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 *residual error series, or the random component.*

- Sometimes the deterministic features capture the time series behavior, so that the residual error series is *white noise.*

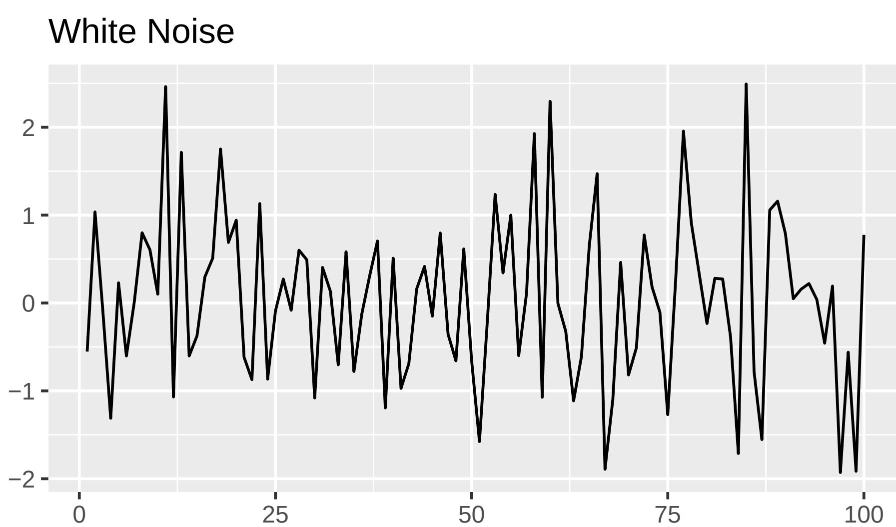
- Otherwise, if the residual error series contains *structure*, this can be exploited to improve forecasts.

White Noise

- If the time series model is defined for value y, then the residual time series can be defined as: *$x_t = y_t - \hat{y}_t$.

- A time series exhibits white noise if $x_t = w_t$, where w_t are *independent and identically distributed with mean 0. Thus* $Cor(w_i, w_j) = 0 \; \forall \; i, j$

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



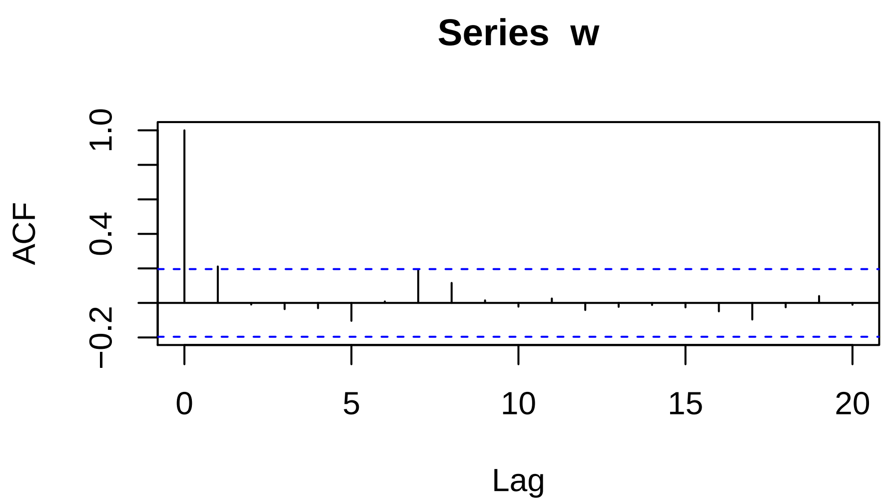
-The second order properties for white noise are: the mean term $\mu_w = 0$

- the covariance $\gamma_k(w_t, w_{t+k}) = 0$ *if* $k \neq 0$ *and* σ^2 *otherwise*

- the correlation $\rho_k(w_t, w_{t+k}) = 0$ *if* $k \neq 0$ *and 1 otherwise.*

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

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




```
acf.obj
```

```
##
## Autocorrelations of series 'w', by lag
##
##      0      1      2      3      4      5      6      7      8      9
##  1.000  0.211 -0.009 -0.036 -0.030 -0.104  0.009  0.194  0.116  0.015
##     10     11     12     13     14     15     16     17     18     19
## -0.022  0.026 -0.041 -0.023 -0.012 -0.026 -0.049 -0.096 -0.025  0.040
##     20
## -0.011
```

- When fitting this model, what parameters would we need to estimate? *Assuming the errors come from a normal distribution, the only necessary parameter is the variance,* σ^2.

