Phosphate - lubridate - dates

Make sure that you have installed and loaded the tidyverse package.

```
library(tidyverse)
```

Introduction

In this exercise we will look at some ways to handle dates and make plots for time series data.

Read and subset the phosphate dataset

Read the phosphate data set using the phosphate dataset.

```
phosphate<-read_tsv("Phosphate.txt")
```

In this dataset 26 different water courses are included. Observations are recorded for 5 variables between 2006 and 2018. Also the station name, an ID and sampling date are part of the dataset.

Click on the data frame `phosphate` in the upper right window (Global Environment) to check if the station names have correct Swedish letter encoding. If not try the following:

```
guess_encoding("Phosphate.txt", n_max = 1000)
```

You will get some suggestions for probably encodings of the file. Choose the first encoding suggested (the one with the highest probability) and enter it in the code below instead of `UTF-8`. If that does not lead to correct Swedish letters ask us for further possible solutions.

```
phosphate<-read_tsv("Phosphate.txt", locale = locale(encoding = "UTF-8"))
```

```
## Rows: 4298 Columns: 8
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: "\t"
## chr (3): Name, EU id, Date
## dbl (5): Abs_OF 420 (/5cm), Abs_F 420 (/5cm), TOC (mg/l), Turb_FNU (FNU), PO4-P (µg/l)
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

We need now to check if the different variables are read correctly, e.g. the last five should be numeric (`num`). We can check this by Using `str(phosphate)`.

```
str(phosphate)
```

```
## spec_tbl_df [4,298 x 8] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Name             : chr [1:4298] "Sävjaån Ingvasta" "Sävjaån Ingvasta" "Sävjaån Ingvasta" "Sävjaån
##  $ EU id            : chr [1:4298] "SE665649-161397" "SE665649-161397" "SE665649-161397" "SE665649-1
##  $ Date             : chr [1:4298] "12/01/2006" "14/02/2006" "15/03/2006" "12/04/2006" ...
##  $ Abs_OF 420 (/5cm): num [1:4298] 0.591 0.482 0.434 0.706 0.734 0.719 0.769 0.753 0.622 0.504 ...
##  $ Abs_F 420 (/5cm) : num [1:4298] 0.391 0.368 0.342 0.456 0.53 0.489 0.317 0.28 0.283 0.272 ...
##  $ TOC (mg/l)       : num [1:4298] 23.9 20.8 20.2 25.5 27.4 29.1 21.2 21.1 22 21.3 ...
##  $ Turb_FNU (FNU)   : num [1:4298] NA NA NA NA NA NA NA NA NA NA ...
```

```
##  $ PO4-P (µg/l)     : num [1:4298] 20 14 18 24 29 27 53 26 31 15 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Name = col_character(),
##   ..   `EU id` = col_character(),
##   ..   Date = col_character(),
##   ..   `Abs_OF 420 (/5cm)` = col_double(),
##   ..   `Abs_F 420 (/5cm)` = col_double(),
##   ..   `TOC (mg/l)` = col_double(),
##   ..   `Turb_FNU (FNU)` = col_double(),
##   ..   `PO4-P (µg/l)` = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

As coding can depend on the local settings on the computer, there is no guarentee that your output will like exactly as above. If \texttt{$ Date} is given as `Date, format:` it is read as correctly as date. In the case above it is read as `chr`, i.e. a character or text variable. To use it further we need to change it into a date variable. For this the package `lubridate` is used. Install it first, if it is not yet avaiable among your packages.

```
library(lubridate)
```

Next, we need to define the format of the date variable. In our case we have e.g. the frist entry as `"12/01/2006"` and the second as `"14/02/2006"` indicating that in the first place the day is given, then month and last year. To change the variable in a date format we can then use the function `dmy`. If the order of day, month and year is different use `?dmy` to access the other codings (or guess them),

```
phosphate1<- phosphate%>%
  mutate(Date=dmy(Date))
```

Check with `str(phosphate1)` that the date variable is now, in fact, a date.

