R and Stats - PDCB topic
Simple linear regression

LCG Leonardo Collado Torres
lcollado@wintergenomics.com – lcollado@ibt.unam.mx

April 27th, 2011

Some theory

lm()

Checking the fitted line

Exercises

Regression

- Why do we want to do a regression?

Simple

- The goal is to describe the behavor of Y variable as a function of X variable.
- The model is *simple*:
- $y_i = \alpha + \beta x_i + \epsilon_i$

Main Assumptions

- ϵ_i independent and $N(0, \sigma^2)$
- *linear* relation between the two variables.

Finding . . .

- the regression coefficient (β): $\hat{\beta} = \frac{\sum(x_i - \overline{x})(y_i - \overline{y})}{\sum(x_i - \overline{x})^2}$
- the intercept (α): $\hat{\alpha} = \overline{y} - \hat{\beta}\overline{x}$
- This is done using the *method of least squares* where the goal is to minimize the sum of squared residuals.

A t-test

- We can test the null hypothesis that $\beta = 0$
- To do is to test that Y is independent of X.
- The test is a t test: $t = \frac{\hat{\beta}}{s.e.(\hat{\beta})}$ with n - 2 degrees of freedom.
- You can also test that $\alpha = 0$ Yet, does it make sense?

lm function

- The main function in R for linear regressions is lm
- Note that it can be used in a wide variety of use cases, but for now we'll only use it to do simple lm.
- The output is simple:

```
> library(ISwR)
> attach(thuesen)
> lm(short.velocity ~ blood.glucose)

Call:
lm(formula = short.velocity ~ blood.glucose)

Coefficients:
  (Intercept)  blood.glucose
      1.09781        0.02196
```

Extract info

- Yet we can extract more information using a variety of functions:

```
> lm1 <- lm(short.velocity ~ blood.glucose)
> class(lm1)

[1] "lm"

> names(lm1)

 [1] "coefficients"  "residuals"
 [3] "effects"       "rank"
 [5] "fitted.values" "assign"
 [7] "qr"            "df.residual"
 [9] "na.action"     "xlevels"
[11] "call"          "terms"
[13] "model"
```

Extract info

```
> summary(lm1)

Call:
lm(formula = short.velocity ~ blood.glucose)

Residuals:
     Min       1Q   Median       3Q
-0.40141 -0.14760 -0.02202  0.03001
     Max
 0.43490

Coefficients:
              Estimate Std. Error
(Intercept)    1.09781    0.11748
blood.glucose  0.02196    0.01045
```

