Forecasting ARIMA(1,1,1) Series

ARIMA(1,1,1)

1. We generate the data assuming the true process is known. Then we can compare the estimation result to the truth to ensure the coding is right.

2. In general, an ARIMA(1,1,1) process is

$$\Delta y_t = d + \eta_t \tag{1}$$
$$\eta_t = \phi_1 \eta_{t-1} + e_t + \theta_1 e_{t-1} \tag{2}$$

   In words, the first difference Δy_t is a zero-mean ARMA(1,1) process η_t plus the drift term d.

3. By substituting $\eta_t = y_t - y_{t-1} - d$, the same ARIMA(1,1,1) process can be written as

$$(y_t - y_{t-1} - d) = \phi_1 (y_{t-1} - y_{t-2} - d) + e_t + \theta_1 e_{t-1} \tag{3}$$

   where d is the drift term; ϕ_1 is the AR coefficient; θ_1 is the MA coefficient.

4. Here we let $d = 0.2, \phi_1 = 0.7, \theta_1 = -0.5$. Notice that the nonzero drift term causes the series to be trending.

Generating ARIMA(1,1,1)

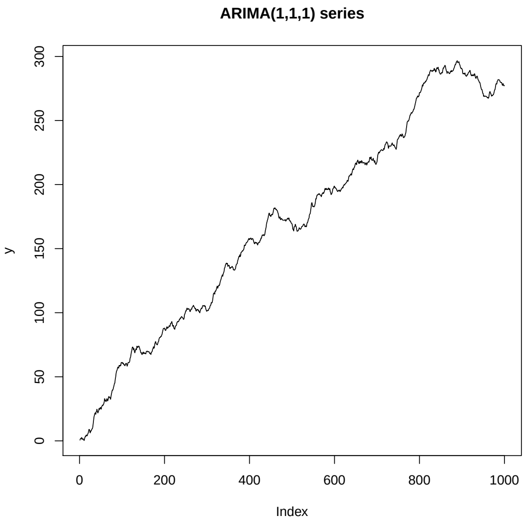
We use R loop to generate η_t and then y_t

```
set.seed(12345)
T = 1000
tr = 1:T
e = rnorm(T)
dy = rep(0, T)
y = rep(0, T)
dy[1] = e[1]
y[1] = e[1]
for (t in 2:T) {
dy[t] = 0.7*dy[t-1]+e[t]-0.5*e[t-1]
y[t] = 0.2 + y[t-1]+dy[t]
}
```

In the codes, the ARMA(1,1) η_t is denoted as $dy[t]$

Plotting ARIMA(1,1,1)

The series has an upward trend due to the positive drift term $d = 0.2$. The trend, along with the smoothness, signifies nonstationarity.



Deterministic and Stochastic Trends

We can show that the ARIMA(1,1,1) process is trending

$$y_t = y_0 + dt + (\eta_1 + \eta_2 + \ldots + \eta_t)$$

There is a global deterministic trend dt if $d \neq 0$. Even if $d = 0$, there can be local stochastic trend $(\eta_1 + \eta_2 + \ldots + \eta_t)$. In graph, the deterministic trend can be dominating. We can estimate the drift term, which is the mean value of Δy_t, without running regression

```
mean(d.y, na.rm=T)
[1] 0.2769492
```

Estimating ARIMA(1,1,1)

1. Estimating ARIMA(1,1,1) for y_t is the same as estimating ARIMA(1,0,1) for Δy_t
2. However, DO NOT use arima(y, order = c(1,1,1)) because this assumes zero drift term!!
3. Instead, use arima(d.y, order = c(1,0,1)). But be careful, the reported intercept actually is the drift term

```
arima(x = d.y, order = c(1, 0, 1))

Coefficients:
         ar1      ma1  intercept
      0.7246  -0.5197     0.2767
s.e.  0.0698   0.0871     0.0550
log likelihood = -1416.09,  aic = 2840.17
```

The estimated $\hat{\phi}_1 = 0.7246, \hat{\theta}_1 = -0.5197, \hat{d} = 0.2767$ are all close to the true values, and are significant.

Estimating Error Terms

The error term e_t in (2) is unobservable. According to (3), we can show

$$\begin{aligned} e_t &= (y_t - y_{t-1} - d) - \phi_1(y_{t-1} - y_{t-2} - d) - \theta_1 e_{t-1} \\ &= y_t - (1-\phi_1)d - (1+\phi_1)y_{t-1} + \phi_1 y_{t-2} - \theta_1 e_{t-1} \end{aligned}$$

So we use the codes below to estimate $e_1, e_2, \ldots e_t$ in a recursive way

```
ehat = rep(0, T)
ehat[1] = y[1]
ehat[2] = y[2] - (1-phi1)*dhat-(1+phi1)*y[1]-theta1*ehat[1]
for (t in 3:T) ehat[t] = y[t] - (1-phi1)*dhat-(1+phi1)*y[t-1]+phi1*
```

Built-in Function for Estimating Error Terms

Except the early observations, our estimated error terms are very close to the ones reported by R built-in function resid

```
cbind(resid(arima(d.y, order = c(1,0,1)))[1:10], ehat[1:10])
            [,1]       [,2]
 [1,]         NA  0.5855288
 [2,]  0.7187304  0.8303763
 [3,] -0.1458968 -0.0739732
 [4,] -0.4963657 -0.4612424
 [5,]  0.5616954  0.5796414
 [6,] -1.8649630 -1.8558010
 [7,]  0.5906055  0.5951970
 [8,] -0.3141356 -0.3117367
 [9,] -0.3219821 -0.3207393
[10,] -0.9561814 -0.9555372
```

