00:02 Monday 18th April, 2016
Copyright ©Cosma Rohilla Shalizi; do not distribute without permission
updates at http://www.stat.cmu.edu/~cshalizi/ADAfaEPoV/

Chapter 21

Time Series

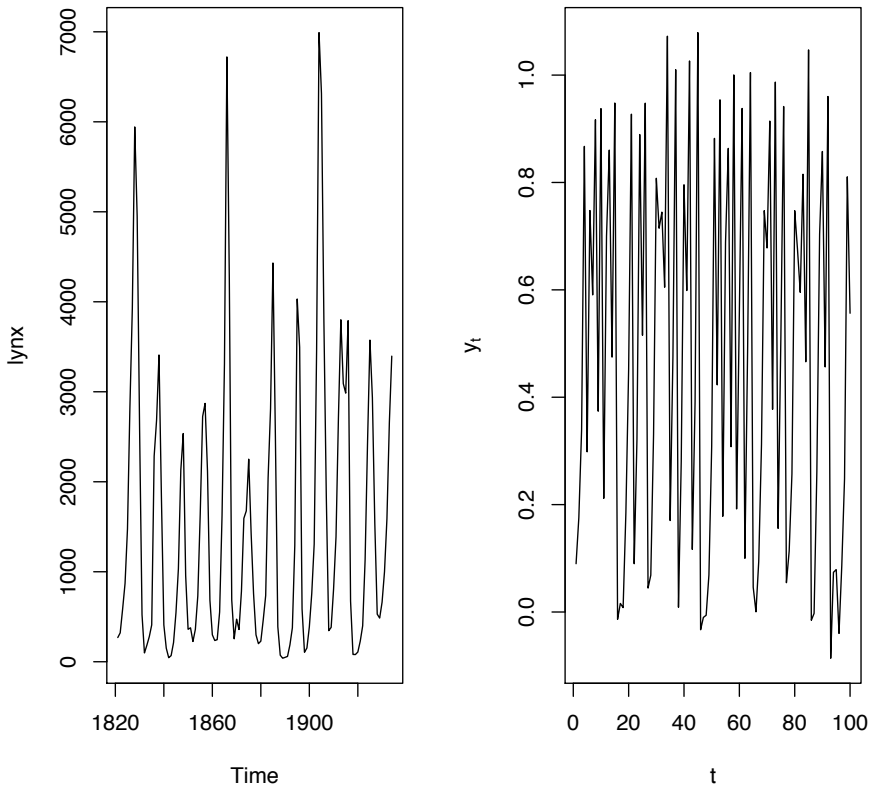
So far, we have assumed that all data points are pretty much independent of each other. In the chapters on regression, we assumed that each Y_i was independent of every other, given its X_i, and we often assumed that the X_i were themselves independent. In the chapters on multivariate distributions and even on causal inference, we allowed for arbitrarily complicated dependence between the *variables*, but each *data-point* was assumed to be generated independently. We will now relax this assumption, and see what sense we can make of dependent data.

21.1 What Time Series Are

The simplest form of dependent data are time series, which are just what they sound like: a series of values recorded over time. The most common version of this, in statistical applications, is to have measurements of a variable or variables X at equally-spaced time-points starting from t, written say $X_t, X_{t+h}, X_{t+2h}, \ldots$, or $X(t), X(t+h), X(t+2h), \ldots$. Here h, the amount of time between observations, is called the "sampling interval", and $1/h$ is the "sampling frequency" or "sampling rate".

Figure 21.1 shows two fairly typical time series. One of them is actual data (the number of lynxes trapped each year in a particular region of Canada); the other is the output of a purely artificial model. (Without the labels, it might not be obvious which one was which.) The core idea of time series analysis is one which we're already familiar with from the rest of statistics: we regard the actual time series we see as one realization of some underlying, partially-random ("stochastic") process, which generated the data. We use the data to make guesses ("inferences") about the process, and want to make *reliable* guesses while being clear about the uncertainty involved. The complication is that each observation is dependent on all the other observations; in fact it's usually this dependence that we want to draw inferences about.

Other kinds of time series One sometimes encounters irregularly-sampled time series, $X(t_1), X(t_2), \ldots$, where $t_i - t_{i-1} \neq t_{i+1} - t_i$. This is mostly an annoyance, unless the observation times are somehow dependent on the values. Continuously-




```
par(mfrow = c(1, 2))
plot(lynx)
plot(y[1:100], xlab = "t", ylab = expression(y[t]), type = "l")
par(mfrow = c(1, 1))
```

FIGURE 21.1: *Left: annual number of trapped lynxes in the Mackenzie River region of Canada. Right: a toy dynamical model, simulated from Code Example 37.*

```
logistic.map <- function(x, r = 4) {
    r * x * (1 - x)
}
logistic.iteration <- function(n, x.init, r = 4) {
    x <- vector(length = n)
    x[1] <- x.init
    for (i in 1:(n - 1)) {
        x[i + 1] <- logistic.map(x[i], r = r)
    }
    return(x)
}
x <- logistic.iteration(1000, x.init = runif(1))
y <- x + rnorm(1000, mean = 0, sd = 0.05)
```

CODE EXAMPLE 37: *Code defining our synthetic data set. Exercise: why is this "logistic"?*

observed processes are rarer — especially now that digital sampling has replaced analog measurement in so many applications. (It is more common to model the process as evolving continuously in time, but observe it at discrete times.) We skip both of these in the interest of space.

Regular, irregular or continuous time series all record the same variable at every moment of time. An alternative is to just record the sequence of times at which some event happened; this is called a "point process". More refined data might record the time of each event and its type — a "marked point process". Point processes include very important kinds of data (e.g., earthquakes), but they need special techniques, and we'll skip them (though see §21.13).

Notation For a regularly-sampled time series, it's convenient not to have to keep writing the actual time, but just the position in the series, as $X_1, X_2, \ldots$, or $X(1), X(2), \ldots$. This leads to a useful short-hand, that $X_i^j = (X_i, X_{i+1}, \ldots X_{j-1}, X_j)$, a whole block of time; some people write $X_{i:j}$ with the same meaning.

21.2 Stationarity

In our old IID world, the distribution of each observation is the same as the distribution of every other data point. It would be nice to have something like this for time series. The property is called **stationarity**, which doesn't mean that the time series never changes, but that its *distribution* doesn't.

More precisely, a time series is **strictly stationary** or **strongly stationary** when X_1^k and X_t^{t+k-1} have the same distribution, for all k and t — the distribution of blocks of length k is **time-invariant**. Again, this doesn't mean that every block of length k has the same value, just that it has the same distribution of values.

If there is strong or strict stationarity, there should be **weak** or **loose** (or **wide-sense**) stationarity, and there is. All it requires is that $\mathbb{E}[X_1] = \mathbb{E}[X_t]$, and that

$\text{Cov}\left[X_1, X_k\right] = \text{Cov}\left[X_t, X_{t+k-1}\right]$. (Notice that it's not dealing with whole blocks of time any more, just single time-points.) Clearly (exercise!), strong stationarity implies weak stationarity, but not, in general, the other way around, hence the names. It may not surprise you to learn that strong and weak stationarity coincide when X_t is a Gaussian process, but not,in general, otherwise.

You should convince yourself that an IID sequence is strongly stationary.

21.2.1 Autocorrelation

Time series are **serially dependent**: X_t is in general dependent on all earlier values in time, and on all later ones. Typically, however, there is **decay of dependence** (sometimes called **decay of correlations**): X_t and X_{t+h} become more and more nearly independent as $h \to \infty$. The oldest way of measuring this is the **autocovariance**,

$$\gamma(h) = \text{Cov}\left[X_t, X_{t+h}\right] \tag{21.1}$$

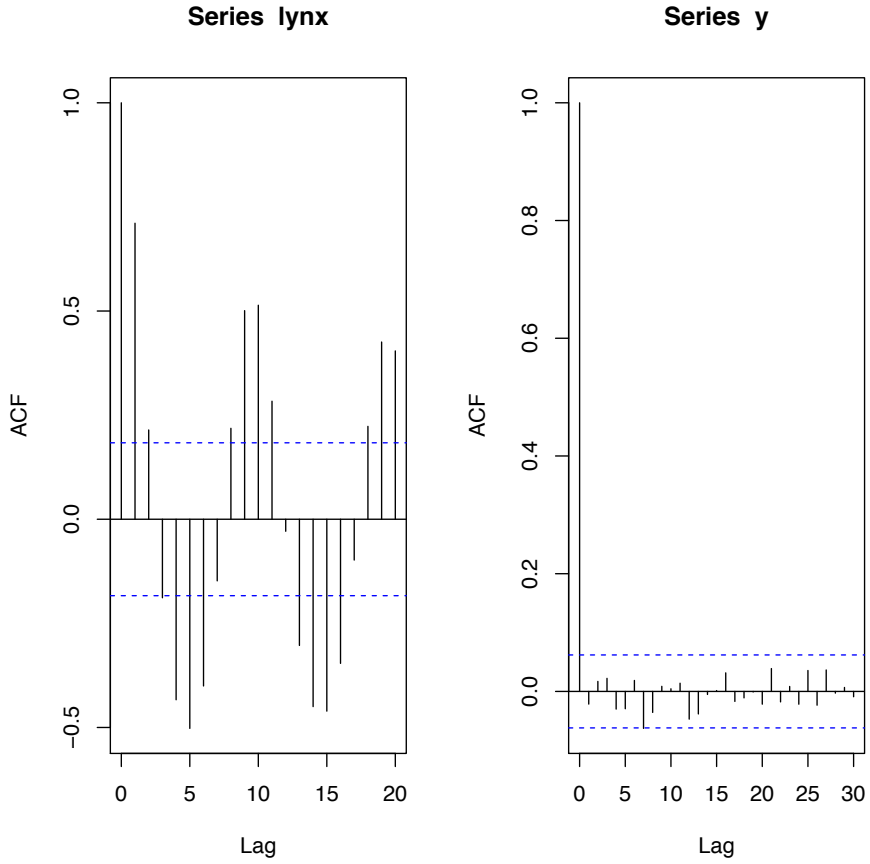
which is well-defined just when the process is weakly stationary. We could equally well use the **autocorrelation**,

$$\rho(h) = \frac{\text{Cov}\left[X_t, X_{t+h}\right]}{\mathbb{V}\left[X_t\right]} = \frac{\gamma(h)}{\gamma(0)} \tag{21.2}$$

again using stationarity to simplify the denominator.

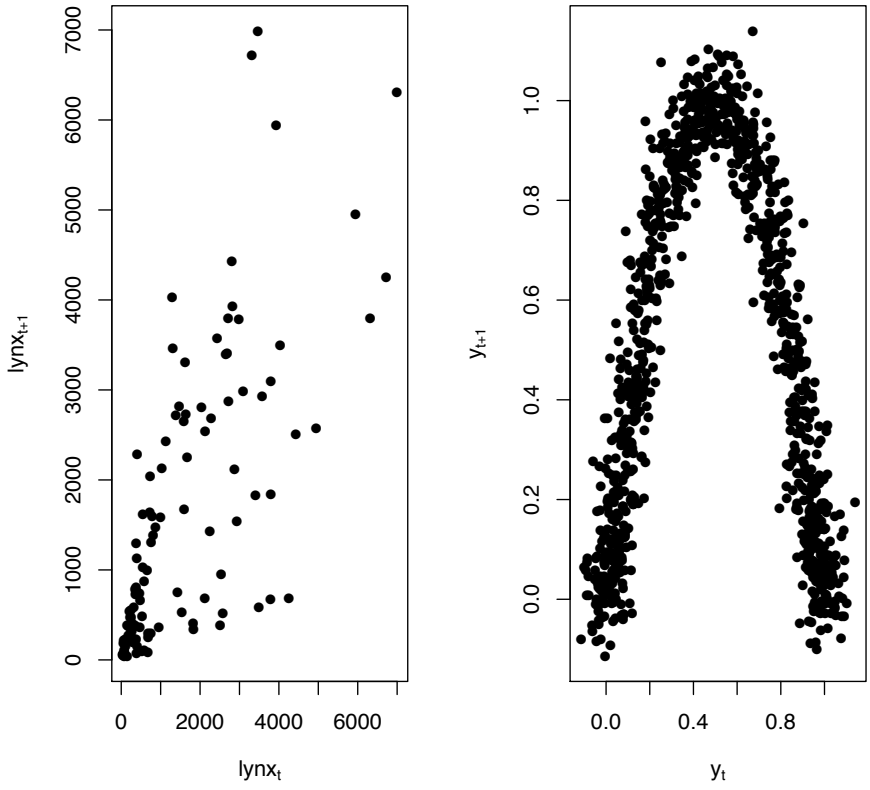
As I said, for most time series $\gamma(h) \to 0$ as h grows. Of course, $\gamma(h)$ could be exactly zero while X_t and X_{t+h} are strongly dependent. Figure 21.2 shows the autocorrelation functions (ACFs) of the lynx data and the simulation model; the correlation for the latter is basically never distinguishable from zero, which doesn't accord at all with the visual impression of the series. Indeed, we can confirm that *something* is going on the series by the simple device of plotting X_{t+1} against X_t (Figure 21.3). More general measures of dependence would include looking at the Spearman rank-correlation of X_t and X_{t+h}, or quantities like mutual information.

Autocorrelation is important for four reasons, however. First, because it is the oldest measure of serial dependence, it has a "large installed base": everybody knows about it, they use it to communicate, and they'll ask you about it. Second, in the rather special case of Gaussian processes, it really does tell us everything we need to know. Third, in the somewhat less special case of linear prediction, it tells us everything we need to know. Fourth and finally, it plays an important role in a crucial theoretical result, which we'll go over next.




```
par(mfrow = c(1, 2))
acf(lynx)
acf(y)
par(mfrow = c(1, 1))
```

FIGURE 21.2: *Autocorrelation functions of the lynx data (above) and the simulation (below). The* `acf` *function plots the autocorrelation function as an automatic side-effect; it actually returns the actual value of the autocorrelations, which you can capture. The 95% confidence interval around zero is computed under Gaussian assumptions which shouldn't be taken too seriously, unless the sample size is quite large, but are useful as guides to the eye.*




```
par(mfrow = c(1, 2))
plot(lag0 ~ lag1, data = design.matrix.from.ts(lynx, 1), xlab = expression(lynx[t]),
    ylab = expression(lynx[t + 1]), pch = 16)
plot(lag0 ~ lag1, data = design.matrix.from.ts(y, 1), xlab = expression(y[t]),
    ylab = expression(y[t + 1]), pch = 16)
par(mfrow = c(1, 1))
```

FIGURE 21.3: *Plots of X_{t+1} versus X_t, for the lynx (left) and the simulation (right); see Exercise 1. Note that even though the correlation between successive iterates is next to zero for the simulation, there is clearly a lot of dependence (see Appendix E.4).*

21.2.2 The Ergodic Theorem

With IID data, the ultimate basis of all our statistical inference is the law of large numbers, which told us that

$$\frac{1}{n}\sum_{i=1}^{n} X_i \to \mathbb{E}[X_1] \tag{21.3}$$

For complicated historical reasons, the corresponding result for time series is called the **ergodic theorem**[1]. The most general and powerful versions of it are quite formidable, and have very subtle proofs, but there is a simple version which gives the flavor of them all, and is often useful enough.

