STAT 436 / 536 - Lecture 7: Key

September 26, 2018

Stochastic Models

- Thus far we have seen two approaches for estimating a time series.
  1. The `decompose` function estimates the trend and seasonal patterns for a time series.

  2. The `HoltWinters` function uses exponentially weighted averages to estimate the mean, trend, and seasonal components.

- When fitting time series models, most of the deterministic features of the time series can be captured in various ways, but regardless of the approach, we still have a

- Sometimes the deterministic features capture the time series behavior, so that the residual error series is

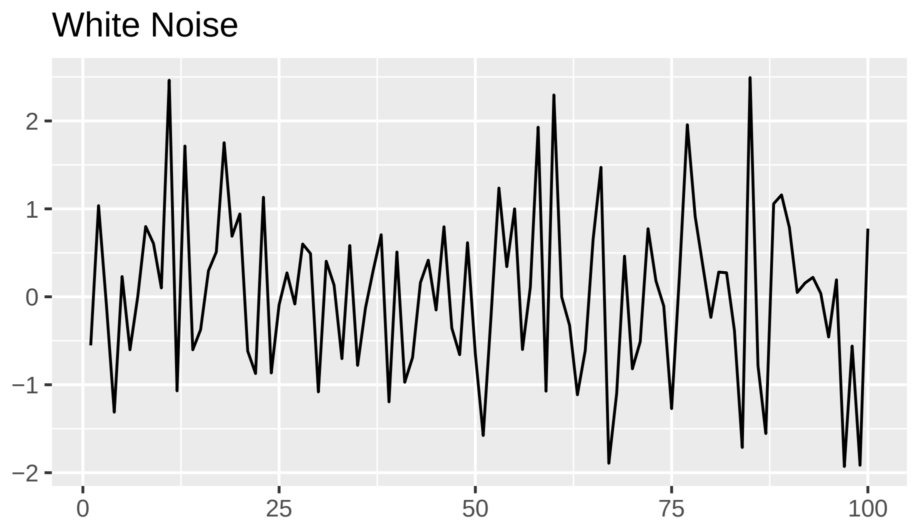
- Otherwise, if the residual error series contains

White Noise

- If the time series model is defined for value y, then the residual time series can be defined as:

- A time series exhibits white noise if $x_t = w_t$, where w_t are

```
set.seed(09192018)
library(dplyr)
library(ggfortify)
rnorm(100) %>% as.ts() %>% autoplot() + ggtitle('White Noise')
```



-The second order properties for white noise are: the mean term

- the covariance $\gamma_k(w_t, w_{t+k}) = 0$

- the correlation $\rho_k(w_t, w_{t+k}) = 0$

- Q: will the sample correlation necessarily be zero from a simulation?

```r
w <- ts(rnorm(100))
acf.obj <- acf(w)
acf.obj
```

- When fitting this model, what parameters would we need to estimate?

Random Walks

- Let $\{x_t\}$ be a time series object, then this is a random walk if...

- Using back substitution, this series can be written as:

- The textbook defines **B**

- The second order properties of the random walk are $\mu_x = 0$ and $\gamma_k(t) = t\sigma^2$ (Note this is on HW4).

- The autocorrelation $\rho_k(t) = \frac{1}{\sqrt{1+k/t}}$

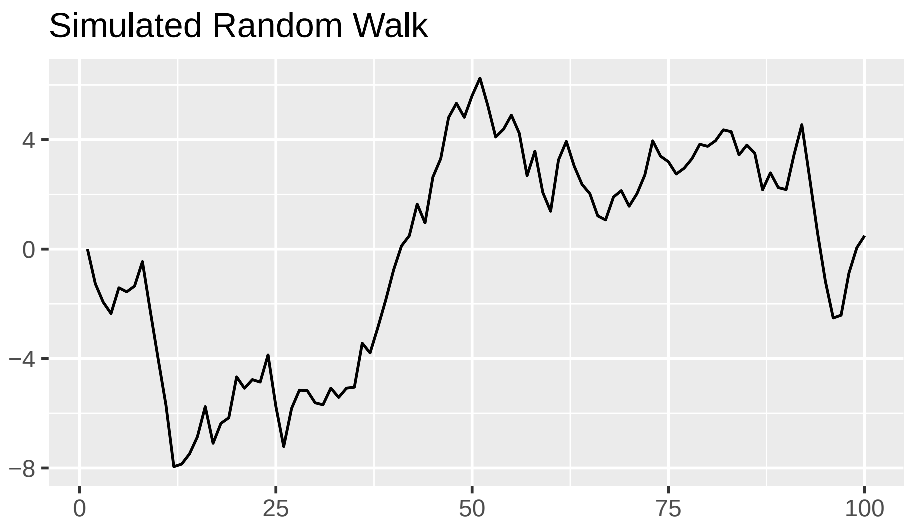
- Note this results in a non-stationary time series as the covariance depends on t.

