Testing a linear regression specification against non-parametric alternatives [Demo by Cosma Shalizi]

First case: linear specification is wrong

Make up sample data; it'll simplify things later if they're sorted for plotting

```
x <- sort(runif(300, 0, 3))
```

Impose true regression function $\log(x+1)$, with Gaussian noise:

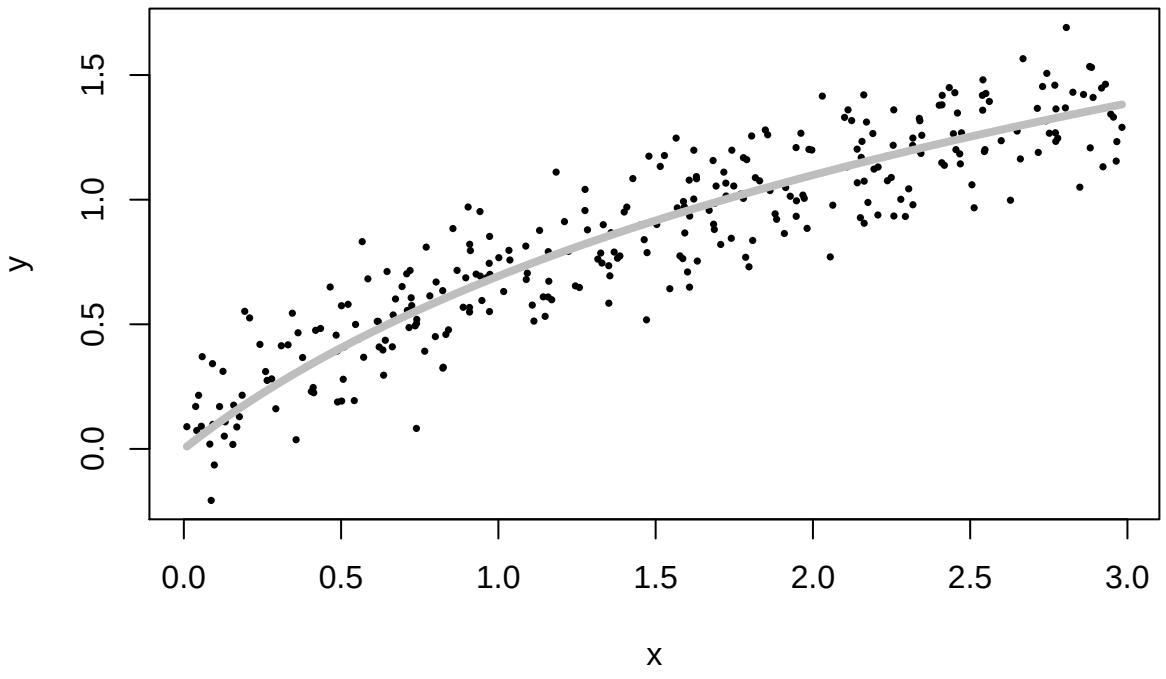
```
yg <- log(x+1) + rnorm(length(x), 0, 0.15)
```

Bind into a data frame:

```
gframe <- data.frame(x=x, y=yg)
```

Plot it, plus the true regression curve

```
plot(y~x, data=gframe, xlab="x", ylab="y", pch=16, cex=0.5)
curve(log(1+x), col="grey", add=TRUE, lwd=4)
```