21.2.2.1 The World's Simplest Ergodic Theorem

Suppose X_t is weakly stationary, and that

$$\sum_{h=0}^{\infty} |\gamma(h)| = \gamma(0)\tau < \infty \tag{21.4}$$

(Remember that $\gamma(0) = \mathbb{V}[X_t]$.) The quantity τ is called the **correlation time**, or **integrated autocorrelation time**.

Now consider the average of the first n observations,

$$\overline{X}_n = \frac{1}{n}\sum_{t=1}^{n} X_t \tag{21.5}$$

This **time average** is a random variable. Its expectation value is

$$\mathbb{E}\left[\overline{X}_n\right] = \frac{1}{n}\sum_{t=1}^{n} \mathbb{E}\left[X_t\right] = \mathbb{E}[X_1] \tag{21.6}$$

[1] In the late 1800s, the physicist Ludwig Boltzmann needed a word to express the idea that if you took an isolated system at constant energy and let it run, any one trajectory, continued long enough, would be representative of the system as a whole. Being a highly-educated nineteenth century German-speaker, Boltzmann knew far too much ancient Greek, so he called this the "ergodic property", from *ergon* "energy, work" and *hodos* "way, path". The name stuck.

because the mean is stationary. What about its variance?

$$\begin{aligned}
\mathbb{V}\left[\overline{X}_n\right] &= \mathbb{V}\left[\frac{1}{n}\sum_{t=1}^{n} X_t\right] && (21.7)\\
&= \frac{1}{n^2}\left[\sum_{t=1}^{n}\mathbb{V}\left[X_t\right] + 2\sum_{t=1}^{n}\sum_{s=t+1}^{n}\mathrm{Cov}\left[X_t, X_s\right]\right] && (21.8)\\
&= \frac{1}{n^2}\left[n\mathbb{V}\left[X_1\right] + 2\sum_{t=1}^{n}\sum_{s=t+1}^{n}\gamma(s-t)\right] && (21.9)\\
&\leq \frac{1}{n^2}\left[n\gamma(0) + 2\sum_{t=1}^{n}\sum_{s=t+1}^{n}|\gamma(s-t)|\right] && (21.10)\\
&\leq \frac{1}{n^2}\left[n\gamma(0) + 2\sum_{t=1}^{n}\sum_{h=1}^{n}|\gamma(h)|\right] && (21.11)\\
&\leq \frac{1}{n^2}\left[n\gamma(0) + 2\sum_{t=1}^{n}\sum_{h=1}^{\infty}|\gamma(h)|\right] && (21.12)\\
&= \frac{n\gamma(0)(1+2\tau)}{n^2} && (21.13)\\
&= \frac{\gamma(0)(1+2\tau)}{n} && (21.14)
\end{aligned}$$

Eq. 21.9 uses stationarity again, and then Eq. 21.13 uses the assumption that the correlation time τ is finite.

Since $\mathbb{E}\left[\overline{X_n}\right] = \mathbb{E}\left[X_1\right]$, and $\mathbb{V}\left[\overline{X}_n\right] \to 0$, we have that

$$\overline{X_n} \to \mathbb{E}\left[X_1\right] \tag{21.15}$$

exactly as in the IID case. ("Time averages converge on expected values.") In fact, we can say a bit more. Remember Chebyshev's inequality: for any random variable Z,

$$\Pr\left(|Z - \mathbb{E}\left[Z\right]| > \epsilon\right) \leq \frac{\mathbb{V}\left[Z\right]}{\epsilon^2} \tag{21.16}$$

so

$$\Pr\left(|\overline{X}_n - \mathbb{E}\left[X_1\right]| > \epsilon\right) \leq \frac{\gamma(0)(1+2\tau)}{n\epsilon^2} \tag{21.17}$$

which goes to zero as n grows for any given ϵ.

You may wonder whether the condition that $\sum_{h=0}^{\infty}|\gamma(h)| < \infty$ is as weak as possible. It turns out that it can in fact be weakened to just $\lim_{n\to\infty}\frac{1}{n}\sum_{h=0}^{n}\gamma(h) = 0$, as indeed the proof above might suggest.

The argument above can actually be extended to some non-stationary processes; see Exercise 6.

21.2.2.2 Rate of Convergence

If the X_t were all IID, or even just uncorrelated, we would have $\mathbb{V}\left[\overline{X}_n\right] = \gamma(0)/n$ exactly. Our bound on the variance is larger by a factor of $(1+2\tau)$, which reflects the influence of the correlations. Said another way, we can more or less pretend that instead of having n correlated data points, we have $n/(1+2\tau)$ independent data points, that $n/(1+2\tau)$ is our **effective sample size**[2]

Generally speaking, dependence between observations reduces the effective sample size, and the stronger the dependence, the greater the reduction. (For an extreme, consider the situation where X_1 is randomly drawn, but thereafter $X_{t+1} = X_t$.) In more complicated situations, finding the effective sample size is itself a tricky undertaking, but it's often got this general flavor.

21.2.2.3 Why Ergodicity Matters

The ergodic theorem is important, because it tells us that a single long time series becomes representative of the whole data-generating process, just the same way that a large IID sample becomes representative of the whole population or distribution. We can therefore actually learn about the process from empirical data.

Strictly speaking, we have established that time-averages converge on expectations only for X_t itself, not even for $f(X_t)$ where the function f is non-linear. It might be that $f(X_t)$ doesn't have a finite correlation time even though X_t does, or indeed vice versa. This is annoying; we don't want to have to go through the analysis of the last section for every different function we might want to calculate.

When people say that the whole process is **ergodic**, they roughly speaking mean that

$$\frac{1}{n}\sum_{t=1}^{n} f(X_t^{t+k-1}) \to \mathbb{E}\left[f(X_1^k)\right] \tag{21.18}$$

for any reasonable function f. This is (again very roughly) equivalent to

$$\frac{1}{n}\sum_{t=1}^{n} \Pr\left(X_1^k \in A, X_t^{t+l-1} \in B\right) \to \Pr\left(X_1^k \in A\right)\Pr\left(X_1^l \in B\right) \tag{21.19}$$

which is a kind of asymptotic independence-on-average[3]

[2]Some people like to define the correlation time as, in this notation, $1+2\tau$ for just this reason.

[3]It's worth sketching a less rough statement. Instead of working with X_t, work with the whole future trajectory $Y_t = (X_t, X_{t+1}, X_{t+2}, \ldots)$. Now the dynamics, the rule which moves us into the future, can be summed up in a very simple, and deterministic, operation T: $Y_{t+1} = TY_t = (X_{t+1}, X_{t+2}, X_{t+3}, \ldots)$. A set of trajectories is **invariant** if it is left unchanged by T: for every $y \in A$, there is another y' in A where $Ty' = y$. A process is **ergodic** if every invariant set either has probability 0 or probability 1. What this means is that (almost) all trajectories generated by an ergodic process belong to a single invariant set, and they all wander from every part of that set to every other part — they are **metrically transitive**. (Because: no smaller set with any probability is invariant.) Metric transitivity, in turn, is equivalent, assuming stationarity, to $n^{-1}\sum_{t=0}^{n-1} \Pr(Y \in A, T^t Y \in B) \to \Pr(Y \in A)\Pr(Y \in B)$. From metric transitivity follows Birkhoff's "individual" ergodic theorem, that $n^{-1}\sum_{t=0}^{n-1} f(T^t Y) \to \mathbb{E}[f(Y)]$, with probability 1. Since a function of the trajectory can be a function of a block of length k, we get Eq. 21.18.

If our data source is ergodic, then what Eq. 21.18 tells us is that sample averages of any reasonable function are representative of expectation values, which is what we need to be in business statistically. This in turn is basically implied by stationarity.[4] What does this let us do?

21.3 Markov Models

For this section, we'll assume that X_t comes from a stationary, ergodic time series. So for any reasonable function f, the time-average of $f(X_t)$ converges on $\mathbb{E}[f(X_1)]$. Among the "reasonable" functions are the indicators, so

$$\frac{1}{n}\sum_{t=1}^{n} \mathbf{1}_A(X_t) \to \Pr(X_1 \in A) \tag{21.20}$$

Since this also applies to functions of blocks,

$$\frac{1}{n}\sum_{t=1}^{n} \mathbf{1}_{A,B}(X_t, X_{t+1}) \to \Pr(X_1 \in A, X_2 \in B) \tag{21.21}$$

and so on. If we can learn joint and marginal probabilities, and we remember how to divide, then we can learn conditional probabilities.

It turns out that pretty much any density estimation method which works for IID data will also work for getting the marginal and conditional distributions of time series (though, again, the effective sample size depends on how quickly dependence decays). So if we want to know $p(x_t)$, or $p(x_{t+1} \mid x_t)$, we can estimate it just as we learned how to do in Chapter 14. Just as in that chapter, much the same techniques apply whether x is discrete or continuous; for brevity, I'll speak as though x is continuous and $p(x_{t+1} \mid x_t)$ is a conditional pdf.

Now, the conditional distribution $p(x_{t+1} \mid x_t)$ always exists, and we can always estimate it. But why stop just one step back into the past? Why not look at $p(x_{t+1} \mid x_t, x_{t-1})$, or for that matter $p(x_{t+1} \mid x_{t-999}^t)$? There are three reasons, in decreasing order of pragmatism.

- Estimating $p(x_{t+1} \mid x_{t-999}^t)$ means estimating a thousand-and-one-dimensional distribution. The curse of dimensionality will crush us.
- Because of the decay of dependence, there shouldn't be much difference, much of the time, between $p(x_{t+1} \mid x_{t-999}^t)$ and $p(x_{t+1} \mid x_{t-998}^t)$, etc. Even if we could

[4] Again, a sketch of a less rough statement. Use Y again for whole trajectories. Every stationary distribution for Y can be written as a mixture of stationary and ergodic distributions, rather as we wrote complicated distributions as mixtures of simple Gaussians in Chapter 19. (This is called the "ergodic decomposition" of the process: see Gray 2009.) We can think of this as first picking an ergodic process according to some fixed distribution, and then generating Y from that process. Time averages computed along any one trajectory thus converge according to Eq. 21.18. If we have only a single trajectory, it looks just like a stationary and ergodic process. It is thus common to assume that the data source is not only stationary but also ergodic. This only becomes a problem if we have multiple trajectories from the same source, each of which one may be converging to a different ergodic component.

go very far back into the past, it shouldn't, usually, change our predictions very much.

- Sometimes, a finite, short block of the past completely screens off the remote past.

You will remember the Markov property from your previous probability classes:

$$X_{t+1} \perp\!\!\!\perp X_1^{t-1} \mid X_t \tag{21.22}$$

When the Markov property holds, there is simply no point in looking at $p(x_{t+1} \mid x_t, x_{t-1})$, because it's got to be just the same as $p(x_{t+1} \mid x_t)$. If the process isn't a simple Markov chain but has a higher-order Markov property,

$$X_{t+1} \perp\!\!\!\perp X_1^{t-k} \mid X_{t-k+1}^{t} \tag{21.23}$$

then we never have to condition on more than the last k steps to learn all that there is to know. The Markov property means that the current state screens off the future from the past.

It is *always* an option to model X_t as a Markov process, or a higher-order Markov process. If it isn't exactly Markov, if there's really some dependence between the past and the future even given the current state, then we're introducing some bias, but it can be small, and dominated by the reduced variance of not having to worry about higher-order dependencies.

21.3.1 Meaning of the Markov Property

The Markov property is a weakening of both being strictly IID and being strictly deterministic.

That being Markov is weaker than being IID is obvious: an IID sequence satisfies the Markov property, because everything is independent of everything else no matter what we condition on.

In a deterministic dynamical system, on the other hand, we have $X_{t+1} = g(X_t)$ for some fixed function g. Iterating this equation, the current state X_t fixes the whole future trajectory $X_{t+1}, X_{t+2}, \ldots$. In a Markov chain, we weaken this to $X_{t+1} = g(X_t, U_t)$, where the U_t are IID noise variables (which we can take to be uniform for simplicity). The current state of a Markov chain doesn't fix the exact future trajectory, but it does fix the *distribution* over trajectories.

The real meaning of the Markov property, then, is about information flow: the current state is the only channel through which the past can affect the future.

[[TODO: Maximum likelihood for Markov models]]

[[TODO: Variable length Markov chains]]

t	x
1821	269
1822	321
1823	585
1824	871
1825	1475
1826	2821
1827	3928
1828	5943
1829	4950
...	

$\Rightarrow$

lag0	lag1	lag2	lag3
871	585	321	269
1475	871	585	321
2821	1475	871	585
3928	2821	1475	871
5943	3928	2821	1475
4950	5943	3928	2821
...			

FIGURE 21.4: *Turning a time series (here, the beginning of* `lynx`*) into a regression-suitable matrix.*

21.4 Autoregressive Models

Instead of trying to estimate the whole conditional distribution of X_t, we can just look at its conditional expectation. This is a regression problem, but since we are regressing X_t on earlier values of the series, it's called an **autoregression**:

$$\mathbb{E}\left[X_t \,|\, X_{t-p}^{t-1} = x_1^p\right] = r(x_1^p) \tag{21.24}$$

If we think the process is Markov of order p, then of course there is no point in conditioning on more than p steps of the past when doing an autoregression. But even if we don't think the process is Markov, the same reasons which inclined us towards Markov approximations also make limited-order autoregressions attractive.

Since this is a regression problem, we can employ all the tools we know for regression analysis: linear models, kernel regression, spline smoothing, additive models, etc., mixtures of regressions, etc. Since we are regressing X_t on earlier values from the same series, it is useful to have tools for turning a time series into a regression-style design matrix (as in Figure 21.4); see Code Example 38.

Suppose $p = 1$. Then we essentially want to draw regression curves through plots like those in Figure 21.3. Figure 21.5 shows an example for the artificial series.