- A common approach with a non-stationary time series, such as a random walk, is to take the difference between consecutive time points. This is denoted as

- **Q:** what is the resulting time series after applying the differencing operator to a random walk time series $\{x_t\}$?

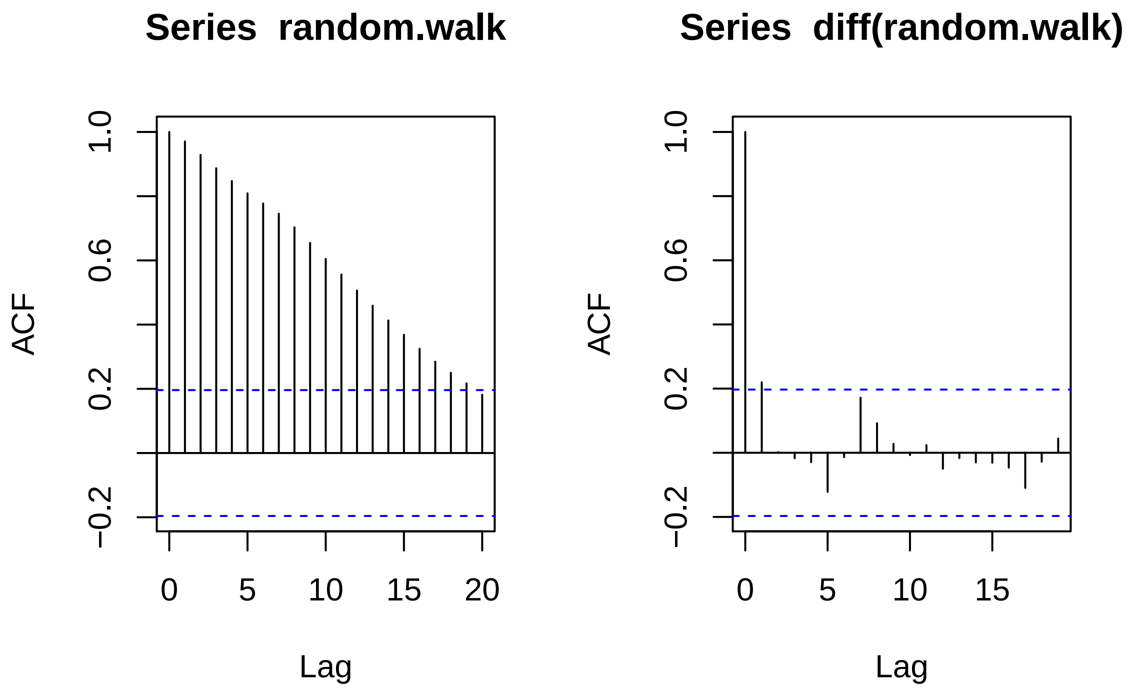
- Sketch out pseudocode to simulate a random walk.

```
time.pts <- 100
random.walk <- rep(0,time.pts)
sigma.w <- 1
for (t in 2:time.pts){
  random.walk[t] <- random.walk[t-1] + rnorm(sigma.w)
}
random.walk %>% as.ts() %>% autoplot() + ggtitle('Simulated Random Walk')
```