For statistical analysis or plots, it can additionally be an advantage to have a new variable that contains only the year or only the month. WE create these with the functions `year` and `month`.

```
phosphate1<- phosphate%>%
  mutate(Date=dmy(Date), year=year(Date), month=month(Date))
```

To get an overview over the dataset, we could create a table that lists how many observations are registered for each year and each station. Using the fanction `tabyl` we can create the output below, where we see that in most stations there are 12 observations per year. Some are more intensively monitores with 16 or more observations per year and most, but not all, start in 2006. For this we need the package `janitor`.

```
library(janitor)
phosphate1%>%
  tabyl(Name, year)
```

```
##                      Name 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018
##                Akkarjåkkå   32   10   12   12   12   12   11   12   12   12   12   12   12
##            Ålbergaån, Kila   12   12   12   11   12   12   12   12   12   12   12   12   12
##                Björnbackån   17   18   21   12   12   12   12   12   12   12   12   12   12
##             Bjurforsbäcken    0   10   12    9   12   12   11   12   12   12   12   12   12
##                   Bulsjöån    0   11   12   12   12   12   12   12   12   12   12   12   12
##       Domneån Utl. Vättern   12   12   12   12   12   12   12   12   12   12   12   12   12
##                  Killingi   12   12   12   12   12   12   11   12   12   12   12   12   12
##                Kitkiöjoki    0   10   12   12   12   12   12   12    9   12   12   12   12
##          Klingavälsån Vomb   23   23   24   23   24   24   23   24   25   22   24   25   23
##                Kukkasjärvi   12   12   12   12   12   12   12   12   12   12   12   12   12
##                   Kvarnån   16   16   15   16   16   16   16   15   16   16   16   16   16
```

```
##            Laxtjärnsbäcken   15   16   16   16   16   16   16   16   16   16   16   16   16
##                    Lekarån    0   12   12   12   12   12   12   12   12   12   12   12   12
##                       Loån    0   11   12   12   12   12   12   12   12   12   12   12   12
##          Lommabäcken Nedre   15   16   14   16   12   16   15   16   16   15   16   16   11
##     Oradbäcken Ö.Kärtmyrås    0    0    4   12   12   12   12   12   12   12   12   12   12
##                     Ostvik   14   12   12   12   12   12   12   12   12   12   12   12   12
##              Övre Lansjärv   12   12   12   12   12   12   12   12   12   12   12   12   12
##           Sävjaån Ingvasta   12   12   12   12   12   12   11   12   12   12   12   12   12
##              Stormyrbäcken   16   16   16   16   16   16   16   16   16   16   16   16   16
##            Svartberget, C7   11   12   12   12   12   12   12   12   12   12   12   12   12
##          Tolångaån Tolånga   12   11   12   12   12   12   12   12   12   12   12   12   12
##  Västergarnsån, Liffedarve   12   11   12   12   12   12   12   12   12   12   12   12   11
##                   Västersel   12   12   12   12   12   12   12   12   12   12   12   12   12
##                       Virån    0   11   12   12   11   12   12   12   12   12   12   12   12
##                     Vretaån    0   11   12   12   12   12   12   12   12   12   12   12   12
```

