starting to view the energy data

Andy South

6 August 2016

```r
library(ggplot2)

# read data
# as.is=TRUE stops conversion of chars to factors
tst <- read.csv('USA_CA_Long.Beach-Daugherty.Field.722970_TMY3_BASE.csv', as.is=TRUE)

# the column names are quite ugly with lots of '.'
str(tst)
```

```
## 'data.frame':    8760 obs. of  14 variables:
##  $ Date.Time                                      : chr  " 01/01  01:00:00" " 01/01  02:00:00" " 01/0
##  $ Electricity.Facility..kW..Hourly.              : num  0.643 0.531 0.487 0.474 0.473 ...
##  $ Gas.Facility..kW..Hourly.                      : num  0.367 0.297 0.28 0.286 0.343 ...
##  $ Heating.Electricity..kW..Hourly.               : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Heating.Gas..kW..Hourly.                       : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Cooling.Electricity..kW..Hourly.               : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ HVACFan.Fans.Electricity..kW..Hourly.          : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Electricity.HVAC..kW..Hourly.                  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Fans.Electricity..kW..Hourly.                  : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ General.InteriorLights.Electricity..kW..Hourly.: num  0.1286 0.075 0.0536 0.0536 0.0536 ...
##  $ General.ExteriorLights.Electricity..kW..Hourly.: num  0.0264 0.0154 0.011 0.011 0.011 ...
##  $ Appl.InteriorEquipment.Electricity..kW..Hourly.: num  0.0929 0.0762 0.0623 0.054 0.0658 ...
##  $ Misc.InteriorEquipment.Electricity..kW..Hourly.: num  0.38 0.35 0.345 0.341 0.328 ...
##  $ Water.Heater.WaterSystems.Gas..kW..Hourly.     : num  0.341 0.273 0.256 0.262 0.321 ...
```

```r
#date format seems to be month/day hr:mi:se

library(lubridate)
```

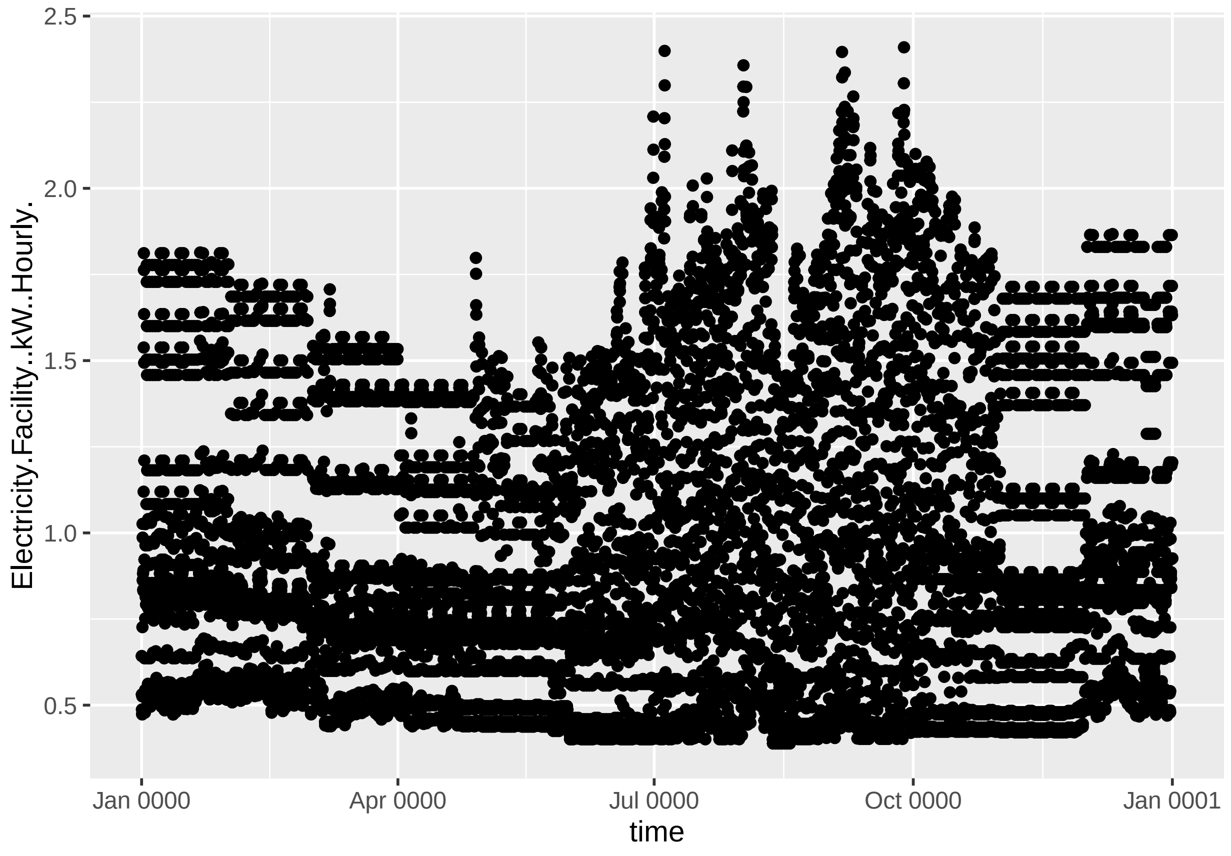
```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date
```

```r
# to avoid failures in time conversion
tst$Date.Time <- gsub('24:00:00', '23:59:59', tst$Date.Time)

# converting time characters to a recognised time format
tst$time <- parse_date_time(tst$Date.Time, "m!* d! H! M! S!")
# the rows with 24:00:00 convert to NA
# e.g.
#parse_date_time(tst$Date.Time[24], "m!* d! H! M! S!")
#parse_date_time(" 01/01  24:00:00", "m!* d! H! M! S!")
```