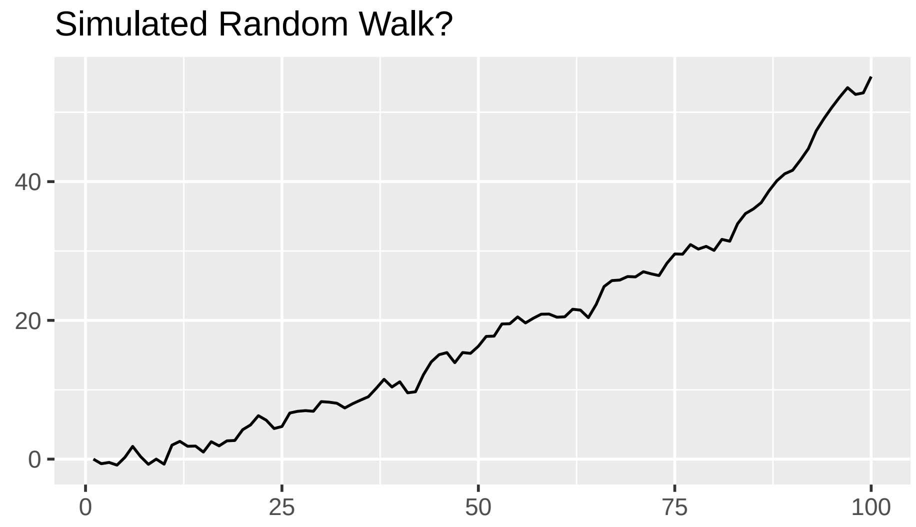


- ACF plots for random walk and differenced random walk.

```
par(mfcol=c(1,2))
acf(random.walk); acf(diff(random.walk))
```



- In some situations, a purely random walk model may not be appropriate. Consider the following figure.



- This is a random walk

- The

```
drift.est <- random.walk.drift %>% diff() %>% mean() %>% round(digits = 2)
```

where the estimate of δ is 0.56.

- The Holt-Winters function can be used to estimate both of these time series data sets.

Random Walk

```
HW.rw <- HoltWinters(random.walk, gamma = FALSE, beta = FALSE)
HW.rw$alpha
```

```
## [1] 0.9999428
```

```
random.walk[time.pts]
```

```
## [1] 0.4902215
```

```
rw.pred <- predict(HW.rw, n.ahead = 5, prediction.interval = T);
rw.pred
plot(HW.rw, rw.pred)
```

Random Walk with Drift

```
HW.rw.drift <- HoltWinters(random.walk.drift, gamma = FALSE)
HW.rw.drift$alpha; HW.rw.drift$coefficients['b']
```

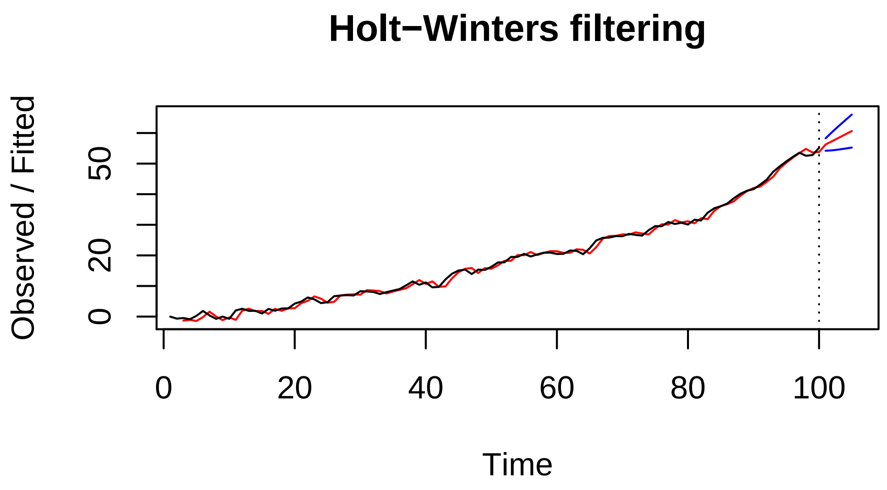
```
##     alpha
## 0.9940048
```

```
##        b
## 1.102261
```

```
rw.drift.pred <- predict(HW.rw.drift, n.ahead = 5, prediction.interval = T)
rw.drift.pred
```

```
## Time Series:
## Start = 101
## End = 105
## Frequency = 1
##          fit      upr      lwr
## 101 56.23373 58.26129 54.20618
## 102 57.33599 60.33236 54.33963
## 103 58.43826 62.27540 54.60111
## 104 59.54052 64.16870 54.91233
## 105 60.64278 66.04096 55.24460
```

```
plot(HW.rw.drift, rw.drift.pred)
```


Holt−Winters filtering
Observed / Fitted
Time