Forecasting ARIMA(1,1,1)

1. Assuming our sample ends at T.
2. Rewrite (3) as
$$y_t = (1-\phi_1)d + (1+\phi_1)y_{t-1} - \phi_1 y_{t-2} + e_t + \theta_1 e_{t-1} \tag{4}$$
3. The one-step and two-step (out-of-sample) forecasts are
$$E(y_{T+1}|\Omega_T) = (1-\phi_1)d + (1+\phi_1)y_T - \phi_1 y_{T-1} + \theta_1 e_T \tag{5}$$
$$E(y_{T+2}|\Omega_T) = (1-\phi_1)d + (1+\phi_1)E(y_{T+1}|\Omega_T) - \phi_1 y_T \tag{6}$$

   For $k-$th horizon $(k > 2)$ we have
$$E(y_{T+k}|\Omega_T) = (1-\phi_1)d + (1+\phi_1)E(y_{T+k-1}|\Omega_T) - \phi_1 E(y_{T+k-2}|\Omega_T) \tag{7}$$

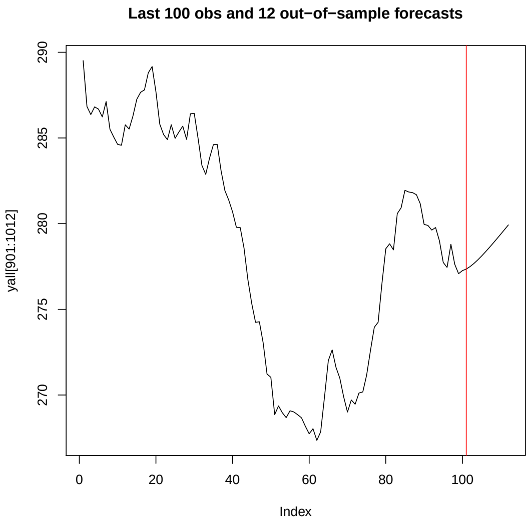
R Codes

Again, we use R loop to generate 12 out-of-sample forecasts, and display the first 5 forecasts

```
f = rep(0, 12)
f[1] = (1-phi1)*dhat+(1+phi1)*y[T]-phi1*y[T-1]+theta1*ehat[T]
f[2] = (1-phi1)*dhat+(1+phi1)*f[1]-phi1*y[T]
for (t in 3:12) f[t] = (1-phi1)*dhat+(1+phi1)*f[t-1]-phi1*f[t-2]
f[1:5]
[1] 277.3508 277.4945 277.6748 277.8817 278.1078
```

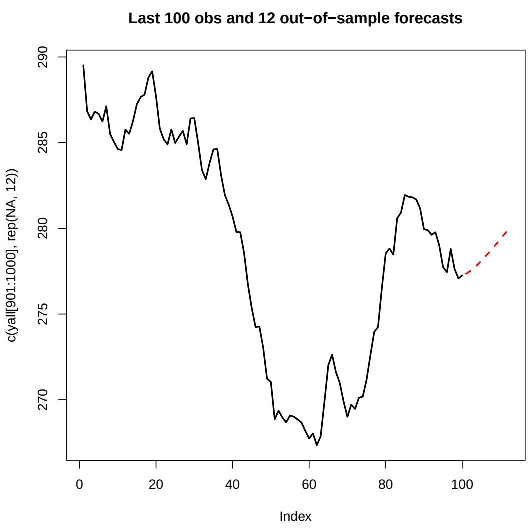
Plotting Forecasting Values I

```
yall = c(y,f)
plot(yall[901:1012], type="l",main="Last 100 obs and 12 out-of-samp
abline(v = c(101), col = "red", lty=1)
```



Plotting Forecasting Values II

```
plot(c(yall[901:1000],rep(NA, 12)), type="l", lwd=2, lty=1, main="L
lines(c(rep(NA, 100), f), lty=2, col = "red", lwd=2)
```



R Forecast Package

There is a Forecast package that can provide the out-of-sample forecasts

```
library(forecast)
forecast(Arima(y,order=c(1,1,1),include.drift=T),h=12)
     Point Forecast     Lo 80    Hi 80    Lo 95    Hi 95
1001       277.3508 276.0693 278.6324 275.3908 279.3108
1002       277.4945 275.4878 279.5012 274.4255 280.5635
1003       277.6748 275.0224 280.3272 273.6182 281.7313
1004       277.8816 274.6349 281.1283 272.9162 282.8471
1005       278.1077 274.3089 281.9066 272.2979 283.9175
1006       278.3478 274.0336 282.6619 271.7498 284.9457
1007       278.5979 273.8010 283.3948 271.2617 285.9341
1008       278.8554 273.6047 284.1060 270.8252 286.8855
1009       279.1181 273.4395 284.7967 270.4335 287.8028
1010       279.3847 273.3011 285.4684 270.0806 288.6889
```

Remarks

1. We obtain the same forecasts!
2. Arima is the function in the forecast package, which is different from arima in the stats package.
3. Notice that `include.drift=T` allows for a nonzero drift term.