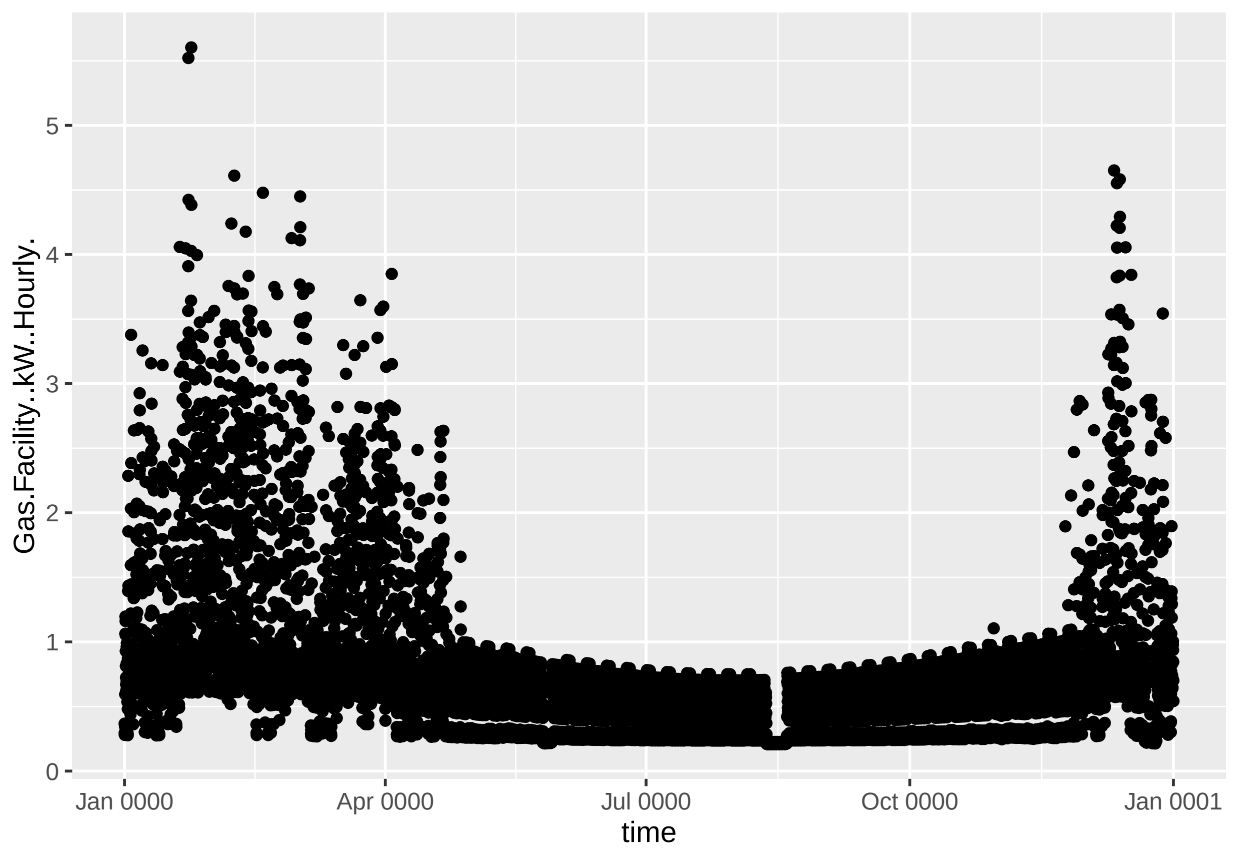
```
# as a hack get around I could set to 1s earlier
# (BEWARE this could create strange issues if looking at intervals later)
parse_date_time(" 01/01  23:59:59", "m!* d! H! M! S!")
```

```
## [1] "0000-01-01 23:59:59 UTC"
```

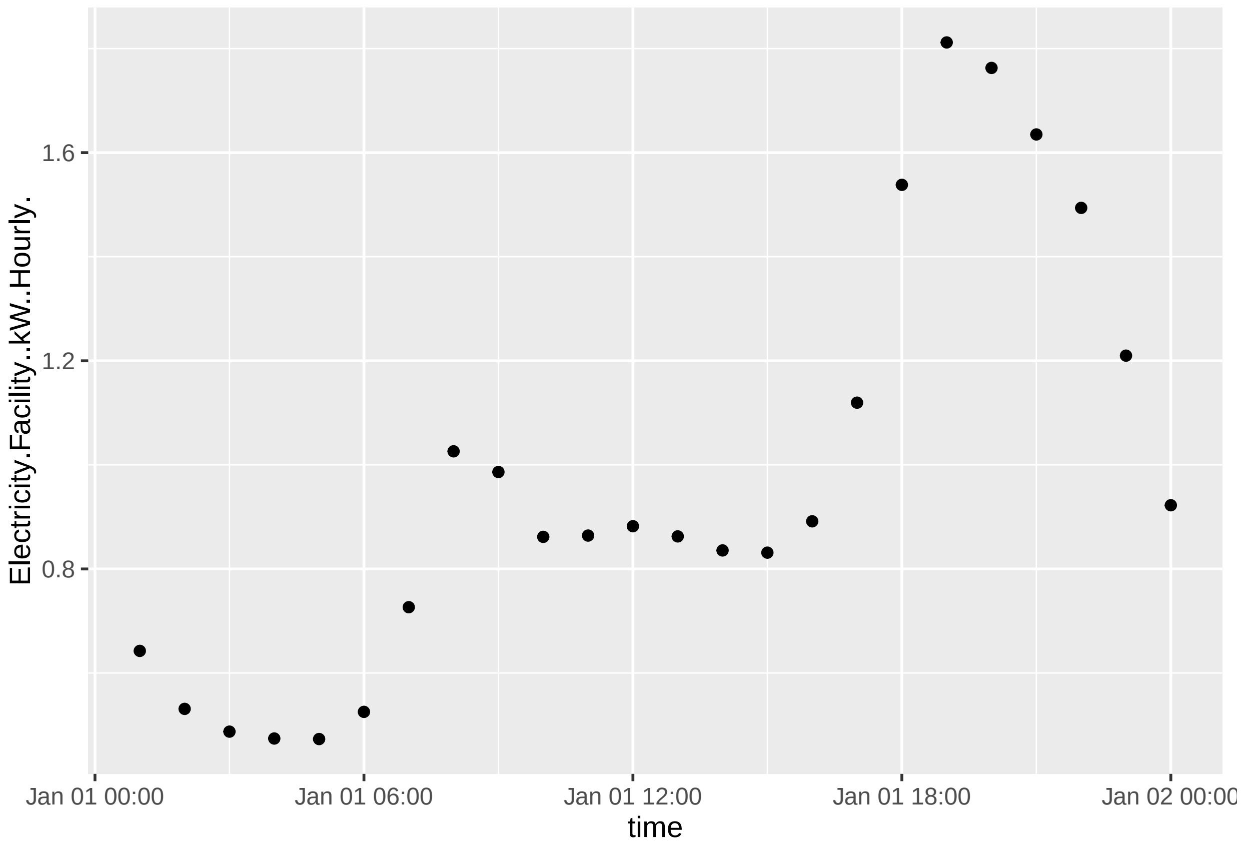
```
# so before the time conversion I can do a string replace
# gsub(pattern,replacement,x)
#tst$Date.Time <- gsub('24:00:00', '23:59:59', tst$Date.Time)

# create a week column to use in facetting below
tst$week <- week(tst$time)

# plot data for the 2nd column for whole year, some strange patterns ...
# difficult to see whats happening across year, too many points
ggplot(tst, aes_string(x='time',y=names(tst)[2])) + geom_point()
```