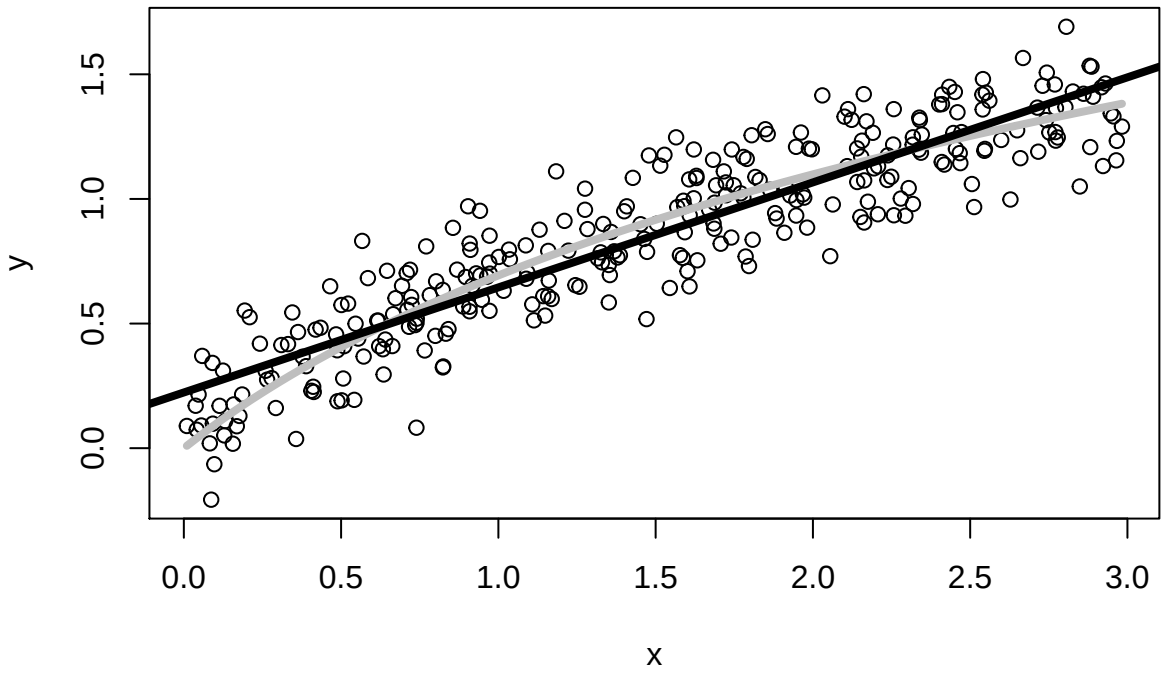
Fit the linear model, add to the plot:

```
glinfit <- lm(y~x, data=gframe)
print(summary(glinfit), signif.stars=FALSE, digits=2)
```

```
##
## Call:
## lm(formula = y ~ x, data = gframe)
##
## Residuals:
##    Min     1Q Median     3Q    Max
```

```
## -0.467 -0.112 -0.004  0.114  0.388
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.223      0.019      12   <2e-16
## x              0.421      0.011      38   <2e-16
##
## Residual standard error: 0.16 on 298 degrees of freedom
## Multiple R-squared:  0.83,   Adjusted R-squared:  0.83
## F-statistic: 1.4e+03 on 1 and 298 DF,  p-value: <2e-16
```

```
plot(x, yg, xlab="x", ylab="y")
curve(log(1+x), col="grey", add=TRUE, lwd=4)
abline(glinfit, lwd=4)
```



MSE of linear model, in-sample: 0.0262.

We'll need to do that a lot, so make it a function:

```
mse.residuals <- function(model) { mean(residuals(model)^2) }
```

- EXERCISE: Write comments giving the inputs and outputs of the function
- EXERCISE: Check that the function works

Fit the non-parametric alternative:

```
library(mgcv)
```

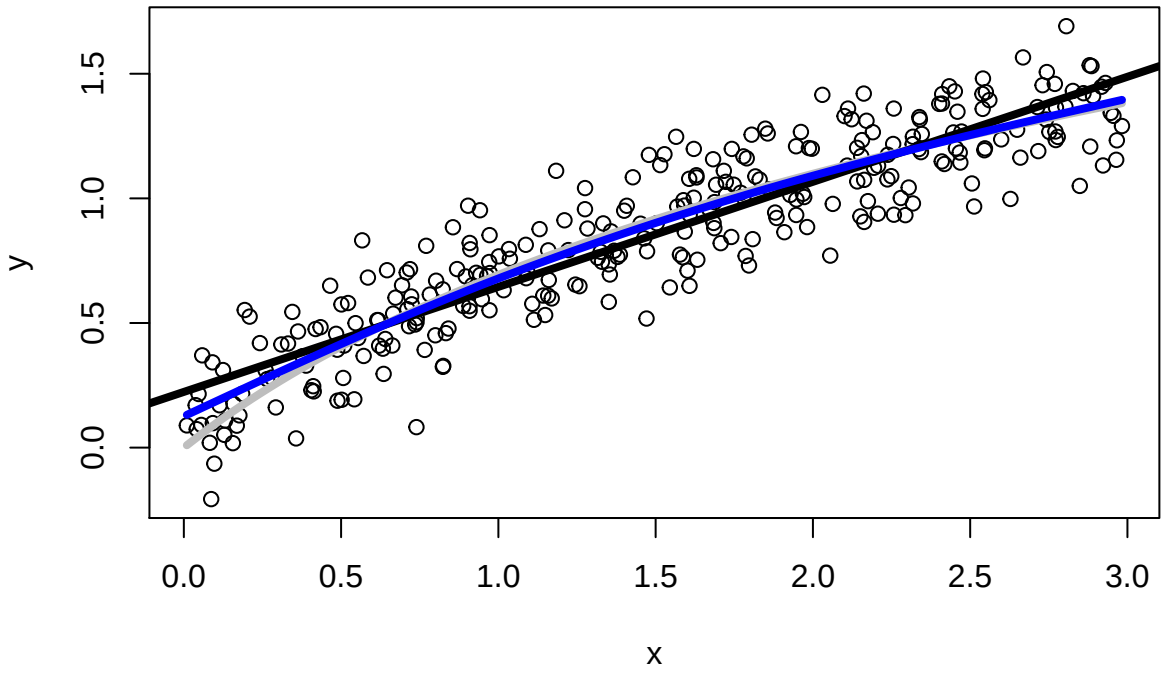
We'll use spline smoothing as provided by the `gam` function:

```
gnpr <- gam(y~s(x),data=gframe)
```

