Using the `PamGeneMixed` Package (version 1.03)

Pushpike Thilakarathne
I-BioStat,
University of Hesselt
Belgium

April 3, 2013

Contents

1 Introduction

The `PamGeneMixed` package is a suite for convenient analysis of PamChip microarray experiments. The package is built based on the paper Thilakarathne et al. (2011). The package currently consists of several functions which can be used to visualize and fitting the models.

This document provides an overview of the analysis workflow for the paper Thilakarathne et al. (2011) and usability of the functions in the package.

2 Preparation of the Data

First we load the package `PamGeneMixed` and the example real-life data set `PamChipData`. This particular dataset consists of an oncology experiment where phosphorylation of peptides is measured in 20 different cell line lysates in the absence (control) and presence of a kinase inhibitor (compound) using peptide arrays with a kinetic read-out. Ten out of twenty cell lines are known to be responsive and others are non-responsive to the specific kinase inhibitor under study. The cell lines were randomly assigned to six 96-well plates, of which each well contains a 144-peptide array. Each cell line was typically replicated 5-6 times. (for more details see Versele et al. 2009)

```
R> library(PamGeneMixed)
R> #packageDescription("PamGeneMixed")
```

Alternatively, simulated data sets can also be very valuable. However, we did not use simulated data.

2.1 Structure of the Data

```
R> data(PamChipData)
R> PamChipData[c(1:5,500:505),1:10]
```

```
     ID ResState ArrayNum  CellName TreatName Time Pep1 Pep2 Pep3 Pep4
30    1       NR  0559C80 MDA231PAR   Control   11 2237 2104  287  759
31    1       NR  0559C80 MDA231PAR   Control   16 3746 3696  391  822
32    1       NR  0559C80 MDA231PAR   Control   21 3979 4051  460  920
33    1       NR  0559C80 MDA231PAR   Control   26 4361 4392  394  912
34    1       NR  0559C80 MDA231PAR   Control   31 4218 4452  393  971
1071  7        R  0559C80       N87   Control   36 4679 4855  832 1592
1072  7        R  0559C80       N87   Control   41 4726 4777  920 1740
1073  7        R  0559C80       N87   Control   46 5072 5072  952 1681
1074  7        R  0559C80       N87   Control   51 5147 5344 1039 1795
1075  7        R  0559C80       N87   Control   56 5091 5204  999 1645
1076  7        R  0559C80       N87   Control   61 5060 5172 1131 1737
```

2.2 Unique Time Points

```
R> UTime<-unique(PamChipData[,c("Time")])
R> UTime
```

```
 [1] 11 16 21 26 31 36 41 46 51 56 61 66 71
```

2.3 Responsive and Non-Responsive Cell Lines

```
R> RnRcellines<-unique(PamChipData[,c("CellName","ResState")])
R> rownames(RnRcellines)<-1:20
R> RnRcellines
```

```
    CellName ResState
1  MDA231PAR       NR
2     SUM159        R
3      H3255        R
```

```
4         N87        R
5        H322        R
6      SHSY5Y       NR
7      SNU484        R
8      HEK239       NR
9      SUM149        R
10     U118MG       NR
11      SKBR3        R
12       A431        R
13      SJSA1       NR
14      DU145        R
15    BT474OD        R
16 MDAMB435s       NR
17       SNU5       NR
18      MKN45       NR
19       MCF7       NR
20      GTL16       NR

R>
```

2.4 Number of Plates (96 well plates)

```
R> Plates<-unique(PamChipData[,c("ArrayNum")])
R> Plates

[1] 0559C80 0559C82 0559C86 0559C87 0681C2  0681C44
Levels: 0559C80 0559C82 0559C86 0559C87 0681C2 0681C44
```

2.5 Peptides Names

```
R> PeptidesNames<-setdiff(colnames(PamChipData),
         c("ID", "ResState", "ArrayNum", "CellName", "TreatName","Time"))
R> PeptidesNames[1:10]

 [1] "Pep1"  "Pep2"  "Pep3"  "Pep4"  "Pep5"  "Pep6"  "Pep7"  "Pep8"  "Pep9"
[10] "Pep10"
```

3 Exploration of Observed Kinase Activity Profiles

This section describes the exploration of observed PamChipData for a specific peptide. Since analysis is done by peptide by peptide, it is worthwhile to explore the behavior of the kinase activity profiles at different experimental factors. Therefore, activity profiles are visualized at cell line level, replicate level and group levels.