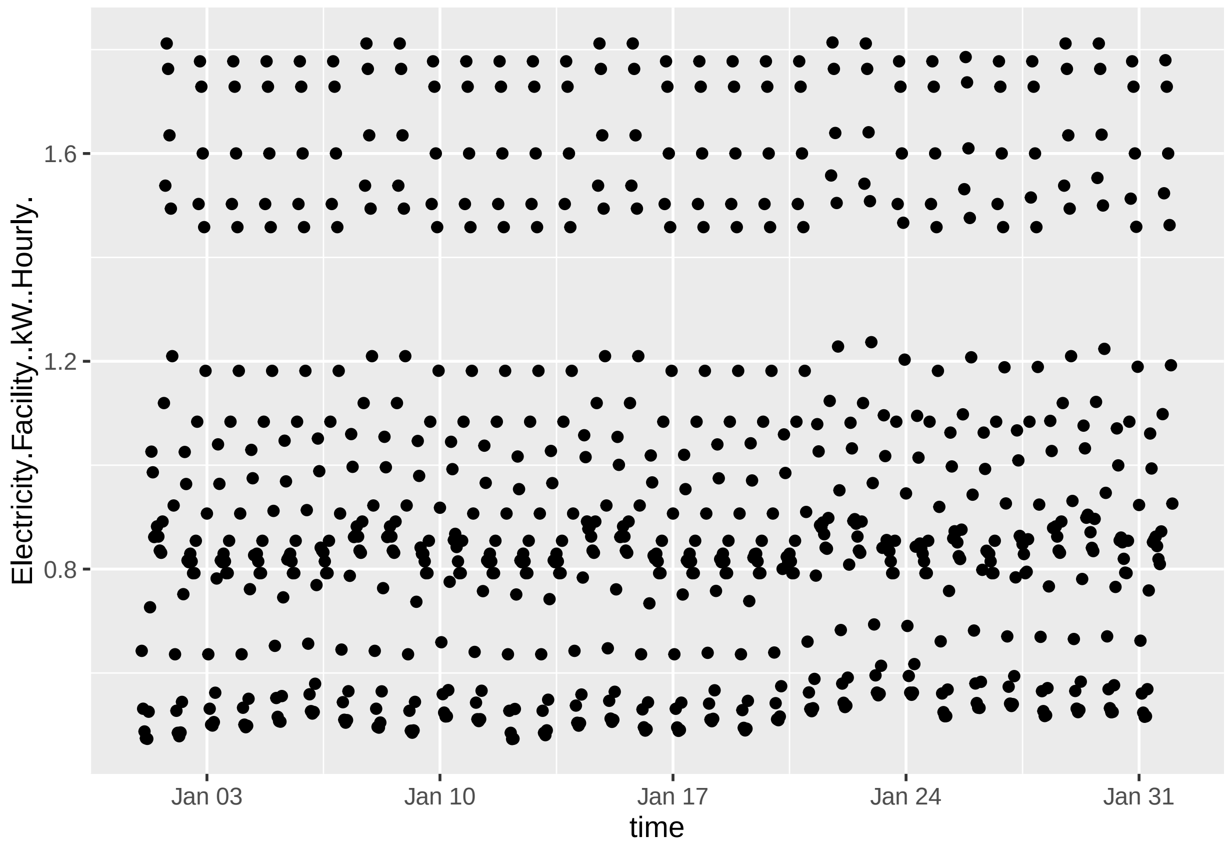
```
# same for next column using explicit column naming
# gas has very diff pattern to electric
# can see when heating not on in winter
# in winter use seems like it may be dependent on temp more variable than electric which seems more con
ggplot(tst, aes(x=time,y=Gas.Facility..kW..Hourly.)) + geom_point()
```



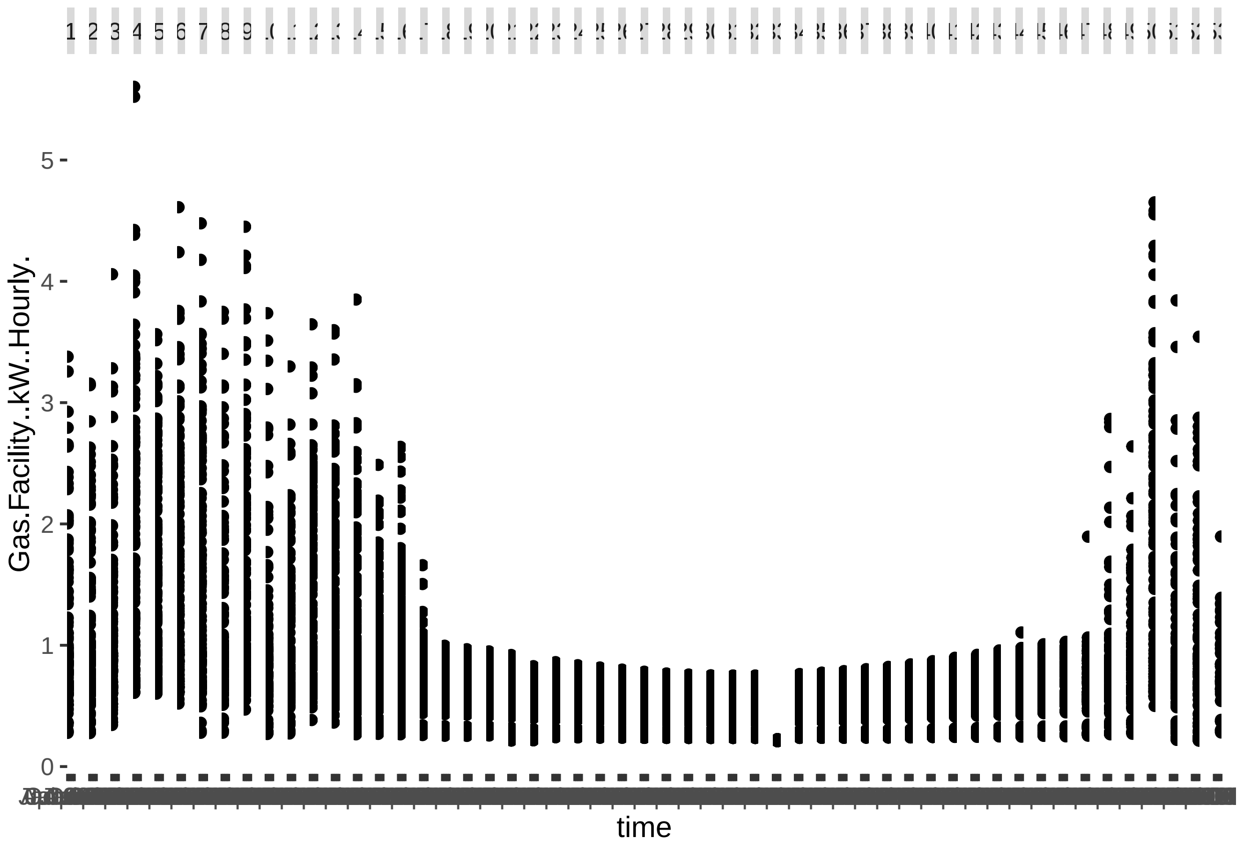
```
# are there repeats per time step ? try plotting a day to test
# there is just one point per day (& 365 rows of NAs)
tstday <- tst[ tst$time < ymd('0000-01-02', tz='UTC'),]
ggplot(tstday, aes_string(x='time',y=names(tst)[2])) + geom_point()
```



