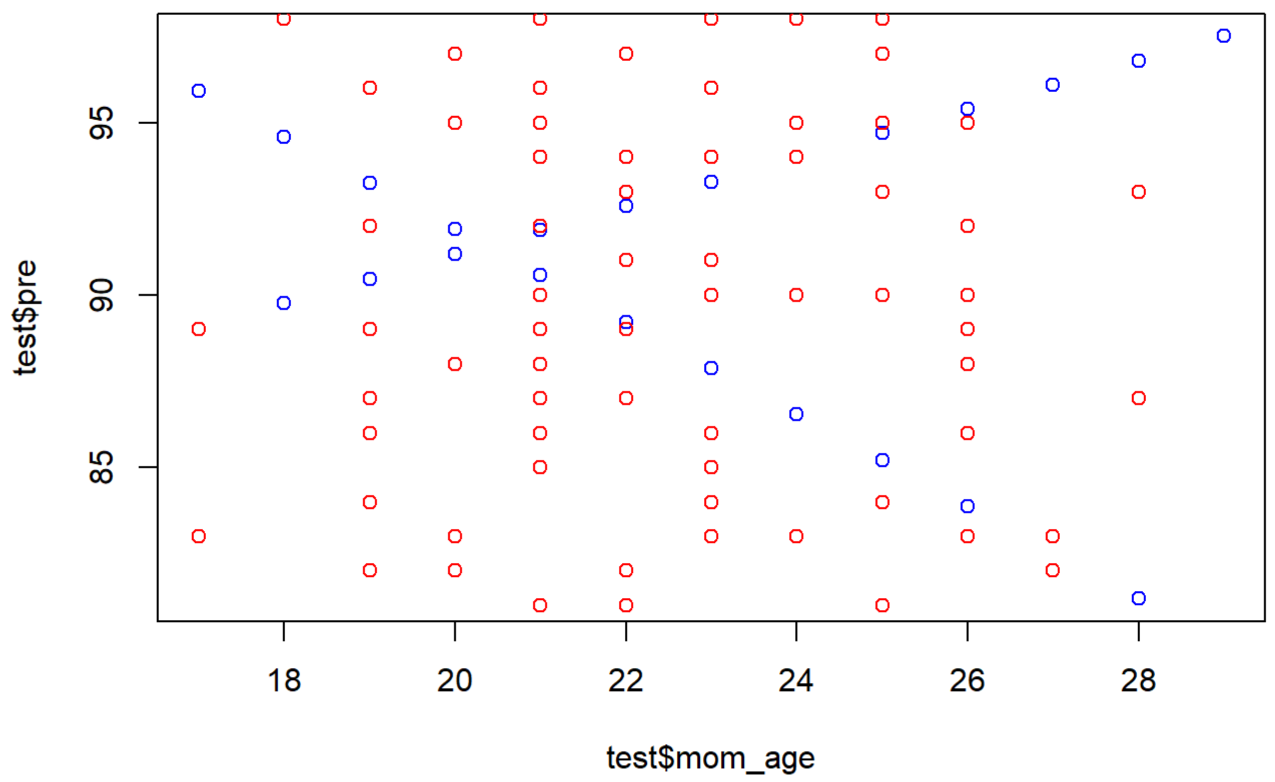
```
ggplot(kidiq,aes(mom_age,kid_score,colour=mom_hs))+geom_point(shape=Kidiq$mom_hs)+geom_abline(i
ntercept=109.3,slope=-1.5)+geom_abline(intercept=70,slope=0.8)
```



10.5d

Finally, fit a regression of child test scores on mother???s age and education level for the first 200 children and use this model to predict test scores for the next 200. Graphically display comparisons of the predicted and actual scores for the final 200 children.

```
train=data.frame(Kidiq[1:200,])
test=data.frame(Kidiq[201:434,])
fit=lm(kid_score~mom_age+mom_hs+mom_age:mom_hs,data=train)
test$pre=predict(fit,test)
plot(test$mom_age,test$pre,col='blue')
points(test$mom_age,test$kid_score,col='red')
```



blue is the predict value and red is the actual value