Time Series

John Tipton

August 24, 2015

Switch forecast and smoother line colors, not sure what is going on...

A time series is a series of observations y_t that occur over a period of time $t = 1, \ldots, T$. Depending on the problem being studied, these measurements could be made at all possible times, at regular intervals in time, or randomly through time. Thus, the analysis that is used must take into account the sampling methods used to obtain the data. The simplest time series is an uncorrelated random walk. In this time series, each observation y_t is just a random move from the previous location y_{t-1}. The random walk model is

$$y_t = y_{t-1} + \epsilon_t \tag{1}$$

where ϵ_t $\mathrm{N}(0, \sigma^2)$ is independent, uncorrelated error.

```r
N <- 1                                    ## pick the number of time series
t <- 100                                  ## pick a time series length
mu <- rep(0, t)                           ## pick a mean structure
s_low <- 0.25                             ## pick standard deviation
s_high <- 1                               ## pick standard deviation

y_s_low <- simTimeSeries(t, N, mu, s_low, phi=0)
y_s_high <- simTimeSeries(t, N, mu, s_high, phi=0)
plot_s_low <- ggplot(data=data.frame(y=y_s_low, t=1:t), aes(y=y, x=t)) +
  geom_line(alpha=1, colour="darkgrey") + coord_cartesian(ylim=c(-2.5, 2.5)) +
  ggtitle(substitute(paste("Plot of a random walk time series with ",
                           sigma, "=", sd), list(sd=s_low))) +
  theme(plot.title = element_text(size=18))
plot_s_high <- ggplot(data=data.frame(y=y_s_high, t=1:t), aes(y=y, x=t)) +
  geom_line(alpha=1, colour="darkgrey") + coord_cartesian(ylim=c(-2.5, 2.5)) +
  ggtitle(substitute(paste("Plot of a random walk time series with ",
                           sigma, "=", sd), list(sd=s_high))) +
  theme(plot.title = element_text(size=18))
multiplot(plot_s_low, plot_s_high, cols=1)
```

We start with the canonical difference equation for the time series autoregressive model of order 1 (AR(1))

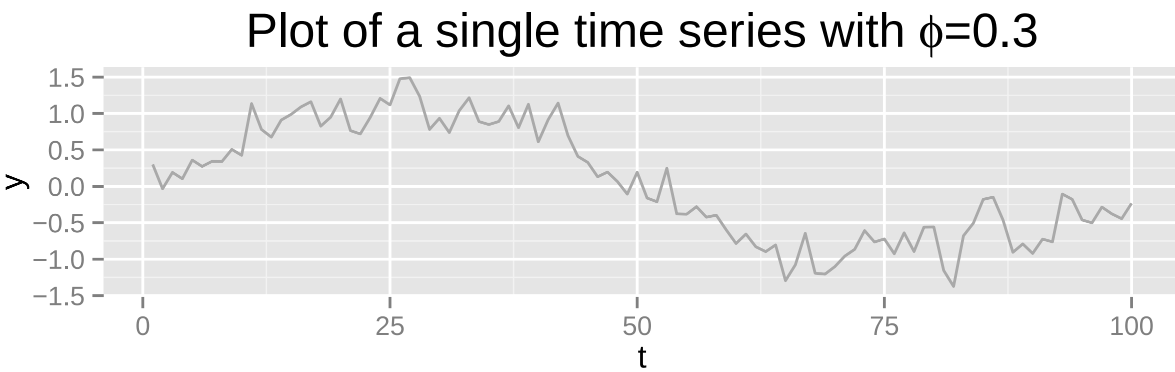
$$y_t = \mu_t + \phi(y_{t-1} - \mu_{t-1}) + \epsilon_t \tag{2}$$
$$\tag{3}$$

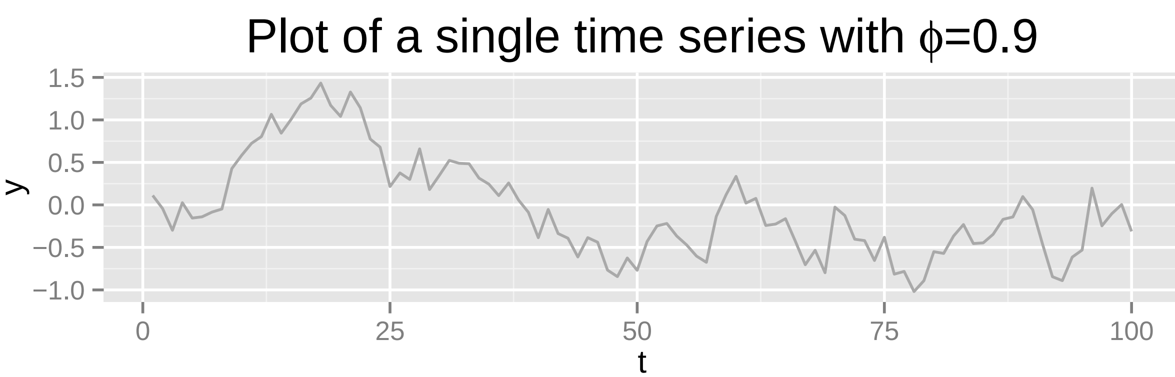
where the time series observations for times $t = 1, \ldots, T$ are given by the vector $\mathbf{y} = (y_1, \ldots, y_T)$ where y_t is the observation of the time serise at time t. μ_t is the temporal mean with of the time series at time t and is assumed known. Often the mean is a trend or seasonal component that in practice is often estimated in a regression framework. The autoregressive parameter ϕ controls the strength of autocorrelation in the time series with $-1 < \phi < 1$ and the random error $\epsilon_t \sim N(0, \sigma^2)$ is independent for different times (i.e. the covariance $\mathrm{Cov}(\epsilon_t, \epsilon_{t+k}) = 0$ for $k \neq 0$).

