ThermogramBaseline Demo

Package Setup and Data

For some of the work below, we must use the package "tidyverse".

```
library(tidyverse, quietly = TRUE)
```

Here we load the package and look at the data. Data must be formatted in this way:

```
library(ThermogramBaseline)
```

```
data(UrineWorking, package = "ThermogramBaseline")
head(UrineWorking)
```

SampleID	Temperature	dCp
1a	20.00053	-1.174067
1a	20.01116	-2.676789
1a	20.01872	-3.461386
1a	20.02728	-3.375907
1a	20.03676	-3.123136
1a	20.04711	-2.763205

Analysis of One Thermogram

Each function can only take in one thermogram sample at a time so we must filter for sample 1a and filter between the temperatures 45-90

```
SampleIDs <- UrineWorking %>% pull(SampleID) %>% unique() %>% as.vector()

Sample.1 <- UrineWorking %>% filter(SampleID == SampleIDs[1]) %>%
  filter(between(Temperature, 45,90))
```

Now that we have our first sample, lets get our two endpoints for baseline detection.

```
endpoints <- endpoint.detection(x = Sample.1, exclusion.lwr = 60,
                                exclusion.upr = 80, point.selection = "innermost",
                                explicit = FALSE)
```

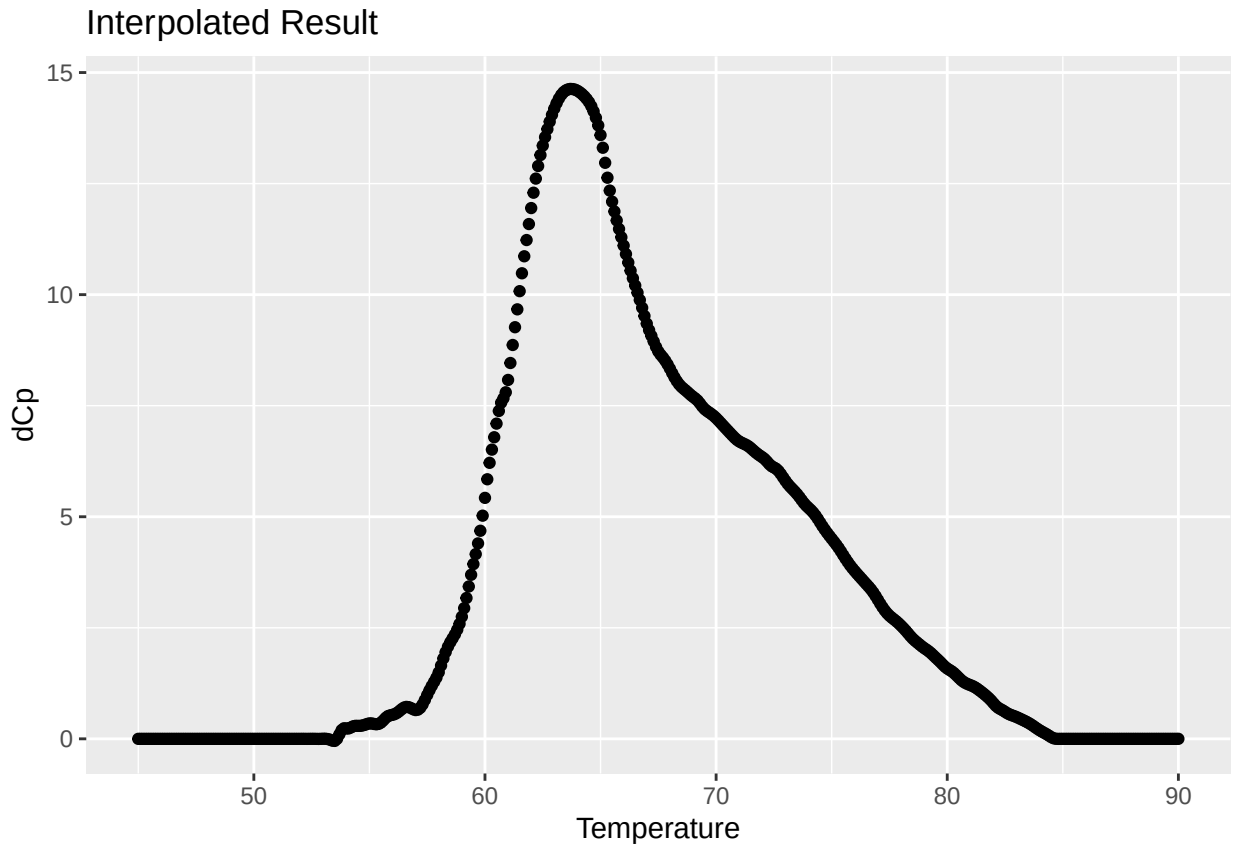
lower	upper	method
53.24153	84.63846	innermost

