Mathematical Modeling of Infectious Diseases, UCSF

Lecture 4

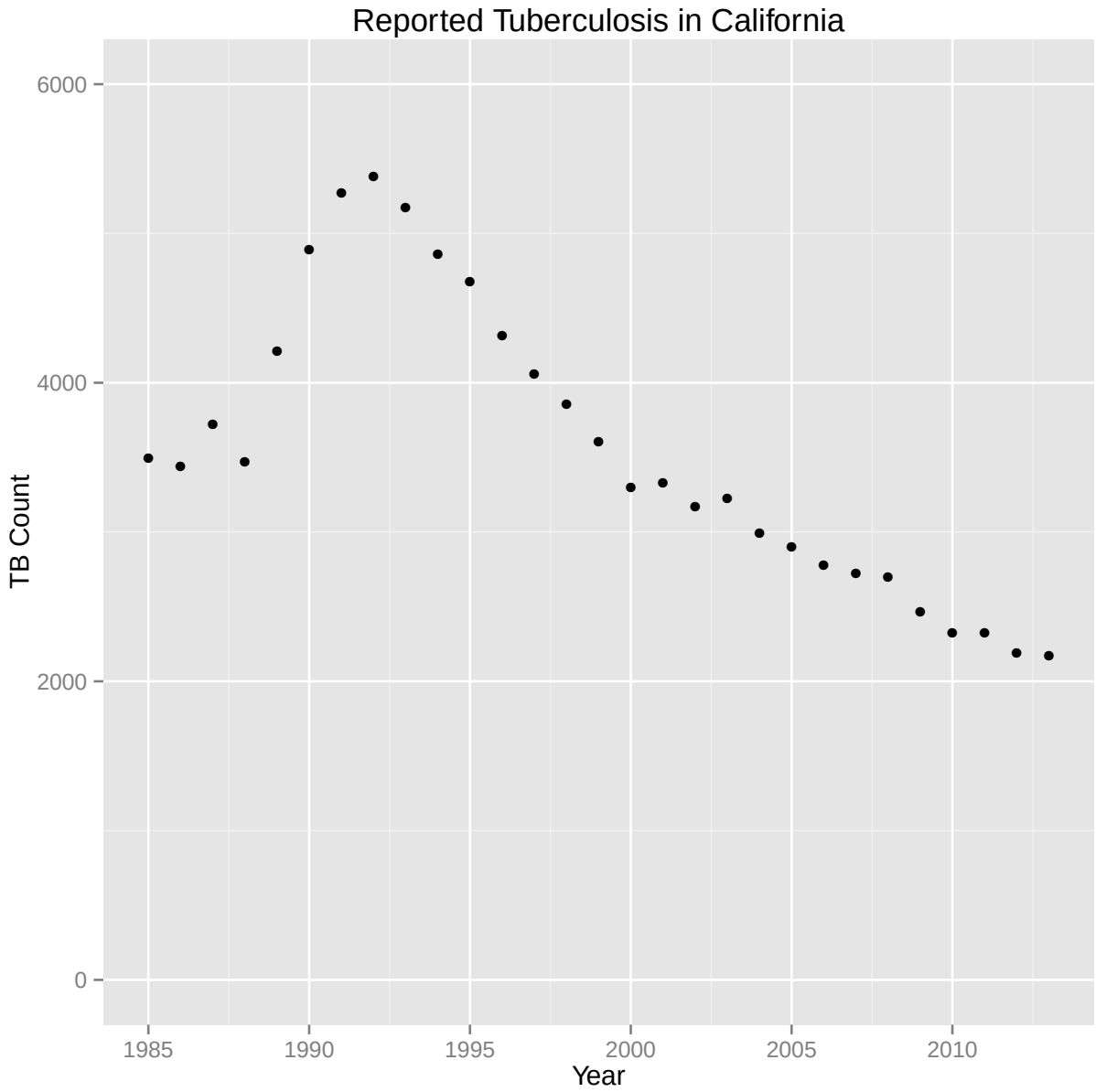
Version 1.0—modified 23 Jan 2015

This week, we're going to take a step back and use some of what we learned to look at data. At the end we'll do a little more review and preparation for next week's model: the Galton-Watson process.

Holt-Winters forecasting

The following data are the reported yearly case counts for Tuberculosis in California since 1985.

```
qplot(tbh[,1],tbh[,2],ylim=c(0,6000),xlab="Year",ylab="TB Count",main="Reported Tuberculosis in Califor
```



We'd like to produce some forecasts for the next year. One idea is that recent events are more important or relevant than things in the far past. Obviously this isn't true in general. It's more a rule of thumb that's often helpful.

Our goal is to look at some simple ways to predict next year's case count, based on the time series.

The first, and most naive thing we could do, would be to just say next year's will be equal to this year's, or to the average of the last few years. Of course that ignores the trend. So we might take a couple of recent

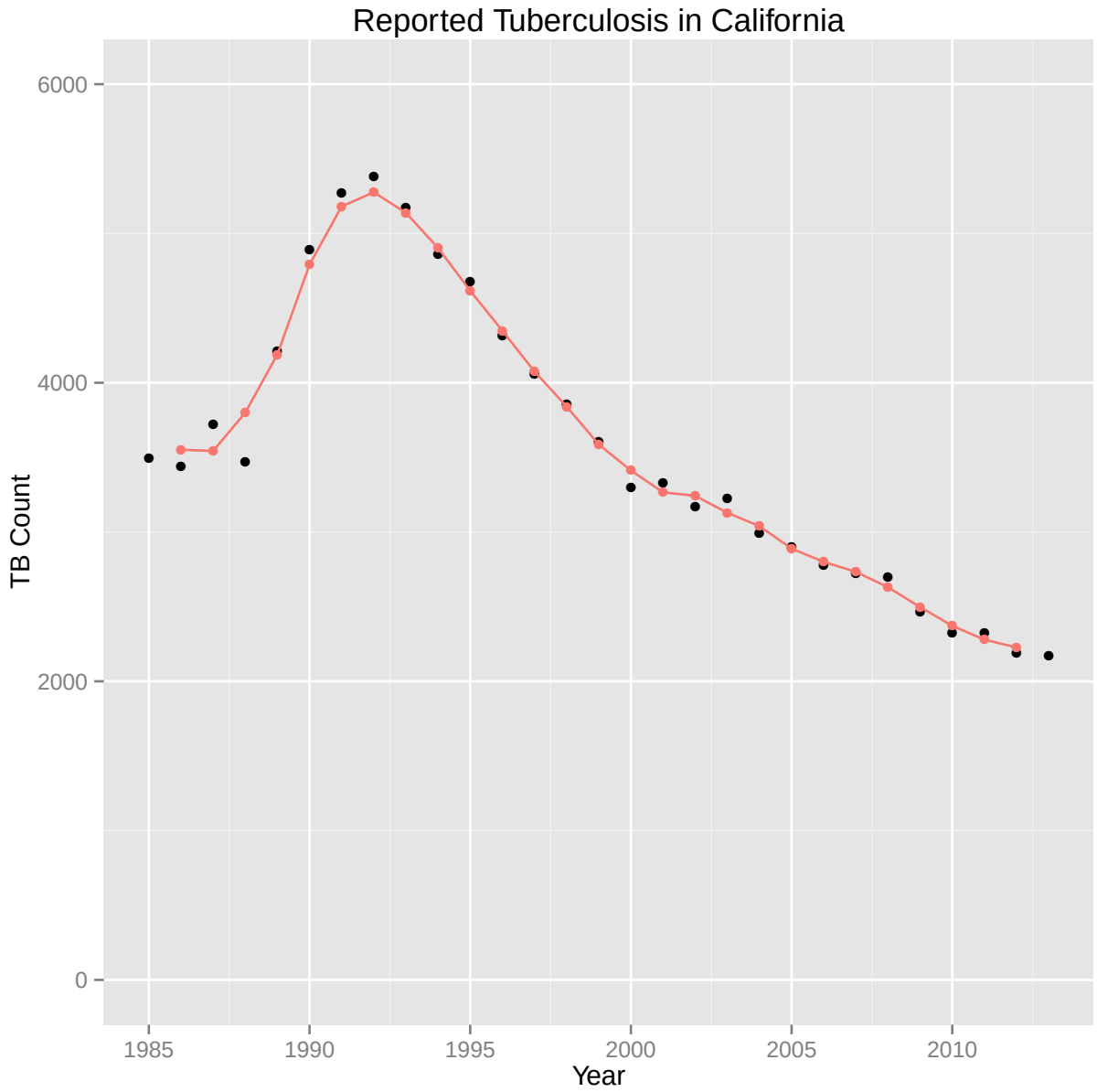
points (how many?) and draw a line thru them, and just extrapolate forward. We might feel this doesn't make good use of the available information.