Lets simulate some data here

```
N <- 1                                    ## pick the number of time series
t <- 100                                  ## pick a time series length
mu <- sin(2 * pi * (1:t)/t)               ## pick a mean structure
# mu <- rep(0, t)                           ## pick a mean structure
# mu <- rep(6, t)                           ## pick a mean structure
s <- 0.25                                 ## pick standard deviation
phi_low <- 0.30                           ## pick autocorrelation parameter
phi_high <- 0.90                          ## pick autocorrelation parameter

y_phi_low <- simTimeSeries(t, N, mu, s, phi=phi_low)
```

```
y_phi_high <- simTimeSeries(t, N, mu, s, phi=phi_high)
plot_phi_low <- ggplot(data=data.frame(y=y_phi_low, t=1:t), aes(y=y, x=t)) +
  geom_line(alpha=1, colour="darkgrey") +
  ggtitle(substitute(paste("Plot of a single time series with ", phi, "=", p), list(p=phi_low))) +
  theme(plot.title = element_text(size=18))
plot_phi_high <- ggplot(data=data.frame(y=y_phi_high, t=1:t), aes(y=y, x=t)) +
  geom_line(alpha=1, colour="darkgrey") +
  ggtitle(substitute(paste("Plot of a single time series with ", phi, "=", p), list(p=phi_high))) +
  theme(plot.title = element_text(size=18))
multiplot(plot_phi_low, plot_phi_high, cols=1)
```





The expected value $\mathrm{E}(y_t)$ of the time series at time t is

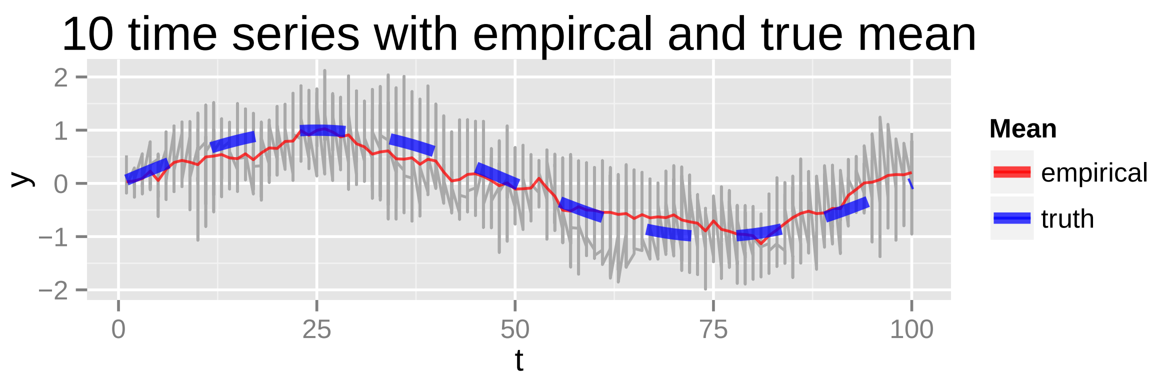
$$
\begin{aligned}
\mathrm{E}(y_t) &= \mathrm{E}(\mu_t + \phi(y_{t-1} - \mu_{t-1}) + \epsilon_t) \\
&= \mathrm{E}(\mu_t) + \mathrm{E}(\phi y_{t-1}) - \mathrm{E}(\phi\mu_{t-1}) + \mathrm{E}(\epsilon_t) \\
&= \mu_t - \phi\mu_{t-1} + \phi\mathrm{E}(y_{t-1}) + 0 \\
&= \mu_t - \phi\mu_{t-1} + \phi(\mathrm{E}(\mu_{t-1}) + \mathrm{E}(\phi(y_{t-2} - \mu_{t-2})) + \mathrm{E}(\epsilon_{t-1})) \\
&= \mu_t - \phi\mu_{t-1} + \phi\mu_{t-1} - \phi^2\mu_{t-2} + \cdots \\
&= \mu_t
\end{aligned}
$$

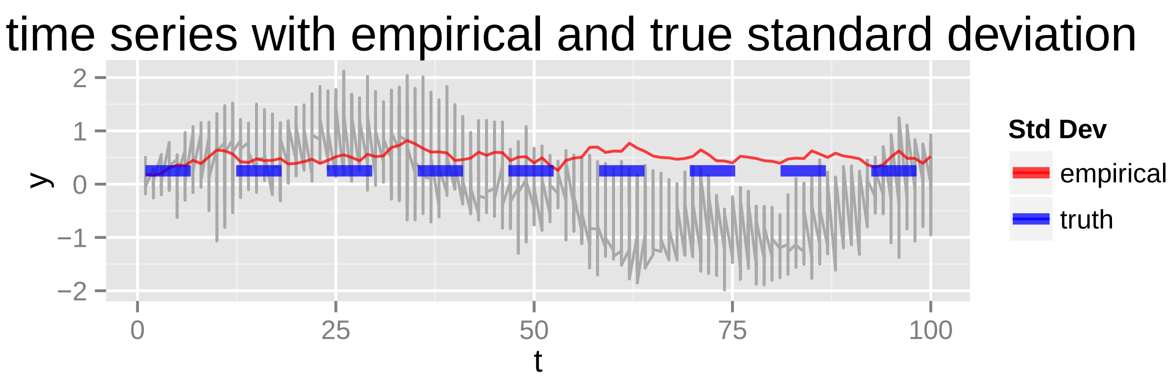
where the $\cdots$ form an infinite recursive sum. The autoregressive model assumes a constant variance through time (homoskedasticity). This means that for any times t and τ $\mathrm{Var}(y_t) = \mathrm{Var}(y_\tau)$. Therefore, the variance

$$
\begin{aligned}
\mathrm{Var}(y_t) &= \mathrm{Var}(\mu_t + \phi(y_{t-1} - \mu_{t-1}) + \epsilon_t) \\
&= \mathrm{Var}(\mu_t) + \mathrm{Var}(\phi(y_{t-1} - \mu_{t-1})) + \mathrm{Var}(\epsilon_t) + 2\mathrm{Cov}(\phi(y_{t-1} - \mu_{t-1}), \epsilon_t) \\
&= 0 + \phi^2 \mathrm{Var}(y_{t-1}) + \phi^2 \mathrm{Var}(\mu_{t-1}) - \phi^2 \mathrm{Cov}(y_{t-1}, \mu_{t-1}) + \sigma^2 + 0 \\
&= \phi^2 \mathrm{Var}(y_t) + 0 - 0 + \sigma^2 \\
&= \frac{\sigma^2}{1 - \phi^2}
\end{aligned}
$$

```
N <- 10                               ## replicates N
phi <- 0.9                            ## autocorrelation parameter
y <- simTimeSeries(t, N, mu, s, phi)  ## simulate time series

mean_y <- apply(y, 1, mean)           ##  calculate the empirical mean
var_y <-apply(y, 1, var)              ##  calculate the empirical variance
sd_y <- sqrt(N-1) / sqrt(N) * apply(y, 1, sd)              ##  calculate the empirical standard deviati

time_data <- data.frame(y=y, t=1:t)
melt_time <- melt(time_data, id="t")
summary_data <- data.frame(mean_y=mean_y, var_y=var_y, sd_y=sd_y,
#                            mu=mu-phi^(0:(t-1))*mu[1], s=s, t=1:t)
mu=mu, s=s, t=1:t)


## plot time series with mean and variance
plot_mean <- ggplot(data = melt_time, aes(y=value, x=t)) +
  geom_line(alpha=1, colour="darkgrey") +
  geom_line(data=summary_data, aes(y=mean_y, x=t, colour="empirical"),
            alpha=0.75) +
  geom_line(data=summary_data, aes(y=mu, x=t, colour="truth"), alpha=0.75,
            lty=2, lwd=2) +
  scale_colour_manual("Mean", labels=c("empirical", "truth"),
                      values=c("empirical"="red","truth"="blue")) +
  scale_y_continuous("y") + scale_x_continuous("t") +
  ggtitle(paste(min(N, 10),
                "time series with empircal and true mean")) +
  theme(plot.title = element_text(size=18))

plot_sd <- ggplot(data = melt_time, aes(y=value, x=t)) +
  geom_line(alpha=1, colour="darkgrey") +
  geom_line(data=summary_data, aes(y=sd_y, x=t, colour="empirical"),
            alpha=0.75) +
  geom_line(data=summary_data, aes(y=s, x=t, colour="truth"), alpha=0.75,
            lty=2, lwd=2) +
  scale_colour_manual("Std Dev", labels=c("empirical", "truth"),
                      values=c("empirical"="red","truth"="blue")) +
  scale_y_continuous("y") + scale_x_continuous("t") +
  ggtitle(paste(min(N, 10),
                "time series with empirical and true standard deviation")) +
  theme(plot.title = element_text(size=18))

## Plot using multiplot
```

```r
multiplot(plot_mean, plot_sd, cols=1)
```