3.1 Observed Replicate Specific-Profiles for a certain cell line of a given peptide

```
R> ObservedRepProfiles(pepname="Pep1",DataMat=PamChipData,
     cells="SKBR3",Trt.Group=c("Treatment","Control"),Res="R",log.true=FALSE)

R>     ObservedRepProfiles(pepname="Pep5",DataMat=PamChipData,
     cells="DU145",Trt.Group=c("Treatment","Control"),Res="R",log.true=FALSE)

R>     ObservedRepProfiles(pepname="Pep130",DataMat=PamChipData,
     cells="SUM159",Trt.Group="Treatment",Res="R",log.true=TRUE)
```

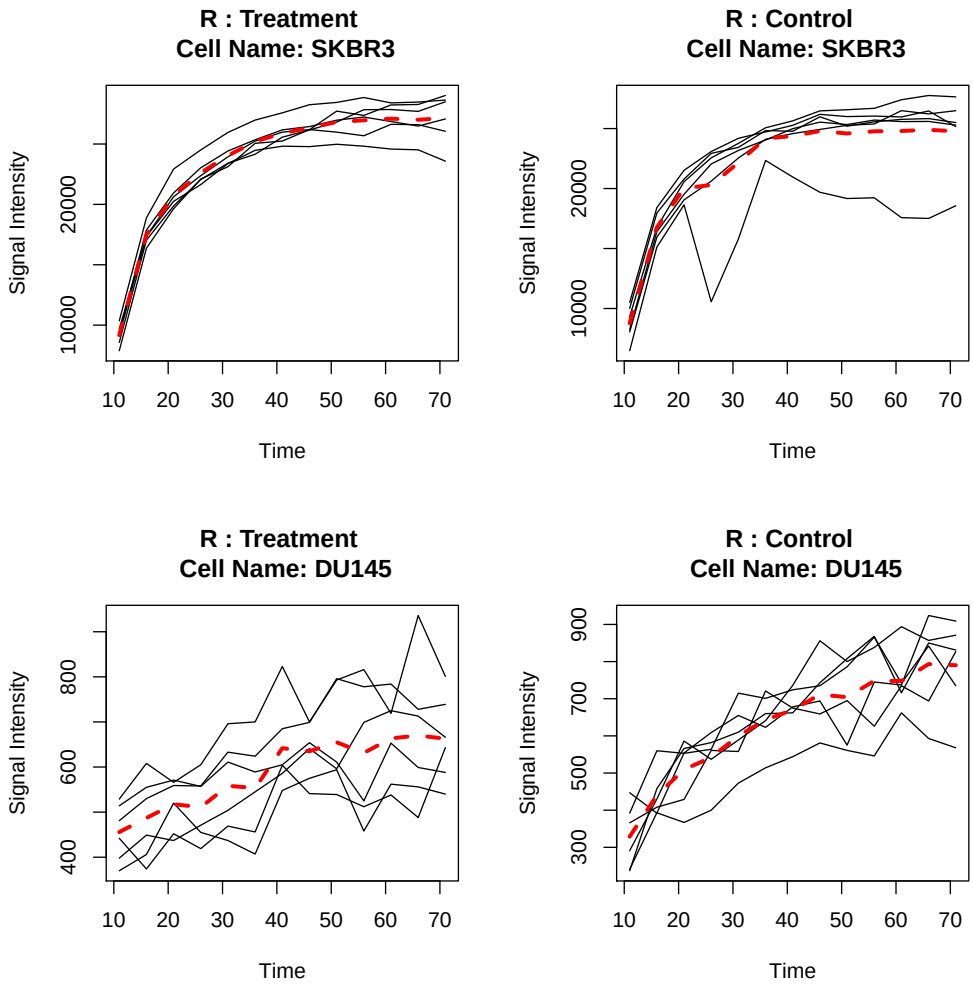

Figure 1: Replicate specific profiles for a cell line and for a peptide

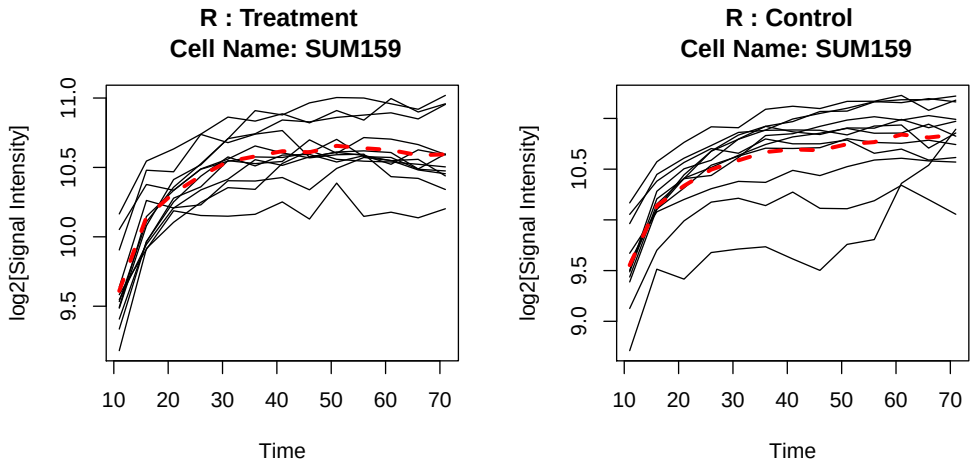

Figure 2: $\log_2$ transformed replicate specific profiles for a cell line and for a peptide