We could take moving averages; R provides a function called `filter` for this. For a three point centered moving average, we do this. We show the centered moving average in red.

$$y_i = \frac{x_{i-1} + x_i + x_{i+1}}{3}$$

```
qplot(tbh[,1],tbh[,2],ylim=c(0,6000),xlab="Year",ylab="TB Count",main="Reported Tuberculosis in Califor
```

```
## Warning in loop_apply(n, do.ply): Removed 2 rows containing missing values (geom_point).
## Warning in loop_apply(n, do.ply): Removed 2 rows containing missing values (geom_path).
```

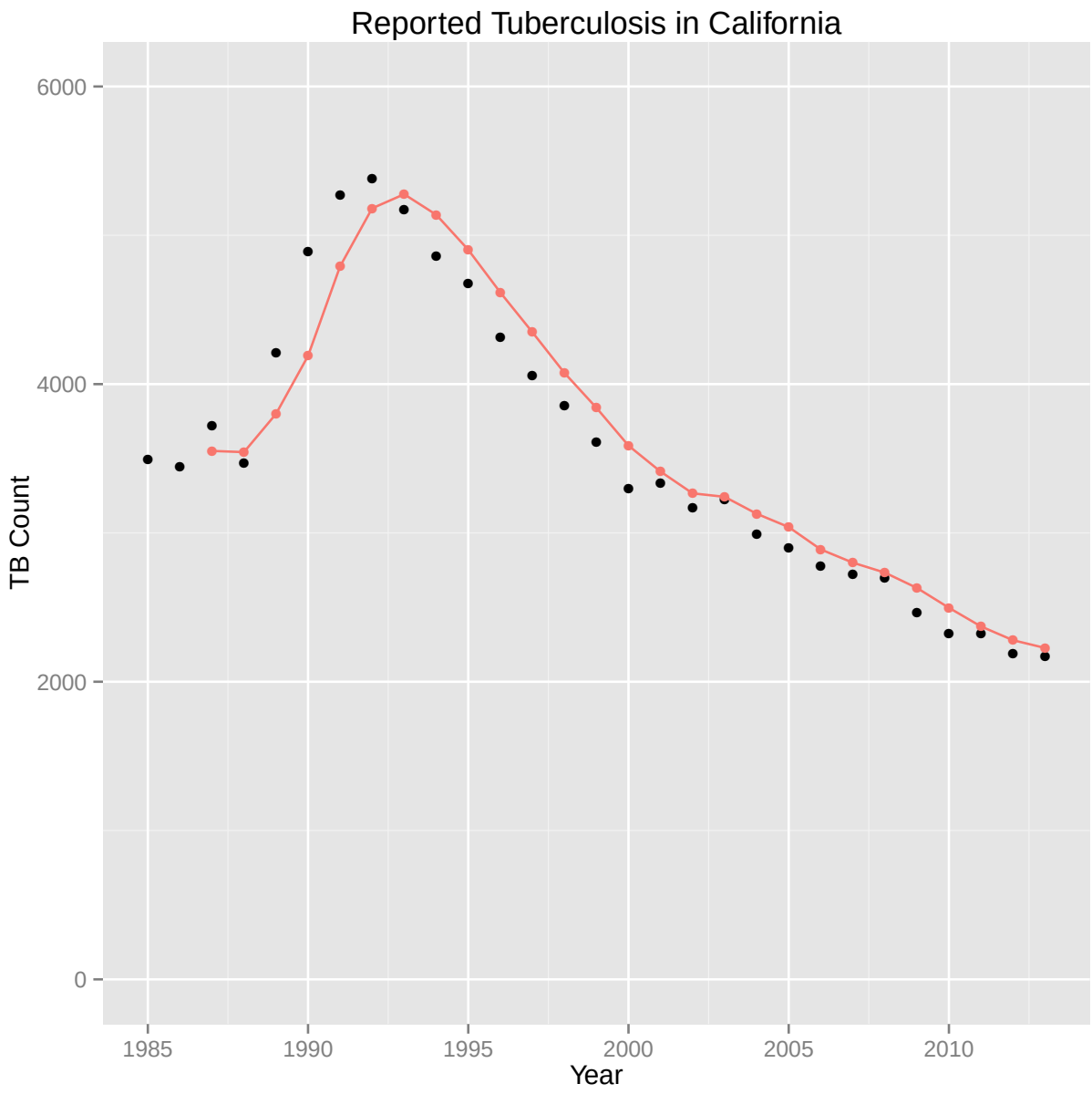


This does smooth out irregularities in the sequence and is useful for some things. But it isn't so great for forecasting since it isn't, strictly speaking, available at the ends. We could work around that.

One idea is to just use the last three.

```
qplot(tbh[,1],tbh[,2],ylim=c(0,6000),xlab="Year",ylab="TB Count",main="Reported Tuberculosis in Califor

## Warning in loop_apply(n, do.ply): Removed 2 rows containing missing values (geom_point).
```

```
## Warning in loop_apply(n, do.ply):  Removed 2 rows containing missing values (geom_path).
```



Now, when it's rising consistently, the data are above their three year moving average; when it's falling, the data are below the moving average.

One idea that tries to use more of the data but still to weight recent data more heavily is called the

exponentially weighted moving average. Remember our geometric series:

$$\sum_{i=0}^{\infty} a^i = \frac{1}{1-a}$$

If we use weights $(1-a)a^i$, they will actually add up to 1 if we had an infinite amount of data going back into past eternity. But in practice we can often go far enough back to where this is so close to 1 the difference is irrelevant to anything. Let $\lambda = 1 - a$.

Then we have a moving average estimate of the current "true value" or level:

$$\mu_\tau = \lambda x_\tau + \lambda(1-\lambda)x_{\tau-1} + \lambda(1-\lambda)^2 x_{\tau-2} + \cdots = \lambda \sum_{i=0}^{T} (1-\lambda)^i x_{\tau-i}.$$

If T is big enough those weights will sum to nearly 1 and be good enough.

One nice thing about this is that we can construct a recursion:

$$\mu_\tau = \lambda x_\tau + \sum_{i=1}^{T} (1-\lambda)^i x_{\tau-i} = \lambda x_\tau + (1-\lambda) \sum_{j=0}^{j=T-1} (1-\lambda)^j x_{\tau-1-j} = \lambda x_\tau + (1-\lambda)\mu_{\tau-1}.$$

with $\mu_0 = 0$ for example.

In practice, what is usually done is to set

$$\mu_1 = x_1.$$

Then

$$\mu_\tau = \lambda x_\tau + (1-\lambda)\mu_{\tau-1}$$

for $\tau = 2, 3, \ldots$. It turns out if you do this, your coefficients for any μ always add up *exactly* to 1, not approximately. I am following the discussion in Harvey.

If you imagine that m_τ is the forecast value for x_τ given the series so far, we could write this as $\hat{x}_{\tau+1|\tau}$:

$$\hat{x}_{\tau+1|\tau} = \lambda x_\tau + (1-\lambda)\hat{x}_{\tau|\tau-1}.$$

Each time, we update the forecast by taking a weighted average of the last forecast and the new data.

The problem with this idea, though, is it assumes a horizontal trend. It doesn't extrapolate a trend forward!

Suppose we extend the simple method above by having an estimated level μ_τ and an estimated slope b_τ at each time. The idea is to forecast the future as a linear trend:

$$\hat{x}_{\tau+k|\tau} = \mu_\tau + b_\tau k$$

for $k = 1, 2, \ldots$. We update and get the new estimated level for time τ by taking the last forecast of it (from one step ago) and making a weighted average of this forecast and the new data:

$$\mu_\tau = \lambda_0 x_t + (1-\lambda_0)\hat{x}_{\tau|\tau-1}.$$

Here,

$$\hat{x}_{\tau|\tau-1} = \mu_{\tau-1} + b_{\tau-1}.$$

To update the slope, take the past estimate $b_{\tau-1}$ and make a different weighted average of this slope with the change in the level:

$$b_\tau = \lambda_1(\mu_\tau - \mu_{\tau-1}) + (1-\lambda_1)b_{\tau-1}.$$

Traditionally, we take $\mu_2 = x_2$, $b_2 = x_2 - x_1$. Here λ_0 and λ_1 are smoothing constants. This is known as the *Holt-Winters* procedure (discussion follows that in Harvey). Thus, the Holt-Winters forecast method is based on simple linear recursions for the mean and slope.

Exercise 1. Let $\hat{e}_\tau$ be the one-step forecast error:

$$\hat{e}_\tau = x_\tau - \hat{x}_{\tau|\tau-1}.$$

Show the forecast can be conducted using these equations:

$$\mu_\tau = \mu_{\tau-1} + b_{\tau-1} + \lambda_0\hat{e}_\tau$$

$$b_\tau = b_{\tau-1} + \lambda_0\lambda_1\hat{e}_\tau$$

∎

Exercise 2. Show that if the data x_t lie along a straight line, the Holt-Winters forecasts will follow the line perfectly. ∎

