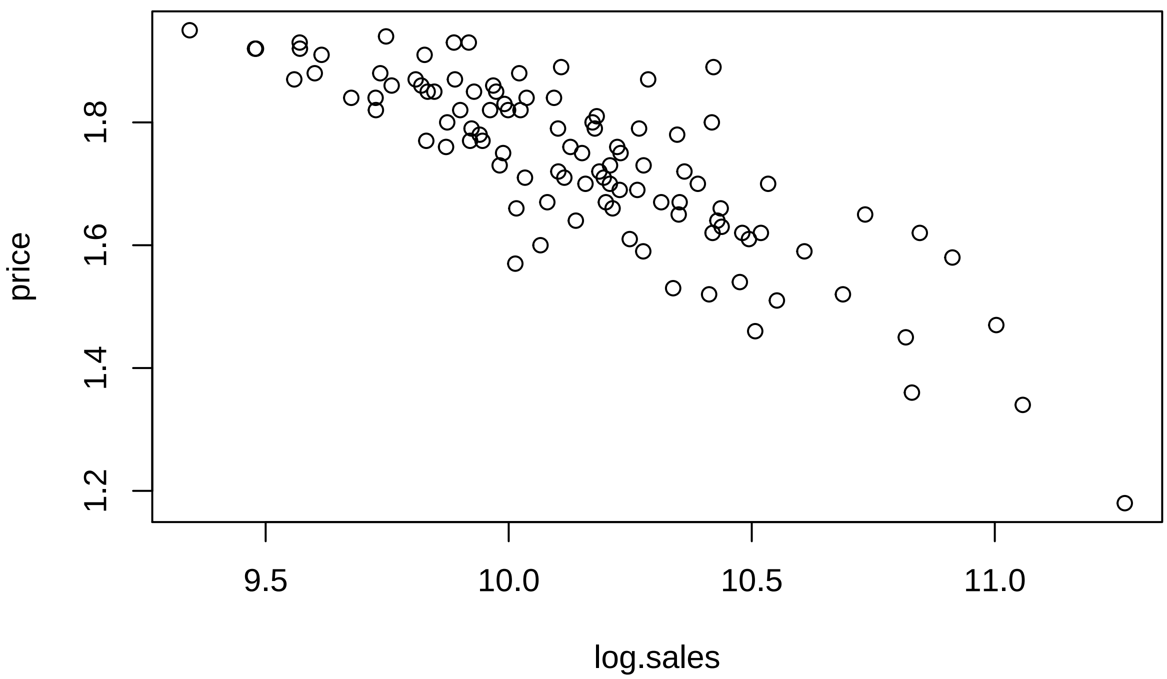
ST565: Time Series HW6

Amirhosein "Emerson" Azarbakht azarbaka@oregonstate.edu

Question 1

The dataset bluebirdlite contains log sales and prices for the "lite" version of bluebird chips. Quantify the relationship between sales and price.

```r
plot(bluebirdlite)
```

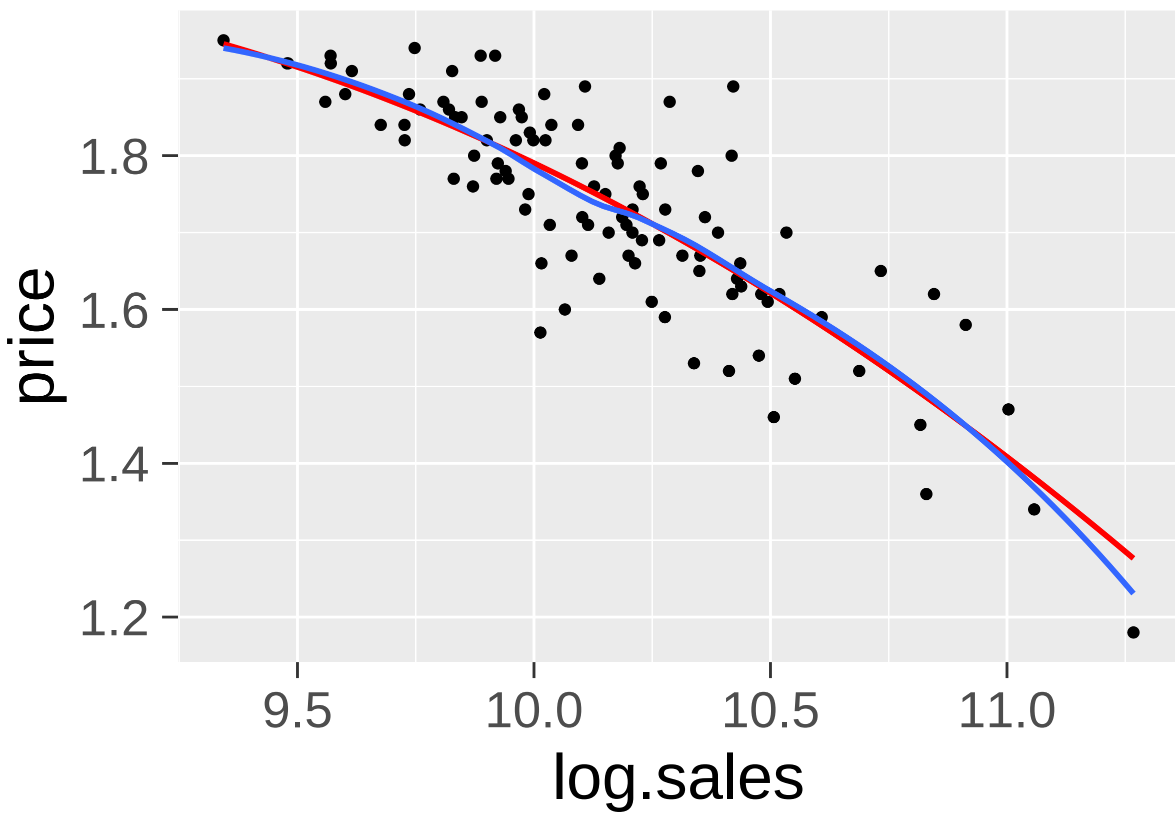



```r
big_font <- theme_grey(base_size =  24)


attach(bluebirdlite)

fit_lm <- lm(log.sales ~ price + I(price^2), data = bluebirdlite)
summary(fit_lm)
```

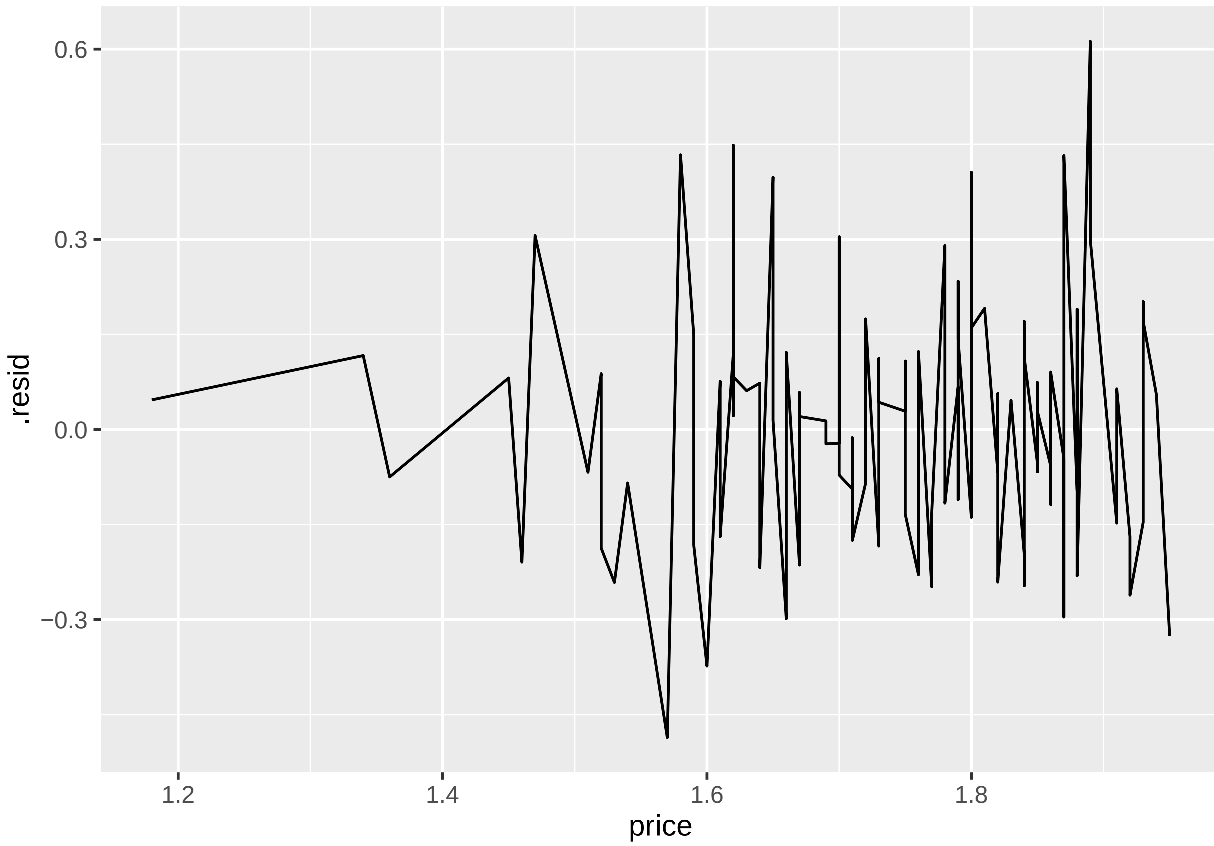
```
##
## Call:
## lm(formula = log.sales ~ price + I(price^2), data = bluebirdlite)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.48609 -0.14069  0.01389  0.11249  0.61228
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  12.5954     1.6938   7.436 3.48e-11 ***
## price        -0.6507     2.0404  -0.319    0.750
## I(price^2)   -0.4357     0.6121  -0.712    0.478
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1968 on 101 degrees of freedom
## Multiple R-squared:  0.7054, Adjusted R-squared:  0.6996
## F-statistic: 120.9 on 2 and 101 DF,  p-value: < 2.2e-16
```

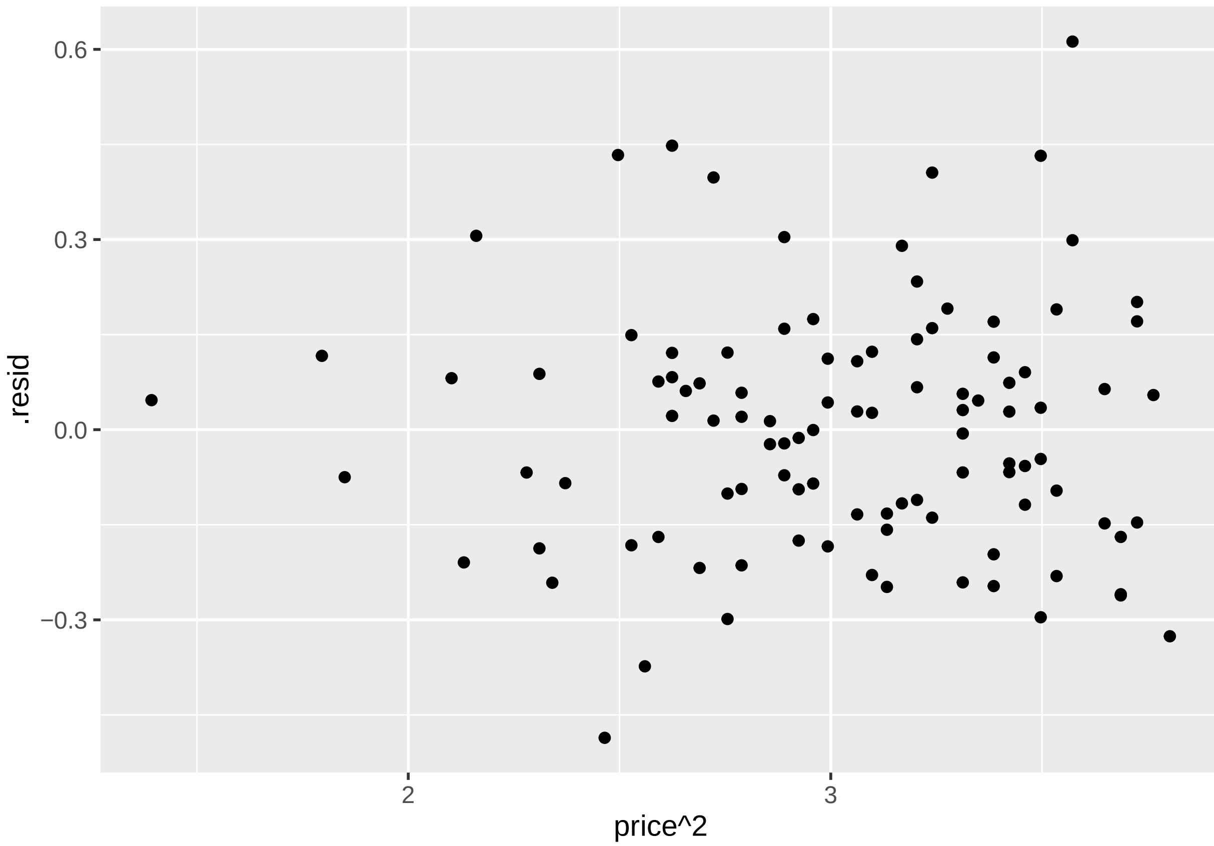
```
qplot(log.sales, price, data = bluebirdlite) +
  geom_smooth(method = "lm", formula = y ~ poly(x, 2), se = FALSE, colour = "red") +
  geom_smooth(se = FALSE)  +
  big_font
```



