- Read the results in CC 3.4 for behavior of the different estimators, but I am going to illustrate the results using the examples from before.

- They essentially argue here that we should choose the seasonal model that produces the smallest prediction error and find that cosine trends have less variability since they share more information.

  - Seasonal means model only has information on *n*f* observations and has a coefficient for each point in the cycle
    - There is no restriction on the pattern of the seasonality (including no continuity from one point in the cycle to another)
    - Uses maximum model degrees of freedom to estimate the seasonal component
  - The cosine trend is fit through all the points and only has two coefficients
    - but it is a pretty restrictive model
  - Suite of sines/cosines and circular spline effects are somewhere between the two approaches… at least two df used and at most the same as the seasonal means.

- There is another direction this section could have taken… Select the simplest model that can adequately capture the seasonal behavior and accommodate the autocorrelation in the process.
  - As we saw before, if we can hit upon the true autocorrelation process then we can use `gls()` to get SEs that match the theoretical SEs for that process pretty well
  - Inference based on selected models tends to be overly optimistic (at least regarding tests) due to the model selection process
    - Using a model that only contains "useful" terms, hypothesis tests will tend to have smaller p-values than they should and reject the null at a slightly higher rate than desired
    - Potentially add a little uncertainty to the SEs to incorporate the uncertainty due to the model selection process

  - Model selection methods can involve a long conversation, but for the moment we'll just use AICs and ΔAICs in Burnham & Anderson style of picking a model

  - Intent is to pick the model that closest to the truth and/or? the model that balances fit and complexity optimally.

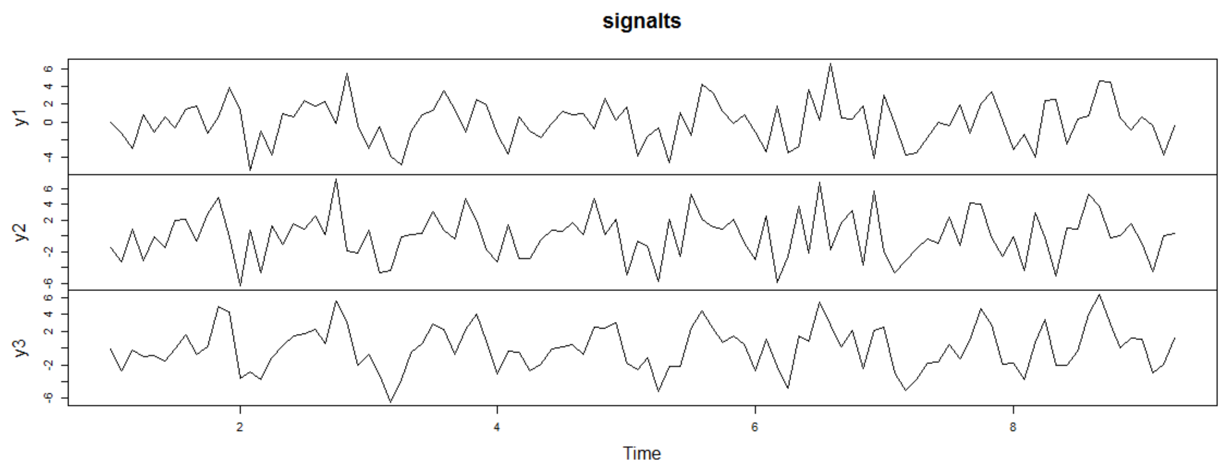
  - AIC=-2*loglikelihood+2p

    - where p=#parameters in the model that need to be estimated, including those related to the variance structure.
    - **Pick the model with smallest AIC**
      - Rule of thumb: If more than one model is within 2 AIC units of the top model, those models are "indistinguishable" in terms of support based on AIC.