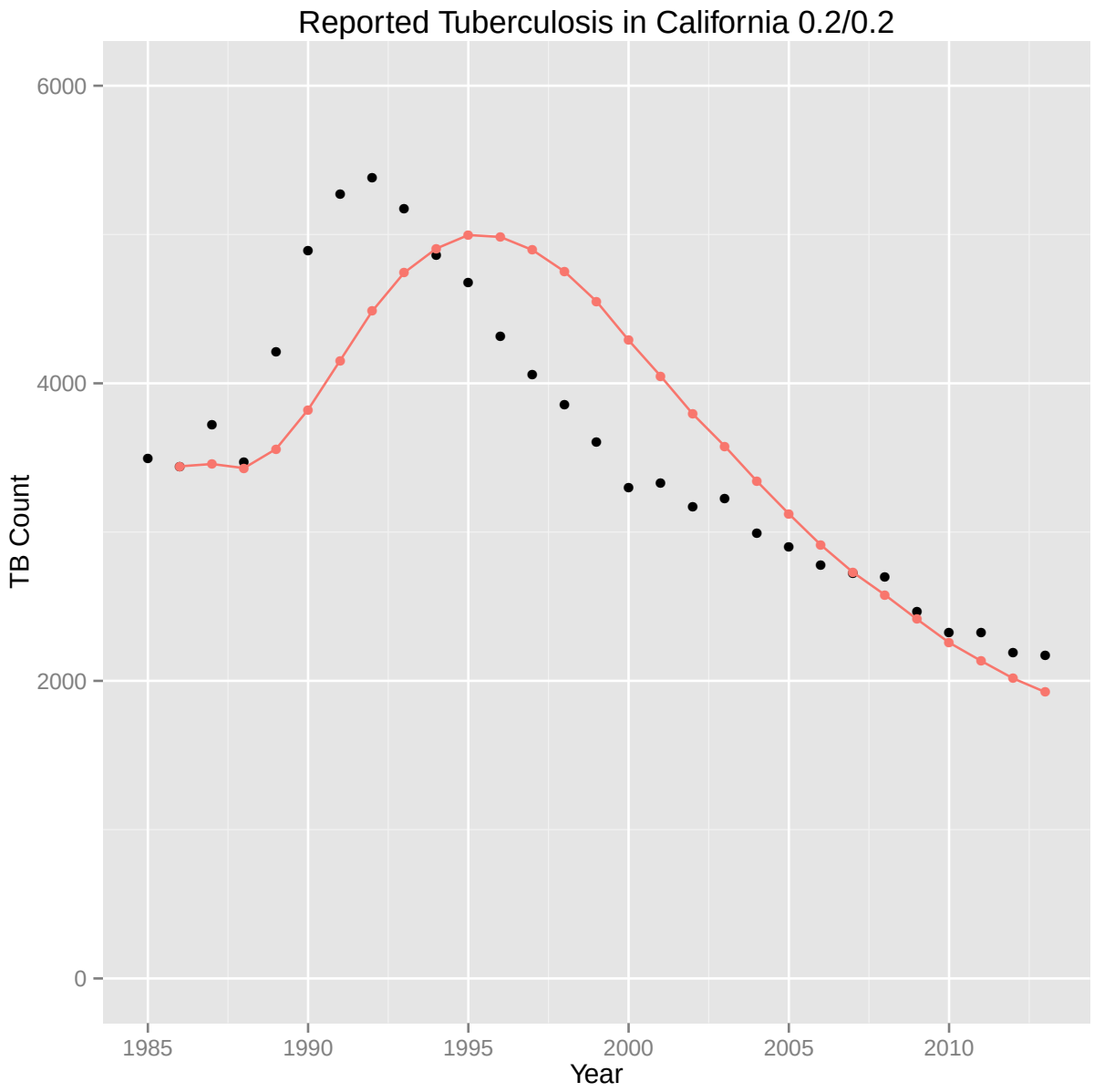
Let's run the Holt-Winters filter on the tuberculosis data and see what we get. R has a builtin function called `HoltWinters` which we want to think about, but let's do it by hand first.

```
hw.filter <- function(xs,lam0,lam1,forecasttime=NA) {
  lx <- length(xs)
  if (lx < 3) {
    stop("xs too short")
  }
  mu <- rep(NA,lx)
  bb <- rep(NA,lx)
  mu[2] <- xs[2]
  bb[2] <- xs[2]-xs[1]
  for (ii in 3:lx) {
```

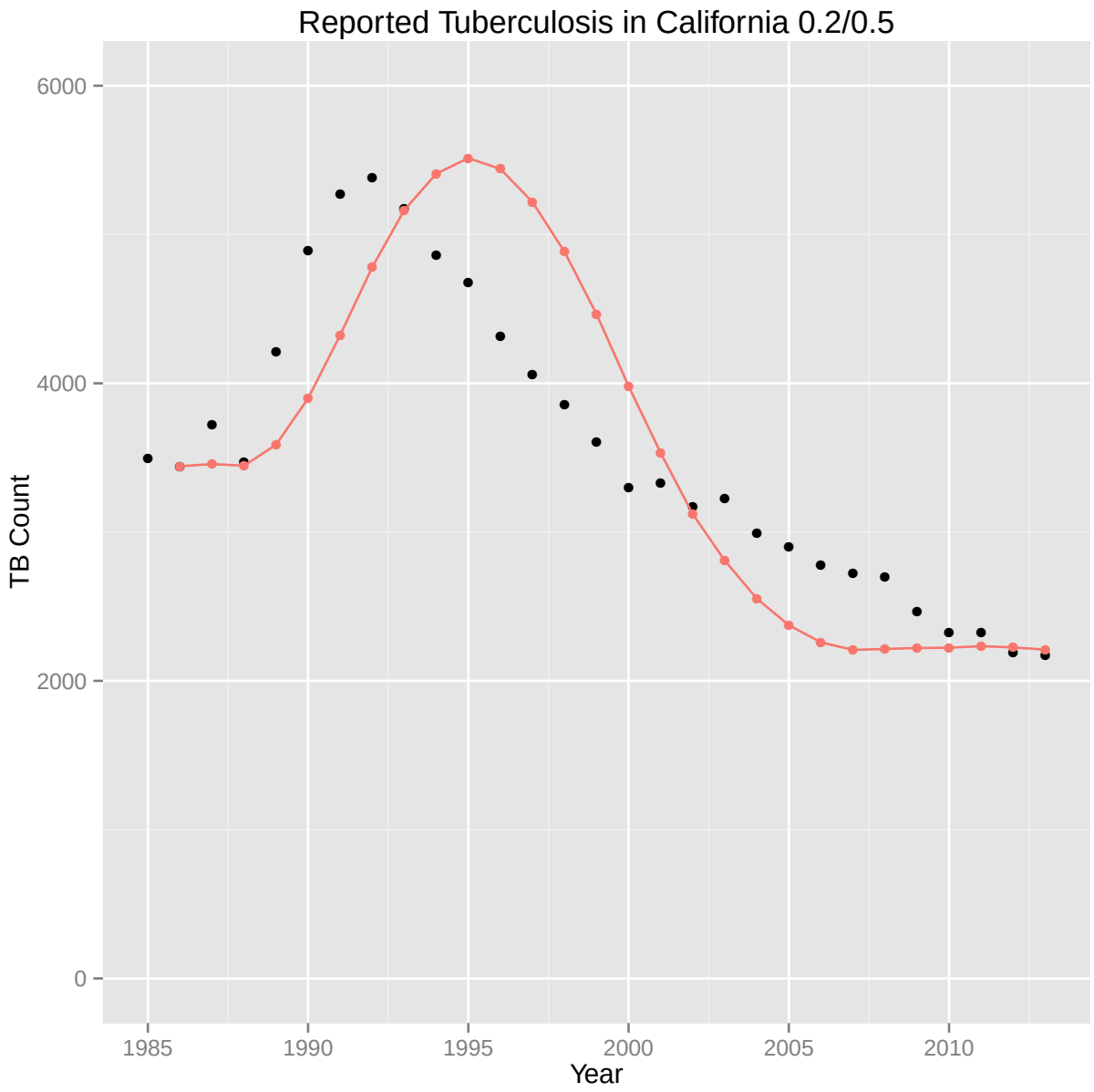
```
    ehat <- xs[ii] - mu[ii-1] - bb[ii-1]
    mu[ii] <- mu[ii-1]+bb[ii-1]+lam0*ehat
    bb[ii] <- bb[ii-1]+lam0*lam1*ehat
  }
  if (is.na(forecasttime)) {
    mu
  } else {
    mu[lx]+forecasttime*bb[lx]
  }
}
```

Let's try a few.

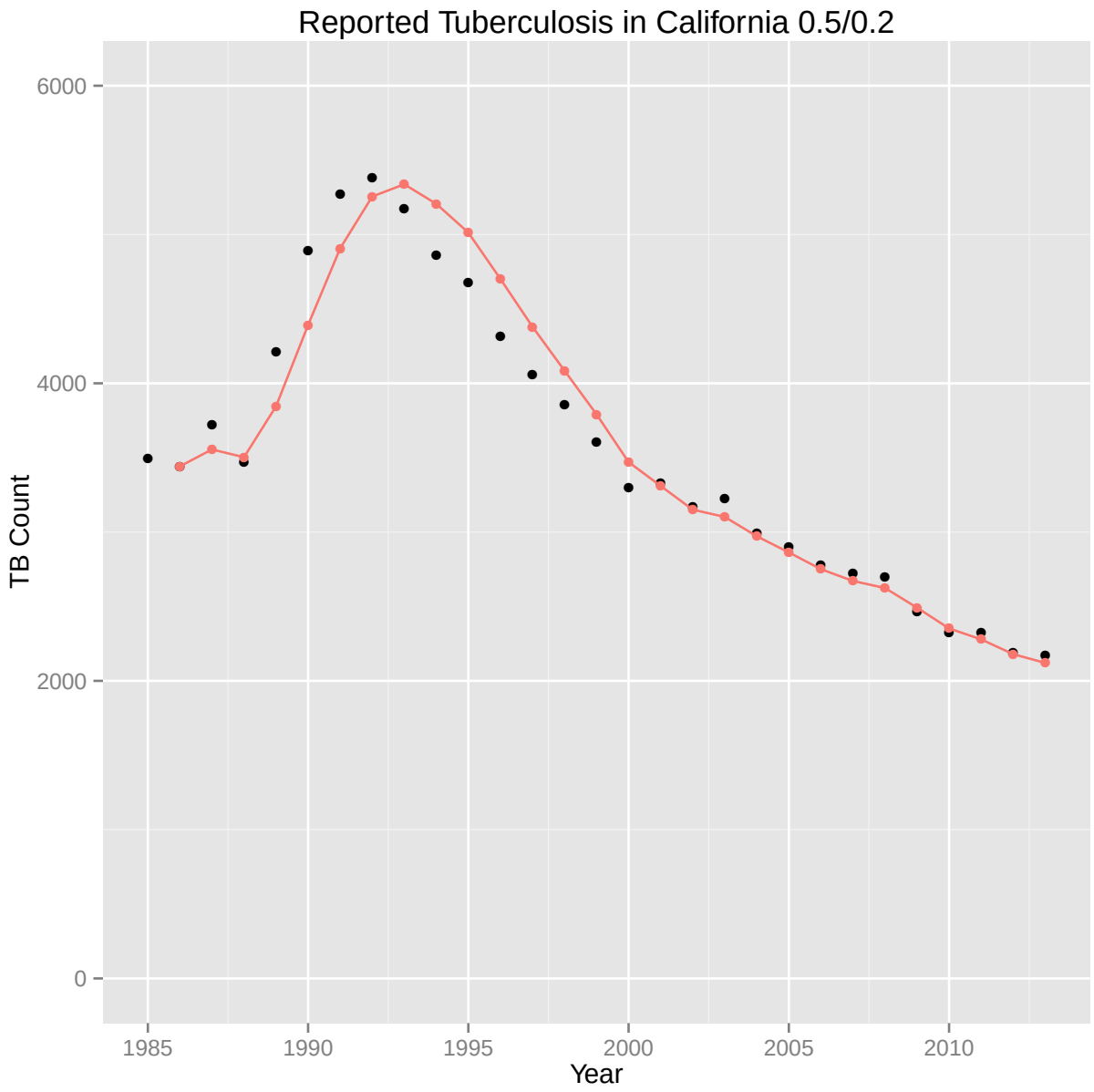
```
tbh$hw22 <- hw.filter(tbh[,2],0.2,0.2)
tbh$hw55 <- hw.filter(tbh[,2],0.5,0.5)
tbh$hw88 <- hw.filter(tbh[,2],0.8,0.8)
tbh$hw25 <- hw.filter(tbh[,2],0.2,0.5)
tbh$hw58 <- hw.filter(tbh[,2],0.5,0.8)
tbh$hw82 <- hw.filter(tbh[,2],0.8,0.2)
tbh$hw28 <- hw.filter(tbh[,2],0.2,0.8)
tbh$hw52 <- hw.filter(tbh[,2],0.5,0.2)
tbh$hw85 <- hw.filter(tbh[,2],0.8,0.5)
qplot(tbh[,1],tbh[,2],ylim=c(0,6000),xlab="Year",ylab="TB Count",main="Reported Tuberculosis in Califor

## Warning in loop_apply(n, do.ply): Removed 1 rows containing missing values (geom_point).
## Warning in loop_apply(n, do.ply): Removed 1 rows containing missing values (geom_path).
```