Single time series plot

A simple time series plot for one series can be made like this

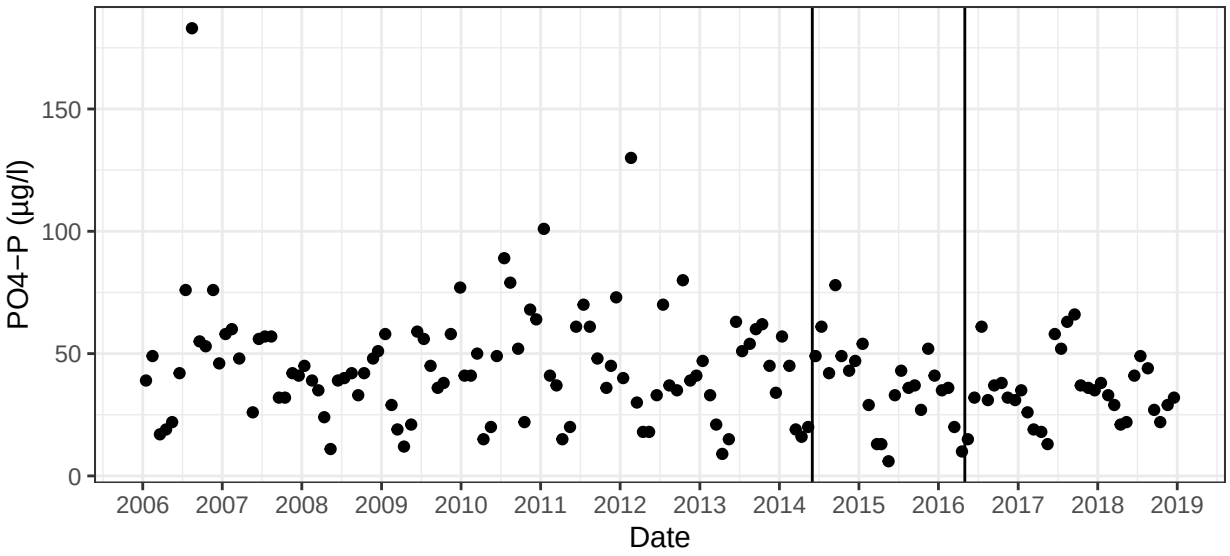
```
phosphate1%>%
  filter(Name=="Tolångaån Tolånga")%>%
  ggplot(aes(x=Date, y=`PO4-P (µg/l)`))+
  geom_point()
```

If we want all years written out on the x-axis we can specify this with the function \texttt{scale_x_date}, using \texttt{date_labels="%Y" only the year is written out. Also the grey background and white gridlines from the plot above do not look so nice. We replace them with white background.

```
phosphate1%>%
  filter(Name=="Tolångaån Tolånga")%>%
  ggplot(aes(x=Date, y=`PO4-P (µg/l)`))+
  geom_point()+
  scale_x_date(breaks="1 year", date_labels = "%Y")+
  theme_bw()
```

For phosphate an improvement in the analysis instrument was made in June 2014 and again in May 2016. In the plot we could indicate these time points as part of the data quality checking.

```
phosphate1%>%
  filter(Name=="Tolångaån Tolånga")%>%
  ggplot(aes(x=Date, y=`PO4-P (µg/l)`))+
  geom_point()+
  scale_x_date(breaks="1 year", date_labels = "%Y")+
  theme_bw()+
  geom_vline(xintercept=ymd("2014-06-01"))+
  geom_vline(xintercept=ymd("2016-05-01"))
```



Exercise A:

1. Make the same graph for Klingavälsån Vomb and see if you can see any effect of the instrument change.
2. Make a plot for Klingavälsån Vomb using the variable `TOC (mg/l)`. Do not include the vertical lines, as no change in instruments was made for this variable at these time points.

Visualising several series

If we want to plot several series at the same time as simple strategy would be to use different colors. Below we plot all data for TOC for all 26 water courses. The statement `col=Name` set different colors for each of the stations. With this we also get a legend av all station names and colors. To position this under the plot we add the statement `theme(legend.position='bottom')`.

```
phosphate1%>%
  ggplot(aes(x=Date, y=`TOC (mg/l)`, col=Name))+
  geom_point()+
  scale_x_date(breaks="1 year", date_labels = "%Y")+
  theme_bw()+
  theme(legend.position='bottom')
```

This is not so easy to overview. Maybe annual means would be better to illustrate the series[1]. Annual means can be computed using the functions \texttt{group_by()} and `summerise()`. The statement `na.rm=TRUE` is needed to get mean values for all non-NA values, otherwise the mean would be NA for all years that include at least one NA value.

```
phosphate1%>%
  group_by(Name, year)%>%
  summarise(TOC_annual_mean=mean(`TOC (mg/l)`, na.rm=TRUE))
```

```
## `summarise()` has grouped output by 'Name'. You can override using the `.groups` argument.
```

```
## # A tibble: 329 x 3
## # Groups:   Name [26]
##    Name        year TOC_annual_mean
##    <chr>      <dbl>           <dbl>
```