21.4.1 Autoregressions with Covariates

Nothing keeps us from adding a variable other than the past of X_t to the regression:

$$\mathbb{E}\left[X_{t+1} \,|\, X_{t-k+1}^{t}, Z\right] \tag{21.25}$$

or even another time series:

$$\mathbb{E}\left[X_{t+1} \,|\, X_{t-k+1}^{t}, Z_{t-l+1}^{t}\right] \tag{21.26}$$

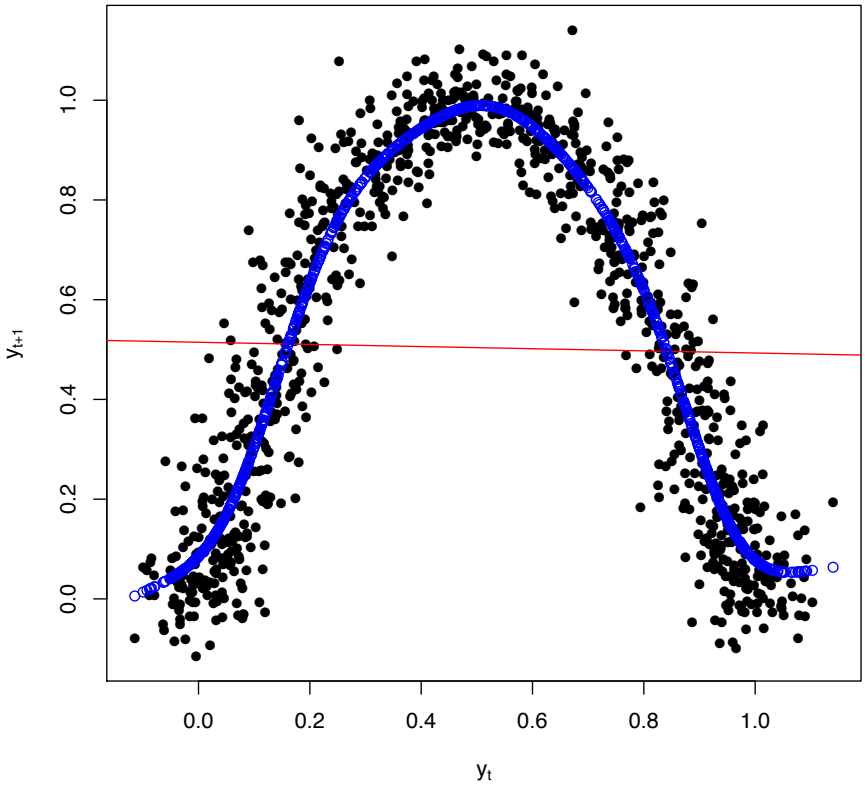
These are perfectly well-defined conditional expectations, and quite estimable in principle. Of course, adding more variables to a regression means having to estimate more, so again the curse of dimensionality comes up, but our methods are very much the same as in the basic regression analyses.

```
design.matrix.from.ts <- function(ts, order, right.older = TRUE) {
    n <- length(ts)
    x <- ts[(order + 1):n]
    for (lag in 1:order) {
        if (right.older) {
            x <- cbind(x, ts[(order + 1 - lag):(n - lag)])
        }
        else {
            x <- cbind(ts[(order + 1 - lag):(n - lag)], x)
        }
    }
    lag.names <- c("lag0", paste("lag", 1:order, sep = ""))
    if (right.older) {
        colnames(x) <- lag.names
    }
    else {
        colnames(x) <- rev(lag.names)
    }
    return(as.data.frame(x))
}
```

CODE EXAMPLE 38: *Example code for turning a time series into a design matrix, suitable for regression.*

```
aar <- function(ts, order) {
    stopifnot(require(mgcv))
    fit <- gam(as.formula(auto.formula(order)), data = design.matrix.from.ts(ts,
        order))
    return(fit)
}
auto.formula <- function(order) {
    inputs <- paste("s(lag", 1:order, ")", sep = "", collapse = "+")
    form <- paste("lag0 ~ ", inputs)
    return(form)
}
```

CODE EXAMPLE 39: *Fitting an additive autoregression of arbitrary order to a time series. See online for comments.*




```
plot(lag0 ~ lag1, data = design.matrix.from.ts(y, 1), xlab = expression(y[t]),
    ylab = expression(y[t + 1]), pch = 16)
abline(lm(lag0 ~ lag1, data = design.matrix.from.ts(y, 1)), col = "red")
yaar1 <- aar(y, order = 1)
points(y[-length(y)], fitted(yaar1), col = "blue")
```

FIGURE 21.5: *Plotting successive values of the artificial time series against each other, along with the linear regression, and a spline curve (see below for the* `aar` *function, which fits additive autoregressive models; with* `order=1`*, it just fits a spline.*

21.4.2 Additive Autoregressions

As before, if we want some of the flexibility of non-parametric smoothing, without the curse of dimensionality, we can try to approximate the conditional expectation as an additive function:

$$\mathbb{E}\left[X_t \mid X_{t-p}^{t-1}\right] \approx \alpha_0 + \sum_{j=1}^{p} g_j(X_{t-j}) \tag{21.27}$$

My personal experience with applied projects is that additive autoregressions tend to work surprisingly well.

Example: The lynx Let's try fitting an additive model for the lynx. Code Example 39 shows some code for doing this. (Most of the work is re-shaping the time series into a data frame, and then automatically generating the right formula for `gam`.) Let's try out $p = 2$.

```
lynx.aar2 <- aar(lynx, 2)
```

This inherits everything we can do with a GAM, so we can do things like plot the partial response functions (Figure 21.6), plot the fitted values against the actual (Figure 21.7), etc. To get a sense of how well it can actually extrapolate, Figure 21.8 re-fits the model to just the first 80 data points, and then predicts the remaining 34.

21.4.3 Linear Autoregression

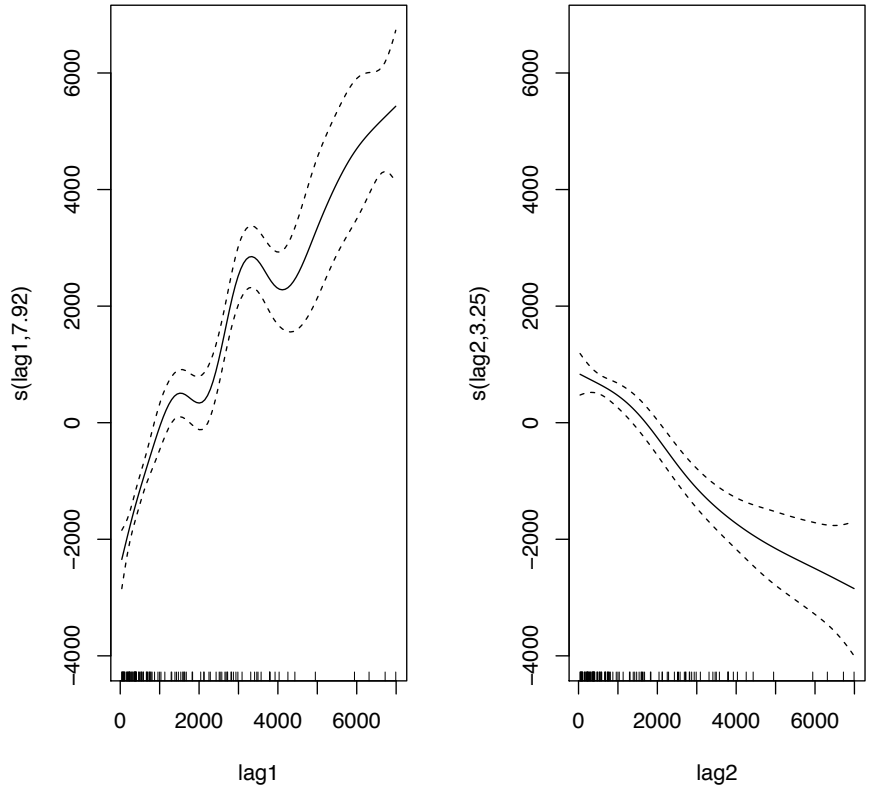
When people talk about autoregressive models, they usually (alas) just mean linear autoregressions. There is almost never any justification in scientific theory for this preference, but we can always ask for the best linear approximation to the true autoregression, if only because it's fast to compute and fast to converge.

The analysis we did in Chapter 2 of how to find the optimal linear predictor carries over with no change whatsoever. If we want to predict X_t as a linear combination of the last k observations, $X_{t-1}, X_{t-2}, \ldots X_{t-p}$, then the ideal coefficients β are

$$\beta = \left(\mathbb{V}\left[X_{t-p}^{t-1}\right]\right)^{-1} \mathrm{Cov}\left[X_{t-p}^{t-1}, X_t\right] \tag{21.28}$$

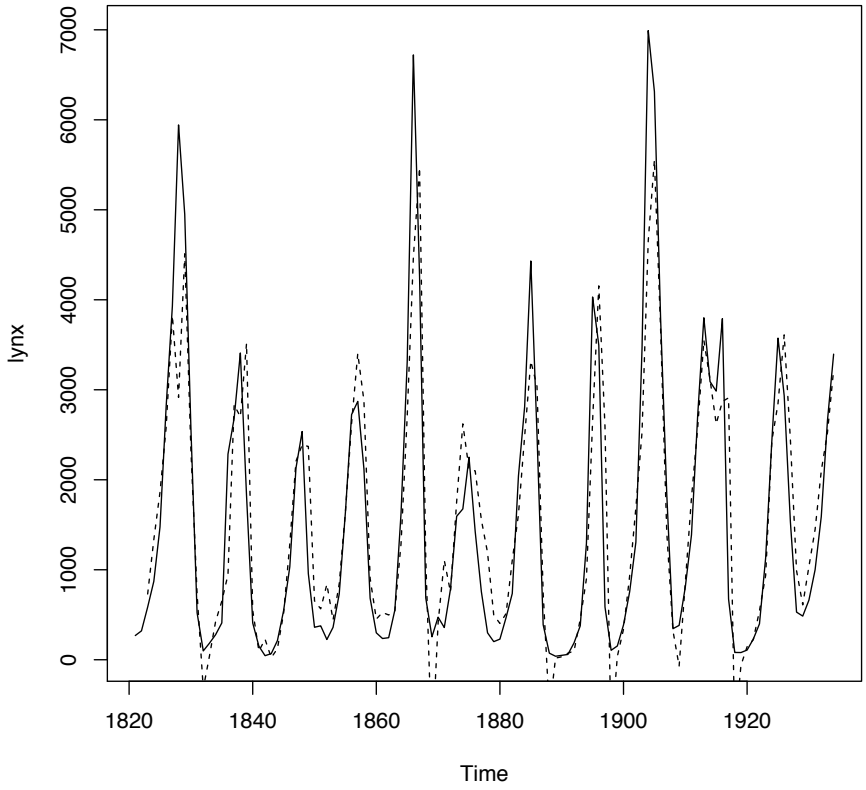
where $\mathbb{V}\left[X_{t-p}^{t}\right]$ is the variance-covariance matrix of $(X_{t-1}, \ldots X_{t-p})$ and similarly $\mathrm{Cov}\left[X_{t-p}^{t-1}, X_t\right]$ is a vector of covariances. Assume stationarity, $\mathbb{V}\left[X_t\right]$ is constant in t, and so the common factor of the over-all variance goes away, and β could be written entirely in terms of the correlation function ρ. Stationarity also lets us estimate these covariances, by taking time-averages.

A huge amount of effort is given over to using linear AR models, which in my opinion is out of all proportion to their utility — but very reflective of what was computationally feasible up to about 1980. My experience is that results like Figure 21.9 is pretty typical.



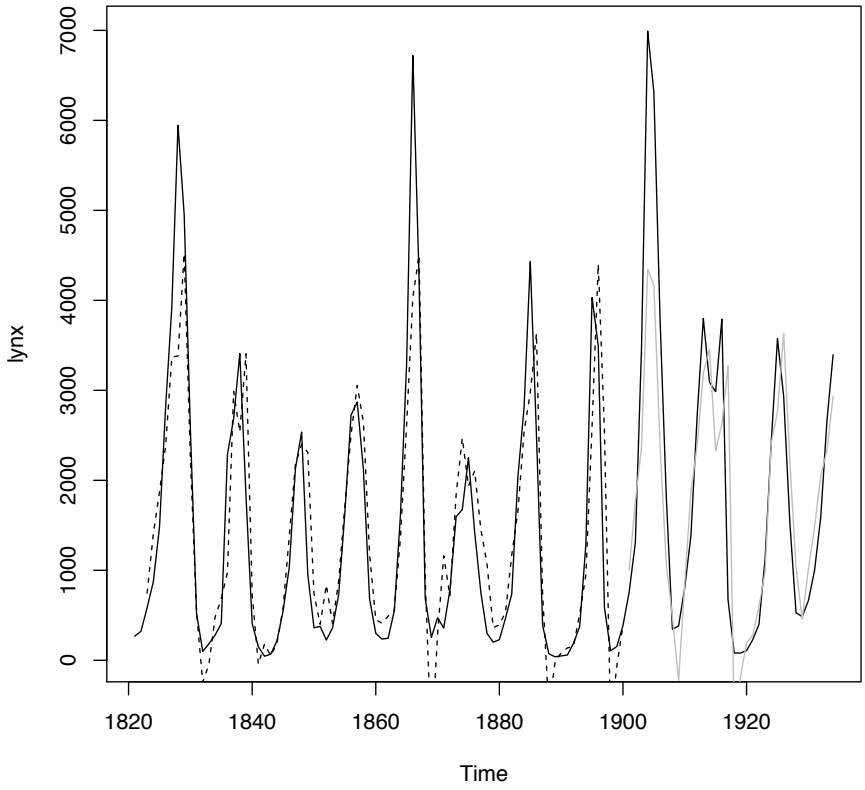

```
plot(lynx.aar2, pages = 1)
```

FIGURE 21.6: *Partial response functions for the second-order additive autoregression model of the lynx. Notice that a high count last year predicts a higher count this year, but a high count* two *years ago predicts a lower count this year. This is the sort of alternation which will tend to drive oscillations.*