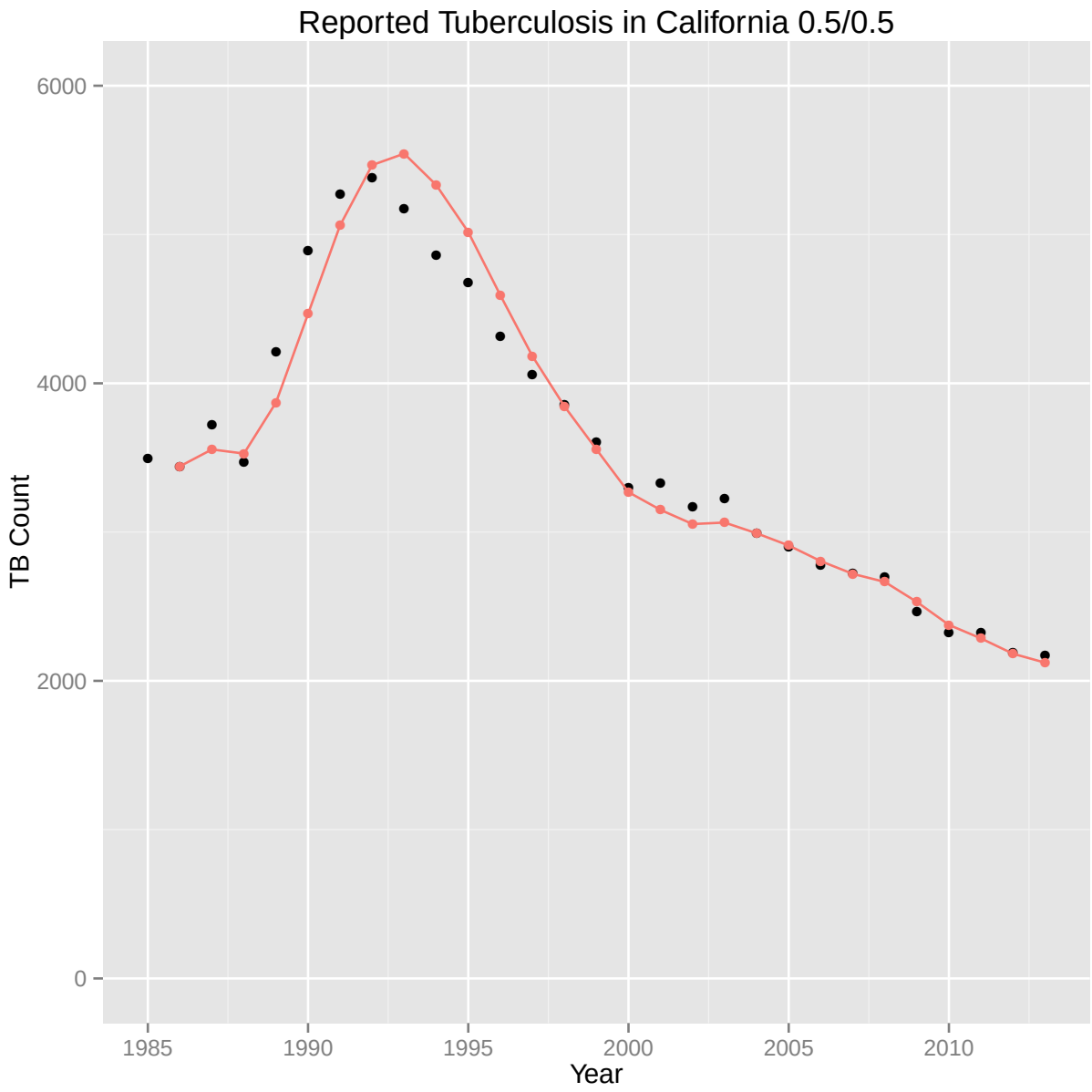

```
qplot(tbh[,1],tbh[,2],ylim=c(0,6000),xlab="Year",ylab="TB Count",main="Reported Tuberculosis in Califor

## Warning in loop_apply(n, do.ply):  Removed 1 rows containing missing values (geom_point).
## Warning in loop_apply(n, do.ply):  Removed 1 rows containing missing values (geom_path).
```




```
qplot(tbh[,1],tbh[,2],ylim=c(0,6000),xlab="Year",ylab="TB Count",main="Reported Tuberculosis in Califor

## Warning in loop_apply(n, do.ply):  Removed 1 rows containing missing values (geom_point).
## Warning in loop_apply(n, do.ply):  Removed 1 rows containing missing values (geom_path).
```



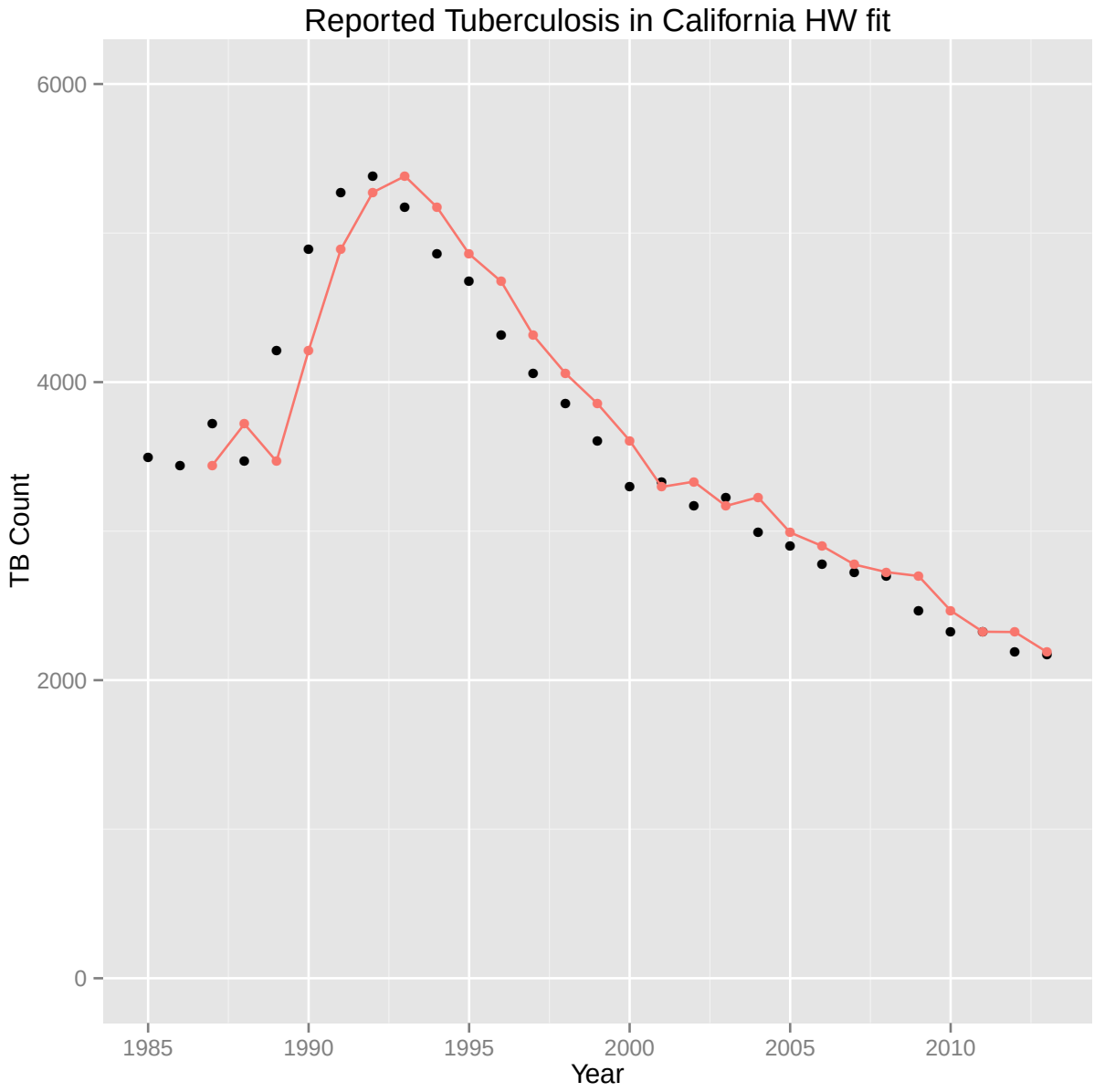

```
qplot(tbh[,1],tbh[,2],ylim=c(0,6000),xlab="Year",ylab="TB Count",main="Reported Tuberculosis in Califor

## Warning in loop_apply(n, do.ply):  Removed 1 rows containing missing values (geom_point).
## Warning in loop_apply(n, do.ply):  Removed 1 rows containing missing values (geom_path).
```



