`msiCompare`: Class comparison for mass spectrometry imaging data

April J. Harry

November 2, 2017

Contents

1 Introduction

The `R` package `msiCompare` was built to provide access to the methods for class comparison in MSI described in this dissertation. `msiCompare` leverages the tools and data structures in the `Cardinal` package. When used in combination with `Cardinal`, it is possible to build a complete workflow from data preprocessing to statistical analysis all within `R`. In this chapter, we explore the main functionality of `msiCompare`.

The package is currently managed through a Github repository. The `devtools` package makes installation straightforward:

```
library(devtools)
install_github("ajharry/msiCompare")
```

Once installed, attach both the `msiCompare` and `Cardinal` packages.

```
library(msiCompare)
library(Cardinal)
```

2 Simulating MS images

We can simulate MSI data approximately according to Model **??**. For the n=1 situation, begin by defining two regions of interest which will represent the conditions to be compared. In the code below, condition 2 is defined as a square region in the center of a 30×30 grid, while condition 1 is the bordering area.

```
conditions <- ifelse((expand.grid(x=1:30, y=1:30)$x %in%
                        (1+floor(30/5)):(30-floor(30/5)) &
                        expand.grid(x=1:30, y=1:30)$y %in%
                        (1+floor(30/5)):(30-floor(30/5))), 2, 1)
```

The following function simulates the MS image. The algorithm works as follows:

1. Given a value for the spatial variance `tau2`, a `size1` $\times$ `size1` square grid of samples are drawn from the $properCAR(\rho, \texttt{tau2}, W)$ distribution, with $\rho = 0.9999$ (that is, nearly the $ICAR$ model). The neighborhood matrix W is assumed to be binary with the 8-neighborhood structure. The samples are centered within each condition by default.

2. $(\texttt{size1})^2$ independent samples are drawn from $\mathcal{N}(\text{mean} = 0, \text{var} = \texttt{sig2})$ to represent measurement error, and then added to the spatially correlated samples.

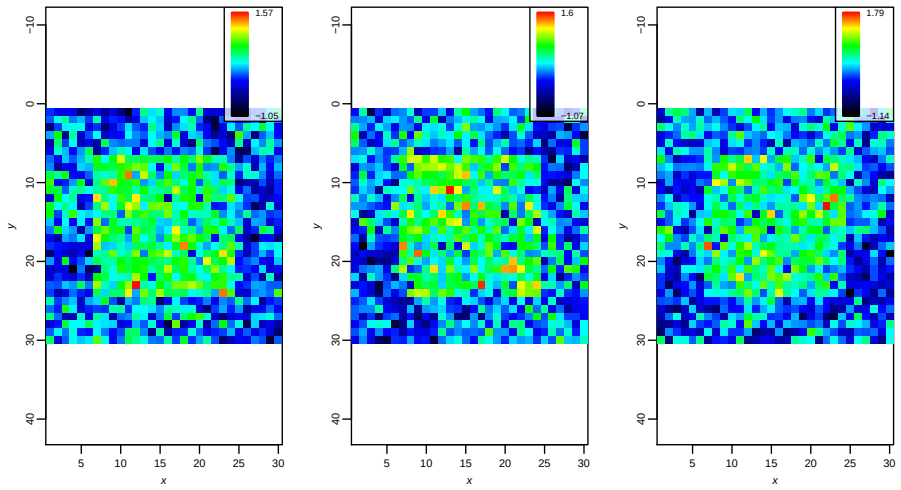
3. A value of `diff` is added to the simulated values for locations in condition 2 and zero to values for locations in condition 1.

4. This process is repeated `reps` times, with the sampled images returned as a `Cardinal MSImageSet` object.

```
 s <- simSingle(
      reps = 3,
      diff = log2(1.5),
      tau2 = 0.1,
      sig2 = 0.1,
      seed = 8372,
      size1 = 30,
      pattern = conditions
      )
summary(s)
##        Length Class      Mode
## simSet 1      MSImageSet S4
## reps   1      -none-     numeric
## diff   1      -none-     numeric
## tau2   1      -none-     numeric
## sig2   1      -none-     numeric
## seed   1      -none-     numeric
## size1  1      -none-     numeric
## size2  1      -none-     numeric
```

Using `Cardinal` plotting tools, we can view the 3 simulated images.

```
image(s$simSet, feature = 1, layout = c(3,1))
image(s$simSet, feature = 2)
image(s$simSet, feature = 3)
```