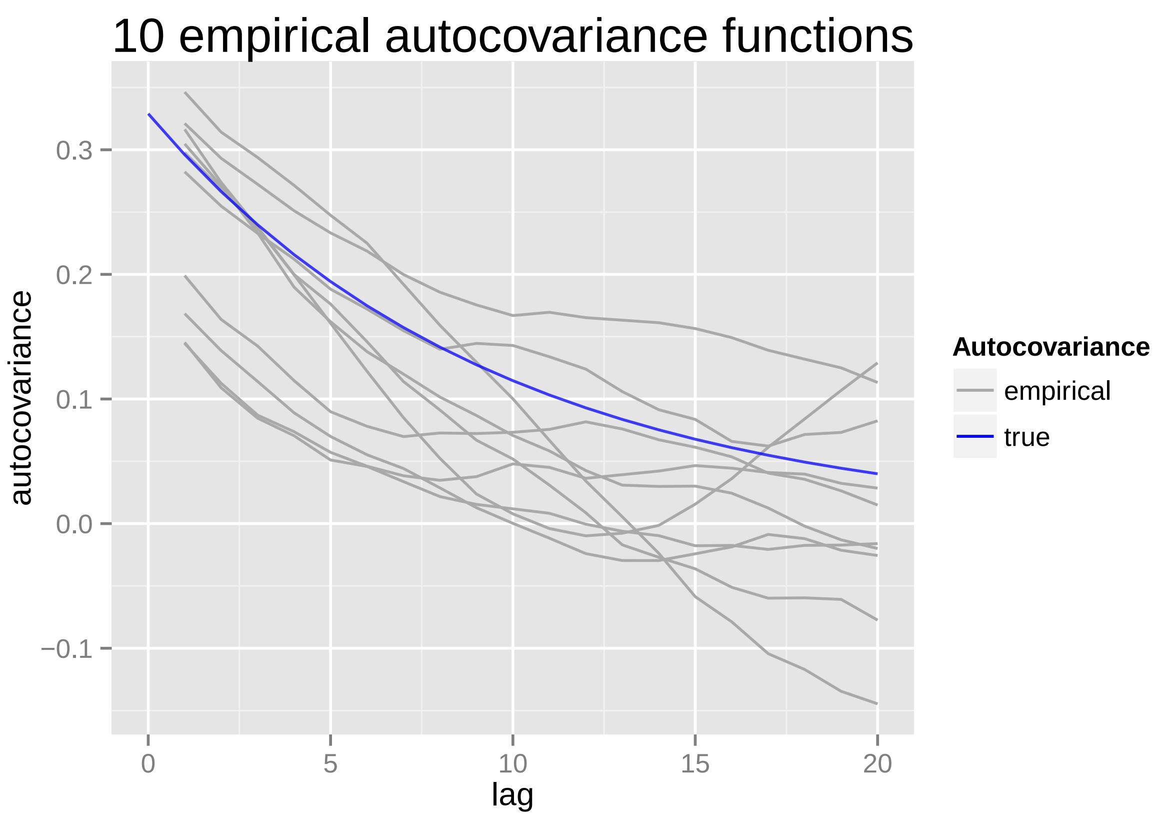


The covariance between centered observations $\mathrm{Cov}(y_t - \mu_t, y_{t+k} - \mu_{t+k})$ at times k lags apart (assuming without loss of generality that $k > 0$) is