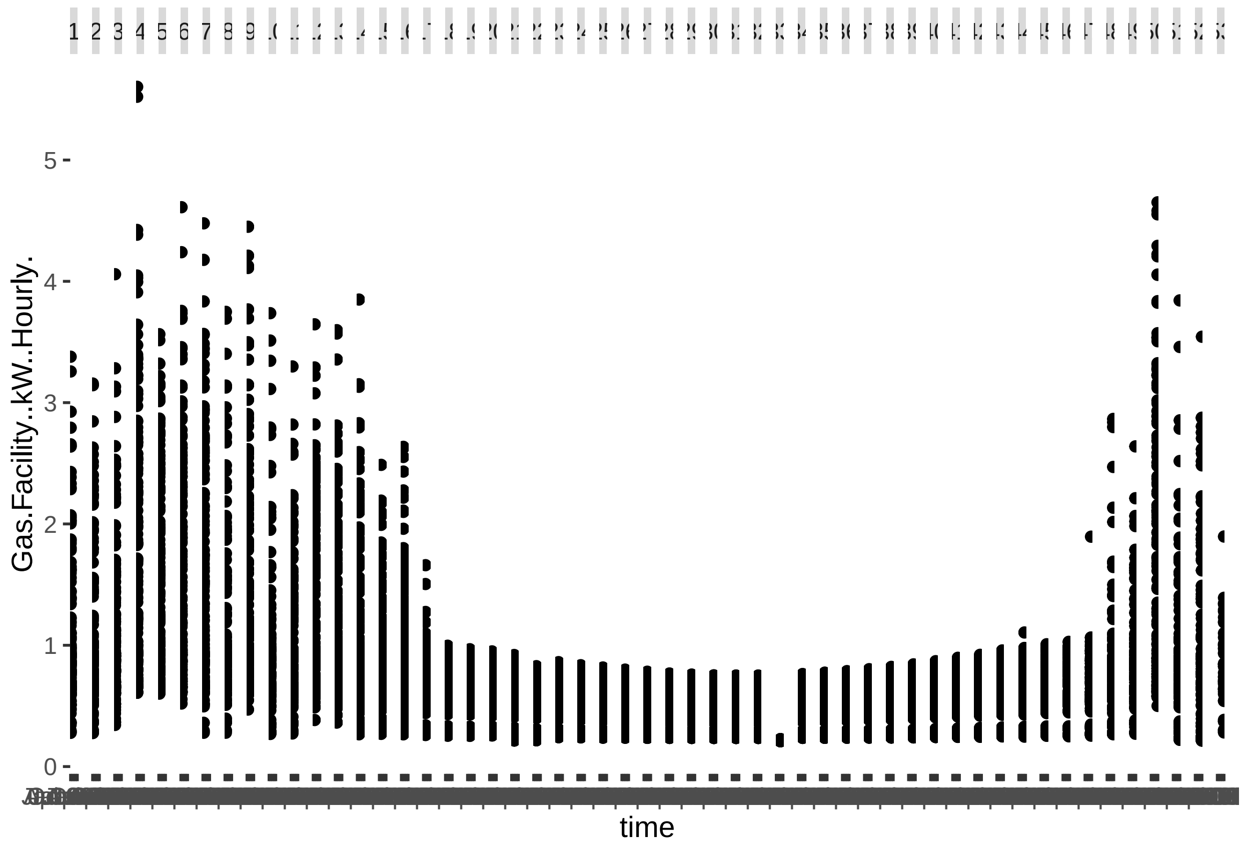
```
# now try january, can see seemingly higher set values at weekends
# consistent patterns suggest constant presets
tstjan <- tst[ tst$time < ymd('0000-02-01', tz='UTC'),]
ggplot(tstjan, aes_string(x='time',y=names(tst)[2])) + geom_point()
```



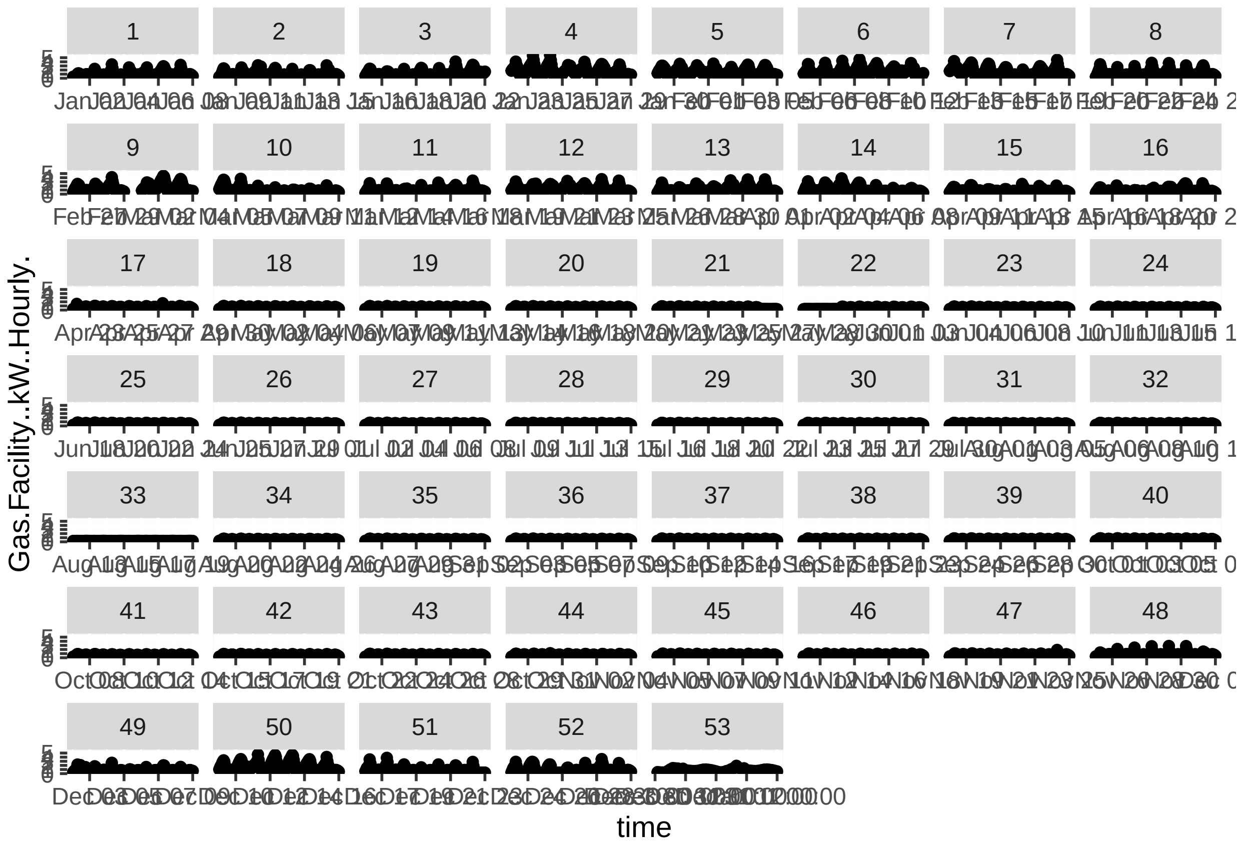
```
# lots of different ways i could dimension the data now
# e.g.
# days of the week

# can I facet by weeks ?
# these below both failed
#ggplot(tst, aes_string(x='time',y=names(tst)[2])) +
#ggplot(tst, aes(x=time,y=Gas.Facility..kW..Hourly.)) +
#  geom_point() +
#  facet_grid(. ~ week(time))

# instaed after I had created a week column above
# this sort of worked (but just one row so not yet very informative)
ggplot(tst, aes(x=time,y=Gas.Facility..kW..Hourly.)) +
  geom_point() +
  facet_grid( ~ week)
```