Extract info

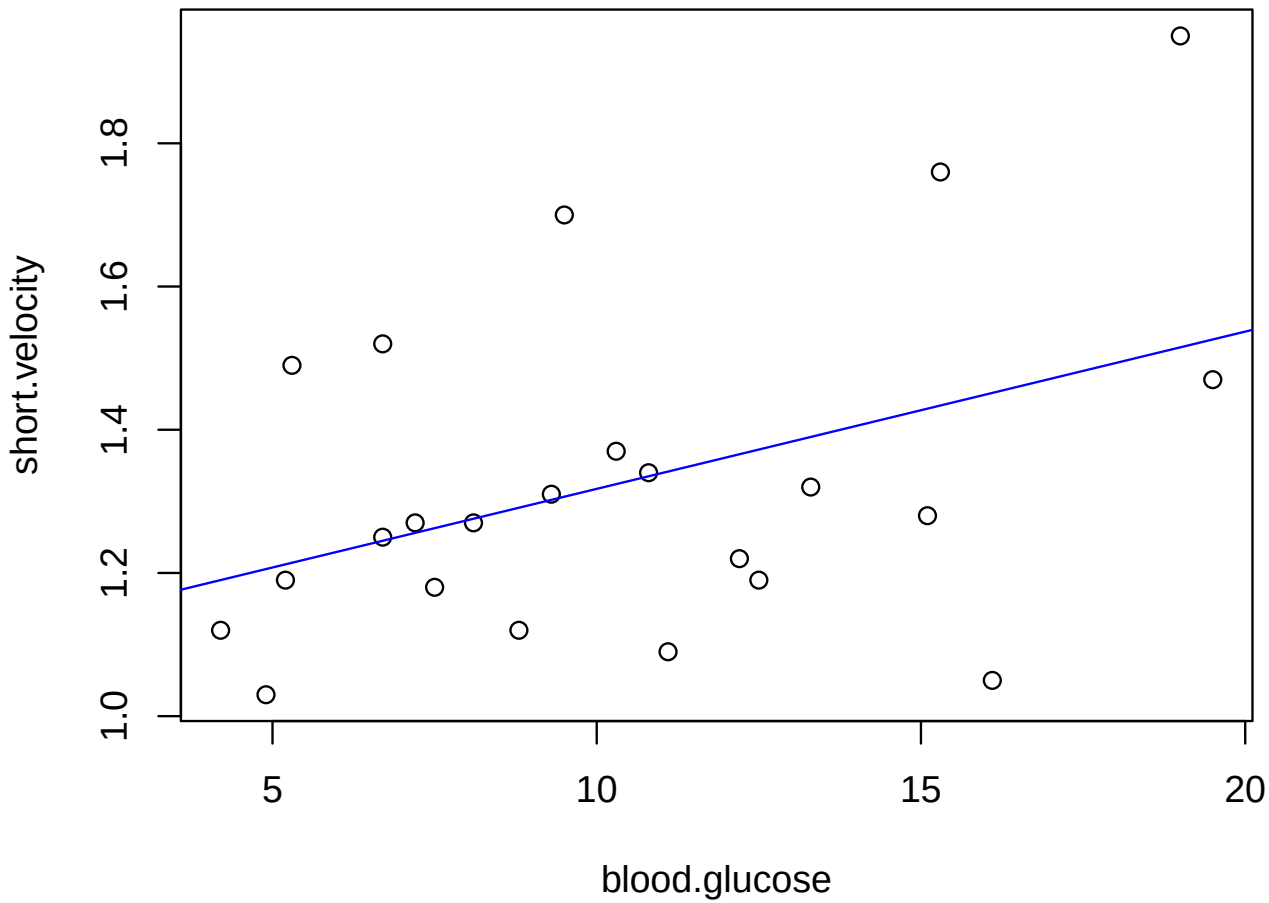
```
                t value Pr(>|t|)
(Intercept)     9.345 6.26e-09 ***
blood.glucose   2.101   0.0479 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '

Residual standard error: 0.2167 on 21 degrees of freedo
  (1 observation deleted due to missingness)
Multiple R-squared: 0.1737, Adjusted R-squared: 0.1343
F-statistic: 4.414 on 1 and 21 DF,  p-value: 0.0479
```

- Explain the output of the summary.

Exploring lm1

```
> plot(blood.glucose, short.velocity)
> abline(lm1, col = "blue")
```