Let's see how the builtin function works. It chooses what it thinks the best values are for you.

```
hwf <- HoltWinters(tbh[,2],gamma=FALSE)
hwf

## Holt-Winters exponential smoothing with trend and without seasonal component.
##
```

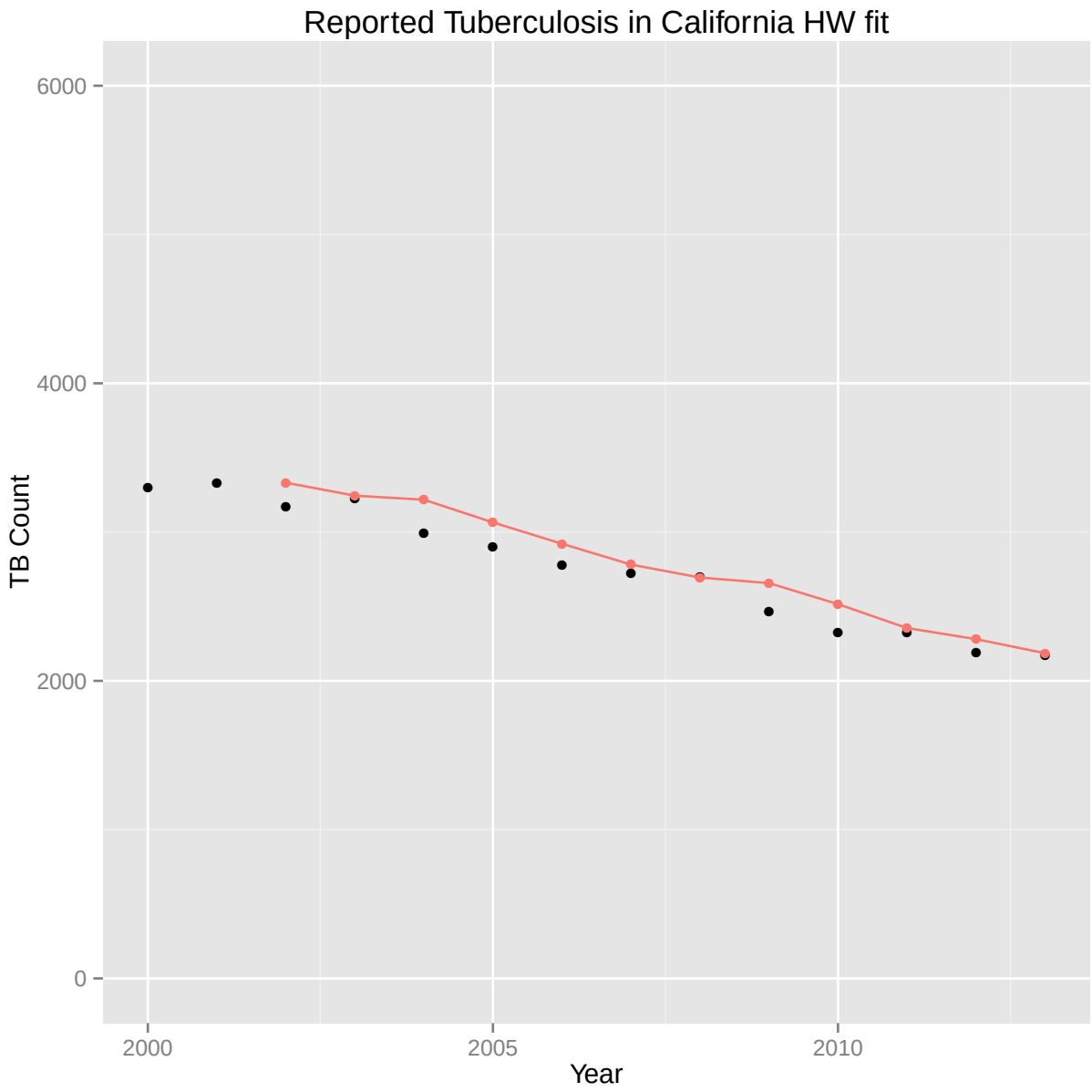
```
## Call:
## HoltWinters(x = tbh[, 2], gamma = FALSE)
##
## Smoothing parameters:
##  alpha: 1
##  beta : 0
##  gamma: FALSE
##
## Coefficients:
##   [,1]
## a 2169
## b  -50

tmp <- cbind(tbh[,1],tbh[,2],tbh[,1])
tmp[1:2,3] <- NA
tmp[3:dim(tmp)[1],3] <- hwf$fitted[,"level"]
tmp <- as.data.frame(tmp)
qplot(tmp[,1],tmp[,2],ylim=c(0,6000),xlab="Year",ylab="TB Count",main="Reported Tuberculosis in Califor

## Warning in loop_apply(n, do.ply):  Removed 2 rows containing missing values (geom_point).
## Warning in loop_apply(n, do.ply):  Removed 2 rows containing missing values (geom_path).
```



What if we fit using only since the year 2000?

```
hwf2 <- HoltWinters(tbh[16:29,2],gamma=FALSE)
hwf2

## Holt-Winters exponential smoothing with trend and without seasonal component.
##
```

```
## Call:
## HoltWinters(x = tbh[16:29, 2], gamma = FALSE)
##
## Smoothing parameters:
##  alpha: 0.6179972
##  beta : 0.6317919
##  gamma: FALSE
##
## Coefficients:
##         [,1]
## a 2138.61761
## b  -65.42894

tmp <- cbind(tbh[16:29,1],tbh[16:29,2],tbh[16:29,1])
tmp[1:2,3] <- NA
tmp[3:dim(tmp)[1],3] <- hwf2$fitted[,"level"]
tmp <- as.data.frame(tmp)
qplot(tmp[,1],tmp[,2],ylim=c(0,6000),xlab="Year",ylab="TB Count",main="Reported Tuberculosis in Califor

## Warning in loop_apply(n, do.ply):  Removed 2 rows containing missing values (geom_point).
## Warning in loop_apply(n, do.ply):  Removed 2 rows containing missing values (geom_path).
```

