SpaceAndTime

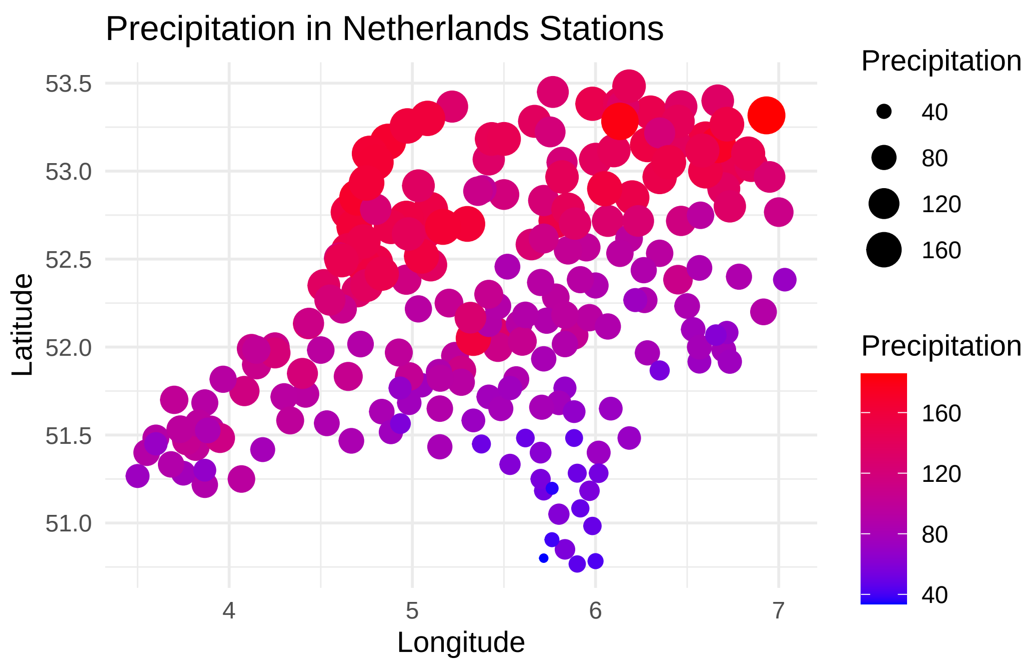
Table of contents

TODO check if I checked all the residuals for all models

Question 1 Spatial modelling Kingdom of the Netherlands

1 a)

```
ggplot(data = netherlandsDF) + geom_point(aes(x = longitude, y = latitude,
    size = precip, color = precip)) + scale_color_continuous(low = "blue",
    high = "red") + labs(title = "Precipitation in Netherlands Stations",
    x = "Longitude", y = "Latitude", size = "Precipitation", color = "Precipitation") +
    theme_minimal()
```



From what we can see from the data it does seem to be spatially correlated as we can that the Dutch provinces of north Holland, Friesland and Groningen has higher precipitation and as we go south the precipitation does decrease as we can see from the Dutch provinces of Zeeland, north Brabant and Limburg where precipitation is significantly lower than their northern counterparts.

From this data, latitude seems to be the biggest factor in the variation of the precipitation as the longitude only suggests some slight variations in the data.

```
# Create geodata object
precipitationNetherland_geoR = as.geodata(netherlandsDF, coords.col = c("longitude",
    "latitude"), data.col = "precip")

summary(precipitationNetherland_geoR)
```

```
Number of data points: 220

Coordinates summary
    longitude latitude
min     3.500   50.767
max     7.033   53.483
```

```
Distance summary
     min      max
0.001000 3.998498

Data summary
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
 33.9000  80.8500 100.1500 106.5705 137.3500 185.6000
```

As we can see from the numerical summary of the data the median is different from the mean, which indicates it is not a symmetric distribution of data points and is instead positively skewed since the mean is bigger than the median. As such there are more values on the left side of the distribution.

1 b)

```r
# set seed for reproducibility
set.seed(26041999)

# Select 3 random rows from the data frame
randomRowsPrecipitation = netherlandsDF %>%
    sample_n(3)

# Add a new column with labels
randomRowsPrecipitation$label = c("A", "B", "C")

# Print the randomly selected rows
randomRowsPrecipitation
```

```
# A tibble: 3 x 5
  station_name   longitude latitude precip label
  <chr>              <dbl>    <dbl>  <dbl> <chr>
1 NIJKERK             5.47     52.2   89.1 A
2 WOLPHAARTSDIJK      3.73     51.5   95.9 B
3 EEXT                6.73     53    147.  C
```

```r
# Remove the selected rows from the original dataset
netherlandsDF_filtered = netherlandsDF %>%
    anti_join(randomRowsPrecipitation)
```

```
Joining, by = c("station_name", "longitude", "latitude", "precip")
```

```
# Print the resulting dataframe
netherlandsDF_filtered
```

```
# A tibble: 217 x 4
   station_name      longitude latitude precip
   <chr>                 <dbl>    <dbl>  <dbl>
 1 WEST TERSCHELLING      5.22     53.4  130.
 2 GRONINGEN-1            6.6      53.2  157.
 3 HOORN                  5.07     52.6  146.
 4 HOOFDDORP              4.7      52.3  130.
 5 WINTERSWIJK            6.7      52.0   77.7
 6 KERKWERVE              3.87     51.7   91.8
 7 WESTDORPE-1            3.87     51.2   87.7
 8 OUDENBOSCH             4.53     51.6   84.2
 9 ROERMOND               5.97     51.2   56.4
10 PETTEN                 4.65     52.8  158.
# ... with 207 more rows
```

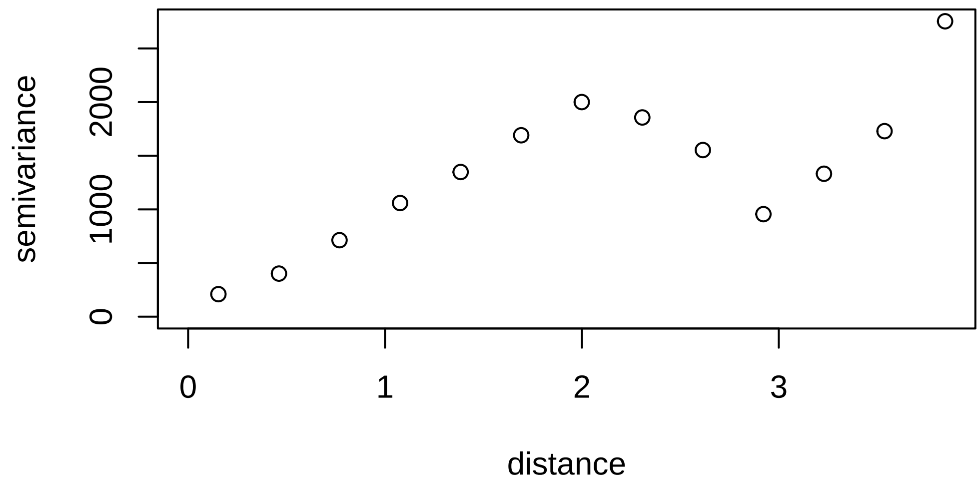
```
## recreate the geoData object with the new filtered dataframe
precipitationNetherland_geoR = as.geodata(netherlandsDF_filtered, coords.col = c("longitud
    "latitude"), data.col = "precip")
```

1 c)

```
# # Calculate empirical variogram
variogramPrecipitationNetherlands = variog(precipitationNetherland_geoR)
```

```
variog: computing omnidirectional variogram
```

```
# Plot empirical variogram
plot(variogramPrecipitationNetherlands)
```




```
variogramPrecipitationNetherlands$n
```

```
 [1] 1069 2482 3384 3834 3647 3221 2534 1541  787  468  318  133   17
```

From the plotted variogram we can see there a very clear need for a nugget as there is a non-zero value around zero distance, this values seems to be around 75 to 100 at the zero distance from how much is it decreasing.

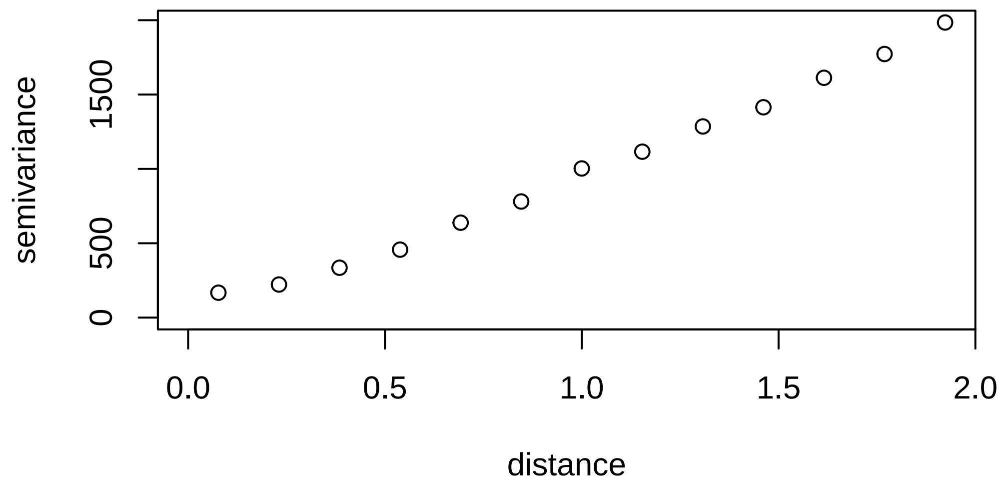
The semi variance continuous to increase with distance till around the distance of 2 degrees distance wise, after this there is a decrease in variance that is not representative of the data as we are more and more uncertain the further we are from our known points, as such we will choose the distance of two as the cut off for the maximum distance.

we know change the maximum distance change and recut our previous variogram.

```
variogramPrecipitationNetherlands = variog(precipitationNetherland_geoR,
    option = "bin", max.dist = 2)
```

```
variog: computing omnidirectional variogram
```

```r
plot(variogramPrecipitationNetherlands)
```



As we can see from the newly updated variogram the increase is almost linear with a curve near 0 where we can see the need for the nugget.

1 d)

Now that we have the variogram we will start by fitting a model to estimate the covariance via weighted least squares. Fitting this variogram we get the estimated values of σ^2, ϕ and τ^2 also known as the nugget

We will first start with the default Matrén = 0,5 which is equivalent to an exponential as the form observed in the previous variogram seems to not fully linear and therefore require the curvature from a function like the exponential function to account for the behaviour at the near 0 distance.

From this we will try different models to search for the model with the best fit.

```r
# ?variofit thau = nugget variability sigmasq = if the model can
# capture more or less of the total variability phi = if the
# correlation extends over a bigger or smaller distance loss value =
```

```r
# goodness of fit (smaller means better fit)

krigingVariogramFittedDefault = variofit(variogramPrecipitationNetherlands,
    nugget = 85)
```

```
variofit: covariance model used is matern
variofit: weights used: npairs
variofit: minimisation function used: optim
```

```
Warning in variofit(variogramPrecipitationNetherlands, nugget = 85): initial
values not provided - running the default search
```

```
variofit: searching for best initial value ... selected values:
              sigmasq   phi    tausq kappa
initial.value "1984.67" "1.54" "85"  "0.5"
status        "est"     "est"  "est" "fix"
loss value: 818327620.928191
```

```r
krigingVariogramFittedDefault
```

```
variofit: model parameters estimated by WLS (weighted least squares):
covariance model is: matern with fixed kappa = 0.5 (exponential)
parameter estimates:
     tausq    sigmasq        phi
      0.00 2561207.09    2590.45
Practical Range with cor=0.05 for asymptotic range: 7760.294

variofit: minimised weighted sum of squares = 35360382
```

Now we will increase the kappa of the Matrén to see if the increased flexibility and smoothness leads to a better fit

```r
krigingVariogramFittedMatrén1.5 = variofit(variogramPrecipitationNetherlands,
    kappa = 1.5, nugget = 85)
```

```
variofit: covariance model used is matern
variofit: weights used: npairs
variofit: minimisation function used: optim
```

```
Warning in variofit(variogramPrecipitationNetherlands, kappa = 1.5, nugget =
85): initial values not provided - running the default search
```

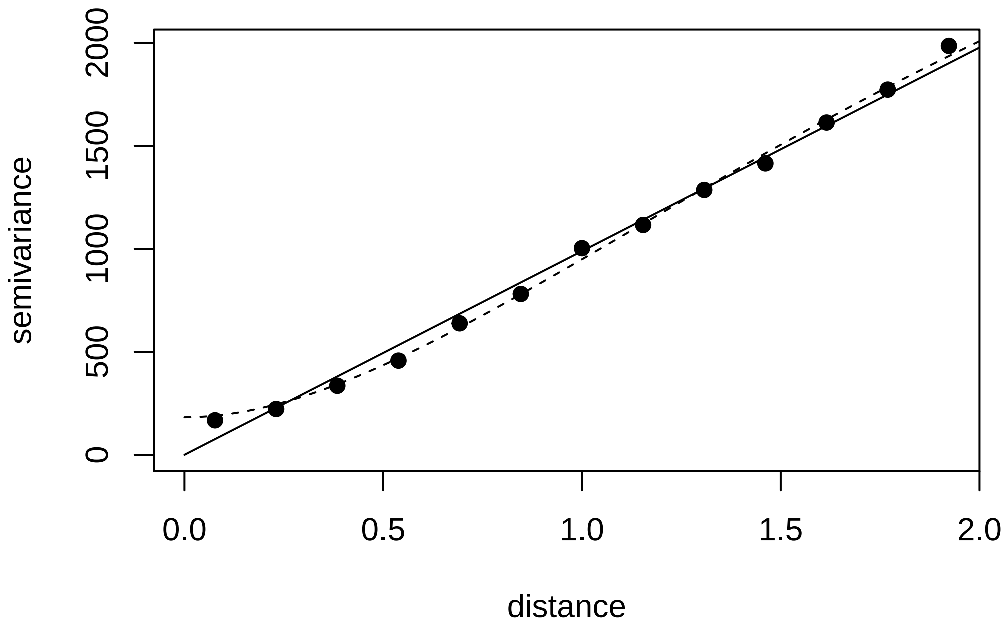
```
variofit: searching for best initial value ... selected values:
              sigmasq   phi    tausq kappa
initial.value "1984.67" "0.62" "85"  "1.5"
status        "est"     "est"  "est" "fix"
loss value: 190482114.063677
```

```
krigingVariogramFittedMatrén1.5
```

```
variofit: model parameters estimated by WLS (weighted least squares):
covariance model is: matern with fixed kappa = 1.5
parameter estimates:
    tausq   sigmasq       phi
 182.0638 3461.1492    1.1316
Practical Range with cor=0.05 for asymptotic range: 5.368367

variofit: minimised weighted sum of squares = 15435206
```

We will first visually compare these 2 models to see which one has a better

```
par(mar = c(4, 4, 2, 2))
plot(variogramPrecipitationNetherlands, pch = 19)
lines(krigingVariogramFittedDefault)
lines(krigingVariogramFittedMatrén1.5, lty = 2)
```




```
# lines(krigingVariogramFittedMatrén2.5, lty = 3)
```

Immediately we can see that the extra flexibility of the Matrén 1,5 not only better follows the actual data, it actually accounts correctly for the initial variance from the nugget which the Matrén 0,5 does not as it simply decreases to 0.

We now will test if any additional flexibility changes can improve the model fit

We will first try to again increase the kappa to see if the model again benefits from the extra flexibility

```
krigingVariogramFittedMatrén2.0 = variofit(variogramPrecipitationNetherlands,
    kappa = 2, nugget = 85)
```

```
variofit: covariance model used is matern
variofit: weights used: npairs
variofit: minimisation function used: optim
```

```
Warning in variofit(variogramPrecipitationNetherlands, kappa = 2, nugget = 85):
initial values not provided - running the default search
```

```
variofit: searching for best initial value ... selected values:
              sigmasq   phi    tausq    kappa
initial.value "1984.67" "0.62" "198.47" "2"
status        "est"     "est"  "est"    "fix"
loss value: 227233080.389154
```

```r
krigingVariogramFittedMatrén2.0
```

```
variofit: model parameters estimated by WLS (weighted least squares):
covariance model is: matern with fixed kappa = 2
parameter estimates:
    tausq   sigmasq       phi
 197.6127 3037.3813    0.8386
Practical Range with cor=0.05 for asymptotic range: 4.502076

variofit: minimised weighted sum of squares = 18197964
```

Here we will instead see if the model will benefit instead form a cut of flexibility to make it less smooth

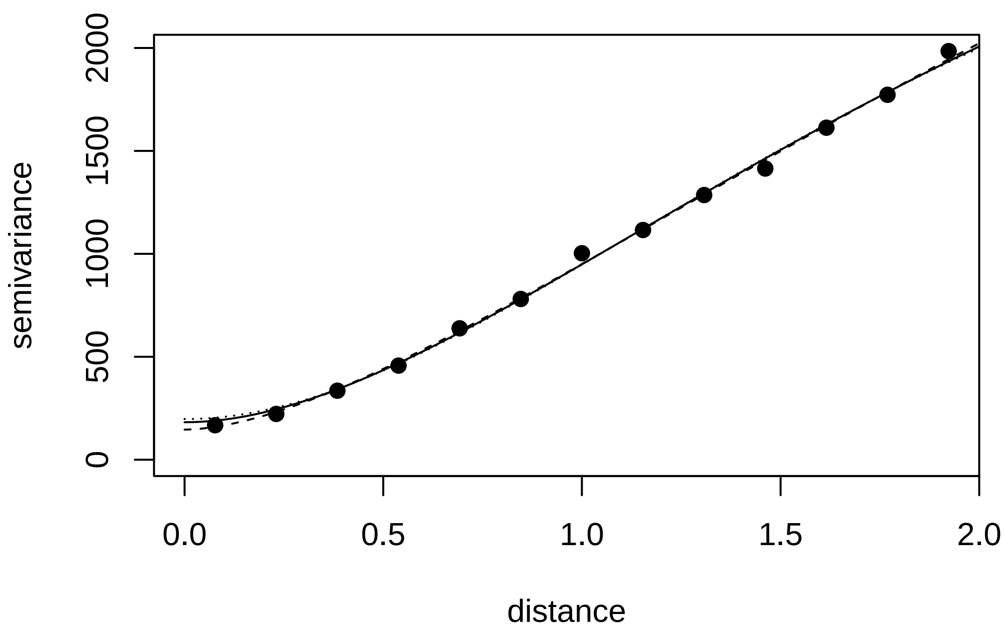
```r
krigingVariogramFittedMatrén1.0 = variofit(variogramPrecipitationNetherlands,
    kappa = 1, nugget = 85)
```

```
variofit: covariance model used is matern
variofit: weights used: npairs
variofit: minimisation function used: optim

Warning in variofit(variogramPrecipitationNetherlands, kappa = 1, nugget = 85):
initial values not provided - running the default search

variofit: searching for best initial value ... selected values:
              sigmasq   phi    tausq    kappa
initial.value "1984.67" "0.92" "198.47" "1"
status        "est"     "est"  "est"    "fix"
loss value: 352001017.756763
```

```r
krigingVariogramFittedMatrén1.0
```

```
variofit: model parameters estimated by WLS (weighted least squares):
covariance model is: matern with fixed kappa = 1
parameter estimates:
    tausq   sigmasq       phi
 146.1354 4832.1862    2.0470
Practical Range with cor=0.05 for asymptotic range: 8.185098

variofit: minimised weighted sum of squares = 12019899
```

```r
par(mar = c(4, 4, 2, 2))
plot(variogramPrecipitationNetherlands, pch = 19)
lines(krigingVariogramFittedMatrén1.5)
lines(krigingVariogramFittedMatrén1.0, lty = 2)
lines(krigingVariogramFittedMatrén2.0, lty = 3)
```



As we can see from the new graph it does seem that actually a lower flexibility Matrén has a better fit since the extra flexibility near the start and end of the data points made the models deviate too much from the points.

1 e)

To fit a model using the maximum likelihood we will have to try multiple initial values to make sure this is indeed the maximum likelihood and not just a local maximum.

```
# ?likest

maximumLikelihoodNetherlandsInitial10.1 = likfit(precipitationNetherland_geoR,
    ini.cov.pars = c(10, 1))
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```
maximumLikelihoodNetherlandsInitial1.10 = likfit(precipitationNetherland_geoR,
    ini.cov.pars = c(1, 10))
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```
maximumLikelihoodNetherlandsInitial100.10 = likfit(precipitationNetherland_geoR,
    ini.cov.pars = c(100, 10))
```

```
---------------------------------------------------------------
```

```
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```r
maximumLikelihoodNetherlandsInitial10.100 = likfit(precipitationNetherland_geoR,
    ini.cov.pars = c(10, 100))
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.

WARNING: estimated range is more than 10 times bigger than the biggest distance between two p
 1) excluding spatial dependence if estimated sill is too low and/or
 2) taking trends (covariates) into account
```

```r
maximumLikelihoodNetherlandsInitial1.1 = likfit(precipitationNetherland_geoR,
    ini.cov.pars = c(1, 1))
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```r
maximumLikelihoodNetherlandsInitial1000.1000 = likfit(precipitationNetherland_geoR,
    ini.cov.pars = c(1000, 1000))
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.

WARNING: estimated range is more than 10 times bigger than the biggest distance between two p
 1) excluding spatial dependence if estimated sill is too low and/or
 2) taking trends (covariates) into account
```

```r
maximumLikelihoodNetherlandsInitial500.500 = likfit(precipitationNetherland_geoR,
    ini.cov.pars = c(500, 500))
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.

WARNING: estimated range is more than 10 times bigger than the biggest distance between two p
 1) excluding spatial dependence if estimated sill is too low and/or
 2) taking trends (covariates) into account
```

```r
maximumLikelihoodNetherlandsInitial10.1
```

```
likfit: estimated model parameters:
```

```
      beta      tausq    sigmasq        phi 
" 102.376" " 114.249" "3036.627" "   7.132" 
Practical Range with cor=0.05 for asymptotic range: 21.36583

likfit: maximised log-likelihood = -896.9
```

```r
maximumLikelihoodNetherlandsInitial1.10
```

```
likfit: estimated model parameters:
      beta      tausq    sigmasq        phi 
" 102.449" " 114.921" "4184.402" "   9.991" 
Practical Range with cor=0.05 for asymptotic range: 29.9294

likfit: maximised log-likelihood = -896.9
```

```r
maximumLikelihoodNetherlandsInitial100.10
```

```
likfit: estimated model parameters:
      beta      tausq    sigmasq        phi 
" 102.449" " 114.921" "4184.402" "   9.991" 
Practical Range with cor=0.05 for asymptotic range: 29.9294

likfit: maximised log-likelihood = -896.9
```

```r
maximumLikelihoodNetherlandsInitial10.100
```

```
likfit: estimated model parameters:
     beta     tausq   sigmasq       phi 
"  102.7" "  117.7" "39625.5" "  100.0" 
Practical Range with cor=0.05 for asymptotic range: 299.5729

likfit: maximised log-likelihood = -897.8
```

```r
maximumLikelihoodNetherlandsInitial1.1
```

```
likfit: estimated model parameters:
      beta      tausq    sigmasq        phi 
```

```
" 102.376" " 114.249" "3036.627" "   7.132"
Practical Range with cor=0.05 for asymptotic range: 21.36583

likfit: maximised log-likelihood = -896.9
```

```
maximumLikelihoodNetherlandsInitial1000.1000
```

```
likfit: estimated model parameters:
      beta      tausq    sigmasq        phi
"   103.2" "   148.3" "246755.1" "  1000.0"
Practical Range with cor=0.05 for asymptotic range: 2995.732

likfit: maximised log-likelihood = -900.9
```

```
maximumLikelihoodNetherlandsInitial500.500
```

```
likfit: estimated model parameters:
      beta      tausq    sigmasq        phi
"   103.1" "   145.0" "130046.9" "   500.0"
Practical Range with cor=0.05 for asymptotic range: 1497.866

likfit: maximised log-likelihood = -900.1
```

As we can see from the maximised log-likelihoods it does seem that have indeed reached the maximum log-likelihood as none of the values are too fastly different and the likelihood worsedned as we started to increase much more our starting values.

Next we will try the REML that takes into account the fact that some of the parameters of the model are related to the variance of the residuals and not the mean.

```
REMLmaximumLikelihoodNetherlandsInitial10.1 = likfit(precipitationNetherland_geoR,
    ini.cov.pars = c(10, 1), lik.method = "REML")
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
```

```
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.

WARNING: estimated range is more than 10 times bigger than the biggest distance between two p
 1) excluding spatial dependence if estimated sill is too low and/or
 2) taking trends (covariates) into account
```

```r
REMLmaximumLikelihoodNetherlandsInitial1.10 = likfit(precipitationNetherland_geoR,
    ini.cov.pars = c(1, 10), lik.method = "REML")
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.

WARNING: estimated range is more than 10 times bigger than the biggest distance between two p
 1) excluding spatial dependence if estimated sill is too low and/or
 2) taking trends (covariates) into account
```

```r
REMLmaximumLikelihoodNetherlandsInitial100.1 = likfit(precipitationNetherland_geoR,
    ini.cov.pars = c(100, 1), lik.method = "REML")
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```
WARNING: estimated range is more than 10 times bigger than the biggest distance between two p
 1) excluding spatial dependence if estimated sill is too low and/or
 2) taking trends (covariates) into account
```

```r
REMLmaximumLikelihoodNetherlandsInitial1.100 = likfit(precipitationNetherland_geoR,
    ini.cov.pars = c(1, 100), lik.method = "REML")
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.

WARNING: estimated range is more than 10 times bigger than the biggest distance between two p
 1) excluding spatial dependence if estimated sill is too low and/or
 2) taking trends (covariates) into account
```

```r
REMLmaximumLikelihoodNetherlandsInitial10.1
```

```
likfit: estimated model parameters:
      beta      tausq    sigmasq        phi
"  102.61" "  114.60" "18253.12" "   43.35"
Practical Range with cor=0.05 for asymptotic range: 129.8698

likfit: maximised log-likelihood = -888.9
```

```r
REMLmaximumLikelihoodNetherlandsInitial1.10
```

```
likfit: estimated model parameters:
      beta      tausq    sigmasq        phi
"  102.62" "  114.83" "18124.93" "   43.19"
Practical Range with cor=0.05 for asymptotic range: 129.3933

likfit: maximised log-likelihood = -888.9
```

```r
REMLmaximumLikelihoodNetherlandsInitial100.1
```

```
likfit: estimated model parameters:
      beta      tausq    sigmasq        phi 
"  102.61" "  114.60" "18253.12" "   43.35" 
Practical Range with cor=0.05 for asymptotic range: 129.8698

likfit: maximised log-likelihood = -888.9
```

```r
REMLmaximumLikelihoodNetherlandsInitial1.100
```

```
likfit: estimated model parameters:
     beta     tausq   sigmasq       phi 
"  102.7" "  116.5" "40885.8" "  100.0" 
Practical Range with cor=0.05 for asymptotic range: 299.5729

likfit: maximised log-likelihood = -888.9
```

As we can see from these new models, the new likelihood method actually did improve our model as we have a lower log-likelihood.

