## some fake regression data

```r
x = rnorm(20)
y = rnorm(20)
plot(x, y)
y.0 = lm(y ~ 1)
abline(h=y.0$coef[1])
d = seq(-2, 2, length=200)
```

## polynomial fits

```r
for(degree in 1:9){
    fm = lm(y ~ poly(x, degree))
    assign(paste("y", degree, sep="."), fm)
    lines(d, predict(fm, data.frame(x=d)), lty=(degree+1))
}
```

## mean squared error in training data

```r
mse = vector(length=10)
for(degree in 0:9){
    fm = get(paste("y", degree, sep="."))
    mse[degree+1]= mean(summary(fm)$residuals^2)
}
```

```r
plot(0:9, mse, type="b", xlab="polynomial degree", ylab="mse")
```

## some new data from the same model

```r
x1 = rnorm(200); y1=rnorm(200)
```

## mean squared error on predictions

```r
mse2 = vector(length=10)
for(degree in 0:9){
    fm = get(paste("y", degree, sep="."))
    mse2[degree+1]= mean((predict(fm, data.frame(x=x1))-y1)^2)
}
```
plot(0:9, mse2, type="b", xlab="poly degree", ylab="mse", log="y", pch=2, lty=2)
points(0:9, mse, type="b")

## comparison of new and old data
## (out of sample error)
plot(x1, y1, pch=2, col="blue")
points(x, y)
d1 = seq(min(x1), max(x1), length=200)
for(degree in 1:9){
    fm = lm(y ~ poly(x, degree))
    assign(paste("y", degree, sep="."), fm)
    lines(d, predict(fm, data.frame(x=d1)), lty=(degree+1))
}

## some of the error is due to extrapolation
## try plotting mse1 and mse2 including only test points
## that fall within the original range of the x's