```r
# quotienten-pyrometry bei spektrum - includes.r
# Definition of functions
# 6.6.2020

library(dplyr)
library(reshape2)
library(ggplot2)
library(cobs)
library(magrittr)

#------------------------------------------------------

bb_calc <- function(lambda,T) {
  lambda = lambda*1e-9
  c = 299792458 # in m s
  h = 6.62607015e-34   # in J s
  k = 1.380649e-23  # in J s
  res = 2*h*c^2/lambda^5*1/(exp(h*c/(lambda*k*T))-1)
  data.frame(lambda=lambda*1e9, L=res)
}

#------------------------------------------------------

make_second_axis <- function(dat_y1,dat_y2) {
  scale.a      <- range(dat_y1)
  scale.b      <- range(dat_y2)
  scale.factor <- diff(scale.a)/diff(scale.b)
  trans        <- ~ ((. - scale.a[1]) / scale.factor) + scale.b[1]
  dat_y2       <- ((dat_y2 - scale.b[1]) * scale.factor) + scale.a[1]
  res = list(factor=trans,data=dat_y2)
  return(res)
}
#------------------------------------------------------

spec_data_make <- function(lambda_range, plot_time_point=c(1), T=2000, eps,
                           sensibility, noise=noise, plot=FALSE) {
  # eps_fac = approx(eps$lambda, eps$eps, lambda_range, rule = 2)
  eps_fac = eps(lambda_range)
  sens_fac = sensibility(lambda_range)
  data_bb = bb_calc(lambda_range,T)
  data = data.frame(lambda=lambda_range, L=data_bb$L * eps_fac * sens_fac)
  # data = data.frame(lambda=lambda_range, L=data_bb$L * eps_fac$y * sens_fac$y)
  # melt(data, varnames=c("lambda","T"), value.name="value")
  data$L = jitter(data$L, factor=0, amount=noise)
  data$L = jitter(data$L, factor=noise)
  if (plot) {
    q = ggplot() + ggtitle("Data generated") + scale_x_continuous(labels = scales::label_number(scale = 1)) + theme_bw()
    q = q + xlab(expression(lambda*" in nm")) + ylab(eval(bquote(expression("L("*lambda*",T="*.(T)*"K)") )))
    q = q+geom_line(data=data_bb, aes(x=lambda,y=L, group = T, colour = "Black body"))
    trans = make_second_axis(data_bb$L,data$L)
    data$L = trans$data
    q = q + scale_y_continuous(sec.axis = sec_axis(trans=trans$factor, name = "Measured signal"))
    q = q+geom_line(data=data, aes(x=lambda,y=L, group = T, colour = "Measured signal")) +
      guides(col = guide_legend(title = "Signal",label.position = "right"))
    print(q)
  }

  return(data)
}
```

```
# spec_data_get(plot=TRUE, eps=emiss_data, noise=noise)

#---------------------------------------------------------

smooth_curve <- function(data.in, lambda, fac=10000, plot=FALSE, title, title_x,
title_y){
  pars <- as.list(match.call()[-1])
  if (!hasArg(title)) title =pars$data.in
  if (!hasArg(title_x)) title_x = expression(lambda*" in nm")
  if (!hasArg(title_y)) title_y = expression(lambda*" in nm")
  colnames(data.in) = c("x", "y")
  flat = cobs(data.in[,1],data.in[,2],lambda=10000)
  data = as.data.frame(predict(flat,lambda))
  # colnames(data) = c("lambda", "L")
  if (plot) {
    q = ggplot() + ggtitle(title) + scale_x_continuous(labels =
scales::label_number(scale = 1)) + theme_bw()
    q = q + xlab(title_x) + ylab(title_y)
    q = q + guides(col = guide_legend(title = "Signal",label.position = "right"))