Add the fitted values from the spline to the plot:

```
plot(x,yg,xlab="x",ylab="y")
curve(log(1+x),col="grey",add=TRUE,lwd=4)
abline(glinfit, lwd=4)
```

```
lines(x,fitted(gnpr),col="blue",lwd=4)
```



Calculate the difference in MSEs:

```
t.hat <- mse.residuals(glinfit) - mse.residuals(gnpr)
```

- "t" for test, NOT the t statistic for a Gaussian mean
- EXERCISE: Why do we believe this is the right number?

Simulate from the parametric model, assuming Gaussian noise

```
sim.lm <- function(linfit, test.x) {
  n <- length(test.x)
  sim.frame <- data.frame(x=test.x)
  sigma <- sqrt(mse.residuals(linfit))  # There are other ways to get sigma
  y.sim <- predict(linfit,newdata=sim.frame)
  y.sim <- y.sim + rnorm(n,0,sigma)
  sim.frame <- data.frame(sim.frame,y=y.sim)
  return(sim.frame)
}
```

- EXERCISE: Write comments describing input and output of this function
- EXERCISE: How can we check that this is working?
- EXERCISE: Non-Gaussian noise?

Calculate difference in MSEs between parametric and nonparametric models on a data frame:

```
calc.T <- function(df) {
  MSE.p <- mse.residuals(lm(y~x,data=df))
  MSE.np <- mse.residuals(gam(y~s(x),data=df))
  return(MSE.p - MSE.np)
}
```

- EXERCISE: Comments once again (with feeling this time)
- EXERCISE: Check this function

Calculate the MSE difference on one simulation run:

```
calc.T(sim.lm(glinfit,x))
```

```
## [1] 0.0001046559
```

Calculate the MSE difference on 200 simulation runs, so we get a sample from the null hypothesis:

```
null.samples.T <- replicate(200,calc.T(sim.lm(glinfit,x)))
```