[1]Observe that computation of annual means only makes sense if the distribution of the observations over different years is similar. If during one year all observations are made in winter and during others all observations are made in summer, the comparison of their means would not be relevant

```
##  1 Akkarjåkkå  2006           2.84
##  2 Akkarjåkkå  2007           1.7
##  3 Akkarjåkkå  2008           1.98
##  4 Akkarjåkkå  2009           1.82
##  5 Akkarjåkkå  2010           1.59
##  6 Akkarjåkkå  2011           1.62
##  7 Akkarjåkkå  2012           1.74
##  8 Akkarjåkkå  2013           1.42
##  9 Akkarjåkkå  2014           1.38
## 10 Akkarjåkkå  2015           1.88
## # ... with 319 more rows
```

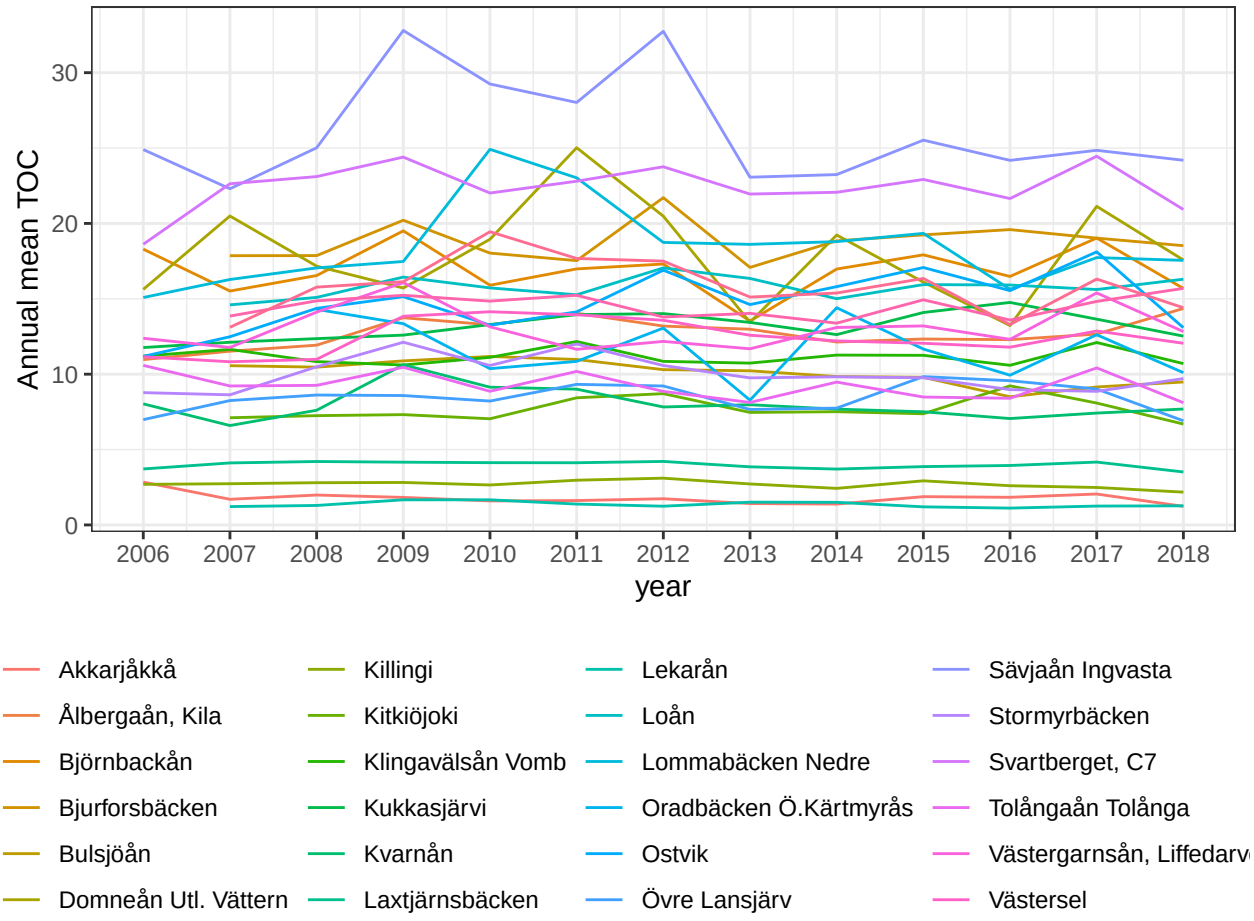
These computed annual means can be forwarded to the plot function. As we do not have a date variable anymore, the TOC means are plotted against the year, i.e. `x=year` and \texttt{y=TOC_annual_mean}.

```
phosphate1%>%
  group_by(Name, year)%>%
  summarise(TOC_annual_mean=mean(`TOC (mg/l)`, na.rm=TRUE))%>%
  ggplot(aes(x=year, y=TOC_annual_mean, col=Name))+
  geom_line()+
  theme_bw()+
  theme(legend.position='bottom')
```

```
## `summarise()` has grouped output by 'Name'. You can override using the `.groups` argument.
```