Exploring lm1



Fitted and residuals

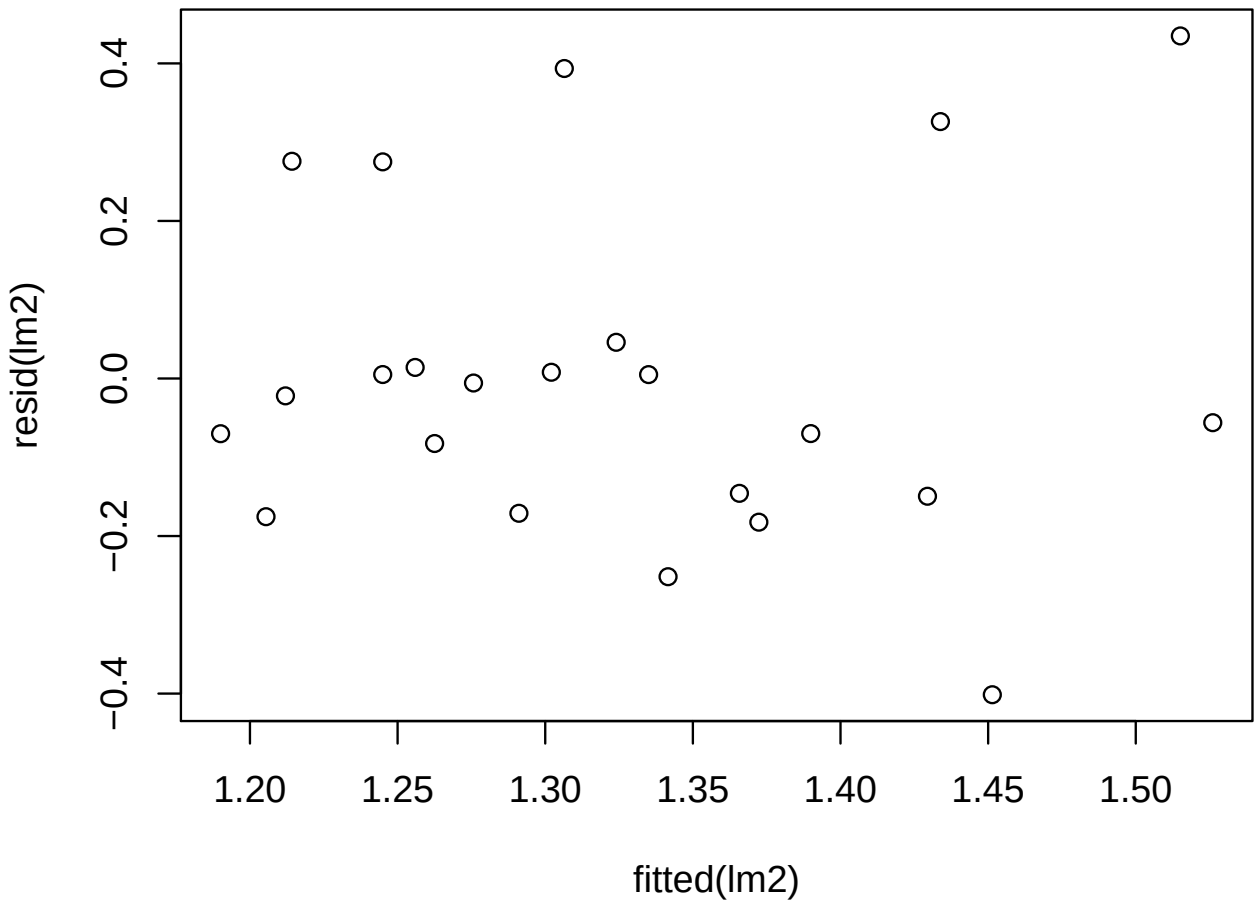
- Interesting pieces of information are the fitted values and the residual values.
- Fitted ones are those values for Y according to our regression.
- Residual values are the difference between the fitted Y values and the real Y values.
- Yet, we first need to remove incomplete cases (rows with NA values):

```
> thu2 <- thuesen[complete.cases(thuesen),
+     ]
> lm2 <- lm(short.velocity ~ blood.glucose,
+     data = thu2)
```