Since the data did not seem to be perfectly stationary as seen in the previous questions, we will now check if adding a linear trend improves our model

```r
linearREMLmaximumLikelihoodNetherlandsInitial10.1 = likfit(precipitationNetherland_geoR,
    trend = "1st", ini.cov.pars = c(10, 1), lik.method = "REML")
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```r
linearREMLmaximumLikelihoodNetherlandsInitial1.10 = likfit(precipitationNetherland_geoR,
    trend = "1st", ini.cov.pars = c(1, 10), lik.method = "REML")
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```r
linearREMLmaximumLikelihoodNetherlandsInitial100.10 = likfit(precipitationNetherland_geoR,
    trend = "1st", ini.cov.pars = c(100, 10), lik.method = "REML")
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```r
linearREMLmaximumLikelihoodNetherlandsInitial10.100 = likfit(precipitationNetherland_geoR,
    trend = "1st", ini.cov.pars = c(10, 100), lik.method = "REML")
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
```

```
------------------------------------------------------------------
likfit: end of numerical maximisation.

WARNING: estimated range is more than 10 times bigger than the biggest distance between two p
 1) excluding spatial dependence if estimated sill is too low and/or
 2) taking trends (covariates) into account
```

```r
linearREMLmaximumLikelihoodNetherlandsInitial10.1
```

```
likfit: estimated model parameters:
       beta0        beta1        beta2        tausq      sigmasq          phi
"-2194.6747" "  -11.4587" "   45.2352" "  112.7395" "  191.5497" "    0.4033"
Practical Range with cor=0.05 for asymptotic range: 1.208203

likfit: maximised log-likelihood = -872
```

```r
linearREMLmaximumLikelihoodNetherlandsInitial1.10
```

```
likfit: estimated model parameters:
      beta0       beta1       beta2       tausq     sigmasq         phi
"-2084.141" "   -6.973" "   42.674" "  125.053" " 3238.867" "    9.905"
Practical Range with cor=0.05 for asymptotic range: 29.67433

likfit: maximised log-likelihood = -873
```

```r
linearREMLmaximumLikelihoodNetherlandsInitial100.10
```

```
likfit: estimated model parameters:
      beta0       beta1       beta2       tausq     sigmasq         phi
"-2084.141" "   -6.973" "   42.674" "  125.053" " 3238.867" "    9.905"
Practical Range with cor=0.05 for asymptotic range: 29.67433

likfit: maximised log-likelihood = -873
```

```r
linearREMLmaximumLikelihoodNetherlandsInitial10.100
```

```
likfit: estimated model parameters:
     beta0      beta1      beta2      tausq    sigmasq        phi 
"-2084.09" "   -6.91" "   42.67" "  125.89" "32191.57" "  100.00" 
Practical Range with cor=0.05 for asymptotic range: 299.5701

likfit: maximised log-likelihood = -873
```

From these models we can see that the linear trend does indeed improve our model, now we will check if there are any other covariance functions that can improve the model further.

```
Matren0.5linearREMLmaximumLikelihoodNetherlandsInitial10.1 = likfit(precipitationNetherlan
    trend = "1st", ini.cov.pars = c(10, 1), lik.method = "REML", cov.model = "matern",
    kappa = 0.5)
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```
Matren1.0linearREMLmaximumLikelihoodNetherlandsInitial10.1 = likfit(precipitationNetherlan
    trend = "1st", ini.cov.pars = c(10, 1), lik.method = "REML", cov.model = "matern",
    kappa = 1)
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```r
Matren1.5linearREMLmaximumLikelihoodNetherlandsInitial10.1 = likfit(precipitationNetherlan
    trend = "1st", ini.cov.pars = c(10, 1), lik.method = "REML", cov.model = "matern",
    kappa = 1.5)
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```r
Matren2.0linearREMLmaximumLikelihoodNetherlandsInitial10.1 = likfit(precipitationNetherlan
    trend = "1st", ini.cov.pars = c(10, 1), lik.method = "REML", cov.model = "matern",
    kappa = 2)
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```r
Matren2.5linearREMLmaximumLikelihoodNetherlandsInitial10.1 = likfit(precipitationNetherlan
    trend = "1st", ini.cov.pars = c(10, 1), lik.method = "REML", cov.model = "matern",
    kappa = 2.5)
```

```
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
```

```
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```
Matren0.5linearREMLmaximumLikelihoodNetherlandsInitial10.1
```

```
likfit: estimated model parameters:
       beta0         beta1         beta2         tausq       sigmasq           phi
"-2194.6747" "  -11.4587" "   45.2352" "  112.7395" "  191.5497" "    0.4033"
Practical Range with cor=0.05 for asymptotic range: 1.208203

likfit: maximised log-likelihood = -872
```

```
Matren1.0linearREMLmaximumLikelihoodNetherlandsInitial10.1
```

```
likfit: estimated model parameters:
       beta0         beta1         beta2         tausq       sigmasq           phi
"-2221.5117" "  -12.1627" "   45.8184" "  125.4406" "  160.1807" "    0.2088"
Practical Range with cor=0.05 for asymptotic range: 0.8347704

likfit: maximised log-likelihood = -871.3
```

```
Matren1.5linearREMLmaximumLikelihoodNetherlandsInitial10.1
```

```
likfit: estimated model parameters:
       beta0         beta1         beta2         tausq       sigmasq           phi
"-2233.7541" "  -12.4219" "   46.0784" "  129.6185" "  149.9124" "    0.1529"
Practical Range with cor=0.05 for asymptotic range: 0.7254113

likfit: maximised log-likelihood = -871.1
```

```
Matren2.0linearREMLmaximumLikelihoodNetherlandsInitial10.1
```

```
likfit: estimated model parameters:
       beta0         beta1         beta2         tausq       sigmasq           phi
"-2240.7056" "  -12.5595" "   46.2252" "  131.4685" "  144.9394" "    0.1248"
```

```
Practical Range with cor=0.05 for asymptotic range: 0.669996

likfit: maximised log-likelihood = -871.1
```

```
Matren2.5linearREMLmaximumLikelihoodNetherlandsInitial10.1
```

```
likfit: estimated model parameters:
      beta0        beta1        beta2        tausq      sigmasq          phi
"-2245.1703" "  -12.6453" "   46.3192" "  132.4280" "  142.0693" "    0.1074"
Practical Range with cor=0.05 for asymptotic range: 0.6358818

likfit: maximised log-likelihood = -871
```

It does seem that the matrén covariance function did indeed slightly improved the model so we will compare it to a model using a spherical covariance function. The spherical covariance function is appropriate for this scenario has the spatial correlation between data points decreases rapidly as the distance between the points increases and we are limited with the range of correlation has after 2 degrees of distance we loose sensible correlation, hence the cut in the variogram.

```
SphericallinearREMLmaximumLikelihoodNetherlandsInitial10.1 = likfit(precipitationNetherlan
    trend = "1st", ini.cov.pars = c(10, 1), lik.method = "REML", cov.model = "spherical")
```

```
kappa not used for the spherical correlation function
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```
SphericallinearREMLmaximumLikelihoodNetherlandsInitial1.10 = likfit(precipitationNetherlan
    trend = "1st", ini.cov.pars = c(1, 10), lik.method = "REML", cov.model = "spherical")
```

```
kappa not used for the spherical correlation function
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```
SphericallinearREMLmaximumLikelihoodNetherlandsInitial100.10 = likfit(precipitationNetherl
    trend = "1st", ini.cov.pars = c(100, 10), lik.method = "REML", cov.model = "spherical"
```

```
kappa not used for the spherical correlation function
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
likfit: end of numerical maximisation.
```

```
SphericallinearREMLmaximumLikelihoodNetherlandsInitial10.100 = likfit(precipitationNetherl
    trend = "1st", ini.cov.pars = c(10, 100), lik.method = "REML", cov.model = "spherical"
```

```
kappa not used for the spherical correlation function
---------------------------------------------------------------
likfit: likelihood maximisation using the function optim.
likfit: Use control() to pass additional
         arguments for the maximisation function.
        For further details see documentation for optim.
likfit: It is highly advisable to run this function several
        times with different initial values for the parameters.
likfit: WARNING: This step can be time demanding!
---------------------------------------------------------------
```

```
likfit: end of numerical maximisation.

WARNING: estimated range is more than 10 times bigger than the biggest distance between two p
 1) excluding spatial dependence if estimated sill is too low and/or
 2) taking trends (covariates) into account
```

```
SphericallinearREMLmaximumLikelihoodNetherlandsInitial10.1
```

```
likfit: estimated model parameters:
      beta0       beta1       beta2       tausq     sigmasq         phi
"-2182.422" "  -11.539" "   45.002" "  127.066" "  200.406" "    1.003"
Practical Range with cor=0.05 for asymptotic range: 1.00273

likfit: maximised log-likelihood = -872.2
```

```
SphericallinearREMLmaximumLikelihoodNetherlandsInitial1.10
```

```
likfit: estimated model parameters:
      beta0       beta1       beta2       tausq     sigmasq         phi
"-2082.850" "   -6.686" "   42.622" "  125.174" " 2151.559" "    9.905"
Practical Range with cor=0.05 for asymptotic range: 9.905214

likfit: maximised log-likelihood = -873
```

```
SphericallinearREMLmaximumLikelihoodNetherlandsInitial100.10
```

```
likfit: estimated model parameters:
      beta0       beta1       beta2       tausq     sigmasq         phi
"-2082.850" "   -6.686" "   42.622" "  125.174" " 2151.559" "    9.905"
Practical Range with cor=0.05 for asymptotic range: 9.905214

likfit: maximised log-likelihood = -873
```

```
SphericallinearREMLmaximumLikelihoodNetherlandsInitial10.100
```

```
likfit: estimated model parameters:
      beta0       beta1       beta2       tausq     sigmasq         phi
"-2083.890" "   -6.902" "   42.667" "  125.695" "21538.008" "   99.999"
Practical Range with cor=0.05 for asymptotic range: 99.99893

likfit: maximised log-likelihood = -873
```

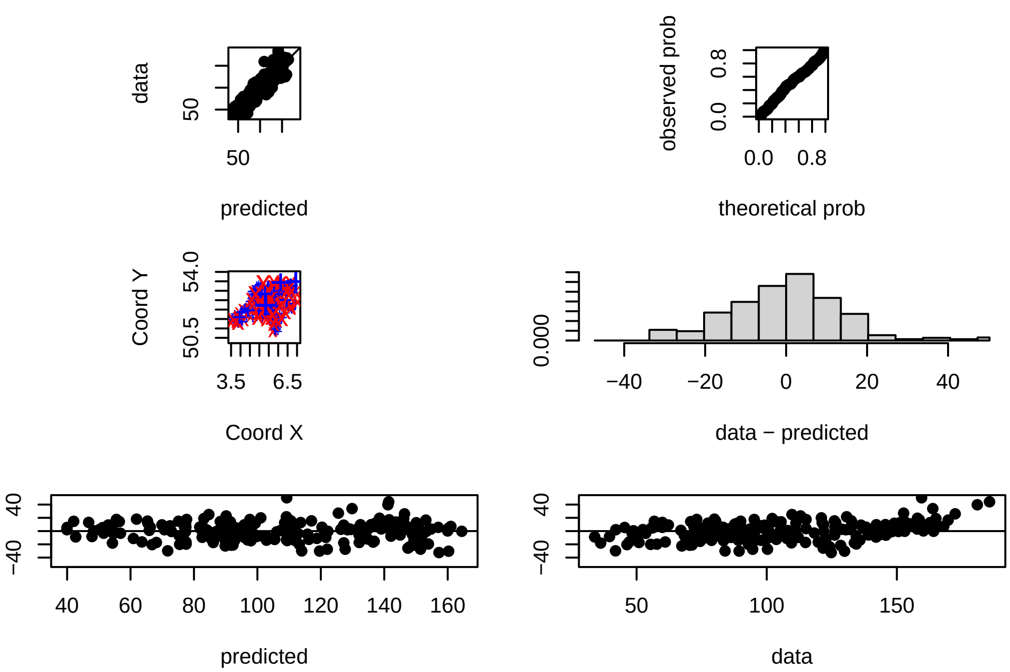
As we can see the spherical covariance function does not provide as good of a fit as the Matrén.

We will now validate the model by doing cross-validation on the model.

```
xv.ml = xvalid(precipitationNetherland_geoR, model = Matren2.5linearREMLmaximumLikelihoodN
```

```
xvalid: number of data locations       = 217
xvalid: number of validation locations = 217
xvalid: performing cross-validation at location ... 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
xvalid: end of cross-validation
```

```
par(mfrow = c(3, 2), mar = c(4, 2, 2, 2))
plot(xv.ml, error = TRUE, std.error = FALSE, pch = 19)
```



From these plots we can see that the residuals seem mostly normal without any quickly identifiable patterns or bias.

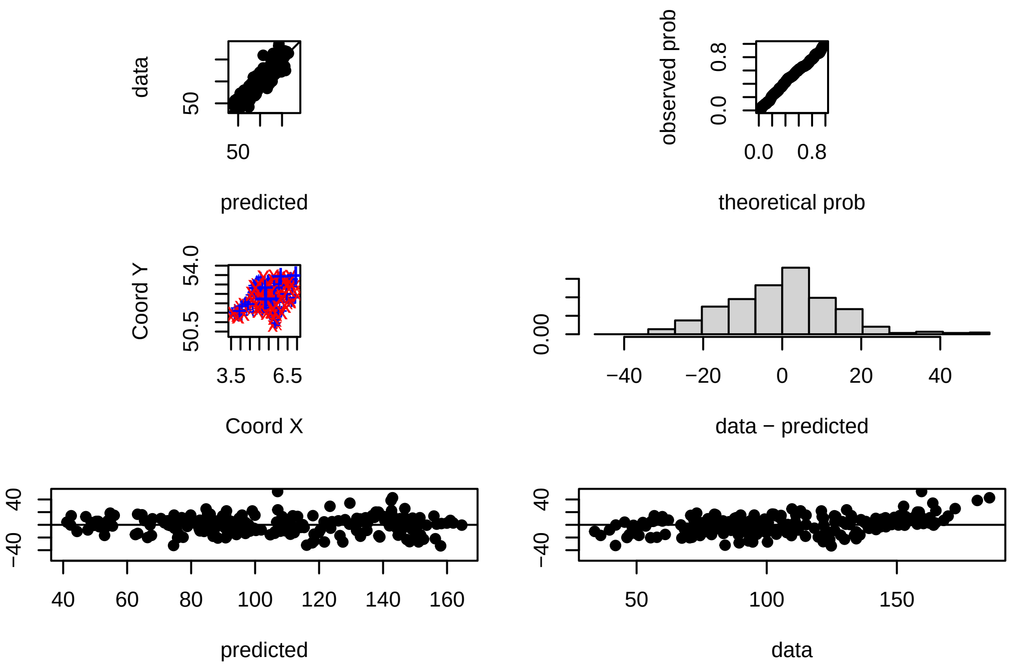
From the first top left graph we can see however that we seem to sightly underestimate more data points.

To account for bias we will also perform cross-validation to the next best performing model using the spherical function instead

```
xv.ml = xvalid(precipitationNetherland_geoR, model = SphericallinearREMLmaximumLikelihoodN
```

```
xvalid: number of data locations       = 217
xvalid: number of validation locations = 217
xvalid: performing cross-validation at location ... 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13
xvalid: end of cross-validation
```

```
par(mfrow = c(3, 2), mar = c(4, 2, 2, 2))
plot(xv.ml, error = TRUE, std.error = FALSE, pch = 19)
```



As we can see the data at the start seems to be systematically underestimated and at the end it seems to overestimated. Furthermore the theoretical data plot seems to be less linear.

This confirms that the spherical covariance function is indeed a worse model than the Matrén model.

1 f)

First we will start by making the predictions using the variogram

```
spatialPointsABC = randomRowsPrecipitation[, c("longitude", "latitude")]


# Set the krige.control parameters
krigeControl = krige.control(type.krige = "OK", cov.model = krigingVariogramFittedMatrén1.
    cov.pars = krigingVariogramFittedMatrén1.0$cov.pars)


# Kriging with the fitted variogram model
krigeResults = krige.conv(precipitationNetherland_geoR, locations = spatialPointsABC,
    krige = krigeControl)
```

```
krige.conv: model with constant mean
krige.conv: Kriging performed using global neighbourhood
```

```
# Extract predictions from the kriging results
predictions = krigeResults$predict

# Compare the predicted values with the actual precipitation values

actualPrecipitationValues = randomRowsPrecipitation[, 4]

comparisonvariogram = data.frame(actualPrecipitationValues, predictions)
```

Now for the maximum likelihood function

```
# done above spatialPointsABC = randomRowsPrecipitation[,
# c('longitude', 'latitude')]


# Set the krige.control parameters
krigeControl = krige.control(type.krige = "OK", cov.model = Matren2.5linearREMLmaximumLike
    cov.pars = Matren2.5linearREMLmaximumLikelihoodNetherlandsInitial10.1$cov.pars)
```

```r
# Kriging with the fitted variogram model
krigeResults = krige.conv(precipitationNetherland_geoR, locations = spatialPointsABC,
    krige = krigeControl)
```

```
krige.conv: model with constant mean
krige.conv: Kriging performed using global neighbourhood
```

```r
# Extract predictions from the kriging results
predictions = krigeResults$predict

# Compare the predicted values with the actual precipitation values

# done above actualPrecipitationValues = randomRowsPrecipitation[,4]
comparisonMaximumLikelihood = data.frame(actualPrecipitationValues, predictions)
```

Now that we have made the predictions for our 2 models we will check the predicted values compared to the real values for each of the models.

```r
comparisonvariogram
```

```
  precip predictions
1   89.1    95.41904
2   95.9    98.90558
3  147.2   147.82038
```

```r
comparisonMaximumLikelihood
```

```
  precip predictions
1   89.1    99.00780
2   95.9    99.36194
3  147.2   140.07734
```

```r
# Calculate Mean Absolute Error (MAE)
MAEVariogram = mean(abs(comparisonvariogram$precip - comparisonvariogram$predictions))
MAEMaximumLikelihood = mean(abs(comparisonMaximumLikelihood$precip - comparisonMaximumLike
```

```
# Calculate Mean Squared Error (MSE)
mseVariogram = mean((comparisonvariogram$precip - comparisonvariogram$predictions)^2)
mseMaximumLikelihood = mean((comparisonMaximumLikelihood$precip - comparisonMaximumLikelih


# Calculate Root Mean Squared Error (RMSE)
rmseVariogram = sqrt(mseVariogram)
rmseMaximumLikelihood = sqrt(mseMaximumLikelihood)


# Display the calculated metrics
cat("Mean Absolute Error (MAE) of the Variogram:", MAEVariogram, "\n")
```

```
Mean Absolute Error (MAE) of the Variogram: 3.315002
```

```
cat("Mean Squared Error (MSE) of the Variogram:", mseVariogram, "\n")
```

```
Mean Squared Error (MSE) of the Variogram: 16.44957
```

```
cat("Root Mean Squared Error (RMSE) of the Variogram:", rmseVariogram, "\n")
```

```
Root Mean Squared Error (RMSE) of the Variogram: 4.055807
```

```
cat("\n\n")
```

```
cat("Mean Absolute Error (MAE) of the maximum likelihood:", MAEMaximumLikelihood,
    "\n")
```

```
Mean Absolute Error (MAE) of the maximum likelihood: 6.830801
```

```
cat("Mean Squared Error (MSE) of the maximum likelihood:", mseMaximumLikelihood,
    "\n")
```

```
Mean Squared Error (MSE) of the maximum likelihood: 53.62729
```

```
cat("Root Mean Squared Error (RMSE) of the maximum likelihood:", rmseMaximumLikelihood,
    "\n")
```

```
Root Mean Squared Error (RMSE) of the maximum likelihood: 7.323066
```

As we can see from both the real values and the MAE, MSE and RMSE the variogram has as much better performance predicting those 3 points than our maximum likelihood model

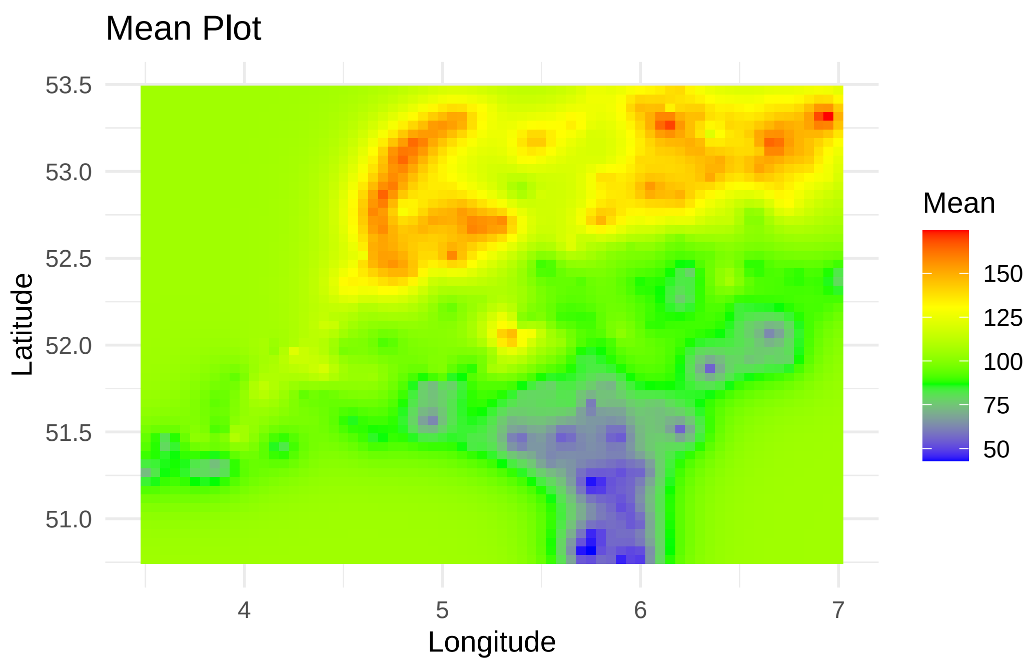
1 g)

```
# Determine the range of the coordinates
xRange = range(precipitationNetherland_geoR$coords[, 1])
yRange = range(precipitationNetherland_geoR$coords[, 2])

# Create a grid with 0.05-degree spacing
gridPoints = expand.grid(x = seq(xRange[1], xRange[2], by = 0.05), y = seq(yRange[1],
    yRange[2], by = 0.05))


# Kriging with the fitted variogram model
krigeResults = krige.conv(precipitationNetherland_geoR, locations = gridPoints,
    krige = krigeControl)
```

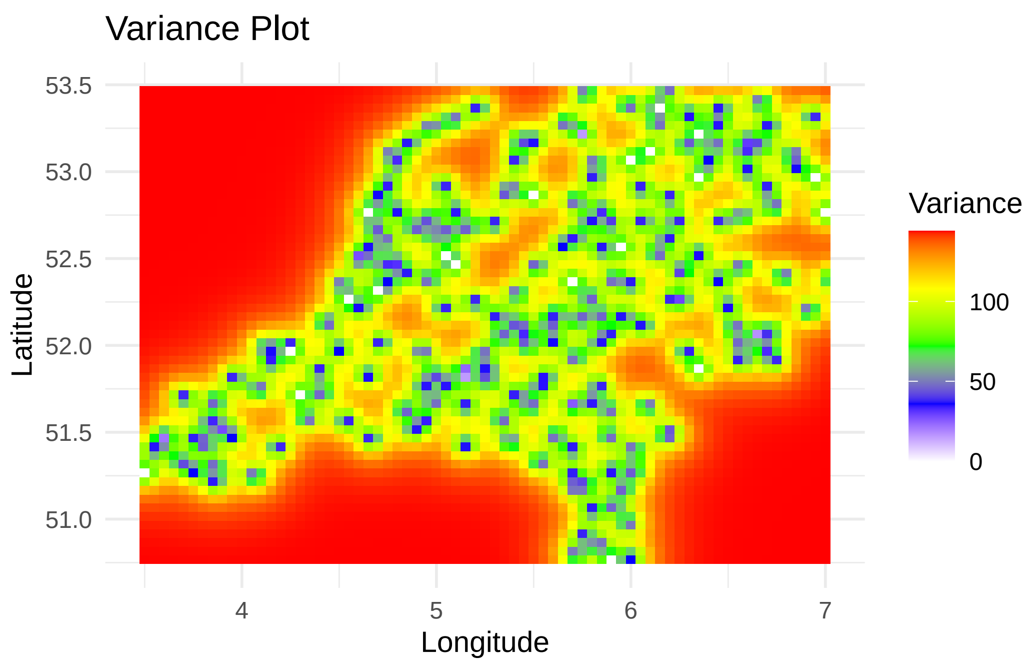
```
krige.conv: model with constant mean
krige.conv: Kriging performed using global neighbourhood
```

```
# Create a data frame for the grid points with the predicted mean and
# variance
gridData = data.frame(gridPoints, mean = krigeResults$predict, variance = krigeResults$kri

# Mean plot
meanPlot = ggplot(gridData, aes(x = x, y = y, fill = mean)) + geom_tile() +
    scale_fill_gradientn(colors = c("blue", "green", "yellow", "red")) +
    theme_minimal() + ggtitle("Mean Plot") + labs(x = "Longitude", y = "Latitude",
    fill = "Mean")

# Variance plot
variancePlot = ggplot(gridData, aes(x = x, y = y, fill = variance)) + geom_tile() +
```

```r
    scale_fill_gradientn(colors = c("white", "blue", "green", "yellow", "red")) +
    theme_minimal() + ggtitle("Variance Plot") + labs(x = "Longitude", y = "Latitude",
    fill = "Variance")

# Display the plots
print(meanPlot)
```




```r
print(variancePlot)
```



1 h)

For the priors I will be using the estimated values from our latest maximum likelihood model.

```
# Extract the estimated parameters

priorPhiVariogram = krigingVariogramFittedMatrén1.0$cov.pars[1]
priorTauSQVariogram = krigingVariogramFittedMatrén1.0$cov.pars[2]


priorPhiMaximumLikelihood = Matren1.0linearREMLmaximumLikelihoodNetherlandsInitial10.1$cov
priorTauSQMaximumLikelihood = Matren1.0linearREMLmaximumLikelihoodNetherlandsInitial10.1$c
```

The function does not support continuous priors directly so we will fit them as discrete priors.

```
# Creating discrete priors for phi and tau^2_rel AKA this is for the
# model with a nugget
phiDiscrete <- seq(min(priorPhiVariogram, priorPhiMaximumLikelihood) * 0.5,
    max(priorPhiVariogram, priorPhiMaximumLikelihood) * 1.5, length.out = 50)

tauSqDiscrete <- seq(min(priorTauSQVariogram, priorTauSQMaximumLikelihood) *
```

```r
    0.5, max(priorTauSQVariogram, priorTauSQMaximumLikelihood) * 1.5, length.out = 50)

# Informative priors based on the parameter estimates
phiProbability <- dnorm(phiDiscrete, mean = (priorPhiVariogram + priorPhiMaximumLikelihood
    sd = abs(priorPhiVariogram - priorPhiMaximumLikelihood)/2)
tauSqProbability <- dnorm(tauSqDiscrete, mean = (priorTauSQVariogram + priorTauSQMaximumLi
    sd = abs(priorTauSQVariogram - priorTauSQMaximumLikelihood)/2)

# Normalizing the probabilities
phiProbability <- phiProbability/sum(phiProbability)
tauSqProbability <- tauSqProbability/sum(tauSqProbability)


ex.grid <- as.matrix(expand.grid(seq(50.5, 53.5, l = 15), seq(3.5, 7, l = 15)))


# Fitting the krige.bayes model with the informative priors
krigeBayesModelWithNugget <- krige.bayes(geodata = precipitationNetherland_geoR,
    loc = ex.grid, prior = prior.control(phi.prior = phiProbability, phi.discrete = phiDis
        tausq.rel.prior = tauSqProbability, tausq.rel.discrete = tauSqDiscrete))
```

```
krige.bayes: model with constant mean
krige.bayes: computing the discrete posterior of phi/tausq.rel
krige.bayes: computing the posterior probabilities.
             Number of parameter sets:  2500
1, 101, 201, 301, 401, 501, 601, 701, 801, 901, 1001, 1101, 1201, 1301, 1401, 1501, 1601, 170

krige.bayes: sampling from posterior distribution
krige.bayes: sample from the (joint) posterior of phi and tausq.rel
                  [,1]
phi         80.0903555
tausq.rel    0.1043848
frequency 1000.0000000

krige.bayes: starting prediction at the provided locations
krige.bayes: phi/tausq.rel samples for the predictive are same as for the posterior
krige.bayes: computing moments of the predictive distribution
krige.bayes: sampling from the predictive
             Number of parameter sets:  1
1,
krige.bayes: preparing summaries of the predictive distribution
```

```r
krigeBayesModelWithoutNugget <- krige.bayes(geodata = precipitationNetherland_geoR,
    loc = ex.grid, prior = prior.control(phi.prior = phiProbability, phi.discrete = phiDis
```

```
krige.bayes: model with constant mean
krige.bayes: computing the discrete posterior of phi/tausq.rel
krige.bayes: computing the posterior probabilities.
             Number of parameter sets:  50
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 2

krige.bayes: sampling from posterior distribution
krige.bayes: sample from the (joint) posterior of phi and tausq.rel
              [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
phi       80.09036 226.3799 372.6695 518.9591 665.2486 811.5382 957.8278
tausq.rel  0.00000   0.0000   0.0000   0.0000   0.0000   0.0000   0.0000
frequency 16.00000  20.0000  14.0000  31.0000  23.0000  30.0000  23.0000
              [,8]     [,9]    [,10]    [,11]    [,12]    [,13]    [,14]
phi       1104.117 1250.407 1396.696 1542.986 1689.276 1835.565 1981.855
tausq.rel    0.000    0.000    0.000    0.000    0.000    0.000    0.000
frequency   24.000   21.000   28.000   21.000   27.000   28.000   31.000
             [,15]    [,16]    [,17]    [,18]    [,19]    [,20]    [,21]
phi       2128.144 2274.434 2420.723 2567.013 2713.303 2859.592 3005.882
tausq.rel    0.000    0.000    0.000    0.000    0.000    0.000    0.000
frequency   46.000   32.000   37.000   28.000   26.000   26.000   28.000
             [,22]    [,23]   [,24]   [,25]   [,26]    [,27]    [,28]    [,29]
phi       3152.171 3298.461 3444.75 3591.04 3737.33 3883.619 4029.909 4176.198
tausq.rel    0.000    0.000    0.00    0.00    0.00    0.000    0.000    0.000
frequency   30.000   37.000   20.00   18.00   23.00   14.000   24.000   20.000
             [,30]    [,31]    [,32]    [,33]    [,34]    [,35]    [,36]
phi       4322.488 4468.777 4615.067 4761.357 4907.646 5053.936 5200.225
tausq.rel    0.000    0.000    0.000    0.000    0.000    0.000    0.000
frequency   21.000   19.000   16.000   19.000   19.000   20.000   20.000
             [,37]    [,38]    [,39]    [,40]    [,41]    [,42]    [,43]
phi       5346.515 5492.804 5639.094 5785.384 5931.673 6077.963 6224.252
tausq.rel    0.000    0.000    0.000    0.000    0.000    0.000    0.000
frequency   17.000   14.000    8.000   17.000    5.000   12.000    9.000
             [,44]    [,45]    [,46]    [,47]  [,48]   [,49]    [,50]
phi       6370.542 6516.831 6663.121 6809.411 6955.7 7101.99 7248.279
tausq.rel    0.000    0.000    0.000    0.000    0.0    0.00    0.000
frequency    9.000    8.000    7.000    5.000    2.0    3.00    4.000

krige.bayes: starting prediction at the provided locations
krige.bayes: phi/tausq.rel samples for the predictive are same as for the posterior
```

```
krige.bayes: computing moments of the predictive distribution
krige.bayes: sampling from the predictive
             Number of parameter sets:  50
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26
krige.bayes: preparing summaries of the predictive distribution
```

```r
summary(krigeBayesModelWithoutNugget$posterior$sample)
```