If we again want to write out each year on the x-axis we use now the statement \texttt{scale_x_continuous} and have to specify each point on the axis. We can also use a better y-axis title.

```
phosphate1%>%
  group_by(Name, year)%>%
  summarise(TOC_annual_mean=mean(`TOC (mg/l)`, na.rm=TRUE))%>%
  ggplot(aes(x=year, y=TOC_annual_mean, col=Name))+
  geom_line()+
  scale_x_continuous(breaks=(seq(from=2006, to=2018, by=1)))+
  ylab("Annual mean TOC")+
  theme_bw()+
  theme(legend.position='bottom')
```



Exercise B:

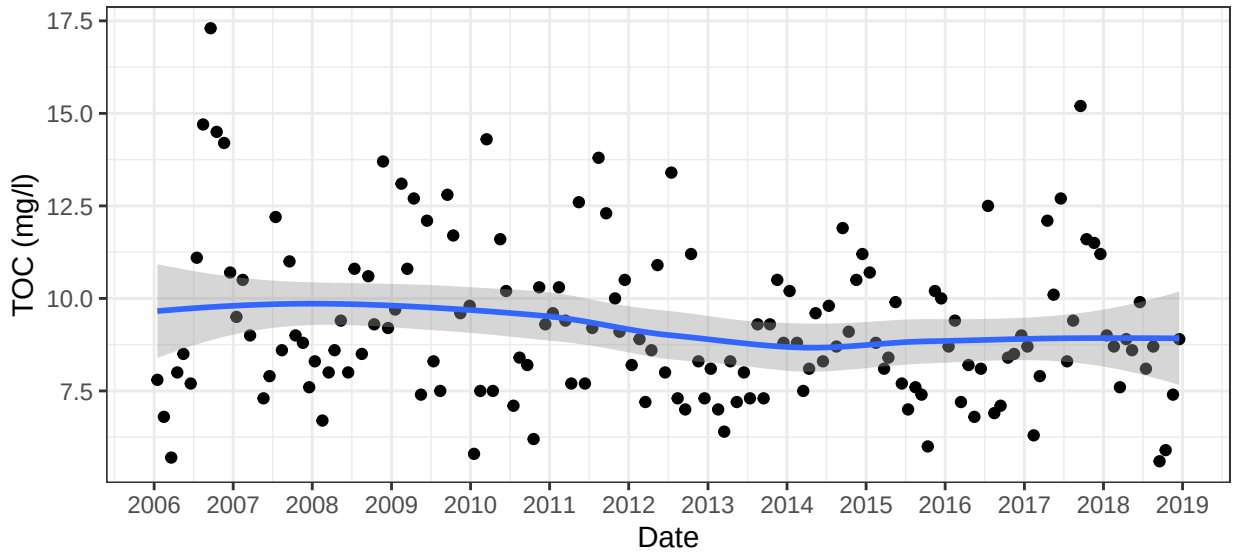
1. Plot the annual maxima of the TOC series for all stations.
2. Plot the annual maxima of Klingavälsån Vomb, Svartberget, C7, Tolångaån Tolånga and Vretaån. See footnote for a hint[2].
3. Plot the seasonal variation of TOC for each stations, i.e. compute one mean for each month and station and plot them.