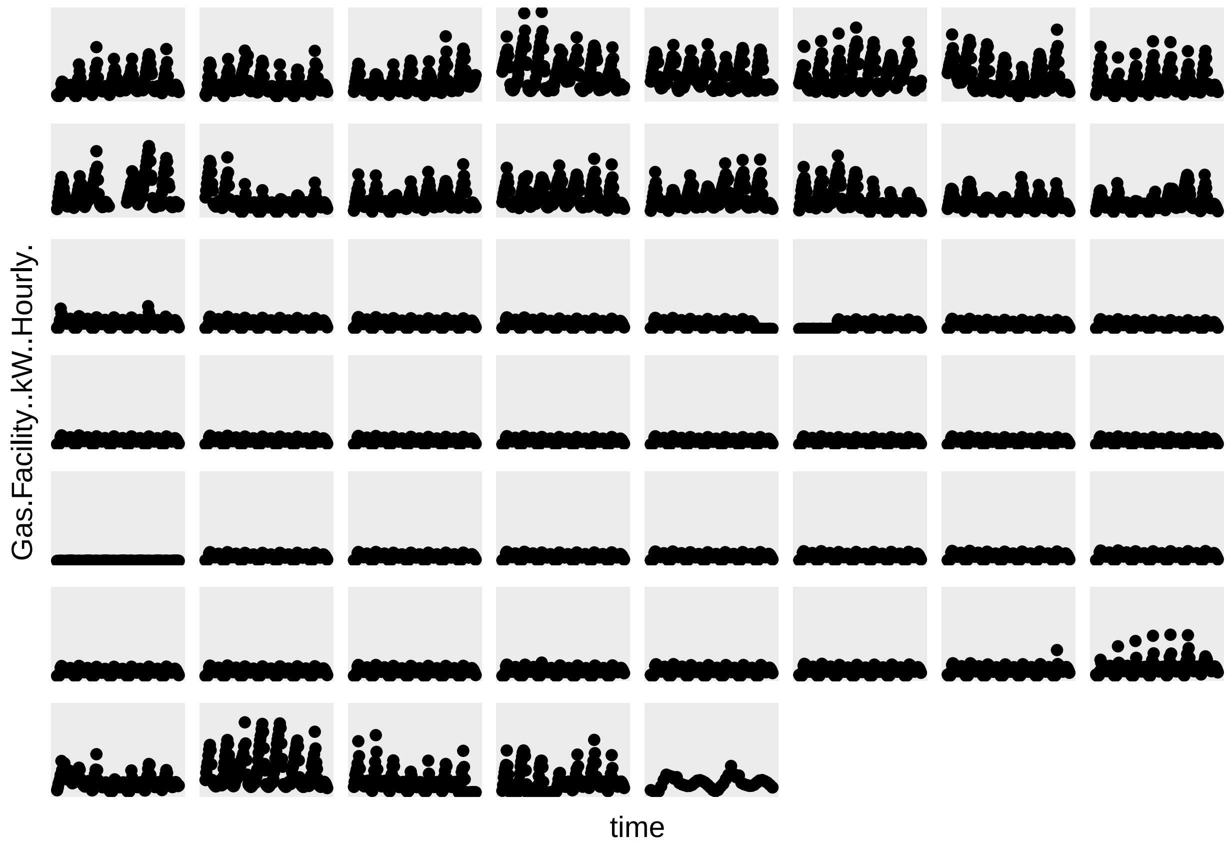
```r
# better with facet_warp have a grid of facets, but the x scale is constant between all
ggplot(tst, aes(x=time,y=Gas.Facility..kW..Hourly.)) +
  geom_point() +
  facet_grid( ~ week)
```



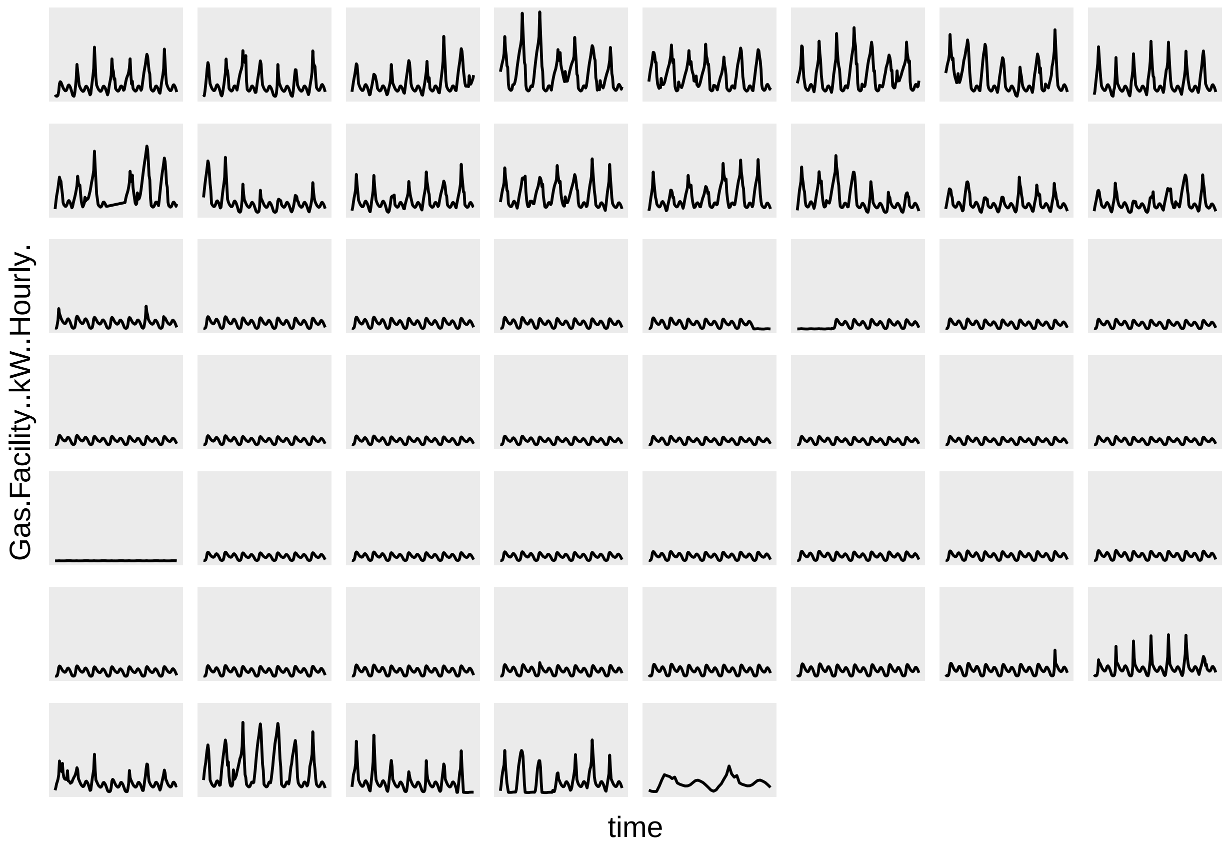
```r
# allow x scale to vary, better still messy
ggplot(tst, aes(x=time,y=Gas.Facility..kW..Hourly.)) +
  geom_point() +
  facet_wrap( ~ week, scales = 'free_x' )
```