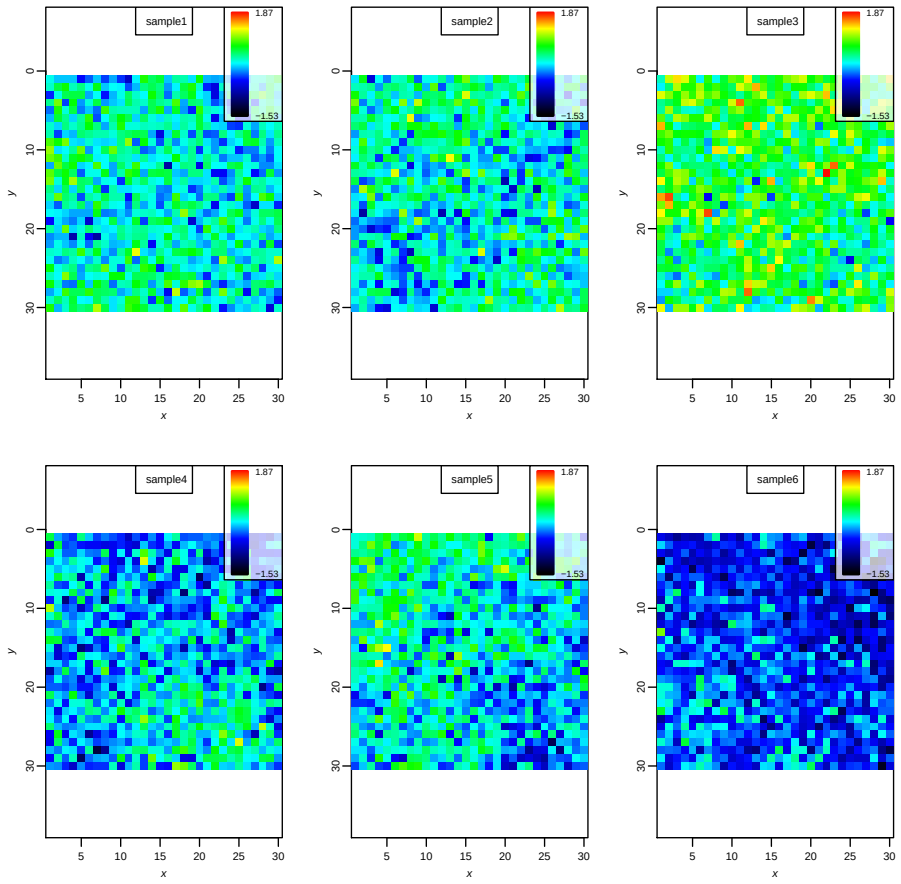
The specified conditions for each location are included in the `pixelData` of the `MSImageSet`.

```
head(pixelData(s$simSet))
## An object of class 'IAnnotatedDataFrame'
##   pixelNames: x = 1, y = 1 x = 2, y = 1 ... x = 6, y = 1 (6 total)
##   varLabels: x y sample diagnosis
##   varMetadata: labelType labelDescription
```

MSI datasets with $n > 1$ are simulated similarly, with `diff` added to the simulated tissues for one of the conditions. Additionally, a value representing tissue-to-tissue biological variation drawn from $\mathcal{N}(\text{mean} = 0, \text{var} = \texttt{sampleVar})$ for each tissue and added to its simulated values.

```
s_multi <- simMulti(
     sampleVar = 0.1,
     reps = 1,
     diff = log2(1.5),
     tau2 = 0.1,
     sig2 = 0.1,
     seed = 8372,
     size1 = 30,
     size2 = 30^2,
     numHealthy = 3,
     numDisease = 3)

image(s_multi$simSet, feature = 1, layout = c(3,2))
```



In addition to the condition at each pixel, the sample names are included in the `pixelData` of the `MSImageSet`.

```
head(pixelData(s_multi$simSet))
## An object of class 'IAnnotatedDataFrame'
##   pixelNames: x = 1, y = 1, sample = sample1 x = 2, y = 1, sample
##     = sample1 ... x = 6, y = 1, sample = sample1 (6 total)
##   varLabels: x y sample diagnosis
##   varMetadata: labelType labelDescription
```

3 Statistical methods for class comparison

3.1 Hiearchical Bayesian Spatial Models

The models described by Model **??** and the extensions in Equations and are all available for use in the package. The models are fit using the Gibbs sampler MCMC algorithm detailed earlier in this chapter, with the hierarchical centring version of Section **??** used for datasets with $n > 1$.