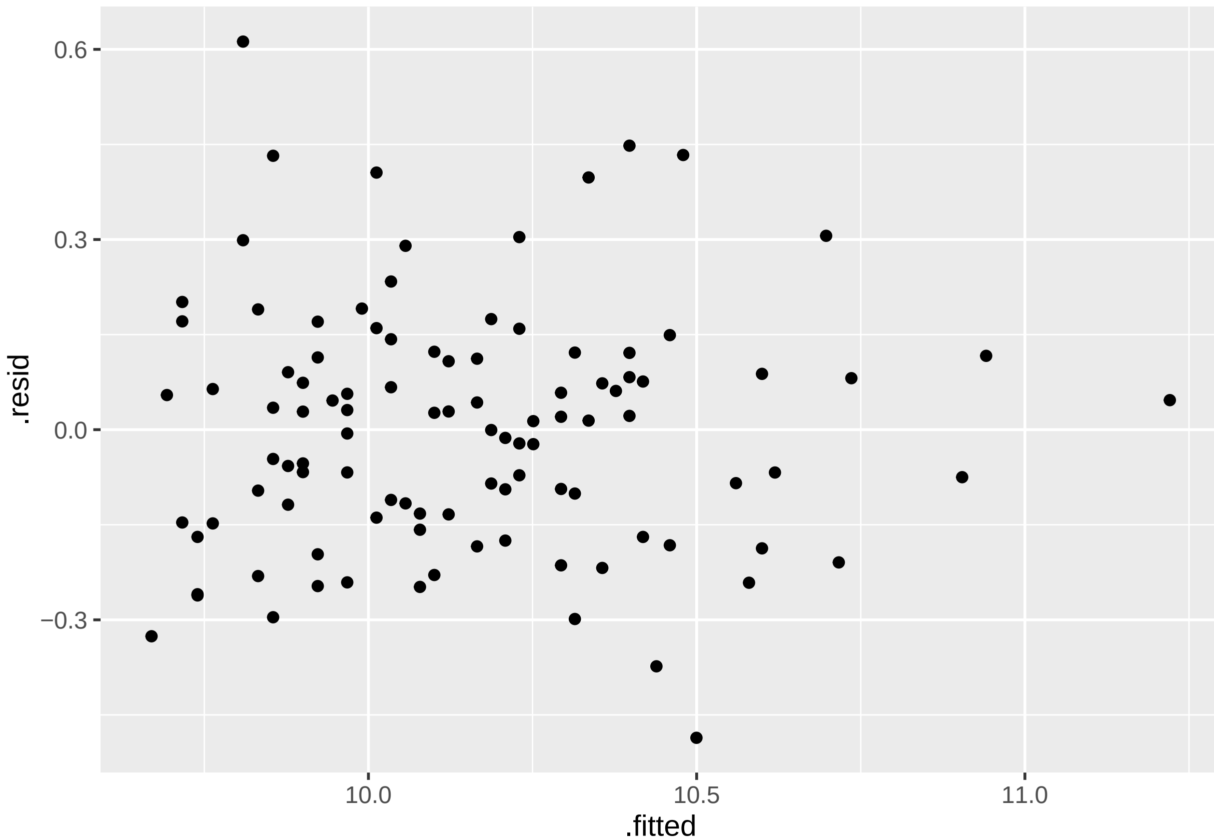

```
# assumptions
# residuals versus covariates
sales_lm <- fortify(fit_lm)
qplot(price, .resid, data = sales_lm, geom= "line")
```



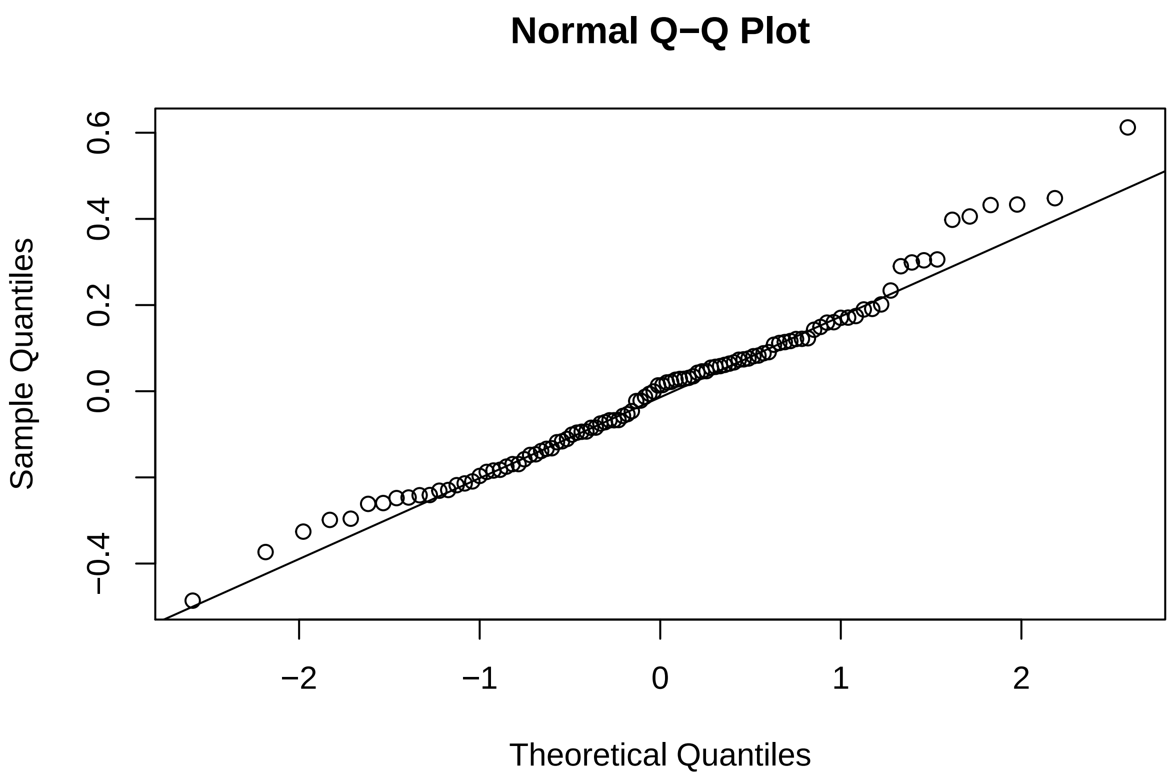

```
qplot(price^2, .resid, data = sales_lm)
```




```
# residuals versus fitted
qplot(.fitted, .resid, data = sales_lm)
```

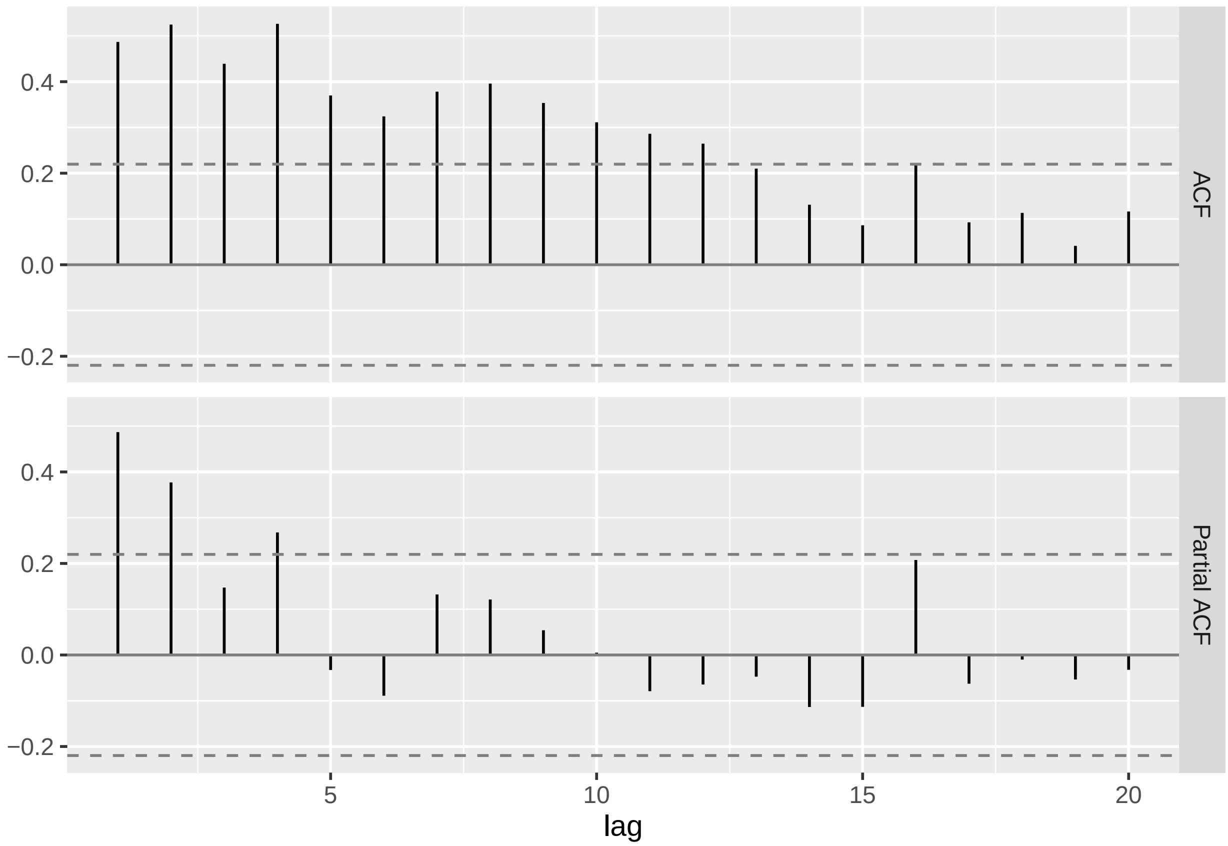



```r
# normality of residuals
qqnorm(sales_lm$.resid)
qqline(sales_lm$.resid)
```

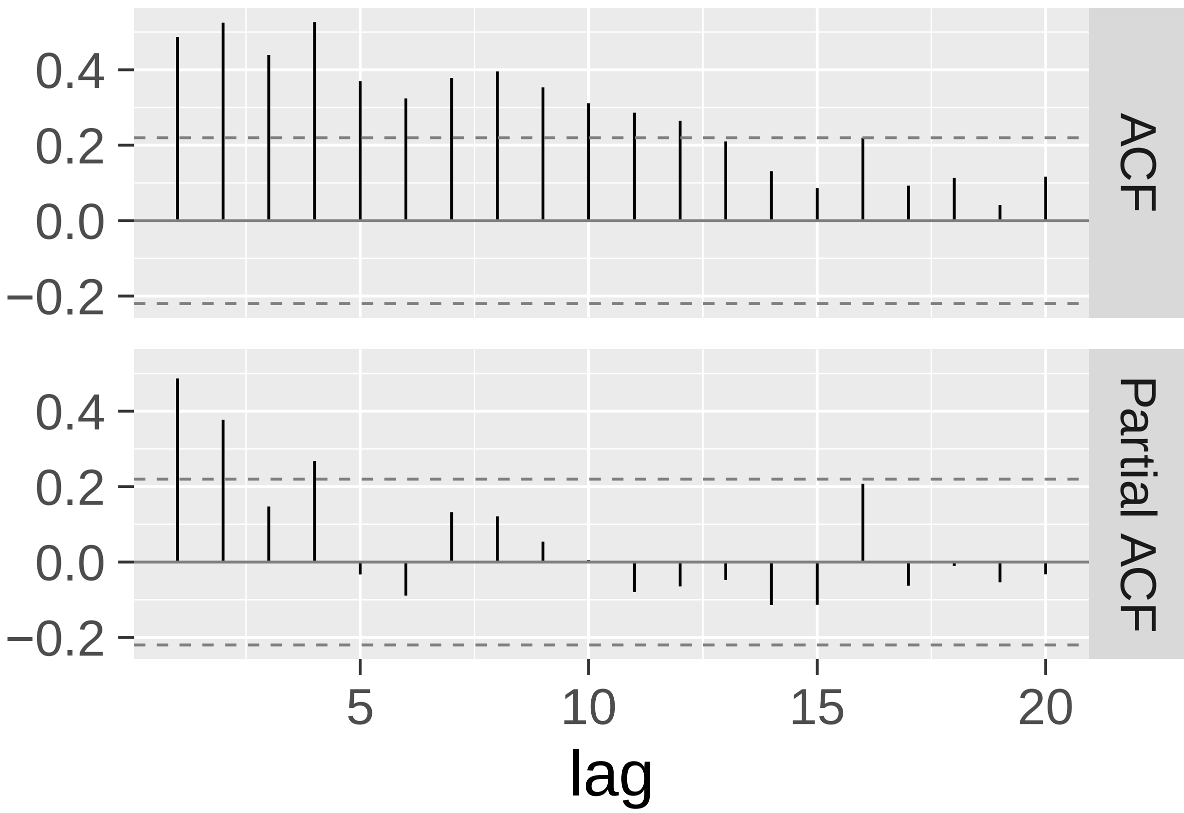



```r
# correlation of residuals
source(url("http://stat565.cwick.co.nz/code/get_acf.R")) # my code for examine_corr
examine_corr(residuals(fit_lm))
```

```
## Warning: closing unused connection 5 (http://stat565.cwick.co.nz/code/
## get_acf.R)
```




```r
last_plot() + big_font
```