Fitted vs Residual

```
> plot(fitted(lm2), resid(lm2))
```

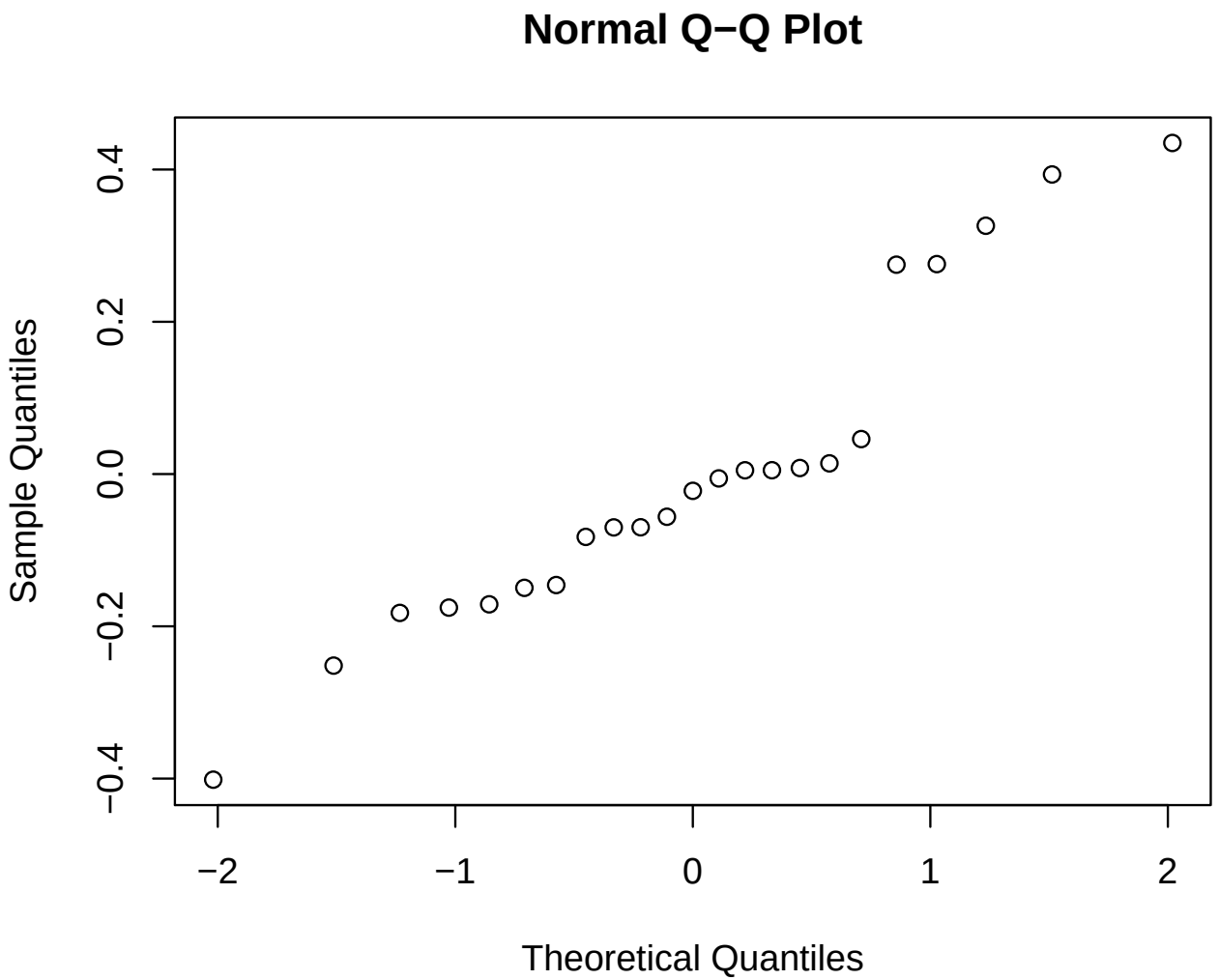
Fitted vs Residual



Verifying the initial assumption

```
> qqnorm(resid(lm2))
```

Verifying the initial assumption



Uncertainity bands

- Confidence bands: reflect the uncertainty about the fitted line itself. Narrow bands are better and this happens with many observations.
- They often are curved at the ends because the fitted line is better determined at the center of the data.
- Prediction bands: include the uncertainty about future observations.
- With more observations, this line approaches the true line +- 2 standard deviations (95% limits).
- With few samples, they show less curvature than confidence bands.
- Note that these limits rely strongly on ϵ_i being $N(0, \sigma^2)$

Getting the bands data

- To get the data for both types of uncertainity bands, we use the function predict:

```
> predict(lm2, interval = "confidence")
```

```
        fit      lwr      upr
1  1.433841 1.291371 1.576312
2  1.335010 1.240589 1.429431
3  1.275711 1.169536 1.381887
4  1.526084 1.306561 1.745607
5  1.255945 1.139367 1.372523
6  1.214216 1.069315 1.359118
7  1.302066 1.205244 1.398889
8  1.341599 1.246317 1.436881
9  1.262534 1.149694 1.375374
```

Getting the bands data

```
10 1.365758 1.263750 1.467765
11 1.244964 1.121641 1.368287
12 1.212020 1.065457 1.358583
13 1.515103 1.305352 1.724854
14 1.429449 1.290217 1.568681
15 1.244964 1.121641 1.368287
17 1.190057 1.026217 1.353898
18 1.324029 1.230050 1.418008
19 1.372346 1.267629 1.477064
20 1.451411 1.295446 1.607377
21 1.389916 1.276444 1.503389
22 1.205431 1.053805 1.357057
23 1.291085 1.191084 1.391086
24 1.306459 1.210592 1.402326
```

Getting the bands data

```
> predict(lm2, interval = "prediction")
```

	fit	lwr	upr
1	1.433841	0.9612137	1.906469
2	1.335010	0.8745815	1.795439
3	1.275711	0.8127292	1.738693
4	1.526084	1.0248161	2.027352
5	1.255945	0.7904672	1.721423
6	1.214216	0.7408499	1.687583
7	1.302066	0.8411393	1.762993
8	1.341599	0.8809929	1.802205
9	1.262534	0.7979780	1.727090
10	1.365758	0.9037136	1.827802
11	1.244964	0.7777510	1.712177
12	1.212020	0.7381424	1.685898