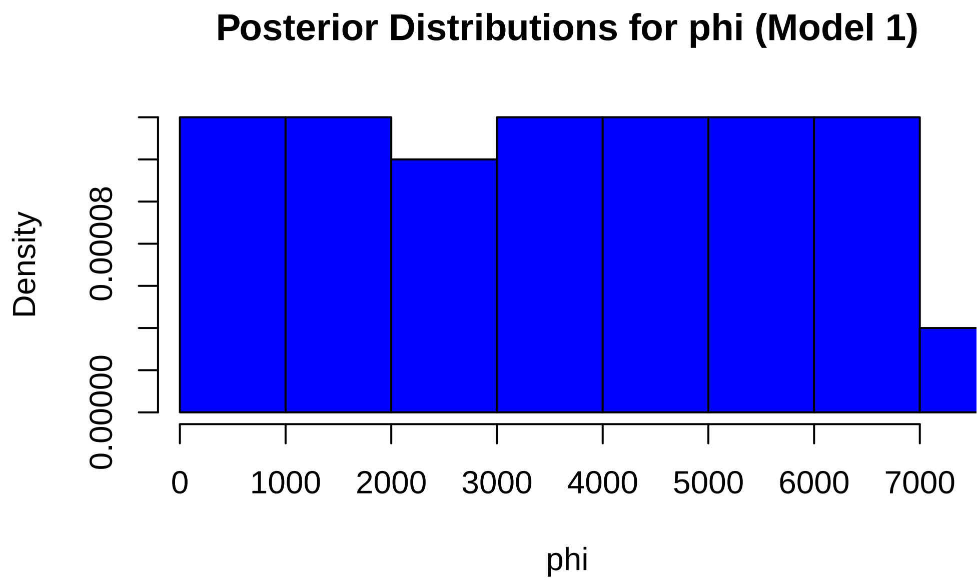
```
      beta              sigmasq               phi             tausq.rel
 Min.   :-14806.5   Min.   :  375543   Min.   :  80.09   Min.   :0
 1st Qu.: -2113.0   1st Qu.: 8331936   1st Qu.:1542.99   1st Qu.:0
 Median :   290.8   Median :14519487   Median :2713.30   Median :0
 Mean   :   299.1   Mean   :15700517   Mean   :2965.95   Mean   :0
 3rd Qu.:  2547.6   3rd Qu.:22223235   3rd Qu.:4322.49   3rd Qu.:0
 Max.   : 16575.4   Max.   :44053886   Max.   :7248.28   Max.   :0
```

```r
summary(krigeBayesModelWithNugget$posterior$sample)
```

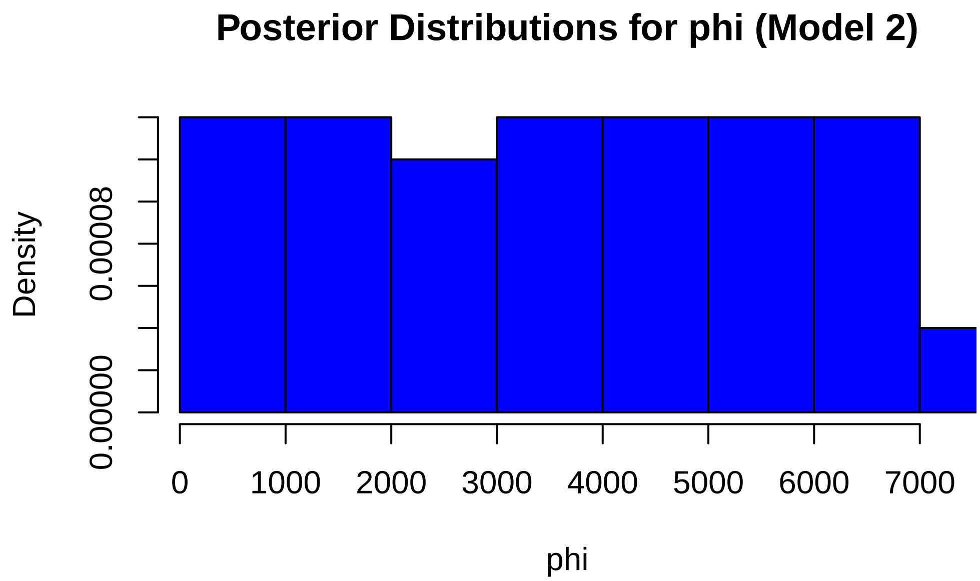
```
      beta            sigmasq          phi            tausq.rel
 Min.   :-49.52   Min.   :2216   Min.   :80.09   Min.   :0.1044
 1st Qu.: 66.57   1st Qu.:3000   1st Qu.:80.09   1st Qu.:0.1044
 Median :106.16   Median :3207   Median :80.09   Median :0.1044
 Mean   :105.66   Mean   :3238   Mean   :80.09   Mean   :0.1044
 3rd Qu.:145.57   3rd Qu.:3433   3rd Qu.:80.09   3rd Qu.:0.1044
 Max.   :282.96   Max.   :4455   Max.   :80.09   Max.   :0.1044
```

Now we will compare the posterior of both of the models to see the impact of the nugget

```r
# Extract posterior samples for phi and tau^2_rel
posterior_samples_model1 <- krigeBayesModelWithNugget$posterior$phi$phi.marginal$phi
posterior_samples_model2 <- krigeBayesModelWithoutNugget$posterior$phi$phi.marginal$phi

# Plot the posterior distributions
hist(posterior_samples_model1, freq = FALSE, main = "Posterior Distributions for phi (Mode
    xlab = "phi", col = "blue", xlim = range(c(posterior_samples_model1,
        posterior_samples_model2)))
```




```
hist(posterior_samples_model2, freq = FALSE, main = "Posterior Distributions for phi (Mode
    xlab = "phi", col = "blue", xlim = range(c(posterior_samples_model1,
        posterior_samples_model2)))
```




```r
# Compare summary statistics
summary_model1 <- summary(posterior_samples_model1)
summary_model2 <- summary(posterior_samples_model2)

cat("model 1 :\n")
```

```
model 1 :
```

```r
summary_model1
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  80.09 1872.14 3664.18 3664.18 5456.23 7248.28 
```

```r
cat("\n\n model 2 :\n")
```

```
 model 2 :
```

```
summary_model2
```

```
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 80.09 1872.14 3664.18 3664.18 5456.23 7248.28 
```

```
# Reset the plot layout
par(mfrow = c(1, 1))
```

As we can seem with a low number of binds we can't see any significant difference in the summaries or histogram between the models with and without a nugget

Question 2

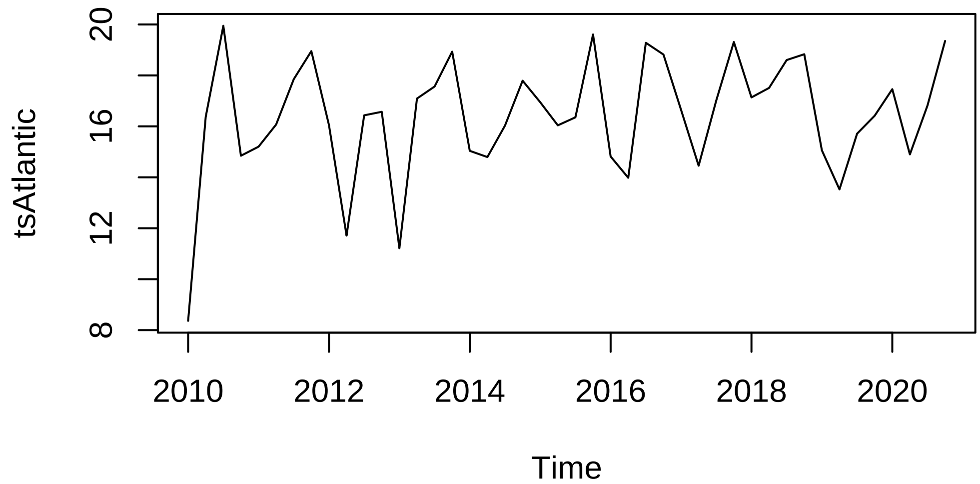
2 a)

We fist start by making the appropriate changes in the data to average the data to quarterly means

```
AMOCDF$Date = as.Date(AMOCDF$Date, format = "%d/%m/%Y")

## I will now make a column with the quarter and year that I will use
## to create the averages per quarter
AMOCDF$YearQuarter = paste(AMOCDF$Year, AMOCDF$Quarter, sep = "-")

YearQuarterAverage = AMOCDF %>%
    group_by(YearQuarter) %>%
    summarise(AverageStrength = mean(Strength))
```

Now we will convert the average data to a time series object to be able to plot it

```
tsAtlantic = ts(YearQuarterAverage, start = c(2010, 1), frequency = 4)

tsAtlantic = tsAtlantic[, "AverageStrength"]

plot.ts(tsAtlantic)
```



Trend analysis

From this graph we can see a yearly oscillation of Sverdrups. We can also identify that the peaks in Sverdrups are usually in the last quarter before the start of a new year and the valleys are on the second quarter of the year.

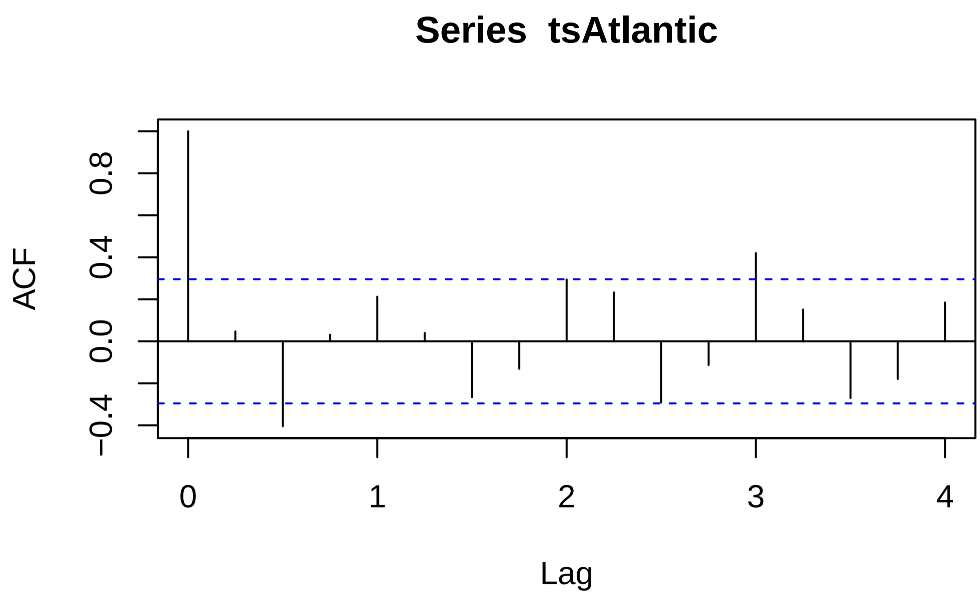
The data does seem stationary enough that if we were to differentiate we would start losing some of the structure.

2 b)

ACF

First we will start by checking the ACF(Autocorrelation Function) and PACF(Partial Autocorrelation Function) to check for if we have stationary data or not to help us decide between an ARMA or an ARIMA model.

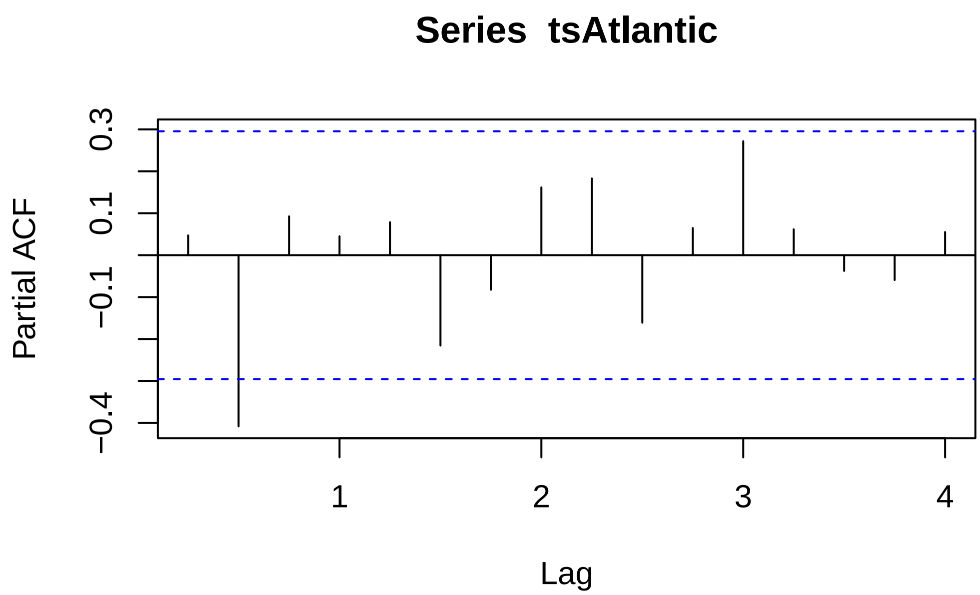
```
acf(tsAtlantic)
```



We can see that for ACF OF Average strength slowly decreases as lag increases to infinity with lag = 3 still being a significant values, meaning it is not a simple MA model as AR is clearly not quickly cut-off.

PACF

```
pacf(tsAtlantic)
```



The PACF seems to be cut-off at lag 0,5 indicating an AR model might be a best fit for our data to be a but with some almost significant values after the cut it might be also appropriate to some non-zero q values to confirm our initial assumption

As such we will now proceed to fit multiple model firstly with the initial assumption that, then I will both use models with non-zero q and the model given by the auto.arima function to double check that the assumptions made by the previous analyses is correct.

```
# it is always a good practice to try multiple values of p,d and q to
# see if we can do better we then obviously compare via the AIC of the
# models and their log likelihoods it is never enough to check those we
# also need to check the residuals

## order is p, d ,q

## initial models under our assumptions

model100 = Arima(tsAtlantic, order = c(1, 0, 0))
model200 = Arima(tsAtlantic, order = c(2, 0, 0))
model300 = Arima(tsAtlantic, order = c(3, 0, 0))

## now I will add postive q values
```

```r
model101 = Arima(tsAtlantic, order = c(1, 0, 1))
model102 = Arima(tsAtlantic, order = c(1, 0, 2))
model103 = Arima(tsAtlantic, order = c(1, 0, 3))

model201 = Arima(tsAtlantic, order = c(2, 0, 1))
model202 = Arima(tsAtlantic, order = c(2, 0, 2))
model203 = Arima(tsAtlantic, order = c(2, 0, 3))

model301 = Arima(tsAtlantic, order = c(3, 0, 1))
model302 = Arima(tsAtlantic, order = c(3, 0, 2))
model303 = Arima(tsAtlantic, order = c(3, 0, 3))



## lastly we will use auto.arima without seasonality to confirm our
## inital assumptions
modelAuto = auto.arima(tsAtlantic, max.d = 0, max.p = 5, max.q = 5, seasonal = FALSE)
```

best model selection

```r
model100
```

```
Series: tsAtlantic
ARIMA(1,0,0) with non-zero mean

Coefficients:
         ar1     mean
      0.0665  16.3878
s.e.  0.1788   0.3726

sigma^2 = 5.572:  log likelihood = -99.2
AIC=204.41   AICc=205.01   BIC=209.76
```

```r
model200
```

```
Series: tsAtlantic
ARIMA(2,0,0) with non-zero mean

Coefficients:
```

```
         ar1      ar2     mean
      0.0990  -0.5565  16.4298
s.e.  0.1576   0.1488   0.2113

sigma^2 = 4.321:  log likelihood = -93.45
AIC=194.9   AICc=195.92   BIC=202.04
```

```
model300
```

```
Series: tsAtlantic
ARIMA(3,0,0) with non-zero mean

Coefficients:
         ar1      ar2     ar3     mean
      0.1626  -0.5690  0.1464  16.4227
s.e.  0.1729   0.1479  0.1708   0.2409

sigma^2 = 4.35:  log likelihood = -93.09
AIC=196.17   AICc=197.75   BIC=205.1
```

As we can see from these inital models ARIMA(2,0,0) is the model that has the best fit has we can see from its lower AIC score of 194,9.

Now we will check against the other models to check the validity of our assumptions.

```
model101
```

```
Series: tsAtlantic
ARIMA(1,0,1) with non-zero mean

Coefficients:
          ar1     ma1     mean
      -0.4204  0.7718  16.3721
s.e.   0.2390  0.1466   0.4067

sigma^2 = 5.045:  log likelihood = -96.64
AIC=201.29   AICc=202.31   BIC=208.43
```

```
model102
```

```
Series: tsAtlantic
ARIMA(1,0,2) with non-zero mean

Coefficients:
         ar1     ma1      ma2     mean
      0.0230  0.1275  -0.4485  16.4289
s.e.  0.3051  0.2420   0.1348   0.2224

sigma^2 = 4.651:  log likelihood = -94.41
AIC=198.81   AICc=200.39   BIC=207.73
```

```
model103
```

```
Series: tsAtlantic
ARIMA(1,0,3) with non-zero mean

Coefficients:
          ar1     ma1      ma2      ma3     mean
      -0.5284  0.7214  -0.3646  -0.3072  16.4299
s.e.   0.9228  0.8649   0.2077   0.3545   0.2195

sigma^2 = 4.733:  log likelihood = -94.25
AIC=200.5   AICc=202.77   BIC=211.21
```

```
model201
```

```
Series: tsAtlantic
ARIMA(2,0,1) with non-zero mean

Coefficients:
          ar1      ar2     ma1     mean
      -0.0669  -0.5475  0.2187  16.4255
s.e.   0.2740   0.1555  0.2883   0.2300

sigma^2 = 4.366:  log likelihood = -93.15
AIC=196.31   AICc=197.88   BIC=205.23
```

```
model202
```

```
Series: tsAtlantic
ARIMA(2,0,2) with non-zero mean

Coefficients:
         ar1      ar2      ma1     ma2     mean
      0.0787  -0.9982  -0.0255  0.9999  16.4015
s.e.  0.0285   0.0066   0.0899  0.1158   0.2684

sigma^2 = 3.378:  log likelihood = -89.46
AIC=190.91   AICc=193.18   BIC=201.62
```

```
model203
```

```
Series: tsAtlantic
ARIMA(2,0,3) with non-zero mean

Coefficients:
         ar1      ar2     ma1     ma2     ma3     mean
      0.0325  -0.9621  0.0499  0.8487  0.4147  16.4028
s.e.  0.0645   0.0442  0.1987  0.2041  0.2511   0.3044

sigma^2 = 3.315:  log likelihood = -89.07
AIC=192.13   AICc=195.25   BIC=204.62
```

```
model301
```

```
Series: tsAtlantic
ARIMA(3,0,1) with non-zero mean

Coefficients:
         ar1      ar2     ar3      ma1     mean
      0.4092  -0.5931  0.2864  -0.2467  16.4191
s.e.  0.6330   0.1651  0.3580   0.6291   0.2537

sigma^2 = 4.449:  log likelihood = -93.03
AIC=198.06   AICc=200.34   BIC=208.77
```

```
model302
```

```
Series: tsAtlantic
ARIMA(3,0,2) with non-zero mean

Coefficients:
         ar1      ar2     ar3      ma1     ma2     mean
      0.2684  -0.9851  0.2222  -0.3030  1.0000  16.4144
s.e.  0.1999   0.0305  0.1995   0.1453  0.1921   0.2922

sigma^2 = 3.392:  log likelihood = -89.53
AIC=193.06   AICc=196.17   BIC=205.54
```

```
model303
```

```
Series: tsAtlantic
ARIMA(3,0,3) with non-zero mean

Coefficients:
          ar1      ar2      ar3     ma1     ma2     ma3     mean
      -0.3983  -0.9518  -0.4263  0.4381  0.7816  0.7352  16.4197
s.e.   0.3385   0.0412   0.3442  0.2946  0.1690  0.2422   0.2691

sigma^2 = 3.352:  log likelihood = -88.54
AIC=193.08   AICc=197.19   BIC=207.35
```

In this initial analysis we have found models that do have a lower AIC lower log likelihood than our previous best model, however these model ma's standard error are to close the the ma values indicating that while we are getting a better fit we might be overfitting to our data.

As such this does confirm our initial assumption for the choice of a zero q value.

Now lastly we will check if the auto.arima function does comfirm our initial assumptions.

```
modelAuto
```

```
Series: tsAtlantic
ARIMA(2,0,0) with non-zero mean

Coefficients:
         ar1      ar2     mean
      0.0990  -0.5565  16.4298
s.e.  0.1576   0.1488   0.2113
```

```
sigma^2 = 4.321:  log likelihood = -93.45
AIC=194.9   AICc=195.92   BIC=202.04
```

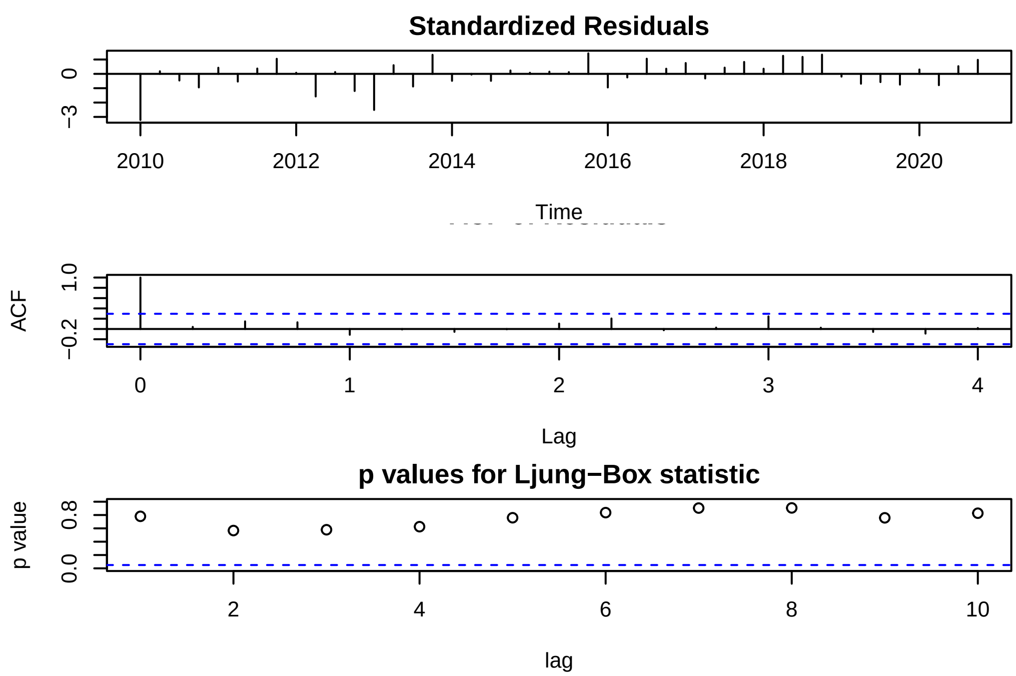
The function does confirm our assumption that ARIMA(2,0,0) is indeed the best model.

We will now check the residuals to verify if any of ou previously selected model validates well or if it is simply the best of bad models.

talk about the model being more easily explainability becaues MA = 0

Best model residual validation

```r
# Set smaller margins
par(mar = c(4, 4, 2, 2))

tsdiag(model200)
```




```r
# Reset margins
par(mar = c(5, 4, 4, 2) + 0.1)
```

Initially from the standardised residuals plot we can identify some sort of sinusoidal pattern, this implies that there is a seasonal trend that is not being accounted for in our model and as such this trends needs to be accounted in future models to better explain and increase the prediction power of a new model.

Forecasting

Now using the forecast function we will forecast the next 4 quarters of 2021

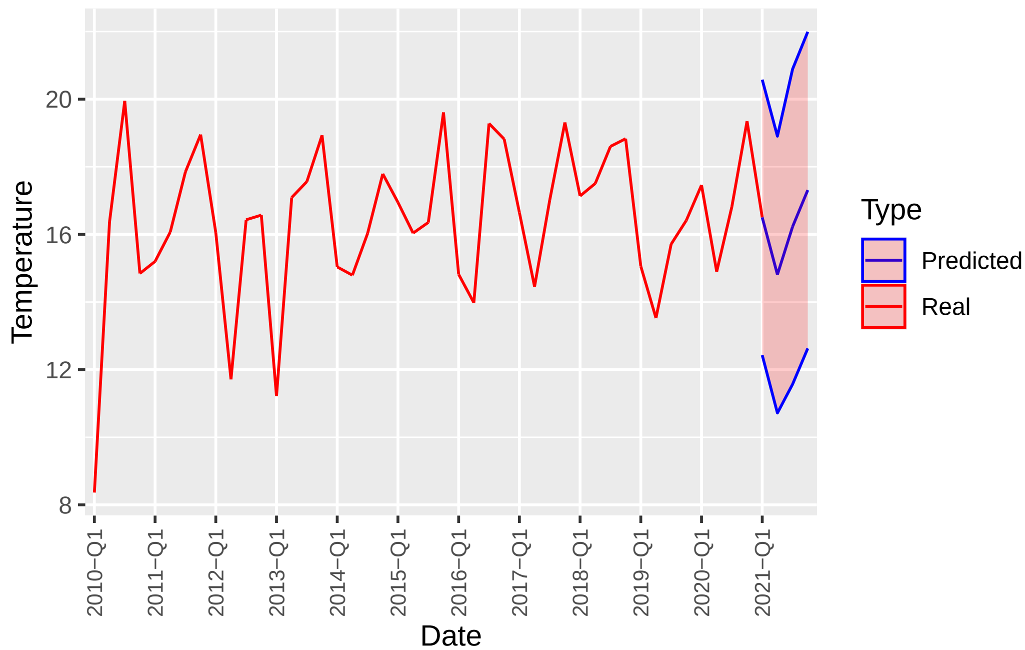
```r
forecast(model200, 4)
```

```
        Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
2021 Q1       16.50240 13.83841 19.16639 12.42818 20.57662
2021 Q2       14.81104 12.13403 17.48804 10.71691 18.90517
2021 Q3       16.22919 13.18168 19.27669 11.56843 20.88994
2021 Q4       17.31076 14.24941 20.37212 12.62882 21.99271
```

But this data is better visualized in a graph to better understand if the predictions are sensible compared to our real data.

```r
predictedArimaDF = data.frame(forecast(model200, 4))

predictedArimaDF$YearQuarter = c("2021-Q1", "2021-Q2", "2021-Q3", "2021-Q4")

# Combine real_data and pred_data into a single data frame
combinedDataframeAMOC = rbind(data.frame(Date = YearQuarterAverage$YearQuarter,
    Temperature = YearQuarterAverage$AverageStrength, Type = "Real"), data.frame(Date = pr
    Temperature = predictedArimaDF$Point.Forecast, Type = "Predicted"))

predictedArimaDF$Temperature = predictedArimaDF$Point.Forecast

predictedArimaDF$Type = "Predicted"


# Create the ggplot
plotARIMA = ggplot(combinedDataframeAMOC, aes(x = Date, y = Temperature,
    color = Type, group = 1)) + geom_line() + scale_color_manual(values = c("blue",
    "red"))

# Add the 95% confidence interval
plotARIMA = plotARIMA + geom_ribbon(data = predictedArimaDF, aes(x = YearQuarter,
    ymin = Lo.95, ymax = Hi.95), fill = "red", alpha = 0.2)
```

```r
# Adjust the x-axis labels
plotARIMA = plotARIMA + scale_x_discrete(breaks = combinedDataframeAMOC$Date[c(TRUE,
    rep(FALSE, 3))], labels = combinedDataframeAMOC$Date[c(TRUE, rep(FALSE,
    3))])

plotARIMA = plotARIMA + theme(axis.text.x = element_text(angle = 90, vjust = 0.5,
    size = 8))

# Display the plot
print(plotARIMA)
```



As we can see from the graph the ARIMA (2,0,0) seems to give us a sensible forecast for the 2021 quarter values, however as we can see the interval of the prediction accuracy our model is not too certain on the values most likely due to our model not accounting for the seasonal cycle of our data.

2 c)

Initial assumptions

From the previous exploratory analysis of the data we have established that the data did not need to be differentiated since it was constant, this translates to polynomial DLM component of order 2 that will use linear model to account for this type of changes in the data.