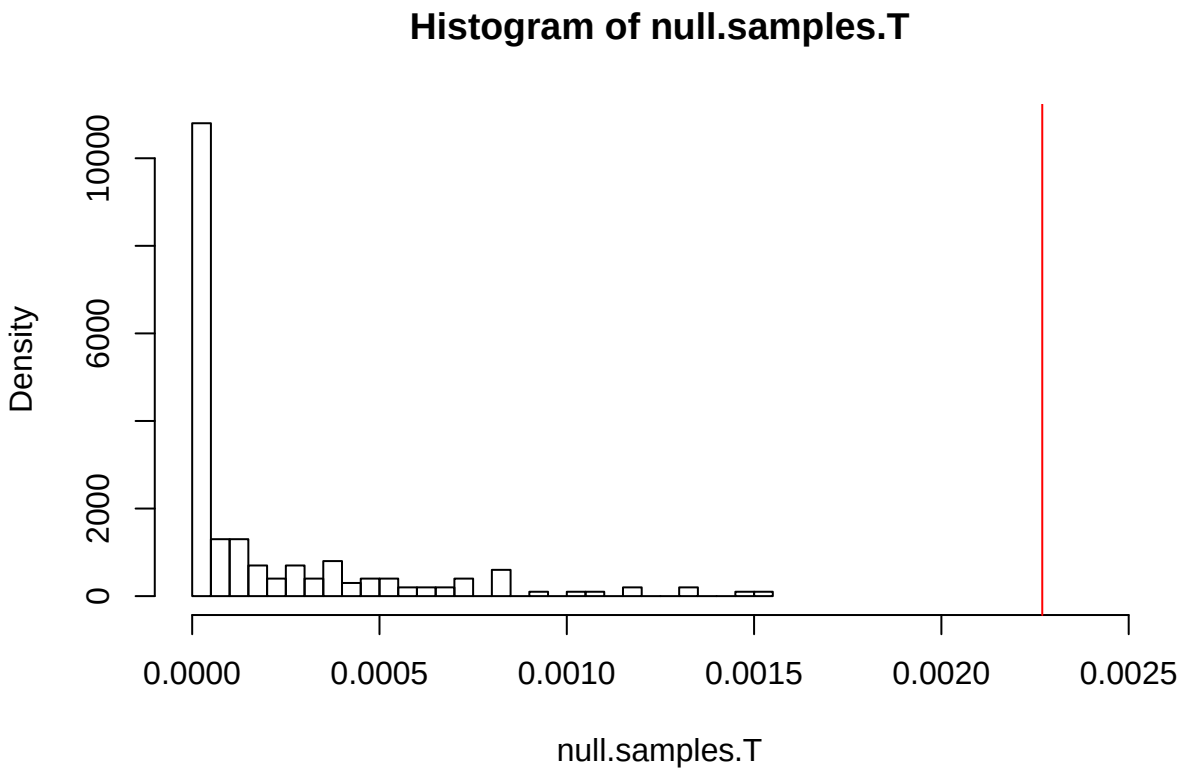
How often does the simulation produce gaps bigger than what we really saw?

```
sum(null.samples.T > t.hat)
```

```
## [1] 0
```

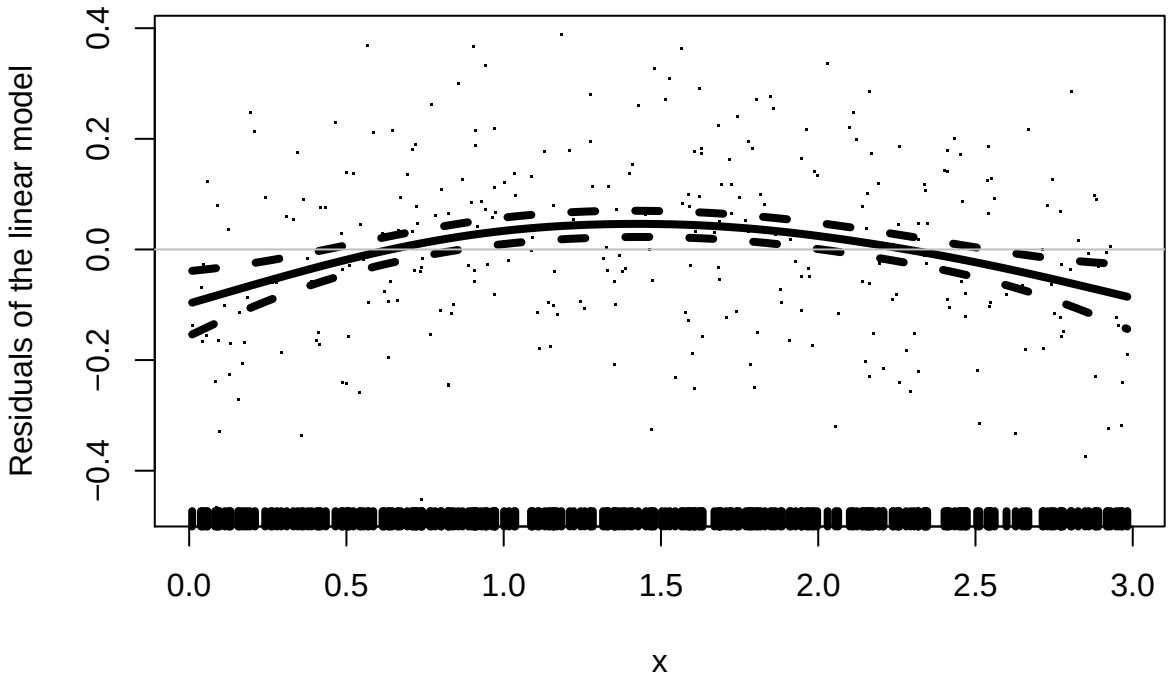
Plot histogram of the sampling distribution, and the observed value:

```
hist(null.samples.T,n=31,xlim=c(min(null.samples.T),1.1*t.hat),probability=TRUE)
abline(v=t.hat, col="red")
```