The appropriate version of the model is chosen automatically based on the supplied arguments. The `conditionOfInterest` argument should be a vector

representing the condition of each pixel, while the `techRep` vector must identify which tissue each pixel belongs to. The `bioRep` vector is optional; if the experiment has a subsampling or paired design then `bioRep` should identify which biological individual each pixel belongs to. These experimeal design arguments will often be columns in the `pixelData` of the `MSImageSet` object.

```
fitHBSM <- compareMSI(msset = s_multi$simSet,
           conditionOfInterest = s_multi$simSet$diagnosis,
           techRep = s_multi$simSet$sample,
           bioRep = NULL,
           feature = 1,
           nsim = 5000, burnin=2500,
           trace = T, dropZeros = F)
```

From the model fit we can access point estimates of the parameters of interest, such as the posterior probability of differential abundance.

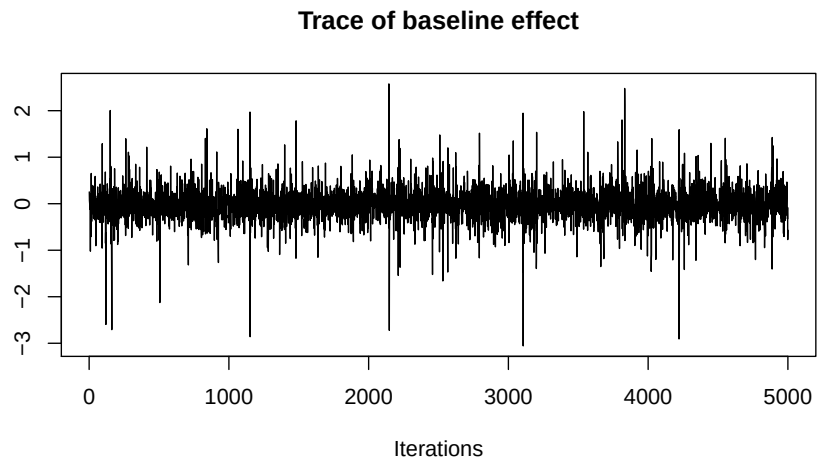
```
fitHBSM$Feature1$gamma
## [1] 0.03278689
```

We have also implemented the BFDR thresholding procedure (Chapter **??** and Ventrucci2011) to adjust for multiple comparisons.

```
postProbs <- runif(100) #simulating 100 posterior probabilities
bfdr <- BFDRdecision(postProbs, alpha = 0.05)
table(bfdr)
## bfdr
## FALSE  TRUE
##    91     9
```

If the `trace` argument in `compareMSI` is set to `TRUE`, then the trace of the MCMC samples is returned.

```
plot(fitHBSM$Feature1$trace[,1],
     density = F, main = "Trace of baseline effect")
```



3.2 Spatial modeling for MSI, as described in Cassese, et al

The spatial model proposed in Cassese2016 is also available in the package. The code is based on the implementation in the supplementary information for Cassese2016, with a wrapper to streamline its use with `Cardinal` objects.

```
fit_spautolm <- cass(msset = s$simSet,
                     roiFactor = factor(s$simSet$diagnosis),
                     logscale = F, thresholds = 1:5)
```

```
fit_spautolm$results$CAR_AIC_min_pvalues
## [1] 0 0 0
```

3.3 ANOVA

ANOVA methods (Models **??** and **??**) are available natively in `R`, but we show them here for completeness:

```
##### Location-wise ANOVA #####
intensities <- c(spectra(s_multi$simSet)[1,])
conditions <- s_multi$simSet$diagnosis

# summary of model fit
coef(summary(lm(intensities~conditions)))
##                      Estimate  Std. Error   t value      Pr(>|t|)
## (Intercept)       -0.03924918 0.008125954 -4.830102  1.402342e-06
## conditionsHealthy  0.44145207 0.011491835 38.414412 1.242852e-285
```

```
##### Tissue-wise ANOVA #####
sampNames <- sampleNames(s_multi$simSet)
samps <- s_multi$simSet$sample

# get tissue averages
averages <- sapply(sampNames,
                   function(s) mean(intensities[samps== s]))

#condition for each tissue
conditions <- factor(c("Healthy","Healthy","Healthy",
                       "Disease","Disease","Disease"))

#summary of model fit
coef(summary(lm(averages ~ conditions)))
##                      Estimate Std. Error   t value  Pr(>|t|)
## (Intercept)       -0.03924918  0.1469225 -0.267142 0.8025676
## conditionsHealthy  0.44145207  0.2077799  2.124614 0.1008170
```