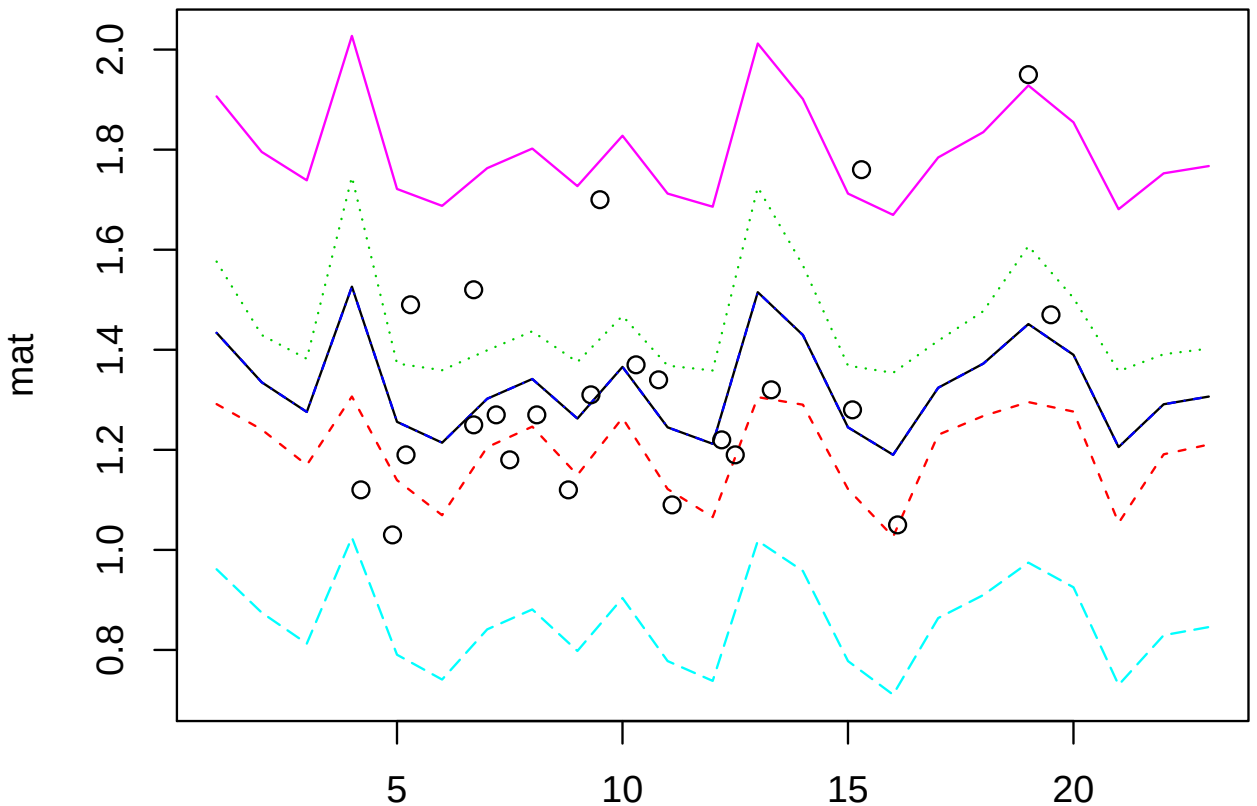
Getting the bands data

```
13 1.515103 1.0180367 2.012169
14 1.429449 0.9577873 1.901111
15 1.244964 0.7777510 1.712177
17 1.190057 0.7105546 1.669560
18 1.324029 0.8636906 1.784367
19 1.372346 0.9096964 1.834996
20 1.451411 0.9745421 1.928281
21 1.389916 0.9252067 1.854626
22 1.205431 0.7299634 1.680899
23 1.291085 0.8294798 1.752690
24 1.306459 0.8457315 1.767186
```

What is the problem?

```
> mat1 <- predict(lm2, int = "c")
> mat2 <- predict(lm2, int = "p")
> mat <- cbind(mat1, mat2)
> matplot(mat, type = "l")
> points(blood.glucose, short.velocity)
```

What is the problem?