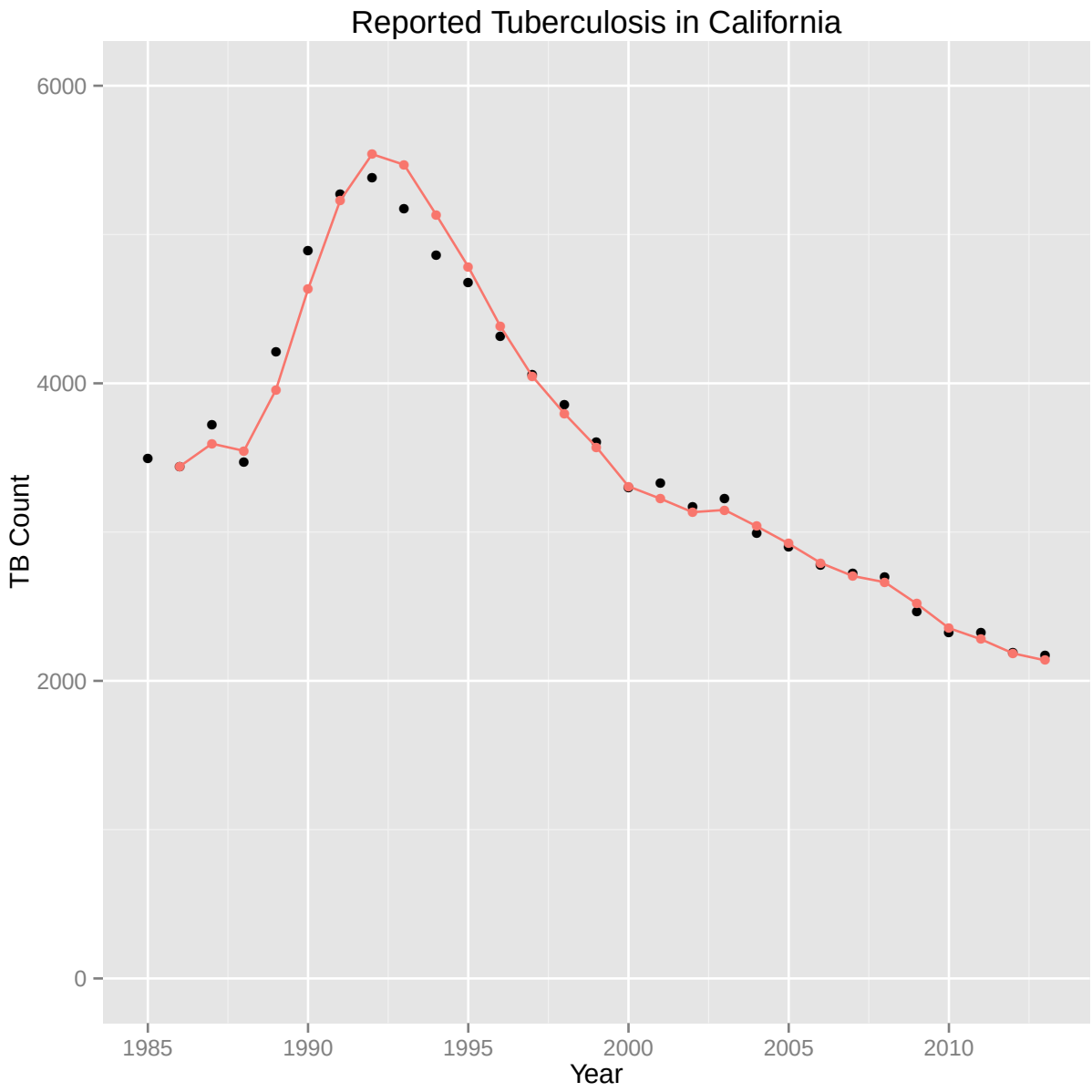


According to R, their coefficients are interpreted according to these equations (I've redacted the seasonal stuff):

```
Yhat[t+h] = a[t] + h * b[t],
where a[t], b[t] are given by
a[t] =  Y[t] + (1-) (a[t-1] + b[t-1])
b[t] =  (a[t] - a[t-1]) + (1-) b[t-1]
```

So with $a = \mu$, $b = b$, $\alpha = \lambda_0$, $\beta = \lambda_1$, $Y = x$, $t = \tau$, this is the same as above. Let's use the estimated parameters from the last part of the series, and use them on the whole series:

```
tbh$hwx <- hw.filter(tbh[,2],hwf2$alpha,hwf2$beta)
qplot(tbh[,1],tbh[,2],ylim=c(0,6000),xlab="Year",ylab="TB Count",main="Reported Tuberculosis in Califor

## Warning in loop_apply(n, do.ply):  Removed 1 rows containing missing values (geom_point).
## Warning in loop_apply(n, do.ply):  Removed 1 rows containing missing values (geom_path).
```



So apparently one issue was just that we were not starting the recurrences at the right place when we looked at the short series.

What is our forecast for 2014?

```
hw.filter(tbh[,2],hwf2$alpha,hwf2$beta,1)

## [1] 2072.733
```

The true answer, from the California DPH web site, was 2147.

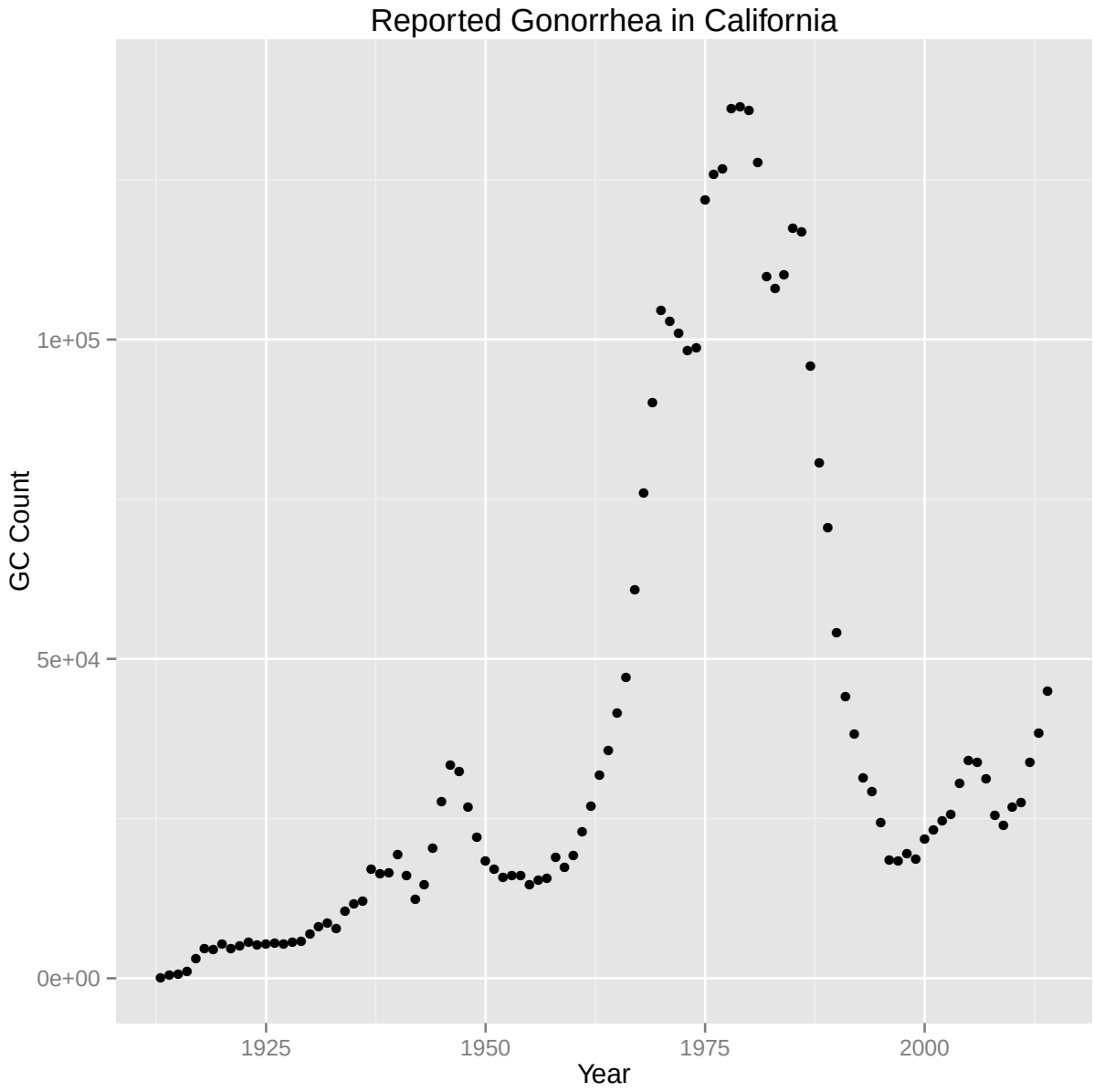
Let's forecast 2015 then, for a better contest.

```
tbhl <- data.frame(year=c(tbh[,1],2014),count=c(tbh[,2],2147))
hwf3 <- HoltWinters(tbhl[tbhl$year>=2000,"count"],gamma=FALSE)
tmp <- tbhl[tbhl$year>=2000,]
tmp$fc <- NA
tmp$fc[3:length(tmp$fc)] <- hw.filter(tbhl[,"count"],hwf3$alpha,hwf3$beta)

## Warning in tmp$fc[3:length(tmp$fc)] <- hw.filter(tbhl[, "count"], hwf3$alpha, :  number
of items to replace is not a multiple of replacement length

ff <- hw.filter(tbhl[,2],hwf3$alpha,hwf3$beta,1)
ff

## [1] 2087.473
```

My forecast for the year 2015 based on the Holt-Winters procedure is therefore 2087.

Incidence of gonorrhea per year in California (source: CDPH).

```
cag <- structure(list(year = 1913:2014, gono = c(117L, 467L, 695L, 1083L,
3006L, 4665L, 4570L, 5305L, 4709L, 5080L, 5704L, 5265L, 5391L,
5570L, 5348L, 5593L, 5842L, 7001L, 8123L, 8702L, 7817L, 10459L,
11634L, 12118L, 17051L, 16336L, 16542L, 19433L, 16098L, 12408L,
14632L, 20365L, 27668L, 33364L, 32396L, 26767L, 22027L, 18394L,
17122L, 15821L, 16081L, 16012L, 14697L, 15346L, 15679L, 18928L,
17327L, 19236L, 22979L, 26987L, 31825L, 35700L, 41551L, 47099L,
60810L, 75998L, 90073L, 104568L, 102804L, 101006L, 98242L, 98639L,
121919L, 125833L, 126768L, 136109L, 136463L, 135885L, 127723L,
109860L, 108066L, 110208L, 117392L, 116895L, 95877L, 80708L,
70596L, 54076L, 44104L, 38182L, 31443L, 29241L, 24369L, 18570L,
18424L, 19550L, 18662L, 21778L, 23285L, 24673L, 25693L, 30484L,
34098L, 33817L, 31191L, 25493L, 24009L, 26842L, 27483L, 33788L,
38364L, 44974L)), .Names = c("year", "gono"), class = "data.frame", row.names = c(NA,
-102L))
qplot(cag$year,cag$gono,ylim=c(0,140000),xlab="Year",ylab="GC Count",main="Reported Gonorrhea in Califo
```



Exercise 3. Use the California state gonorrhea counts to produce Holt-Winters forecasts for next year's case count. Please consider log transforming the data before analysis; you may wish to omit the first five years. Why do you think the peaks and troughs occurred? Do you think the Holt-Winters procedure is appropriate for the data? ∎

More about time series

Another interesting process is derived from this recurrence:

$$X_{\tau+1} = \alpha X_\tau + \epsilon_\tau$$

where we assume for now $|alpha| < 1$ and $E[\epsilon_\tau] = 0$ for all τ. We need to assume more though about the process ϵ_τ. One simple assumption is that $E[\epsilon_\tau \epsilon_{\tau'}] = E[\epsilon_\tau]E[\epsilon_{\tau'}]$ for $\tau \neq \tau'$. Such as series is *uncorrelated*; an uncorrelated series of random variables is called *white noise*.