Another approach: smooth the residuals of the parametric model:

```
plot(gam(residuals(glinfit) ~ s(x)),residuals=TRUE,
  lwd=4,ylab="Residuals of the linear model")
abline(h=0,col="grey")
```

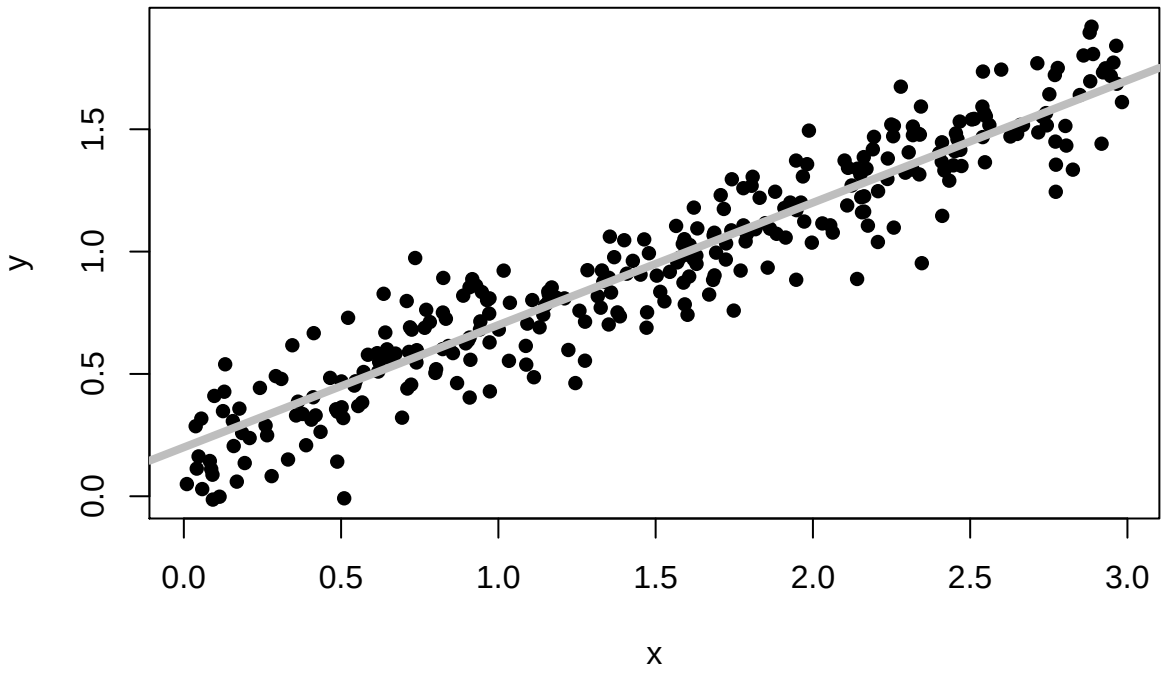


- We'd need to calculate how far that curve is from 0, and then simulate to get a null distribution for that test statistic

Second case: linear model is properly specified

Deliberately left uncommented so that you can explore

```
y2 <- 0.2+0.5*x + rnorm(length(x),0,0.15)
  # Why don't I have to make up x values again?
y2.frame <- data.frame(x=x,y=y2)
plot(x,y2,xlab="x",ylab="y",pch=16)
abline(0.2,0.5,col="grey",lwd=4)
```




```
y2.fit <- lm(y~x,data=y2.frame)
null.samples.T.y2 <- replicate(200,calc.T(sim.lm(y2.fit,x)))
t.hat2 <- calc.T(y2.frame)
hist(null.samples.T.y2,n=31,probability=TRUE)
abline(v=t.hat2, col="red")
```