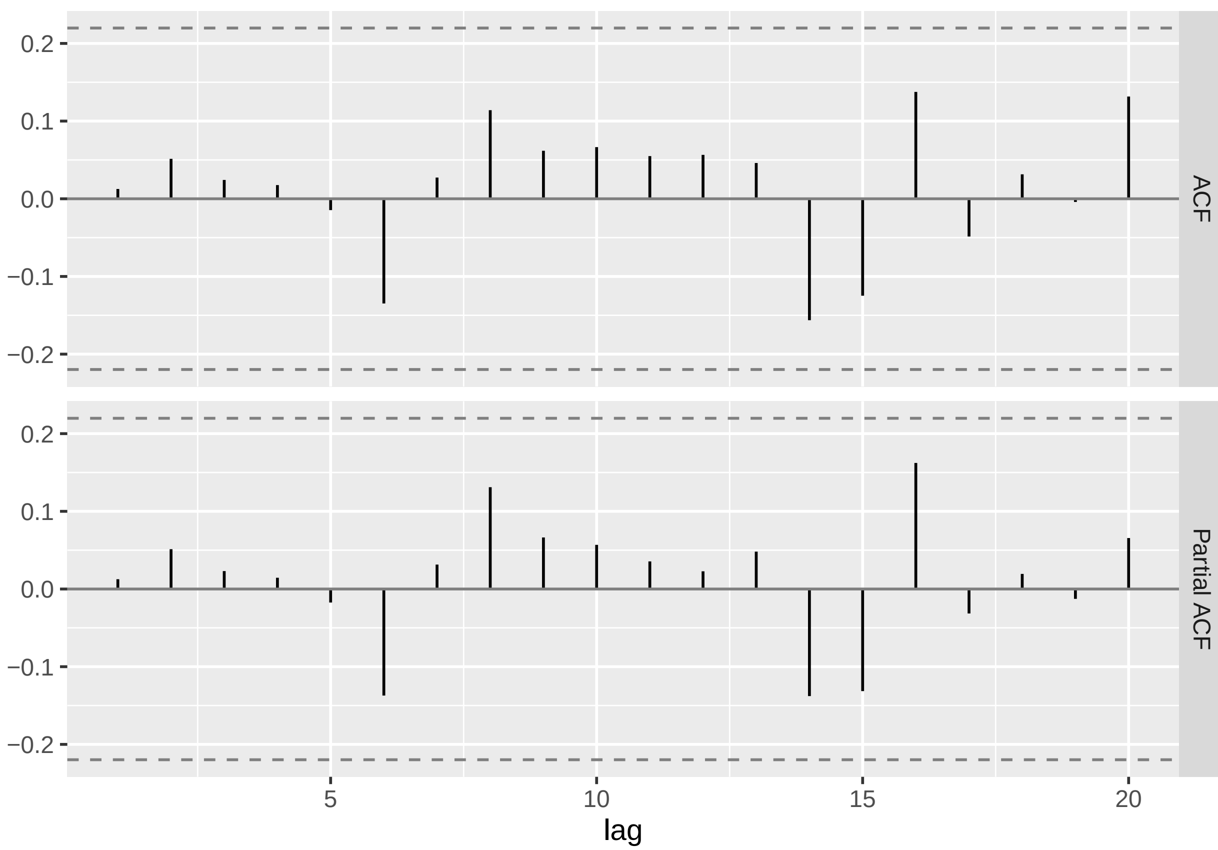



```r
# AR (4)? violates regression assumptions

library(nlme)
gls_fit <- gls(log.sales ~ price + I(price^2), data = bluebirdlite,
    correlation = corARMA(p = 4), method = "ML")
summary(gls_fit)
```

```
## Generalized least squares fit by maximum likelihood
##   Model: log.sales ~ price + I(price^2)
##   Data: bluebirdlite
##         AIC       BIC   logLik
##   -92.6015 -71.44638 54.30075
##
## Correlation Structure: ARMA(4,0)
##  Formula: ~1
##  Parameter estimate(s):
##      Phi1      Phi2      Phi3      Phi4
## 0.1851510 0.2132629 0.1345612 0.3039307
##
## Coefficients:
##                 Value Std.Error   t-value p-value
## (Intercept) 14.899928 1.1651084 12.788448  0.0000
## price       -3.658602 1.3918337 -2.628620  0.0099
## I(price^2)   0.523881 0.4163226  1.258355  0.2112
##
##  Correlation:
```

```
##            (Intr) price
## price      -0.994
## I(price^2)  0.985 -0.997
##
## Standardized residuals:
##         Min          Q1         Med          Q3         Max
## -2.23177526 -0.70553535  0.04652695  0.65792580  2.90668252
##
## Residual standard error: 0.1943017
## Degrees of freedom: 104 total; 101 residual
```

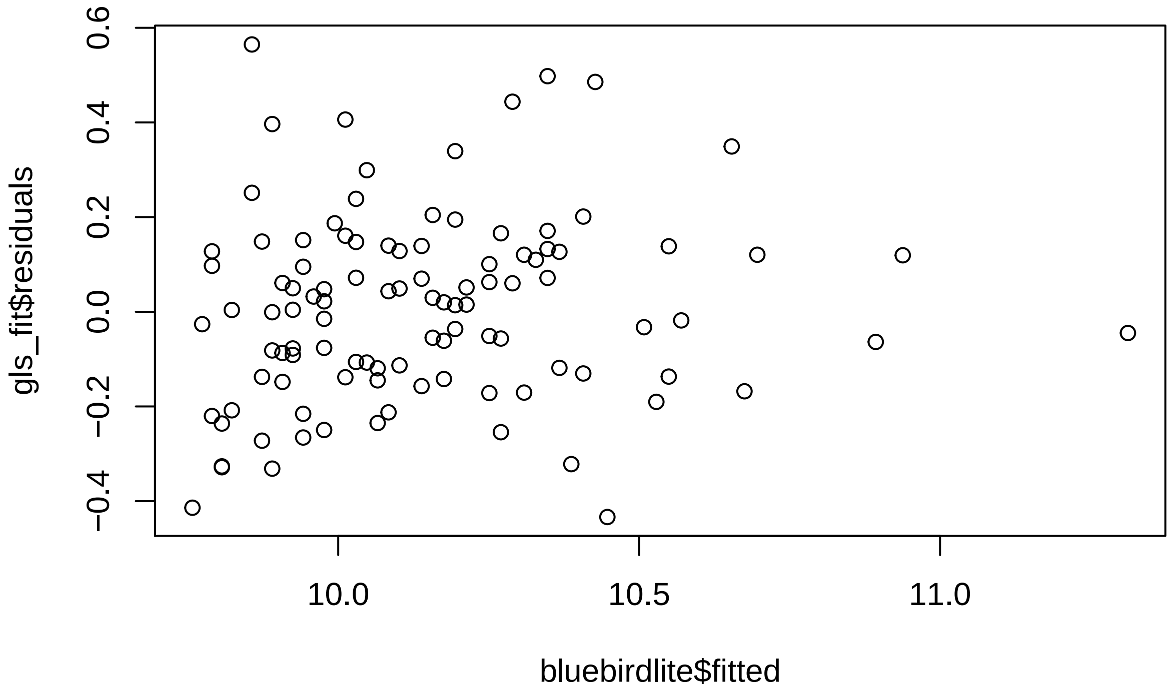
```
# or
# arima_fit <- with(bluebirdlite,
#   arima(log.sales, order = c(4, 0, 1), xreg = cbind(price, I(price^2))))
# arima_fit

# diagnostics
fit_lm$residuals <- residuals(gls_fit, type = "normalized")
bluebirdlite$fitted <- fitted(gls_fit)

examine_corr(fit_lm$residuals) #looks good
```



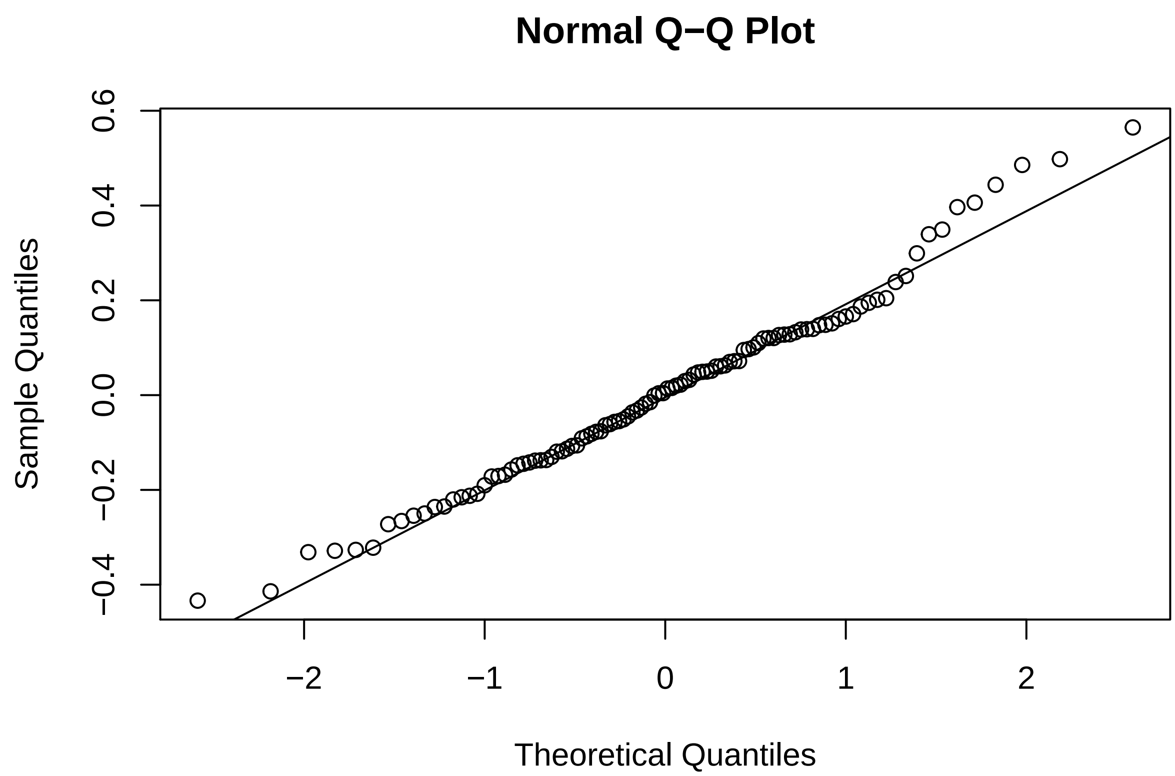

```
plot(bluebirdlite$fitted, gls_fit$residuals, data = bluebirdlite)
```

```
## Warning in plot.window(...): "data" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "data" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "data" is not
## a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "data" is not
## a graphical parameter

## Warning in box(...): "data" is not a graphical parameter

## Warning in title(...): "data" is not a graphical parameter
```




```r
qqnorm(gls_fit$residuals)
qqline(gls_fit$residuals) #Looks okay
```




```r
confint(gls_fit)
```

```
##                  2.5 %     97.5 %
## (Intercept) 12.6163578 17.1834987
## price       -6.3865463 -0.9306585
## I(price^2)  -0.2920958  1.3398587
```

```r
confint(fit_lm)
```

```
##                  2.5 %    97.5 %
## (Intercept)  -4.733007 29.923726
## price       -21.524851 20.223381
## I(price^2)   -6.697864  5.826414
```

```r
intervals(gls_fit)
```

```
## Approximate 95% confidence intervals
##
##  Coefficients:
##                  lower       est.      upper
## (Intercept) 12.5886666 14.8999282 17.2111898
## price       -6.4196260 -3.6586024 -0.8975787
## I(price^2)  -0.3019906  0.5238814  1.3497534
## attr(,"label")
## [1] "Coefficients:"
##
```

```
##  Correlation structure:
##           lower      est.       upper
## Phi1 0.10104695 0.1851510 -0.06254519
## Phi2 0.12509406 0.2132629  0.23165667
## Phi3 0.01130536 0.1345612  0.32715590
## Phi4 0.11381837 0.3039307  0.47257119
## attr(,"label")
## [1] "Correlation structure:"
##
##  Residual standard error:
##     lower      est.     upper
## 0.1394571 0.1943017 0.2707152
```

```r
# backtransform and use % decrease with price increase of 0.1
est_gls <- 100*(1 - exp(0.1 * coef(gls_fit)["price"]))
ci_gls <- 100*(1 - exp(0.1 * confint(gls_fit)["price", ]))
```

It is estimated that an increase in price of 10 cents is associated with decrease in median sales of 30.64 (95% CI 47.2 to 8.89).

Question 2

In class we looked at modelling the relationship between mortality, temperature and particulate matter. Repeat the analysis but seasonally difference all three series first. Compare the results.

```r
library(ggplot2)
library(dplyr)
# install.packages("tidyr")
library(tidyr)
load(url("http://www.stat.pitt.edu/stoffer/tsa3/tsa3.rda"))
source(url("http://stat565.cwick.co.nz/code/fortify-ts.r"))
source(url("http://stat565.cwick.co.nz/code/get_acf.R")) # my code for examine_corr
big_font <- theme_grey(base_size =  24)

# mort <- c(rep(NA,52), diff(cmort, lag = 52))
# temp <- c(rep(NA,52), diff(tempr, lag = 52))
# part <- c(rep(NA,52), diff(part, lag = 52))

mort <- diff(cmort, lag = 52)
temp <- diff(tempr, lag = 52)
part <- diff(part, lag = 52)


mort <- data.frame(mortality = mort, part = part, temp = temp)
mort$time <- fortify(cmort)$time[53:508]
# mort
head(mort)
```

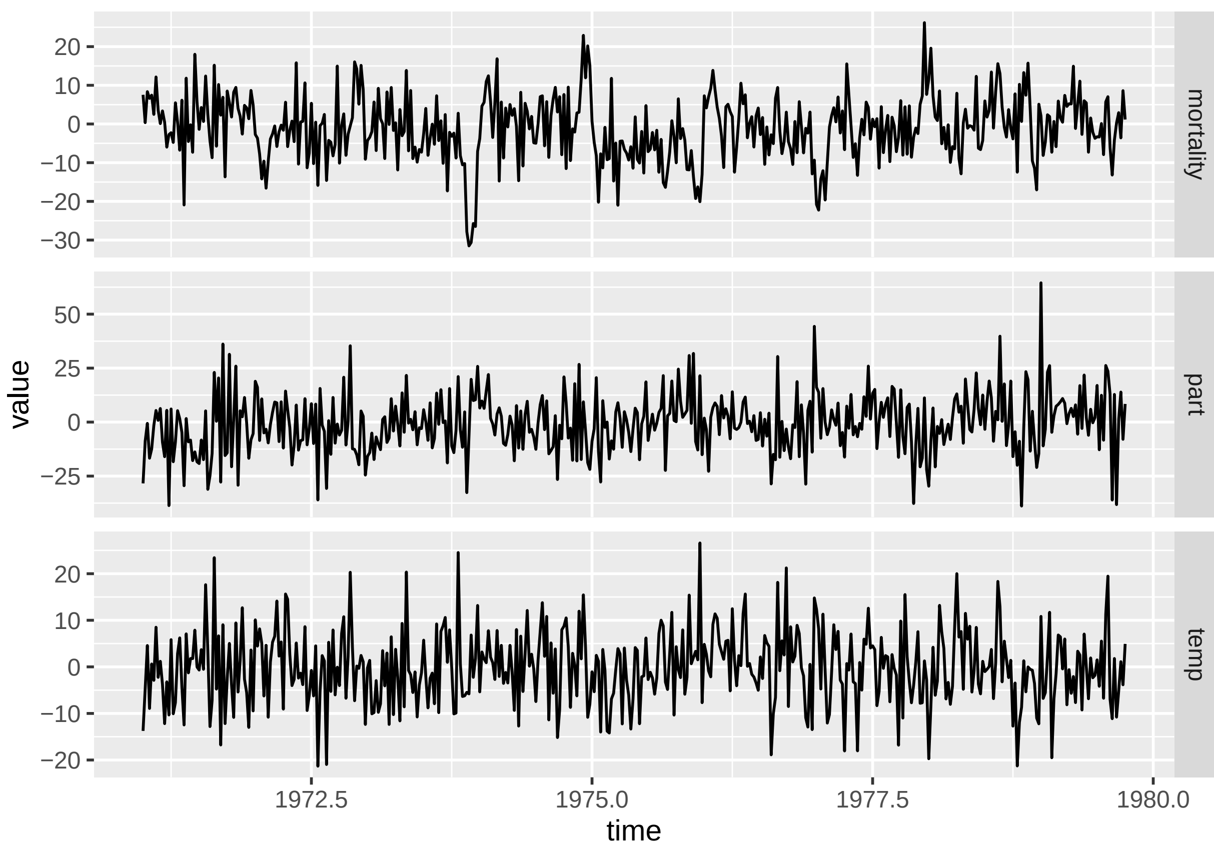
```
##   mortality   part   temp     time
## 1      7.54 -28.38 -13.75 1971.000
```

```
## 2      0.35  -8.80  -4.69 1971.019
## 3      8.36  -0.59   4.58 1971.038
## 4      6.62 -16.72  -8.93 1971.058
## 5      7.45 -12.48   0.62 1971.077
## 6      2.48  -1.55  -2.90 1971.096
```

```
qplot(time, value, data = gather(mort, variable, value, -time), geom = "line") +
  facet_grid(variable ~ ., scale = "free")
```

