I. **Review of Exponential Smoothing**

a. The level of a time series may vary unpredictably as a random walk.

\[ Y_t = Y_{t-1} + e_t, \quad e_t \sim N(0, \sigma^2) \]

R code:

```r
y = ts(sapply(rep(500,5),
        function(n) cumsum(rnorm(n, mean=0, sd=1.5))))
plot(y, plot.type="single", col=1:5, main="5 Random Walks")
```

A decent estimate of the level at the next time period can be constructed as a weighted average with weights decreasing as we go back in time. Weight \( \alpha \) is placed on the last observed time point, and \( \alpha(1-\alpha), \alpha(1-\alpha)^2, \text{etc.} \) are used backwards in time. Maximum likelihood can be used to estimate \( \alpha \).

b. If the mean of the series is predictably increasing or decreasing, a trend term is added. The \( \beta \) parameter controls the weights on the one-time-interval slopes that are used to estimate the current slope. Use maximum likelihood estimation of \( \alpha \) and \( \beta \).

```r
y = ts(sapply(rep(500,5),
        function(n) cumsum(rnorm(n, mean=0.1, sd=1.5))))
plot(y, plot.type="single", col=1:5, main="5 Random Walks with a Trend")
```
c. A practical compromise between no trend (single exponential smoothing) and trend (double exponential smoothing) is a trend with damping. Parameter $\phi$ controls how quickly the current trend decays towards no trend.

```r
N = 500
phi = 0.005
y = ts(sapply(rep(N,5),
  function(n) {
    mn = 100*phi*(1-phi)^(0:(n-1))
    cumsum(rnorm(n, sd=1.5, mean=mn))
  })),
plot(y, plot.type="single", col=1:5,
  main="5 Random Walks with a Damped Trend")
```

d. If the time series is periodic we can use triple exponential smoothing, with parameter $\gamma$ controlling the weighting back in time for estimating the seasonal pattern (figure). The $\gamma$ parameter can also be damped.
II. Prediction accuracy:
   a. MSE (mean squared error) gives large penalties for outliers
   b. MAE (mean absolute error) is less sensitive to outliers
   c. MPE (mean percent error) is less sensitive to large errors paired with large values
   d. Calculated separately for training and test sets
   e. Example:

```r
# Australian beer with 8 year holdout
library(fpp)
plot(ausbeer)

# abTrain = window(ausbeer, 1956, 2000)
abTest = window(ausbeer, 2000.25, 2008.5)
nTest = length(abTest)
abm = ets(abTrain)  # default is model="ZZZ": all automatic
abm$method  # ETS(M,A,M)

# abm2 = ets(abTrain, model="AAA")

oldPar = par(no.readonly=TRUE)
par(mfrow=c(2,1), mar=0.1+c(5,3,2,2))
plot(abTrain, main="red=MAM, green=AAA", ylab="Aust. beer production")
lines(fitted(abm), col="red")
lines(fitted(abm2), col="green")

abmf = forecast(abm, h=nTest, level=95)
abm2f = forecast(abm2, h=nTest, level=95)

# function to plot y using tsp() from ts.x
xplot = function(y, ts.x, ...) {
  tsp=tsp(ts.x)
  lines(seq(tsp[1],tsp[2],by=1/tsp[3]), y, ...)
}

ylim = range(abmf$lower, abm2f$lower, abmf$upper, abm2f$upper)
plot(abTest, main="Holdback", ylim=ylim, ylab="Aust. beer production")
lines(abmf$mean, col=2)
xplot(abmf$lower[,1], abmf$mean, col=2, lty=2)
xplot(abmf$upper[,1], abmf$mean, col=2, lty=2)
lines(abm2f$mean, col=3)
xplot(abm2f$lower[,1], abm2f$mean, col=3, lty=3)
xplot(abm2f$upper[,1], abm2f$mean, col=3, lty=3)
par(oldPar)
```
# Compare accuracy

```r
accuracy(abmf, abTest)
#                  RMSE      MAE         MPE      MASE
# Training set 15.83936 12.07689 -0.06494774 0.7567194
# Test set     19.57117 13.92569 -1.72426993 0.8725623
accuracy(abm2f, abTest)
#                  RMSE      MAE        MPE      MASE
# Training set 16.21478 12.49661 -0.1016742 0.7830185
# Test set     19.33952 13.65736 -1.318029  0.8557491
abmf$model$aicc # [1] 1880.649
abm2f$model$aicc # [1] 1919.252
```
f. Australian beer predication at an inauspicious time

red=MAM, green=AAA

Holdback

accuracy(abmI, abTestI)[,c(2:6)]
#                   RMSE       MAE        MPE     MAPE      MASE
# Training set  9.269449  7.794063  0.4713767 2.479188 0.5897719
# Test set     36.914042 28.401269 -2.8823515 5.729134 2.1491065
accuracy(abm2fI, abTestI)[,c(2:6)]
#                   RMSE      MAE        MPE     MAPE     MASE
# Training set  9.530419  7.94209  0.5043071 2.49404 0.600973
# Test set     40.161940 32.21580 -4.3219102 6.710963 2.437750
g. Eurozone electrical equipment manufacturing

```
plot(elecequip)
```

```
tsp(elecequip) # 1996.000 2011.833   12.000
me = ets(elecequip)
summary(me)
# ETS(M,Md,M)
#   Smoothing parameters:   alpha = 0.527    beta  = 0.1709
#     gamma = 1e-04   phi   = 0.971
#   Initial states:    l = 82.6198     b = 0.9855
#     s=1.1199 1.041 1.0314 1.0767 0.823 0.987
#          1.0808 0.9468 0.9323 1.0824 0.9356 0.9432
# sigma:  0.0323
plot(forecast(me, h=60))
```

```
Forecasts from ETS(M,Md,M)
```

```
```
```R
test <- ets(elecequip, "AAA")
summary(test)
plot(forecast(test, h=60))

# ETS(A,Ad,A)
# Smoothing parameters: alpha = 0.3857 beta = 0.2281
# gamma = 0.0001 phi = 0.8

accuracy(test)
accuracy(test)

III. Miscellaneous info from the multiple regression chapter of Hyndman and Athanasopoulos

a. Piecewise linear with breakpoint at t=\tau (in regression):
   \[ X_{2,t} = t, \quad X_{2,t} = 0 + (t-\tau)I(t\geq\tau) \]
   where \( I() \) is the 0=FALSE, 1=TRUE indicator function

b. Lagged X can be useful: \( X_{2,t} = \text{NA}, X_{2,t}=X_{1,t-1} \)

c. An intervention at time \tau can be represented as X variable “spikes”, \( X_{1,t} = I(t=\tau) \), or “steps”, \( X_{1,t} = I(t\geq\tau) \).
```