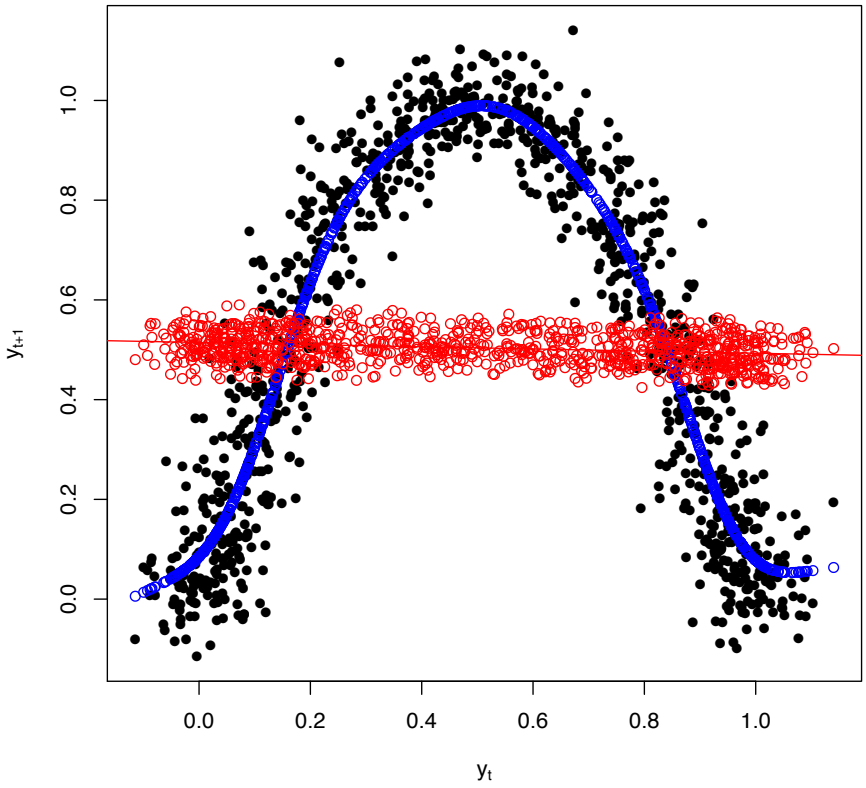

```
plot(lynx)
lines(1823:1934, fitted(lynx.aar2), lty = "dashed")
```

FIGURE 21.7: *Actual time series (solid line) and predicted values (dashed) for the second-order additive autoregression model of the lynx. The match is quite good, but of course every one of these points was used to learn the model, so it's not quite as impressive as all that. (Also, the occasional prediction of a negative number of lynxes is less than ideal.)*




```
lynx.aar2b <- aar(lynx[1:80], 2)
out.of.sample <- design.matrix.from.ts(lynx[-(1:78)], 2)
lynx.preds <- predict(lynx.aar2b, newdata = out.of.sample)
plot(lynx)
lines(1823:1900, fitted(lynx.aar2b), lty = "dashed")
lines(1901:1934, lynx.preds, col = "grey")
```

FIGURE 21.8: *Out-of-sample forecasting. The same model specification as before is estimated on the first 80 years of the lynx data, then used to predict the remaining 34 years. Solid black line, data; dashed line, the in-sample prediction on the training data; grey lines, predictions on the testing data. The RMS errors are* 723 *lynxes/year in-sample,* 922 *lynxes/year out-of-sample.*




```
library(tseries)
yar8 <- arma(y, order = c(8, 0))
points(y[-length(y)], fitted(yar8)[-1], col = "red")
```

FIGURE 21.9: *Adding the predictions of an eighth-order linear AR model (red dots) to Figure 21.5. We will see the* `arma` *function in more detail in §21.9.1.2; for now, it's enough to know that when the second component of its* `order` *argument is 0, it estimates and fits a linear AR model.*

21.4.3.1 "Unit Roots" and Stationary Solutions

Suppose we really believed a first-order linear autoregression,

$$X_{t+1} = \alpha + \beta X_t + \epsilon_t \tag{21.29}$$

with ϵ_t some IID noise sequence. Let's suppose that the mean is zero for simplicity, so $\alpha = 0$. Then

$$X_{t+2} = \beta^2 X_t + \beta\epsilon_t + \epsilon_{t+1} \tag{21.30}$$

$$X_{t+3} = \beta^3 X_t + \beta^2\epsilon_t + \beta\epsilon_{t+1} + \epsilon_{t+2}\,, \tag{21.31}$$

etc. If this is going to be stationary, it'd better be the case that what happened at time t doesn't go on to dominate what happens at all later times, but clearly that will happen if $|\beta| > 1$, whereas if $|\beta| < 1$, eventually all memory of X_t (and ϵ_t) fades away. The linear AR(1) model in fact can only produce stationary distributions when $|\beta| < 1$.

For higher-order linear AR models, with parameters $\beta_1, \beta_2, \ldots \beta_p$, the corresponding condition is that all the roots of the polynomial

$$\sum_{j=1}^{p} \beta_j z^j - 1 \tag{21.32}$$

must be outside the unit circle. When this fails, when there is a "unit root", the linear AR model cannot generate a stationary process[5].

There is a fairly elaborate machinery for testing for unit roots, which is sometimes also used to test for non-stationarity. It is not clear how much this really matters. A non-stationary but truly linear AR model can certainly be estimated[6]; a linear AR model can be non-stationary even if it has no unit roots[7]; and if the linear model is just an approximation to a non-linear one, the unit-root criterion doesn't apply to the true model anyway.

See §21.6.1 for an alternative way of checking stationarity, which presumes no particular parametric form.

21.4.4 Conditional Variance

Having estimated the conditional expectation, we can estimate the conditional variance $\mathbb{V}\left[X_t \mid X_{t-p}^{t-1}\right]$ just as we estimated other conditional variances, in Chapter 7.

Example: lynx The lynx series seems ripe for fitting conditional variance functions — presumably when there are a few thousand lynxes, the noise is going to be larger than when there are only a few hundred.

[5] The same argument applies to ARMA models (§21.9.1.2) more generally.

[6] Because the correlation structure stays the same, even as the means and variances can change. Consider $X_t = X_{t-1} + \epsilon_t$, with ϵ_t IID.

[7] Start it with X_1 very far from the expected value.

```
sq.res <- residuals(lynx.aar2)^2
lynx.condvar1 <- gam(sq.res ~ s(lynx[-(1:2)]))
lynx.condvar2 <- gam(sq.res ~ s(lag1) + s(lag2), data = design.matrix.from.ts(lynx,
    2))
```

I have fit two different models for the conditional variance here, just because. Figure 21.10 shows the data, and the predictions of the second-order additive AR model, but with just the standard deviation bands corresponding to the first of these two models; you can try making the analogous plot for `lynx.condvar2`.

21.4.5 Regression with Correlated Noise; Generalized Least Squares

Suppose we have an old-fashioned regression problem

$$Y_t = r(X_t) + \epsilon_t \tag{21.33}$$

only now the noise terms ϵ_t are themselves a dependent time series. Ignoring this dependence, and trying to estimate m by minimizing the mean squared error, is very much like ignoring heteroskedasticity. (In fact, heteroskedastic ϵ_t are a special case.) What we saw in Chapter 7 is that ignoring heteroskedasticity doesn't lead to bias, but it does mess up our understanding of the uncertainty of our estimates, and is generally inefficient. The solution was to weight observations, with weights inversely proportional to the variance of the noise.

With correlated noise, we do something very similar. Suppose we knew the covariance function $\gamma(h)$ of the noise. From this , we could construct the variance-covariance matrix $\mathbf{\Gamma}$ of the ϵ_t (since $\Gamma_{ij} = \gamma(i-j)$, of course).

We can use this as follows. Say that our guess about the regression function is m. Stacking $y_1, y_2, \ldots y_n$ into a matrix $\mathbf{y}$ as usual in regression, and likewise creating $\mathbf{m}(x)$, the Gauss-Markov theorem (Appendix J) tells us that the most efficient estimate is the solution to the **generalized least squares** problem,

$$\widehat{m}_{GLS} = \operatorname*{argmin}_m \frac{1}{n}(\mathbf{y} - \mathbf{m}(\mathbf{x}))^T \Gamma^{-1} (\mathbf{y} - \mathbf{m}(\mathbf{x})) \tag{21.34}$$

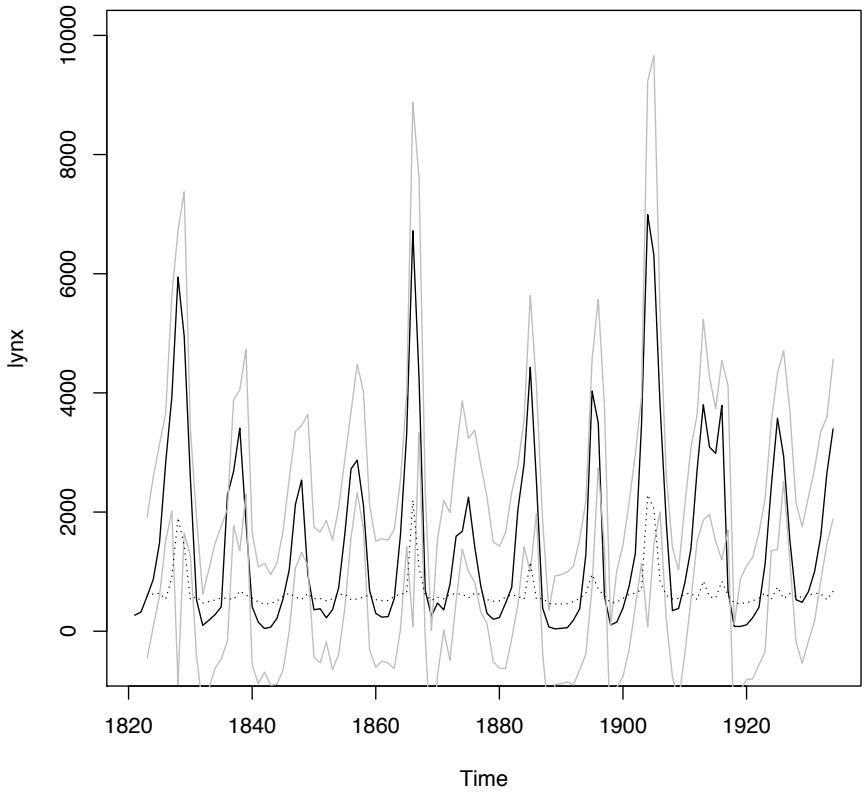
as opposed to just minimizing the mean-squared error,

$$\widehat{m}_{OLS} = \operatorname*{argmin}_m \frac{1}{n}(\mathbf{y} - \mathbf{m}(\mathbf{x}))^T (\mathbf{y} - \mathbf{m}(\mathbf{x})) \tag{21.35}$$

Multiplying by the inverse of Γ appropriately discounts for observations which are very noisy, *and* discounts for correlations between observations introduced by the noise.[8]

This raises the question of how to get $\gamma(h)$ in the first place. If we knew the true regression function r, we could use the covariance of $Y_t - r(X_t)$ across different t. Since we don't know r, but have only an estimate $\hat{m}$, we can try alternating between using a guess at γ to estimate $\hat{m}$, and using $\hat{m}$ to improve our guess at γ. We used this sort of iterative approximation for weighted least squares, and it can work here, too.

[8] If you want to use a linear model for m, this can be carried through to an explicit modification of the usual ordinary-least-squares estimate — Exercise 2.




```
plot(lynx, ylim = c(-500, 10000))
sd1 <- sqrt(fitted(lynx.condvar1))
lines(1823:1934, fitted(lynx.aar2) + 2 * sd1, col = "grey")
lines(1823:1934, fitted(lynx.aar2) - 2 * sd1, col = "grey")
lines(1823:1934, sd1, lty = "dotted")
```

FIGURE 21.10: *The lynx data (black line), together with the predictions of the additive autoregression* ± 2 *conditional standard deviations. The dotted line shows how the conditional standard deviation changes over time; notice how it ticks upwards around the big spikes in population.*

21.5 Bootstrapping Time Series

The big picture of bootstrapping doesn't change: simulate a distribution which is close to the true one, repeat our estimate (or test or whatever) on the simulation, and then look at the distribution of this statistic over many simulations. The catch is that the surrogate data from the simulation has to have the same sort of dependence as the original time series. This means that simple resampling is just wrong (unless the data are independent), and our simulations will have to be more complicated.

21.5.1 Parametric or Model-Based Bootstrap

Conceptually, the simplest situation is when we fit a full, generative model — something which we could step through to generate a new time series. If we are confident in the model specification, then we can bootstrap by, in fact, simulating from the fitted model. This is the parametric bootstrap we saw in Chapter 6.

21.5.2 Block Bootstraps

Simple resampling won't work, because it destroys the dependence between successive values in the time series. There is, however, a clever trick which does work, and is almost as simple. Take the full time series x_1^n and divide it up into overlapping blocks of length k, so $x_1^k, x_2^{k+1}, \ldots x_{n-k+1}^n$. Now draw $m = n/k$ of these *blocks* with replacement[9], and set them down in order. Call the new time series $\tilde{x}_1^n$.

Within each block, we have preserved *all* of the dependence between observations. It's true that successive observations are now completely independent, which generally wasn't true of the original data, so we're introducing some inaccuracy, but we're certainly coming closer than just resampling individual observations (which would be $k = 1$). Moreover, we can make this inaccuracy smaller and smaller by letting k grow as n grows. One can show[10] that the optimal $k = O(n^{1/3})$; this gives a growing number ($O(n^{2/3})$) of increasingly long blocks, capturing more and more of the dependence. (We will consider how exactly to pick k in the next chapter.)

The block bootstrap scheme is extremely clever, and has led to a great many variants. Three in particular are worth mentioning.

1. In the **circular block bootstrap** (or **circular bootstrap**), we "wrap the time series around a circle", so that it goes $x_1, x_2, \ldots x_{n_1}, x_n, x_1, x_2, \ldots$. We then sample the n blocks of length k this gives us, rather than the merely $n - k$ blocks of the simple block bootstrap. This makes better use of the information we have about dependence on distances $< k$.