    q = q+geom_line(data=data.in, aes(x=x,y=y, group = T, colour = "with noice"))
    q = q+geom_line(data=data, aes(x=z, y=fit, group = T, colour = "smoothed"))
    print(q)
  }
  return(data)
}

# smooth_curve(spec_data, lambda = spec_data$lambda, plot=TRUE)

#---------------------------------------------------------

black_body_ref = function(T_range, lambda, plot=FALSE, T_plot=1000) {
  bb_matrix = lapply(T_range, bb_calc, lambda=lambda)

  bb_melt = melt(bb_matrix, id=c("lambda","L"), value.name="L")
  names(bb_melt)[3] <- "T"
  bb_melt$T = T_range[bb_melt$T]

  # Calc ratio
  bb_melt_group = group_by(bb_melt, T)
  # bb_ratioio_Calc = function(L) L$L[-length(L$lambda)]/L$L[-1]
  bb_ratio_Calc = function(L) {
    data.frame(lambda=L$lambda[-length(L$lambda)]+diff(L$lambda)/2, bb_ratio=L$L[
length(L$lambda)]/L$L[-1])
  }

  bb_ratio_group = group_modify(bb_melt_group, ~ {bb_ratio_Calc(.x) })

  if (plot) {
    # windows(height=4.5, width=7, xpos=10, ypos=100)
    L = filter(bb_melt_group, T==T_plot)
    q = ggplot() + ggtitle("Black body radiation") + scale_x_continuous(labels =
scales::label_number(scale = 1)) + theme_bw()
    q = q + xlab(expression(lambda*" in nm")) +
ylab(eval(bquote(expression("L("*lambda*",T="*.(T)*"K)") )))
    q = q + guides(col = guide_legend(title = "Signal",label.position = "right"))
    q = q+geom_line(data=L, aes(x=lambda,y=L, group = T, colour = "Radiation"))

    bb_ratio = filter(bb_ratio_group, T==T_plot)

    trans = make_second_axis(L$L,bb_ratio$bb_ratio)
    bb_ratio$bb_ratio = trans$data
    q = q + scale_y_continuous(sec.axis = sec_axis(trans=trans$factor, name =
"dL"))

```

```
    # windows(height=4.5, width=7, xpos=10, ypos=100)
    q = q+geom_line(data=bb_ratio, aes(x=lambda,y=bb_ratio, group = T, colour = "Ratio"))
    print(q)
  }
  bb = list(L=ungroup(bb_melt_group), ratio=ungroup(bb_ratio_group))
  return(bb)
}

#----------------------------------------------------
# Calc ratio
# lambda = lambda_range
# data = spec_cal
spec_data_ratio = function(data,T_range, lambda, plot=FALSE) {
  ratio_calc = function(L) {
    data.frame(lambda=L$lambda[-length(L$lambda)]+diff(L$lambda)/2, ratio=L$L[-length(L$lambda)]/L$L[-1])
  }

  ratio = group_modify(data, ~ {ratio_calc(.x) })
  ratio = mutate(ratio, ratio=case_when(ratio>1.1 ~1.1, ratio< -0.1 ~-0.1, TRUE~ratio))
  if (plot) {
    # windows(height=4.5, width=7, xpos=10, ypos=100)
    q = ggplot() + ggtitle("Ratio") + scale_x_continuous(labels = scales::label_number(scale = 1)) + theme_bw()
    q = q + xlab(expression(lambda*" in nm")) + ylab(eval(bquote(expression("L("*lambda*",T="*.(T)*"K)") )))
    q = q + guides(col = guide_legend(title = "Signal",label.position = "right"))
    q = q+geom_line(data=data, aes(x=lambda,y=L, group = T, colour = "normed signal"))

    ratio2 = ratio
    trans = make_second_axis(data$L,ratio$ratio)
    ratio2$ratio = trans$data
    q = q + scale_y_continuous(sec.axis = sec_axis(trans=trans$factor, name = "Ratio"))

    # windows(height=4.5, width=7, xpos=10, ypos=100)
    q = q+geom_line(data=ratio2, aes(x=lambda,y=ratio, colour = "Ratio"))
    print(q)
  }
  return(ratio)
}
```