Newdata

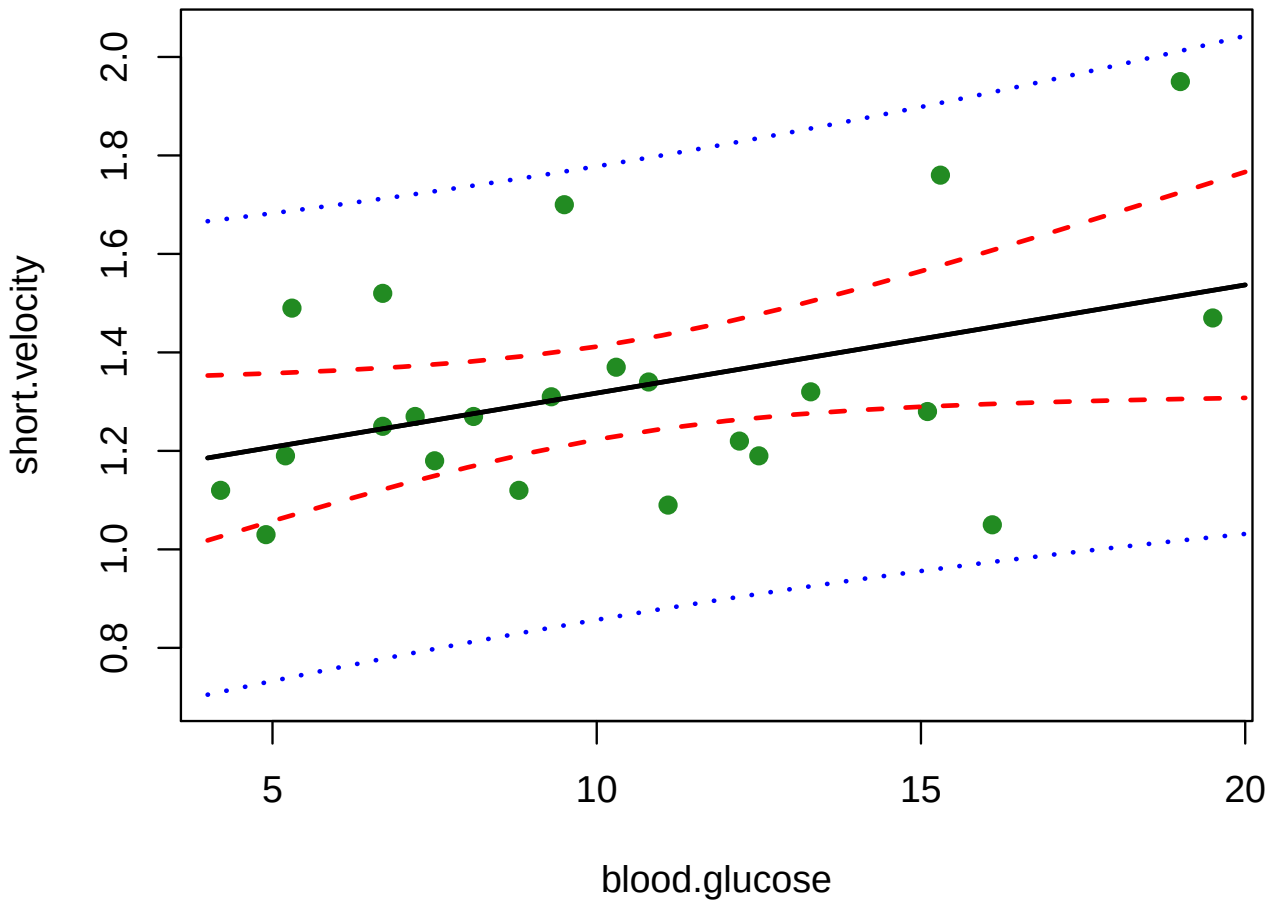
- To exaluate the line correctly, we have to do so with new values of X:

```
> new <- data.frame(blood.glucose = 4:20)
> pp <- predict(lm2, int = "p", newdata = new)
> pc <- predict(lm2, int = "c", newdata = new)
```

Nice little plot!

```
> plot(blood.glucose, short.velocity,
+     ylim = range(short.velocity,
+         pp, na.rm = T), pch = 19,
+     col = "forest green")
> pred.gluc <- new$blood.glucose
> matlines(pred.gluc, pc, lty = c(1,
+     2, 2), col = c("black", "red",
+     "red"), lwd = 2)
> matlines(pred.gluc, pp, lty = c(1,
+     3, 3), col = c("black", "blue",
+     "blue"), lwd = 2)
```

Nice little plot!