3.2 Observed Cell Line Specific-Profiles for a certain cell line of a given peptide

```
R> ObservedCellProfiles(pepname="Pep130",DataMat=PamChipData,
 Trt.Group=c("Treatment","Control"),Res=c("R","NR"),log.true=TRUE)
```

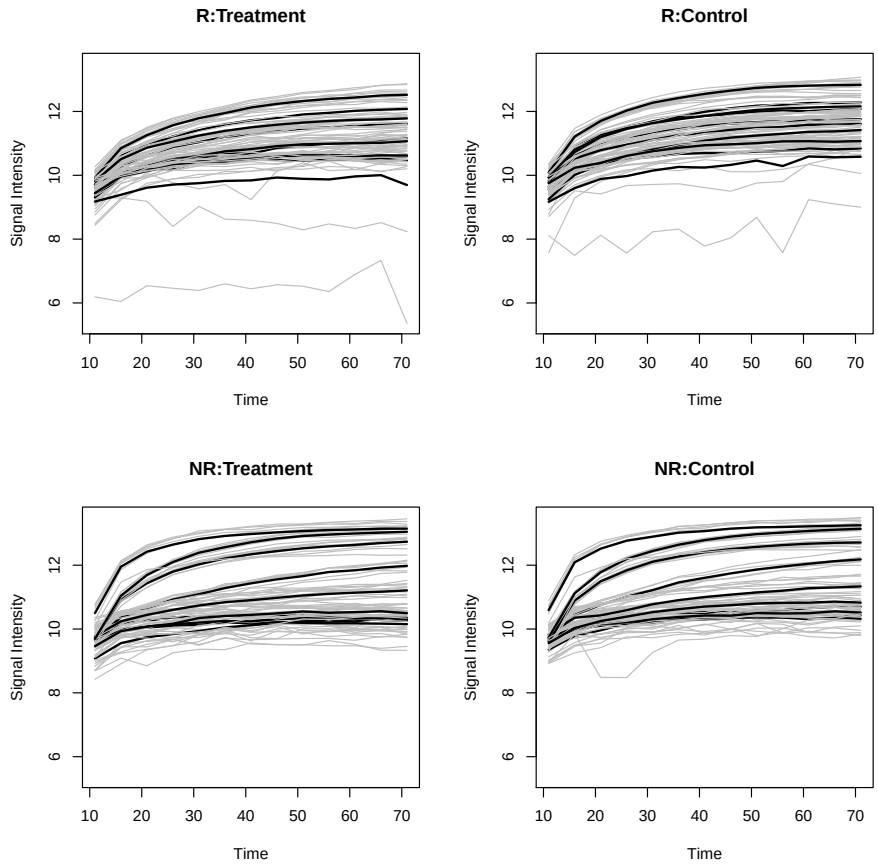

Figure 3: Observed Cell Line Specific-Profiles for a peptide and $\log_2$ transformed signals.

3.3 Observed Group Specific-Profiles for a certain cell line of a given peptide

```
R> ObservedGroupProfiles(pepname="Pep58",DataMat=PamChipData,
 Trt.Group=c("Treatment","Control"),Res=c("R","NR"),log.true=TRUE)
```