$$\begin{aligned}
\mathrm{Cov}(y_t - \mu_t, y_{t+k} - \mu_{t+k}) &= \mathrm{E}((y_t - \mu_t)(y_{t+k} - \mu_{t+k})) - \mathrm{E}(y_t - \mu_t)\mathrm{E}(y_{t+k} - \mu_{t+k}) \\
&= \mathrm{E}(y_t y_{t+k}) - \mathrm{E}(y_t \mu_{t+k}) - \mathrm{E}(y_{t+k}\mu_t) + \mathrm{E}(\mu_t \mu_{t+k}) - 0 \\
&= \mathrm{E}(y_t(\mu_{t+k} + \phi(y_{t+k-1} - \mu_{t+k-1}) + \epsilon_{t+k})) - \mu_{t+k}\mathrm{E}(y_t) - \mu_t \mathrm{E}(y_{t+k}) + \mu_t \mu_{t+k} \\
&= \mathrm{E}(y_t \mu_{t+k}) + \mathrm{E}(\phi y_t y_{t+k-1}) - \mathrm{E}(\phi y_t \mu_{t+k-1}) + \mathrm{E}(y_t \epsilon_{t+k}) - \mu_t \mu_{t+k} \\
&= \mu_{t+k}\mathrm{E}(y_t) + \phi \mathrm{E}(y_t y_{t+k-1}) - \phi \mu_{t+k-1}\mathrm{E}(y_t) + \mathrm{E}(y_t)\mathrm{E}(\epsilon_{t+k}) \\
&= \mu_t \mu_{t+k} + \phi \mathrm{E}(y_t y_{t+k-1}) - \phi \mu_{t+k-1}\mu_t + 0 - \mu_t \mu_{t+k} \\
&= \phi \mathrm{E}(y_t(\mu_{t+k-1} + \phi(y_{t+k-2} - \mu_{t+k-2}) + \epsilon_{t+k-1})) - \phi \mu_{t+k-1}\mu_t \\
&= \vdots \\
&= \phi^k \mathrm{E}(y_t^2) \\
&= \phi^k \frac{\sigma^2}{1 - \phi^2}.
\end{aligned}$$

```r
covariances <- matrix(0, min(t, 20), N)
for(i in 1:N){
    for(k in 1:min(t, 20)-1){
      covariances[k+1, i] <- cov(y[1:(t-k), i] - mu[1:(t-k)], y[1:(t-k) + k, i] - mu[1:(t-k) + k])
  }
```

```
}

cov_data <- data.frame(y=covariances, t=1:20)
melt_cov <- melt(cov_data, id="t")
ggplot(data = melt_cov, aes(y=value, x=t)) +
  geom_line(data=melt_cov, aes(y=value, x=t, group=variable, colour="empirical"), alpha=1) +
  geom_line(data=data.frame(y=s^2/(1-phi^2) * phi^(0:min(t,20)), x=0:min(t, 20)), aes(y=y, x=x, colour=
            alpha=0.75) +
  scale_colour_manual("Autocovariance", labels=c("empirical", "true"),
                      values=c("empirical"="darkgrey","truth"="blue")) +
  scale_y_continuous("autocovariance") + scale_x_continuous("lag") +
  ggtitle(paste(min(N, 10),
              "empirical autocovariance functions")) +
  theme(plot.title = element_text(size=18))
```



Typically of interest in a time series model are the forecast distribution (used for prediction) and the smoothing distribution (used for estimation of parameters and learning about the processes that generate the observed data). The forecast distribution at time $\tau + 1$ consists of knowledge of all of the observations of the time series up to the time τ $y_{1:\tau} = (y_1, \ldots, y_\tau)$. The forecast distribution is

$$[y_{\tau+1}|y_{1:\tau}] = [y_{\tau+1}|y_\tau],$$

where the Markov assumption in the autoregressive model says if you know the value of the time series at time τ, the distribution of the forecast at $\tau + 1$ depends only the on the observations $y_{1:\tau}$ only through the value y_τ. Therefore, the one step ahead forecast expected value is

$$
\begin{aligned}
\mathrm{E}(y_{\tau+1}|y_{1:\tau}) &= \mathrm{E}(y_{\tau+1}|y_\tau) && (4)\\
&= \mathrm{E}(\mu_{\tau+1} + \phi(y_\tau - \mu_\tau) + \epsilon_{\tau+1}|y_\tau) && (5)\\
&= \mathrm{E}(\mu_{\tau+1}|y_\tau) + \mathrm{E}(\phi y_\tau|y_\tau) - \mathrm{E}(\phi\mu_\tau|y_\tau) + \mathrm{E}(\epsilon_{\tau+1}|y_\tau) && (6)\\
&= \mu_{\tau+1} + \phi\mathrm{E}(y_\tau|y_\tau) - \phi\mathrm{E}(\mu_\tau|y_\tau) + 0 && (7)\\
&= \mu_{\tau+1} + \phi(y_\tau - \mu_\tau). && (8)\\
& && (9)
\end{aligned}
$$

The k step ahead expected forecast is calculated by using a recursive formula of the equation above giving

$$
\begin{aligned}
\mathrm{E}(y_{\tau+k}|y_{1:\tau}) = \mu_{\tau+k} + \phi^k(y_\tau - \mu_\tau). && (10)\\
&& (11)
\end{aligned}
$$

The k step ahead forecast shows that in the future, we expect the time series to be mean reverting. In other words, as k becomes large, our expected forecast will be close to the mean $\mathrm{E}(y_{t+k}) = \mu_{t+k}$. To fully specify our predictions, we need to calculate the forecast variances. The one step ahead forecast variance is

$$
\begin{aligned}
\mathrm{Var}(y_{\tau+1}|y_{1:\tau}) &= \mathrm{Var}(y_{\tau+1}|y_\tau)\\
&= \mathrm{Var}(\mu_{\tau+1} + \phi(y_\tau - \mu_\tau) + \epsilon_{\tau+1}|y_\tau)\\
&= \mathrm{Var}(\mu_{\tau+1}|y_\tau) + \phi^2\mathrm{Var}(y_\tau - \mu_\tau|y_\tau) + 2\mathrm{Cov}(\mu_{\tau+1} + \phi(y_\tau - \mu_\tau), \epsilon_{\tau+1}|y_\tau) + \mathrm{Var}(\epsilon_{\tau+1}|y_\tau)\\
&= 0 + 0 + 0 + \sigma^2.
\end{aligned}
$$