Illustrating time series with smoothers

Another way to illustrate a time series is by enhancing the long-term changes (trend) in the series. This can be done by the function \texttt{geom_smooth}, e.g. for TOC in Tolongaån:

```
phosphate1%>%
  filter(Name=="Tolångaån Tolånga")%>%
  ggplot(aes(x=Date, y=`TOC (mg/l)`))+
  geom_point()+
  geom_smooth()+
  scale_x_date(breaks="1 year", date_labels = "%Y")+
  theme_bw()
```

[2]To combine several conditons use |, e.g. `Name == "Klingavälsån Vomb" | Name == "Svartberget, C7"`

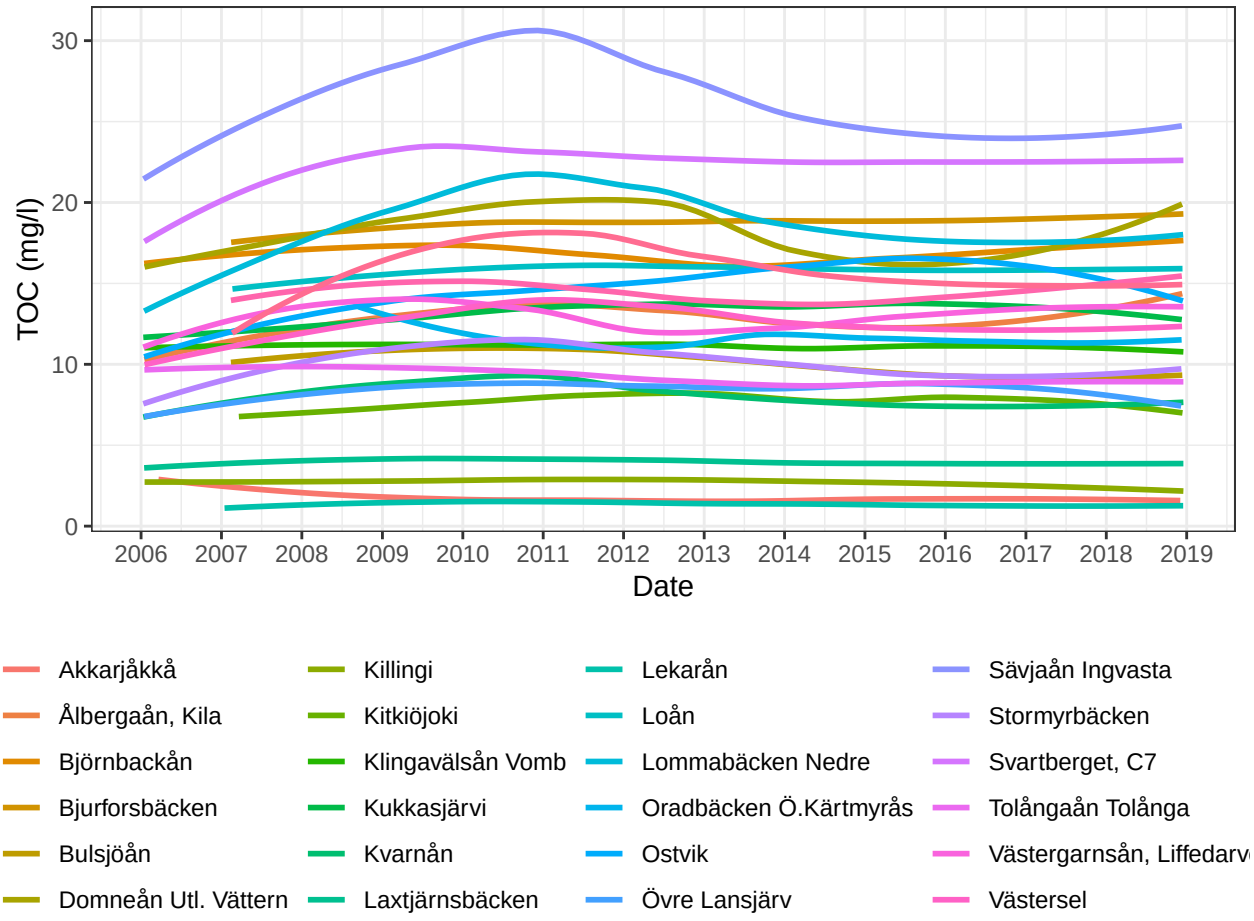


Or for all series:

```
phosphate1%>%
  ggplot(aes(x=Date, y=`TOC (mg/l)`, col=Name))+
  geom_point()+
  geom_smooth(se=FALSE)+
  scale_x_date(breaks="1 year", date_labels = "%Y")+
  theme_bw()+
  theme(legend.position='bottom')
```

Or plotting only the smoothed out lines:

```
phosphate1%>%
  ggplot(aes(x=Date, y=`TOC (mg/l)`, col=Name))+
  geom_smooth(se=FALSE)+
  scale_x_date(breaks="1 year", date_labels = "%Y")+
  theme_bw()+