2. In the **block-of-blocks bootstrap**, we first divide the series into blocks of length k_2, and then subdivide each of those into sub-blocks of length $k_1 < k_2$. To generate a new series, we sample blocks with replacement, and then sample

[9] If n/k isn't a whole number, round.
[10] I.e., I will not show.

```
rblockboot <- function(ts, block.length, len.out = length(ts)) {
    the.blocks <- as.matrix(design.matrix.from.ts(ts, block.length - 1, right.older = FALSE))
    blocks.in.ts <- nrow(the.blocks)
    stopifnot(blocks.in.ts == length(ts) - block.length + 1)
    blocks.needed <- ceiling(len.out/block.length)
    picked.blocks <- sample(1:blocks.in.ts, size = blocks.needed, replace = TRUE)
    x <- the.blocks[picked.blocks, ]
    x.vec <- as.vector(t(x))
    return(x[1:len.out])
}
```

CODE EXAMPLE 40: *The basic block bootstrap for univariate time series. See Exercise 4 for variants and extensions.*

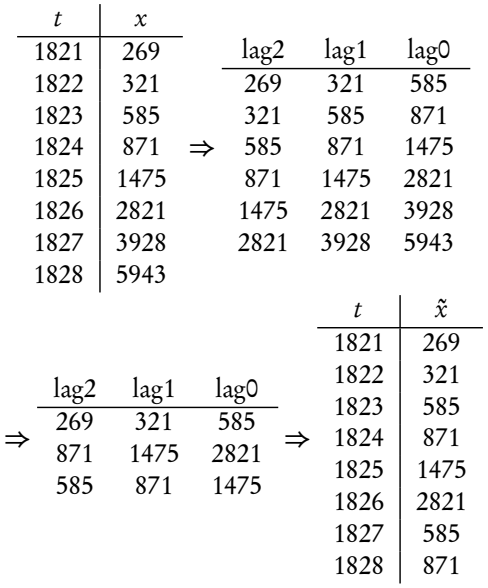
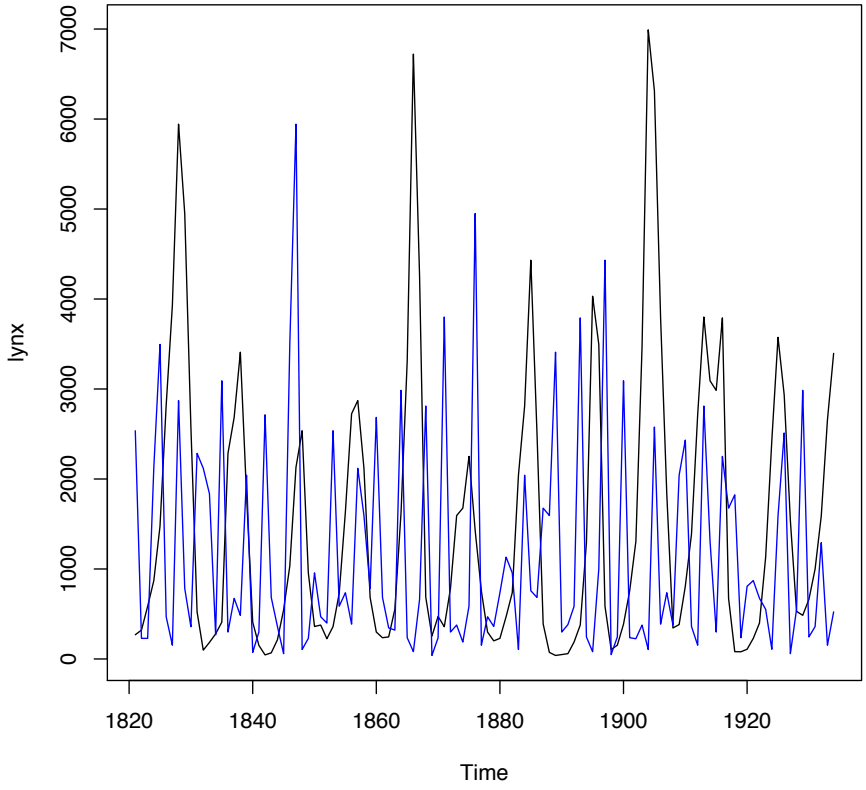


FIGURE 21.11: *Scheme for block bootstrapping: turn the time series (here, the first eight years of* `lynx`*) into blocks of consecutive values; randomly resample enough of these blocks to get a series as long as the original; then string the blocks together in order. See* `rblockboot` *online for code. [[TODO: R-ify]]*




```
plot(lynx)
lines(1821:1934, rblockboot(lynx, 4), col = "blue")
```

FIGURE 21.12: *The lynx time series, and one run of resampling it with a block bootstrap, block length* = 4. *(See online for the code to* `rblockboot`*.)*

sub-blocks within each block with replacement. This gives a somewhat better idea of longer-range dependence, though we have to pick two block-lengths.

3. In the **stationary bootstrap**, the length of each block is random, chosen from a geometric distribution of mean k. Once we have chosen a sequence of block lengths, we sample the appropriate blocks with replacement. The advantage of this is that the ordinary block bootstrap doesn't quite give us a stationary time series. (The distribution of X_{k-1}^{k} is not the same as the distribution of X_{k}^{k+1}.) Averaging over the random choices of block lengths, the stationary bootstrap does. It tends to be slightly slower to converge that the block or circular bootstrap, but there are some applications where the surrogate data really needs to be strictly stationary.

21.5.3 Sieve Bootstrap

A compromise between model-based and non-parametric bootstraps is to use a **sieve bootstrap**. This also simulates from models, but we don't really believe in them; rather, we just want them to be reasonable easy to fit and simulate, yet flexible enough that they can capture a wide range of processes if we just give them enough capacity. We then (slowly) let them get more complicated as we get more data[11]. One popular choice is to use linear AR(p) models, and let p grow with n — but there is nothing special about linear AR models, other than that they are very easy to fit and simulate from. Additive autoregressive models, for instance, would often work at least as well.

21.6 Cross-Validation

[[Straight-forward way on design matrix]]

[[Leave-out-buffers]]

[[Going-forward scheme, borrow from Fan and Yao §8.3.5 but also cite others]]

[[Fan and Yao suggest: divide series into Q chunks, with a look-ahead region of length m following the first one; re-estimate on each chunk; then average forecasting errors on the next m observations; suggest that good results are often obtained, without too much computational cost, by using $m = n/10$ and $Q = 4$ — real reason not to re-estimate every time step is just that it takes too long!]]

21.6.1 Testing Stationarity by Cross-Prediction

[[visual inspection, formal tests]]

[[TODO: Cross-ref to unit roots]]

[11] This is where the metaphor of the "sieve" comes in: the idea is that the mesh of the sieve gets finer and finer, catching more and more subtle features of the data.

21.7 Trends and De-Trending

The sad fact is that a lot of important time series are not even approximately stationary. For instance, Figure 21.13 shows US national income per person (adjusted for inflation) over the period from 1952 (when the data series begins) until now. It is *possible* that this is sample from a stationary process. But in that case, the correlation time is evidently much longer than 50 years, on the order of centuries, and so the theoretical stationarity is irrelevant for anyone but a very ambitious quantitative historian.

More sensibly, we should try to treat data like this as a non-stationary time series. The conventional approach is try separating time series like this into a persistent **trend**, and stationary **fluctuations** (or **deviations**) around the trend,

$$\begin{aligned} Y_t &= X_t + Z_t \\ \text{series} &= \text{fluctuations} + \text{trend} \end{aligned} \tag{21.36}$$

Since we could add or subtract a constant to each X_t without changing whether they are stationary, we'll stipulate that $\mathbb{E}\left[X_t\right] = 0$, so $\mathbb{E}\left[Y_t\right] = \mathbb{E}\left[Z_t\right]$. (In other situations, the decomposition might be multiplicative instead of additive, etc.) How might we find such a decomposition?

If we have multiple independent realizations $Y_{i,t}$ of the same process, say m of them, and they all have the same trend Z_t, then we can try to find the common trend by averaging the time series:

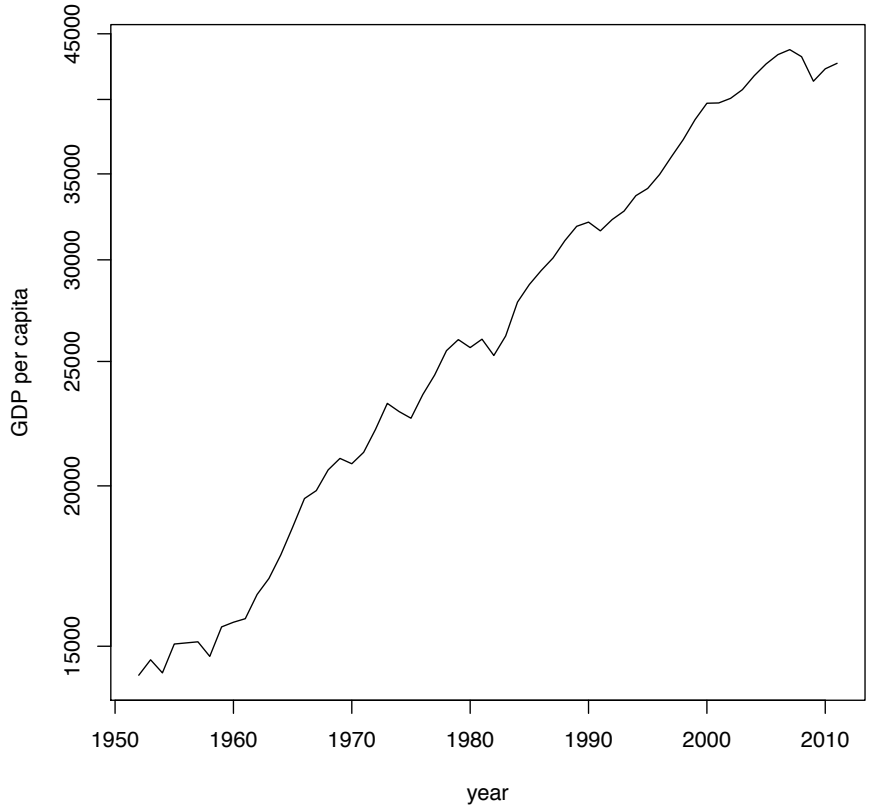
$$Z_t = \mathbb{E}\left[Y_{\cdot,t}\right] \approx \sum_{i=1}^{m} Y_{i,t} \tag{21.37}$$

Multiple time series with the same trend do exist, especially in the experimental sciences. $Y_{i,t}$ might be the measurement of some chemical in a reactor at time t in the i^{th} repetition of the experiment, and then it would make sense to average the $Y_{i,t}$ to get the common Z_t trend, the average trajectory of the chemical concentration. One can tell similar stories about experiments in biology or even psychology, though those are complicated by the tendency of animals to get tired and to learn[12].

For better or for worse, however, we have only one realization of the post-WWII US economy, so we can't average multiple runs of the experiment together. If we have a theoretical model of the trend, we can try to fit that model. For instance, some (simple) models of economic growth predict that series like the one in Figure 21.13 should, on average, grow at a steady exponential rate[13]. We could then estimate Z_t by fitting a model to Y_t of the form $\beta_0 e^{\beta t}$, or even by doing a linear regression of $\log Y_t$ on t. The fluctuations X_t are then taken to be the residuals of this model.

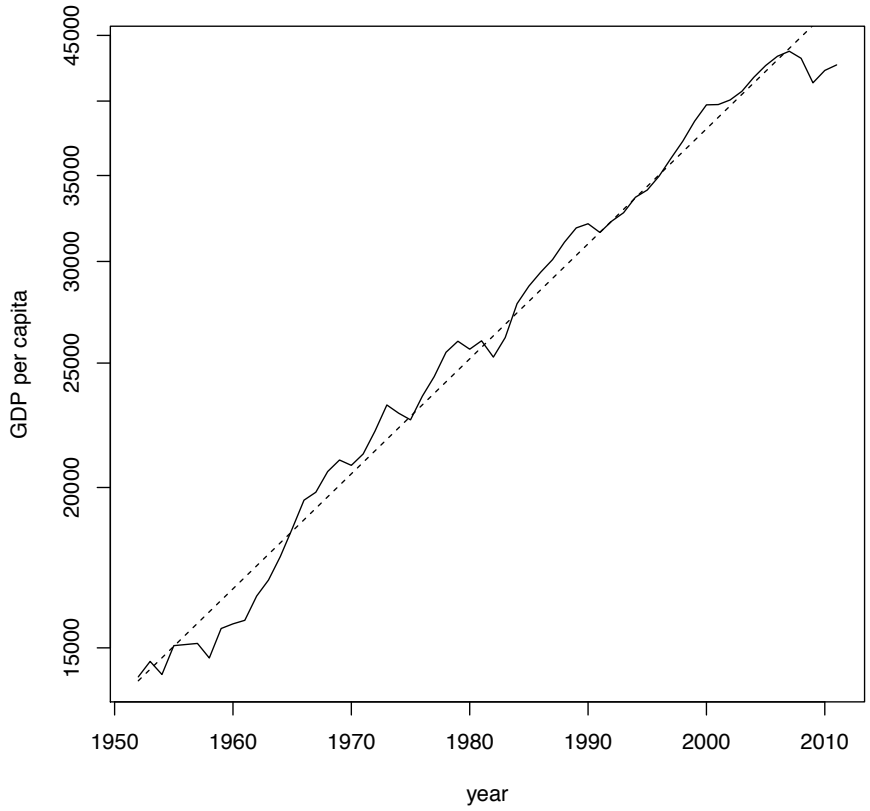
[12] Even if we do have multiple independent experimental runs, it is very important to get them aligned in time, so that $Y_{i,t}$ and $Y_{j,t}$ refer to the *same* point in time relative to the start of the experiment; otherwise, averaging them is just mush. It can also be important to ensure that the initial state, before the experiment, is the same for every run. Chu *et al.* (2003) explains how the later problem can lead to complications in studying gene regulation.

[13] This is not *quite* what is claimed by Solow (1970), which you should read anyway if this kind of question is at all interesting to you.



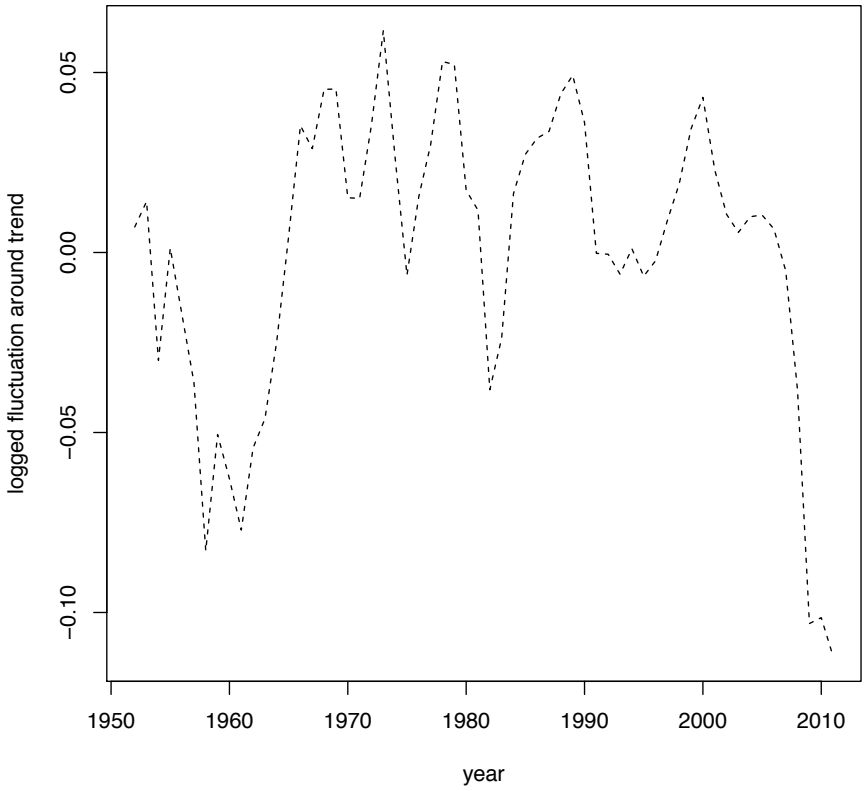

```
gdppc <- read.csv("gdp-pc.csv")
gdppc$y <- gdppc$y * 1e+06
plot(gdppc, log = "y", type = "l", ylab = "GDP per capita")
```

FIGURE 21.13: *US GDP per capita, adjusted for inflation (consumer price index deflator), with a log scale on the vertical axis. (The values were initially recorded in the file in millions of dollars per person per year, hence the correction.)*




```
gdppc.exp <- lm(log(y) ~ year, data = gdppc)
beta0 <- exp(coefficients(gdppc.exp)[1])
beta <- coefficients(gdppc.exp)[2]
curve(beta0 * exp(beta * x), lty = "dashed", add = TRUE)
```

FIGURE 21.14: *As in Figure 21.13, but with an exponential trend fitted.*




```
plot(gdppc$year, residuals(gdppc.exp), xlab = "year", ylab = "logged fluctuation around trend",
    type = "l", lty = "dashed")
```

FIGURE 21.15: *The hopefully-stationary fluctuations around the exponential growth trend in Figure 21.14. Note that these are* $\log \frac{Y_t}{\hat{\beta}_0 e^{\hat{\beta} t}}$*, and so unitless.*

If we only have one time series (no replicates), and we don't have a good theory which tells us what the trend should be, we fall back on curve fitting. In other words, we regress Y_t on t, call the fitted values Z_t, and call the residuals X_t. This is frankly rests more on hope than on theorems. The hope is that the characteristic time-scale for the fluctuations X_t (say, their correlation time τ) is short compared to the characteristic time-scale for the trend Z_t[14]. Then if we average Y_t over a band-width which is large compared to τ, but small compared to the scale of Z_t, we should get something which is mostly Z_t — there won't be too much bias from averaging, and the fluctuations should mostly cancel out.