The k step ahead forecast variance can also be calcuated recursively

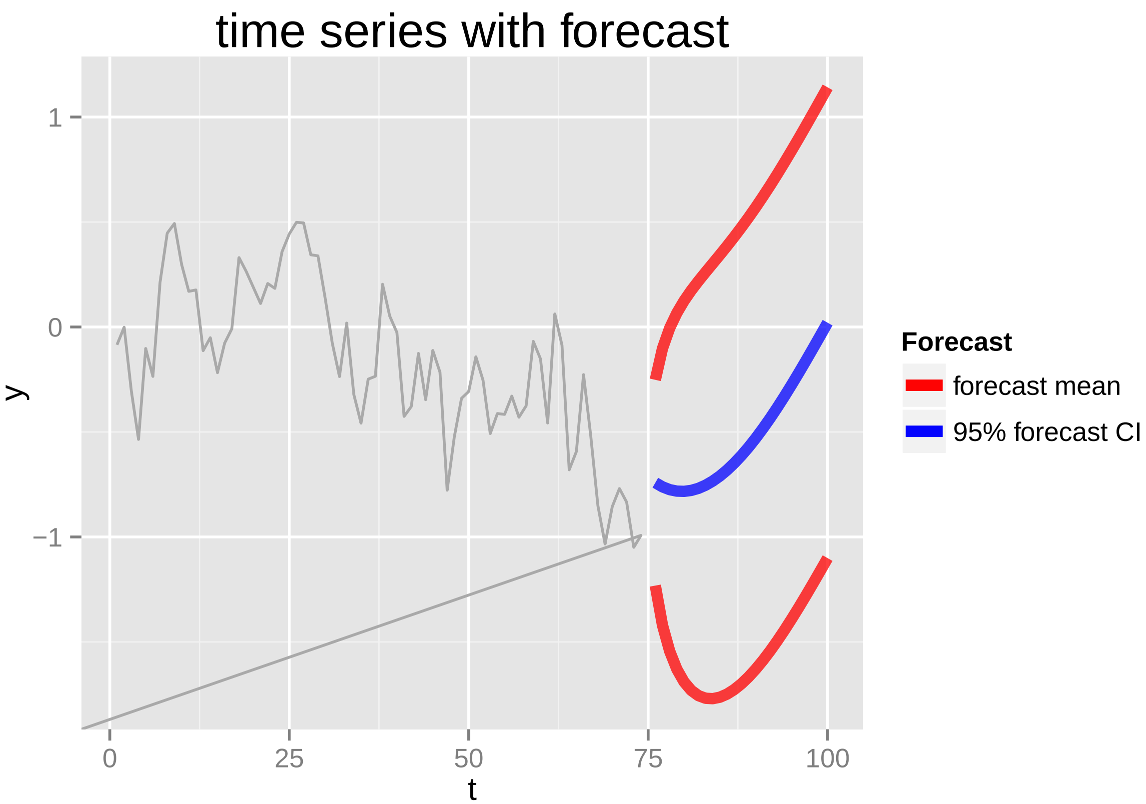
$$
\begin{aligned}
\mathrm{Var}(y_{\tau+k}|y_{1:\tau}) &= \mathrm{Var}(y_{\tau+k}|y_\tau)\\
&= \mathrm{Var}(\mu_{\tau+k} + \phi y_{\tau+k-1} + \epsilon_{\tau+k-1}|y_\tau)\\
&= \mathrm{Var}(\mu_{\tau+k}|y_\tau) + \phi^2\mathrm{Var}(y_{\tau+k-1} - \mu_{\tau+k}|y_\tau) + 2\mathrm{Cov}(\mu_{\tau+k} + \phi(y_{\tau+k-1} - \mu_{\tau+k}), \epsilon_{\tau+k-1}|y_\tau) + \mathrm{Var}(\epsilon_{\tau+k-1}|y_\tau)\\
&= 0 + \phi^2\mathrm{Var}(y_{\tau+k-1}|y_\tau) + 0 + \sigma^2\\
&= 0 + \phi^4\mathrm{Var}(y_{\tau+k-2}|y_\tau) + \phi^2\sigma^2 + \sigma^2\\
&= \vdots\\
&= \sum_{i=1}^{k} \phi^{2(i-1)}\sigma^2,
\end{aligned}
$$

where, as k gets large, the one step ahead forecast variance increases.

```r
set.seed(11)
N <- 1                                  ## replicates N
phi <- 0.9                              ## autocorrelation parameter
mu <- sin(2 * pi * (1:t)/t)           ## pick a mean structure
# mu <- rep(0, t)                         ## pick a mean structure
# mu <- rep(6, t)                         ## pick a mean structure
```

```r
s <- 0.25                              ## pick standard deviation
y <- simTimeSeries(t, N, mu, s, phi)  ## simulate time series

tau <- 75
y[(tau+1):t] <- NA
forecast_data <- data.frame(y=y, t=1:t)
melt_forecast <- melt(forecast_data, id="t")
forecast_mean <- rep(NA, t-tau)
forecast_sd <- rep(NA, t-tau)
for(i in (tau+1):t){
  lag <- i-tau
  forecast_mean[lag] <- mu[i] + phi^(lag) *
                          (y[tau] - mu[tau])
  forecast_sd[lag] <- sqrt(sum(phi^(2 * (1:lag - 1)) * s^2))
}
forecast_data <- data.frame(mean=forecast_mean, sd=forecast_sd,
                            lower_CI=forecast_mean - 1.96 * forecast_sd,
                            upper_CI=forecast_mean + 1.96 * forecast_sd,
                            t=(tau+1):t)
ggplot(data = melt_forecast, aes(y=value, x=t)) +
  geom_line(alpha=1, colour="darkgrey") +
  geom_line(data=forecast_data, aes(y=mean, x=t, colour="forecast mean"),
            alpha=0.75, lwd=2) +
  geom_line(data=forecast_data, aes(y=lower_CI, x=t,
                                    colour="95% forecast CI"), alpha=0.75,
            lty=1, lwd=2) +
  geom_line(data=forecast_data, aes(y=upper_CI, x=t,
                                    colour="95% forecast CI"), alpha=0.75,
            lty=1, lwd=2) +
  scale_colour_manual("Forecast", labels=c("forecast mean", "95% forecast CI"),
                      values=c("forecast mean"="blue",
                               "95% forecast CI"="red")) +
  scale_y_continuous("y") + scale_x_continuous("t") +
  ggtitle("time series with forecast") +
  theme(plot.title = element_text(size=18))
```

```
## Warning in loop_apply(n, do.ply): Removed 25 rows containing missing values
## (geom_path).
```



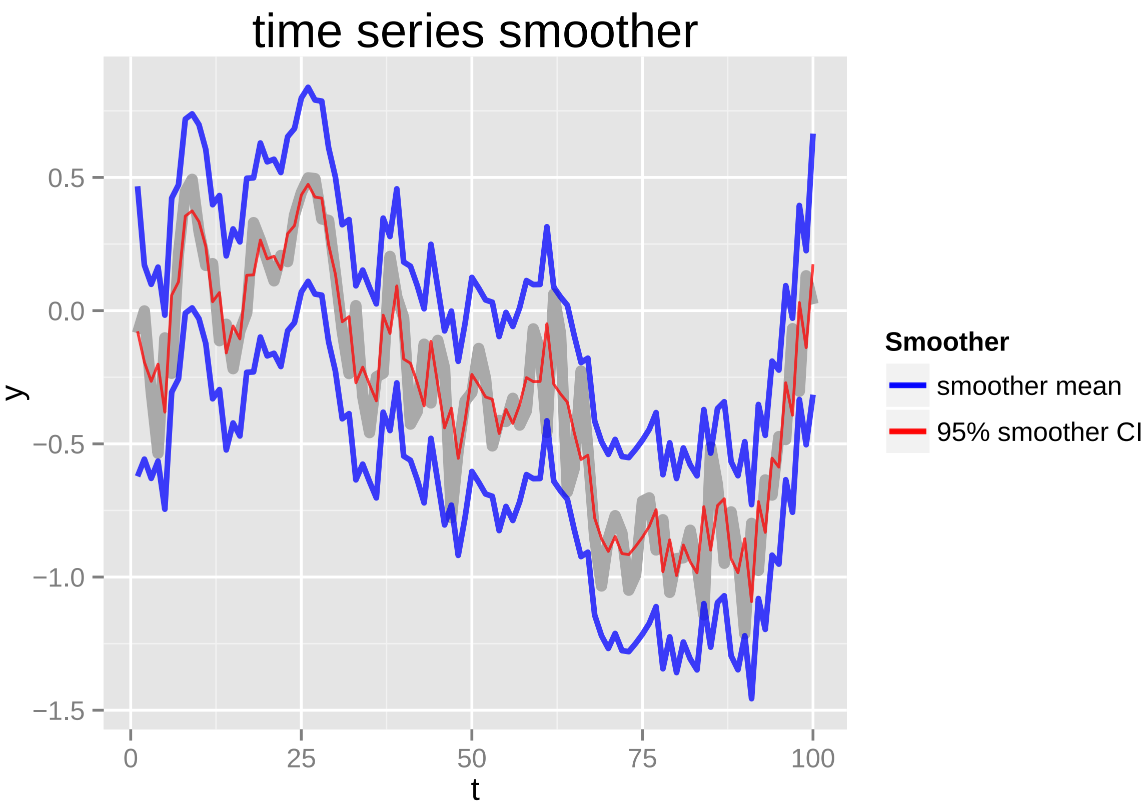
The other distribution of interest is the smoothing distrubtion. The smoothing distribution at time τ is given by $[y_\tau|\mathbf{y}_{-\tau}]$ where $\mathbf{y}_{-\tau}$ consists of all the data points except y_τ. Using our similar difference equation approach as above, we can write the mean of the smoothing distribution at time τ as