Random Walks

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

$$x_t = x_{t-1} + w_t$$

  where w_t is white noise.

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

$$x_t = (x_{t-2} + w_{t_1}) + w_t$$

$$x_t = w_1 + w_2 + \cdots + w_t + x_0$$

- The textbook defines **B** *as the backward shift operator, such that* $\mathbf{Bx_t} = \mathbf{x_{t-1}}$ *and* $\mathbf{B^n x_t} = \mathbf{x_{t-n}}$

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

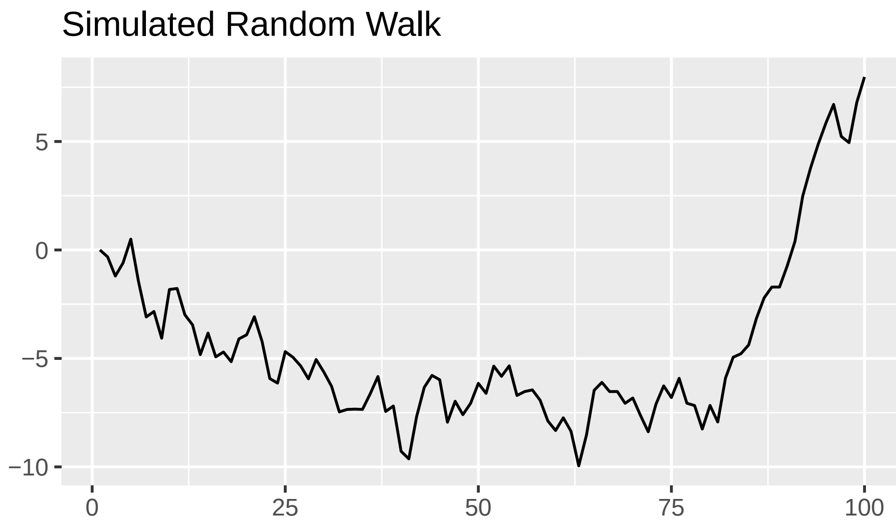
- Note this results in a non-stationary time series as the covariance depends on t.

- 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 $\bigtriangledown x_t = x_t - x_{t-1}$.

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

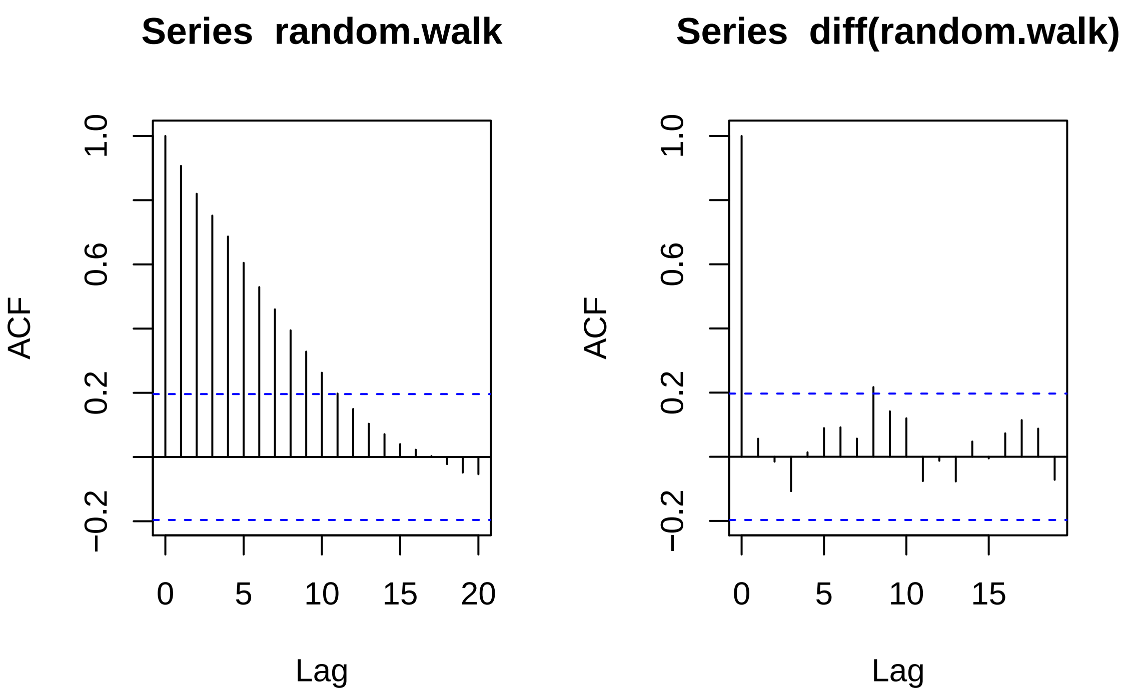
- Sketch out pseudocode to simulate a random walk.

```
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')
```