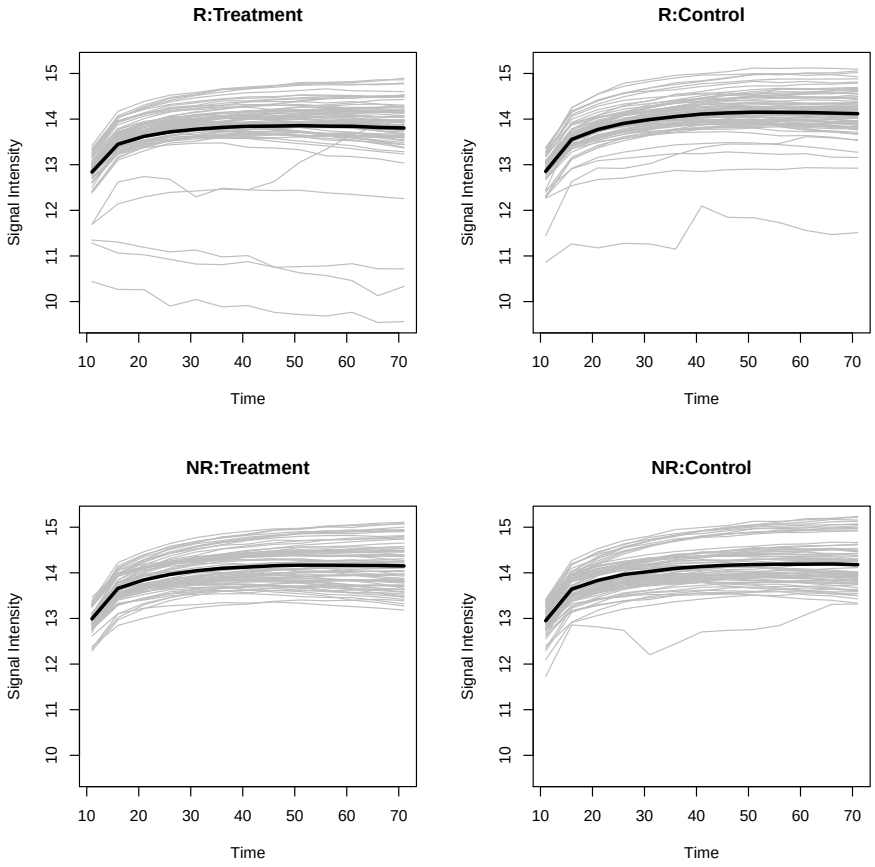

Figure 4: Observed Group Specific-Profiles for a peptide and $\log_2$ transformed signals. Gray colour lines indicate observed replicate specific profiles.

4 Cleaning Phase

Prior to fitting the model, we examine the data for unusual observations and extreme profiles. Outlier removal has been done in three steps. First, we notice that some of the observations are negative for a given kinase activity profile. The PamChip software reports background corrected intensities. Hence, negative observations might occur when the measured intensity for a particular peptide in a certain well does not exceed the background signal. Negative intensity measurements are replaced by half of the positive minimum intensity measurement of that profile. Second, we compute the area under the curve (AUC) for each profile of a given cell line. Analysis of variance has been carried out based on those AUC values while taking the treatment as covariate with two levels. Extreme profiles whose residuals are beyond $\pm 2\times$ the square root of the mean squared error are removed. Third, we look at the observations within a profile and remove the extreme observations if they differ more than $\pm 2\times$ standard deviations of the profile-specific mean. In this step, we are able to remove extremely high intensity measurements (spikes). All these steps are repeated for each peptide and each cell line.

The following function creates a S4 class objects as output results and can be used to remove the outline profiles and extreme observations.

```
R> PreProcessPam(p   =1, cel =11,   d   =2, PamS=PamChipData, plotting=T )

R> PreProcessPam(p   =6, cel =4,   d   =2, PamS=PamChipData, plotting=T )
```

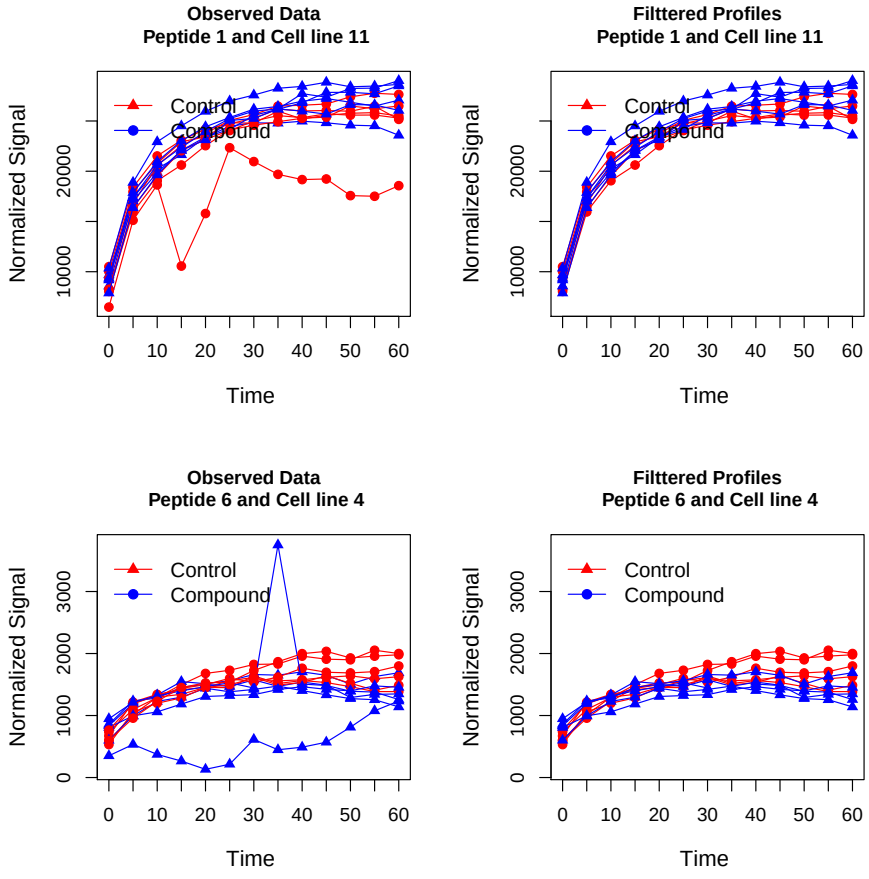

Figure 5: Cell line specific data cleaning phase.