$$
\begin{aligned}
[y_\tau|\mathbf{y}_{-\tau}] &= [y_\tau|y_{\tau-1}][y_{\tau+1}|y_\tau] && (12)\\
&\propto \exp\left\{-\frac{\left(y_\tau-\mu_\tau-\phi\left(y_{\tau-1}-\mu_{\tau-1}\right)\right)^2}{2\sigma^2}\right\}\exp\left\{-\frac{\left(y_{\tau+1}-\mu_{\tau+1}-\phi\left(y_\tau-\mu_\tau\right)\right)^2}{2\sigma^2}\right\} && (13)\\
&\propto \exp\left\{-\frac{y_\tau^2\left(1+\phi^2\right)-2y_\tau\left(\mu_\tau+\phi\left(y_{\tau-1}-\mu_{\tau-1}\right)+\phi\left(y_{t+1}-\mu_{\tau+1}+\phi\mu_\tau\right)\right)}{2\sigma^2}\right\} && (14)\\
&\propto \exp\left\{-\frac{y_\tau^2\left(1+\phi^2\right)-2y_\tau\left(\left(1+\phi^2\right)\mu_\tau+\phi\left(y_{\tau-1}-\mu_{\tau-1}\right)+\phi\left(y_{t+1}-\mu_{\tau+1}\right)\right)}{2\sigma^2}\right\} && (15)
\end{aligned}
$$

which is normally distributed with mean $\frac{\left(\left(1+\phi^2\right)\mu_\tau+\phi\left(y_{\tau-1}-\mu_{\tau-1}+y_{t+1}-\mu_{\tau+1}\right)\right)}{1+\phi^2}$ and variance $\frac{\sigma^2}{1+\phi^2}$. At the starting time point $t=1$, the smoothing distribution is $\mathrm{N}\left(\frac{\phi\left(y_{t+1}-\mu_{\tau+1}+\phi\mu_\tau\right)}{\phi^2}, \frac{\sigma^2}{\phi^2}\right)$ and at the endpoint $t=T$, the smoothing distribution is $\mathrm{N}(\mu_\tau+\phi\left(y_{\tau-1}-\mu_{\tau-1}\right), \sigma^2)$.

```r
set.seed(11)
N <- 1                              ## replicates N
phi <- 0.9                          ## autocorrelation parameter
mu <- sin(2 * pi * (1:t)/t)         ## pick a mean structure
# mu <- rep(0, t)                     ## pick a mean structure
# mu <- rep(6, t)                     ## pick a mean structure
s <- 0.25                           ## pick standard deviation
y <- simTimeSeries(t, N, mu, s, phi) ## simulate time series
```

```
smoother_mean <- rep(NA, t)
smoother_sd <- rep(NA, t)
for(i in 1:t){
  if (i==1) {
    smoother_mean[i] <- (phi * (y[i+1] - mu[i+1] + phi * mu[i])) / (phi^2)
    smoother_sd[i] <- sqrt(s^2/phi^2)
  } else if (i == t) {
    smoother_mean[i] <- (mu[i] + phi * (y[i-1] - mu[i-1]))
    smoother_sd[i] <- sqrt(s^2)
  } else {
    smoother_mean[i] <- (mu[i] + phi * (y[i-1] - mu[i-1]) + phi *
                          (y[i+1] - mu[i+1] + phi * mu[i])) / (1 + phi^2)
    smoother_sd[i] <- sqrt(s^2/(1 + phi^2))
  }
}

smoother_data <- data.frame(mean=smoother_mean, sd=smoother_sd, t=1:t,
                            lower_CI=smoother_mean - 1.96 * smoother_sd,
                            upper_CI=smoother_mean + 1.96 * smoother_sd)
melt_smoother <- melt(data.frame(y=y, t=1:t), id="t")
ggplot(data = melt_smoother, aes(y=value, x=t)) +
  geom_line(alpha=1, colour="darkgrey", lwd=2) +
  geom_line(data=smoother_data, aes(y=mean, x=t, colour="smoother mean"),
            alpha=0.75) +
  geom_line(data=smoother_data, aes(y=lower_CI, x=t,
                                    colour="95% smoother CI"), alpha=0.75,
            lty=1, lwd=1) +
  geom_line(data=smoother_data, aes(y=upper_CI, x=t,
                                    colour="95% smoother CI"), alpha=0.75,
            lty=1, lwd=1) +
  scale_colour_manual("Smoother",
                      labels=c("smoother mean", "95% smoother CI"),
                      values=c("smoother mean"="red",
                               "95% smoother CI"="blue")) +
  scale_y_continuous("y") + scale_x_continuous("t") +
  ggtitle("time series smoother") +
  theme(plot.title = element_text(size=18))
```