[[TODO: Formally introduce the moving average filter here]]

Once we have the fluctuations, and are reasonably satisfied that they're stationary, we can model them like any other stationary time series. Of course, to actually make predictions, we need to extrapolate the trend, which is a harder business.

21.7.1 Forecasting Trends

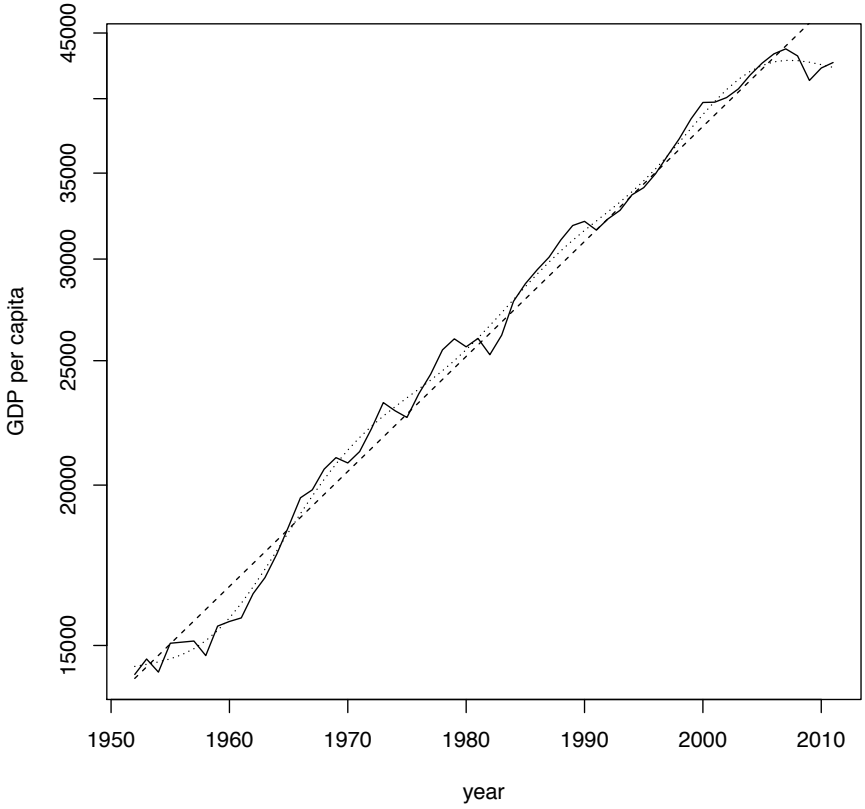
The problem with making predictions when there is a substantial trend is that it is usually hard to know how to continue or extrapolate the trend beyond the last data point. If we are in the situation where we have multiple runs of the same process, we can at least extrapolate up to the limits of the different runs. If we have an actual model which tells us that the trend should follow a certain functional form, and we've estimated that model, we can use it to extrapolate. But if we have found the trend purely through curve-fitting, we have a problem.

Suppose that we've found the trend by spline smoothing, as in Figure 21.16. The fitted spline model will cheerfully make predictions for the what the trend of GDP per capita will be in, say, 2252, far outside the data. This will be a simple linear extrapolation, because splines are always linear outside the data range (Chapter 8, p. 195). This is just because of the way splines are set up, not because linear extrapolation is such a good idea. Had we used kernel regression, or any of many other ways of fitting the curve, we'd get different extrapolations. People in 2252 could look back and see whether the spline had fit well, or some other curve would have done better. (But why would they want to?) Right now, if *all* we have is curve-fitting, we are in a dubious position even as regards next year, never mind 2252[15]

[[TODO: Mention result of Bosq (1998, §3.4.2) that just blindly doing a regression of Y_{t+1} on Y_t can actually work perfectly well in some situations, e.g., when the trend is periodic or asymptotically constant]]

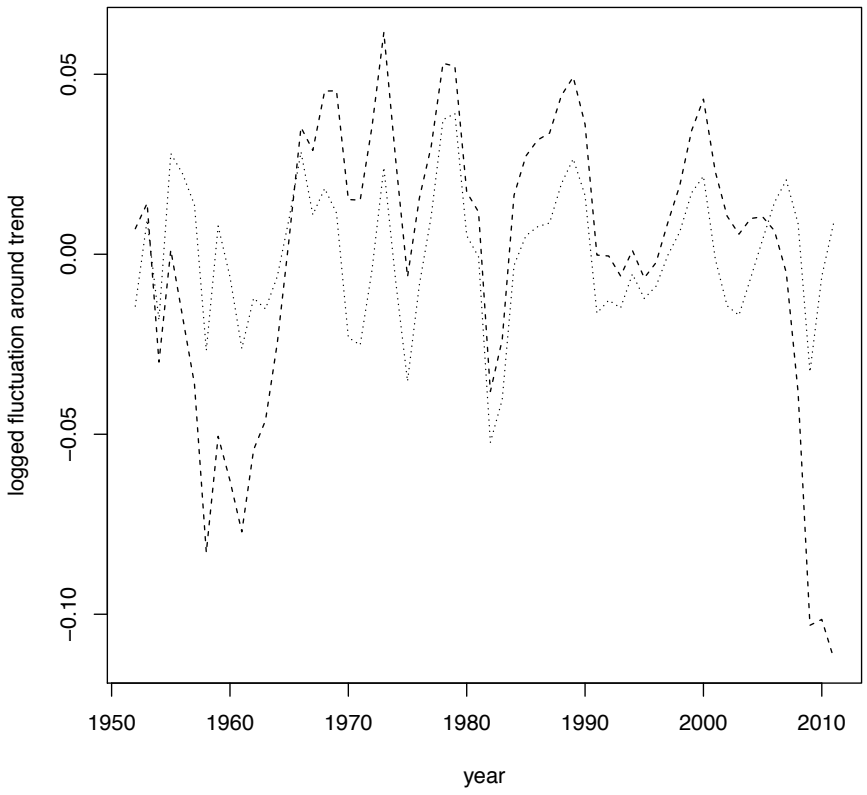
[14] I am being deliberately vague about what "the characteristic time scale of Z_t" means. Intuitively, it's the amount of time required for Z_t to change substantially. You might think of it as something like $n^{-1}\sum_{t=1}^{n-1} 1/|Z_{t+1} - Z_t|$, if you promise not to treat that too seriously. Trying to get an exact statement of what's involved in identifying trends requires being very precise, and getting into topics at the intersection of statistics and functional analysis which are beyond the scope of this class.

[15] Yet again, we hit a basic philosophical obstacle, which is the problem of induction. We have so far evaded it, by assuming that we're dealing with IID or a stationary probability distribution; these assumptions let us *deductively* extrapolate from past data to future observations, with more or less confidence. (For more on this line of thought, see Hacking (2001); Spanos (2011); Gelman and Shalizi (2013).) If we assume a certain form or model for the trend, then again we can deduce future behavior on that basis. But if we have neither probabilistic nor mechanistic assumptions, we are, to use a technical term, stuck with induction. Whether there is some principle which might help — perhaps a form of Occam's Razor (Kelly, 2007)? — is a nice question.




```
gdp.spline <- fitted(gam(y ~ s(year), data = gdppc))
lines(gdppc$year, gdp.spline, lty = "dotted")
```

FIGURE 21.16: *Figure 21.14, but with the addition of a spline curve for the time trend (dotted line). This is, perhaps unsurprisingly, not all* that *different from the simple exponential-growth trend.*




```
lines(gdppc$year, log(gdppc$y/gdp.spline), xlab = "year", ylab = "logged fluctuations around trend",
    lty = "dotted")
```

FIGURE 21.17: *Adding the logged deviations from the spline trend (dotted) to Figure 21.15.*

21.7.2 Seasonal Components

Sometimes we know that time series contain components which repeat, pretty exactly, over regular periods. These are called **seasonal** components, after the obvious example of trends which cycle each year with the season. But they could cycle over months, weeks, days, etc.

The decomposition of the process is thus

$$Y_t = X_t + Z_t + S_t \tag{21.38}$$

where X_t handles the stationary fluctuations, Z_t the long-term trends, and S_t the repeating seasonal component.

If $Z_t = 0$, or equivalently if we have a good estimate of it and can subtract it out, we can find S_t by averaging over multiple cycles of the seasonal trend. Suppose that we know the period of the cycle is T, and we can observe $m = n/T$ full cycles. Then

$$S_t \approx \frac{1}{m}\sum_{j=0}^{m-1} Y_{t+jT} \tag{21.39}$$

This works because, with Z_t out of the picture, $Y_t = X_t + S_t$, and S_t is periodic, $S_t = S_{t+T}$. Averaging over multiple cycles, the stationary fluctuations tend to cancel out (by the ergodic theorem), but the seasonal component does not.

For this trick to work, we need to know the period. If the true $T = 355$, but we use $T = 365$ without thinking[16], we can get mush.

We also need to know the over-all trend. Of course, if there are seasonal components, we really ought to subtract them out before trying to find Z_t. So we have yet another vicious cycle, or, more optimistically, another case for iterative approximation.

21.7.3 Detrending by Differencing

Suppose that Y_t has a linear time trend:

$$Y_t = \beta_0 + \beta t + X_t \tag{21.40}$$

with X_t stationary. Then if we take the difference between successive values of Y_t, the trend goes away:

$$Y_t - Y_{t-1} = \beta + X_t - X_{t-1} \tag{21.41}$$

Since X_t is stationary, $\beta + X_t - X_{t-1}$ is also stationary. Taking differences has removed the trend.

Differencing will not only get rid of linear time trends. Suppose that

$$Z_t = Z_{t-1} + \epsilon_t \tag{21.42}$$

where the "innovations" or "shocks" ϵ_t are IID, and that

$$Y_t = Z_t + X_t \tag{21.43}$$

[16] Exercise: come up with an example of a time series where the periodicity *should* be 355 days.

with X_t stationary, and independent of the ϵ_t. It is easy to check that (i) Z_t is not stationary (Exercise 3), but that (ii) the first difference

$$Y_t - Y_{t-1} = \epsilon_t + X_t - X_{t-1} \tag{21.44}$$

is stationary. So differencing can get rid of trends which are built out of the summation of *persistent* random shocks.

Differencing gives us another way of making a time series stationary: instead of trying to model the time trend, take the difference between successive values, and see if that is stationary. (The `diff()` function in R does this; see Figure 21.18.) If such "first differences" don't look stationary, take differences among differences, third differences, etc., until you have something satisfying.

Differencing is like taking the discrete version of a derivative. Repeated differencing will eventually get rid of trends if they correspond to curves (e.g., polynomials) with only finitely many non-zero derivatives. It fails for trends which aren't like that, like exponentials or sinusoids, though you can hope that eventually the higher differences are small enough that they don't matter much.

Notice that now we can continue to the trend (a little): once we predict $Y_{t+1} - Y_t$, we add it on to Y_t (which we observed) to get Y_{t+1}.

21.7.4 Cautions with Detrending

The fact that I've explained multiple different ways of detrending non-stationary time series may have made you uneasy: how are you to know which one to use? My unhelpful answer is "it depends", namely, on what you think is a plausible about the trend and the fluctuations around it. (E.g., if you think the trend is linear, then differencing should work.) My *advice* is to try several different ways of detrending your data, and to examine them very carefully if they give substantially different results.

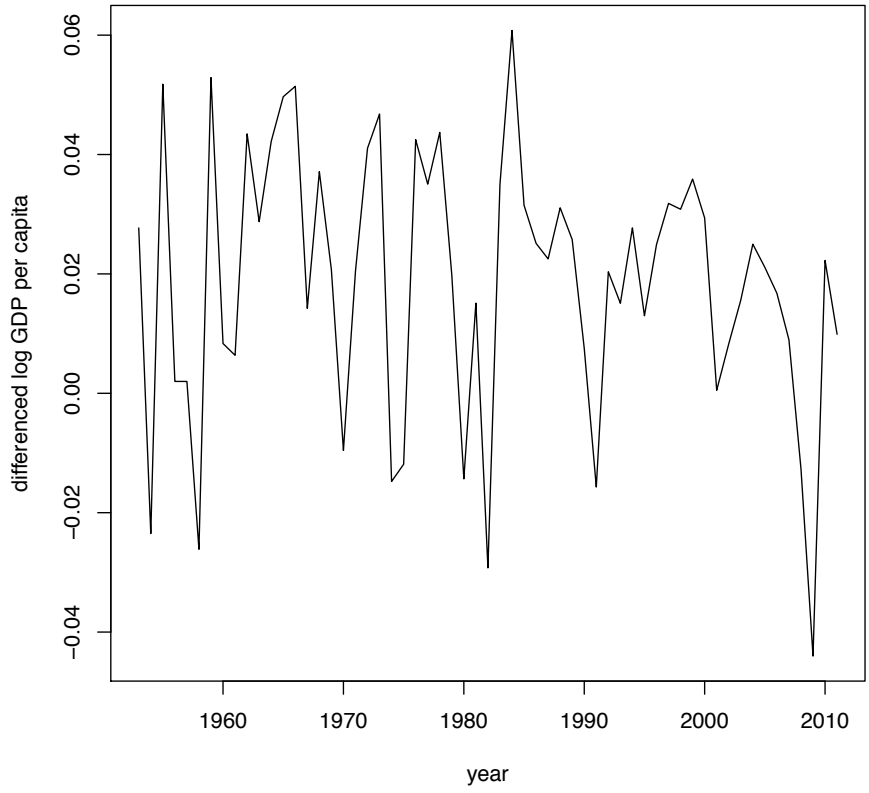
Finally, it is worth considering how much damage you might do by de-trending if the process really is stationary. E.g., if the original series is really uncorrelated, differencing will *create* correlations (Exercise 5).