Following R code chunk can be used to continuously visualize the pre-processing.

```
R> for (i in 1:148){
     for (cc in 1:20){
```

```
        par(ask=T)
        PreProcessPam(p   =i, cel =cc,   d   =2, PamS=PamChipData, plotting=T )
    }
 }
```

This process can be automated using the following function. In this particular example, five peptides are preprocessed sequentially.

```
R> PeptidesNames<-setdiff(colnames(PamChipData),
 c("ID", "ResState", "ArrayNum", "CellName", "TreatName","Time"))
R> PreProcessAllPeptides(pep.names=PeptidesNames[1:5],PamSig=PamChipData)

-------- PreProcessed List of Peptides --------
Class         : CleanedPamData
No. Peptides  : 5
No. Cell lines: 20
Columns names for a certain peptide:
 ID ResState ArrayNum CellName TreatName Time Pep1
Unique Time points: 0 5 10 15 20 25 30 35 40 45 50 55 60
PreProcessed Data frames are saved as RData Objects in
: C:/Users/lucp2520/AppData/Local/Temp/RtmpOAdxv9/Rbuild1478306f4125/PamGeneMixed/inst/doc
-----------------------------------------
```

5 Modeling Kinase Activity Profiles

5.1 Modeling Responsive Cell Line Data only

In this section, we use only Responsive cell line data such that we have only two groups: treatment and control for the analysis.

5.1.1 Model 1:

In this particular model, the cell line-specific random intercept is considered to capture correlation of the intensity measurement over time within the cell line. We assumed cell line-specific random slopes for linear as well as for quadratic time effects to capture different evolution of kinase activity over time. Moreover, we allow these cell lines-specific random structures to be different for each group. We assumed group-specific random structure for cell lines.

```
R> data(TestPepModelData)
R> head(TestPepModelData)
```

```
  ID ResState ArrayNum CellName TreatName Time Peptide
1  6        R  0559C80    H3255   Control   11    2706
2  6        R  0559C80    H3255   Control   16    4442
3  6        R  0559C80    H3255   Control   21    5045
4  6        R  0559C80    H3255   Control   26    5491
5  6        R  0559C80    H3255   Control   31    5950
6  6        R  0559C80    H3255   Control   36    6124
```

```
R> PTx<-TestPepModelData
R> # log2 transform the response
R> PTx$y<-log2(PTx[,c("Peptide")])
R> PTx<-PTx[PTx[,c("ResState")]=="R",]
R> n.groups<-length(unique(PTx[,c("ResState")]))*length(unique(PTx[,c("TreatName")]))
R> #n.groups
R>
R> xhlp<-lm(y~-1+TreatName,data=PTx,x=TRUE)$x
R> for (i in 1:n.groups) PTx$ResTrt[xhlp[,i]==1]<-i
R> PTx$ResTrt<-as.factor(PTx$ResTrt)
R> levels(PTx$ResTrt)<-colnames(xhlp)
R> #unique cell lines
R> cellLines<-levels(PTx$CellName)
R> PTx$ResState<-as.factor(PTx$ResState)
R> # create interaction between Treatment and cell lines
R> PTx$CellLineResTrt<-0
R> xhlp<-lm(y~-1+CellName:TreatName,data=PTx,x=TRUE)$x
R> ncols<-ncol(xhlp)
R> for (i in 1:ncols) PTx$CellLineResTrt[xhlp[,i]==1]<-i
R> PTx$CellLineResTrt<-as.factor(PTx$CellLineResTrt)
R> levels(PTx$CellLineResTrt)<-colnames(xhlp)
R> M1gamm<-PamGeneMix(formula=y~-1+ResTrt+s(time,by=ResTrt,bs="tp",m=3),
          PTx=PTx,Random.structure=list(ArrayNum=~1,
          CellLineResTrt=~1+time+time2,ID=~1+time+time2),
          Control.list=list(maxIter=200, msMaxIter=250 ,msMaxEval=1000,apVar=TRUE))
```

We can now visualize the results using function VisualizePamGeneMix.

```
R> VisualizePamGeneMix(M1gamm,plot.type="smooth.fit")
R> VisualizePamGeneMix(M1gamm,plot.type="cellline")
R> VisualizePamGeneMix(M1gamm, plot.type="SubSpeVelocity")
R> VisualizePamGeneMix(M1gamm,plot.type="replicate",name.cell= "A431" )
R> VisualizePamGeneMix(M1gamm,plot.type="velocity")
R> VisualizePamGeneMix(M1gamm,plot.type="velocityCI")
```