  - ΔAIC =

- So let's try AIC for comparing the seasonal mean and cosine trend models assuming white noise, and two types of MA(1) error processes (where the seasonal effect is smooth) and see what happens...

```
> set.seed(9751)
> time<-seq(1:100)
> det<-2*cos(2*pi*time/12+.6*pi)
>  e<-rnorm(101,0,2)
>  x2<-e-0.5*zlag(e)
>  x3<-e+0.5*zlag(e)
>  y1<-det+e[1:100]
>  y2<-det+x2[2:101]
>  y3<-det+x3[2:101]
>  signalts<-ts(data.frame(y1,y2,y3),freq=12)
>  plot(signalts)
```



```
>  x_sm<-season(signalts[,1])
>  x_h<-harmonic(signalts[,1])
>  WN_sm1<-lm(signalts[,1]~x_sm)
>  WN_h1<-lm(signalts[,1]~x_h)
>
>  require(nlme)
>  AR1_h1<-gls(signalts[,1]~x_h,correlation=corARMA(.5,p=1,q=0),method="ML")
>  MA1_h1<-gls(signalts[,1]~x_h,correlation=corARMA(.5,p=0,q=1),method="ML")
>
>  AR1_sm1<-gls(signalts[,1]~x_sm,correlation=corARMA(.5,p=1,q=0),method="ML")
>  MA1_sm1<-gls(signalts[,1]~x_sm,correlation=corARMA(.5,p=0,q=1),method="ML")
>
>  AIC1<-AIC(WN_sm1,WN_h1,AR1_sm1,MA1_sm1,AR1_h1,MA1_h1)
>  AIC1[order(AIC1[,2]),]
        df      AIC
MA1_h1   5 419.6923
AR1_h1   5 424.4222
WN_h1    4 427.5995
MA1_sm1 14 429.3389
AR1_sm1 14 432.9179
WN_sm1  13 434.8982
>
>  summary(WN_h1)

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     0.04821    0.20045   0.241  0.81043
x_hcos(2*pi*t) -0.88461    0.28336  -3.122  0.00237 **
x_hsin(2*pi*t) -1.76388    0.28336  -6.225 1.23e-08 ***

Residual standard error: 2.002 on 97 degrees of freedom
Multiple R-squared:  0.3362,   Adjusted R-squared:  0.3226
F-statistic: 24.57 on 2 and 97 DF,  p-value: 2.33e-09

>  predict(WN_h1,se.fit=T)
```

```
$fit
          1           2           3           4           5           6 
-0.83639905 -1.59982313 -1.92165700 -1.71566553 -1.03704398 -0.06762844 
          7           8           9          10          11          12 
 0.93282698  1.69625106  2.01808493  1.81209346  1.13347191  0.16405637 

$se.fit
  [1] 0.3416661 0.3379937 0.3379937 0.3416661 0.3467467 0.3506703 0.3524366
  [8] 0.3527774 0.3527774 0.3524366 0.3506703 0.3467467

>  predict(WN_sm1,se.fit=T)
$fit
         1          2          3          4          5          6          7 
-0.2514467 -2.2367262 -2.0597970 -1.6236118 -1.1949841  0.4764464  0.3616639 
         8          9         10         11         12         13         14 
 2.6128156  1.6850783  0.5600182  2.0794161  0.1819492 -0.2514467 -2.2367262 

$se.fit
  [1] 0.6642193 0.6642193 0.6642193 0.6642193 0.7045109 0.7045109 0.7045109
  [8] 0.7045109 0.7045109 0.7045109 0.7045109 0.7045109 0.6642193 0.6642193

>  AR1_h1
Generalized least squares fit by maximum likelihood
  Model: signalts[, 1] ~ x_h 
  Data: NULL 
  Log-likelihood: -207.2111

Coefficients:
   (Intercept) x_hcos(2*pi*t) x_hsin(2*pi*t) 
    0.04454153    -0.88478087    -1.76707461 

Correlation Structure: AR(1)
 Formula: ~1 
 Parameter estimate(s):
       Phi 
-0.2241415 
Degrees of freedom: 100 total; 97 residual
Residual standard error: 1.971289 
>  AR1_sm1
Generalized least squares fit by maximum likelihood
  Model: signalts[, 1] ~ x_sm 
  Data: NULL 
  Log-likelihood: -202.4589

Coefficients:
  (Intercept)  x_smFebruary     x_smMarch     x_smApril       x_smMay 
   -0.2514467    -1.9852794    -1.8083502    -1.3721651    -0.9737256 
     x_smJune      x_smJuly    x_smAugust x_smSeptember   x_smOctober 
    0.7338406     0.6119398     2.8644882     1.9365051     0.8113443 