See also §21.10.1 on the Yule-Slutsky effect

[[TODO: Move Yule-Slutsky here]]

21.7.5 Bootstrapping with Trends

All the bootstraps discussed in §21.5 work primarily for stationary time series. (Parametric bootstraps are an exception, since we *could* include trends in the model.) If we have done extensive de-trending, the reasonable thing to do is to use a bootstrap to generate a series of fluctuations, add it to the estimated trend, and then repeat the *whole* analysis on the new, non-stationary surrogate series, including the de-trending. This works on the same sort of principle as resampling residuals in regressions (§6.4, especially 6.4.3).




```
plot(gdppc$year[-1], diff(log(gdppc$y)), type = "l", xlab = "year", ylab = "differenced log GDP
```

FIGURE 21.18: *First differences of log GDP per capita, i.e., the year-to-year growth rate of GDP per capita.*

21.8 Breaks in Time Series

Figure 21.19 shows the employment to population ratio[17] for the US since 1990. There are fairly periodic oscillations — it's not seasonally adjusted — but it seems to be fluctuating within a not-too-wide band, and then 2008 happens, and the Lesser Depression begins.

What should we, as time series analysts, do with something like this? It goes against intuition to say that this sort of abrupt and dramatic break is all part of a single stationary process, but by this point I hope you are all thoroughly suspicious of that sort of intuition. The two big routes to dealing with series which look like this are (1) to treat them as stationary, never mind our gut, or (2) to give up on global stationarity, to say that sometimes things just change abruptly.

21.8.1 Long Memory Series

The simplest option for dealing with series that look like Figure 21.19 is to say that they are fairly ordinary stationary time series, except that the decay of dependence is very slow — that the time series has a **long memory**. A formal definition of a long-memory time series is one where the covariance function $\gamma(h) = O(h^{-\alpha})$ for some $\alpha > 0$. If α is big enough, $\sum_{h=0}^{\infty} |\gamma(h)|$ is still finite — but the slow decay of $\gamma(h)$ means that the sum, and so the correlation time, is quite large. A large correlation time means that we need to wait a very long time before any one trajectory becomes representative of the whole system — in this case, perhaps, several centuries.[18]

[[TODO: Example of a generative model]]

21.8.2 Change Points and Structural Breaks

We could of course give up on the idea that all the data come from a single stationary process. The most popular alternative is the idea of a **change point** or **structural break**. Up to some time, call it t_b, the process followed one stationary process. After this change point, it follows a different stationary process, perhaps bearing no relationship at all to what went before.

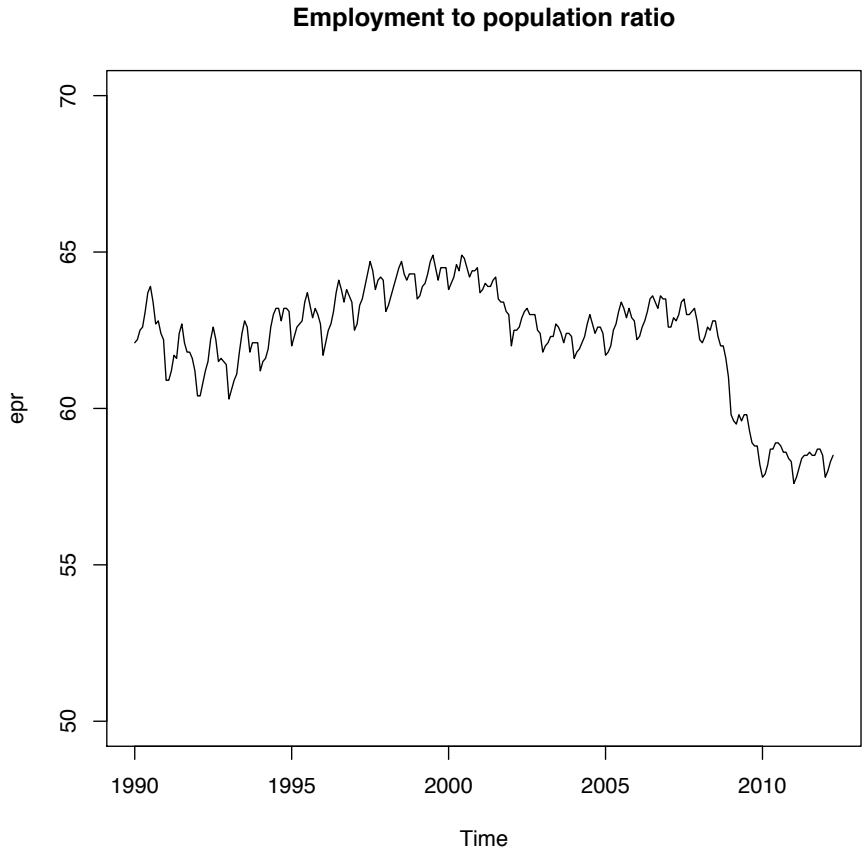
If we think we're dealing with a change point, the natural questions are, When did it change?, and What does the process look like after the change?

21.8.2.1 Change Points and Long Memory

Suppose that the change-point manifests itself by a shift in the expectation value of X_t, say from μ_1 before the change to μ_2 after. The global mean of the time series $n^{-1}\sum_t X_t$ is the somewhere between μ_1 and μ_2. If h is not too large, then for most t, X_t and X_{t+h} will be on the same side of the change point. If they are both before, then X_t and X_{t+h} will both be somewhere around μ_1, and if both times are after the

[17] That is, the ratio of the total number of employed people to all people. This is not one minus the unemployment rate, because the denominator in the unemployment rate excludes those who wouldn't be looking for paid work anyway, such as retirees.

[18] See also §21.9.1.3 on "regime switching" models.




```
epr <- read.csv("employment-pop-ratio.csv")[, 2]
epr <- ts(data = epr, start = 1990, deltat = 1/12)
plot(epr, ylim = c(50, 70), main = "Employment to population ratio", type = "l")
```

FIGURE 21.19: *Monthly employment to population ratio for the US, in percent, without seasonal adjustment. (Source: series LNU02300000 from FRED, , for 1990-01-01 to 2012-04-01 (retrieved 2012-05-04)..*

point, both values will be around μ_2. Therefore, it will tend to be the case that either both X_t and X_{t+h} are above the global mean, or both of them are below it — and so they're correlated. This argument applies even if the X_t are really all independent, as in Figure 21.20.

This phenomenon makes it *very* hard to distinguish empirically between time series which have change points and those which have a slow decay of dependence.

21.8.3 Change Point Detection

It is often reasonable to set aside such scruples, assume there are change points, and try to find them. A large number of methods have been developed for this purpose, often under very strong parametric restrictions — say that $X_t \sim_{IID} \mathcal{N}(\mu_1, \sigma^2)$ when $t < t_b$, and $X_t \sim_{IID} \mathcal{N}(\mu_2, \sigma^2)$ when $t \geq t_b$. Many of these have the flavor of looking for "runs" of values which are cumulatively very unlikely — for instance, we might look for a long run of values which are far from μ_1 and on the same side of it. Other procedures boil down to "will dividing this time series here, and letting the parameters change, work better?"

[[Cross-validation test for change points, due to Arlot and Celisse (2011)]]

[[Alternately, use the cross-prediction approach; should see a block-diagonal structure]]

21.9 Time Series with Latent Variables

[[Not all variables in time series get observed]]

[[Good reasons to add latent components: interpretability; realism; less good reason: flexibility; not good reason: soak up mis-specification error]]

[[General schemes for handling hidden variables: sum up everything we need from the past in a **state** which evolves according to a Markov process, but which we don't directly observe, thus HMM or CCC.]]

[[graphical models]]

[[The problems: (a) simulation; (b) state estimation (filtering or smoothing); (c) parameter inference (estimation or testing); (d) prediction]]

21.9.1 Examples

21.9.1.1 General Gaussian-Linear State Space Model

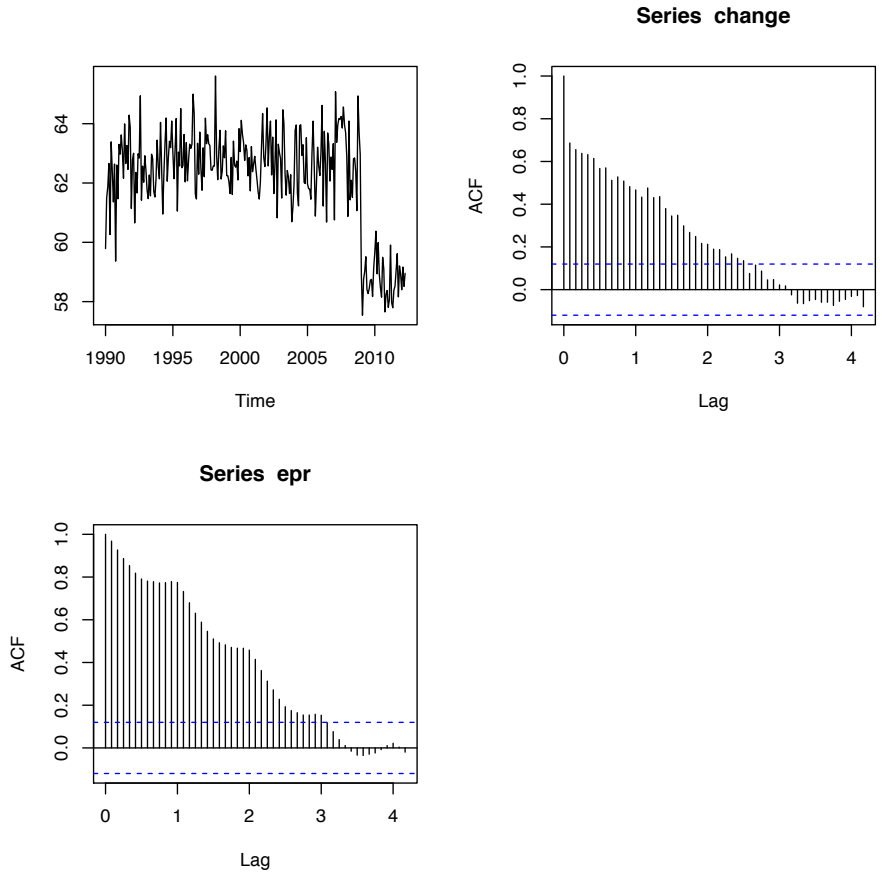
21.9.1.2 Autoregressive-Moving Average (ARMA) Models

[[Wold decomposition]]

[[Try combining AR and MA parts:

$$X_t = \beta \cdot X_{t-p}^{t-1} + Z_t + \theta_1 Z_{t-1} + \ldots + \theta_q Z_{t-q} \tag{21.45}$$

$$= \beta \cdot X_{t-p}^{t-1} + \varepsilon_t \tag{21.46}$$




```
par(mfrow = c(2, 2))
pre <- rnorm(228, mean(epr[1:228]), sd(epr[1:228]))
post <- rnorm(40, mean(epr[-(1:228)]), sd(epr[-(1:228)]))
change <- ts(c(pre, post), start = 1990, deltat = 1/12)
plot(change, ylab = "")
acf(change, lag.max = 50)
acf(epr, lag.max = 50)
par(mfrow = c(1, 1))
```

FIGURE 21.20: *A time series with a change-point. Before and after the change point, the series is an IID sequence of Gaussians, but both the expected value and the variance switch at the change-point. (These are matched to the employment-population ratio's values up to 2008 and after 2008.) The middle panel shows the resulting autocorrelation function. The bottom panel shows the actual ACF of the employment-population ratio. There is more correlation in the data than the change-point alone can account for, but it comes surprisingly close.*

where ε_t is serially correlated and correlated with X_t]]

[[but

$$\mathbb{E}\left[X_t | X_{t-p}^{t-1}\right] - X_t$$

is always uncorrelated with X_{t-p}^{t-1}, so this won't estimate β — OTOH it's not clear if that's actually a problem for fixed-memory prediction!]]

[[The σARMA model from UAI in 2004 to simplify stuff...]]

21.9.1.3 Regime Switching

Hidden-state models give us another way of dealing with apparent non-stationarity, in addition to change-points and long memory processes (§21.8), namely **regime switching**. The idea is that there observed time series is in some sense driven or controlled by a discrete latent variable, the **regime**, and can show very different dynamics in different regimes. The regime itself evolves according to its own dynamics, often taken to be Markovian. If every regime has a high probability of transitioning to itself, we will see long stretches of time where the observables seem to follow one stationary process, punctuated by rare but rapid transitions to what looks like a realization of a different stationary process. If the Markov chain for regimes is stationary, the over-all process will also be stationary, but one would, so to speak, need to look over very long time scales to see it.

[[Concrete model]]

[[References]]

21.9.1.4 Noisily-Observed Dynamical Systems

[[Define]]

[[Example with, say, Lotka-Volterra but observe Poisson counts based on true values]]

21.9.2 State Estimation

[[definitions; filtering, smoothing]]

[[formal solution via Bayes's rule]]

$$p(x_1^n | s_1^n) = \prod_{t=1}^{n} p(x_t | s_t) \tag{21.47}$$

$$p(x_1^n) = \sum p(x_1^n | s_1^n) p(s_1^n) \tag{21.48}$$

$$p(s_1^n | x_1^n) = \frac{p(x_1^n | s_1^n) p(s_1^n)}{p(x_1^n)} \tag{21.49}$$

$$= \frac{p(x_1^n | s_1^n) p(s_1^n)}{\sum p(x_1^n | s_1^n) p(s_1^n)} \tag{21.50}$$

which is unpleasant-looking, but there are tricks...