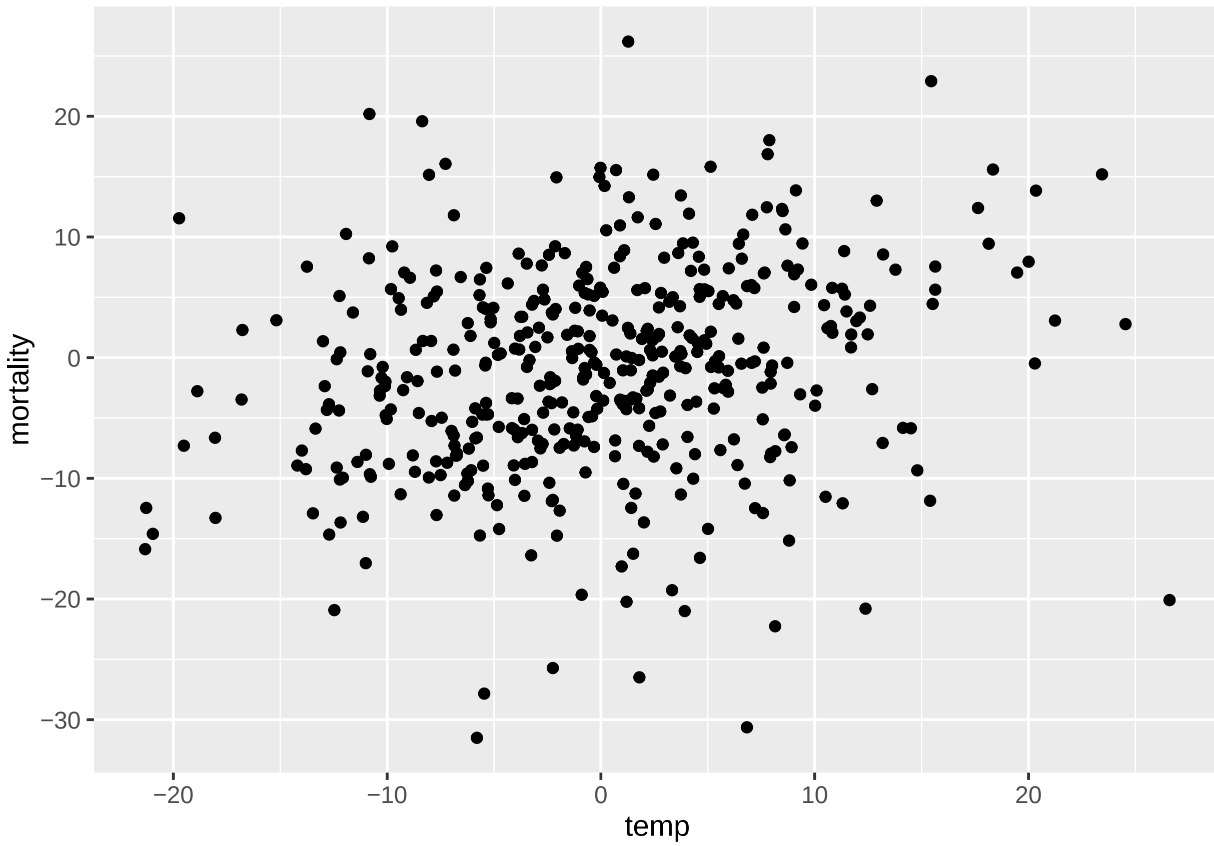
```
## Warning: closing unused connection 7 (http://stat565.cwick.co.nz/code/
## get_acf.R)
```

```
## Warning: closing unused connection 6 (http://stat565.cwick.co.nz/code/
## fortify-ts.r)
```



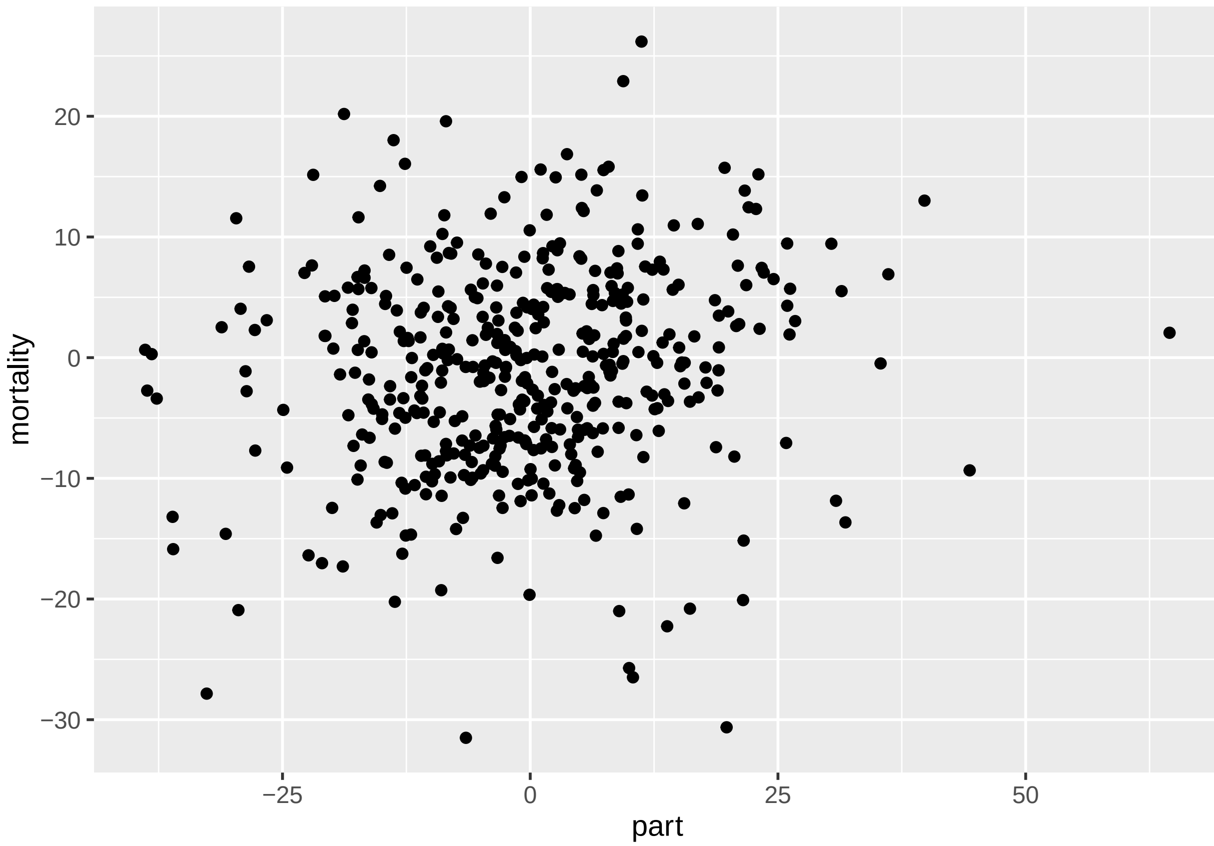

```
qplot(temp, mortality, data = mort)
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
```



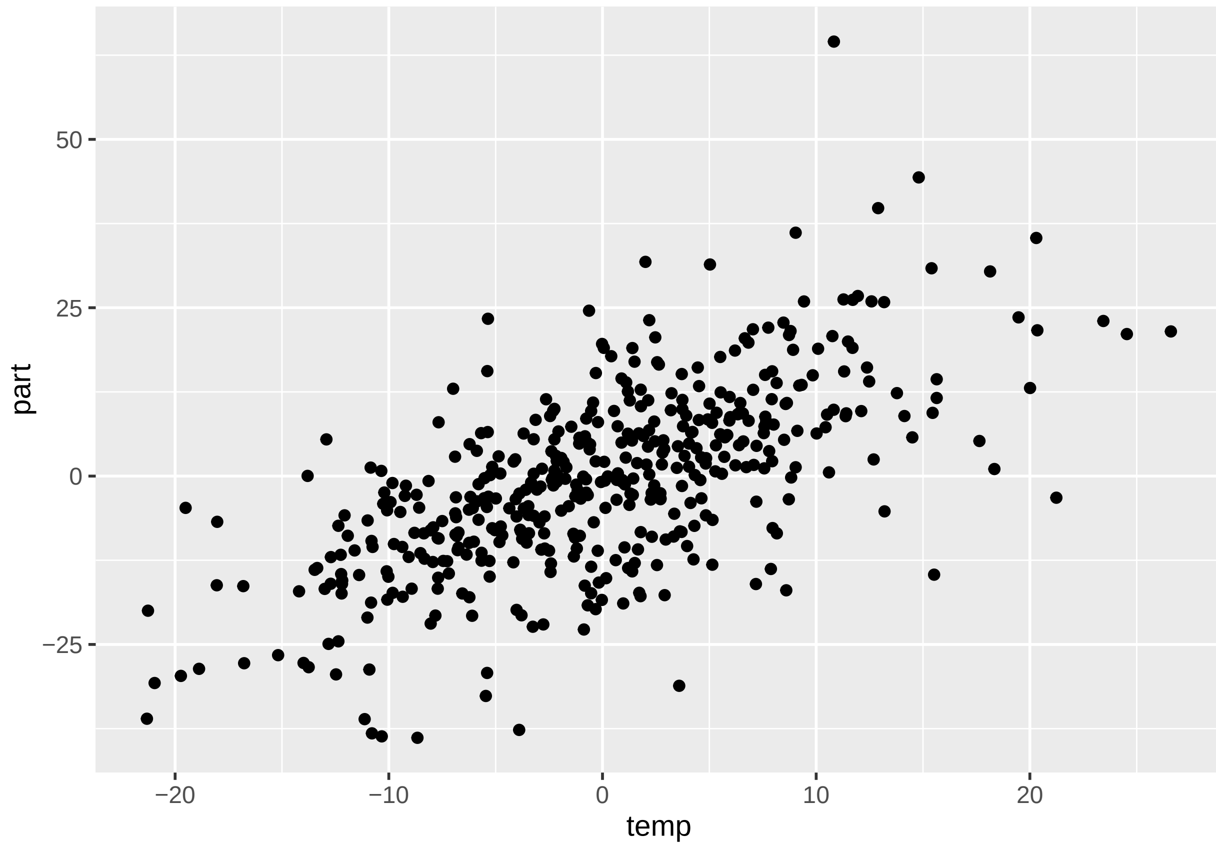

```r
qplot(part, mortality, data = mort)
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
```



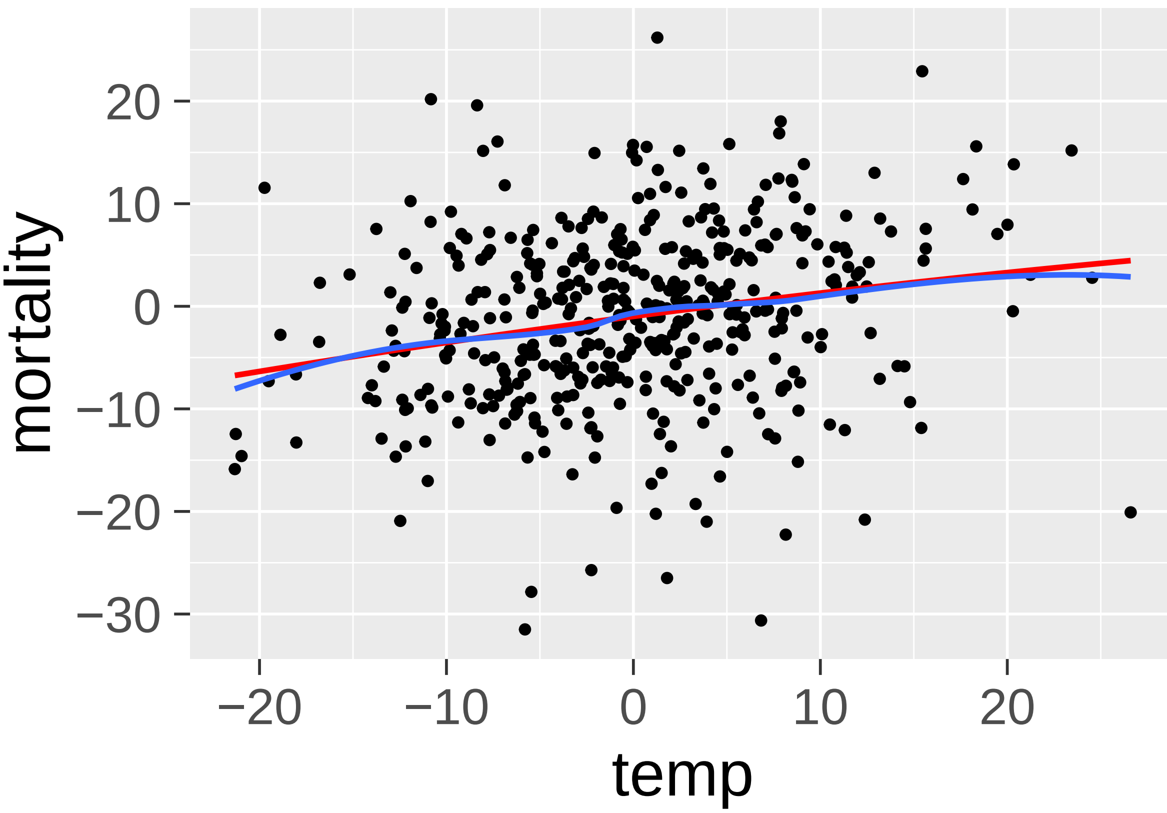

```r
qplot(temp, part, data = mort)
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
```