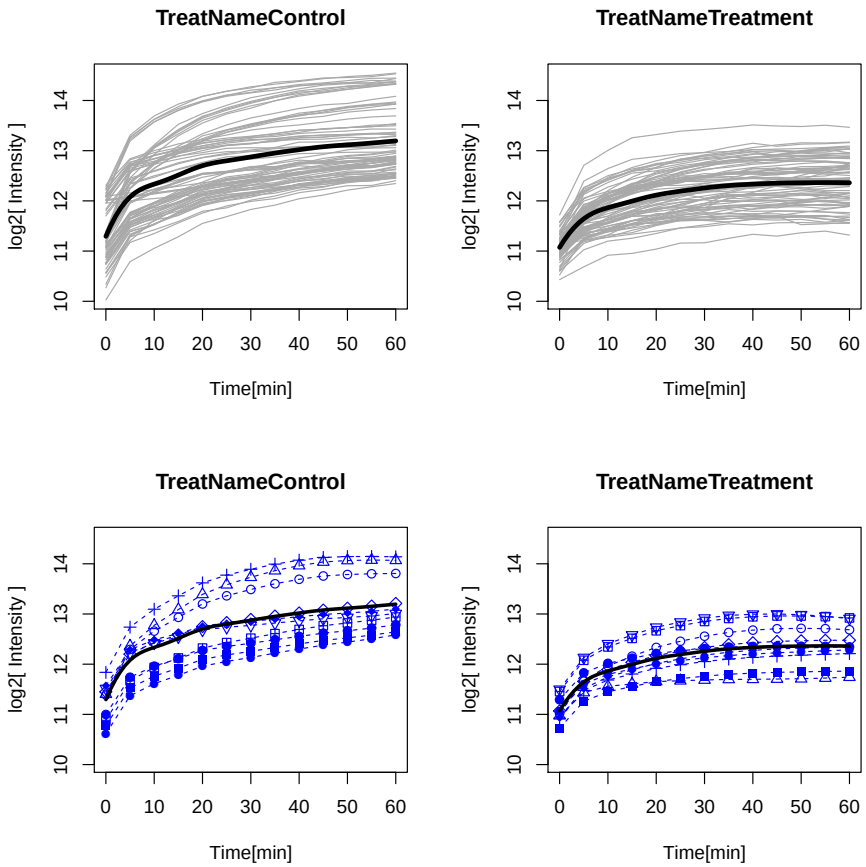

Figure 6: First row: Fitted smoothing curves. Second row: Fitted cell line specific profiles.

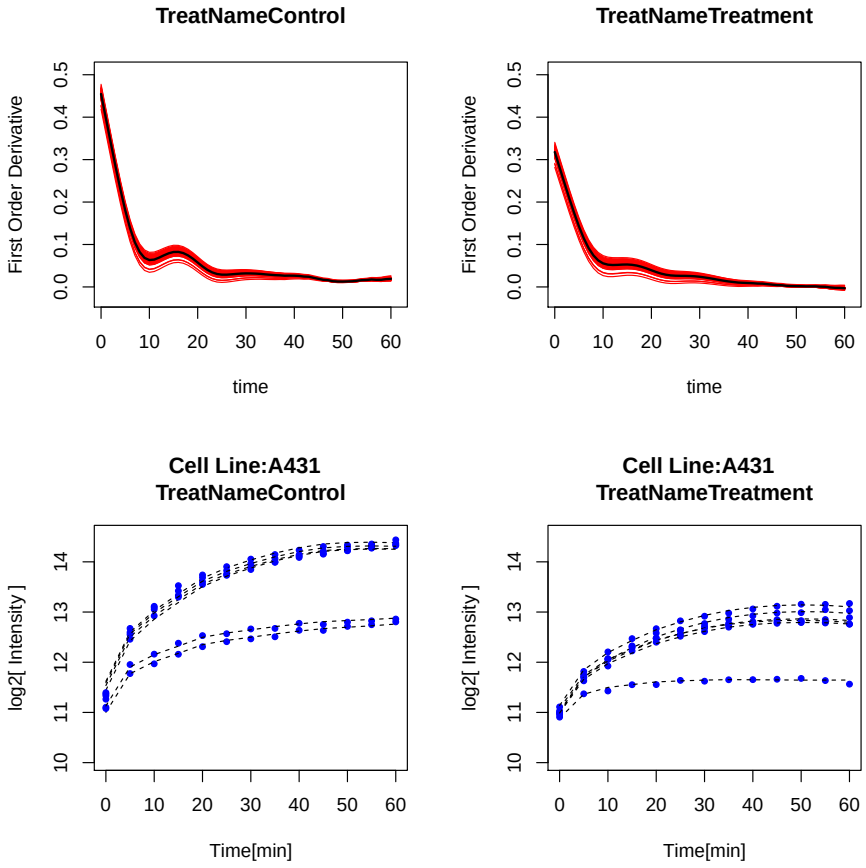

Figure 7: First row: Fitted subject specific velocity curves. Second row: Fitted replicate specific smoothing curves.