One

- With the `rmr` data set, plot metabolic rate versus body weight. Fit a linear regression model to the relation. According to the fitted model, what is the predicted metabolic rate for a body weight of 70 kg? Give a 95% confidence interval for the slope of the line.
- For the confidence interval, you'll need to use a new function. Look for it with `apropos` :)

```
> library(ISwR)
> fit <- lm(metabolic.rate ~ body.weight,
+     data = rmr)
> summary(fit)
```

One

```
Call:
lm(formula = metabolic.rate ~ body.weight, data = rmr)

Residuals:
    Min      1Q  Median      3Q     Max
-245.74 -113.99  -32.05  104.96  484.81

Coefficients:
             Estimate Std. Error t value
(Intercept) 811.2267    76.9755  10.539
body.weight   7.0595     0.9776   7.221
             Pr(>|t|)
(Intercept) 2.29e-13 ***
body.weight 7.03e-09 ***
```

One

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '

Residual standard error: 157.9 on 42 degrees of freedom
Multiple R-squared: 0.5539, Adjusted R-squared: 0.5433
F-statistic: 52.15 on 1 and 42 DF,  p-value: 7.025e-09
```

```
> 811.2267 + 7.0595 * 70
```

```
[1] 1305.392
```

```
> predict(fit, newdata = data.frame(body.weight = 70))
```

```
       1
1305.394
```

```
> qt(0.975, 42)
```

One

```
[1] 2.018082
> 7.0595 + c(1, -1) * 2.018 * 0.9776
[1] 9.032297 5.086703
> confint(fit)
                  2.5 %   97.5 %
(Intercept) 655.883819 966.5695
body.weight   5.086656   9.0324
```

Two

- In the `juul` data set, fit a linear regresion model for the square root of the IGF-1 concentration versus age to the group of subjects over 25 years old.

```
> summary(lm(sqrt(igf1) ~ age, data = juul,
+     subset = age > 25))

Call:
lm(formula = sqrt(igf1) ~ age, data = juul, subset = ag

Residuals:
    Min      1Q  Median      3Q     Max
-4.8642 -1.1661  0.1018  0.9450  4.1136

Coefficients:
```

Two

```
             Estimate Std. Error t value
(Intercept) 18.71025    0.49462  37.828
age         -0.10533    0.01072  -9.829
             Pr(>|t|)
(Intercept)   <2e-16 ***
age           <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '

Residual standard error: 1.741 on 120 degrees of freedo
  (9 observations deleted due to missingness)
Multiple R-squared: 0.446, Adjusted R-squared: 0.4414
F-statistic:  96.6 on 1 and 120 DF,  p-value: < 2.2e-16
```

Three

- In the `malaria` data set, analyze the log-transformed antibody level versus age. Make a plot of the relation. Do you notice anything peculiar?

```
> lmal <- lm(log(ab) ~ age, data = malaria)
> summary(lmal)

Call:
lm(formula = log(ab) ~ age, data = malaria)

Residuals:
    Min      1Q  Median      3Q     Max
-4.0753 -1.0622  0.1181  1.1012  2.7335

Coefficients:
```

Three

```
             Estimate Std. Error t value
(Intercept)  3.83697    0.38021  10.092
age          0.10350    0.03954   2.618
             Pr(>|t|)
(Intercept)   <2e-16 ***
age           0.0103 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '