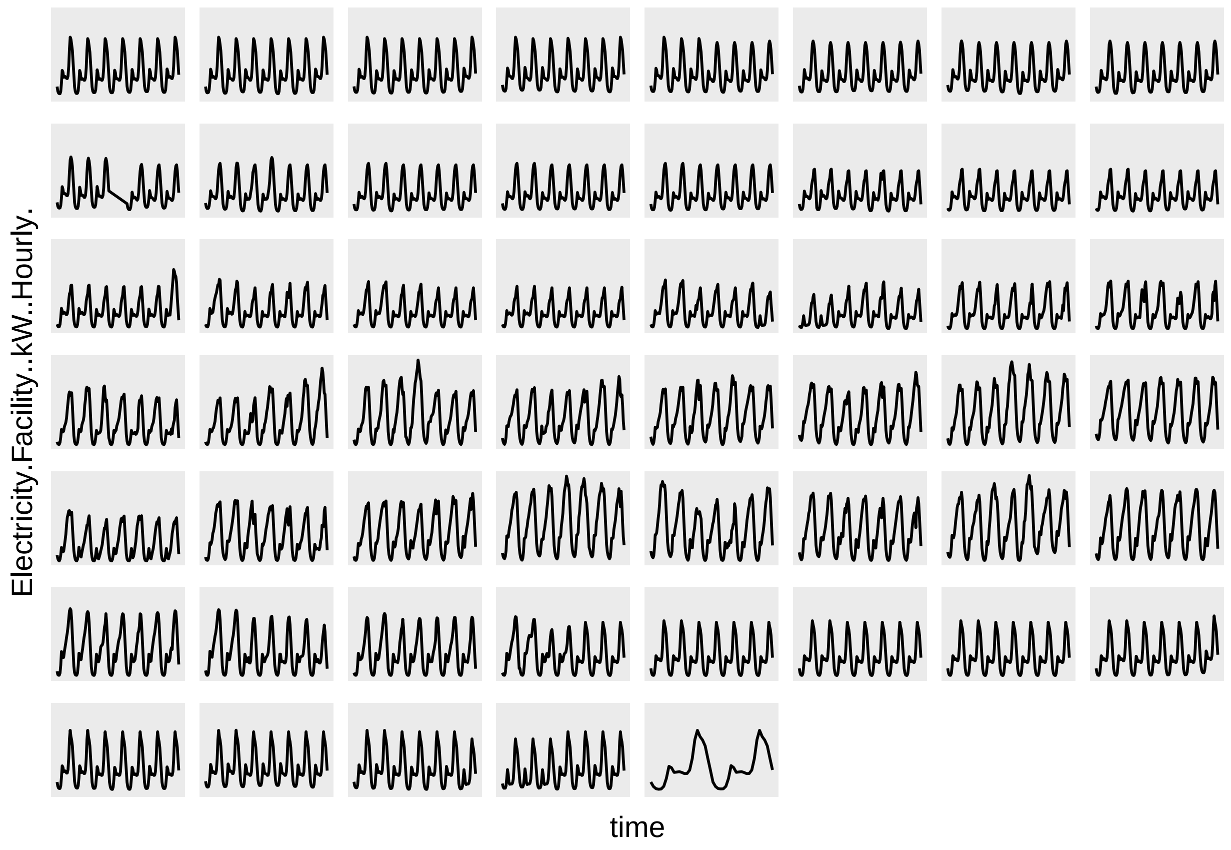
```
#initially got :
#Error in seq.int(0, to0 - from, by) : 'to' cannot be NA, NaN or infinite
#so had to remove NAs above

# try to make cleaner by removing non-data ink
ggplot(tst, aes(x=time,y=Gas.Facility..kW..Hourly.)) +
  geom_point() +
  facet_wrap( ~ week, scales = 'free_x' ) +
  theme(axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.line = element_blank(),
        strip.text = element_blank(),
        panel.grid = element_blank()
        )
```



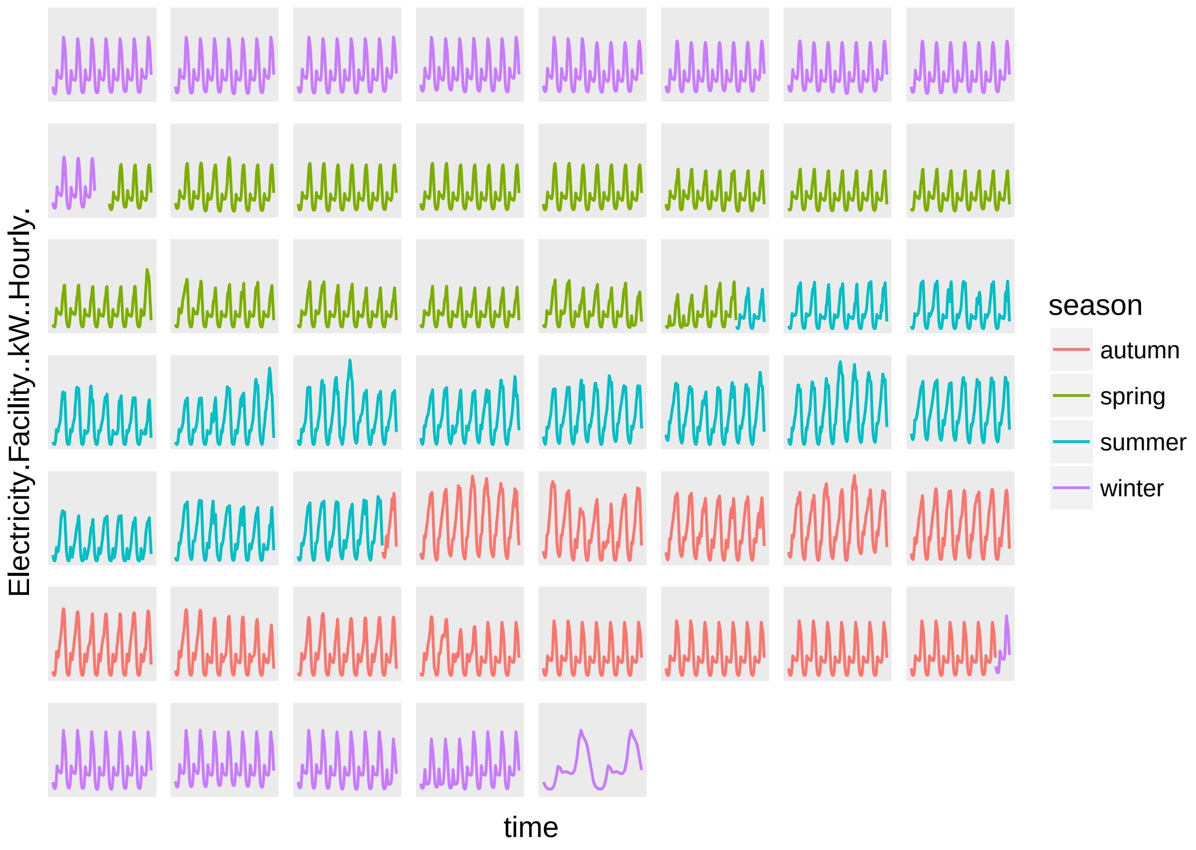
```r
# change from point to line - nice !
# daily patterns, seemingly no weekend effect
# can clearly see the presumably 2 coldest weeks in the year, the 4th & 4th from last
# could colour by season or month ...
ggplot(tst, aes(x=time,y=Gas.Facility..kW..Hourly.)) +
  geom_line() +
  facet_wrap( ~ week, scales = 'free_x' ) +
  theme(axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.line = element_blank(),
        strip.text = element_blank(),
        panel.grid = element_blank()
  )
```