Furthermore, from the residual analysis we have inferred that there is an underlying seasonal trend present on the data, this seasonal trend will be represented by a seasonal component of frequency 4 to represent the 4 quarters per year.

model fitting

```
## linear model, order = 2, quadratic order = 3 , etc

## what we want is a linear model with a seasonal component so we add
## the 2 components together in a model

## things to try, another term like quadratic, or a arma component
## stacked on top of this

## Initial model with a linear polynomial and a seasonal component

buildFun = function(x) {
    dlmModPoly(order = 2, dV = exp(x[1]), dW = c(0, exp(x[2]))) + dlmModSeas(frequency = 4
        dV = 0, dW = c(exp(x[3]), rep(0, 2)))
}

linearDLM = dlmMLE(tsAtlantic, parm = c(0, 0, 0), build = buildFun)

linearDLM$par
```

```
[1]   1.151339 -18.078101  -2.189479
```

```
fittedLinearDLM = buildFun(linearDLM$par)

V(fittedLinearDLM)
```

```
         [,1]
[1,] 3.162425
```

```
W(fittedLinearDLM)
```

```
     [,1]         [,2]     [,3] [,4] [,5]
[1,]    0 0.000000e+00 0.000000    0    0
[2,]    0 1.408576e-08 0.000000    0    0
[3,]    0 0.000000e+00 0.111975    0    0
[4,]    0 0.000000e+00 0.000000    0    0
[5,]    0 0.000000e+00 0.000000    0    0
```

```
## second model with a quadratic polynomial and a seasonal component

buildFunQuad = function(x) {
    dlmModPoly(order = 3, dV = exp(x[1]), dW = c(0, exp(x[2]), exp(x[3]))) +
        dlmModSeas(frequency = 4, dV = 0, dW = c(exp(x[4]), rep(0, 2)))
}

quadraticDLM = dlmMLE(tsAtlantic, parm = c(0, 0, 0, 0), build = buildFunQuad)

quadraticDLM$par
```

```
[1]   1.161355 -17.807081 -28.603103  -2.352292
```

```
fittedQuadraticDLM = buildFunQuad(quadraticDLM$par)

V(fittedQuadraticDLM)
```

```
         [,1]
[1,] 3.194257
```

```
W(fittedQuadraticDLM)
```

```
     [,1]         [,2]         [,3]       [,4] [,5] [,6]
[1,]    0 0.000000e+00 0.000000e+00 0.00000000    0    0
[2,]    0 1.847069e-08 0.000000e+00 0.00000000    0    0
[3,]    0 0.000000e+00 3.782948e-13 0.00000000    0    0
[4,]    0 0.000000e+00 0.000000e+00 0.09515082    0    0
[5,]    0 0.000000e+00 0.000000e+00 0.00000000    0    0
[6,]    0 0.000000e+00 0.000000e+00 0.00000000    0    0
```

Now we will compare both models through their log likelihood using the dlmLL function and see if the extra flexibility from the extra polynomial function is providing a better fit

```r
dlmLL(tsAtlantic, fittedLinearDLM)
```

```
[1] 94.98804
```

```r
dlmLL(tsAtlantic, fittedQuadraticDLM)
```

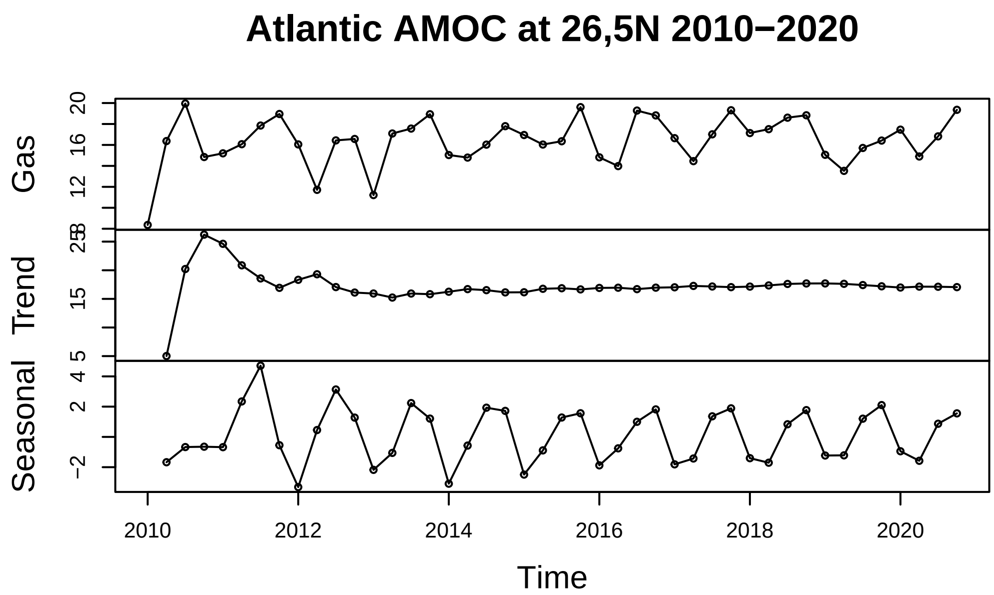
```
[1] 108.043
```

As we can see the dlm model using only a linear polynomial has a lower log likelihood than the model with an extra quadratic term, meaning this extra flexibility does not contribute to a better model fit and as such we will use the linear fitted model to do our forecasting.

```r
amocPredict = dlmFilter(tsAtlantic, mod = fittedLinearDLM)
summary(amocPredict)
```

```
    Length Class  Mode
y    44    ts     numeric
mod  10    dlm    list
m   225    mts    numeric
U.C  45    -none- list
D.C 225    -none- numeric
a   220    mts    numeric
U.R  44    -none- list
D.R 220    -none- numeric
f    44    ts     numeric
```

```r
x = cbind(tsAtlantic, dropFirst(amocPredict$a[, c(1, 3)]))
x = window(x, start = c(2010, 1))
colnames(x) = c("Gas", "Trend", "Seasonal")
plot(x, type = "o", main = "Atlantic AMOC at 26,5N 2010-2020")
```



Forecast

```
amocForecast = dlmForecast(amocPredict, nAhead = 4)
summary(amocForecast)
```

```
  Length Class  Mode
a 20     mts    numeric
R  4     -none- list
f  4     ts     numeric
Q  4     -none- list
```

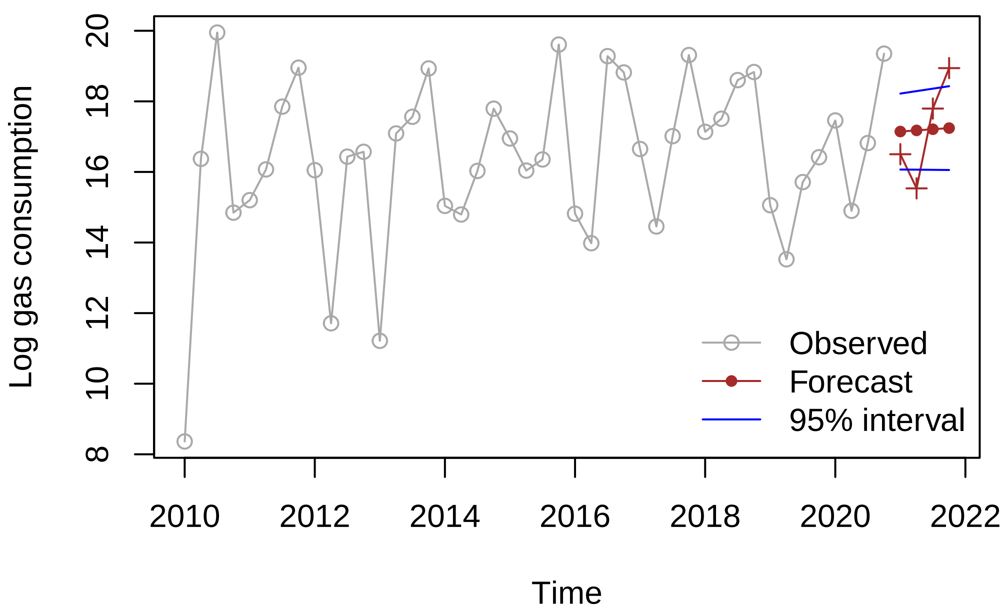
```
dim(amocForecast$a)
```

```
[1] 4 5
```

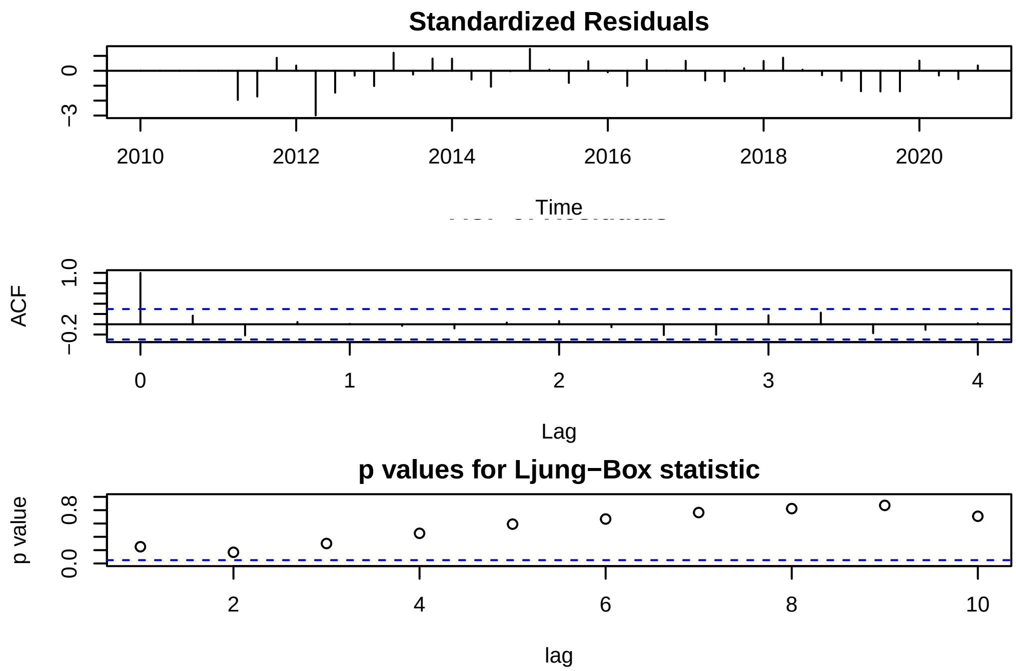
```
dim(amocForecast$f)
```

```
[1] 4 1
```

```r
sqrtR = sapply(amocForecast$R, function(x) sqrt(x[1, 1]))
pl = amocForecast$a[, 1] + qnorm(0.025, sd = sqrtR)
pu = amocForecast$a[, 1] + qnorm(0.975, sd = sqrtR)
x = ts.union(window(tsAtlantic, start = c(2010, 1)), amocForecast$a[, 1],
    amocForecast$f, pl, pu)
par(mar = c(4, 4, 2, 2))
plot(x, plot.type = "single", type = "o", pch = c(1, 20, 3, NA, NA), col = c("darkgrey",
    "brown", "brown", "blue", "blue"), ylab = "Log gas consumption")

legend("bottomright", legend = c("Observed", "Forecast", "95% interval"),
    bty = "n", pch = c(1, 20, NA), lty = 1, col = c("darkgrey", "brown",
        "blue"))
```




```r
# Set smaller margins
par(mar = c(4, 4, 2, 2))

tsdiag(amocPredict)
```

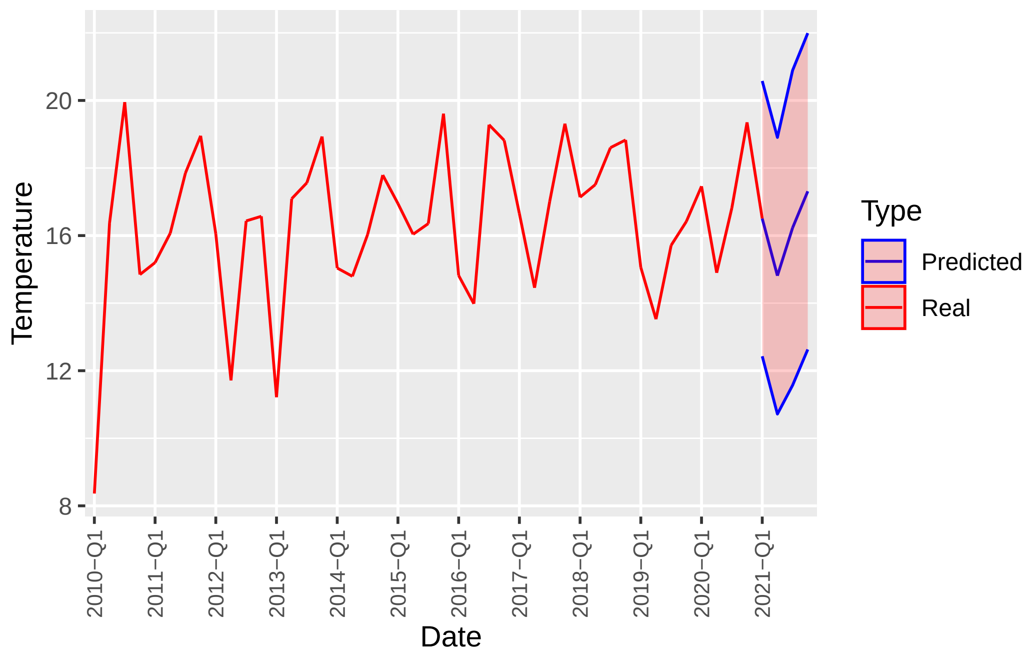



```
# Reset margins
par(mar = c(5, 4, 4, 2) + 0.1)
```

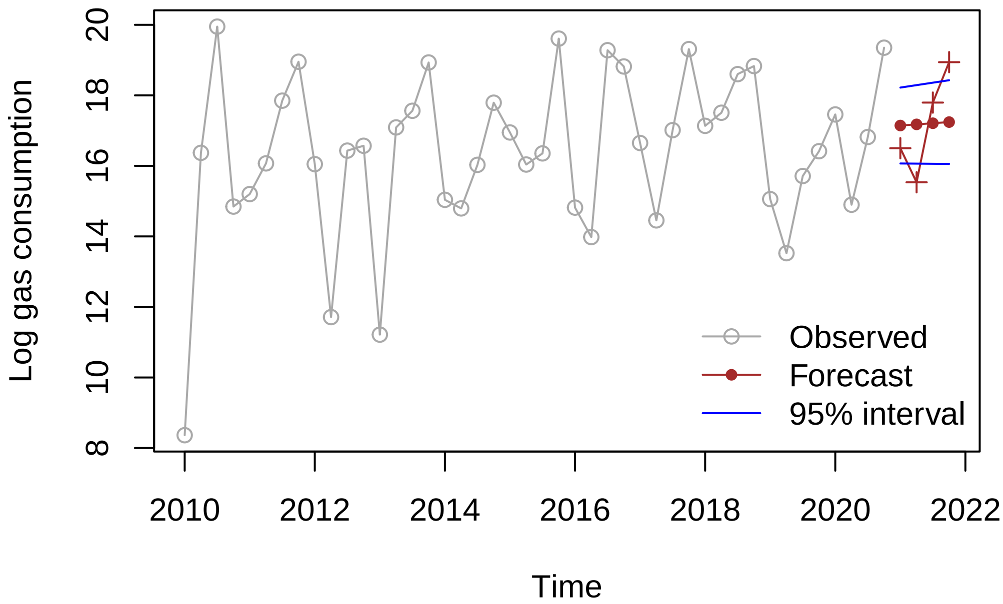
2 d)

Again comparing the forecast values and their respective prediction intervals as we can see from the graphs bellow the dlm model has smaller prediction intervals, most likely due to being able to explain the underlying seasonal trend reducing therefore the uncertainty in comparison the ARIMA model.

```
print(plotARIMA)
```




```r
sqaretRoot = sapply(amocForecast$R, function(x) sqrt(x[1, 1]))
predictionLow = amocForecast$a[, 1] + qnorm(0.025, sd = sqaretRoot)  ## Low
predictionUpper = amocForecast$a[, 1] + qnorm(0.975, sd = sqaretRoot)  ## Upper
x = ts.union(window(tsAtlantic, start = c(2010, 1)), amocForecast$a[, 1],
    amocForecast$f, predictionLow, predictionUpper)
par(mar = c(4, 4, 2, 2))
plot(x, plot.type = "single", type = "o", pch = c(1, 20, 3, NA, NA), col = c("darkgrey",
    "brown", "brown", "blue", "blue"), ylab = "Log gas consumption")

legend("bottomright", legend = c("Observed", "Forecast", "95% interval"),
    bty = "n", pch = c(1, 20, NA), lty = 1, col = c("darkgrey", "brown",
        "blue"))
```

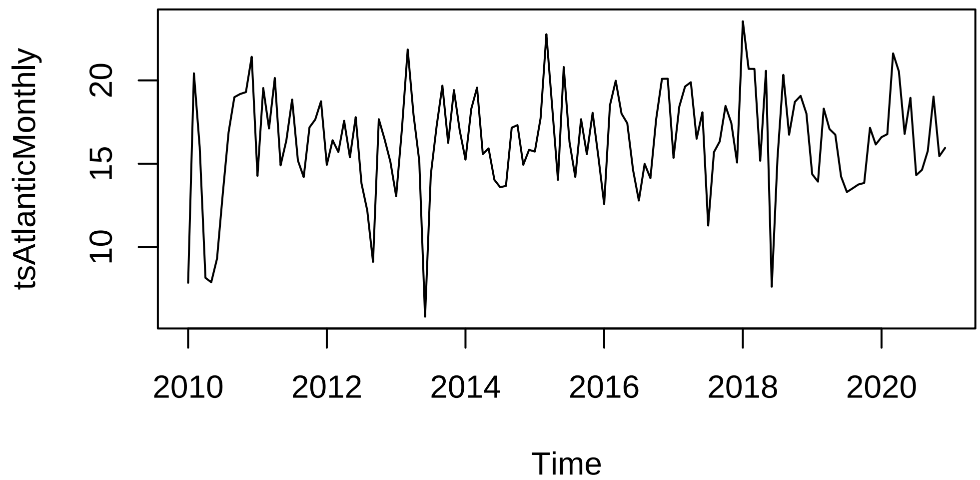


2 e)

```
# AMOCDFMonthly =AMOCDF %>% mutate(YearMonth = paste0(year(Date), '-',
# month(Date, label = TRUE, abbr = FALSE)))


## I will now make a column with the month and year that I will use to
## create the monthly averages
AMOCDF$YearMonth = paste(AMOCDF$Year, AMOCDF$Month, sep = "-")

YearMonthlyAverage = AMOCDF %>%
    group_by(YearMonth) %>%
    summarise(AverageStrength = mean(Strength))
```

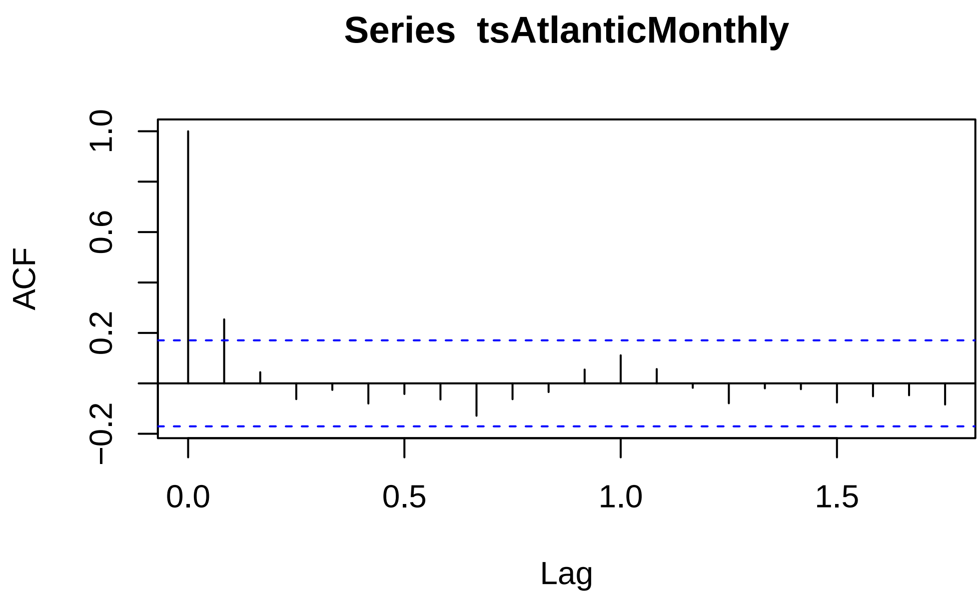
Now we will create a new montly time series object and make it univariate

```
tsAtlanticMonthly = ts(YearMonthlyAverage, start = c(2010, 1), frequency = 12)

tsAtlanticMonthly = tsAtlanticMonthly[, "AverageStrength"]

plot.ts(tsAtlanticMonthly)
```

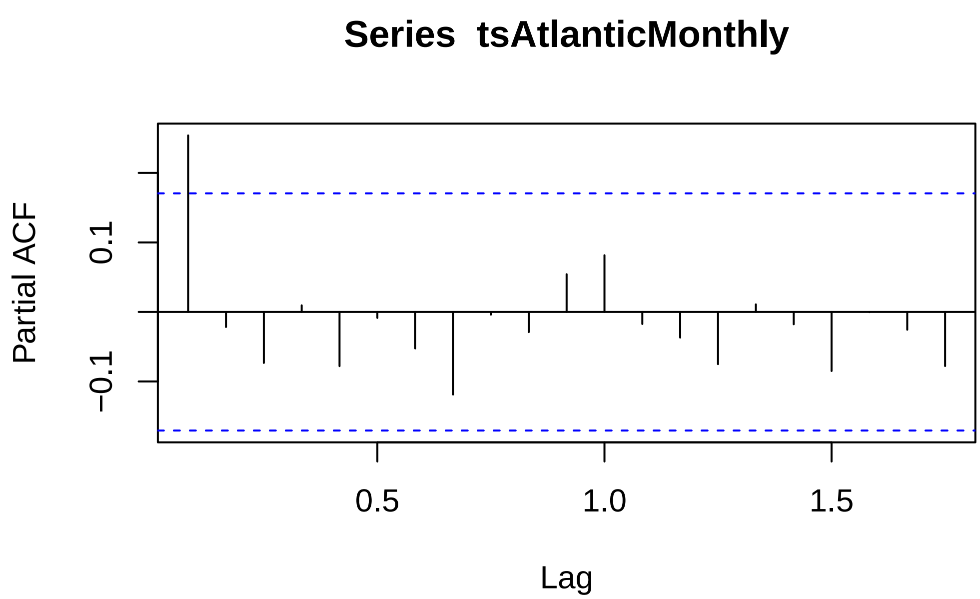


Seeing this graph we can observe that the data continues being stationary for the ARIMA model but a seasonal trend not only is more apparently but it also appear to need to be differentiated as it seems to have a decreasing linear trend

```
acf(tsAtlanticMonthly)
```




```
pacf(tsAtlanticMonthly)
```



The acf has a very clear cut-off as only 3 the values are significant which is very similar to what we had observed previously.

The main difference is in the pacf, where we can now say for sure that there is a very clear cut-off after the first value.

Model testing

These pattern suggests that an ARMA/ARIMA model might be the most appropriate so first we will check them out with the seasonal component of order 1, the so quick cut-off of both the ACF and PACF also might suggest that p and q will be smaller values.

Seasonal check

```
## initial assumption


modelMonthlySeasonal100.110 = Arima(tsAtlanticMonthly, order = c(1, 0, 0),
    seasonal = list(order = c(1, 1, 0), period = 12))

modelMonthlySeasonal100.011 = Arima(tsAtlanticMonthly, order = c(1, 0, 0),
    seasonal = list(order = c(0, 1, 1), period = 12))

modelMonthlySeasonal200.210 = Arima(tsAtlanticMonthly, order = c(2, 0, 0),
    seasonal = list(order = c(2, 1, 0), period = 12))
modelMonthlySeasonal200.012 = Arima(tsAtlanticMonthly, order = c(2, 0, 0),
    seasonal = list(order = c(0, 1, 2), period = 12))

modelMonthlySeasonal001.110 = Arima(tsAtlanticMonthly, order = c(0, 0, 1),
    seasonal = list(order = c(1, 1, 0), period = 12))

modelMonthlySeasonal001.011 = Arima(tsAtlanticMonthly, order = c(0, 0, 1),
    seasonal = list(order = c(0, 1, 1), period = 12))

modelMonthlySeasonal002.210 = Arima(tsAtlanticMonthly, order = c(0, 0, 2),
    seasonal = list(order = c(2, 1, 0), period = 12))
modelMonthlySeasonal002.012 = Arima(tsAtlanticMonthly, order = c(0, 0, 2),
    seasonal = list(order = c(0, 1, 2), period = 12))
```

```
modelMonthlySeasonal100.110
```

```
Series: tsAtlanticMonthly
ARIMA(1,0,0)(1,1,0)[12]

Coefficients:
         ar1     sar1
      0.1779  -0.4618
s.e.  0.0909   0.0839

sigma^2 = 12.06:  log likelihood = -320.1
AIC=646.2   AICc=646.4   BIC=654.56
```

```
modelMonthlySeasonal100.011
```

```
Series: tsAtlanticMonthly
ARIMA(1,0,0)(0,1,1)[12]

Coefficients:
         ar1     sma1
      0.1844  -0.8500
s.e.  0.0918   0.1123

sigma^2 = 8.74:  log likelihood = -306.88
AIC=619.76   AICc=619.97   BIC=628.12
```

```
modelMonthlySeasonal200.210
```

```
Series: tsAtlanticMonthly
ARIMA(2,0,0)(2,1,0)[12]

Coefficients:
        ar1     ar2     sar1     sar2
      0.115  0.0374  -0.7271  -0.4814
s.e.  0.094  0.0930   0.0932   0.0933

sigma^2 = 9.741:  log likelihood = -309.65
AIC=629.29   AICc=629.82   BIC=643.23
```

```
modelMonthlySeasonal200.012
```

```
Series: tsAtlanticMonthly
ARIMA(2,0,0)(0,1,2)[12]

Coefficients:
         ar1     ar2     sma1    sma2
      0.1566  0.0546  -0.9817  0.1895
s.e.  0.0947  0.0950   0.1384  0.1503

sigma^2 = 8.778:  log likelihood = -305.88
AIC=621.76   AICc=622.29   BIC=635.7
```

```
modelMonthlySeasonal001.110
```

```
Series: tsAtlanticMonthly
ARIMA(0,0,1)(1,1,0)[12]

Coefficients:
         ma1     sar1
      0.1680  -0.4634
s.e.  0.0861   0.0840

sigma^2 = 12.07:  log likelihood = -320.19
AIC=646.39   AICc=646.59   BIC=654.75
```

```
modelMonthlySeasonal001.011
```

```
Series: tsAtlanticMonthly
ARIMA(0,0,1)(0,1,1)[12]

Coefficients:
         ma1     sma1
      0.1606  -0.8446
s.e.  0.0847   0.1093

sigma^2 = 8.804:  log likelihood = -307.14
AIC=620.28   AICc=620.49   BIC=628.65
```

```
modelMonthlySeasonal002.210
```

```
Series: tsAtlanticMonthly
ARIMA(0,0,2)(2,1,0)[12]

Coefficients:
         ma1     ma2     sar1     sar2
      0.1145  0.0452  -0.7275  -0.4806
s.e.  0.0943  0.0882   0.0933   0.0936

sigma^2 = 9.745:  log likelihood = -309.66
AIC=629.33   AICc=629.85   BIC=643.26
```

```
modelMonthlySeasonal002.012
```

```
Series: tsAtlanticMonthly
ARIMA(0,0,2)(0,1,2)[12]

Coefficients:
         ma1     ma2     sma1    sma2
      0.1583  0.0745  -0.9786  0.1868
s.e.  0.0953  0.0893   0.1377  0.1495

sigma^2 = 8.784:  log likelihood = -305.89
AIC=621.78   AICc=622.3   BIC=635.72
```

So as suspected from both the time series plot and the last exercise analysis the added seasonality does increase our model goodness of fit while also penalising the increased in complexity with so far.

Now lets compare them to bigger p and q values to see if our initial assumptions do hold up

```
modelMonthlySeasonal301 = Arima(tsAtlanticMonthly, order = c(3, 0, 1), seasonal = list(ord
    1, 1), period = 12))
modelMonthlySeasonal302 = Arima(tsAtlanticMonthly, order = c(3, 0, 2), seasonal = list(ord
    1, 0), period = 12))
modelMonthlySeasonal303 = Arima(tsAtlanticMonthly, order = c(3, 1, 3), seasonal = list(ord
    1, 0), period = 12))

modelMonthlySeasonal103 = Arima(tsAtlanticMonthly, order = c(1, 0, 3), seasonal = list(ord
    1, 1), period = 12))
modelMonthlySeasonal203 = Arima(tsAtlanticMonthly, order = c(2, 1, 3), seasonal = list(ord
    1, 0), period = 12))
```

```r
modelMonthlySeasonal301
```

```
Series: tsAtlanticMonthly
ARIMA(3,0,1)(1,1,1)[12]

Coefficients:
          ar1     ar2      ar3     ma1     sar1     sma1
      -0.5728  0.1878  -0.0048  0.7442  -0.1158  -0.8073
s.e.   0.7595  0.1720   0.1369  0.7564   0.1224   0.1140

sigma^2 = 8.939:  log likelihood = -306.02
AIC=626.04   AICc=627.04   BIC=645.55
```

```r
modelMonthlySeasonal302
```

```
Series: tsAtlanticMonthly
ARIMA(3,0,2)(1,1,0)[12]

Coefficients:
          ar1      ar2     ar3     ma1     ma2     sar1
      -1.4367  -0.5902  0.1291  1.6983  1.0000  -0.4382
s.e.   0.0944   0.1532  0.0945  0.0436  0.0489   0.0871

sigma^2 = 11.34:  log likelihood = -316.59
AIC=647.19   AICc=648.19   BIC=666.7
```

```r
modelMonthlySeasonal303
```

```
Series: tsAtlanticMonthly
ARIMA(3,1,3)(1,1,0)[12]

Coefficients:
          ar1      ar2     ar3     ma1      ma2      ma3     sar1
      -1.4279  -0.5751  0.1373  0.6984  -0.6984  -1.0000  -0.4313
s.e.   0.0952   0.1547  0.0954  0.0533   0.0552   0.0576   0.0877

sigma^2 = 11.54:  log likelihood = -316.86
AIC=649.73   AICc=651.04   BIC=671.96
```

```r
modelMonthlySeasonal103
```

```
Series: tsAtlanticMonthly
ARIMA(1,0,3)(1,1,1)[12]

Coefficients:
          ar1     ma1     ma2      ma3     sar1     sma1
      -0.6951  0.8708  0.1931  -0.0013  -0.1134  -0.8029
s.e.   0.4854  0.4812  0.1503   0.1321   0.1194   0.1136

sigma^2 = 8.954:  log likelihood = -306.02
AIC=626.03   AICc=627.03   BIC=645.55
```

```r
modelMonthlySeasonal203
```

```
Series: tsAtlanticMonthly
ARIMA(2,1,3)(1,1,0)[12]

Coefficients:
          ar1      ar2     ma1      ma2      ma3     sar1
      -1.3087  -0.6017  0.5001  -0.6702  -0.8299  -0.4545
s.e.   0.2576   0.1557  0.2045   0.1332   0.1180   0.0979

sigma^2 = 12.02:  log likelihood = -318.1
AIC=650.19   AICc=651.2   BIC=669.65
```

As we can see here the initial assumption that a smaller p and q value would better fit the model.