 x_smNovember  x_smDecember 
    2.3315191     0.4300560 

Correlation Structure: AR(1)
 Formula: ~1 
 Parameter estimate(s):
       Phi 
-0.1970193 
Degrees of freedom: 100 total; 88 residual
Residual standard error: 1.868709 

>  x_sm<-season(signalts[,2])
>  x_h<-harmonic(signalts[,2])
>  WN_sm2<-lm(signalts[,2]~x_sm)
>  WN_h2<-lm(signalts[,2]~x_h)
>  AR1_h2<-gls(signalts[,2]~x_h,correlation=corARMA(.5,p=1,q=0),method="ML")
>  MA1_h2<-gls(signalts[,2]~x_h,correlation=corARMA(.5,p=0,q=1),method="ML")
>  AR1_sm2<-gls(signalts[,2]~x_sm,correlation=corARMA(.5,p=1,q=0),method="ML")
>  MA1_sm2<-gls(signalts[,2]~x_sm,correlation=corARMA(.5,p=0,q=1),method="ML")
>
>  AIC1<-AIC(WN_sm2,WN_h2,AR1_sm2,MA1_sm2,AR1_h2,MA1_h2)
>  AIC1[order(AIC1[,2]),]
        df      AIC
MA1_h2   5 419.4904
MA1_sm2 14 428.9627
AR1_h2   5 445.5115
AR1_sm2 14 453.3552
WN_h2    4 466.9999
WN_sm2  13 472.6685

>  summary(MA1_h2)
Generalized least squares fit by maximum likelihood
  Model: signalts[, 2] ~ x_h
  Data: NULL
       AIC      BIC    logLik
  419.4904 432.5162 -204.7452

Correlation Structure: ARMA(0,1)
 Formula: ~1
 Parameter estimate(s):
    Theta1
-0.8144641

Coefficients:
                    Value  Std.Error    t-value p-value
(Intercept)     0.0172245 0.03677895   0.468325  0.6406
x_hcos(2*pi*t) -1.4630583 0.13481008 -10.852737  0.0000
x_hsin(2*pi*t) -1.7806081 0.13481008 -13.208271  0.0000

 Correlation:
               (Intr) x_hc(2**)
x_hcos(2*pi*t) -0.004
x_hsin(2*pi*t) -0.004 -0.004

Residual standard error: 2.404849
Degrees of freedom: 100 total; 97 residual
>  MA1_sm2
Generalized least squares fit by maximum likelihood
  Model: signalts[, 2] ~ x_sm
  Data: NULL
  Log-likelihood: -200.4814

Coefficients:
  (Intercept)  x_smFebruary     x_smMarch     x_smApril       x_smMay
   -2.3841421     0.4103785     0.2755205     0.7290691     1.9099679
     x_smJune      x_smJuly    x_smAugust x_smSeptember    x_smOctober
    1.9483441     5.0892651     3.7951424     3.6165295     5.7751175
 x_smNovember  x_smDecember
    2.7681523     2.5265549

Correlation Structure: ARMA(0,1)
 Formula: ~1
 Parameter estimate(s):
    Theta1
-0.7892828
Degrees of freedom: 100 total; 88 residual
Residual standard error: 2.277609
```

```
>  WN_sm3<-lm(signalts[,3]~x_sm)
>  WN_h3<-lm(signalts[,3]~x_h)
>  AR1_h3<-gls(signalts[,3]~x_h,correlation=corARMA(.5,p=1,q=0),method="ML")
>  MA1_h3<-gls(signalts[,3]~x_h,correlation=corARMA(.5,p=0,q=1),method="ML")
>  AR1_sm3<-gls(signalts[,3]~x_sm,correlation=corARMA(.5,p=1,q=0),method="ML")
>  MA1_sm3<-gls(signalts[,3]~x_sm,correlation=corARMA(.5,p=0,q=1),method="ML")
>  AIC1<-AIC(WN_sm3,WN_h3,AR1_sm3,MA1_sm3,AR1_h3,MA1_h3)
>  AIC1[order(AIC1[,2]),]
        df      AIC
MA1_h3   5 430.0226
WN_h3    4 432.5966
AR1_h3   5 432.9350
MA1_sm3 14 438.3750
AR1_sm3 14 442.5288
WN_sm3  13 442.8780

>  summary(MA1_h3)
Generalized least squares fit by maximum likelihood
  Model: signalts[, 3] ~ x_h
  Data: NULL
       AIC      BIC    logLik
  430.0226 443.0484 -210.0113

Correlation Structure: ARMA(0,1)
 Formula: ~1
 Parameter estimate(s):
   Theta1
0.3292192

Coefficients:
                    Value Std.Error   t-value p-value
(Intercept)     0.1046024 0.2662309  0.392901  0.6953
x_hcos(2*pi*t) -0.8956900 0.3668421 -2.441623  0.0164
x_hsin(2*pi*t) -2.1469545 0.3668421 -5.852530  0.0000
```