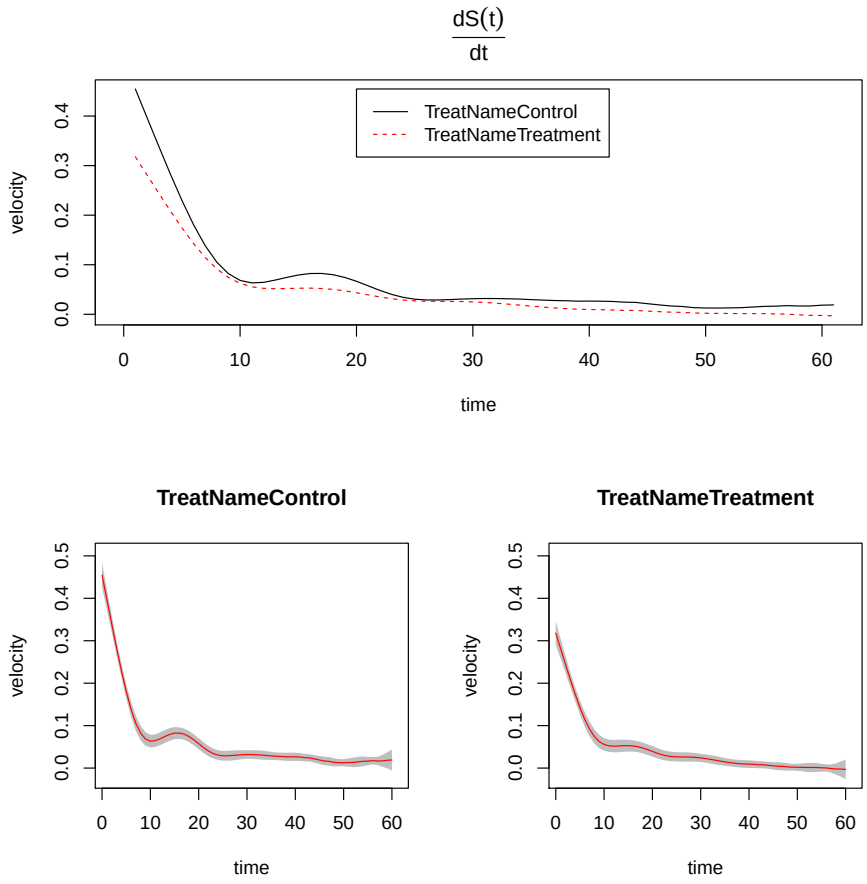

Figure 8: First row: Fitted group specific velocity curves. Second row: Fitted velocity curves with 95% CI.

Summarizing the model fit.

```
R> summary(M1gamm)
```

5.1.2 Model 2:

In this particular model is more or less the same as M1. However, correlation structure for the random effects are now common for treatment and control groups.

```
R> M2gamm<-PamGeneMix(formula=y~-1+ResTrt+s(time,by=ResTrt,bs="tp",m=3),
        PTx=PTx,Random.structure=list(ArrayNum=~1,
        CellName=~1+time+time2,ID=~1+time+time2),
        Control.list=list(maxIter=200, msMaxIter=250 ,msMaxEval=1000,apVar=TRUE))
```

5.2 Modeling four groups: Treatment and Responsive statuses

In this section we demonstrate how to use entire dataset (four groups) to fit the model while incorporating different correlation structures.

```
R> #-----------:  ------------------
R> data(TestPepModelData)
R> PTx<-TestPepModelData
R> # log2 transform the response
R> PTx$y<-log2(PTx[,c("Peptide")])
R> # we now create interaction between ResState and Treatment as follows.
R>
R> n.groups<-length(unique(PTx[,c("ResState")]))*length(unique(PTx[,c("TreatName")]))
R> PTx$ResTrt<-0
R> xhlp<-lm(y~-1+ResState:TreatName,data=PTx,x=TRUE)$x
R> for (i in 1:n.groups) PTx$ResTrt[xhlp[,i]==1]<-i
R> PTx$ResTrt<-as.factor(PTx$ResTrt)
R> levels(PTx$ResTrt)<-colnames(xhlp)
R> #unique cell lines
R> cellLines<-levels(PTx$CellName)
R> PTx$ResState<-as.factor(PTx$ResState)
R> # create interaction between Treatment and cell lines
R> PTx$CellLineResTrt<-0
R> xhlp<-lm(y~-1+CellName:TreatName,data=PTx,x=TRUE)$x
R> ncols<-ncol(xhlp)
R> for (i in 1:ncols) PTx$CellLineResTrt[xhlp[,i]==1]<-i
R> PTx$CellLineResTrt<-as.factor(PTx$CellLineResTrt)
R> levels(PTx$CellLineResTrt)<-colnames(xhlp)
```