Now we will use auto.arima to verify if our assumptions were indeed correct

auto arima check

```r
`?`(auto.arima)
```

```
starting httpd help server ... done
```

```r
monthlyModelAuto = auto.arima(tsAtlanticMonthly, max.d = 0, max.p = 5, max.q = 5,
    D = 1)
monthlyModelAuto
```

```
Series: tsAtlanticMonthly
ARIMA(1,0,0)(0,1,1)[12]

Coefficients:
         ar1     sma1
      0.1844  -0.8500
s.e.  0.0918   0.1123

sigma^2 = 8.74:  log likelihood = -306.88
AIC=619.76   AICc=619.97   BIC=628.12
```

From what we can see the auto arima has indeed confirmed our initial assumption by picking a model that we already had seen as the best performer ARIMA(1,0,0)(0,1,1)[12]

Forecasting

Now using the forecast function we will forecast the next 4 quarters of 2021

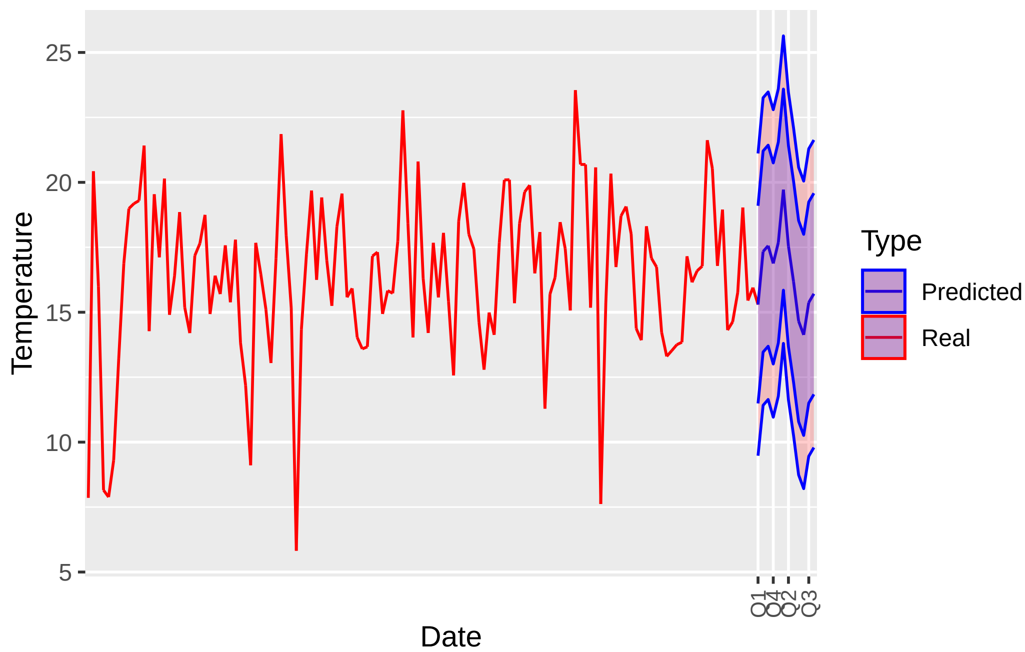
```r
forecast(modelMonthlySeasonal100.011, 12)
```

```
         Point Forecast    Lo 80    Hi 80     Lo 95    Hi 95
Jan 2021       15.29474 11.49064 19.09884  9.476872 21.11261
Feb 2021       17.68442 13.81636 21.55247 11.768741 23.60009
Mar 2021       19.72032 15.85011 23.59053 13.801347 25.63929
Apr 2021       17.53922 13.66894 21.40951 11.620140 23.45831
May 2021       16.17383 12.30355 20.04411 10.254741 22.09292
Jun 2021       14.65775 10.78746 18.52803  8.738657 20.57683
Jul 2021       14.12800 10.25772 17.99829  8.208914 20.04709
Aug 2021       15.37476 11.50447 19.24504  9.455670 21.29385
Sep 2021       15.71081 11.84052 19.58109  9.791718 21.62989
Oct 2021       17.33660 13.46632 21.20689 11.417514 23.25569
Nov 2021       17.56114 13.69086 21.43141 11.642055 23.48022
Dec 2021       16.87677 13.00663 20.74691 10.957906 22.79563
```

But this data is better visualized in a graph to better understand if the predictions are sensible compared to our real data.

```
predictedArimaSeasonalDF = data.frame(forecast(modelMonthlySeasonal100.011,
    12))

predictedArimaSeasonalDF$YearMonth = c("2021-1", "2021-2", "2021-3", "2021-4",
    "2021-5", "2021-6", "2021-7", "2021-8", "2021-9", "2021-10", "2021-11",
    "2021-12")

predictedArimaSeasonalDF$Type = "Predicted"

predictedArimaSeasonalDF$Temperature = predictedArimaSeasonalDF$Point.Forecast

YearMonthlyAverage$Type = "Real"
YearMonthlyAverage$Temperature = YearMonthlyAverage$AverageStrength


# Combine real_data and pred_data into a single data frame
combinedDataframeAMOC = rbind(data.frame(Date = YearMonthlyAverage$YearMonth,
    Temperature = YearMonthlyAverage$Temperature, Type = "Real"), data.frame(Date = predic
    Temperature = predictedArimaSeasonalDF$Temperature, Type = "Predicted"))
```

```
# Create the ggplot
plotARIMA2 = ggplot(combinedDataframeAMOC, aes(x = Date, y = Temperature,
    color = Type, group = 1)) + geom_line() + scale_color_manual(values = c("blue",
    "red"))

# Add the 80% and 95% confidence intervals
plotARIMA2 = plotARIMA2 + geom_ribbon(data = predictedArimaSeasonalDF, aes(x = YearMonth,
    ymin = Lo.95, ymax = Hi.95), fill = "red", alpha = 0.2) + geom_ribbon(data = predicted
    aes(x = YearMonth, ymin = Lo.80, ymax = Hi.80), fill = "blue", alpha = 0.2)

# Adjust the x-axis labels
plotARIMA2 = plotARIMA2 + scale_x_discrete(breaks = c("2021-1", "2021-4",
    "2021-8", "2021-12"), labels = c("Q1", "Q2", "Q3", "Q4"))

plotARIMA2 = plotARIMA2 + theme(axis.text.x = element_text(angle = 90, vjust = 0.5,
    size = 8))

# Display the plot
print(plotARIMA2)
```



DLM

model fitting

```
## linear model, order = 2, quadratic order = 3 , etc

## what we want is a linear model with a seasonal component so we add
## the 2 components together in a model

## things to try, another term like quadratic, or a arma component
## stacked on top of this

## Initial model with a linear polynomial and a seasonal component

buildFun = function(x) {
    dlmModPoly(order = 2, dV = exp(x[1]), dW = c(0, exp(x[2]))) + dlmModSeas(frequency = 1
        dV = 0, dW = c(exp(x[3]), rep(0, 10)))
}

linearDLM = dlmMLE(tsAtlanticMonthly, parm = c(0, 0, 0), build = buildFun)
```

```r
linearDLM$par
```

```
[1]   2.044026 -11.586366  -4.421181
```

```r
fittedLinearDLM = buildFun(linearDLM$par)
V(fittedLinearDLM)
```

```
         [,1]
[1,] 7.721632
```

```r
W(fittedLinearDLM)
```

```
      [,1]         [,2]       [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
 [1,]    0 0.000000e+00 0.00000000    0    0    0    0    0    0     0     0
 [2,]    0 9.291914e-06 0.00000000    0    0    0    0    0    0     0     0
 [3,]    0 0.000000e+00 0.01202003    0    0    0    0    0    0     0     0
 [4,]    0 0.000000e+00 0.00000000    0    0    0    0    0    0     0     0
 [5,]    0 0.000000e+00 0.00000000    0    0    0    0    0    0     0     0
 [6,]    0 0.000000e+00 0.00000000    0    0    0    0    0    0     0     0
 [7,]    0 0.000000e+00 0.00000000    0    0    0    0    0    0     0     0
 [8,]    0 0.000000e+00 0.00000000    0    0    0    0    0    0     0     0
 [9,]    0 0.000000e+00 0.00000000    0    0    0    0    0    0     0     0
[10,]    0 0.000000e+00 0.00000000    0    0    0    0    0    0     0     0
[11,]    0 0.000000e+00 0.00000000    0    0    0    0    0    0     0     0
[12,]    0 0.000000e+00 0.00000000    0    0    0    0    0    0     0     0
[13,]    0 0.000000e+00 0.00000000    0    0    0    0    0    0     0     0
      [,12] [,13]
 [1,]     0     0
 [2,]     0     0
 [3,]     0     0
 [4,]     0     0
 [5,]     0     0
 [6,]     0     0
 [7,]     0     0
 [8,]     0     0
 [9,]     0     0
[10,]     0     0
[11,]     0     0
```

```
[12,]     0     0
[13,]     0     0
```

```r
## second model with a quadratic polynomial and a seasonal component

buildFunQuad = function(x) {
    dlmModPoly(order = 3, dV = exp(x[1]), dW = c(0, exp(x[2]), exp(x[3]))) +
        dlmModSeas(frequency = 12, dV = 0, dW = c(exp(x[4]), rep(0, 10)))
}

quadraticDLM = dlmMLE(tsAtlanticMonthly, parm = c(0, 0, 0, 0), build = buildFunQuad)

quadraticDLM$par
```

```
[1]   2.047135 -21.060474 -56.104784 -12.300531
```

```r
fittedQuadraticDLM = buildFunQuad(quadraticDLM$par)

V(fittedQuadraticDLM)
```

```
         [,1]
[1,] 7.745678
```

```r
W(fittedQuadraticDLM)
```

```
      [,1]         [,2]         [,3]         [,4] [,5] [,6] [,7] [,8] [,9]
 [1,]    0 0.000000e+00 0.000000e+00 0.000000e+00    0    0    0    0    0
 [2,]    0 7.137603e-10 0.000000e+00 0.000000e+00    0    0    0    0    0
 [3,]    0 0.000000e+00 4.305286e-25 0.000000e+00    0    0    0    0    0
 [4,]    0 0.000000e+00 0.000000e+00 4.549326e-06    0    0    0    0    0
 [5,]    0 0.000000e+00 0.000000e+00 0.000000e+00    0    0    0    0    0
 [6,]    0 0.000000e+00 0.000000e+00 0.000000e+00    0    0    0    0    0
 [7,]    0 0.000000e+00 0.000000e+00 0.000000e+00    0    0    0    0    0
 [8,]    0 0.000000e+00 0.000000e+00 0.000000e+00    0    0    0    0    0
 [9,]    0 0.000000e+00 0.000000e+00 0.000000e+00    0    0    0    0    0
[10,]    0 0.000000e+00 0.000000e+00 0.000000e+00    0    0    0    0    0
[11,]    0 0.000000e+00 0.000000e+00 0.000000e+00    0    0    0    0    0
[12,]    0 0.000000e+00 0.000000e+00 0.000000e+00    0    0    0    0    0
```

```
[13,]    0 0.000000e+00 0.000000e+00 0.000000e+00    0    0    0    0    0
[14,]    0 0.000000e+00 0.000000e+00 0.000000e+00    0    0    0    0    0
      [,10] [,11] [,12] [,13] [,14]
 [1,]     0     0     0     0     0
 [2,]     0     0     0     0     0
 [3,]     0     0     0     0     0
 [4,]     0     0     0     0     0
 [5,]     0     0     0     0     0
 [6,]     0     0     0     0     0
 [7,]     0     0     0     0     0
 [8,]     0     0     0     0     0
 [9,]     0     0     0     0     0
[10,]     0     0     0     0     0
[11,]     0     0     0     0     0
[12,]     0     0     0     0     0
[13,]     0     0     0     0     0
[14,]     0     0     0     0     0
```

Now we will compare both models through their log likelihood using the dlmLL function and see if the extra flexibility from the extra polynomial function is providing a better fit

```r
dlmLL(tsAtlanticMonthly, fittedLinearDLM)
```

```
[1] 309.5446
```

```r
dlmLL(tsAtlanticMonthly, fittedQuadraticDLM)
```

```
[1] 324.4748
```

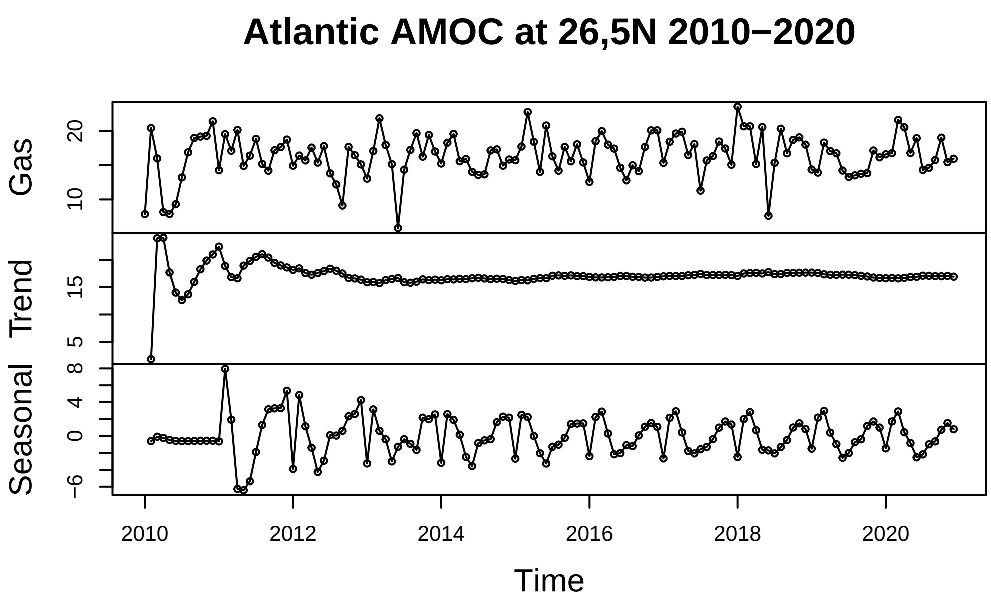
As we can see the dlm model using only a linear polynomial has a lower log likelihood than the model with an extra quadratic term, meaning this extra flexibility does not contribute to a better model fit and as such we will use the linear fitted model to do our forecasting.

```r
amocPredict = dlmFilter(tsAtlanticMonthly, mod = fittedLinearDLM)
summary(amocPredict)
```

```
    Length Class  Mode
y   132    ts     numeric
mod  10    dlm    list
```

```
m   1729   mts    numeric
U.C  133   -none- list
D.C 1729   -none- numeric
a   1716   mts    numeric
U.R  132   -none- list
D.R 1716   -none- numeric
f    132   ts     numeric
```

```r
x = cbind(tsAtlanticMonthly, dropFirst(amocPredict$a[, c(1, 3)]))
x = window(x, start = c(2010, 1))
colnames(x) = c("Gas", "Trend", "Seasonal")
plot(x, type = "o", main = "Atlantic AMOC at 26,5N 2010-2020")
```



Forecast

```r
amocForecastMonthly = dlmForecast(amocPredict, nAhead = 12)
summary(amocForecastMonthly)
```

```
  Length Class  Mode
a 156    mts    numeric
```

```
R  12    -none- list
f  12    ts     numeric
Q  12    -none- list
```

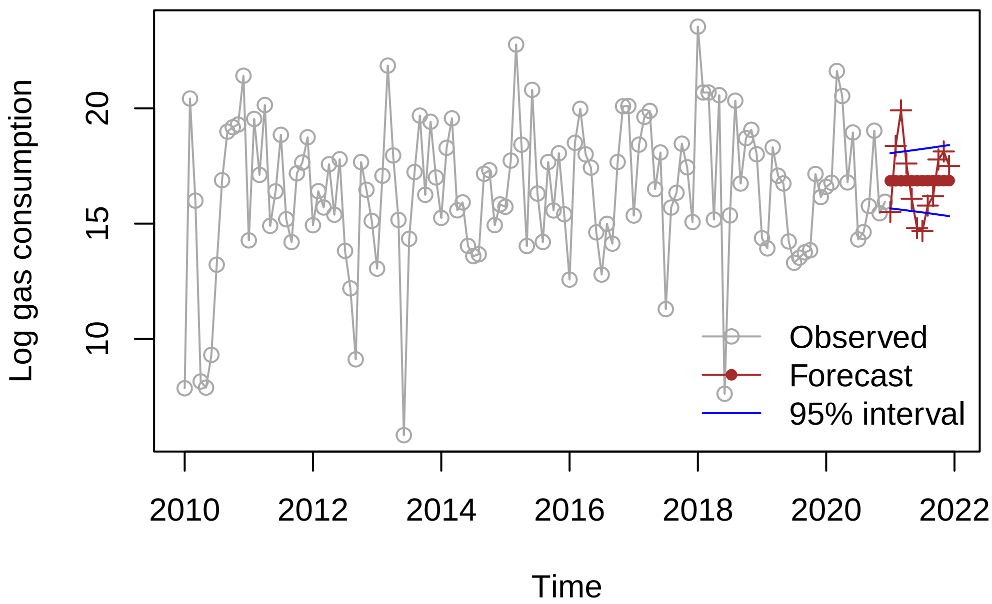
```r
dim(amocForecastMonthly$a)
```

```
[1] 12 13
```

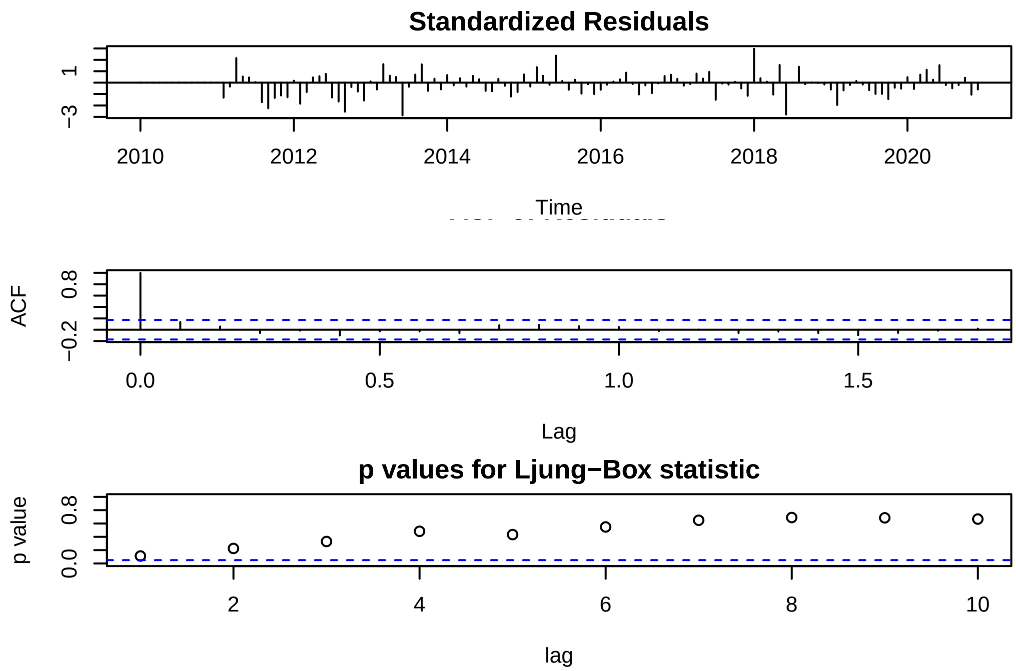
```r
dim(amocForecastMonthly$f)
```

```
[1] 12  1
```

```r
sqrtR = sapply(amocForecastMonthly$R, function(x) sqrt(x[1, 1]))
pl = amocForecastMonthly$a[, 1] + qnorm(0.025, sd = sqrtR)
pu = amocForecastMonthly$a[, 1] + qnorm(0.975, sd = sqrtR)

x = ts.union(window(tsAtlanticMonthly, start = c(2010, 1)), amocForecastMonthly$a[,
    1], amocForecastMonthly$f, pl, pu)
par(mar = c(4, 4, 2, 2))
plot(x, plot.type = "single", type = "o", pch = c(1, 20, 3, NA, NA), col = c("darkgrey",
    "brown", "brown", "blue", "blue"), ylab = "Log gas consumption")

legend("bottomright", legend = c("Observed", "Forecast", "95% interval"),
    bty = "n", pch = c(1, 20, NA), lty = 1, col = c("darkgrey", "brown",
        "blue"))
```




```
# Set smaller margins
par(mar = c(4, 4, 2, 2))

tsdiag(amocPredict)
```




```r
# Reset margins
par(mar = c(5, 4, 4, 2) + 0.1)
```

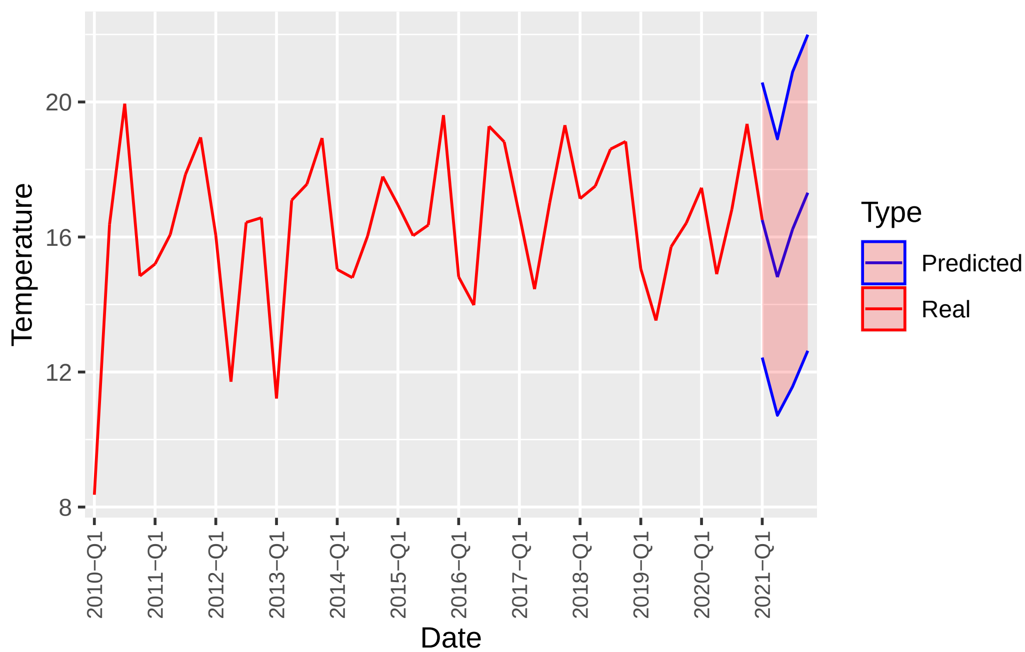
Lastly checking the residuals, they seem to be mostly normally distributed with a good mixture of over and under estimations, especially in the middle with some slight seasonality on both ends being present

2 f)

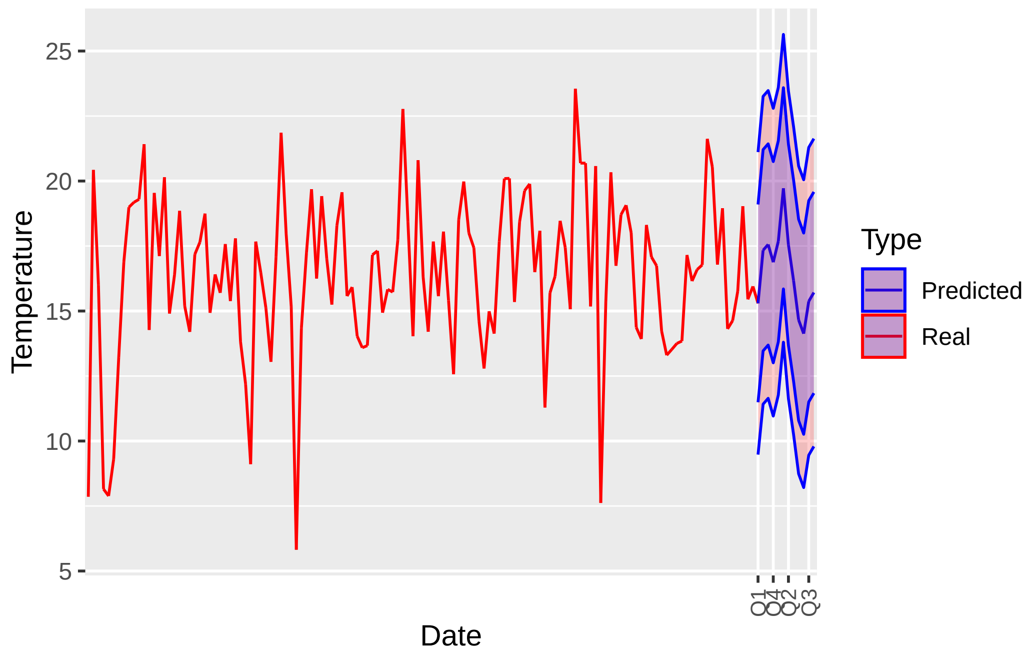
Now starting with the ARIMA models

```r
predictedArimaDF = data.frame(forecast(model200, 4))

predictedArimaDF$YearQuarter = c("2021-Q1", "2021-Q2", "2021-Q3", "2021-Q4")

# Combine real_data and pred_data into a single data frame
combinedDataframeAMOC = rbind(data.frame(Date = YearQuarterAverage$YearQuarter,
    Temperature = YearQuarterAverage$AverageStrength, Type = "Real"), data.frame(Date = pr
    Temperature = predictedArimaDF$Point.Forecast, Type = "Predicted"))

predictedArimaDF$Temperature = predictedArimaDF$Point.Forecast
```

```
predictedArimaDF$Type = "Predicted"


# Create the ggplot
plotARIMA = ggplot(combinedDataframeAMOC, aes(x = Date, y = Temperature,
    color = Type, group = 1)) + geom_line() + scale_color_manual(values = c("blue",
    "red"))

# Add the 95% confidence interval
plotARIMA = plotARIMA + geom_ribbon(data = predictedArimaDF, aes(x = YearQuarter,
    ymin = Lo.95, ymax = Hi.95), fill = "red", alpha = 0.2)

# Adjust the x-axis labels
plotARIMA = plotARIMA + scale_x_discrete(breaks = combinedDataframeAMOC$Date[c(TRUE,
    rep(FALSE, 3))], labels = combinedDataframeAMOC$Date[c(TRUE, rep(FALSE,
    3))])

plotARIMA = plotARIMA + theme(axis.text.x = element_text(angle = 90, vjust = 0.5,
    size = 8))

print(plotARIMA)
```



```r
print(plotARIMA2)
```



The most obvious difference is the level of detail on the seasonality, with the monthly averages capturing an almost opposite effect than the quarterly averages in both the predicted and also in some of the real data, the forecast also has the opposite trend, with an actual expected decrease in Sverdrups around between the 2nd quarter and the mid 3rd quarter which again seems to follow the opposite trend on the quartely data.

```r
sqrtR = sapply(amocForecastMonthly$R, function(x) sqrt(x[1, 1]))
pl = amocForecastMonthly$a[, 1] + qnorm(0.025, sd = sqrtR)
pu = amocForecastMonthly$a[, 1] + qnorm(0.975, sd = sqrtR)

x = ts.union(window(tsAtlanticMonthly, start = c(2010, 1)), amocForecastMonthly$a[,
    1], amocForecastMonthly$f, pl, pu)
par(mar = c(4, 4, 2, 2))
plot(x, plot.type = "single", type = "o", pch = c(1, 20, 3, NA, NA), col = c("darkgrey",
    "brown", "brown", "blue", "blue"), ylab = "Log gas consumption")

legend("bottomright", legend = c("Observed", "Forecast", "95% interval"),
    bty = "n", pch = c(1, 20, NA), lty = 1, col = c("darkgrey", "brown",
        "blue"))
```

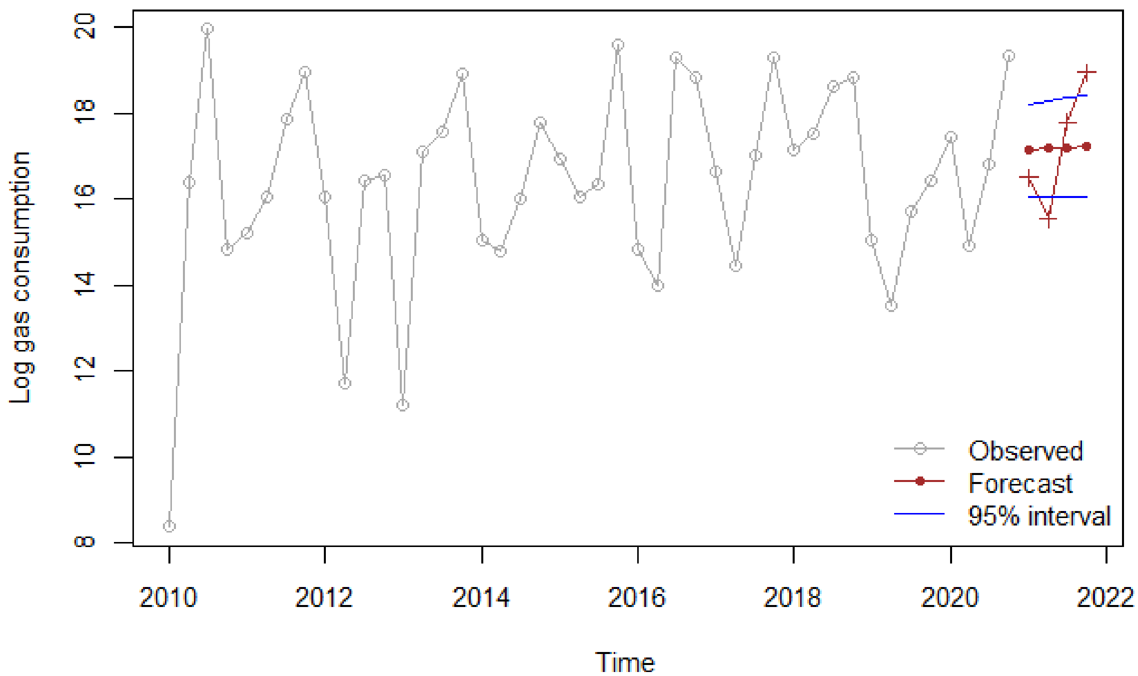
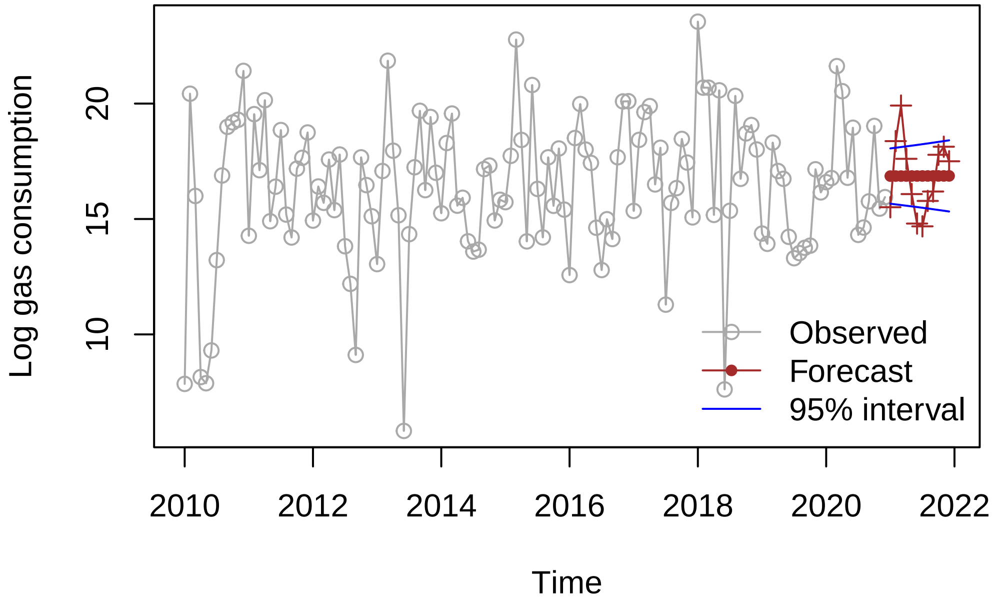


Figure 1: 1st dlm



Observing the 2 graphs of the dlm models we can see that although the quarterly predictions do not completly predict the spike during the first half of the year compared to the monthly data the second half of the year seems to be relative similarly forecasted with the exception of December where the monthly data show again a decrease but the quarterly data is not capable of capturing.