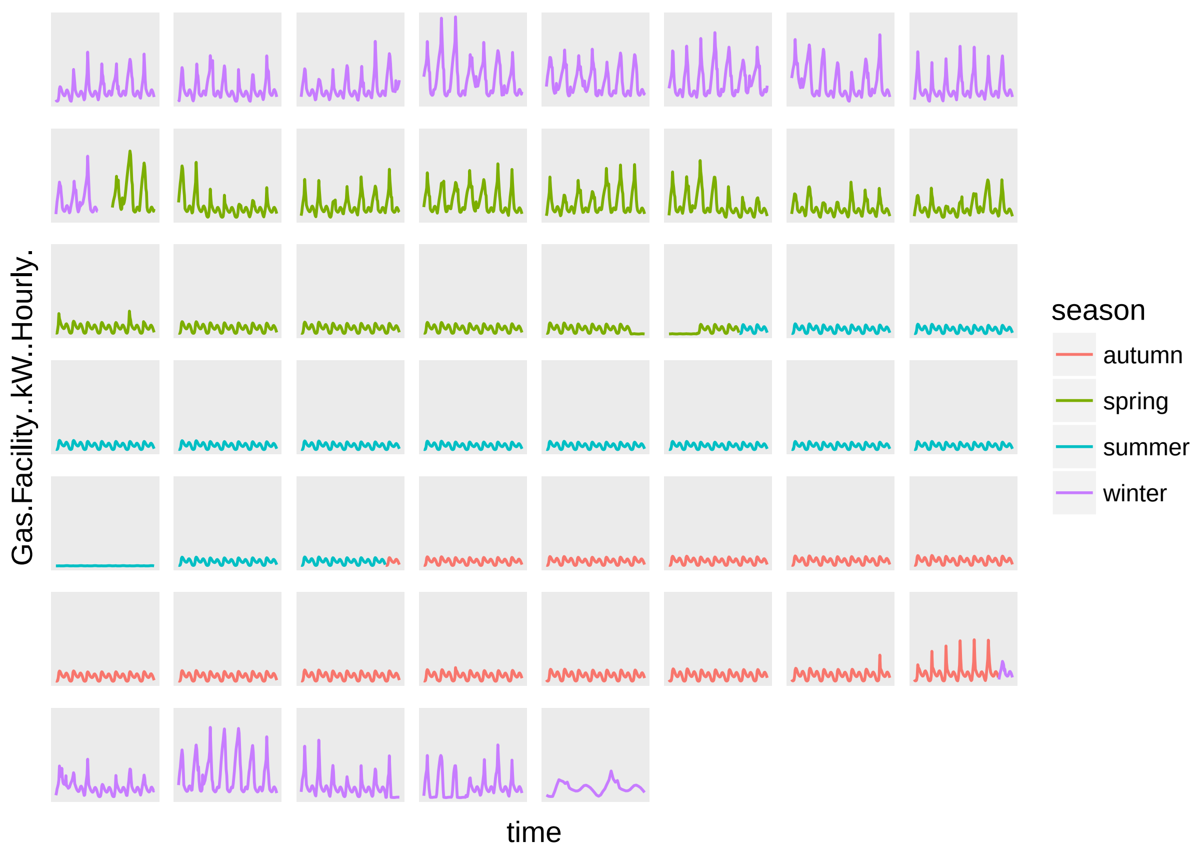
```r
# same as previous but for electric, very different pattern
# bigger magnitude of daily change in summer. aircon ?
# last facet looks different because it's not a whole week, would want to correct that
ggplot(tst, aes(x=time, y=Electricity.Facility..kW..Hourly.)) +
  geom_line() +
  facet_wrap( ~ week, scales = 'free_x' ) +
  theme(axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.line = element_blank(),
        strip.text = element_blank(),
        panel.grid = element_blank()
  )
```