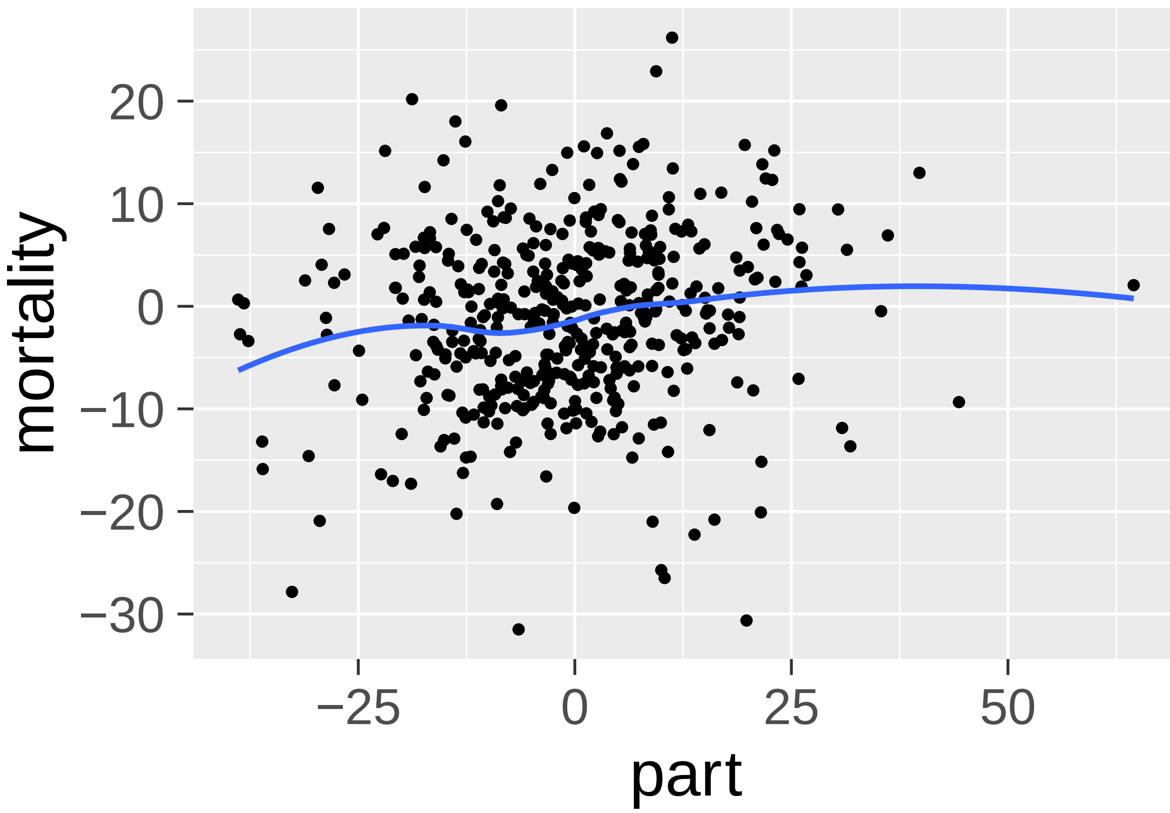

```r
qplot(temp, mortality, data = mort) +
  geom_smooth(method = "lm", formula = y ~ poly(x, 2), se = FALSE, colour = "red") +
  geom_smooth(se = FALSE)  +
  big_font
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
```




```
qplot(part, mortality, data = mort) + geom_smooth(se = FALSE) +
  big_font
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
```

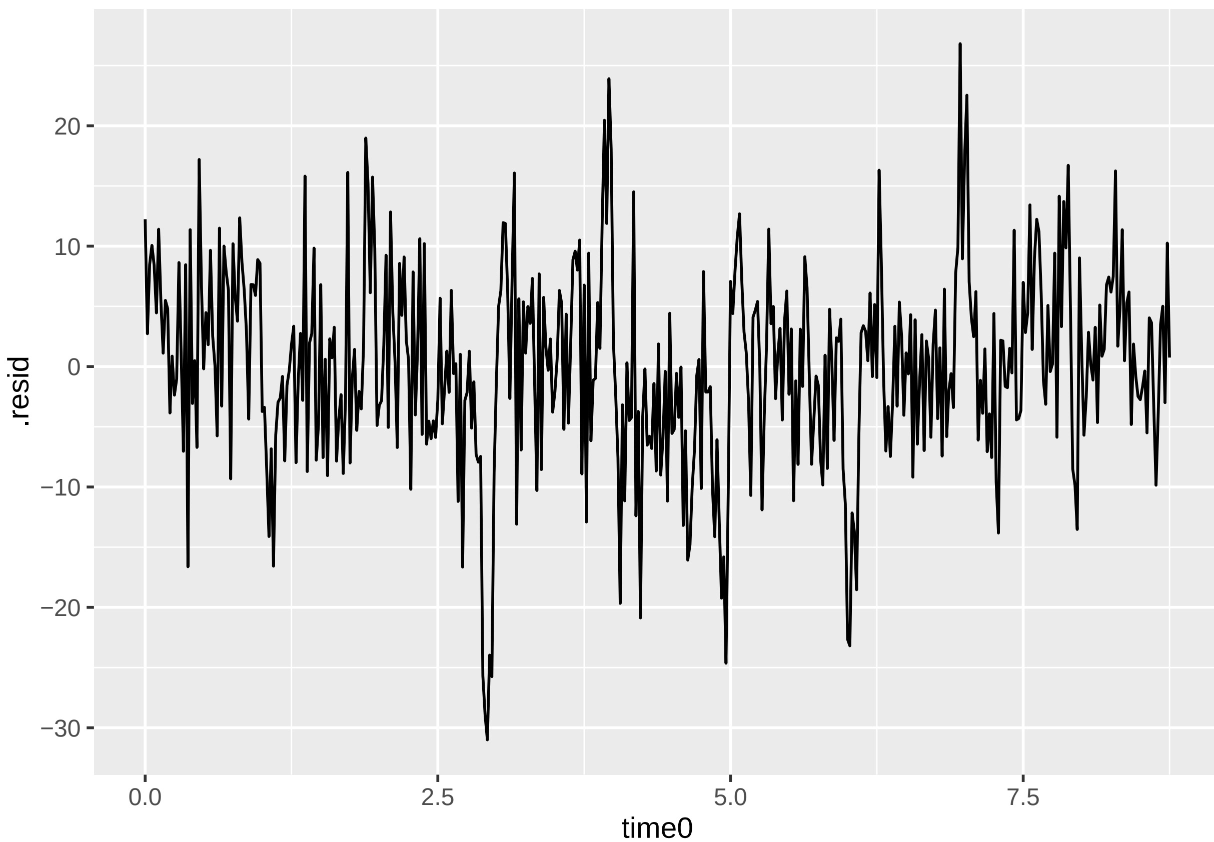



```
mort <- mutate(mort, temp_sc = temp - mean(temp),
               temp_2 = temp_sc^2,
               time0 = time - min(time))

fit_lm <- lm(mortality ~ time0 + temp_sc + temp_2 + part, data = mort, na.action = na.omit)
summary(fit_lm)
```

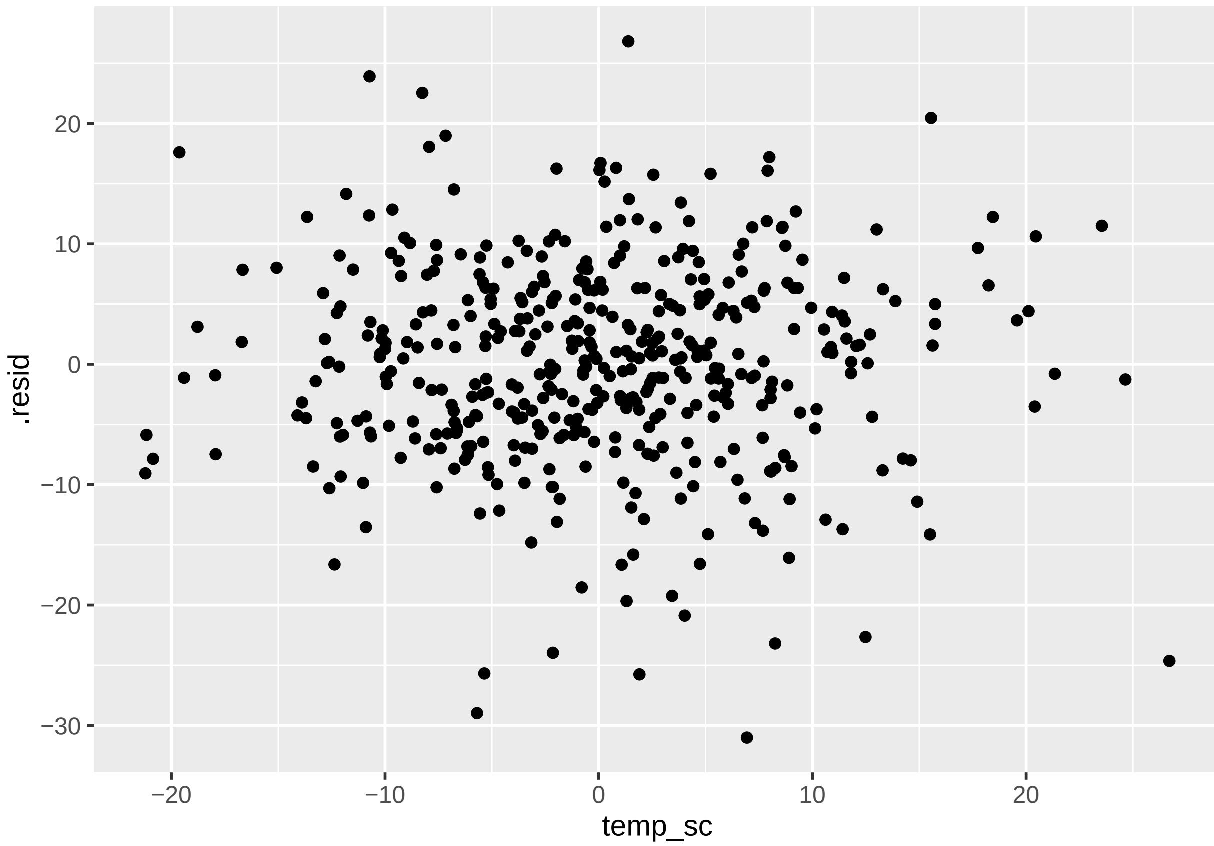
```
##
## Call:
## lm(formula = mortality ~ time0 + temp_sc + temp_2 + part, data = mort,
##     na.action = na.omit)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -31.0049  -4.8348   0.2759   5.3296  26.8181
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.281887   0.836345  -1.533 0.126045
## time0        0.060907   0.157172   0.388 0.698554
## temp_sc      0.252980   0.065382   3.869 0.000125 ***
## temp_2      -0.001407   0.004095  -0.344 0.731281
## part        -0.010496   0.038038  -0.276 0.782725
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 8.306 on 451 degrees of freedom
## Multiple R-squared:  0.05112,    Adjusted R-squared:  0.0427
## F-statistic: 6.074 on 4 and 451 DF,  p-value: 9.107e-05
```

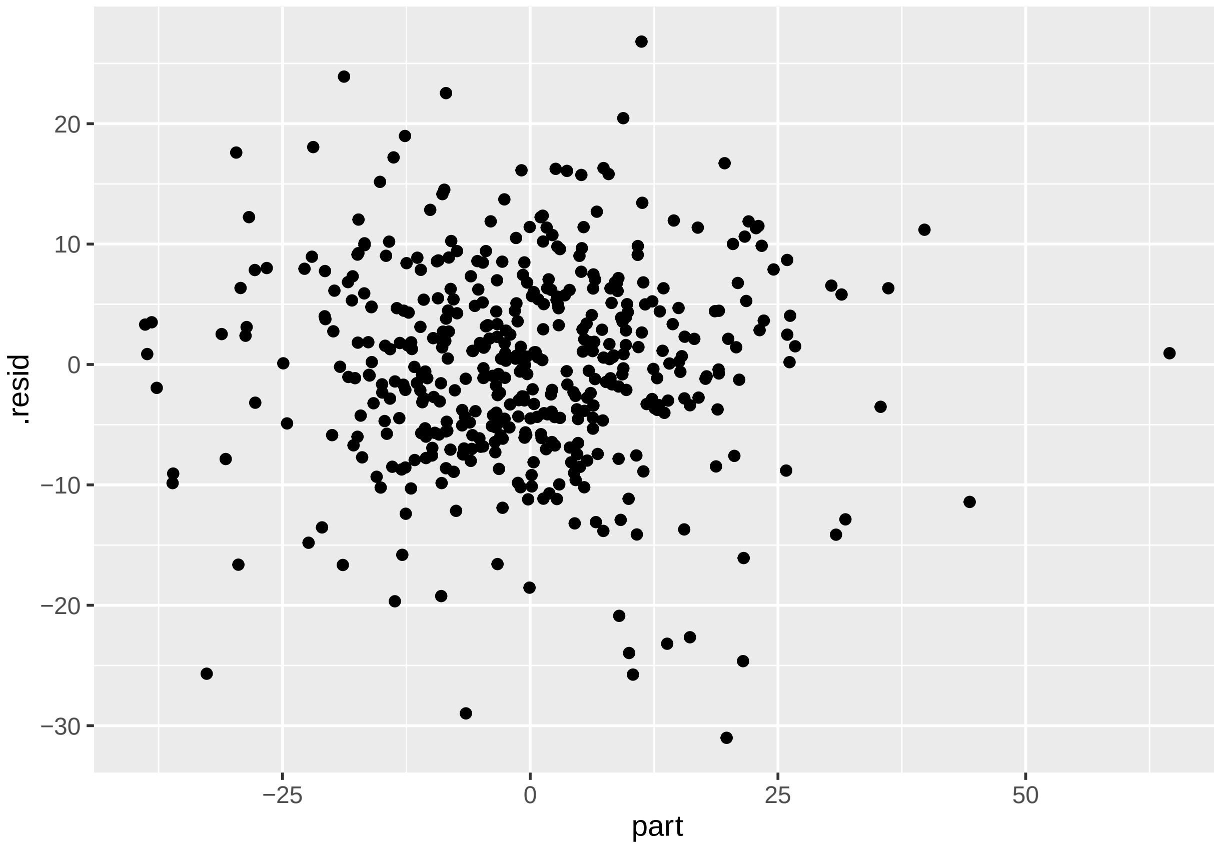
```r
# assumptions
# residuals versus covariates
mort_lm <- fortify(fit_lm)
qplot(time0, .resid, data = mort_lm, geom= "line")
```