The model selected the temperatures 53.24153 and 84.63846 as our endpoints for our baseline subtraction. Next lets subtract the baseline using these endpoints.

```
baseline <- baseline.subtraction.byhand(x= Sample.1, lwr.temp = endpoints$lower,
                                        upr.temp = endpoints$upper, plot.on = FALSE)
```

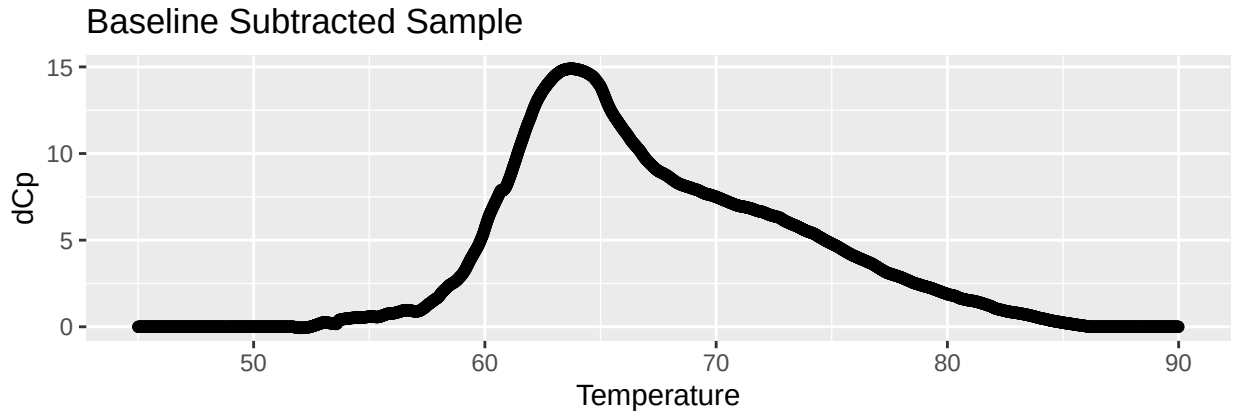
Now that we have subtracted the baseline, we will interpolate the data onto a grid of set temperatures. Notice plot.on is true by

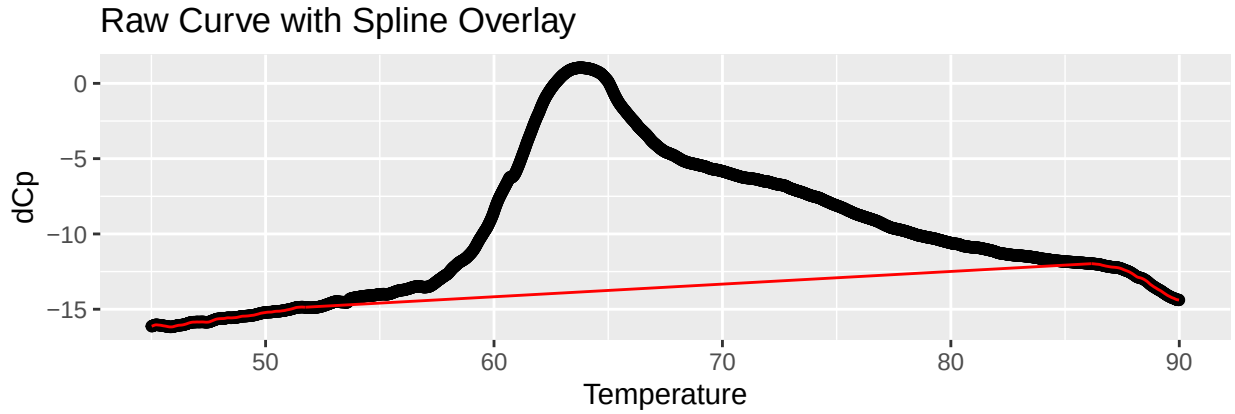
```
final <- final.sample.interpolate(x=baseline,grid.temp = seq(from = 45,to = 90,by = 0.1))
```