  theme(legend.position='bottom')
```



Solutions to exercises A:

1.

```
phosphate1%>%
  filter(Name=="Klingavälsån Vomb")%>%
  ggplot(aes(x=Date, y=`PO4-P (µg/l)`))+
  geom_point()+
  scale_x_date(breaks="1 year", date_labels = "%Y")+
  theme_bw()+
  geom_vline(xintercept=ymd("2014-06-01"))+
  geom_vline(xintercept=ymd("2016-05-01"))
```

2.

```
phosphate1%>%
  filter(Name=="Klingavälsån Vomb")%>%
    ggplot(aes(x=Date, y=`TOC (mg/l)`))+
  geom_point()+
  scale_x_date(breaks="1 year", date_labels = "%Y")+
  theme_bw()
```

Exercise B:

```
phosphate1%>%
  group_by(Name, year)%>%
  summarise(TOC_annual_mean=max(`TOC (mg/l)`))%>%
  ggplot(aes(x=year, y=TOC_annual_mean, col=Name))+
  geom_line()+
  scale_x_continuous(breaks=(seq(from=2006, to=2018, by=1)))+
  ylab("Annual maximum TOC")+
  theme_bw()+
  theme(legend.position='bottom')
```

```
## `summarise()` has grouped output by 'Name'. You can override using the `.groups` argument.
```

```
## Warning: Removed 17 row(s) containing missing values (geom_path).
```

2.