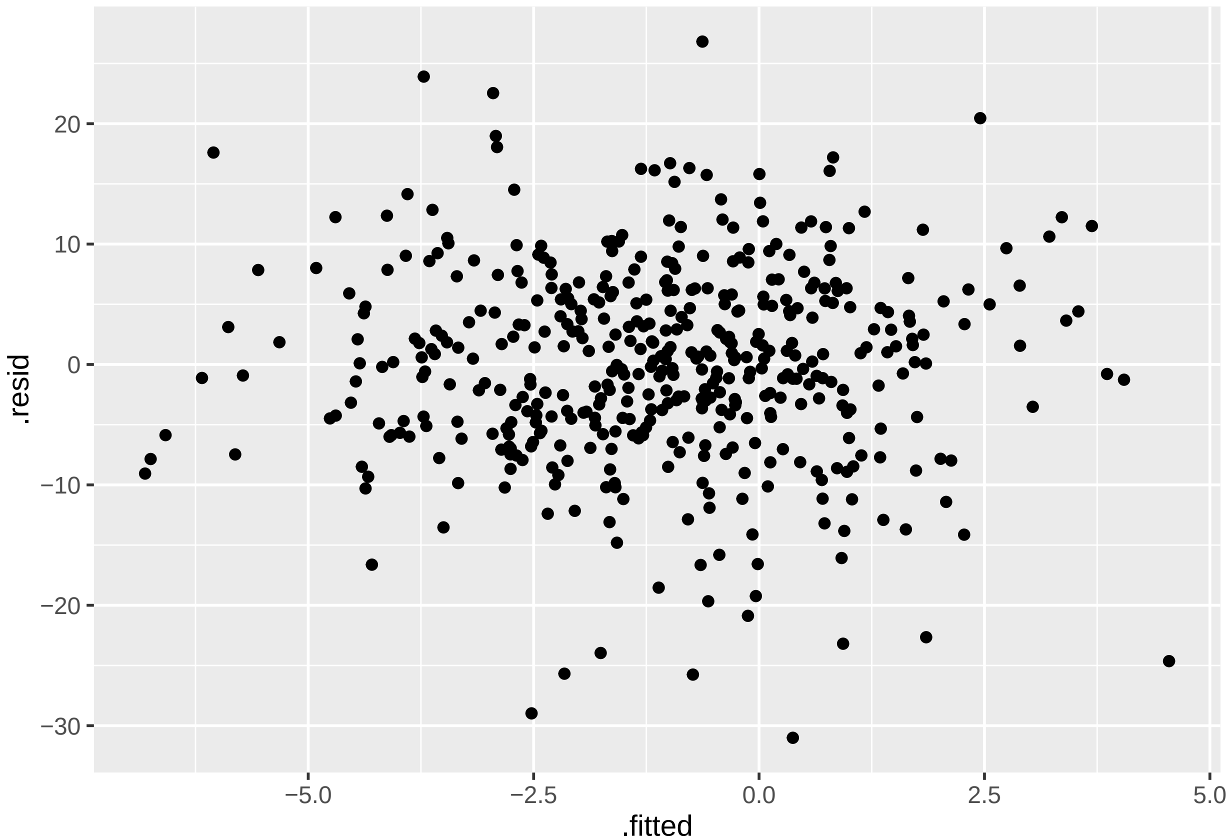

```r
qplot(temp_sc, .resid, data = mort_lm)
```



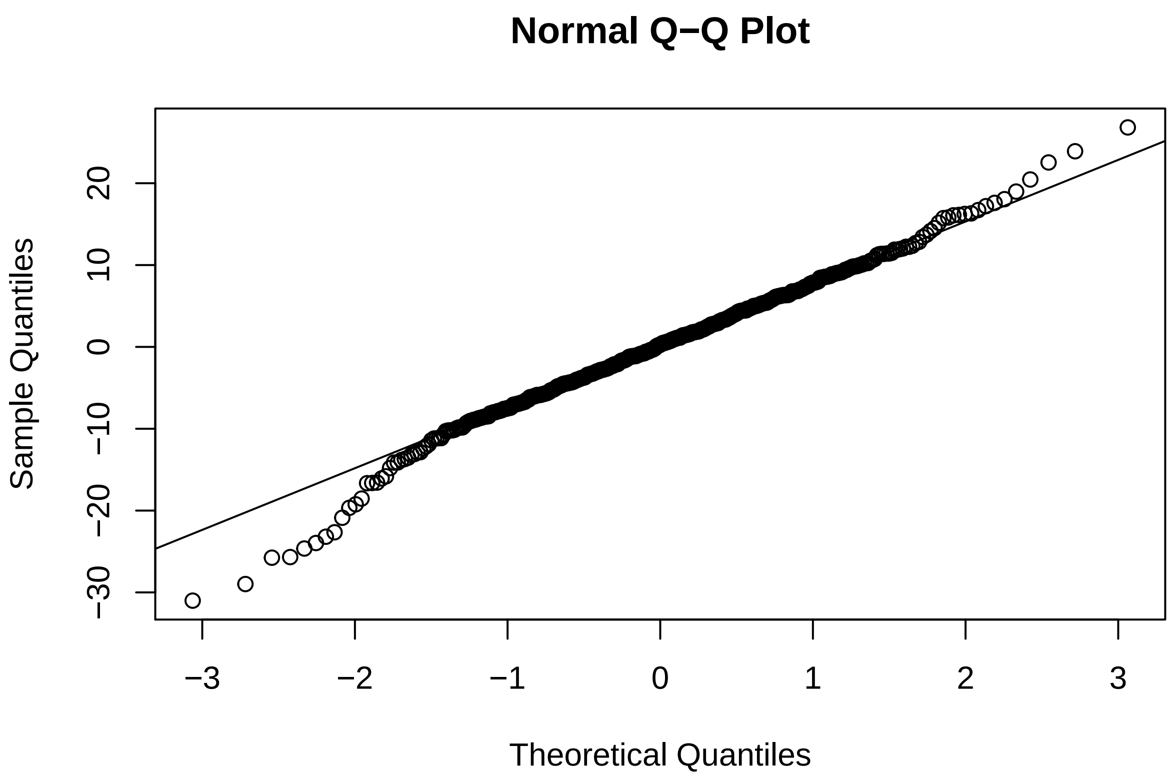

```
qplot(part, .resid, data = mort_lm)
```



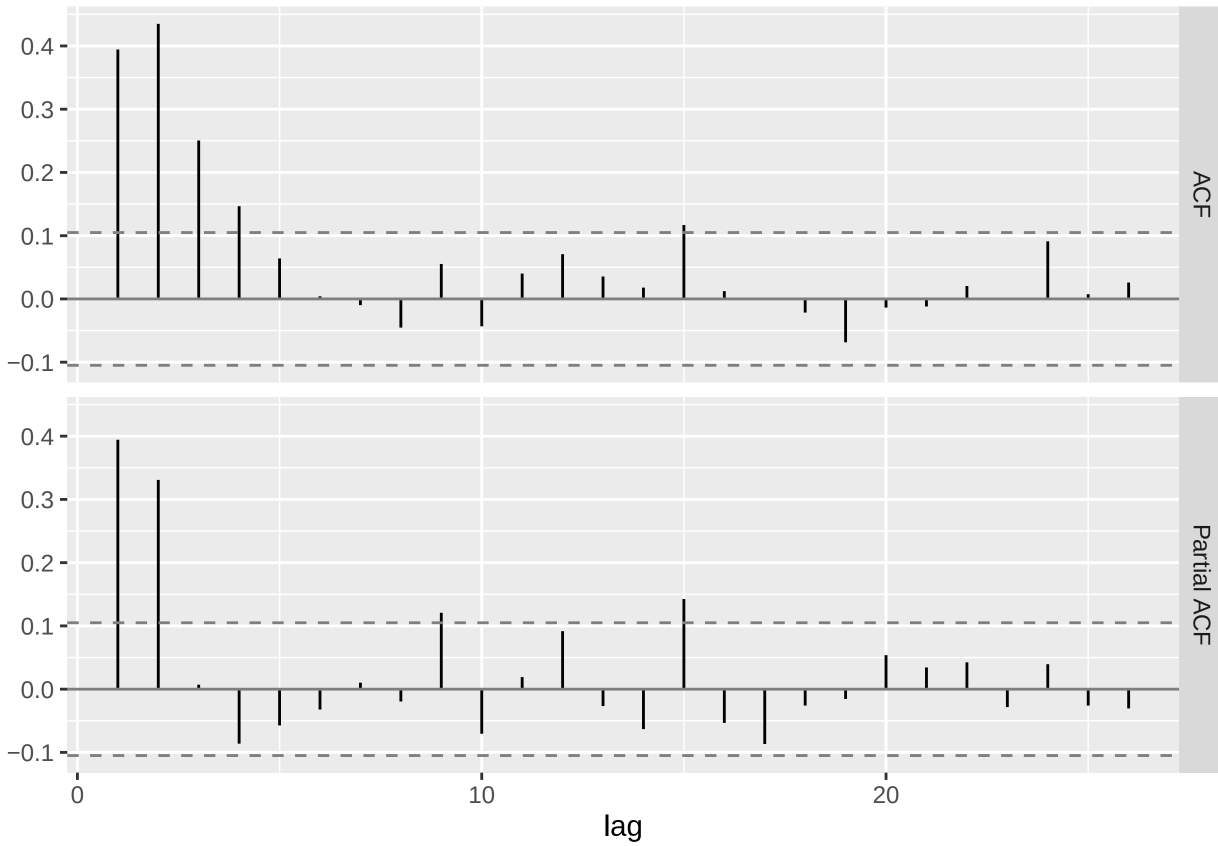

```
# residuals versus fitted
qplot(.fitted, .resid, data = mort_lm)
```



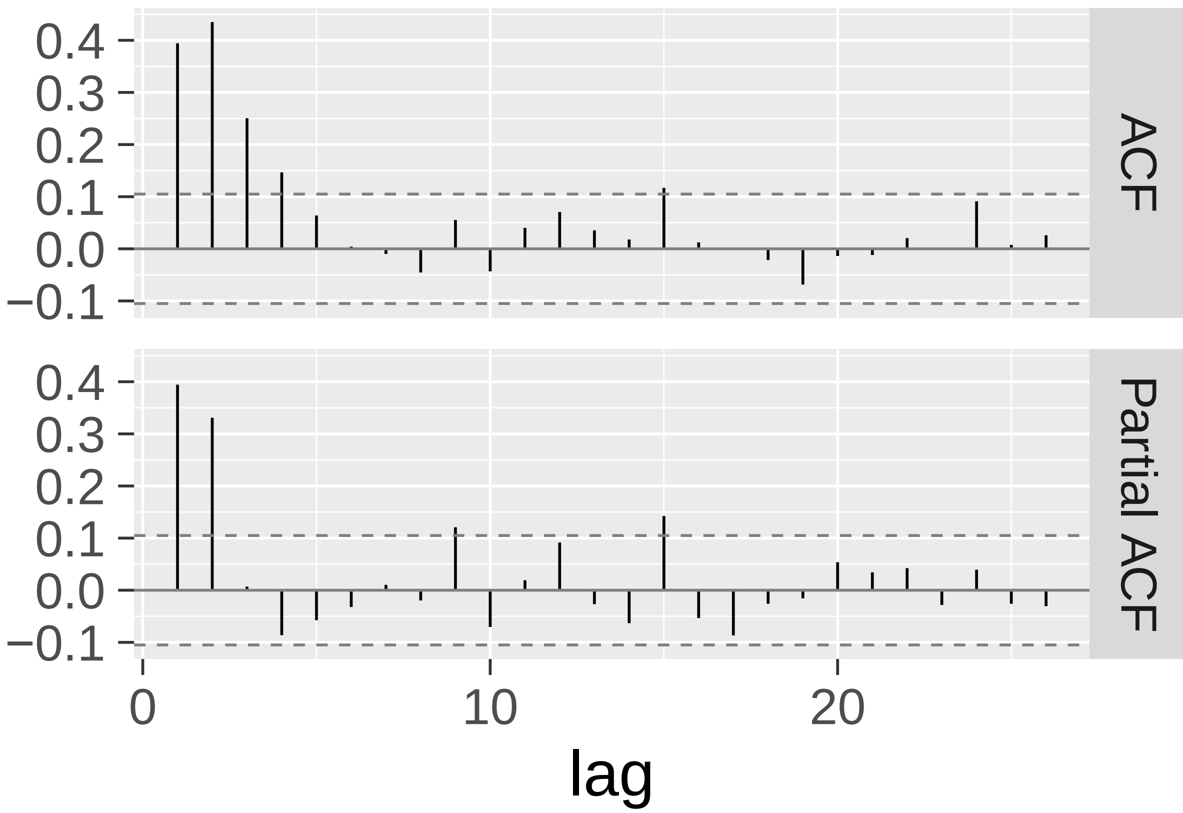

```
# normality of residuals
qqnorm(mort_lm$.resid)
qqline(mort_lm$.resid)
```




```r
# correlation of residuals
examine_corr(residuals(fit_lm))
```




```
last_plot() + big_font
```




```r
# AR (2)? violates regression assumptions

# two ways to fit
library(nlme)
gls_fit <- gls(mortality ~ time0 + temp_sc + temp_2 + part, data = mort,
    correlation = corARMA(p = 15), method = "ML")
summary(gls_fit)
```