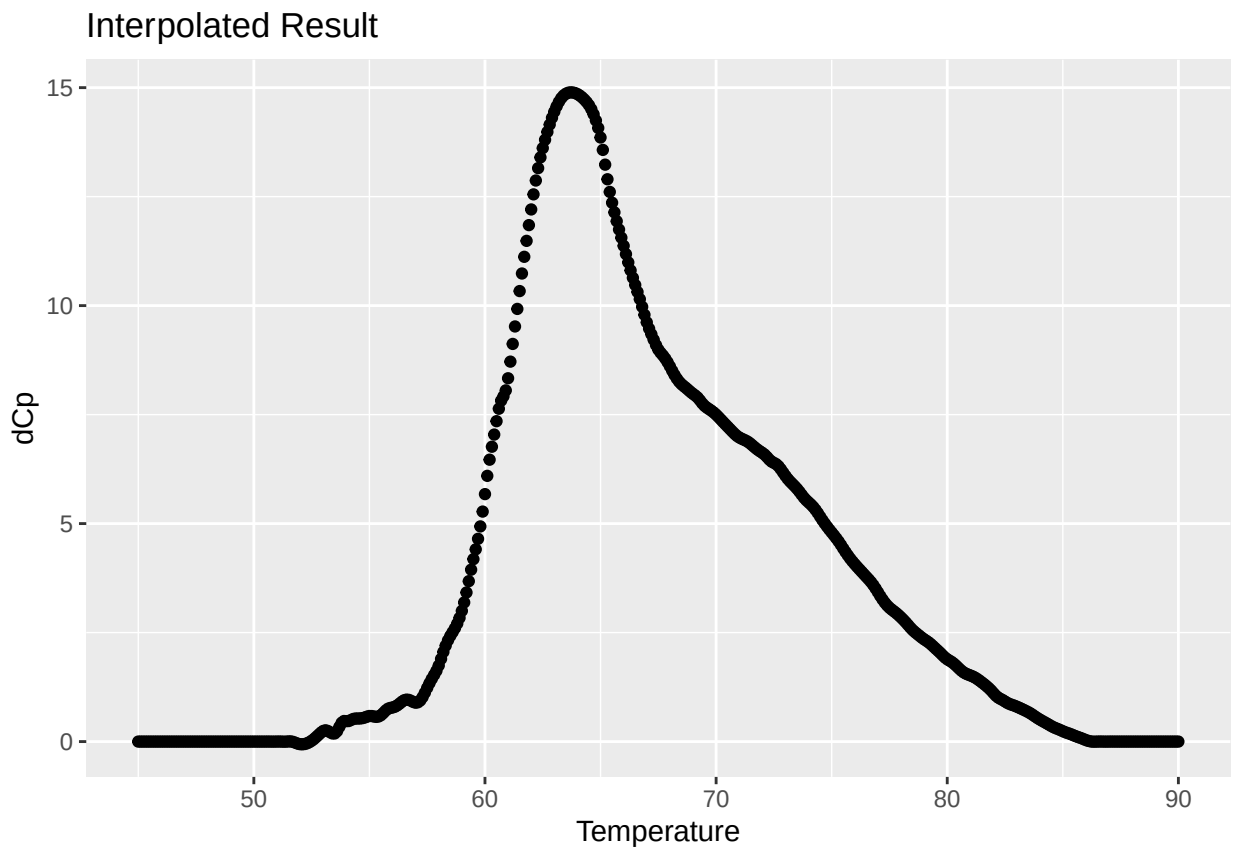


Here we have our final baseline subtracted interpolated sample. We can also use the auto function to complete these in one step. Notice that the plot.on default is false so we must set it to true if we want graphs to appear.

```
auto <- auto.baseline(x=Sample.1, exclusion.lwr = 60, exclusion.upr = 80,
                      grid.temp = seq(45,90,0.1),plot.on = TRUE,
                      explicit = FALSE)
```


Baseline Subtracted Sample
15
10
5
0
dCp
50
60
70
80
90
Temperature


Raw Curve with Spline Overlay
0
−5
−10
−15
dCp
50
60
70
80
90
Temperature



Running More Than One

Lastly, lets run all 20 thermograms through our function and graph the end product. This code might take a while to process.

```
final.results <- multiple.thermogram.subtraction(x = UrineWorking, exclusion.lwr = 60,
                                                 exclusion.upr = 80, grid.temp = seq(45, 90, 0.1),
                                                 plot.on = FALSE, point = "innermost",
                                                 explicit = FALSE, file.on = FALSE)
```

The code above provides a data frame with 20 baseline subtracted thermograms interpolated all to the same temperature grid. Below it is shown how to produce a graph of each thermogram output.

```
pdf('/Final.Results.pdf')
{
  for(j in 1:length(All.IDs))
  {
    g1 <- Final.Results %>% select(Temperature, All.IDs[j]) %>%
      ggplot(aes(x = Temperature, y = .[,2])) +
      geom_point() +
      labs(title = paste0('Final Automated Sample for ', str_sub(All.IDs[j], 2)))
    print(g1)
  }
```

```
}
dev.off()
```

Signal Detection

If we want to see if a thermogram is signal or noise, we can use the signal.detection function. In order to use this function, we must have the forecast package installed.

```
library(forecast,quietly = TRUE)
#> Warning: package 'forecast' was built under R version 4.1.3
#> Registered S3 method overwritten by 'quantmod':
#>   method            from
#>   as.zoo.data.frame zoo
result <- signal.detection(Sample.1)
```

```
head(result)
#>   result
#> 1 Signal
```