Now that we have derived the quantities and distributions of most interest, we can re-write (2) as the vectorized model

$$\mathbf{y} = \boldsymbol{\mu} + \boldsymbol{\eta} \tag{16}$$

where $\boldsymbol{\mu}$ is a vector of means (often $\boldsymbol{\mu} = \mathbf{X}\boldsymbol{\beta}$ the regression model) and $\boldsymbol{\eta} \sim \text{N}\left(\mathbf{0}, \sigma^2\mathbf{R}(\phi)\right)$ where

$$\mathbf{R}(\phi) = \begin{pmatrix} 1 & \phi & \phi^2 & \cdots & \phi^{T-2} & \phi^{T-1} \\ \phi & 1 & \phi & \cdots & \phi^{T-3} & \phi^{T-2} \\ \phi^2 & \phi & 1 & \cdots & \phi^{T-4} & \phi^{T-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \phi^{T-1} & \phi^{T-2} & \phi^{T-3} & \cdots & \phi & 1 \end{pmatrix}.$$

Notice that the mean is the same as in the difference equations and the parameters for the covariance matrix $\mathbf{\Sigma}$ include the variance on the diagonal and the covariances on the off diagonal. A natural question is what was gained by writing (2) in a vector format? Primarily, the benefits are reduced computation time when fitting models and generating predicitons. Additionally, one can use a basis expansion to reduce computational effort. If we are focused on the forecast distribution, we can write the forecast distribution (10) as

$$\boldsymbol{\mu}_{2:T} + \mathbf{Z}\mathbf{R}(\phi)^{-1}(\mathbf{y} - \boldsymbol{\mu})$$

for a proper choice of $T-1 \times t$ matrix $\mathbf{Z}$. We know from (10) that $\mathbf{Z}\mathbf{R}(\phi)^{-1}(\mathbf{y} - \boldsymbol{\mu}) = (\mu_2 + \phi(y_1 - \mu_1), \mu_3 + \phi(y_2 - \mu_2), \ldots, \mu_T + \phi(y_{T-1} - \mu_{T-1}))'$ which implies

$$\mathbf{Z}\mathbf{R}^{-1}(\phi) = \begin{pmatrix} \phi & 0 & 0 & \cdots & 0 & 0 \\ 0 & \phi & 0 & \cdots & 0 & 0 \\ 0 & 0 & \phi & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \phi & 0 \end{pmatrix}$$

which, when solved for $\mathbf{Z}$ is

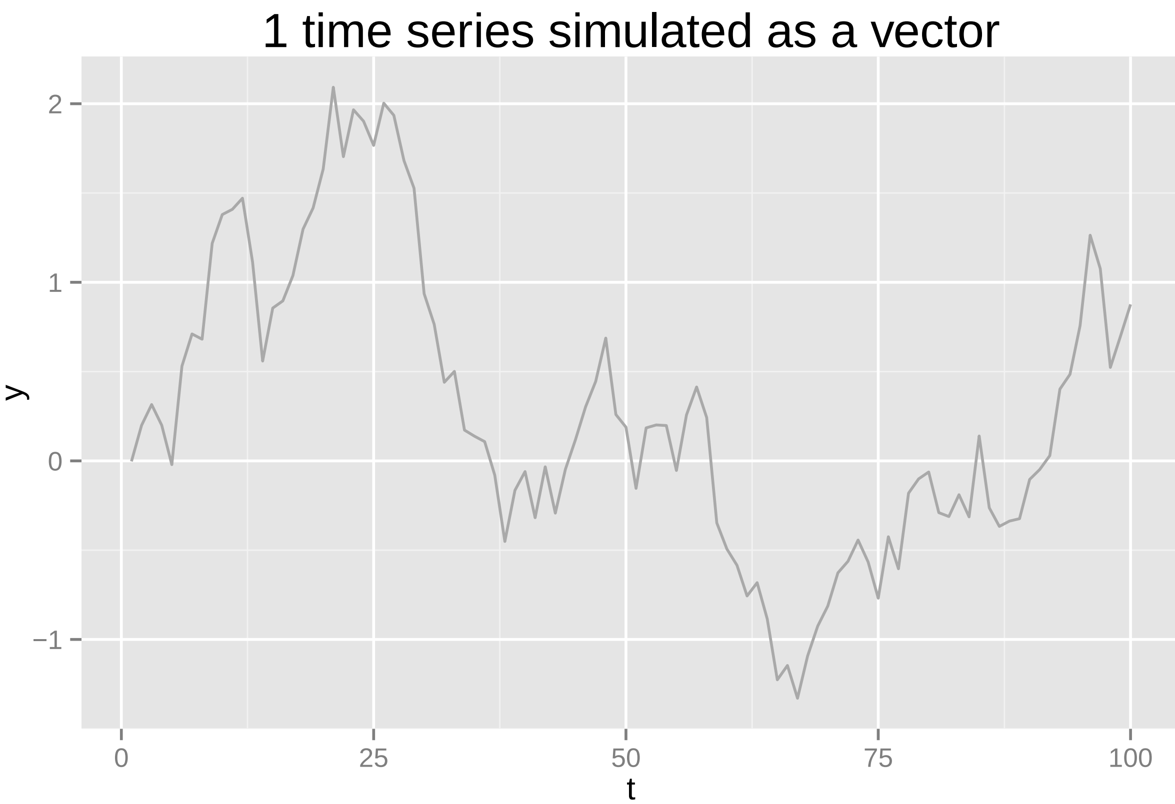
$$\mathbf{Z} = \begin{pmatrix} \phi & 0 & 0 & \cdots & 0 & 0 \\ 0 & \phi & 0 & \cdots & 0 & 0 \\ 0 & 0 & \phi & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \phi & 0 \end{pmatrix} \mathbf{R}(\phi) \tag{17}$$

$$= \begin{pmatrix} \phi & 0 & 0 & \cdots & 0 & 0 \\ 0 & \phi & 0 & \cdots & 0 & 0 \\ 0 & 0 & \phi & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \phi & 0 \end{pmatrix} \begin{pmatrix} 1 & \phi & \phi^2 & \cdots & \phi^{T-2} & \phi^{T-1} \\ \phi & 1 & \phi & \cdots & \phi^{T-3} & \phi^{T-2} \\ \phi^2 & \phi & 1 & \cdots & \phi^{T-4} & \phi^{T-3} \\ \phi^3 & \phi^2 & \phi & \cdots & \phi^{T-5} & \phi^{T-4} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \phi^{T-1} & \phi^{T-2} & \phi^{T-3} & \cdots & \phi & 1 \end{pmatrix} \tag{18}$$