Residual standard error: 1.478 on 98 degrees of freedom
Multiple R-squared: 0.06536, Adjusted R-squared: 0.0558
F-statistic: 6.853 on 1 and 98 DF,  p-value: 0.01025
```

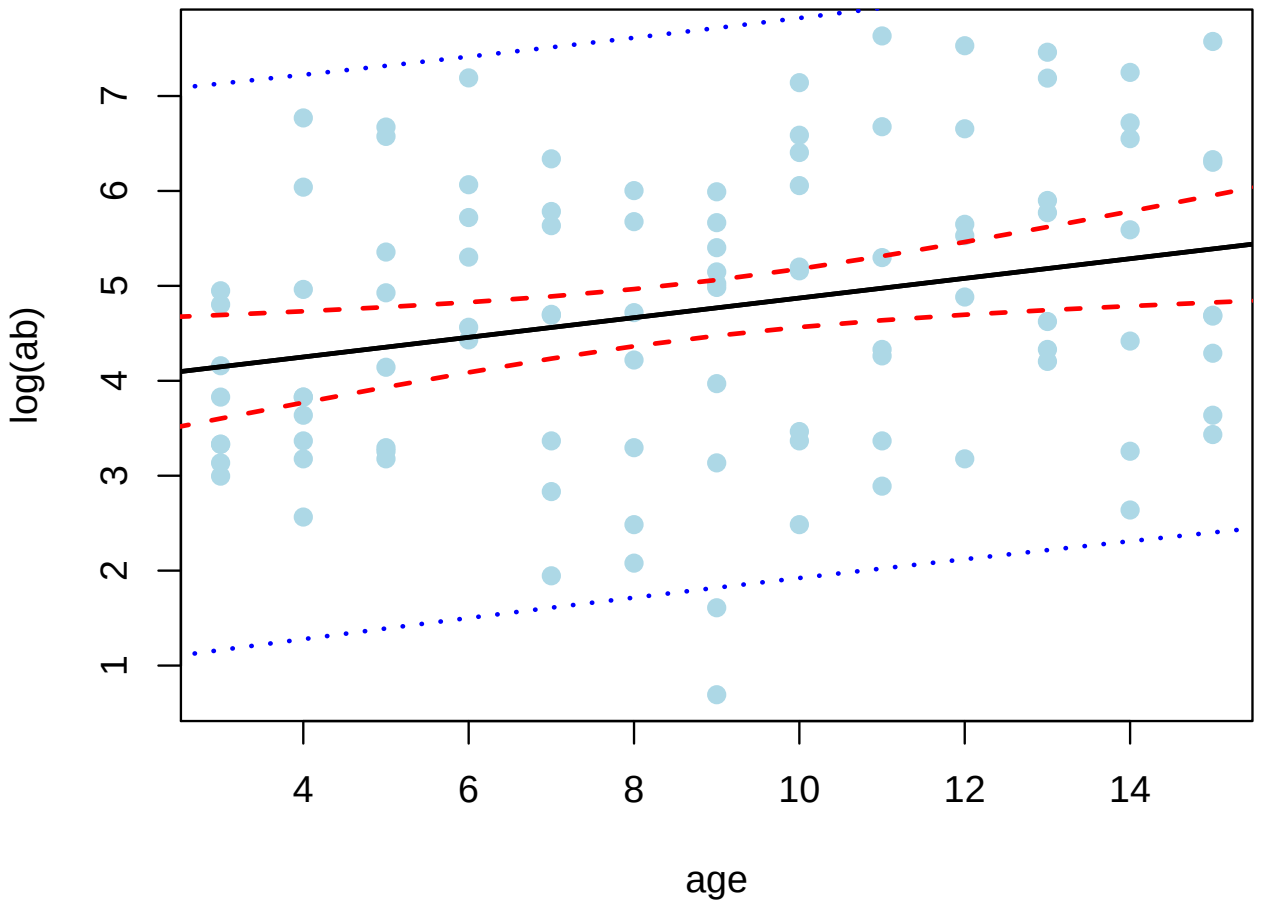
Three

```
> new <- data.frame(age = 2:16)
> pp <- predict(lmal, int = "p",
+     newdata = new)
> pc <- predict(lmal, int = "c",
+     newdata = new)
> pred <- new$age
> plot(log(ab) ~ age, data = malaria,
+     col = "light blue", pch = 19)
> matlines(pred, pc, lty = c(1, 2,
+     2), col = c("black", "red",
+     "red"), lwd = 2)
> matlines(pred, pp, lty = c(1, 3,
```

Three

```
+      3), col = c("black", "blue",
+      "blue"), lwd = 2)
```

Three



Session Information

```
> sessionInfo()

R version 2.12.0 (2010-10-15)
Platform: i386-pc-mingw32/i386 (32-bit)

locale:
[1] LC_COLLATE=English_United States.1252
[2] LC_CTYPE=English_United States.1252
[3] LC_MONETARY=English_United States.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.1252

attached base packages:
[1] stats     graphics  grDevices
[4] utils     datasets  methods
[7] base

other attached packages:
[1] ISwR_2.0-5
```