Despise these changes the predicted overall trend is quite similar with the quaterly trend very slighty increasing the monthly data seeming to remaind constant.

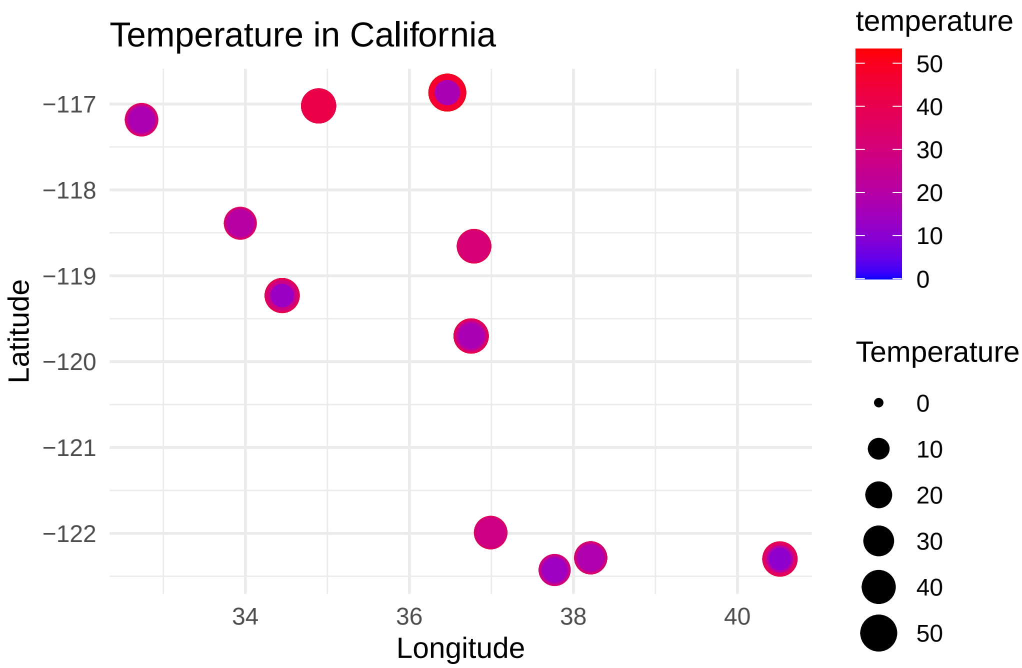
Question 3

Question 3 a)

my approach is to see the max temp in the entire state with 8 cities

```
californiaLongTempDF = pivot_longer(californiaTempDF, cols = -Date, names_to = "Location",
    values_to = "Temperature")

spatialTemperatureCaliforniaDF = merge(californiaLongTempDF, californiaSpatialDataDF)


ggplot(data = spatialTemperatureCaliforniaDF) + geom_point(aes(x = Lat, y = Long,
```

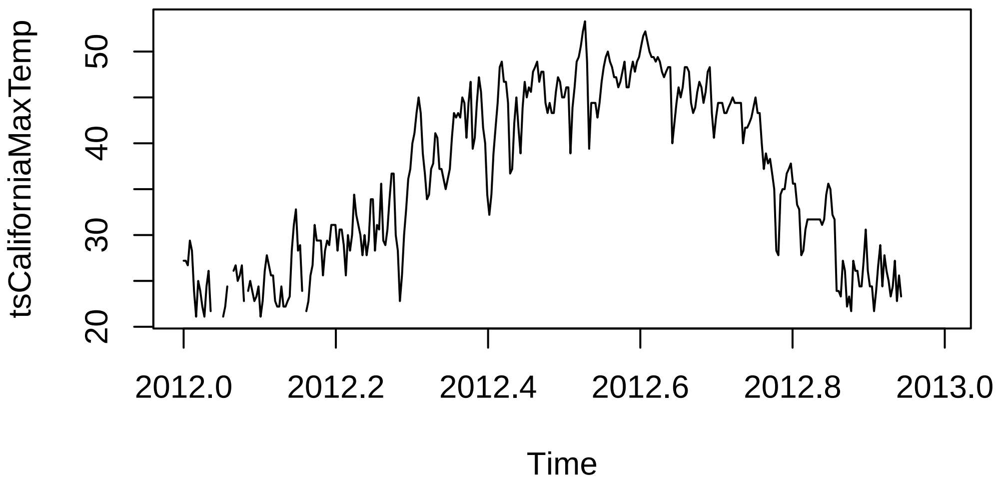
```
    color = Temperature, size = Temperature)) + scale_color_continuous(low = "blue",
    high = "red") + labs(title = "Temperature in California", x = "Longitude",
    y = "Latitude", color = "temperature") + theme_minimal()
```




```
californiaTempDF$Date = as.Date(as.character(californiaTempDF$Date), format = "%Y%m%d",
    origin = "1970-01-01")

californiaTempDF$max <- apply(californiaTempDF, 1, max, na.rm = TRUE)


tsCaliforniaMaxTemp = ts(californiaTempDF$max, start = c(2012, 1), frequency = 366)
```

```
plot.ts(tsCaliforniaMaxTemp)
```

```
Warning in xy.coords(x, NULL, log = log, setLab = FALSE): NAs introduced by
coercion

Warning in xy.coords(x, y): NAs introduced by coercion
```



As we can see from the time series trend, august to September seems to be the hottest months while january to feburary seems to be the coldest months in the californian state.

3 b)

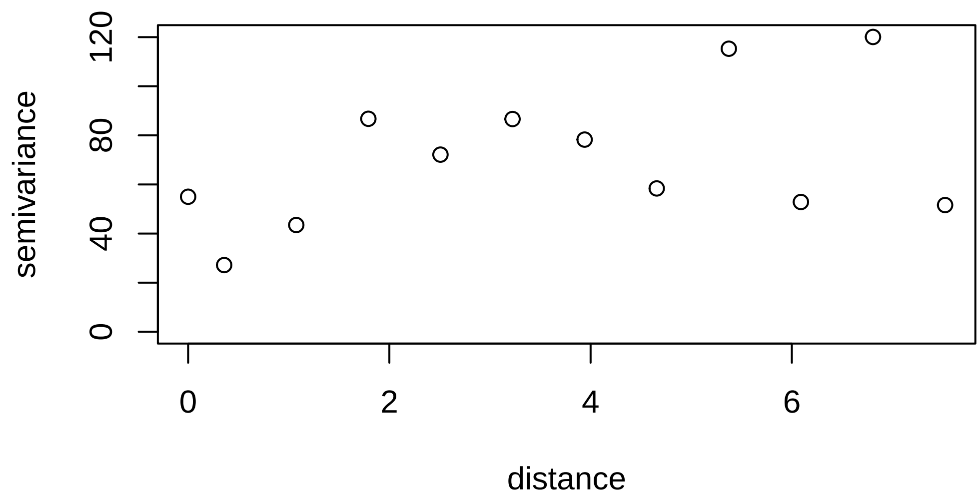
```
geoDataCalifornia = as.geodata(spatialTemperatureCaliforniaDF, coords.col = 4:5,
    data.col = "Temperature", covar.col = "Elev")
```

```
as.geodata: 4004 replicated data locations found.
 Consider using jitterDupCoords() for jittering replicated locations.
WARNING: there are data at coincident or very closed locations, some of the geoR's functions
 Use function dup.coords() to locate duplicated coordinates.
 Consider using jitterDupCoords() for jittering replicated locations
```

```
variogramCalifornia = variog(geoDataCalifornia)
```

```
variog: computing omnidirectional variogram
variog: co-locatted data found, adding one bin at the origin
```

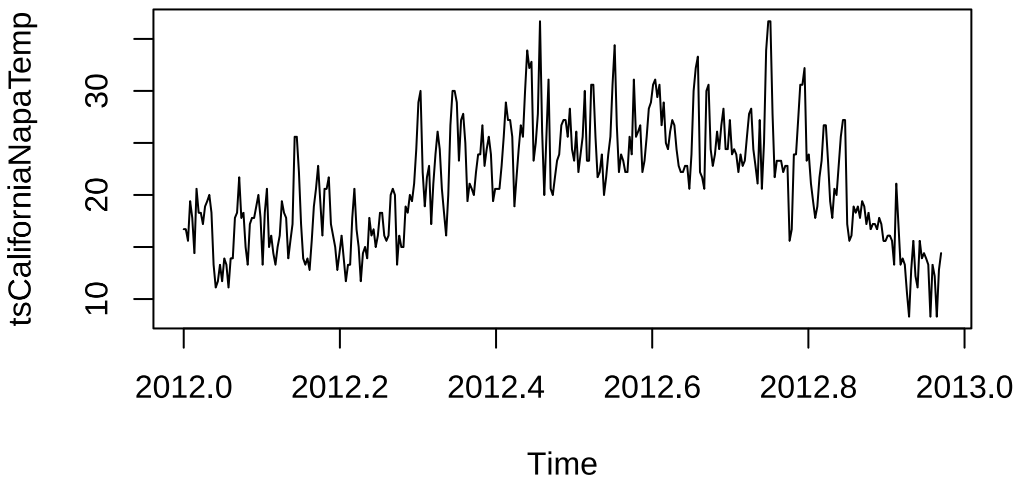
```r
plot(variogramCalifornia)
```



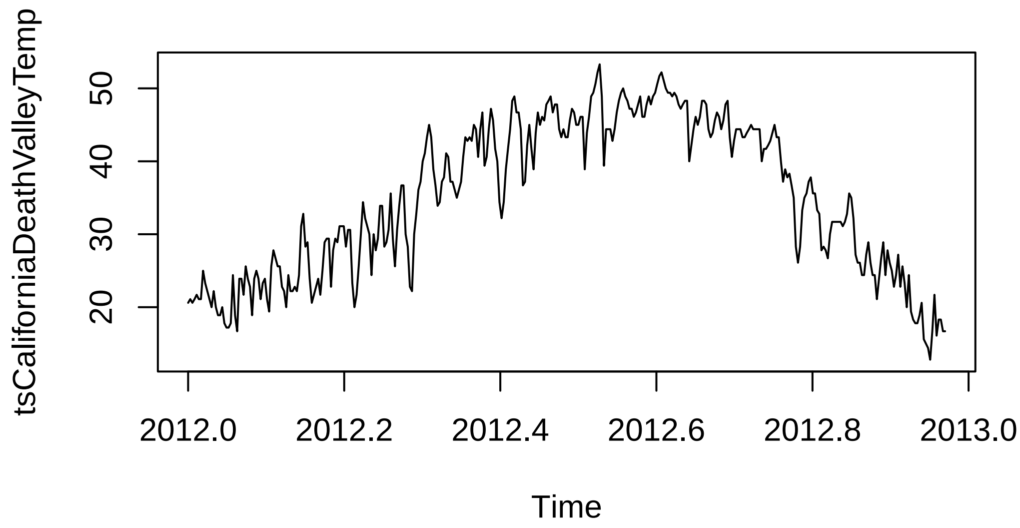
3 c)

Here we create a ts object with frequency 366 because 2012 does indeed have the 29th of February, removing the dates from the 9th to the 17th of november

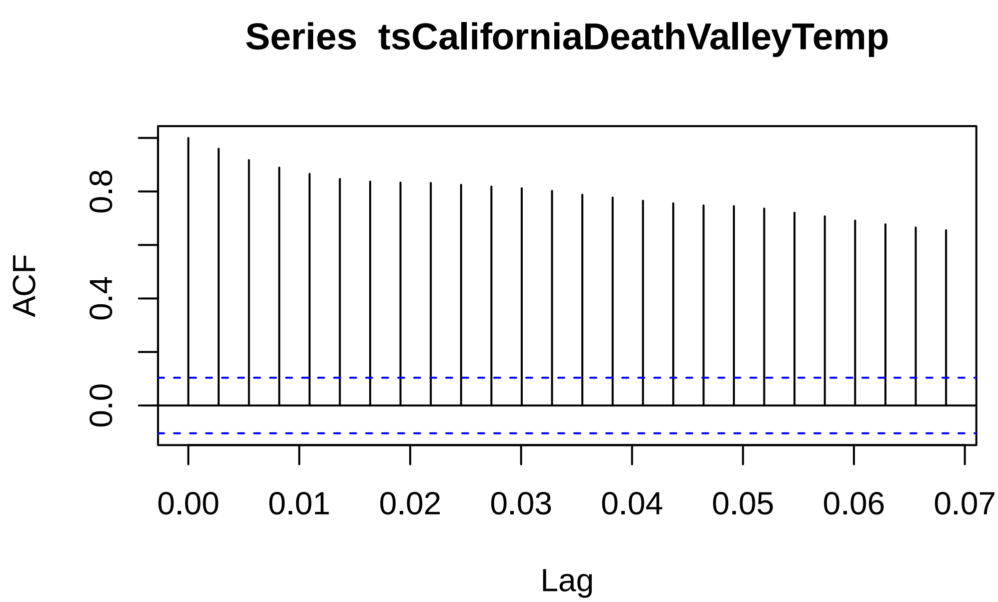
```r
toPredictValues = californiaTempDF %>%
    filter((Date >= as.Date("2012-11-09") & Date <= as.Date("2012-11-17")))

californiaTempDF = californiaTempDF %>%
    filter(!(Date >= as.Date("2012-11-09") & Date <= as.Date("2012-11-17")))


tsCaliforniaNapaTemp = ts(californiaTempDF$Napa, start = c(2012, 1), frequency = 366)

plot.ts(tsCaliforniaNapaTemp)
```



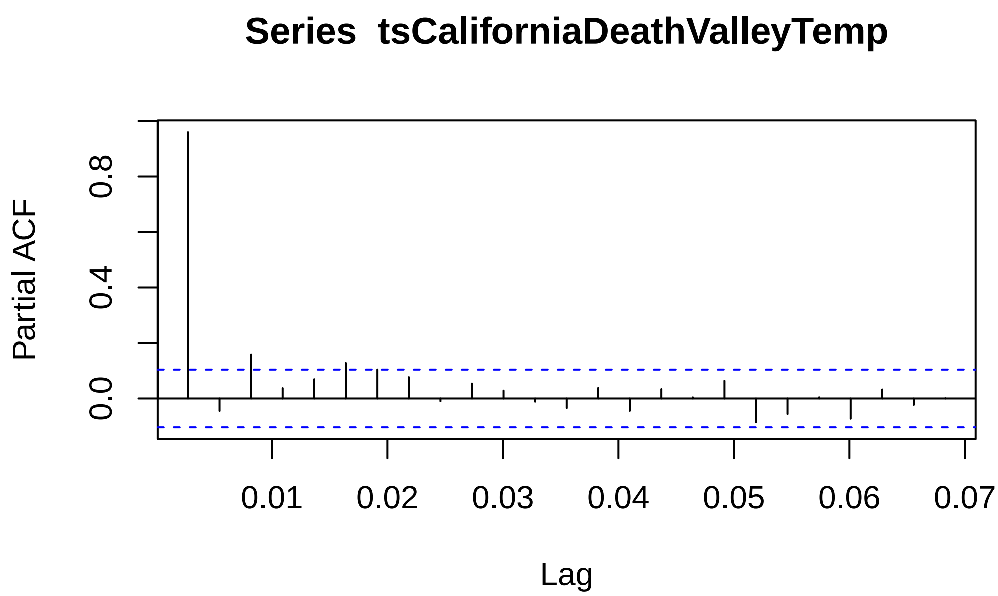

```
tsCaliforniaDeathValleyTemp = ts(californiaTempDF$`Death Valley`, start = c(2012,
    1), frequency = 366)

plot.ts(tsCaliforniaDeathValleyTemp)
```




```
acf(tsCaliforniaDeathValleyTemp)
```



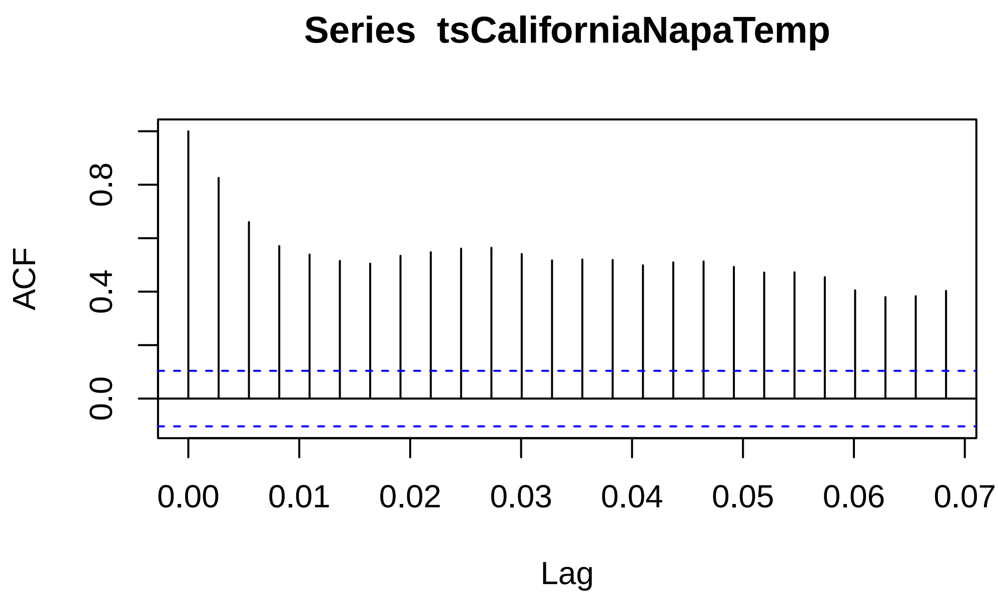
```r
pacf(tsCaliforniaDeathValleyTemp)
```



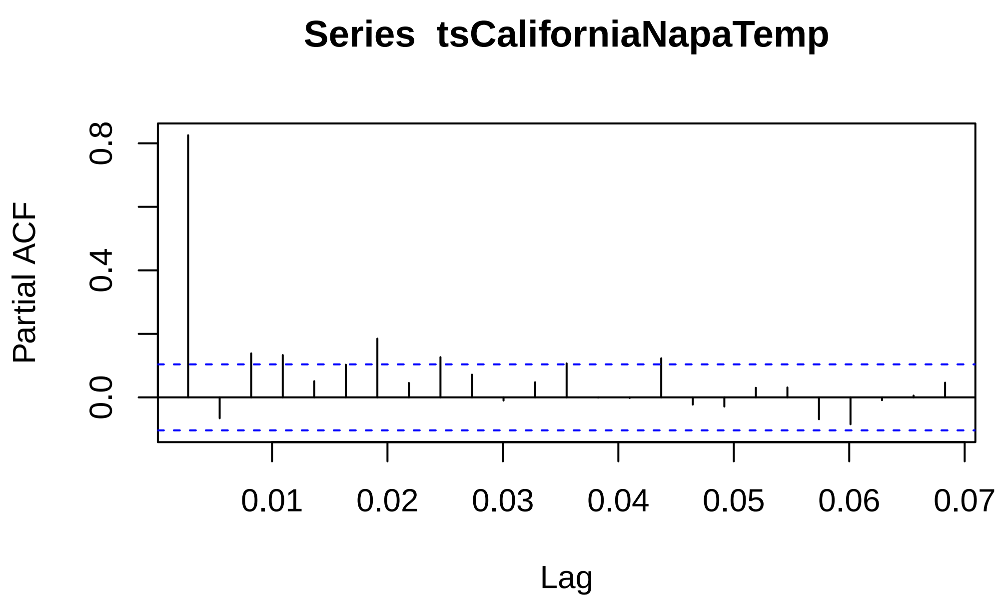
In death valley we can clearly see that ACF is very slowly descending without any cut-off and PAC seems to have a very quick cut off, this means that the best model will most likely will be AR with a possibly larger p value

Now checking for Napa

```r
acf(tsCaliforniaNapaTemp)
```




```
pacf(tsCaliforniaNapaTemp)
```



We can once again see the same pattern of a very slowly decreasing ACF but this time PACF does seem to at least not has such a clear cut off meanig there might a smaller q non-zero q value requiring an ARIMA for this city

The model for both cities does seem stationary with a linear trend on seasonality however it is not clear enough since we only have 1 year worth of data

model checking Nappa

```
## initial assumption


modelMonthlySeasonal100.110Napa = Arima(tsCaliforniaNapaTemp, order = c(1,
    0, 0), seasonal = list(order = c(1, 1, 0), period = 12))

modelMonthlySeasonal100.011Napa = Arima(tsCaliforniaNapaTemp, order = c(1,
    0, 0), seasonal = list(order = c(0, 1, 1), period = 12))

modelMonthlySeasonal200.210Napa = Arima(tsCaliforniaNapaTemp, order = c(2,
    0, 0), seasonal = list(order = c(2, 1, 0), period = 12))
modelMonthlySeasonal200.012Napa = Arima(tsCaliforniaNapaTemp, order = c(2,
    0, 0), seasonal = list(order = c(0, 1, 2), period = 12))

modelMonthlySeasonal001.110Napa = Arima(tsCaliforniaNapaTemp, order = c(0,
    0, 1), seasonal = list(order = c(1, 1, 0), period = 12))

modelMonthlySeasonal001.011Napa = Arima(tsCaliforniaNapaTemp, order = c(0,
    0, 1), seasonal = list(order = c(0, 1, 1), period = 12))

modelMonthlySeasonal002.210Napa = Arima(tsCaliforniaNapaTemp, order = c(0,
    0, 2), seasonal = list(order = c(2, 1, 0), period = 12))
modelMonthlySeasonal002.012Napa = Arima(tsCaliforniaNapaTemp, order = c(0,
    0, 2), seasonal = list(order = c(0, 1, 2), period = 12))
```

```
modelMonthlySeasonal100.110Napa
```

```
Series: tsCaliforniaNapaTemp
ARIMA(1,0,0)(1,1,0)[12]

Coefficients:
         ar1     sar1
```

```
      0.6323  -0.4915
s.e.  0.0418   0.0474

sigma^2 = 14.03:  log likelihood = -943.36
AIC=1892.73   AICc=1892.8   BIC=1904.25
```

```
modelMonthlySeasonal100.011Napa
```

```
Series: tsCaliforniaNapaTemp
ARIMA(1,0,0)(0,1,1)[12]

Coefficients:
         ar1     sma1
      0.8307  -0.9637
s.e.  0.0329   0.0780

sigma^2 = 10.12:  log likelihood = -900.8
AIC=1807.61   AICc=1807.68   BIC=1819.13
```

```
modelMonthlySeasonal200.210Napa
```

```
Series: tsCaliforniaNapaTemp
ARIMA(2,0,0)(2,1,0)[12]

Coefficients:
         ar1      ar2     sar1     sar2
      0.7592  -0.1464  -0.6312  -0.2890
s.e.  0.0535   0.0540   0.0539   0.0534

sigma^2 = 12.47:  log likelihood = -923.09
AIC=1856.19   AICc=1856.36   BIC=1875.39
```

```
modelMonthlySeasonal200.012Napa
```

```
Series: tsCaliforniaNapaTemp
ARIMA(2,0,0)(0,1,2)[12]

Coefficients:
         ar1      ar2     sma1    sma2
```

```
      0.8807  -0.0576  -0.9749  0.0349
s.e.  0.0546   0.0550   0.0730  0.0634

sigma^2 = 10.26:  log likelihood = -900.05
AIC=1810.11   AICc=1810.29   BIC=1829.31
```

```
modelMonthlySeasonal001.110Napa
```

```
Series: tsCaliforniaNapaTemp
ARIMA(0,0,1)(1,1,0)[12]

Coefficients:
         ma1     sar1
      0.5787  -0.4244
s.e.  0.0365   0.0493

sigma^2 = 15.15:  log likelihood = -955.99
AIC=1917.98   AICc=1918.05   BIC=1929.5
```

```
modelMonthlySeasonal001.011Napa
```

```
Series: tsCaliforniaNapaTemp
ARIMA(0,0,1)(0,1,1)[12]

Coefficients:
         ma1     sma1
      0.6150  -0.5909
s.e.  0.0358   0.0459

sigma^2 = 13.72:  log likelihood = -940.43
AIC=1886.85   AICc=1886.92   BIC=1898.37
```

```
modelMonthlySeasonal002.210Napa
```

```
Series: tsCaliforniaNapaTemp
ARIMA(0,0,2)(2,1,0)[12]

Coefficients:
         ma1     ma2     sar1     sar2
```

```
      0.7380  0.3614  -0.6225  -0.2583
s.e.  0.0488  0.0552   0.0561   0.0531

sigma^2 = 12.66:  log likelihood = -925.48
AIC=1860.95   AICc=1861.13   BIC=1880.16
```

```r
modelMonthlySeasonal002.012Napa
```

```
Series: tsCaliforniaNapaTemp
ARIMA(0,0,2)(0,1,2)[12]

Coefficients:
         ma1     ma2     sma1    sma2
      0.7903  0.3957  -0.7285  0.0584
s.e.  0.0494  0.0537   0.0609  0.0608

sigma^2 = 11.87:  log likelihood = -915.85
AIC=1841.71   AICc=1841.88   BIC=1860.91
```

So far it seems the second model is the most adequate

```r
NapaModelAuto = auto.arima(tsCaliforniaNapaTemp, max.d = 0, max.p = 5, max.q = 5)

NapaModelAuto
```

```
Series: tsCaliforniaNapaTemp
ARIMA(1,0,0) with non-zero mean

Coefficients:
         ar1     mean
      0.8280  21.0196
s.e.  0.0296   0.9505

sigma^2 = 9.813:  log likelihood = -911.23
AIC=1828.45   AICc=1828.52   BIC=1840.08
```

it seems there isn't enough data for auto.arima to fully caught the seasonality needed

Forecasting

Now using the forecast function we will produce an out of time cross-validation to forecast the values in the 2 weeks of November

```
forecast(modelMonthlySeasonal100.011Napa, 366)
```

```
          Point Forecast    Lo 80    Hi 80     Lo 95    Hi 95
2012.9727       14.92598 10.83022 19.02173  8.662061 21.18990
2012.9754       16.26633 10.94209 21.59057  8.123605 24.40905
2012.9781       18.45885 12.43152 24.48617  9.240851 27.67684
2012.9809       18.64136 12.17375 25.10897  8.750004 28.53272
2012.9836       18.76622 12.01212 25.52031  8.436723 29.09571
2012.9863       18.98751 12.04219 25.93284  8.365557 29.60947
2012.9891       19.08760 12.01335 26.16185  8.268467 29.90673
2012.9918       19.32427 12.16243 26.48611  8.371186 30.27735
2012.9945       18.47618 11.25455 25.69780  7.431651 29.52070
2012.9973       19.05158 11.78903 26.31414  7.944460 30.15870
2013.0000       20.09832 12.80771 27.38893  8.948296 31.24835
2013.0027       20.43042 13.12059 27.74024  9.251007 31.60983
2013.0055       19.93524 12.59879 27.27170  8.715105 31.15538
2013.0082       20.42736 13.07277 27.78195  9.179482 31.67524
2013.0109       21.91528 14.54846 29.28209 10.648704 33.18185
2013.0137       21.51250 14.13764 28.88735 10.233632 32.79137
2013.0164       21.15118 13.77133 28.53102  9.864681 32.43767
2013.0191       20.96862 13.58467 28.35257  9.675842 32.26140
2013.0219       20.73324 13.34646 28.12001  9.436144 32.03033
2013.0246       20.69125 13.30255 28.07995  9.391206 31.99129
2013.0273       19.61168 12.22167 27.00168  8.309643 30.91371
2013.0301       19.99481 12.60394 27.38567  8.691452 31.29816
2013.0328       20.88183 13.49042 28.27323  9.577646 32.18601
2013.0355       21.08125 13.68955 28.47295  9.776619 32.38588
2013.0383       20.47587 13.08008 27.87166  9.164985 31.78675
2013.0410       20.87644 13.47801 28.27486  9.561519 32.19135
2013.0437       22.28831 14.88833 29.68829 10.971011 33.60561
2013.0464       21.82237 14.42169 29.22304 10.504010 33.14072
2013.0492       21.40857 14.00797 28.80917 10.090331 32.72681
2013.0519       21.18243 13.78119 28.58367  9.863208 32.50165
2013.0546       20.91084 13.50917 28.31251  9.590959 32.23072
2013.0574       20.83878 13.43683 28.24073  9.518469 32.15909
2013.0601       19.73423 12.33211 27.13635  8.413663 31.05479
2013.0628       20.09660 12.69441 27.49880  8.775920 31.41729
```