[[Limited special cases: Kalman filter, Baum-Welch algorithm]]
[[Mention deterministic approximation (e.g., LGF) but don't go into details]]

21.9.2.1 Particle Filtering

[[A direct stochastic implementation of the formal solution]]
[[Start with an initial guess set of particles]]
[[Move each particle independently according to the Markovian state evolution]]
[[Calculate likelihood of next evolution per the observation model]]
[[Resample particles with weights proportional to the likelihoods]]
[[Repeat]]
[[Demo: Lotka-Volterra?]]

21.9.2.2 Parameter Estimation

[[maximize some tractable approximation to the likelihood]]
[[Generally, EM]]
[[Need/want to infer hidden state, also to optimize parameters]]
[[e.g., particle Monte Carlo EM]]
[[Numerical details involved; punt by reference to Douc/Moulines/Stoffer]]

Much of the effort of the EM algorithm and of particle filtering goes into estimating the time-evolution of the latent state. If what we are willing to ignore that, and just focus on estimating the parameters, we can sometimes save greatly on time and effort by using techniques of simulation-based inference, basically adjusting the parameters until simulated trajectories of the model look like the data; see Chapter 23 for details. We could then always go back and estimate the states for *one* parameter value.

21.9.2.3 Prediction

[[Take the filtering distribution and extrapolate it forward via simulation]]

21.10 Moving Averages and Cycles

[[TODO: Originally written as start of a separate chapter; integrate into new context]]

[[Moving average, what it is]]

The basic equation for a moving average (MA) model of order q is

$$X_t = Z_t + \sum_{i=1}^{q} \theta_i Z_{t-i} \tag{21.51}$$

with the Z_t being IID noise terms. That is, what we observe is a weighted average[19] of the $q+1$ most recent noise variables.

[19] The right-hand side would look more like a weighted *average* if we wrote it $X_t = \frac{Z_t + \sum_{i=1}^{q} \theta_i Z_{t-i}}{1+\sum_{i=1}^{q} \theta_i}$, but since the Z_t are latent we could just re-scale each of them by the denominator. (Likewise, we can always impose weight 1 on the most recent Z.)

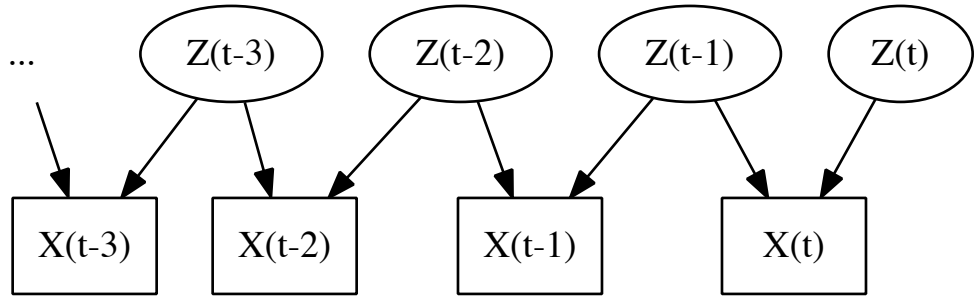


FIGURE 21.21: *The DAG for a first-order moving average model.*

Figure 21.21 shows the graphical model for an MA(1) model. It's evident from it that $X_t \not\perp\!\!\!\perp X_{t-1}$, but $X_t \perp\!\!\!\perp X_{t-k}$, $k > 1$ — observables are only dependent on each other through the hidden noise variables, and X_t and X_{t-k} have no common parents. In general, in an MA(q), $X_t \perp\!\!\!\perp X_{t-k}$ when $k > q$.

Suppose that we try to predict X_t from its past values. We condition X_t on X_{t-1}, and ask whether there is still more information to be had about X_t from X_{t-2}. This is asking whether X_t and X_{t-2} are dependent, given X_{t-1}. The answer is clearly yes from Figure 21.21: there is one path linking X_t to X_{t-2}, and X_{t-1} is a collider on that path, so conditioning on it activates the path.

Why does X_{t-2} give us information about X_t, conditional on X_{t-1}? To determine X_t, we'd need to know Z_t and Z_{t-1}. Since X_{t-1} is a child of Z_{t-1} and Z_{t-2}, knowing X_{t-1} tells us something about Z_{t-1}, but we learn even more from *also* knowing X_{t-2}.

Nothing daunted, we try conditioning X_t on X_{t-2}^{t-1}. Is $X_t \perp\!\!\!\perp X_{t-3} | X_{t-2}^{t-1}$? Clearly not. There is again only a single path, which goes over two colliders — and we condition on both of them, activating the path. Knowing X_{t-3} would tell us more about Z_{t-3}, and that, with X_{t-2}, tells us more about Z_{t-2}, which, together with X_{t-1}, helps us pin down Z_{t-1} even better. The chain of inferences is getting longer and longer, but it's not breaking, and it's evident that it will never break, no matter how many steps back into the past we condition.

To sum up, an MA(1) process, and by extension any MA(q), is not Markov, no matter what order of Markov chain we consider. Nonetheless, all of the dependence of future on the past is carried by a simple, low-dimensional variable, (Z_{t-1}, Z_t). Conditional on that, X_t is independent of all other X_s's[20].

[20]Because Z_{t-1} and Z_t are the only parents of X_t, which has no descendants.

21.10.1 Yule-Slutsky

That applying a moving average to independent noise creates a process with complicated dependence was noticed independently by two pioneers of time series analysis, [[G. Udny Yule]] and [[E. Slutsky]]. It is therefore known as the **Yule-Slutsky effect**. But Yule and Slutsky gave very different interpretations to it — both are valid in their own circumstances, but the contrast is instructive.

Slutsky Slutsky was primarily interested under the fluctuations of the economy — in the business cycle. The way he thought of a moving average process was that the economy is (under capitalism) continually subjected to random, unpredictable shocks, but it *takes time* for the economy to respond to them, for them to work through the system, as it were. The coefficients θ represent how the economy responds over time to any given shock. That this leads to fluctuations with a characteristic amplitude and (nearly) duration was a feature, not a bug — it was how Slutsky proposed to *explain* the business cycle[21]. It is not at all clear that any subsequent theory of the business cycle has any more *predictive* power.

Yule Moving averages are of course a very common way of smoothing time series. We can think of them as being rather like kernel smoothing, but with a one-sided kernel. That is, we start with our original data Z_t, and then average it together locally to get a smoother series X_t, with some of the noise removed. What Yule recognized is that doing this will, all by itself, create correlations among the X_t (cf. Chapter 4), and complicated predictive relationships. Indeed, even if the Z_t are all independent of each other, the X_t will be correlated, and will have non-zero linear regression coefficients (or other regression functions, if you use them). Part of what we infer on the X_t is then just the effects of our smoothing.

This Yule effect is very basic, and very easy to understand as soon as one sees Figure 21.21, but it continues to trip up researchers in a wide range of applied fields[22] Don't be like that.

21.11 Longitudinal Data

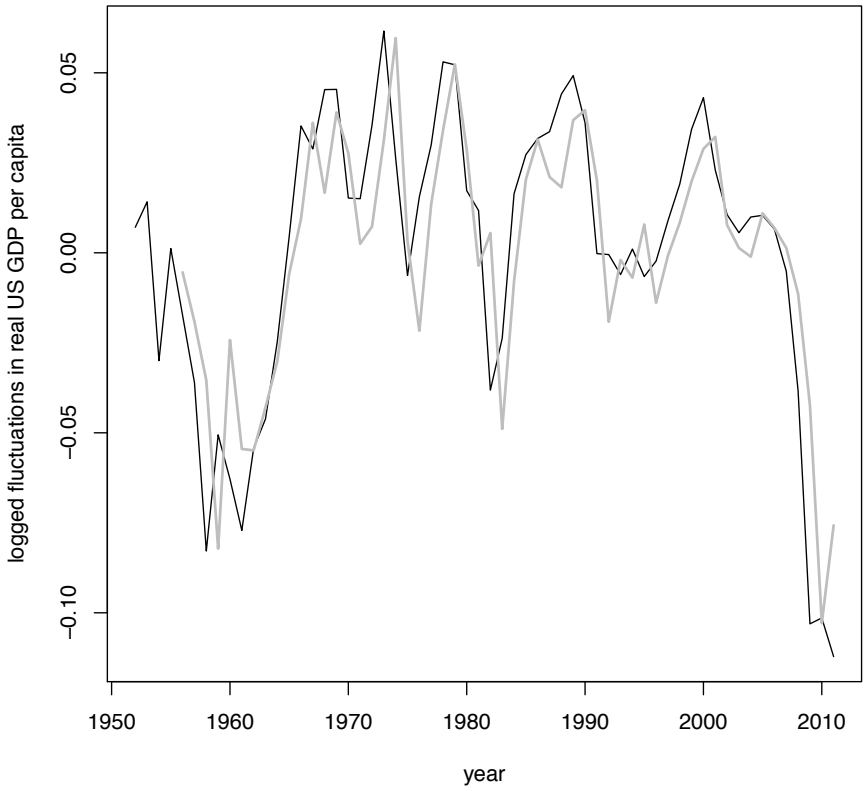
[[Basically: multiple, not necessarily stationary, time series (e.g., growth curves)]]

21.12 Multivariate Time Series

[[dynamic "Bayesian" networks]]
[[Granger causality, a distraction]]
[[Real causality a possibility]]

[21] The USSR in the 1920s being what it was, Slutsky had to do some fast talking to try to reconcile this with Marxism, and was lucky to be allowed to escape into pure probability theory.

[22] For instance, Martindale (1990); see discussion at `http://bactra.org/weblog/666.html`.




```
gdppc.ma4 <- arma(x = residuals(gdppc.exp), order = c(0, 4))
plot(1952:2011, residuals(gdppc.exp), type = "l", xlab = "year", ylab = "logged fluctuations in real US GDP
lines(1952:2011, fitted(gdppc.ma4), col = "grey", lwd = 2)
```

FIGURE 21.22: *Logged fluctuations for the United States's GDP per capita (with exponential trend removed, as in Figure 21.15), versus a fourth-order moving average model. (Since each unit of time is a quarter, four quarters is a year.) The mean squared-error, in sample, is* $4.5 \times 10^{-}4$*, corresponding to an* R^2 *of* 0.72*. [[TODO: replace numbers with R]]*

21.13 Further Reading

Shumway and Stoffer (2000) is a good introduction to conventional time series analysis, covering R practicalities. In particular, it includes both ARMA models, and the very important subject of frequency-domain methods, which I have deliberately omitted because it relies on Fourier analysis, otherwise not needed for this book. Lindsey (2004) surveys a broader range of situations in less depth; it is readable, but opinionated, and I don't always agree with the opinions. Fan and Yao (2003) is a deservedly-standard reference on nonparametric time series models. The theoretical portions would be challenging for most readers of this book, but the methodology isn't, and it devotes about the right amount of space (no more than a quarter of the book) to the usual linear-model theory. Douc *et al.* (2014) plays a similar role for parametric nonlinear statistical models; part II in particular is a self-contained treatment of stochastic process theory, and part III of particle filters.

The best introduction to stochastic processes I know of, by a very wide margin, is Grimmett and Stirzaker (1992). However, like most textbooks on stochastic processes, it says next to nothing about how to use them as models of data. A notable exception is the excellent Guttorp (1995), which both introduces the theory of a range of highly-applicable stochastic processes, and covers their statistical inference with real scientific examples. Bartlett (1955), while similar in intent, is old enough that it now makes a better second book than a first.

The basic ergodic theorem in §21.2.2.1 follows a continuous-time argument in Frisch (1995); see Exercise 6 for an extension to non-stationary processes. My general treatment of ergodicity is heavily shaped by Gray (1988) and Shields (1996).

The block bootstrap was introduced by Künsch (1989). Davison and Hinkley (1997, §8.2) has a characteristically-clear treatment of the main flavors of bootstrap for time series; Lahiri (2003) is thorough but theoretical. Bühlmann (2002) is also useful.

[[CV for time series references]]

ARMA models have spawned a huge number of modifications, extensions, and re-interpretations. Holan *et al.* (2010) is a recent survey of this "alphabet soup" of a lineage.

In parallel to the treatment of time series by statisticians, physicists and mathematicians developed their own tradition of time-series analysis (Packard *et al.*, 1980), where the basic models are not stochastic processes but deterministic, yet unstable, dynamical systems. Perhaps the best treatment of this are Abarbanel (1996); Kantz and Schreiber (2004). There are in fact very deep connections between this approach and the question of why probability theory works in the first place (Ruelle, 1991), but that's not a subject for *data analysis*.

[[Point-process references]]

21.14 Exercises

1. Write a function which takes in a time series X and makes a plot of X_{t+1} versus X_t, as in Figure 21.3. *Hint:* Use Code Example 38.

2. In Eq. 21.34, assume that $m(x)$ has to be a linear function, $m(x) = \beta \cdot x$. Solve for the optimal β in terms of $\mathbf{y}$, $\mathbf{x}$, and $\mathbf{\Gamma}$. This "generalized least squares" (GLS) solution should reduce to ordinary least squares when $\Gamma = \sigma^2 \mathbf{I}$.

3. If $Z_t = Z_{t-1} + \epsilon_t$, with ϵ_t IID, prove that Z_t is not stationary. *Hint:* consider $\mathbb{V}\left[Z_t\right]$.

4. Start with `rblockboot` from Code Example 40.

   (a) Modify the function to perform the circular block bootstrap. (*Hint:* Extend `ts`.)

   (b) Modify the function to work with multivariate time series, given as an array with time points as the rows and variables as the columns. Ensure that the *same* blocks are used for all variables, to preserve dependencies across them.

   (c) Modify the function to work with multivariate time series, given as a collection of univariate time series. Again, make sure the same blocks are used for all series. (*Hint:* Reduce to the previous sub-exercise.)

5. Suppose that X_i are IID, but we difference them and so look at $Y_i = X_i - X_{i-1}$. Find the autocovariance function of the Y series, in terms of the moments of the X_i.

6. *A non-stationary ergodic theorem* Suppose that the X_t are non-stationary, but they all have finite (not necessarily equal) means $\mathbb{E}\left[X_t\right]$, and finite covariances $\mathrm{Cov}\left[X_t, X_s\right]$. Define

$$m_n \equiv \frac{1}{n}\sum_{t=1}^{n}\mathbb{E}\left[X_t\right] \tag{21.52}$$

   and

$$V_n \equiv \sum_{t=1}^{n}\sum_{s=1}^{n}\mathrm{Cov}\left[X_t, X_s\right] \tag{21.53}$$

   Show that if $V_n = o(n^2)$, then

$$\mathbb{E}\left[\left(m_n - \overline{X}_n\right)^2\right] \to 0 \tag{21.54}$$

   and so that $\overline{X}_n \to m_n$. Does this result imply Eq. 21.15 under the conditions of §21.2.2.1? Could you deduce this result from Eq. 21.15?