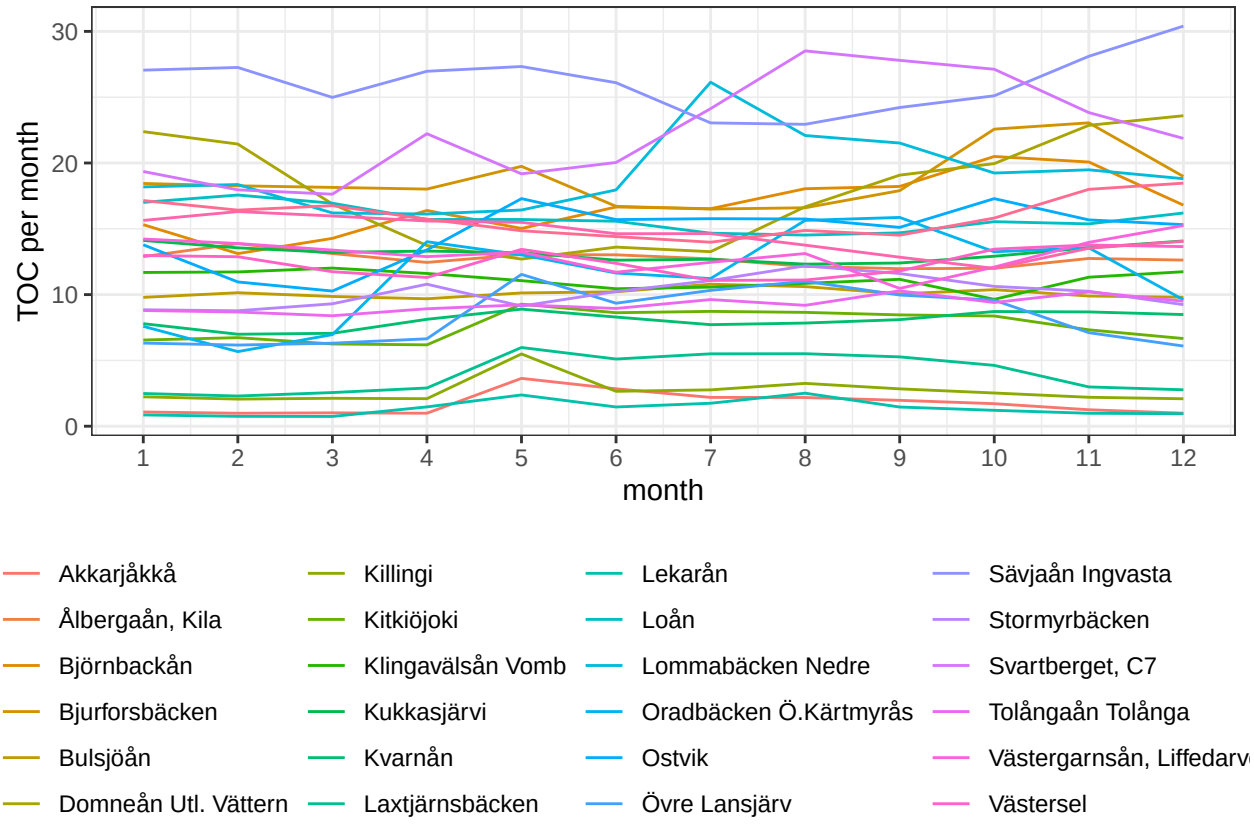
Klingavälsån Vomb, Svartberget, C7, Tolångaån Tolånga and Vretaån

```
phosphate1%>%
  group_by(Name, year)%>%
  summarise(TOC_annual_mean=max(`TOC (mg/l)`, na.rm=TRUE))%>%
  filter(Name=="Klingavälsån Vomb" | Name=="Svartberget, C7" | Name=="Tolångaån Tolånga" | Name=="Vreta
  ggplot(aes(x=year, y=TOC_annual_mean, col=Name))+
  geom_line()+
  scale_x_continuous(breaks=(seq(from=2006, to=2018, by=1)))+
  ylab("Annual maximum TOC")+
  theme_bw()+
  theme(legend.position='bottom')
```

```
## `summarise()` has grouped output by 'Name'. You can override using the `.groups` argument.
```

```
phosphate1%>%
  group_by(Name, month)%>%
  summarise(TOC_monthly_mean=mean(`TOC (mg/l)`, na.rm=TRUE))%>%
  ggplot(aes(x=month, y=TOC_monthly_mean, col=Name))+
  geom_line()+
  scale_x_continuous(breaks=(seq(from=1, to=12, by=1)))+
  ylab("TOC per month")+
  theme_bw()+
  theme(legend.position='bottom')
```

```
## `summarise()` has grouped output by 'Name'. You can override using the `.groups` argument.
```


TOC per month
30
20
10
0
1
2
3
4
5
6
7
8
9
10
11
12
month
Akkarjåkkå
Ålbergaån, Kila
Björnbackån
Bjurforsbäcken
Bulsjöån
Domneån Utl. Vättern
Killingi
Kitkiöjoki
Klingavälsån Vomb
Kukkasjärvi
Kvarnån
Laxtjärnsbäcken
Lekarån
Loån
Lommabäcken Nedre
Oradbäcken Ö.Kärtmyrås
Ostvik
Övre Lansjärv
Sävjaån Ingvasta
Stormyrbäcken
Svartberget, C7
Tolångaån Tolånga
Västergarnsån, Liffedarv
Västersel