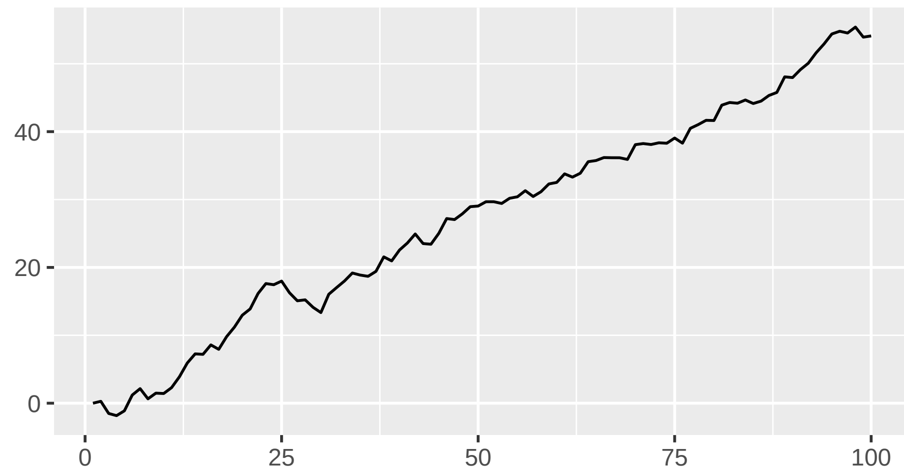


- ACF plots for random walk and differenced random walk.

```
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 *with a drift term*

$$x_t = x_{t-1} + \delta + w_t,$$

where δ is the drift parameter.

- The *drift term can be estimated from the differenced series*,

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

where the estimate of δ is 0.55.

- 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] 0.9999586
```

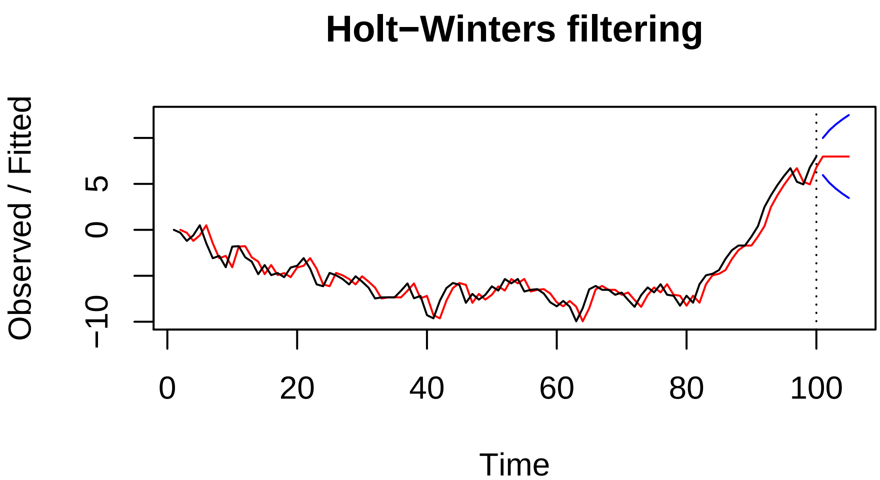
```
random.walk[time.pts]
```

```
## [1] 7.975058
```

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

```
## Time Series:
## Start = 101
## End = 105
## Frequency = 1
##          fit       upr      lwr
## 101 7.975009  9.993004 5.957014
## 102 7.975009 10.828826 5.121192
## 103 7.975009 11.470182 4.479835
## 104 7.975009 12.010874 3.939144
## 105 7.975009 12.487234 3.462784
```

```
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
##     1
```

```
##         b
## 0.5032844
```

```
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 54.60496 56.57987 52.63004
## 102 55.10824 57.92832 52.28816
## 103 55.61153 59.09873 52.12432
## 104 56.11481 60.18008 52.04954
## 105 56.61810 61.20648 52.02971
```

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