We will assume the series ϵ_τ is uncorrelated, and also that ϵ_τ is uncorrelated with every value of X_i for $i = 1, 2, \ldots, \tau$.

So now

$$E[X_{\tau+1}] = E[\alpha X_\tau + \epsilon_\tau]$$

Apply linearity:

$$E[X_{\tau+1}] = E[\alpha X_\tau] + E[\epsilon_\tau]$$

Again:

$$E[X_{\tau+1}] = E[\alpha X_\tau] + E[\epsilon_\tau]$$

$$E[X_{\tau+1}] = \alpha E[X_\tau] + E[\epsilon_\tau]$$

The epsilons have mean zero:

$$E[X_{\tau+1}] = \alpha E[X_\tau] + E[\epsilon_\tau]$$

$$E[X_{\tau+1}] = \alpha E[X_\tau] + 0$$

So

$$E[X_{\tau+1}] = \alpha E[X_\tau].$$

Let $\mu_\tau = E[X_\tau]$. Therefore $\mu_{\tau+1} = \alpha \mu_\tau$. We can see that

$$\mu_{\tau+\ell} = \alpha^\ell \mu_\tau$$

in general, and

$$\mu_T = \alpha^T \mu_0.$$

Exercise 4. What is the limit of μ_T for large T? ∎

Let's compute the variance in the far future.

$$\text{var}(X_{\tau+1}) = \text{var}(\alpha X_\tau + \epsilon_\tau) = \alpha^2 \text{var}(X_\tau) + \sigma^2.$$

This happens to be another linear recurrence (difference equation); the equilibrium is

$$\sigma'^2 = \frac{\sigma^2}{1-\alpha^2}.$$

Let's compute the correlation coefficient of two values, one time apart.

$$E[X_\tau X_{\tau+1}] = E[X_\tau(\alpha X_\tau + \epsilon_\tau)] = E[\alpha X_\tau^2] + E[X_\tau \epsilon_\tau]$$

We assumed the noise ϵ_τ was uncorrelated with the value, so the second term is zero. So

$$E[X_\tau X_{\tau+1}] = \alpha E[X_\tau^2].$$

Similarly, we have

$$E[X_\tau]E[X_{\tau+1}] = E[X_\tau](E[\alpha X_\tau + \epsilon_\tau]) = E[X_\tau]\alpha(E[X_\tau]) = \alpha(E[X_\tau])^2$$

The correlation coefficient of X and Y is

$$\rho = \frac{E[XY] - EXEY}{\sqrt{\text{var}(X)\text{var}(Y)}}.$$

So our numerator is $\alpha \text{var}(X_\tau)$.

What is the variance of $X_{\tau+1}$? It must be $\text{var}(\alpha X_\tau + \epsilon_\tau)$. This is going to be $\alpha^2 \text{var}(X_\tau) + \sigma^2$.

So

$$\rho = \frac{\alpha \text{var}(X_\tau)}{\sqrt{\text{var}(X_\tau)(\alpha^2 \text{var}(X_\tau) + \sigma^2)}}$$

Let's go ahead and use $\text{var}(X_\tau) = \sigma^2/(1-\alpha^2)$ for large times:

$$\rho = \frac{\alpha \frac{\sigma^2}{1-\alpha^2}}{\sqrt{\frac{\sigma^2}{1-\alpha^2}\left(\alpha^2 \frac{\sigma^2}{1-\alpha^2} + \sigma^2\right)}}$$

Let's clean up this mess:

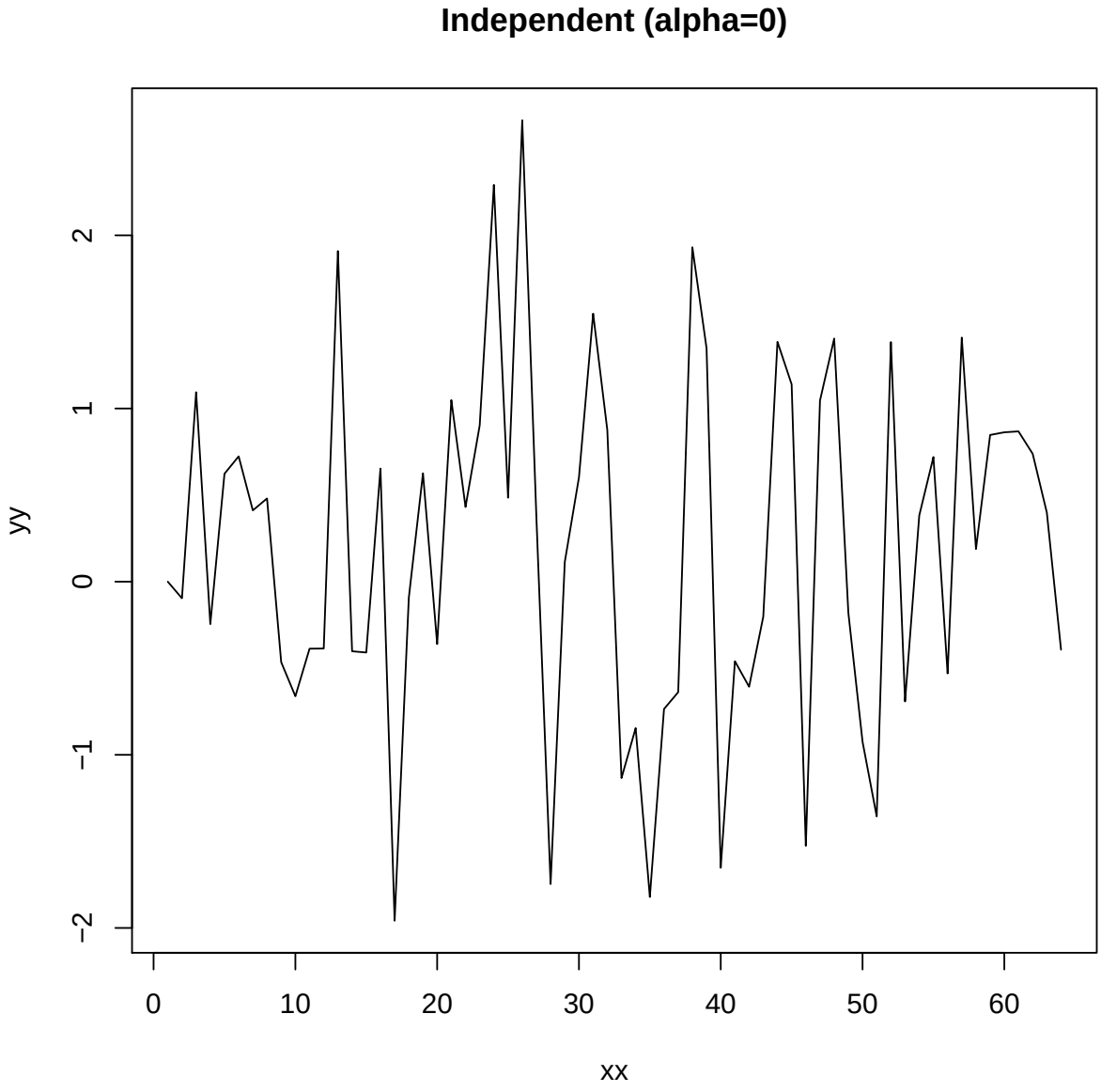
$$\rho = \frac{\alpha \frac{1}{1-\alpha^2}}{\sqrt{\frac{1}{1-\alpha^2}\left(\alpha^2 \frac{1}{1-\alpha^2} + \frac{1-\alpha^2}{1-\alpha^2}\right)}}$$

$$\rho = \frac{\alpha \frac{1}{1-\alpha^2}}{\sqrt{\frac{1}{1-\alpha^2}\frac{1}{1-\alpha^2}}} = \alpha$$