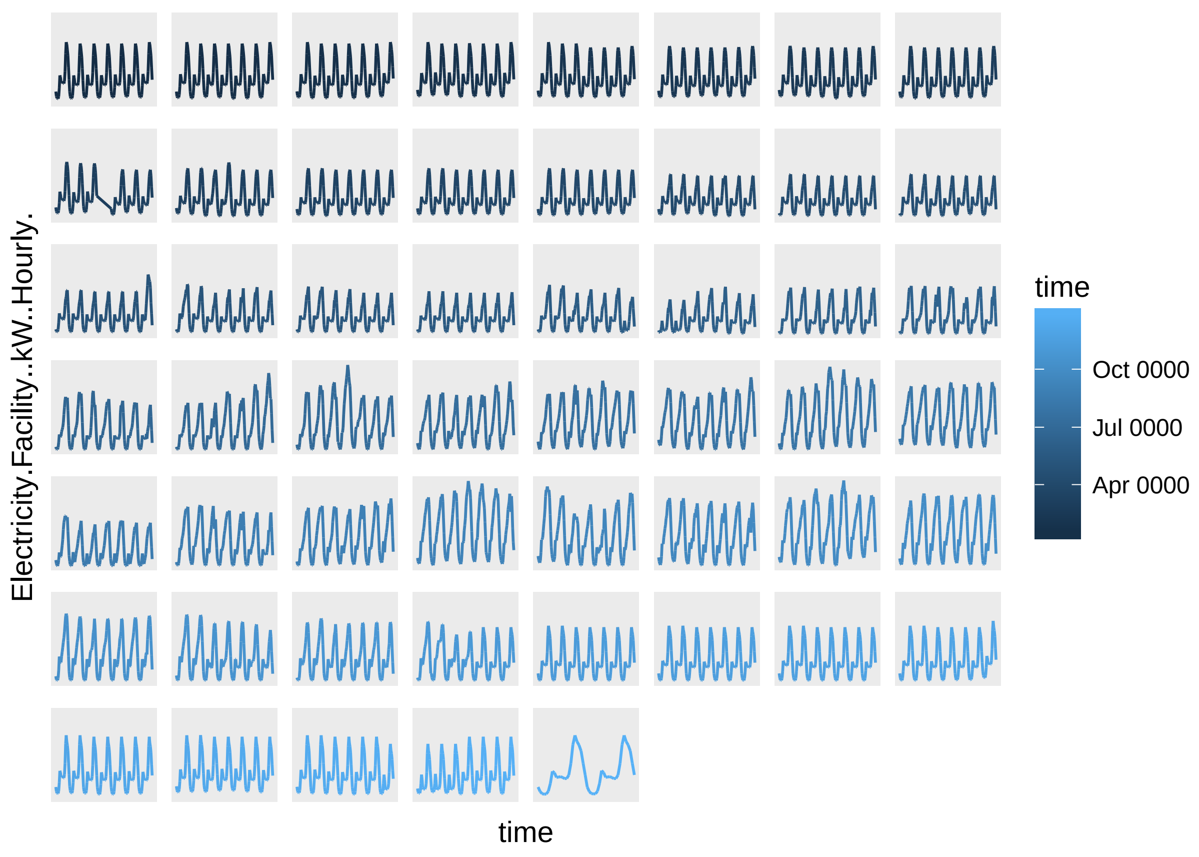
```r
# get month & approx season
tst$month <- month(tst$time, label = TRUE)
tst$season <- ifelse(tst$month %in% c("Dec","Jan","Feb"), "winter",
                     ifelse( tst$month %in% c("Mar","Apr","May"), "spring",
                     ifelse( tst$month %in% c("Jun","Jul","Aug"), "summer", "autumn")))

# add seasonal colour to the plots
# actually i think i find the colour slightly distracting
ggplot(tst, aes(x=time, y=Electricity.Facility..kW..Hourly., colour=season)) +
  geom_line() +
  facet_wrap( ~ week, scales = 'free_x' ) +
  theme(axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.line = element_blank(),
        strip.text = element_blank(),
        panel.grid = element_blank()
  )
```



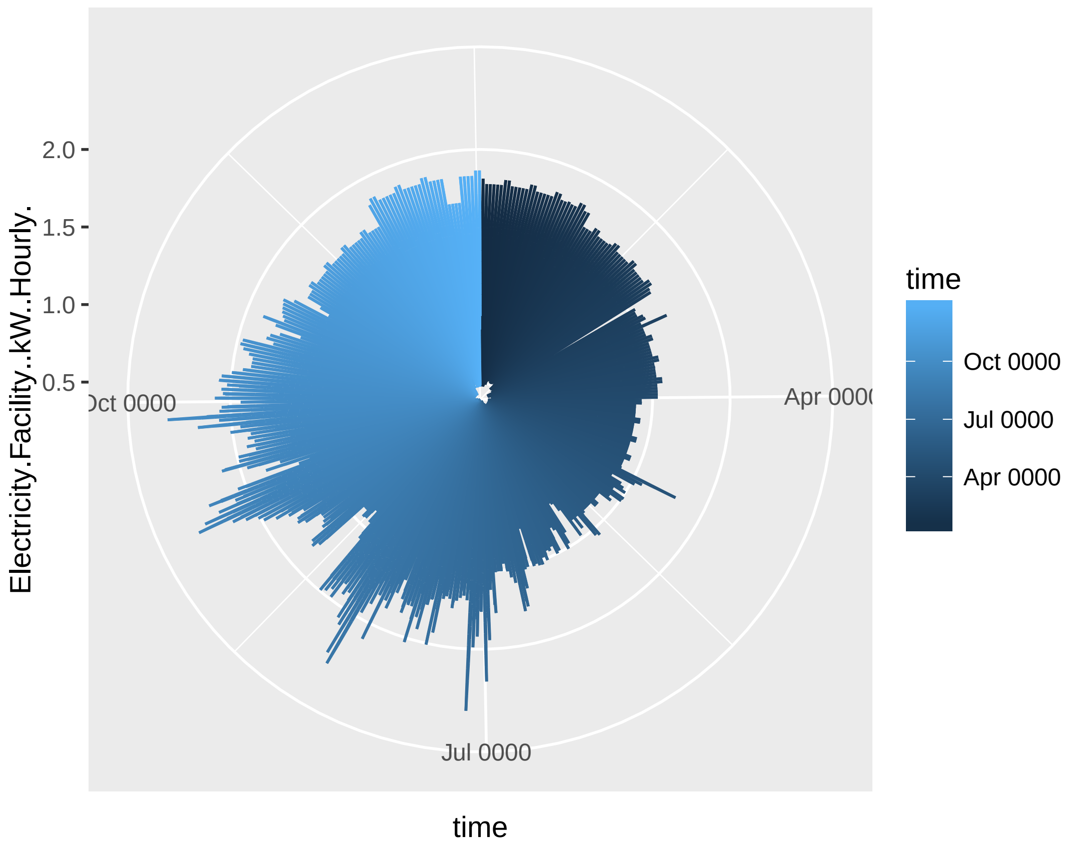
```r
# add seasonal colour to gas plot
# here the colour is more useful in telling seemingly when heating switched on & off in year
ggplot(tst, aes(x=time, y=Gas.Facility..kW..Hourly., colour=season)) +
  geom_line() +
  facet_wrap( ~ week, scales = 'free_x' ) +
  theme(axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.line = element_blank(),
        strip.text = element_blank(),
        panel.grid = element_blank()
  )
```



```r
# i notice there seems to be a scale_colour_date and datetime in ggplot2
# perhaps they get used by default on tim data like this
# i still find this slightly less clear than the single colour plot
# but would want a way of showing the reader what the seasons are
# weather icons would be cool and probably fairly easy to do
ggplot(tst, aes(x=time, y=Electricity.Facility..kW..Hourly., colour=time)) +
  geom_line() +
  facet_wrap( ~ week, scales = 'free_x' ) +
  theme(axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.line = element_blank(),
        strip.text = element_blank(),
        panel.grid = element_blank()
  )
```



```
# try a circular polar coordinate plot
# initially similar to above but with coord_polar instead of the facet_wrap()
# ok for seeing maximums
# not very good for seeing minimums or daily patterns because too many data
ggplot(tst, aes(x=time, y=Electricity.Facility..kW..Hourly., colour=time)) +
  geom_line() +
  coord_polar()
```



try plotting all columns

might want to reshape data to long format to facilitate that

maybe create a function to do for each column so I can pass each file to it