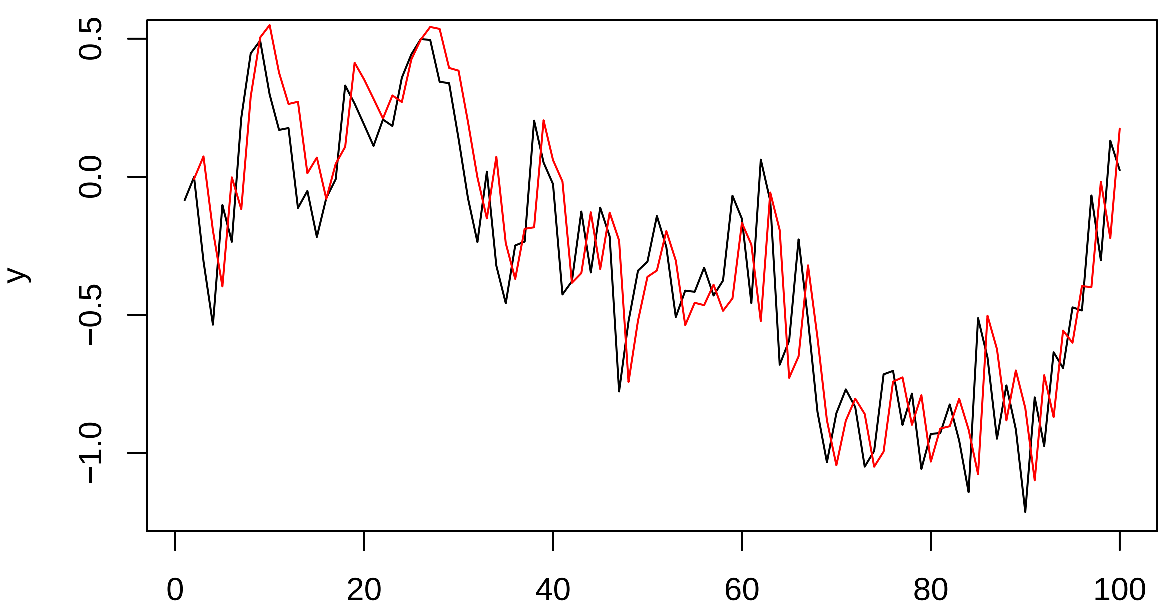
$$= \begin{pmatrix} \phi & \phi^2 & \phi^3 & \cdots & \phi^{T-1} & \phi^T \\ \phi^2 & \phi & \phi^2 & \cdots & \phi^{T-2} & \phi^{T-1} \\ \phi^3 & \phi^2 & \phi & \cdots & \phi^{T-3} & \phi^{T-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \phi^{T-1} & \phi^{T-2} & \phi^{T-3} & \cdots & \phi & \phi^2 \end{pmatrix} \tag{19}$$

$$\tag{20}$$

```r
R <- toeplitz(phi^(0:(t-1)))
Sigma <- s^2 / (1 - phi^2) * R
library(mvtnorm)
y_vec <- t(rmvnorm(N, mu, Sigma))

vec_data <- data.frame(y=y_vec, t=1:t)
melt_vec <- melt(vec_data, id="t")

ggplot(data = melt_vec, aes(y=value, x=t)) +
  geom_line(alpha=1, colour="darkgrey") +
  scale_y_continuous("y") + scale_x_continuous("t") +
  ggtitle(paste(min(N, 10),
                "time series simulated as a vector")) +
  theme(plot.title = element_text(size=18))
```




```
Z <- toeplitz(c(phi^(1:t)))[1:(t-1), ]       ## construct the (t-1) by t basis
                                              ## matrix for one step ahead
                                              ## forecasting

y_hat_one_step_ahead <- mu[2:t] + Z %*% solve(R) %*% (y - mu)
matplot(y, type = 'l')
lines(2:t, y_hat_one_step_ahead, col = "red")
```



```
# sd_y_hat_one_step_ahead <-
#
#   diag(Z %*% solve(R) %*% (y - mu) %*% t(solve(R) %*% (y - mu)) %*% t(Z))
#
#   diag(Z %*% solve(R) %*% (s^2 * R) %*% t(solve(R)) %*% t(Z))
#
# sqrt(s^2 / (1-phi^2))
```

Likewise for the smoother distribution (12) gives $\mathbf{ZR}(\phi)^{-1}(\mathbf{y}-\boldsymbol{\mu}) = (\phi^2\mu_1 + \phi(y_2-\mu_2), (1+\phi^2)\mu_2 + \phi(y_1 - \mu_1 + y_3 - \mu_3), \ldots, (1+\phi^2)\mu_{T-1} + \phi(y_{T-2} - \mu_{T-2} + y_T - \mu_T), \mu_T + \phi(y_{T-1} - \mu_{T-1}))'$ which implies

$$\mathbf{ZR}^{-1}(\phi) = \begin{pmatrix} \phi & 0 & 0 & \cdots & 0 & 0 \\ 0 & \phi & 0 & \cdots & 0 & 0 \\ 0 & 0 & \phi & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \phi & 0 \end{pmatrix}$$

which, when solved for $\mathbf{Z}$ is

$$\mathbf{Z} = \begin{pmatrix} \phi & 0 & 0 & \cdots & 0 & 0 \\ 0 & \phi & 0 & \cdots & 0 & 0 \\ 0 & 0 & \phi & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \phi & 0 \end{pmatrix} \mathbf{R}(\phi) \tag{21}$$

$$= \begin{pmatrix} \phi & 0 & 0 & \cdots & 0 & 0 \\ 0 & \phi & 0 & \cdots & 0 & 0 \\ 0 & 0 & \phi & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \phi & 0 \end{pmatrix} \begin{pmatrix} 1 & \phi & \phi^2 & \cdots & \phi^{T-2} & \phi^{T-1} \\ \phi & 1 & \phi & \cdots & \phi^{T-3} & \phi^{T-2} \\ \phi^2 & \phi & 1 & \cdots & \phi^{T-4} & \phi^{T-3} \\ \phi^3 & \phi^2 & \phi & \cdots & \phi^{T-5} & \phi^{T-4} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \phi^{T-1} & \phi^{T-2} & \phi^{T-3} & \cdots & \phi & 1 \end{pmatrix} \tag{22}$$

$$= \begin{pmatrix} \phi & \phi^2 & \phi^3 & \cdots & \phi^{T-1} & \phi^T \\ \phi^2 & \phi & \phi^2 & \cdots & \phi^{T-2} & \phi^{T-1} \\ \phi^3 & \phi^2 & \phi & \cdots & \phi^{T-3} & \phi^{T-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \phi^{T-1} & \phi^{T-2} & \phi^{T-3} & \cdots & \phi & \phi^2 \end{pmatrix} \tag{23}$$

$$\tag{24}$$