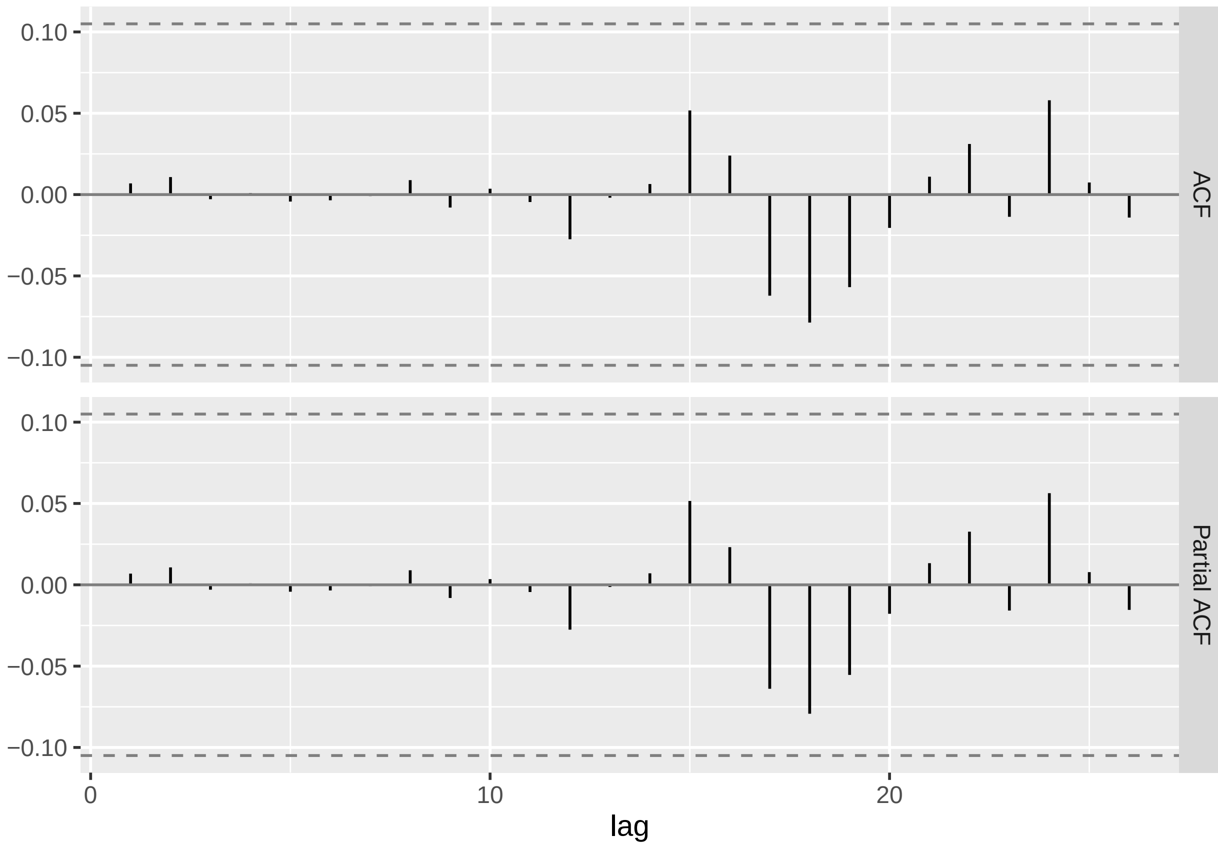
```
## Generalized least squares fit by maximum likelihood
##   Model: mortality ~ time0 + temp_sc + temp_2 + part
##   Data: mort
##        AIC      BIC    logLik
##   3095.665 3182.237 -1526.832
##
## Correlation Structure: ARMA(15,0)
##  Formula: ~1
##  Parameter estimate(s):
##         Phi1         Phi2         Phi3         Phi4         Phi5
##  0.281087676  0.376211288  0.022572796 -0.069685677 -0.009436378
##         Phi6         Phi7         Phi8         Phi9        Phi10
## -0.051876224 -0.036587786 -0.008971095  0.135990895 -0.102843892
##        Phi11        Phi12        Phi13        Phi14        Phi15
##  0.006614401  0.117305504 -0.070173766 -0.097902762  0.144008956
##
## Coefficients:
##                  Value Std.Error   t-value p-value
```

```
## (Intercept) -0.9142608 1.7486759 -0.522830  0.6013
## time0       -0.0203654 0.3421046 -0.059530  0.9526
## temp_sc      0.2300512 0.0501430  4.587903  0.0000
## temp_2       0.0001867 0.0029156  0.064029  0.9490
## part         0.0437174 0.0282271  1.548771  0.1221
##
##  Correlation:
##         (Intr) time0  tmp_sc temp_2
## time0   -0.857
## temp_sc -0.030  0.042
## temp_2  -0.109  0.003 -0.120
## part     0.068 -0.070 -0.683  0.040
##
## Standardized residuals:
##         Min          Q1         Med          Q3         Max
## -3.87947076 -0.60426277  0.01300854  0.65797250  3.19226482
##
## Residual standard error: 8.281303
## Degrees of freedom: 456 total; 451 residual
```

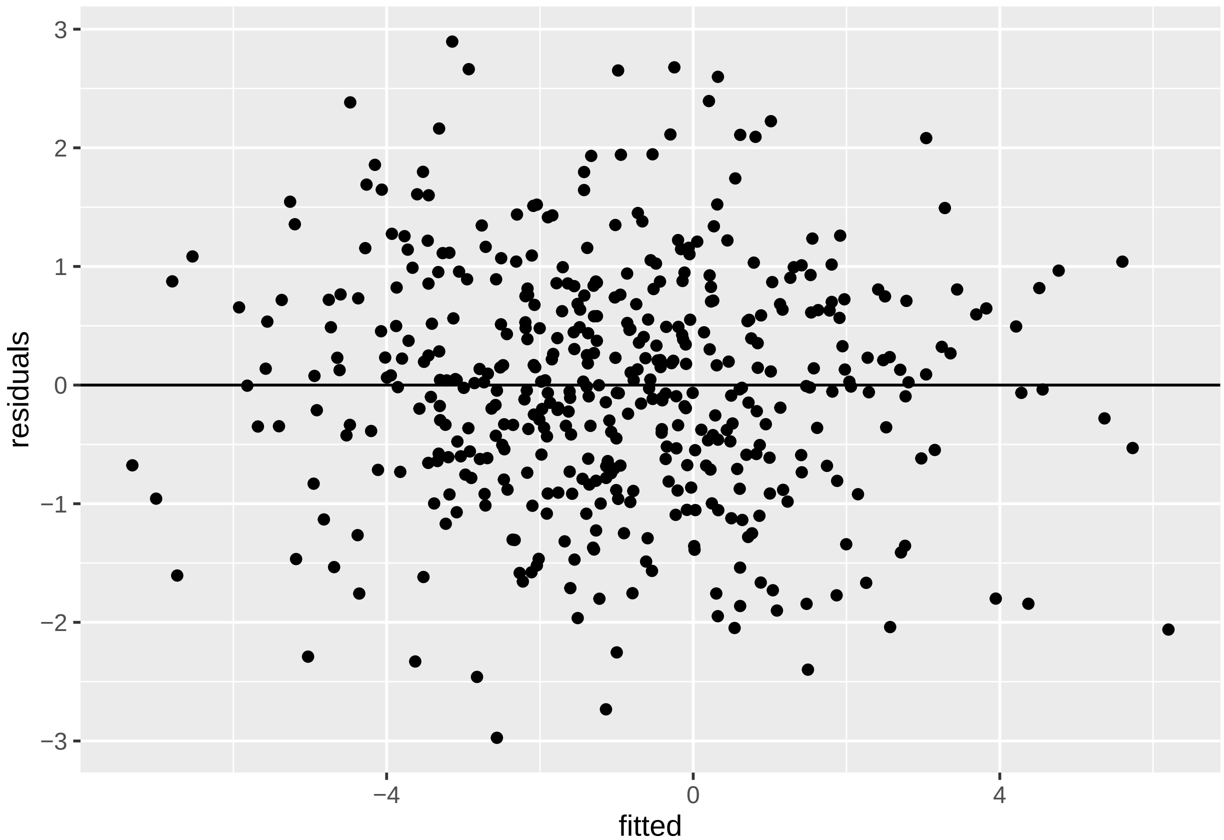
```r
# or
# auto.arima(mort$mortality)

arima_fit <- with(mort,
  arima(mortality, order = c(15, 0, 0), xreg = cbind(time0, temp_sc, temp_2, part)))
arima_fit
```

```
##
## Call:
## arima(x = mortality, order = c(15, 0, 0), xreg = cbind(time0, temp_sc, temp_2,
##     part))
##
## Coefficients:
##          ar1     ar2     ar3      ar4      ar5      ar6      ar7      ar8
##       0.2811  0.3762  0.0225  -0.0696  -0.0095  -0.0519  -0.0366  -0.0090
## s.e.  0.0470  0.0487  0.0514   0.0521   0.0526   0.0507   0.0513   0.0513
##          ar9     ar10    ar11    ar12     ar13     ar14    ar15  intercept
##       0.1360  -0.1029  0.0066  0.1173  -0.0702  -0.0979  0.1440    -0.9144
## s.e.  0.0507   0.0513  0.0516  0.0511   0.0524   0.0487  0.0465     1.7469
##         time0  temp_sc  temp_2    part
##       -0.0213   0.2300   2e-04  0.0437
## s.e.   0.3412   0.0541   3e-03  0.0292
##
## sigma^2 estimated as 47.29:  log likelihood = -1526.83,  aic = 3093.66
```

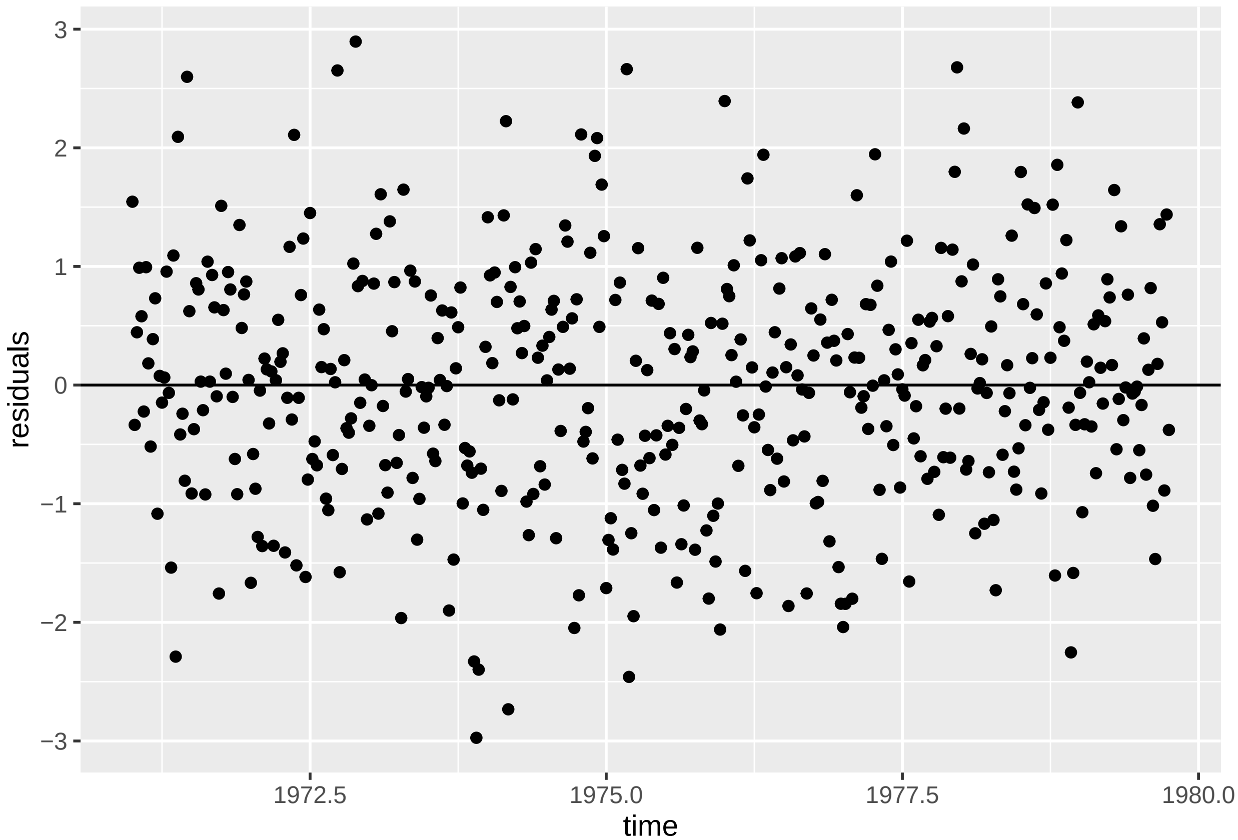
```r
# diagnostics
mort$residuals <- residuals(gls_fit, type = "normalized")
mort$fitted <- fitted(gls_fit)

examine_corr(mort$residuals)
```




```
qplot(fitted, residuals, data = mort) + geom_hline(yintercept = 0)
```

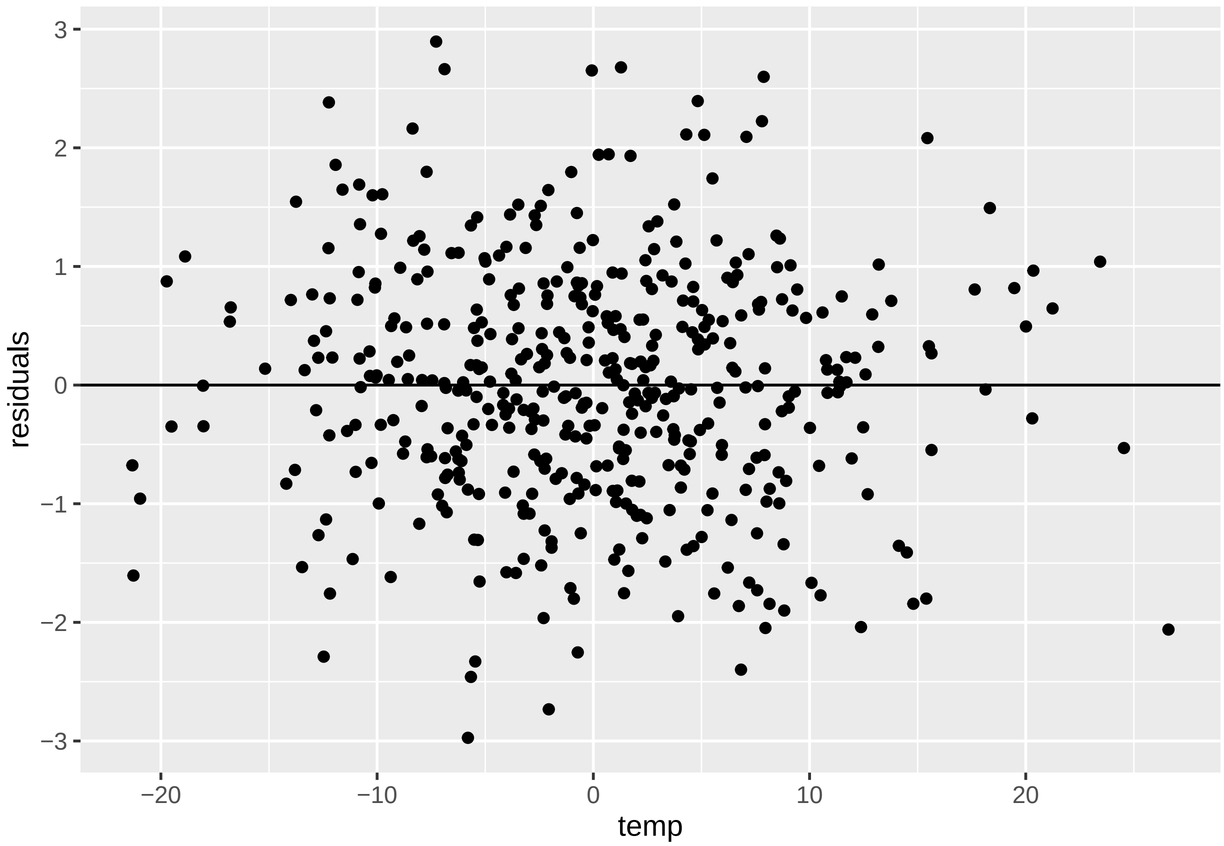



```r
qplot(time, residuals, data = mort) + geom_hline(yintercept = 0)
```