5.2.1 Model 3

The more complex model for which we assumed that group-specific smoothing parameter and group-specific variance covariance structure. we used thin plate regression splines with third order derivative. In this model nested random effects structure is assumed for replicates within the cell lines. And for the cell lines - specific random effects are assumed to be realized from each group separately.

```
R> system.time(
 M3gamm<-PamGeneMix(formula=y~-1+ResTrt+s(time,by=ResTrt,bs="tp",m=3),
        Weights=varIdent(form=~1|ResTrt) ,
        PTx=PTx,Random.structure=list(ArrayNum=~1,
        CellLineResTrt=~1+time+time2,ID=~1+time+time2),
        Control.list=list(maxIter=200, msMaxIter=250 ,msMaxEval=1000,apVar=TRUE))
 )
R> M3gamm
```

We can now visualize the results using function VisualizePamGeneMix.

```
R> VisualizePamGeneMix(M3gamm,plot.type="smooth.fit")
```

```
R> VisualizePamGeneMix(M3gamm,plot.type="velocityCI")
```

we can now use testVarCom function to test the hypothesis that H_0: $Var(group_{R,i}) - Var(group_{R,j}) = 0$ and $Var(group_{NR,i}) - Var(group_{NR,j}) = 0$. That is testing whether there is a difference between treatment and control conditional on the responsive statues.

```
R> testVarCom(M3gamm)
```

6 Automated way of fitting model to each peptide for treatment and controls groups

6.1 Group-specific velocities are compared at Initial time point, End time point and for entire profile

```
R>     data(PamChipData)
R>     PeptidesNames<-setdiff(colnames(PamChipData),
     c("ID", "ResState", "ArrayNum", "CellName", "TreatName","Time"))
R>     #------- only two groups in the data -------------------
R>     temp1<-PreProcessAllPeptides(pep.names=PeptidesNames[1:20],
     PamSig=PamChipData[PamChipData[,c("ResState")]=="R",],PathOutPut="C:/Temp")
R>     temp1
R>     #---------------------------------
R>     Results1<-AutoPamGeneMix(formula=y~-1+ResTrt+s(time,by=ResTrt,bs="tp",m=3),
             Weights=varIdent(form=~1),
             Random.structure=list(ArrayNum=~1,
             CellLineResTrt=~1+time+time2,ID=~1+time+time2),
             temp1,PathOutPut="C:/FittedGamm")
R>
```

6.2 Group-specific velocities are compared additionally at half way of the time

```
R> Results2<-AutoPamGeneMix(formula=y~-1+ResTrt+s(time,by=ResTrt,bs="tp",m=3),
           Weights=varIdent(form=~1),
           Random.structure=list(ArrayNum=~1,
           CellLineResTrt=~1+time+time2,ID=~1+time+time2),
           temp1,TestAt=30,PathOutPut="C:/FittedGamm")
```

7 Automated way of fitting model to each peptide for four groups

```
R> #------- Four groups in the data ---
R> #-Preprocessing step
R>
R>     temp2<-PreProcessAllPeptides(pep.names=PeptidesNames,
     PamSig=PamChipData,PathOutPut="C:/Temp")
R>     temp2
R> #  Fit the model for each peptide
R> #  NOTE: This will take considerable time depending on your system speed.
R> ResultsAll<-AutoPamGeneMix(formula=y~-1+ResTrt+s(time,by=ResTrt,bs="tp",m=3),
              Weights=varIdent(form=~1|ResTrt),
              Random.structure=list(ArrayNum=~1,
             CellLineResTrt=~1+time+time2,ID=~1+time+time2),
              temp2,TestAt=30,PathOutPut="C:/FittedGamm")
R>
R> #save(ResultsAll,file="D:\\Projects\\PamGene\\RpamChip\\Rscripts\\ResultsAll.RData")
```

7.1 Visualize AutoPamGeneMix results

```
R> show(ResultsAll)

-------- Results for set of Peptides --------
Class        : PamChipAllPepResults
No. Peptides : 148
Total Run time (in seconds) : 20870.18
Number of Groups : 4
Fitted Gamm Objects are saved as RData Objects in :
 C:/FittedGamm
-----------------------------------------

R> Results<-VolcanoPam(ResultsAll,Topp=5,plotting=TRUE)
```