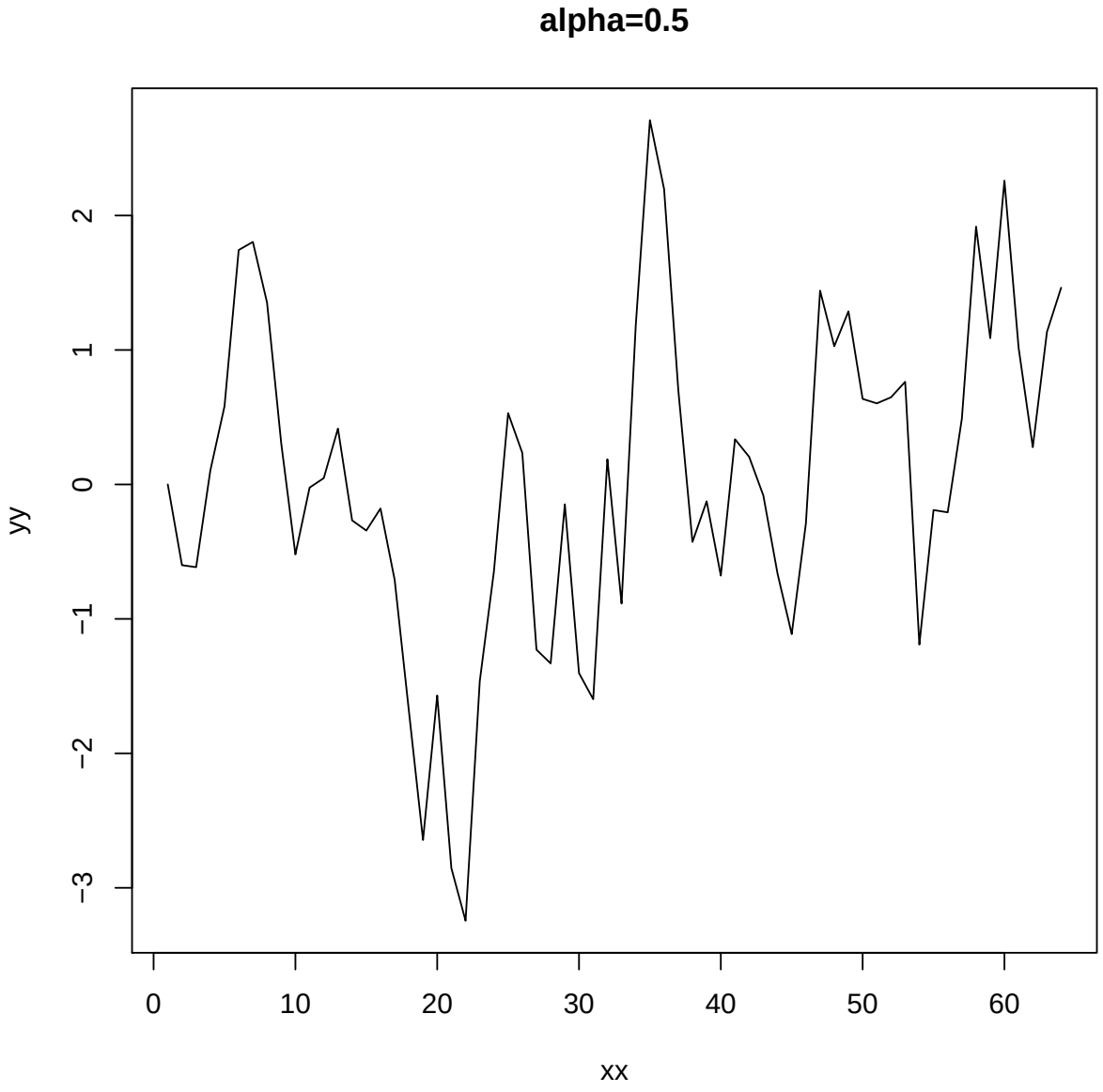
Since all we need here is $|\alpha| < 1$, this process can have negative autocorrelation as well as positive. This is called an AR(1) process and is important in time series and repeated measures analysis.

Let's look at some examples; I'm going to assume independent normal (Gaussian) white noise with variance 1. Here is R code for the simulation.

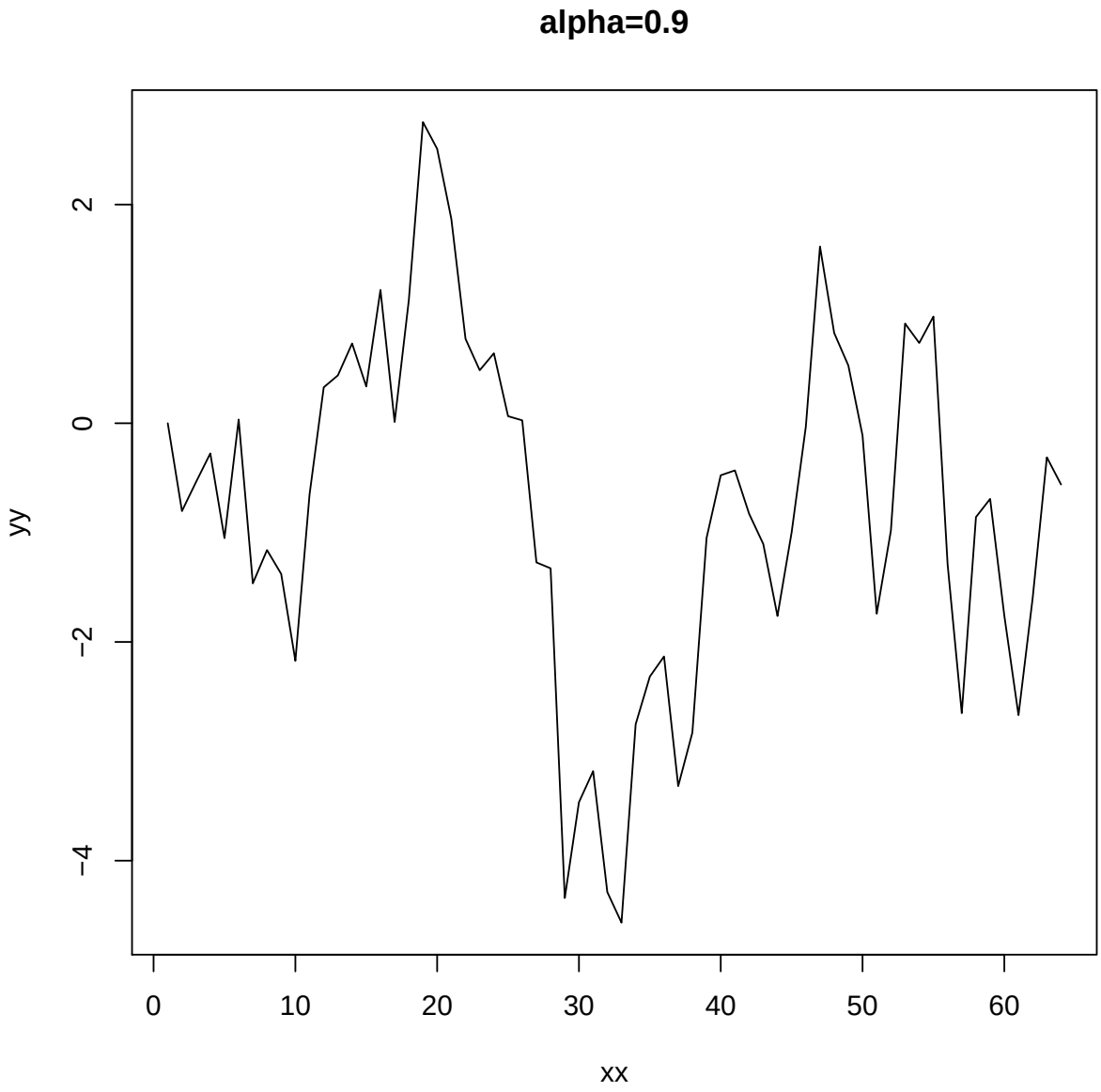
```
ar1sim <- function(len,alpha,starting=0) {
  answer <- rep(NA,len)
  answer[1] <- starting
  cur <- starting
  for (ii in 2:len) {
    cur <- alpha*cur + rnorm(1,mean=0,sd=1)
    answer[ii] <- cur
  }
  answer
}
xx <- 1:64
yy <- ar1sim(length(xx),0)
plot(xx,yy,main="Independent (alpha=0)",type="l")
```




```
yy <- ar1sim(length(xx),0.5)
plot(xx,yy,main="alpha=0.5",type="l")
```




```
yy <- ar1sim(length(xx),0.9)
plot(xx,yy,main="alpha=0.9",type="l")
```



Exercise 5. Using R or some other language of your choice, plot simulated AR(1) time series for the normal model with unit variance. Choose $\alpha = 0.99, 0.8, 0.5, 0.2, 0.1, -0.1, -0.2, -0.5, -0.8, -0.99$. Run series longer than 64; run replications over and over again. (You only need to turn in one replication.) Please comment on the differences and similarities. ∎

Review and preparation for the Galton-Watson process

Suppose we parameterize the geometric distribution in such a way that we ask how many failures (not how many trials: how many failures) before we have our first success. Then we get $P(Y = i) = p(1-p)^i$ for $i = 0, 1, 2, \ldots$. This is still called the geometric; it's just defined from 0 to infinity. You see both formulations in the literature; R, for example, uses this one. The generating function in this case is

$$E[u^Y] = \sum_{y=0}^{\infty} u^y P(Y = i) = \sum_{y=0}^{\infty} u^y p(1-p)^y = p\frac{1}{1-u(1-p)}.$$

Note something interesting when comparing this distribution to the version from before—the probability generating function of the version shifted 1 to the right is just this one multiplied by u.

Let's ask the question how many failures before we get 2 successes. That's a geometric waiting time for the first success, and another for the second. It's the sum of two independent geometrics (with the same p). Let the first one be Y_1, the second Y_2:

$$E[u^{Y_1+Y_2}] = E[u_{Y_1}]E[u_{Y_2}] = p\frac{1}{1-u(1-p)}p\frac{1}{1-u(1-p)} = \frac{p^2}{(1-u(1-p))^2}$$

If we are waiting for k successes, we have to add up k independent identical geometrics, and we get

$$E[u^{Y_1+\cdots+Y_k}] = \frac{p^k}{(1-u(1-p))^k}$$

a distribution known as the *negative binomial* distribution.

Even though we defined it as a waiting time until k successes, in fact there's nothing stopping us from using the generating function we just derived, but with k any positive number, integer or not. This distribution is widely used in modeling and applied statistics for all sorts of things having nothing to do with waiting times, though it can be used for those of course.

Let's get the mean of a negative binomial Y with parameters p and k:

$$E[Y] = g'(1) = \frac{d}{du}\frac{p^k}{(1-u(1-p))^k} = p^k(-k)(1-u(1-p))^{-k-1}(-(1-p))|_{u=1}.$$

This gives us

$$E[Y] = \frac{p^k k(1-p)}{p^{k+1}} = \frac{k(1-p)}{p}.$$

Of course—this was from adding up independent geometrics (geometrics starting from 0).

Exercise 6. Find the variance of the negative binomial. ∎

It turns out that if you take $k \to \infty$ holding the mean constant, you get a Poisson. Just as the binomial has a Poisson limit, so does the negative binomial. We will show this now.

Exercise 7. How long did this assignment take you? ∎

Reference:

Harvey, AC. Forecasting, structural time series, and the Kalman filter.