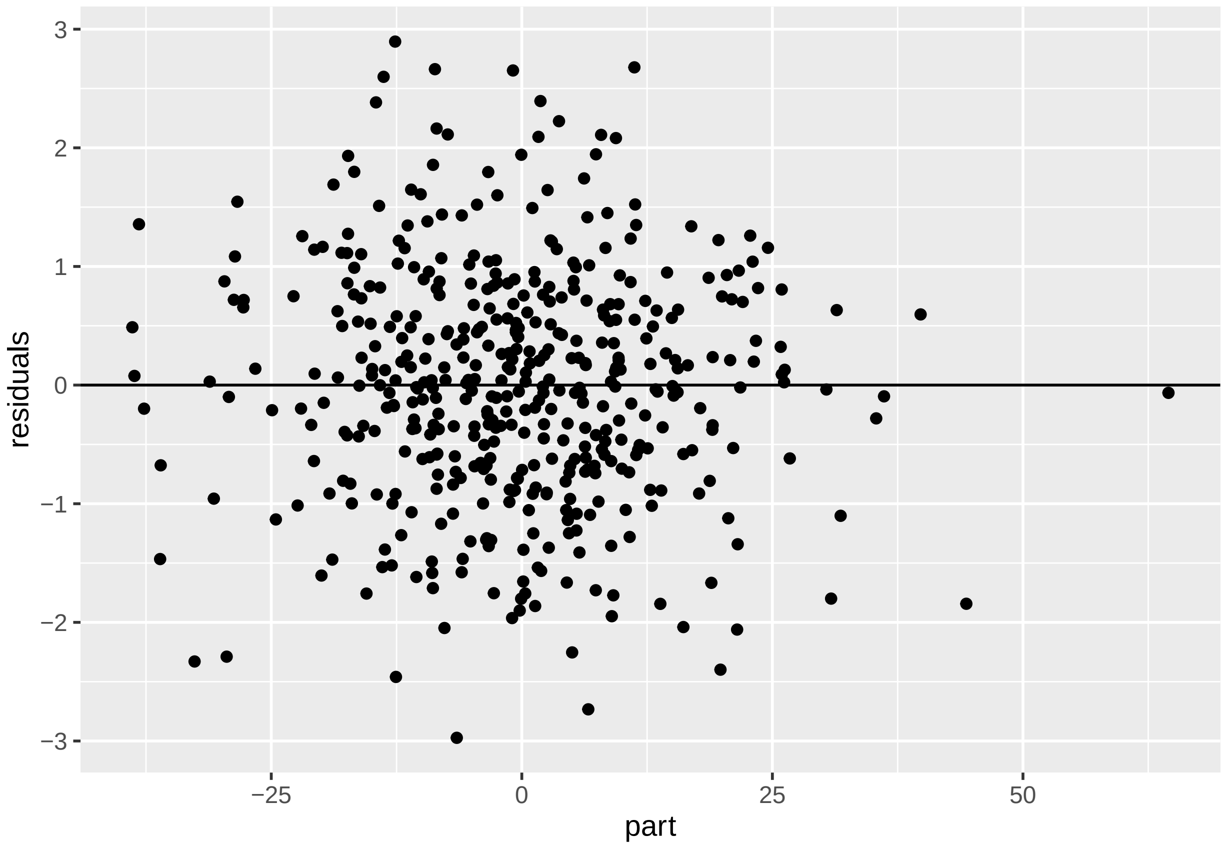

```
qplot(temp, residuals, data = mort) + geom_hline(yintercept = 0)
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
```

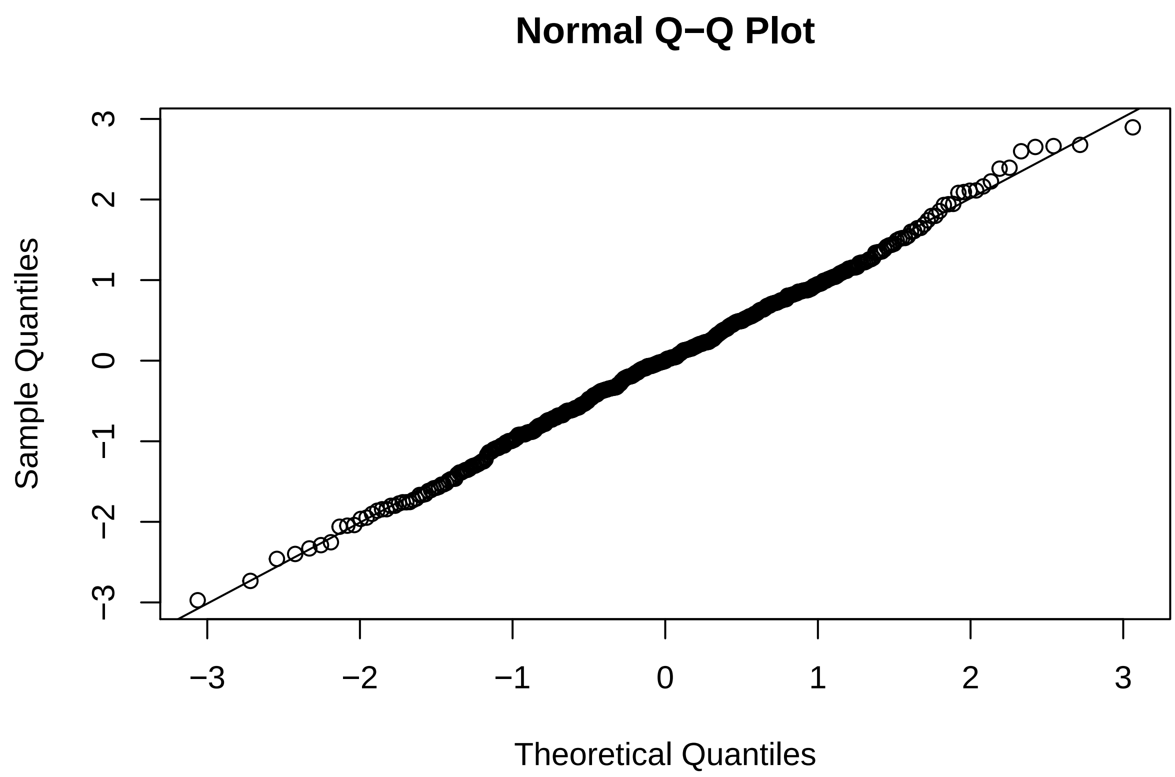



```
qplot(part, residuals, data = mort) + geom_hline(yintercept = 0)
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
```




```r
qqnorm(mort$residuals)
qqline(mort$residuals)
```




```r
confint(arima_fit)
```

```
##                 2.5 %       97.5 %
## ar1         0.18887839  0.373302306
## ar2         0.28067323  0.471767161
## ar3        -0.07819816  0.123256413
## ar4        -0.17179392  0.032500759
## ar5        -0.11245853  0.093541568
## ar6        -0.15128301  0.047550479
## ar7        -0.13722378  0.063954810
## ar8        -0.10943930  0.091526134
## ar9         0.03666492  0.235335205
## ar10       -0.20337533 -0.002367212
## ar11       -0.09447303  0.107653745
## ar12        0.01720230  0.217465367
## ar13       -0.17296648  0.032542975
## ar14       -0.19331659 -0.002469489
## ar15        0.05277746  0.235243173
## intercept -4.33830438  2.509522744
## time0      -0.69001009  0.647365473
## temp_sc     0.12391961  0.336159556
## temp_2     -0.00566624  0.006041411
## part       -0.01360277  0.101041880
```

```r
confint(fit_lm)
```

```
##                     2.5 %      97.5 %
```

```
## (Intercept) -2.925503168 0.361729983
## time0       -0.247972794 0.369787229
## temp_sc      0.124489216 0.381471429
## temp_2      -0.009455327 0.006640811
## part        -0.085249027 0.064257196
```

```r
intervals(gls_fit)
```

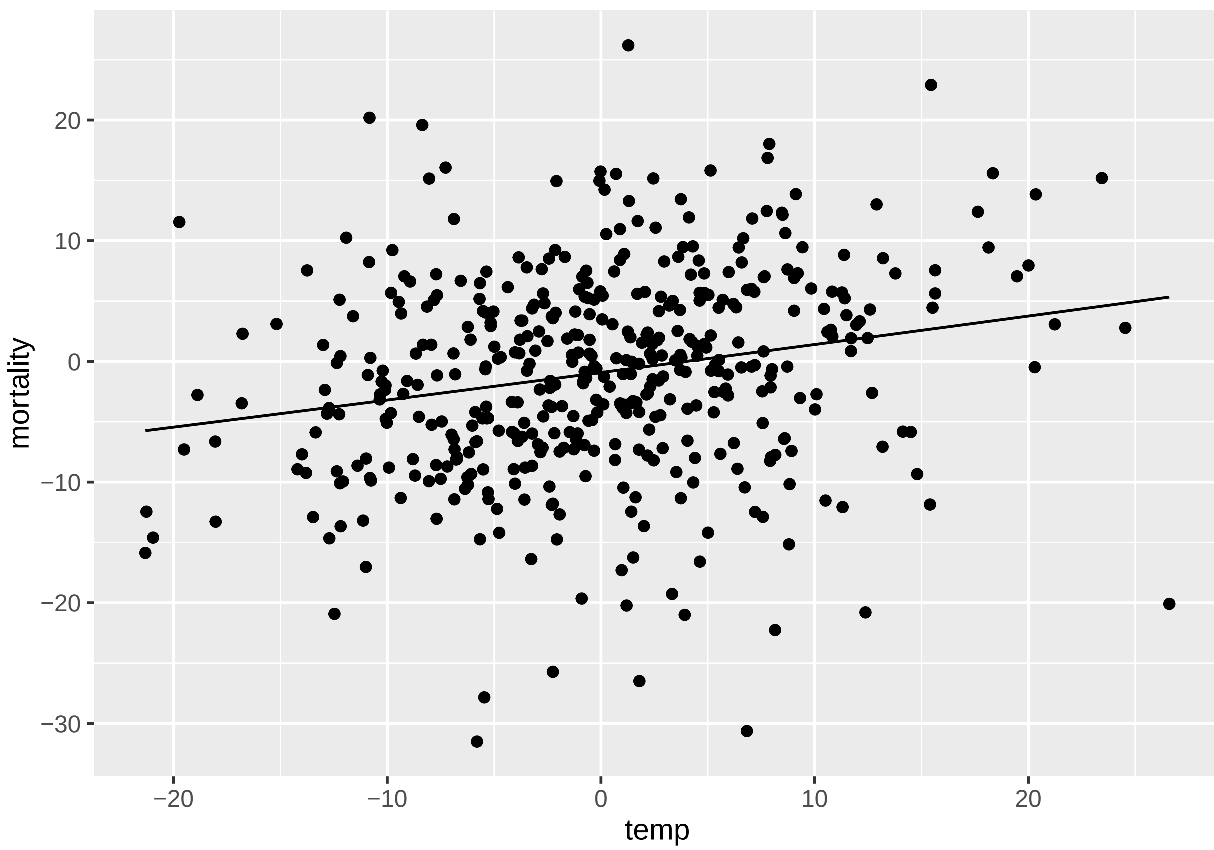
```
## Approximate 95% confidence intervals
##
##  Coefficients:
##                    lower          est.       upper
## (Intercept) -4.350825057 -0.9142608019 2.522303453
## time0       -0.692682374 -0.0203654481 0.651951478
## temp_sc      0.131508264  0.2300511561 0.328594048
## temp_2      -0.005543178  0.0001866847 0.005916548
## part        -0.011755686  0.0437173682 0.099190422
## attr(,"label")
## [1] "Coefficients:"
##
##  Correlation structure:
##              lower         est.        upper
## Phi1   0.13296541  0.281087676  0.166031008
## Phi2   0.25778977  0.376211288  0.310109711
## Phi3  -0.06733034  0.022572796  0.019586118
## Phi4  -0.16190861 -0.069685677 -0.057467380
## Phi5  -0.10097754 -0.009436378 -0.004552784
## Phi6  -0.13706313 -0.051876224 -0.039901390
## Phi7  -0.12968346 -0.036587786 -0.008844102
## Phi8  -0.09685403 -0.008971095  0.026611710
## Phi9   0.04799215  0.135990895  0.162242420
## Phi10 -0.17031658 -0.102843892 -0.052828465
## Phi11 -0.05241476  0.006614401  0.050032658
## Phi12  0.05774212  0.117305504  0.148565429
## Phi13 -0.11844013 -0.070173766 -0.018916003
## Phi14 -0.15913453 -0.097902762 -0.004405277
## Phi15  0.05133356  0.144008956  0.234225287
## attr(,"label")
## [1] "Correlation structure:"
##
##  Residual standard error:
##    lower     est.    upper
## 7.546556 8.281303 9.087585
```

```r
# if you refit the model with white noise errors using gls, you can actually
# test to see if the correlation structure improved the fit
gls_wn <- gls(mortality ~ time0 + temp_sc + temp_2 + part, data = mort, method = "ML")
anova(gls_wn, gls_fit)
```

```
##         Model df      AIC      BIC    logLik   Test  L.Ratio p-value
## gls_wn      1  6 3231.778 3256.513 -1609.889
## gls_fit     2 21 3095.665 3182.237 -1526.832 1 vs 2 166.1131  <.0001
```

```
# plot prediction on temp versus mortality plot
mort$pred <- predict(gls_fit, newdata =
  data.frame(time0 = 0, temp_sc = mort$temp_sc,
             temp_2 = mort$temp_2, part = mean(mort$part)))

qplot(temp, mortality, data = mort) +
  geom_line(aes(y = pred))
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous
```



My analysis shows that removing the seasonality by differencing with lag 52, results in a better-fitting model, in comparison with when seasonality is present in data and is not dealt with properly. This is because when there exists a clear seasonal trend in the data, the data is non-stationary and we shoud remove non-stationarity for our analysis and model to be meaningful. In our case, the annual changes in temperature and other seasonal trends are a result of one confounding factor "the time of the year". Differencing gets rid of this, and gives us a clean stationary time series. Our model shows different significant explanatory variables, and the AIC for our model is better than Charlotte's provided code without removing seasonality.