2013.0656	20.96639	13.56419	28.36858	9.645705	32.28707
2013.0683	21.15149	13.74937	28.55361	9.830930	32.47205
2013.0710	20.53421	13.12929	27.93914	9.209353	31.85908
2013.0738	20.92490	13.51822	28.33159	9.597352	32.25246
2013.0765	22.32857	14.92093	29.73621	10.999564	33.65757
2013.0792	21.85581	14.44790	29.26372	10.526385	33.18523
2013.0820	21.43635	14.02880	28.84390	10.107483	32.76522
2013.0847	21.20551	13.79751	28.61350	9.875956	32.53506
2013.0874	20.93001	13.52172	28.33830	9.600007	32.26001
2013.0902	20.85470	13.44622	28.26318	9.524413	32.18499
2013.0929	19.74745	12.33887	27.15603	8.417010	31.07790
2013.0956	20.10759	12.69898	27.51620	8.777096	31.43808
2013.0984	20.97551	13.56693	28.38409	9.645067	32.30596
2013.1011	21.15907	13.75059	28.56755	9.828779	32.48936
2013.1038	20.54051	13.12935	27.95168	9.206111	31.87491
2013.1066	20.93013	13.51730	28.34297	9.593176	32.26709
2013.1093	22.33291	14.91919	29.74664	10.994595	33.67123
2013.1120	21.85942	14.44546	29.27338	10.520742	33.19810
2013.1148	21.43935	14.02578	28.85292	10.101271	32.77743
2013.1175	21.20800	13.79400	28.62199	9.869266	32.54673
2013.1202	20.93208	13.51780	28.34636	9.592916	32.27124
2013.1230	20.85642	13.44196	28.27087	9.516986	32.19585
2013.1257	19.74888	12.33433	27.16343	8.409302	31.08846
2013.1284	20.10878	12.69419	27.52336	8.769152	31.44840
2013.1311	20.97650	13.56195	28.39104	9.636927	32.31607
2013.1339	21.15989	13.74544	28.57433	9.820475	32.49930
2013.1366	20.54119	13.12408	27.95831	9.197693	31.88469
2013.1393	20.93070	13.51192	28.34948	9.584659	32.27674
2013.1421	22.33338	14.91372	29.75304	10.985992	33.68077
2013.1448	21.85981	14.43992	29.27970	10.512067	33.20755
2013.1475	21.43967	14.02018	28.85917	10.092534	32.78681
2013.1503	21.20827	13.78835	28.62819	9.860478	32.55605
2013.1530	20.93230	13.51210	28.35250	9.584085	32.28052
2013.1557	20.85660	13.43623	28.27698	9.508119	32.20509
2013.1585	19.74903	12.32856	27.16951	8.400404	31.09766
2013.1612	20.10890	12.68840	27.52940	8.760230	31.45758
2013.1639	20.97660	13.55614	28.39707	9.627983	32.32522
2013.1667	21.15998	13.73962	28.58034	9.811514	32.50844
2013.1694	20.54126	13.11824	27.96429	9.188721	31.89381
2013.1721	20.93076	13.50607	28.35545	9.575678	32.28584
2013.1749	22.33343	14.90786	29.75900	10.977004	33.68986
2013.1776	21.85985	14.43405	29.28565	10.503071	33.21663
2013.1803	21.43971	14.01430	28.86511	10.083531	32.79589

2013.1831	21.20829	13.78247	28.63412	9.851470	32.56512
2013.1858	20.93233	13.50622	28.35843	9.575073	32.28958
2013.1885	20.85662	13.43034	28.28291	9.499103	32.21414
2013.1913	19.74905	12.32267	27.17543	8.391385	31.10672
2013.1940	20.10892	12.68251	27.53532	8.751208	31.46663
2013.1967	20.97661	13.55024	28.40299	9.618958	32.33427
2013.1995	21.15999	13.73372	28.58626	9.802487	32.51749
2013.2022	20.54127	13.11234	27.97021	9.179697	31.90285
2013.2049	20.93077	13.50017	28.36136	9.566655	32.29488
2013.2077	22.33344	14.90196	29.76491	10.967980	33.69890
2013.2104	21.85985	14.42815	29.29156	10.494046	33.22566
2013.2131	21.43971	14.00840	28.87102	10.074506	32.80492
2013.2158	21.20830	13.77657	28.64003	9.842444	32.57415
2013.2186	20.93233	13.50032	28.36434	9.566047	32.29861
2013.2213	20.85663	13.42444	28.28881	9.490077	32.22317
2013.2240	19.74905	12.31677	27.18133	8.382359	31.11575
2013.2268	20.10892	12.67661	27.54123	8.742181	31.47566
2013.2295	20.97662	13.54434	28.40889	9.609932	32.34330
2013.2322	21.15999	13.72781	28.59216	9.793460	32.52652
2013.2350	20.54127	13.10644	27.97611	9.170673	31.91187
2013.2377	20.93077	13.49427	28.36726	9.557633	32.30390
2013.2404	22.33344	14.89607	29.77081	10.958959	33.70792
2013.2432	21.85986	14.42225	29.29746	10.485025	33.23468
2013.2459	21.43971	14.00250	28.87692	10.065484	32.81394
2013.2486	21.20830	13.77067	28.64593	9.833423	32.58317
2013.2514	20.93233	13.49442	28.37024	9.557026	32.30763
2013.2541	20.85663	13.41854	28.29471	9.481056	32.23220
2013.2568	19.74905	12.31087	27.18723	8.373338	31.12477
2013.2596	20.10892	12.67071	27.54713	8.733161	31.48468
2013.2623	20.97662	13.53844	28.41479	9.600911	32.35232
2013.2650	21.15999	13.72192	28.59806	9.784439	32.53554
2013.2678	20.54127	13.10054	27.98201	9.161655	31.92089
2013.2705	20.93077	13.48838	28.37315	9.548617	32.31292
2013.2732	22.33344	14.89017	29.77671	10.949944	33.71694
2013.2760	21.85986	14.41636	29.30335	10.476011	33.24370
2013.2787	21.43971	13.99661	28.88282	10.056470	32.82296
2013.2814	21.20830	13.76477	28.65182	9.824409	32.59219
2013.2842	20.93233	13.48852	28.37613	9.548012	32.31664
2013.2869	20.85663	13.41265	28.30060	9.472043	32.24121
2013.2896	19.74905	12.30498	27.19313	8.364325	31.13378
2013.2923	20.10892	12.66482	27.55302	8.724147	31.49369
2013.2951	20.97662	13.53255	28.42068	9.591898	32.36133
2013.2978	21.15999	13.71602	28.60395	9.775426	32.54455

2013.3005	20.54127	13.09465	27.98790	9.152645	31.92990
2013.3033	20.93077	13.48249	28.37905	9.539609	32.32193
2013.3060	22.33344	14.88428	29.78260	10.940937	33.72594
2013.3087	21.85986	14.41047	29.30924	10.467004	33.25271
2013.3115	21.43971	13.99072	28.88871	10.047462	32.83196
2013.3142	21.20830	13.75888	28.65771	9.815402	32.60119
2013.3169	20.93233	13.48263	28.38202	9.539006	32.32565
2013.3197	20.85663	13.40676	28.30649	9.463036	32.25022
2013.3224	19.74905	12.29909	27.19902	8.355318	31.14279
2013.3251	20.10892	12.65893	27.55891	8.715141	31.50270
2013.3279	20.97662	13.52666	28.42657	9.582891	32.37034
2013.3306	21.15999	13.71013	28.60984	9.766419	32.55356
2013.3333	20.54127	13.08876	27.99379	9.143642	31.93891
2013.3361	20.93077	13.47660	28.38493	9.530607	32.33093
2013.3388	22.33344	14.87840	29.78848	10.931937	33.73494
2013.3415	21.85986	14.40459	29.31512	10.458004	33.26171
2013.3443	21.43971	13.98484	28.89459	10.038462	32.84096
2013.3470	21.20830	13.75300	28.66360	9.806402	32.61019
2013.3497	20.93233	13.47675	28.38791	9.530006	32.33465
2013.3525	20.85663	13.40087	28.31238	9.454037	32.25922
2013.3552	19.74905	12.29321	27.20490	8.346319	31.15179
2013.3579	20.10892	12.65304	27.56479	8.706142	31.51170
2013.3607	20.97662	13.52078	28.43246	9.573892	32.37934
2013.3634	21.15999	13.70425	28.61573	9.757420	32.56256
2013.3661	20.54127	13.08288	27.99967	9.134645	31.94790
2013.3689	20.93077	13.47072	28.39081	9.521613	32.33992
2013.3716	22.33344	14.87252	29.79436	10.922943	33.74394
2013.3743	21.85986	14.39871	29.32100	10.449011	33.27070
2013.3770	21.43971	13.97895	28.90047	10.029468	32.84996
2013.3798	21.20830	13.74712	28.66948	9.797409	32.61919
2013.3825	20.93233	13.47087	28.39379	9.521013	32.34364
2013.3852	20.85663	13.39499	28.31826	9.445044	32.26821
2013.3880	19.74905	12.28733	27.21078	8.337327	31.16078
2013.3907	20.10892	12.64716	27.57067	8.697149	31.52069
2013.3934	20.97662	13.51490	28.43834	9.564900	32.38833
2013.3962	21.15999	13.69837	28.62161	9.748427	32.57155
2013.3989	20.54127	13.07700	28.00555	9.125656	31.95689
2013.4016	20.93077	13.46485	28.39669	9.512626	32.34891
2013.4044	22.33344	14.86664	29.80024	10.913957	33.75292
2013.4071	21.85986	14.39283	29.32688	10.440025	33.27968
2013.4098	21.43971	13.97308	28.90635	10.020482	32.85894
2013.4126	21.20830	13.74124	28.67535	9.788424	32.62817
2013.4153	20.93233	13.46500	28.39966	9.512028	32.35263

2013.4180	20.85663	13.38912	28.32413	9.436059	32.27719
2013.4208	19.74905	12.28145	27.21666	8.328342	31.16976
2013.4235	20.10892	12.64129	27.57655	8.688164	31.52967
2013.4262	20.97662	13.50902	28.44421	9.555915	32.39732
2013.4290	21.15999	13.69249	28.62748	9.739442	32.58053
2013.4317	20.54127	13.07113	28.01142	9.116674	31.96587
2013.4344	20.93077	13.45897	28.40256	9.503646	32.35789
2013.4372	22.33344	14.86077	29.80611	10.904978	33.76190
2013.4399	21.85986	14.38696	29.33275	10.431047	33.28866
2013.4426	21.43971	13.96721	28.91222	10.011503	32.86792
2013.4454	21.20830	13.73537	28.68122	9.779445	32.63715
2013.4481	20.93233	13.45912	28.40553	9.503050	32.36161
2013.4508	20.85663	13.38325	28.33000	9.427081	32.28617
2013.4536	19.74905	12.27558	27.22253	8.319364	31.17874
2013.4563	20.10892	12.63542	27.58242	8.679186	31.53865
2013.4590	20.97662	13.50315	28.45008	9.546937	32.40630
2013.4617	21.15999	13.68662	28.63335	9.730464	32.58951
2013.4645	20.54127	13.06526	28.01729	9.107699	31.97485
2013.4672	20.93077	13.45311	28.40843	9.494673	32.36686
2013.4699	22.33344	14.85490	29.81198	10.896006	33.77087
2013.4727	21.85986	14.38109	29.33862	10.422075	33.29764
2013.4754	21.43971	13.96134	28.91808	10.002531	32.87690
2013.4781	21.20830	13.72951	28.68709	9.770473	32.64612
2013.4809	20.93233	13.45326	28.41140	9.494078	32.37058
2013.4836	20.85663	13.37738	28.33587	9.418110	32.29514
2013.4863	19.74905	12.26972	27.22839	8.310393	31.18771
2013.4891	20.10892	12.62955	27.58828	8.670215	31.54762
2013.4918	20.97662	13.49728	28.45595	9.537966	32.41527
2013.4945	21.15999	13.68076	28.63922	9.721493	32.59848
2013.4973	20.54127	13.05940	28.02315	9.098731	31.98382
2013.5000	20.93077	13.44724	28.41429	9.485707	32.37583
2013.5027	22.33344	14.84904	29.81784	10.887041	33.77984
2013.5055	21.85986	14.37523	29.34448	10.413110	33.30660
2013.5082	21.43971	13.95548	28.92395	9.993565	32.88586
2013.5109	21.20830	13.72364	28.69295	9.761509	32.65509
2013.5137	20.93233	13.44740	28.41726	9.485114	32.37954
2013.5164	20.85663	13.37152	28.34173	9.409146	32.30411
2013.5191	19.74905	12.26385	27.23425	8.301429	31.19668
2013.5219	20.10892	12.62369	27.59415	8.661251	31.55659
2013.5246	20.97662	13.49142	28.46181	9.529002	32.42423
2013.5273	21.15999	13.67490	28.64508	9.712529	32.60745
2013.5301	20.54127	13.05354	28.02901	9.089770	31.99278
2013.5328	20.93077	13.44139	28.42015	9.476748	32.38479

2013.5355	22.33344	14.84318	29.82369	10.878084	33.78880
2013.5383	21.85986	14.36937	29.35034	10.404153	33.31556
2013.5410	21.43971	13.94962	28.92980	9.984607	32.89482
2013.5437	21.20830	13.71779	28.69881	9.752551	32.66405
2013.5464	20.93233	13.44154	28.42312	9.476157	32.38850
2013.5492	20.85663	13.36566	28.34759	9.400189	32.31306
2013.5519	19.74905	12.25800	27.24011	8.292472	31.20563
2013.5546	20.10892	12.61783	27.60000	8.652294	31.56554
2013.5574	20.97662	13.48557	28.46767	9.520045	32.43319
2013.5601	21.15999	13.66904	28.65094	9.703572	32.61640
2013.5628	20.54127	13.04768	28.03486	9.080816	32.00173
2013.5656	20.93077	13.43553	28.42600	9.467796	32.39374
2013.5683	22.33344	14.83733	29.82955	10.869133	33.79775
2013.5710	21.85986	14.36352	29.35619	10.395202	33.32451
2013.5738	21.43971	13.94377	28.93566	9.975656	32.90377
2013.5765	21.20830	13.71194	28.70466	9.743600	32.67300
2013.5792	20.93233	13.43569	28.42897	9.467206	32.39745
2013.5820	20.85663	13.35981	28.35344	9.391239	32.32201
2013.5847	19.74905	12.25215	27.24596	8.283522	31.21458
2013.5874	20.10892	12.61198	27.60585	8.643344	31.57449
2013.5902	20.97662	13.47971	28.47352	9.511095	32.44214
2013.5929	21.15999	13.66319	28.65679	9.694622	32.62535
2013.5956	20.54127	13.04183	28.04071	9.071869	32.01068
2013.5984	20.93077	13.42968	28.43185	9.458851	32.40268
2013.6011	22.33344	14.83148	29.83540	10.860189	33.80669
2013.6038	21.85986	14.35767	29.36204	10.386258	33.33345
2013.6066	21.43971	13.93792	28.94150	9.966712	32.91271
2013.6093	21.20830	13.70609	28.71051	9.734657	32.68194
2013.6120	20.93233	13.42984	28.43482	9.458263	32.40639
2013.6148	20.85663	13.35396	28.35929	9.382296	32.33096
2013.6175	19.74905	12.24630	27.25181	8.274579	31.22353
2013.6202	20.10892	12.60614	27.61170	8.634401	31.58344
2013.6230	20.97662	13.47387	28.47937	9.502152	32.45108
2013.6257	21.15999	13.65734	28.66264	9.685678	32.63430
2013.6284	20.54127	13.03599	28.04656	9.062929	32.01962
2013.6311	20.93077	13.42384	28.43769	9.449913	32.41162
2013.6339	22.33344	14.82564	29.84124	10.851252	33.81563
2013.6366	21.85986	14.35183	29.36788	10.377322	33.34239
2013.6393	21.43971	13.93208	28.94735	9.957775	32.92165
2013.6421	21.20830	13.70024	28.71635	9.725720	32.69088
2013.6448	20.93233	13.42400	28.44066	9.449327	32.41533
2013.6475	20.85663	13.34812	28.36513	9.373359	32.33989
2013.6503	19.74905	12.24045	27.25765	8.265643	31.23246

2013.6530	20.10892	12.60029	27.61754	8.625466	31.59237
2013.6557	20.97662	13.46802	28.48521	9.493216	32.46002
2013.6585	21.15999	13.65150	28.66848	9.676742	32.64323
2013.6612	20.54127	13.03015	28.05240	9.053996	32.02855
2013.6639	20.93077	13.41800	28.44353	9.440982	32.42055
2013.6667	22.33344	14.81980	29.84708	10.842322	33.82456
2013.6694	21.85986	14.34599	29.37372	10.368392	33.35132
2013.6721	21.43971	13.92624	28.95319	9.948845	32.93058
2013.6749	21.20830	13.69441	28.72219	9.716790	32.69981
2013.6776	20.93233	13.41816	28.44650	9.440397	32.42426
2013.6803	20.85663	13.34228	28.37097	9.364430	32.34882
2013.6831	19.74905	12.23462	27.26349	8.256714	31.24139
2013.6858	20.10892	12.59445	27.62338	8.616536	31.60130
2013.6885	20.97662	13.46219	28.49105	9.484286	32.46895
2013.6913	21.15999	13.64566	28.67432	9.667813	32.65216
2013.6940	20.54127	13.02431	28.05824	9.045070	32.03748
2013.6967	20.93077	13.41217	28.44937	9.432058	32.42948
2013.6995	22.33344	14.81397	29.85291	10.833399	33.83348
2013.7022	21.85986	14.34016	29.37955	10.359469	33.36024
2013.7049	21.43971	13.92040	28.95902	9.939921	32.93950
2013.7077	21.20830	13.68857	28.72803	9.707868	32.70873
2013.7104	20.93233	13.41232	28.45233	9.431475	32.43318
2013.7131	20.85663	13.33645	28.37680	9.355508	32.35774
2013.7158	19.74905	12.22878	27.26932	8.247792	31.25031
2013.7186	20.10892	12.58862	27.62922	8.607614	31.61022
2013.7213	20.97662	13.45635	28.49688	9.475364	32.47787
2013.7240	21.15999	13.63982	28.68015	9.658891	32.66108
2013.7268	20.54127	13.01848	28.06407	9.036151	32.04640
2013.7295	20.93077	13.40633	28.45520	9.423141	32.43839
2013.7322	22.33344	14.80814	29.85874	10.824482	33.84240
2013.7350	21.85986	14.33433	29.38538	10.350553	33.36916
2013.7377	21.43971	13.91457	28.96485	9.931005	32.94842
2013.7404	21.20830	13.68274	28.73386	9.698952	32.71764
2013.7432	20.93233	13.40649	28.45816	9.422559	32.44210
2013.7459	20.85663	13.33062	28.38263	9.346593	32.36666
2013.7486	19.74905	12.22295	27.27515	8.238876	31.25923
2013.7514	20.10892	12.58279	27.63505	8.598699	31.61914
2013.7541	20.97662	13.45052	28.50271	9.466449	32.48678
2013.7568	21.15999	13.63400	28.68598	9.649976	32.67000
2013.7596	20.54127	13.01265	28.06990	9.027239	32.05531
2013.7623	20.93077	13.40051	28.46103	9.414230	32.44730
2013.7650	22.33344	14.80231	29.86457	10.815573	33.85131
2013.7678	21.85986	14.32850	29.39121	10.341644	33.37807

2013.7705	21.43971	13.90875	28.97068	9.922096	32.95733
2013.7732	21.20830	13.67692	28.73968	9.690043	32.72655
2013.7760	20.93233	13.40067	28.46399	9.413651	32.45101
2013.7787	20.85663	13.32479	28.38846	9.337684	32.37557
2013.7814	19.74905	12.21713	27.28098	8.229968	31.26814
2013.7842	20.10892	12.57697	27.64087	8.589791	31.62805
2013.7869	20.97662	13.44470	28.50853	9.457541	32.49569
2013.7896	21.15999	13.62817	28.69181	9.641067	32.67891
2013.7923	20.54127	13.00683	28.07572	9.018334	32.06421
2013.7951	20.93077	13.39469	28.46685	9.405327	32.45621
2013.7978	22.33344	14.79649	29.87039	10.806671	33.86021
2013.8005	21.85986	14.32268	29.39703	10.332742	33.38697
2013.8033	21.43971	13.90293	28.97650	9.913193	32.96623
2013.8060	21.20830	13.67110	28.74550	9.681141	32.73546
2013.8087	20.93233	13.39485	28.46981	9.404749	32.45991
2013.8115	20.85663	13.31897	28.39428	9.328783	32.38447
2013.8142	19.74905	12.21131	27.28680	8.221067	31.27704
2013.8169	20.10892	12.57115	27.64669	8.580889	31.63695
2013.8197	20.97662	13.43888	28.51435	9.448639	32.50459
2013.8224	21.15999	13.62235	28.69763	9.632165	32.68781
2013.8251	20.54127	13.00101	28.08154	9.009435	32.07311
2013.8279	20.93077	13.38887	28.47266	9.396430	32.46510
2013.8306	22.33344	14.79067	29.87621	10.797775	33.86910
2013.8333	21.85986	14.31686	29.40285	10.323847	33.39586
2013.8361	21.43971	13.89711	28.98232	9.904297	32.97513
2013.8388	21.20830	13.66528	28.75132	9.672245	32.74435
2013.8415	20.93233	13.38903	28.47562	9.395854	32.46880
2013.8443	20.85663	13.31316	28.40009	9.319888	32.39336
2013.8470	19.74905	12.20549	27.29261	8.212172	31.28593
2013.8497	20.10892	12.56533	27.65251	8.571995	31.64584
2013.8525	20.97662	13.43306	28.52017	9.439745	32.51349
2013.8552	21.15999	13.61653	28.70344	9.623271	32.69670
2013.8579	20.54127	12.99520	28.08735	9.000544	32.08200
2013.8607	20.93077	13.38306	28.47848	9.387541	32.47399
2013.8634	22.33344	14.78486	29.88202	10.788887	33.87799
2013.8661	21.85986	14.31105	29.40866	10.314958	33.40475
2013.8689	21.43971	13.89130	28.98813	9.895409	32.98402
2013.8716	21.20830	13.65947	28.75713	9.663357	32.75324
2013.8743	20.93233	13.38322	28.48143	9.386966	32.47769
2013.8770	20.85663	13.30735	28.40590	9.311000	32.40225
2013.8798	19.74905	12.19968	27.29843	8.203284	31.29482
2013.8825	20.10892	12.55952	27.65832	8.563107	31.65473
2013.8852	20.97662	13.42725	28.52598	9.430857	32.52238

```
2013.8880       21.15999 13.61072 28.70925  9.614383 32.70559
2013.8907       20.54127 12.98939 28.09316  8.991659 32.09089
2013.8934       20.93077 13.37725 28.48429  9.378658 32.48288
2013.8962       22.33344 14.77905 29.88783 10.780005 33.88688
2013.8989       21.85986 14.30525 29.41446 10.306077 33.41363
2013.9016       21.43971 13.88549 28.99394  9.886526 32.99290
2013.9044       21.20830 13.65366 28.76294  9.654476 32.76212
2013.9071       20.93233 13.37741 28.48724  9.378085 32.48657
2013.9098       20.85663 13.30154 28.41171  9.302119 32.41113
2013.9126       19.74905 12.19387 27.30423  8.194404 31.30370
2013.9153       20.10892 12.55371 27.66413  8.554226 31.66361
2013.9180       20.97662 13.42144 28.53179  9.421976 32.53126
2013.9208       21.15999 13.60492 28.71506  9.605502 32.71447
2013.9235       20.54127 12.98358 28.09897  8.982781 32.09977
2013.9262       20.93077 13.37145 28.49009  9.369782 32.49175
2013.9290       22.33344 14.77325 29.89363 10.771130 33.89575
2013.9317       21.85986 14.29944 29.42027 10.297202 33.42251
2013.9344       21.43971 13.87969 28.99974  9.877651 33.00177
2013.9372       21.20830 13.64786 28.76874  9.645601 32.77100
2013.9399       20.93233 13.37161 28.49304  9.369210 32.49545
2013.9426       20.85663 13.29574 28.41751  9.293245 32.42001
2013.9454       19.74905 12.18807 27.31004  8.185529 31.31258
2013.9481       20.10892 12.54791 27.66993  8.545352 31.67248
2013.9508       20.97662 13.41564 28.53759  9.413102 32.54013
2013.9536       21.15999 13.59911 28.72086  9.596628 32.72335
2013.9563       20.54127 12.97778 28.10477  8.973910 32.10864
2013.9590       20.93077 13.36565 28.49589  9.360913 32.50062
2013.9617       22.33344 14.76745 29.89943 10.762262 33.90462
2013.9645       21.85986 14.29364 29.42607 10.288334 33.43138
2013.9672       21.43971 13.87389 29.00554  9.868783 33.01064
2013.9699       21.20830 13.64206 28.77454  9.636733 32.77986
```

```r
toPredictValues$Napa
```

```
[1] 14.4 15.0 14.4 16.7 20.6 22.2 20.0 17.2 17.8
```

Death valley

```r
## initial assumption
```

```r
modelMonthlySeasonal100.110DeathValley = Arima(tsCaliforniaDeathValleyTemp,
    order = c(1, 0, 0), seasonal = list(order = c(1, 1, 0), period = 12))

modelMonthlySeasonal100.011DeathValley = Arima(tsCaliforniaDeathValleyTemp,
    order = c(1, 0, 0), seasonal = list(order = c(0, 1, 1), period = 12))

modelMonthlySeasonal200.210DeathValley = Arima(tsCaliforniaDeathValleyTemp,
    order = c(2, 0, 0), seasonal = list(order = c(2, 1, 0), period = 12))
modelMonthlySeasonal200.012DeathValley = Arima(tsCaliforniaDeathValleyTemp,
    order = c(2, 0, 0), seasonal = list(order = c(0, 1, 2), period = 12))

modelMonthlySeasonal001.110DeathValley = Arima(tsCaliforniaDeathValleyTemp,
    order = c(0, 0, 1), seasonal = list(order = c(1, 1, 0), period = 12))

modelMonthlySeasonal001.011DeathValley = Arima(tsCaliforniaDeathValleyTemp,
    order = c(0, 0, 1), seasonal = list(order = c(0, 1, 1), period = 12))

modelMonthlySeasonal002.210DeathValley = Arima(tsCaliforniaDeathValleyTemp,
    order = c(0, 0, 2), seasonal = list(order = c(2, 1, 0), period = 12))
modelMonthlySeasonal002.012DeathValley = Arima(tsCaliforniaDeathValleyTemp,
    order = c(0, 0, 2), seasonal = list(order = c(0, 1, 2), period = 12))
```

```r
modelMonthlySeasonal100.110DeathValley
```

```
Series: tsCaliforniaDeathValleyTemp
ARIMA(1,0,0)(1,1,0)[12]

Coefficients:
         ar1     sar1
      0.8082  -0.4482
s.e.  0.0321   0.0491

sigma^2 = 9.802:  log likelihood = -881.56
AIC=1769.13   AICc=1769.2   BIC=1780.65
```

```r
modelMonthlySeasonal100.011DeathValley
```

```
Series: tsCaliforniaDeathValleyTemp
ARIMA(1,0,0)(0,1,1)[12]
```

```
Coefficients:
         ar1     sma1
      0.9679  -0.8985
s.e.  0.0164   0.0443

sigma^2 = 7.777:  log likelihood = -850.22
AIC=1706.45   AICc=1706.52   BIC=1717.97
```

```
modelMonthlySeasonal200.210DeathValley
```

```
Series: tsCaliforniaDeathValleyTemp
ARIMA(2,0,0)(2,1,0)[12]

Coefficients:
         ar1      ar2     sar1     sar2
      0.9640  -0.1497  -0.5483  -0.1975
s.e.  0.0538   0.0546   0.0555   0.0578

sigma^2 = 9.163:  log likelihood = -869.61
AIC=1749.21   AICc=1749.39   BIC=1768.42
```

```
modelMonthlySeasonal200.012DeathValley
```

```
Series: tsCaliforniaDeathValleyTemp
ARIMA(2,0,0)(0,1,2)[12]

Coefficients:
         ar1      ar2     sma1     sma2
      1.0154  -0.0521  -0.8544  -0.0521
s.e.  0.0538   0.0544   0.0608   0.0560

sigma^2 = 7.769:  log likelihood = -849.32
AIC=1708.63   AICc=1708.81   BIC=1727.84
```

```
modelMonthlySeasonal001.110DeathValley
```

```
Series: tsCaliforniaDeathValleyTemp
ARIMA(0,0,1)(1,1,0)[12]
```

```
Coefficients:
         ma1     sar1
      0.7398  -0.2828
s.e.  0.0308   0.0525

sigma^2 = 13.16:  log likelihood = -931.29
AIC=1868.57   AICc=1868.64   BIC=1880.09
```

```
modelMonthlySeasonal001.011DeathValley
```

```
Series: tsCaliforniaDeathValleyTemp
ARIMA(0,0,1)(0,1,1)[12]

Coefficients:
         ma1     sma1
      0.7460  -0.2453
s.e.  0.0315   0.0468

sigma^2 = 13.31:  log likelihood = -933.06
AIC=1872.12   AICc=1872.19   BIC=1883.64
```

```
modelMonthlySeasonal002.210DeathValley
```

```
Series: tsCaliforniaDeathValleyTemp
ARIMA(0,0,2)(2,1,0)[12]

Coefficients:
         ma1     ma2     sar1    sar2
      0.9398  0.3858  -0.3944  0.0105
s.e.  0.0493  0.0432   0.0571  0.0564

sigma^2 = 10.94:  log likelihood = -899.09
AIC=1808.17   AICc=1808.35   BIC=1827.38
```

```
modelMonthlySeasonal002.012DeathValley
```

```
Series: tsCaliforniaDeathValleyTemp
ARIMA(0,0,2)(0,1,2)[12]
```

```
Coefficients:
         ma1     ma2     sma1    sma2
      0.9326  0.3751  -0.3968  0.1196
s.e.  0.0497  0.0429   0.0581  0.0464

sigma^2 = 10.99:  log likelihood = -899.77
AIC=1809.53   AICc=1809.71   BIC=1828.73
```