Back to CC 3.5:

- Note that when you are working with a time series object, say it is called y, and you have TSA loaded, `time(y)` will extract the time variable from the ts object to use in regression models

- Everything else in 3.5 should be review
- All results from OLS assume white noise errors
- Once we go beyond OLS, R^2 and adjusted R^2 there is no agreement for how they should be defined
    - Percentage of deviance explained has been suggested as an extension of percentage of variance explained

CC 3.6: Residual Analysis

- The unobserved stochastic component can be predicted by the residual: $\hat{X}_t = Y_t - \hat{\mu}_t$
- If the trend is approximately correct, then $\hat{X}_t = Y_t - \hat{\mu}_t$ should have dynamics that are close to the stochastic process

- So we can try to deal with the trend first, then diagnose the stochastic process to consider in addition to it.
- Residuals or standardardized residuals can be used
  - standardized residuals help with diagnosing non-normal (t?) errors and outliers
  - Unstandardized residuals are fine for detecting asymmetric errors or problems in the trend (seasonality?)

  - If a seasonal pattern is detected in the residuals, then a seasonal component should be included in the trend
    - Adding season labels to the residual plot can aid in detecting this.

- New aspects of residual plots:
  - **Residuals vs time:**
    - Look for trends over time (seasonal)
      - Suggest alternative trend model including seasonal trend

    - Outliers
      - Look for a reason (intervention/ outlier models)

    - “Shape” of residuals
      - potential indication of lack of normality

      - Changing variability over time?

  - **Residuals vs fitted values:**
    - Trends in mean of the errors:
      - suggests missing trend component if fitted values picking up on trend to some degree

    - Variability changing over fitted values
      - suggests variance is a function of the mean (consider variance stabilizing transformation or non-constant error model)

  - Histogram or density plot of standardized residuals:
    - Use `rstudent(model)` but be careful with gls-type models and this won’t work for GAM models…
    - Non-normal histogram of residuals (especially skew):
      - If no evidence of trend missed in previous diagnostics, then suggestion of non-normal errors

    - Outliers
      - If no evidence of trend missed in previous diagnostics, then investigate source, consider removal, …

  - QQ-plot vs Normal Distribution:
    - Systematic deviation from 1-1 relationship
      - Suggests alternative error distribution (if no trend problems detected)

    - Outliers in plot, deal with as above

- Visual diagnostics are subjective (but the best way to start)

- Shapiro-Wilk is one test that we can use to test the normality of the residuals:
  - Null hypothesis: Residuals are normally distributed
  - Alternative: Residuals are not normally distributed
  - `shapiro.test()`
  - Lack of normality is most a problem when the sample size is small but this test has low power when the sample size is small

- If we "pass" on the previous visual diagnostics and the normality test, we really only know that the trend is reasonably well specified.
  - OLS assumes white noise which includes an independence assumption
  - RUNS tests looks generally for patterns as evidence against independence
    - Counts # of observations in a row above or below median

    - Positive correlation: a small number of runs of each type are generated
    - Negative correlation: a large number of runs of short duration are generated as the series crosses its median repeatedly
    - Looks for deviations from the expected number of runs in either direction
    - Null hypothesis: independent, identically distributed measurements
    - Alternative hypothesis: dependent measurements (either positive or negative dependency) or non-identically distributed measures
      - Could reject if the distribution changes shape over time (maybe if variance changes?)
    - `require(TSA); runs()` assumes that the median is 0 which is only true if the residuals are symmetric
      - Cryer and Chan recommend `runs(rstudent(model))`
      - Can use `runs(rstudent(model),k=median(rstudent(model)))`

  - Suppose we reject the null hypothesis with a runs test…
    - We know that assuming independence is dangerous and can look as to whether we have positive or negative dependence based on observed # runs vs expected
    - But that doesn't diagnose the type of model we should consider…

  - We could try some different stochastic models and compare AICs (not that unreasonable even after the following more conventional method)

  - Estimate the autocorrelation to diagnose which *ar* or *ma* model that might be appropriate for the stochastic error process

The Sample Autocorrelation Function (SACF) (p35 CM, p 46 CC)
- Or just ACF since we use it so much
- If a series is stationary, then μ_t is constant and we can use $\bar{y} =$

- If a series is not stationary in the mean, then we need to estimate the nonstationary trend to get to our predicted stationary stochastic residual process and explore its structure

  - Pearson correlation coefficient:

- Sample autocovariance function:

$$\hat{\gamma}_k =$$

  - Sample autocorrelation function (SACF):

$$\hat{\rho}_k = r_k = \frac{\hat{\gamma}_k}{\hat{\gamma}_0} =$$

  - Large sample distribution of ACF: