Vignette - GPDA

GPDAAuthors

20/09/2021

#Load required packages

```
Package.Names <- c("Matrix", "Rcpp", "RcppArmadillo", "mvtnorm",
    "optimization", "matrixStats", "matrixStats", "ggplot2",
    "formatR")
lapply(Package.Names, library, character.only = TRUE)
sourceCpp("GPDAnonStatFinal.cpp")
source("GPDARversionFinal.R")
```

Toy example - Breast Cancer MS dataset

In this toy example, we will fit the GPDA model to the breast cancer MS dataset in the R package ProData. This dataset has been analyzed in section 5 of the reference paper. We begin with some pre-processing steps, which are similar to the steps described in Li et al. (2005).

Load and import data

```
# BiocManager::install('PROcess')
# BiocManager::install('ProData')
library(ProData)
library(PROcess)
f45c <- system.file("f45c", package = "ProData")
fs <- dir(f45c, full.names = TRUE)
data(f45cbmk)
SpecGrp <- pData(f45cbmk)
table(SpecGrp[, 1])
```

```
##
##  A  B  C  D
## 55 64 35 13
```

Processed markers

```
explevels = exprs(f45cbmk)
detpeaks = rownames(explevels)
```

Match spectra to sample data

```
gi <- regexpr("i+[0-9]+", fs)
specName <- substr(fs, gi, gi + attr(gi, "match.length") - 1)
mt <- match(SpecGrp[, 2], toupper(specName))
```

Prepare classification response

```
N = dim(SpecGrp)[1]
y = rep(0, N)  # HER2 positive
y[SpecGrp[, 1] == "B"] = 1  #healthy
y[SpecGrp[, 1] == "C"] = 2  #ER/PR positive
y[SpecGrp[, 1] == "D"] = 3  #single individual
```

Use the PROcess package for basic processing of the spectrum data. Process spectrum: baseline subtraction and renormalization

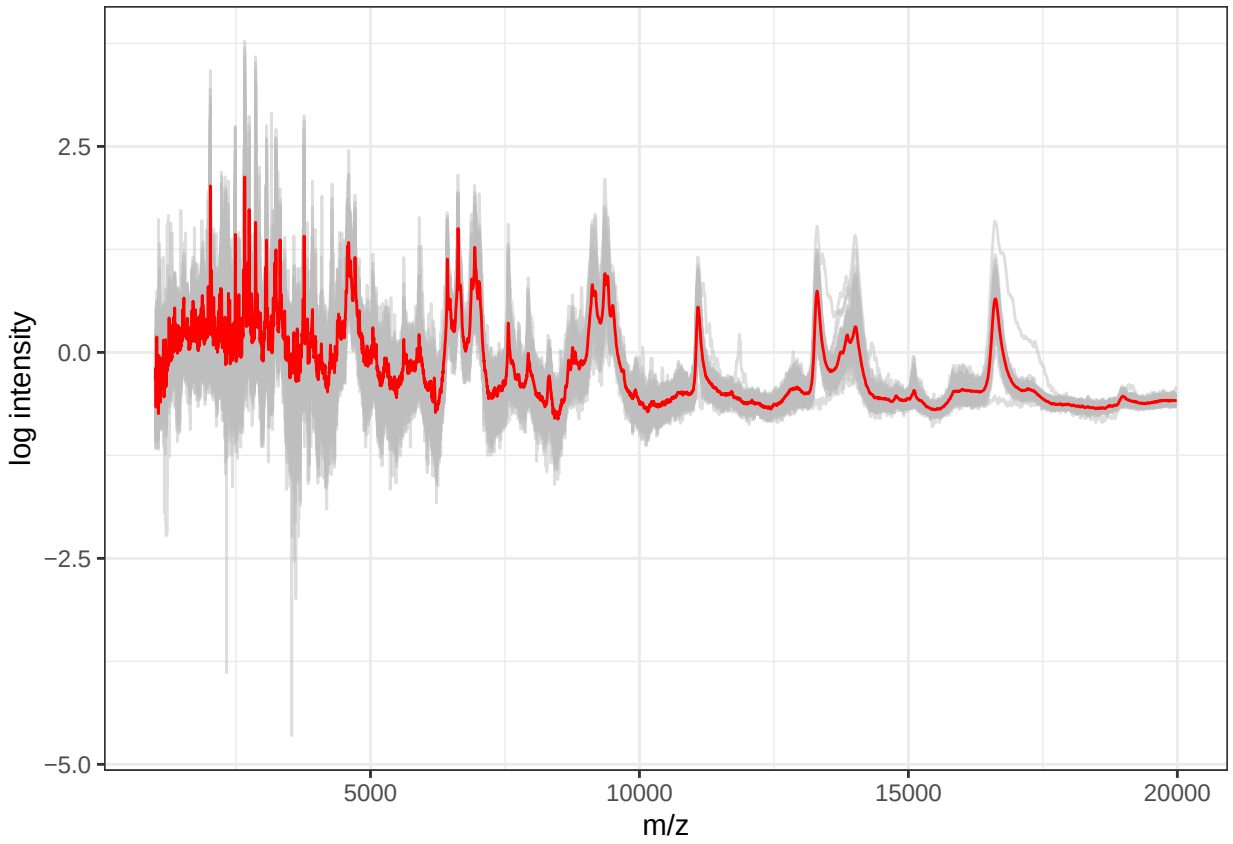
```
spec <- rmBaseline(f45c, method = "loess", bw = 0.1)
spec <- spec[, mt]  #match ordering to phenotype data
colnames(spec) <- SpecGrp[, 2]
prospec <- renorm(spec, cutoff = 1000)
prospec = prospec + 0.5
prospec = log(prospec)
```

Time points

```
t2 = as.numeric(rownames(prospec))
TP2 = length(t2)
```

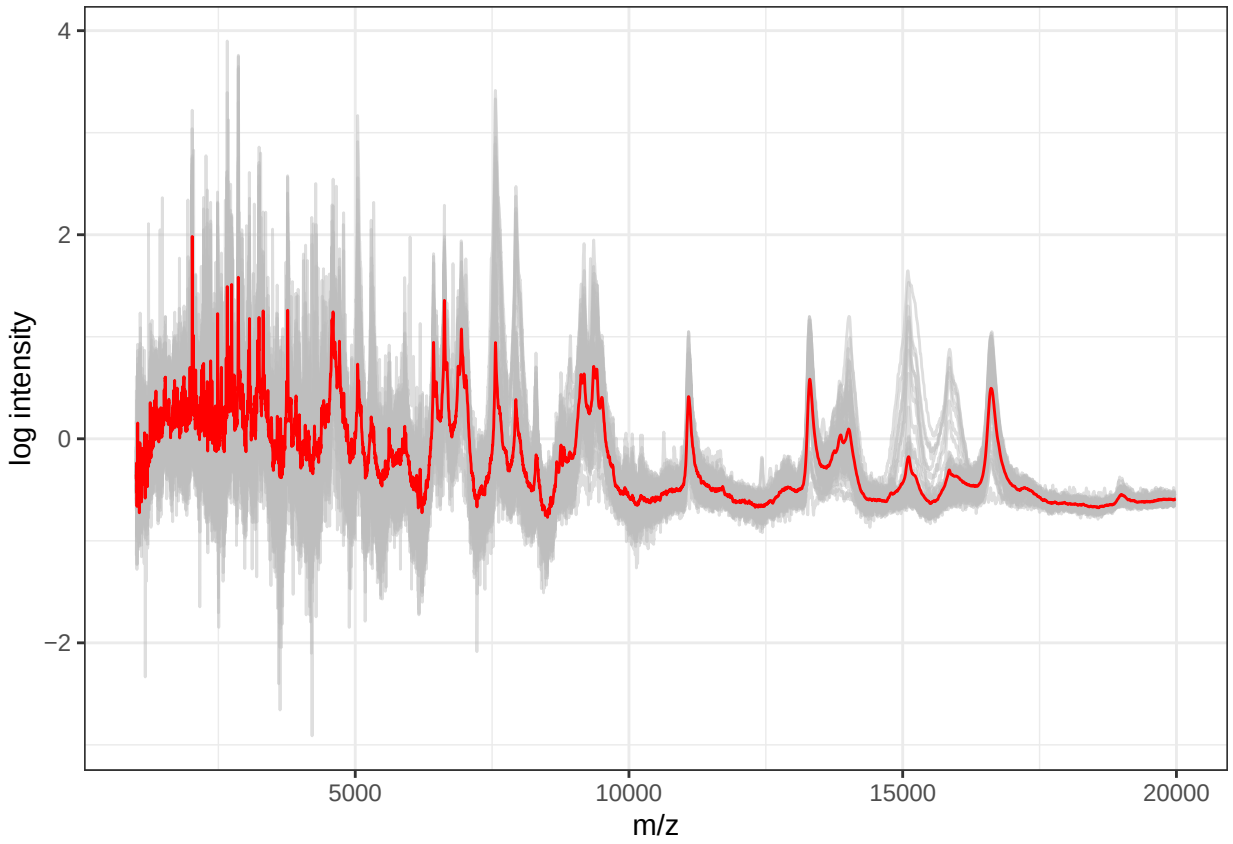
Plot processed spectrum data for healthy women

```
df = data.frame(x = t(matrix(prospec[, y == 1], 1, TP2 * sum(y ==
    1))), t = matrix(t2, TP2 * sum(y == 1), 1), ind = c(1:sum(y ==
    1)) %x% rep(1, TP2))
df2 = data.frame(x = apply(t(prospec[, y == 1]), 2, mean), t = t2)
ggplot(df2, mapping = aes(x = t, y = x)) + geom_line(data = df,
    mapping = aes(x = t, y = x, group = ind), colour = "GRAY",
    alpha = 1/2, size = 1/2) + geom_line(col = "red") + theme_bw() +
    xlab("m/z") + ylab("log intensity")
```



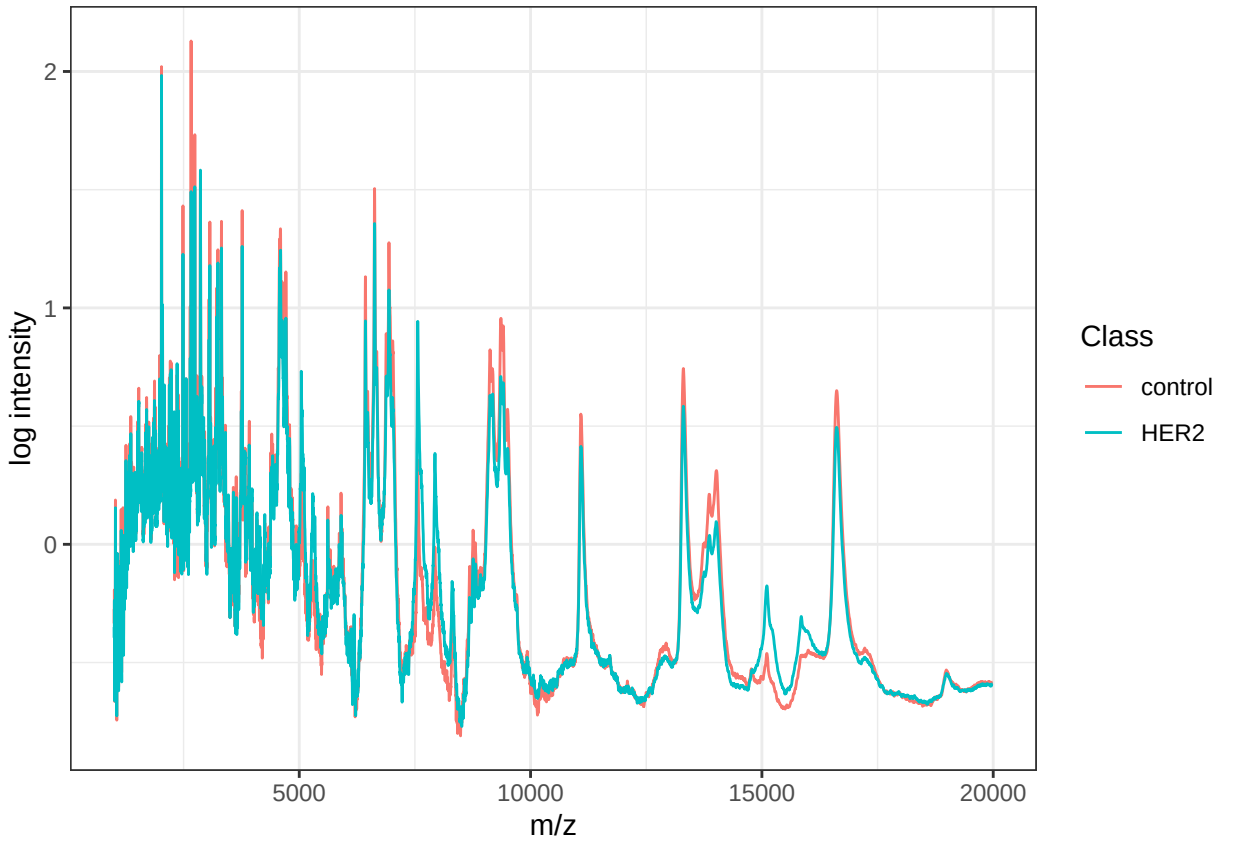
Plot processed spectrum data for HER2 positive

```
df = data.frame(x = t(matrix(prospec[, y == 0], 1, TP2 * sum(y ==
    0))), t = matrix(t2, TP2 * sum(y == 0), 1), ind = c(1:sum(y ==
    0)) %x% rep(1, TP2))
df2 = data.frame(x = apply(t(prospec[, y == 0]), 2, mean), t = t2)
ggplot(df2, mapping = aes(x = t, y = x)) + geom_line(data = df,
    mapping = aes(x = t, y = x, group = ind), colour = "GRAY",
    alpha = 1/2, size = 1/2) + geom_line(col = "red") + theme_bw() +
    xlab("m/z") + ylab("log intensity")
```



Plot mean spectrum across groups

```
df = data.frame(x = c(apply(t(prospec[, y == 0]), 2, mean), apply(t(prospec[,
    y == 1]), 2, mean)), t = rep(t2, 2), Class = c(rep("HER2",
    TP2), rep("control", TP2)))
ggplot(df, mapping = aes(x = t, y = x)) + geom_line(data = df,
    mapping = aes(x = t, y = x, colour = Class)) + theme_bw() +
    xlab("m/z") + ylab("log intensity")
```



Subset to HER2 positive and healthy women only, then scale data

```
vy = as.numeric(as.factor(y[y <= 1])) - 1
combineX = t(prospec[, y <= 1])
combineXscale <- 10 * (combineX - mean(combineX))
p = ncol(combineXscale)
n <- length(vy)
```

Fit GPDA model. Description of the input arguments: mXtrain is an $n_{\text{train}} \times p$ matrix of predictors for training dataset. vytrain is a vector of size n_{train} of class labels for training dataset. mXtest is an $n_{\text{test}} \times p$ matrix of predictors for testing dataset. train.cycles is the number of VB cycles for posterior inference phase of the algorithm. test.cycles is the number of VB cycles for classification phase of the algorithm. delta is the distance between adjacent locations.

```
GPobj <- GPDA.sparse.NonStat(mXtrain = combineXscale, vytrain = vy,
    mXtest = combineXscale, train.cycles = 5, test.cycles = 3,
    delta = 1)
```

Obtain predicted class labels

```
#c(GPobj$xi)
```

Obtain posterior probability of $\gamma(t) = 1$ for all locations

```
#c(GPobj$vw)
```

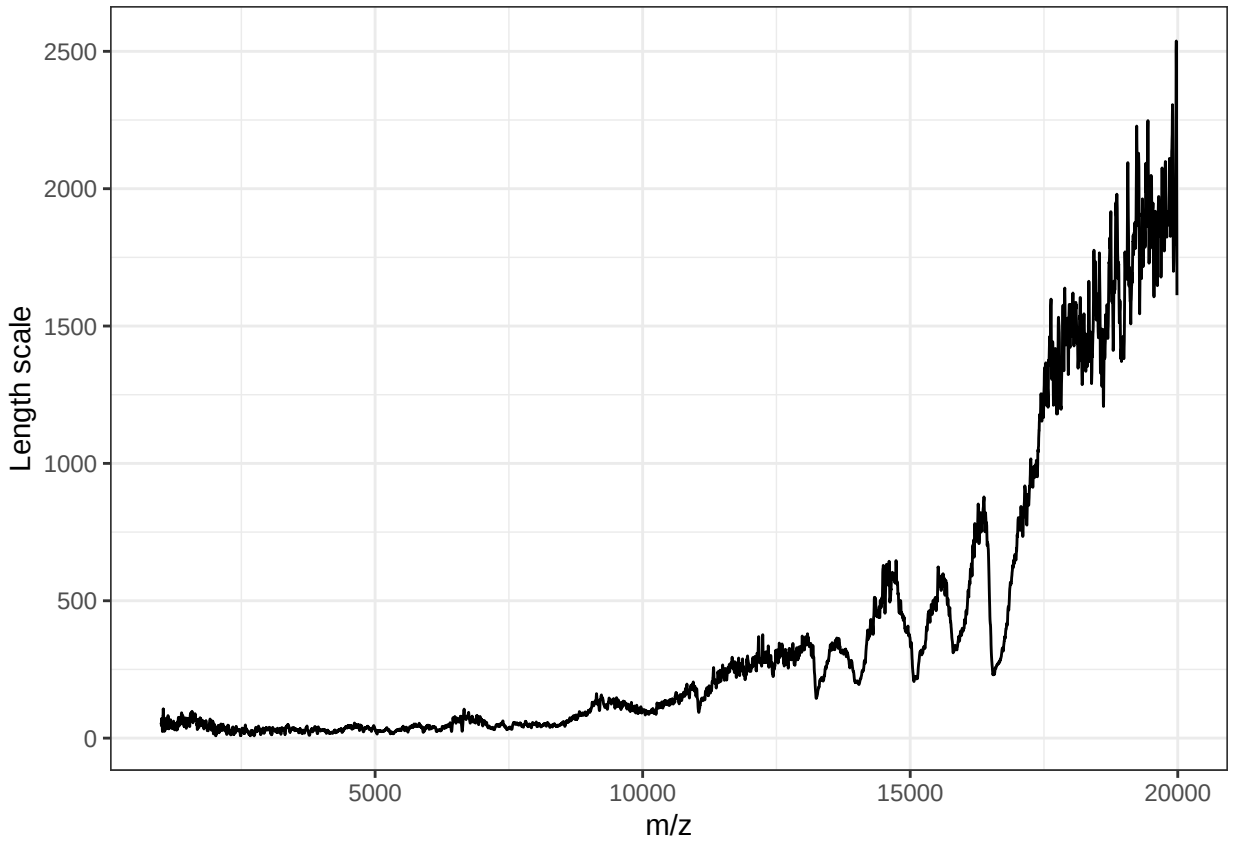
Plot posterior expectation of location-varying length scale of first observation.

```
# Observation-specific log length scale perturbation
GPobj$zeta
```

```
##   [1] 0.7228770 0.5848458 1.2179306 2.1893002 0.3434005 0.4623882 1.4548308
##   [8] 1.5861764 0.8760459 0.6170778 1.1030380 1.3799817 1.5783172 1.4355446
##  [15] 1.4840994 1.6595570 0.9322424 1.2800585 0.9803965 0.9882535 1.2691432
##  [22] 1.1947078 0.6720359 1.1188259 1.7503269 1.6480848 1.5440363 1.0857415
##  [29] 0.4900830 1.2190270 0.9622929 0.6706544 0.8372693 0.6412076 1.6463093
##  [36] 0.8068172 0.6176920 0.9457404 1.6929764 1.1993982 1.8297548 1.1744953
##  [43] 1.1230353 0.5944033 1.3876124 1.1851432 1.1766911 1.4216919 1.2075779
##  [50] 0.3192181 1.0805168 1.1429334 0.6698436 1.2144910 1.0017156 1.1582511
##  [57] 0.5050482 0.6488858 0.7400207 0.7677700 1.1178487 0.8705785 1.0575802
##  [64] 1.5258681 1.2032703 0.6402142 0.6872689 0.7645776 1.0837231 0.6114080
##  [71] 0.7363013 0.8025172 1.5277248 1.3718630 0.7763975 0.6774687 0.8647822
##  [78] 1.2442883 0.9772427 1.3360842 1.0147288 1.0832297 0.8278467 0.6958499
##  [85] 1.2512067 1.4548829 0.7867006 1.2905619 1.1924077 1.6167486 1.3421301
##  [92] 1.7495683 1.0874095 0.5979930 0.9528994 0.9041351 1.0187376 0.9596194
##  [99] 1.7172691 0.8160188 1.8518861 0.9897972 1.3624032 1.7030401 0.9129508
## [106] 0.6272179 0.9878677 1.3948417 1.0391496 1.0316473 1.1486568 1.9016787
## [113] 1.4365226 1.3803630 1.0845601 1.1195871 1.3930008 1.1737743 0.9323104
```

```r
# Common component of log length scale perturbation
# c(GPobj$mS) Posterior expectation of length-scale for
# first observation
LengthScaleObs1 <- exp(c(GPobj$mS) + GPobj$zeta[1])

df = data.frame(x = LengthScaleObs1, t = t2[1:(TP2 - 1)])
ggplot(df, mapping = aes(x = t, y = x)) + geom_line(data = df,
    mapping = aes(x = t, y = x)) + theme_bw() + xlab("m/z") +
    ylab("Length scale")
```



Plot posterior expectation of $z_1(t)$. Note that $q(z_1) = \text{LogN}(m_{z_1}, \Sigma_{z_1})$.

```
# Diagonal entries of \Sigma_{z_1} GPobj$Sigma_Z[,1,1]
# m_{z_1} GPobj$mZ[1,]
z <- GPobj$mZ[1, ] + 0.5 * GPobj$Sigma_Z[, 1, 1]
df = data.frame(x = z, t = t2)
ggplot(df, mapping = aes(x = t, y = x)) + geom_line(data = df,
    mapping = aes(x = t, y = x)) + theme_bw() + xlab("m/z") +
    ylab("E_q[Z(t)]")
```