Forecast

```
forecast(modelMonthlySeasonal100.011DeathValley, 366)
```

```
          Point Forecast     Lo 80    Hi 80       Lo 95    Hi 95
2012.9727       17.55612 13.981621 21.13061 12.08939732 23.02284
2012.9754       17.24408 12.269606 22.21855  9.63627874 24.85188
2012.9781       17.65015 11.653802 23.64650  8.47952529 26.82078
2012.9809       18.51313 11.697162 25.32909  8.08900843 28.93725
2012.9836       19.08324 11.580327 26.58615  7.60852551 30.55795
2012.9863       19.63382 11.540130 27.72751  7.25559039 32.01205
2012.9891       20.56363 11.953235 29.17402  7.39516750 33.73209
2012.9918       19.34573 10.277995 28.41346  5.47782621 33.21363
2012.9945       20.52485 11.048715 30.00098  6.03235322 35.01734
2012.9973       20.43781 10.594486 30.28113  5.38374523 35.49188
2013.0000       20.60068 10.425416 30.77594  5.03895769 36.16240
2013.0027       20.53368 10.057025 31.01033  4.51102113 36.55633
2013.0055       21.26659 10.427741 32.10544  4.68999978 37.84318
2013.0082       20.83532  9.667849 32.00278  3.75614945 37.91448
2013.0109       21.12598  9.659248 32.59271  3.58912675 38.66283
2013.0137       21.87725 10.137131 33.61738  3.92228531 39.83222
2013.0164       22.33925 10.348714 34.32979  4.00130618 40.67720
2013.0191       22.78520 10.564706 35.00569  4.09556783 41.47483
2013.0219       23.61374 11.181703 36.04577  4.60058211 42.62689
2013.0246       22.29782  9.670850 34.92478  2.98653762 41.60910
2013.0273       23.38207 10.575197 36.18893  3.79565092 42.96848
2013.0301       23.20321 10.230095 36.17632  3.36254261 43.04388
2013.0328       23.27721 10.150285 36.40413  3.20131186 43.35310
2013.0355       23.12419  9.854822 36.39356  2.83044225 43.41794
2013.0383       23.77386 10.316638 37.23108  3.19281554 44.35490
2013.0410       23.26201  9.631187 36.89283  2.41546625 44.10855
2013.0437       23.47469  9.683250 37.26612  2.38250479 44.56687
2013.0464       24.15048 10.210292 38.09067  2.83080066 45.47016
```

2013.0492	24.53943	10.461345	38.61752	3.00885729	46.07000
2013.0519	24.91467	10.708583	39.12075	3.18833571	46.64100
2013.0546	25.67477	11.349829	39.99972	3.76666261	47.58288
2013.0574	24.29262	9.857236	38.72801	2.21560500	46.36964
2013.0601	25.31276	10.774696	39.85083	3.07870717	47.54682
2013.0628	25.07186	10.438271	39.70545	2.69171583	47.45201
2013.0656	25.08581	10.363311	39.80831	2.56969224	47.60192
2013.0683	24.87467	10.069390	39.67995	2.23194734	47.51739
2013.0710	25.46808	10.544278	40.39189	2.64409263	48.29208
2013.0738	24.90179	9.867821	39.93575	1.90931983	47.89426
2013.0765	25.06177	9.925359	40.19818	1.91262643	48.21091
2013.0792	25.68656	10.454834	40.91829	2.39164307	48.98148
2013.0820	26.02615	10.705690	41.34660	2.59553020	49.45676
2013.0847	26.35361	10.950449	41.75677	2.79650950	49.91071
2013.0874	27.06747	11.587250	42.54769	3.39251649	50.74242
2013.0902	25.64056	10.088512	41.19261	1.85575337	49.42537
2013.0929	26.61739	10.998360	42.23641	2.73014701	50.50463
2013.0956	26.33456	10.653064	42.01606	2.35178164	50.31734
2013.0984	26.30793	10.568152	42.04770	2.23601723	50.37984
2013.1011	26.05752	10.263354	41.85168	1.90242980	50.21260
2013.1038	26.61292	10.734721	42.49111	2.32931187	50.89652
2013.1066	26.00983	10.053336	41.96633	1.60647739	50.41318
2013.1093	26.13420	10.104731	42.16368	1.61923948	50.64917
2013.1120	26.72453	10.627018	42.82205	2.10550731	51.34356
2013.1148	27.03076	10.869790	43.19173	2.31468784	51.74683
2013.1175	27.32594	11.105694	43.54618	2.51921605	52.13266
2013.1202	28.00855	11.732991	44.28411	3.11722908	52.89988
2013.1230	26.55140	10.224201	42.87860	1.58110273	51.52170
2013.1257	27.49896	11.123540	43.87437	2.45491901	52.54299
2013.1284	27.18780	10.767358	43.60824	2.07490106	52.30070
2013.1311	27.13375	10.671252	43.59624	1.95653341	52.31096
2013.1339	26.85680	10.355018	43.35857	1.61950392	52.09409
2013.1366	27.38651	10.819799	43.95322	2.04991052	52.72311
2013.1393	26.75856	10.131272	43.38586	1.32931539	52.18781
2013.1421	26.85888	10.175056	43.54270	1.34317528	52.37458
2013.1448	27.42592	10.689338	44.16250	1.82952766	53.02231
2013.1475	27.70961	10.923776	44.49544	2.03789435	53.38132
2013.1503	27.98297	11.151075	44.81486	2.24081022	53.72512
2013.1530	28.64447	11.769550	45.51938	2.83650878	54.45243
2013.1557	27.16688	10.251768	44.08199	1.29744885	53.03631
2013.1585	28.09466	11.141988	45.04732	2.16778717	54.02152
2013.1612	27.76435	10.776592	44.75212	1.78381418	53.74489
2013.1639	27.69177	10.671215	44.71233	1.66107508	53.72247

2013.1667	27.39689	10.345679	44.44810	1.31931247	53.47447
2013.1694	27.90925	10.804432	45.01407	1.74968904	54.06881
2013.1721	27.26450	10.109641	44.41937	1.02840561	53.50060
2013.1749	27.34856	10.146966	44.55015	1.04099414	53.65612
2013.1776	27.89986	10.654633	45.14509	1.52555967	54.27416
2013.1803	28.16832	10.882333	45.45430	1.73168481	54.60495
2013.1831	28.42694	11.102802	45.75108	1.93195789	54.92192
2013.1858	29.07417	11.714381	46.43396	2.52466377	55.62368
2013.1885	27.58278	10.189663	44.97589	0.98230578	54.18324
2013.1913	28.49719	11.072926	45.92144	1.84908084	55.14529
2013.1940	28.15395	10.700574	45.60732	1.46131568	54.84658
2013.1967	28.06885	10.588255	45.54944	1.33458828	54.80311
2013.1995	27.76185	10.255808	45.26789	0.98867086	54.53502
2013.2022	28.26248	10.709905	45.81505	1.41813391	55.10682
2013.2049	27.60638	10.010346	45.20241	0.69556760	54.51719
2013.2077	27.67945	10.042816	45.31608	0.70654716	54.65235
2013.2104	28.22012	10.545562	45.89467	1.18921668	55.25102
2013.2131	28.47828	10.768295	46.18827	1.39319402	55.56337
2013.2158	28.72694	10.983768	46.47012	1.59109906	55.86278
2013.2186	29.36453	11.590337	47.13873	2.18124613	56.54782
2013.2213	27.86381	10.060608	45.66700	0.63616455	55.09145
2013.2240	28.76919	10.938872	46.59950	1.50007470	56.03830
2013.2268	28.41721	10.561543	46.27287	1.10932453	55.72509
2013.2295	28.32365	10.444278	46.20302	0.97951120	55.66778
2013.2322	28.00846	10.106924	45.90999	0.63042426	55.38649
2013.2350	28.50116	10.557696	46.44463	1.05899888	55.94333
2013.2377	27.83739	9.854753	45.82004	0.33531815	55.33947
2013.2404	27.90304	9.883795	45.92228	0.34498415	55.46109
2013.2432	28.43652	10.383081	46.48997	0.82616521	56.04688
2013.2459	28.68773	10.602334	46.77313	1.02850109	56.34697
2013.2486	28.92966	10.814317	47.04501	1.22463197	56.63469
2013.2514	29.56074	11.417396	47.70408	1.81289005	57.30859
2013.2541	28.05371	9.884185	46.22323	0.26582087	55.84159
2013.2568	28.95298	10.758982	47.14698	1.12765950	56.77831
2013.2596	28.59510	10.378207	46.81199	0.73476718	56.45543
2013.2623	28.49582	10.257524	46.73412	0.60275308	56.38889
2013.2650	28.17510	9.916785	46.43341	0.25141766	56.09878
2013.2678	28.66245	10.365308	46.95959	0.67938725	56.64551
2013.2705	27.99350	9.660078	46.32692	-0.04504888	56.03204
2013.2732	28.05412	9.686802	46.42145	-0.03627101	56.14452
2013.2760	28.58275	10.183750	46.98176	0.44390521	56.72160
2013.2787	28.82927	10.400653	47.25788	0.64513402	57.01340
2013.2814	29.06665	10.610280	47.52301	0.84006963	57.29322

2013.2842	29.69332	11.211003	48.17564	1.42705557	57.95959
2013.2869	28.18203	9.675442	46.68861	-0.12135186	56.48541
2013.2896	29.07718	10.547900	47.60646	0.73909220	57.41527
2013.2923	28.71530	10.164801	47.26581	0.34475782	57.08585
2013.2951	28.61216	10.041812	47.18252	0.21126221	57.01307
2013.2978	28.28770	9.698789	46.87662	-0.14158694	56.71699
2013.3005	28.77143	10.145866	47.39700	0.28608559	57.25678
2013.3033	28.09898	9.439158	46.75880	-0.43875603	56.63672
2013.3060	28.15622	9.464381	46.84805	-0.43047974	56.74292
2013.3087	28.68157	9.959810	47.40332	0.04910995	57.31402
2013.3115	28.92490	10.175181	47.67462	0.24967742	57.60013
2013.3142	29.15921	10.383269	47.93515	0.44388724	57.87453
2013.3169	29.78291	10.982451	48.58337	1.03009037	58.53573
2013.3197	28.26874	9.445350	47.09212	-0.51914929	57.05662
2013.3224	29.16110	10.316271	48.00593	0.34041997	57.98179
2013.3251	28.79653	9.931643	47.66142	-0.05482552	57.64789
2013.3279	28.69078	9.807135	47.57442	-0.18926289	57.57082
2013.3306	28.36379	9.462605	47.26498	-0.54307931	57.27066
2013.3333	28.84508	9.908810	47.78135	-0.11544537	57.80560
2013.3361	28.17026	9.201202	47.13931	-0.84040949	57.18093
2013.3388	28.22520	9.225504	47.22491	-0.83233060	57.28274
2013.3415	28.74834	9.719992	47.77668	-0.35300502	57.84968
2013.3443	28.98953	9.934409	48.04464	-0.15276092	58.13181
2013.3470	29.22176	10.141534	48.30198	0.04107439	58.40244
2013.3497	29.84345	10.739745	48.94715	0.62685696	59.06004
2013.3525	28.32733	9.201669	47.45299	-0.92284374	57.57750
2013.3552	29.21781	10.071614	48.36401	-0.06377087	58.49939
2013.3579	28.85142	9.686010	48.01682	-0.45954344	58.16238
2013.3607	28.74390	9.560530	47.92728	-0.59453438	58.08234
2013.3634	28.41521	9.215032	47.61538	-0.94892701	57.77934
2013.3661	28.89484	9.660767	48.12892	-0.52113654	58.31082
2013.3689	28.21842	8.952664	47.48418	-1.24601307	57.68286
2013.3716	28.27182	8.976446	47.56720	-1.23790860	57.78155
2013.3743	28.79346	9.470396	48.11651	-0.75861313	58.34552
2013.3770	29.03320	9.684259	48.38213	-0.55844907	58.62484
2013.3798	29.26402	9.890818	48.63723	-0.36473706	58.89278
2013.3825	29.88435	10.488452	49.28025	0.22088251	59.54782
2013.3852	28.36692	8.949791	47.78405	-1.32901651	58.06286
2013.3880	29.25613	9.819146	48.69312	-0.47017329	58.98244
2013.3907	28.88851	9.432947	48.34406	-0.86620322	58.64321
2013.3934	28.77980	9.306870	48.25273	-1.00147590	58.56107
2013.3962	28.44995	8.960775	47.93913	-1.35617162	58.25607
2013.3989	28.92847	9.406317	48.45062	-0.92808624	58.78502

2013.4016	28.25097	8.697993	47.80394	-1.65272718	58.15466
2013.4044	28.30332	8.721532	47.88511	-1.64444089	58.25108
2013.4071	28.82394	9.215219	48.43267	-1.16501228	58.81290
2013.4098	29.06270	9.428801	48.69661	-0.96475912	59.09017
2013.4126	29.29258	9.635064	48.95010	-0.77099631	59.35616
2013.4153	29.91200	10.232391	49.59160	-0.18536165	60.00935
2013.4180	28.39367	8.693412	48.09394	-1.73527771	58.52263
2013.4208	29.28203	9.562438	49.00161	-0.87648043	59.44053
2013.4235	28.91357	9.175905	48.65123	-1.27258223	59.09972
2013.4262	28.80406	9.049487	48.55862	-1.40794998	59.01606
2013.4290	28.47343	8.703046	48.24381	-1.76276148	58.70962
2013.4317	28.95119	9.148581	48.75380	-1.33428769	59.23667
2013.4344	28.27296	8.440223	48.10570	-2.05859311	58.60451
2013.4372	28.32461	8.463706	48.18551	-2.05001950	58.69923
2013.4399	28.84454	8.957315	48.73177	-1.57034752	59.25944
2013.4426	29.08264	9.170801	48.99448	-1.36989103	59.53518
2013.4454	29.31188	9.376953	49.24681	-1.17595989	59.79972
2013.4481	29.93067	9.974154	49.88719	-0.59018990	60.45154
2013.4508	28.41175	8.435035	48.38847	-2.14000065	58.96351
2013.4536	29.29952	9.303912	49.29513	-1.28112531	59.88017
2013.4563	28.93050	8.917218	48.94378	-1.67717376	59.53818
2013.4590	28.82045	8.790632	48.85026	-1.81251055	59.45340
2013.4617	28.48929	8.444016	48.53457	-2.16731136	59.14589
2013.4645	28.96654	8.889667	49.04342	-1.73838914	59.67148
2013.4672	28.28782	8.181401	48.39424	-2.46229441	59.03794
2013.4699	28.33899	8.204951	48.47303	-2.45336493	59.13134
2013.4727	28.85846	8.698607	49.01832	-1.97337764	59.69031
2013.4754	29.09612	8.912119	49.28011	-1.77264303	59.96487
2013.4781	29.32492	9.118282	49.53156	-1.57846638	60.22831
2013.4809	29.94329	9.715478	50.17111	-0.99248193	60.87907
2013.4836	28.42397	8.176340	48.67160	-2.54210663	59.39004
2013.4863	29.31135	9.045186	49.57751	-1.68307125	60.30576
2013.4891	28.94194	8.658450	49.22544	-2.07898344	59.96287
2013.4918	28.83152	8.531811	49.13123	-2.21420571	59.87725
2013.4945	28.50001	8.185134	48.81489	-2.56891186	59.56893
2013.4973	28.97692	8.630983	49.32286	-2.13950522	60.09334
2013.5000	28.29786	7.922889	48.67284	-2.86297096	59.45870
2013.5027	28.34871	7.946587	48.75083	-2.85364324	59.55106
2013.5055	28.86787	8.440370	49.29537	-2.37329573	60.10904
2013.5082	29.10522	8.653991	49.55645	-2.17223589	60.38268
2013.5109	29.33373	8.860245	49.80722	-1.97776481	60.64523
2013.5137	29.95182	9.457516	50.44613	-1.39151543	61.29516
2013.5164	28.43222	7.918439	48.94601	-2.94090234	59.80535

2013.5191	29.31933	8.787332	49.85134	-2.08165415	60.72032
2013.5219	28.94968	8.400631	49.49872	-2.47737650	60.37673
2013.5246	28.83901	8.274017	49.40399	-2.61243008	60.29044
2013.5273	28.50725	7.927354	49.08715	-2.96698697	59.98150
2013.5301	28.98393	8.373452	49.59441	-2.53707677	60.50494
2013.5328	28.30465	7.665581	48.94371	-3.26008117	59.86938
2013.5355	28.35528	7.689480	49.02107	-3.25033109	59.96088
2013.5383	28.87423	8.183443	49.56501	-2.76959720	60.51805
2013.5410	29.11137	8.397224	49.82552	-2.56818422	60.79093
2013.5437	29.33969	8.603621	50.07575	-2.37338929	61.05276
2013.5464	29.95759	9.201019	50.71415	-1.78684426	61.70202
2013.5492	28.43780	7.662054	49.21355	-3.33596160	60.21156
2013.5519	29.32473	8.531046	50.11842	-2.47646789	61.12593
2013.5546	28.95490	8.144431	49.76537	-2.87196697	60.78177
2013.5574	28.84406	8.017891	49.67023	-3.00681781	60.69494
2013.5601	28.51215	7.671292	49.35301	-3.36119093	60.38549
2013.5628	28.98867	8.117669	49.85967	-2.93076964	60.90810
2013.5656	28.30923	7.410054	49.20841	-3.65330300	60.27177
2013.5683	28.35971	7.434186	49.28524	-3.64311891	60.36255
2013.5710	28.87852	7.928361	49.82869	-3.16198530	60.91903
2013.5738	29.11553	8.142334	50.08872	-2.96020435	61.19126
2013.5765	29.34371	8.348907	50.33851	-2.76506947	61.45249
2013.5792	29.96148	8.946465	50.97650	-2.17821159	62.10117
2013.5820	28.44157	7.407646	49.47549	-3.72704115	60.61018
2013.5847	29.32838	8.276768	50.37999	-2.86728290	61.52405
2013.5874	28.95843	7.890272	50.02659	-3.26253896	61.17941
2013.5902	28.84748	7.763839	49.93112	-3.39716672	61.09213
2013.5929	28.51546	7.417335	49.61358	-3.75133526	60.78225
2013.5956	28.99187	7.864010	50.11973	-3.32040315	61.30414
2013.5984	28.31233	7.156668	49.46799	-4.04246389	60.66713
2013.6011	28.36271	7.181051	49.54437	-4.03184259	60.75727
2013.6038	28.88143	7.675457	50.08739	-3.55030458	61.31315
2013.6066	29.11834	7.889643	50.34703	-3.34814966	61.58483
2013.6093	29.34643	8.096412	50.59645	-3.15266774	61.84552
2013.6120	29.96411	8.694149	51.23407	-2.56548887	62.49371
2013.6148	28.44412	7.155494	49.73274	-4.11402163	61.00226
2013.6175	29.33085	8.024768	50.63693	-3.25398903	61.91568
2013.6202	28.96082	7.638410	50.28323	-3.64899158	61.57063
2013.6230	28.84979	7.512103	50.18747	-3.78338520	61.48296
2013.6257	28.51769	7.165716	49.86967	-4.13733756	61.17272
2013.6284	28.99403	7.612696	50.37537	-3.70590006	61.69396
2013.6311	28.31442	6.905636	49.72321	-4.42749200	61.05634
2013.6339	28.36474	6.930281	49.79920	-4.41643585	61.14591

2013.6366	28.88339	7.424927	50.34185	-3.93449451	61.70127
2013.6393	29.12024	7.639336	50.60114	-3.73196551	61.97244
2013.6421	29.34827	7.846311	50.85022	-3.53613542	62.23267
2013.6448	29.96589	8.444240	51.48754	-2.94863351	62.88041
2013.6475	28.44584	6.905761	49.98591	-4.49686657	61.38854
2013.6503	29.33251	7.775197	50.88983	-3.63655598	62.30158
2013.6530	28.96243	7.388989	50.53587	-4.03130072	61.95616
2013.6557	28.85135	7.262821	50.43988	-4.16545529	61.86815
2013.6585	28.51920	6.916561	50.12184	-4.51918607	61.55759
2013.6612	28.99549	7.363849	50.62714	-4.08725191	62.07824
2013.6639	28.31584	6.657075	49.97460	-4.80838235	61.44006
2013.6667	28.36611	6.681984	50.05023	-4.79689733	61.52911
2013.6694	28.88471	7.176876	50.59255	-4.31455745	62.08398
2013.6721	29.12152	7.391513	50.85153	-4.11165811	62.35470
2013.6749	29.34951	7.598699	51.10031	-3.91548265	62.61450
2013.6776	29.96709	8.196824	51.73736	-3.32765962	63.26184
2013.6803	28.44700	6.658527	50.23547	-4.87559412	61.76960
2013.6831	29.33364	7.528132	51.13914	-4.01500596	62.68228
2013.6858	28.96352	7.142080	50.78496	-4.40949267	62.33653
2013.6885	28.85240	7.016056	50.68875	-4.54340740	62.24821
2013.6913	28.52022	6.669930	50.37051	-4.89691528	61.93736
2013.6940	28.99648	7.117525	50.87544	-4.46449522	62.45746
2013.6967	28.31680	6.411035	50.22256	-5.18517359	61.81876
2013.6995	28.36703	6.436209	50.29786	-5.17326797	61.90733
2013.7022	28.88561	6.931347	50.83987	-4.69053671	62.46175
2013.7049	29.12239	7.146211	51.09856	-4.48727320	62.73204
2013.7077	29.35035	7.353611	51.34708	-4.29075766	62.99145
2013.7104	29.96790	7.951933	51.98387	-3.70261799	63.63843
2013.7131	28.44779	6.413820	50.48175	-5.25025767	62.14583
2013.7158	29.33440	7.283595	51.38520	-4.38939499	63.05819
2013.7186	28.96426	6.897702	51.03081	-4.78362611	62.71214
2013.7213	28.85312	6.771826	50.93441	-4.91730287	62.62353
2013.7240	28.52091	6.425837	50.61599	-5.27058921	62.31241
2013.7268	28.99715	6.873733	51.12056	-4.83769516	62.83199
2013.7295	28.31744	6.167525	50.46736	-5.55793219	62.19282
2013.7322	28.36766	6.192960	50.54236	-5.54561555	62.28093
2013.7350	28.88621	6.688341	51.08408	-5.06250152	62.83493
2013.7377	29.12297	6.903432	51.34251	-4.85888149	63.10482
2013.7404	29.35091	7.111043	51.59078	-4.66203273	63.36386
2013.7432	29.96845	7.709562	52.22734	-4.07358249	64.01049
2013.7459	28.44832	6.171632	50.72500	-5.62093270	62.51757
2013.7486	29.33491	7.041578	51.62825	-4.75980022	63.42962
2013.7514	28.96475	6.655844	51.27366	-5.15377986	63.08329

2013.7541	28.85360	6.530115	51.17708	-5.28722222	62.99442
2013.7568	28.52138	6.184264	50.85849	-5.64029010	62.68305
2013.7596	28.99760	6.632457	51.36274	-5.20693454	63.20213
2013.7623	28.31788	5.926525	50.70923	-5.92674158	62.56250
2013.7650	28.36808	5.952217	50.78394	-5.91402427	62.65019
2013.7678	28.88662	6.447838	51.32540	-5.43053683	63.20378
2013.7705	29.12337	6.663151	51.58358	-5.22656880	63.47330
2013.7732	29.35130	6.870971	51.83162	-5.02939456	63.73199
2013.7760	29.96882	7.469684	52.46796	-4.44064077	64.37829
2013.7787	28.44868	5.931935	50.96542	-5.98770785	62.88506
2013.7814	29.33526	6.802050	51.86847	-5.12631128	63.79683
2013.7842	28.96509	6.416473	51.51371	-5.52004458	63.45022
2013.7869	28.85392	6.290891	51.41695	-5.65325716	63.36110
2013.7896	28.52169	5.945175	51.09821	-6.00611070	63.04950
2013.7923	28.99790	6.393659	51.60215	-5.57230631	63.56811
2013.7951	28.31817	5.687997	50.94835	-6.29169497	62.92804
2013.7978	28.36837	5.713941	51.02279	-6.27858763	63.01532
2013.8005	28.88690	6.209796	51.56400	-5.79473655	63.56853
2013.8033	29.12363	6.425328	51.82194	-5.59042944	63.83770
2013.8060	29.35155	6.633352	52.06976	-5.39293793	64.09605
2013.8087	29.96907	7.232256	52.70589	-4.80388810	64.74203
2013.8115	28.44892	5.694685	51.20315	-6.35067892	63.24852
2013.8142	29.33549	6.564965	52.10602	-5.48902453	64.16001
2013.8169	28.96532	6.179543	51.75109	-5.88251721	63.81315
2013.8197	28.85414	6.054105	51.65418	-6.01550523	63.72379
2013.8224	28.52191	5.708525	51.33529	-6.36814920	63.41196
2013.8251	28.99811	6.157291	51.83893	-5.93390856	63.93013
2013.8279	28.31837	5.451893	51.18485	-6.65289045	63.28963
2013.8306	28.36856	5.478082	51.25904	-6.63940376	63.37652
2013.8333	28.88708	5.974167	51.80000	-6.15519889	63.92937
2013.8361	29.12382	6.189913	52.05772	-5.95056178	64.19819
2013.8388	29.35173	6.398137	52.30532	-5.75276138	64.45622
2013.8415	29.96924	6.997227	52.94126	-5.16342323	65.10191
2013.8443	28.44908	5.459830	51.43833	-6.70994489	63.60811
2013.8470	29.33565	6.330273	52.34103	-5.84803923	64.51934
2013.8497	28.96547	5.945002	51.98594	-6.24129732	64.17224
2013.8525	28.85429	5.819706	51.88888	-6.37406631	64.08265
2013.8552	28.52205	5.474258	51.56984	-6.72650578	63.77060
2013.8579	28.99825	5.923299	52.07320	-6.29184124	64.28834
2013.8607	28.31851	5.218158	51.41886	-7.01042778	63.64744
2013.8634	28.36869	5.244588	51.49279	-6.99657232	63.73395
2013.8661	28.88721	5.740895	52.03352	-6.51202340	64.28644
2013.8689	29.12394	5.956850	52.29102	-6.30706531	64.55494

```
2013.8716        29.35185  6.165269 52.53843 -6.10896440 64.81266
2013.8743        29.96936  6.764542 53.17417 -5.51934567 65.45806
2013.8770        28.44919  5.227315 51.67107 -7.06560534 63.96399
2013.8798        29.33576  6.097916 52.57360 -6.20345503 64.87497
2013.8825        28.96557  5.712794 52.21836 -6.59648464 64.52763
2013.8852        28.85439  5.587636 52.12115 -6.72904026 64.43782
2013.8880        28.52215  5.242317 51.80198 -7.08128045 64.12557
2013.8907        28.99834  5.691627 52.30506 -6.64620403 64.64289
2013.8934        28.31860  4.986735 51.65046 -7.36440635 64.00160
2013.8962        28.36878  5.013398 51.72416 -7.35019241 64.08775
2013.8989        28.88730  5.509924 52.26467 -6.86530902 64.63990
2013.9016        29.12402  5.726082 52.52196 -6.66003881 64.90808
2013.9044        29.35193  5.934691 52.76916 -6.46164563 65.16550
2013.9071        29.96943  6.534141 53.40473 -5.87175395 65.81062
2013.9098        28.44927  4.997080 51.90146 -7.41775872 64.31629
2013.9126        29.33583  5.867836 52.80383 -6.55537032 65.22703
2013.9153        28.96564  5.482858 52.44843 -6.94817755 64.87947
2013.9180        28.85446  5.357836 52.35108 -7.08052544 64.78944
2013.9208        28.52221  5.012643 52.03178 -7.43257159 64.47699
2013.9235        28.99841  5.462213 52.53460 -6.99709491 64.99391
2013.9262        28.31866  4.757566 51.87975 -7.71492378 64.35224
2013.9290        28.36884  4.784456 51.95322 -7.70036139 64.43803
2013.9317        28.88735  5.281193 52.49351 -7.21515280 64.98986
2013.9344        29.12408  5.497549 52.75060 -7.00957910 65.25773
2013.9372        29.35198  5.706343 52.99762 -6.81090167 65.51486
2013.9399        29.96949  6.305966 53.63301 -6.22074452 66.15972
2013.9426        28.44932  4.769067 52.12957 -7.76650131 64.66514
2013.9454        29.33588  5.639974 53.03179 -6.90388127 65.57564
2013.9481        28.96569  5.255137 52.67625 -7.29647211 65.22786
2013.9508        28.85451  5.130246 52.57876 -7.42861783 65.13763
2013.9536        28.52226  4.785176 52.25934 -7.78047511 64.82499
2013.9563        28.99845  5.235001 52.76190 -7.34460935 65.34151
2013.9590        28.31870  4.530590 52.10681 -8.06207518 64.69948
2013.9617        28.36888  4.557701 52.18005 -8.04717399 64.78493
2013.9645        28.88739  5.054645 52.72014 -7.56164919 65.33643
2013.9672        29.12411  5.271193 52.97703 -7.35578034 65.60401
2013.9699        29.35202  5.480168 53.22387 -7.15682645 65.86086
```

```r
toPredictValues$`Death Valley`
```

```
[1] 31.7 23.9 20.0 19.4 20.6 21.7 22.2 21.1 21.7
```