单细胞组学培训之时序分析——Monocle2 tutorial

zhongyu1@genomics.cn

本文档适用于 monocle2 版本的使用介绍，测试数据来源于 2017 年北大徐成冉课题组发表在《*Hepatology*》杂志上小鼠胚胎 E10.5-E17.5 双潜能的成肝细胞分化为肝实质细胞和肝内胆管细胞的表达数据 GSE90047。

一、 软件包安装及加载

如还未安装，请参考安装指南进行安装。

```
install.packages("BiocManager")
BiocManager::install("monocle")
```

安装完成后，加载 R 包:

```
library(monocle)
library(RColorBrewer)
```

二、 数据获取、导入及分析

1. 数据获取

所需数据在培训资料中的 5.3monocle_test_data 目录中，请自行至网盘链接中下载。

2. 数据导入及数据预处理

在进行时序分析前需要导入所需分析的数据，并根据表达量定量的分布选取合适的标准化方法进行标准化，对数据进行指控。

读取测试数据集，包括表达矩阵 `expr_matrix`，细胞相应注释信息 `sample_sheet`，基因注释信息 `gene_annotation`。 Monocle 适用于相对表达量值(例如 FPKM 或 TPM)或绝对转录本数目 UMI 的表达量矩阵。本例中使用的测试数据为 TPM 定量的表达矩阵。

```
## 读取测试数据集
expr_matrix <- read.table("scRNA-seq_TPM_GSE90047.xls",header=T,row.names=1, sep = "\t")
sample_sheet <- read.table("XCR_paper_cell_anno_sub.txt",header=T,row.names=1, sep = "\t")
gene_annotation <- read.table("Mus_ref.txt",header=T,row.names=1, sep = "\t")
```

```
## 数据预处理，并创建细胞注释及基因注释对象
pos <- which(rownames(gene_annotation) %in% rownames(expr_matrix))
gene_annotation <- gene_annotation[pos,]
expr_matrix <- expr_matrix[rownames(gene_annotation),rownames(sample_sheet)]
pd <- new("AnnotatedDataFrame", data = sample_sheet)
fd <- new("AnnotatedDataFrame", data = gene_annotation)
```

根据表达量数据的类型选择合适的分布（默认为 UMI 输入，适用于负二项分布；FPKM/TPM 适用于对数正态分布）。`lowerDetectionLimit` 为基因表达检出的最低阈值，默认 0.1。

```
cd <- newCellDataSet(as.matrix(expr_matrix), phenoData = pd, featureData = fd, lowerD
etectionLimit = 0.1, expressionFamily = tobit(Lower = 0.1))
```

RPC (mRNA per cell) 值通常比 FPKM 或 TPM 值更容易分析和利用工具进行建模，建议将 FPKM/TPM 数据转化为 RNA 数目后新建 **CellDataSet** 对象。

```
rpc_matrix <- relative2abs(cd) # estimate RNA counts
cd <- newCellDataSet(as(as.matrix(rpc_matrix),"sparseMatrix"), phenoData = pd, featur
eData = fd, expressionFamily=negbinomial.size(), lowerDetectionLimit = 0.5)
```

计算数据的 size factors 和 dispersions。size factors 有助于消除细胞间 mRNA 捕获的差异；dispersions 用于后续的差异表达分析。

```
## Estimate size factors and dispersions
cd <- estimateSizeFactors(cd)
cd <- estimateDispersions(cd)
```

过滤低于 1%细胞中检出的基因，最低表达阈值为 0.5。detectGenes 函数计算每个细胞里面表达的基因数量。

```
## Genes expressed in at least 1% cells with expressoin > 0.5
cd <- detectGenes(cd, min_expr = 0.5)
expressed_genes <- row.names(subset(fData(cd), num_cells_expressed > nrow(sample_shee
t) * 0.01))
length(expressed_genes)

## [1] 13026
```

3. 单细胞发育路径的构建

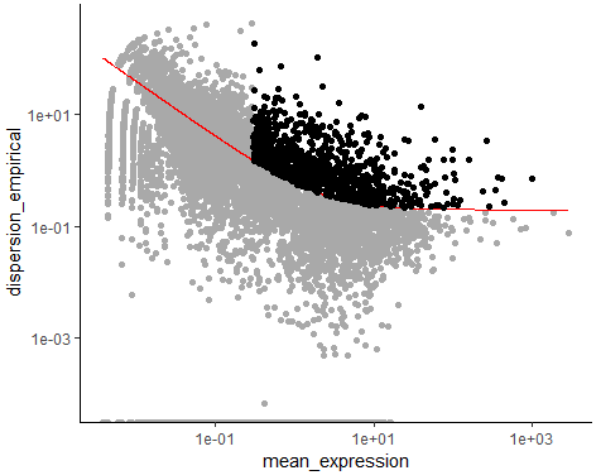
发育过程中细胞处于动态变化的过程，细胞在不同状态表达的基因表达谱有所差异，monocle 根据每个细胞基因的表达谱的相似和连续变化对细胞构建发育轨迹。第一个步骤是挑选合适的基因，包括3种方法，分别是：

A. 挑选细胞间高度离散的基因；
B. 挑选在 clusters/stages 间差异表达的基因；
C. 根据已知标记基因对细胞进行排序.

下面主要介绍第一种和第二种：

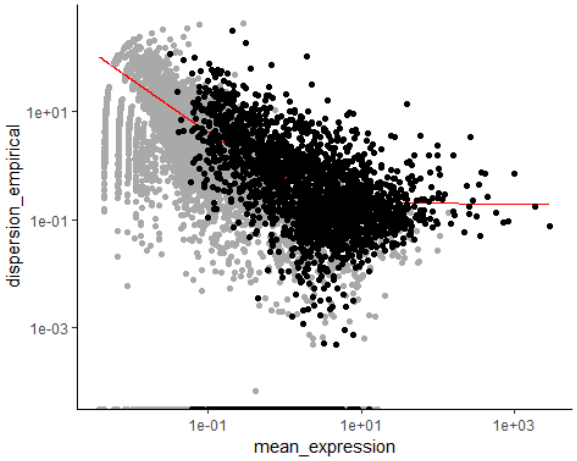
方法 A: 挑选细胞间高度离散的基因。

```
## Choose genes that define a cell's progress
## Select genes with high dispersion across cells
disp_table <- dispersionTable(cd)
ordering_genes <- as.character(subset(disp_table,mean_expression >= 0.3 & dispersion_
empirical >= 1 * dispersion_fit)$gene_id)
cd <- setOrderingFilter(cd, ordering_genes)
plot_ordering_genes(cd)
```



方法 B: 挑选在 clusters/stages 间差异表达的基因。本例中根据`"~Stage"`的差异进行差异分析，选取`qval < 1e-5`的基因并进行可视化。该步骤耗时较长，取决于用于差异分析细胞数目及基因数目。

```
## Select genes that differ between clusters/stages
diff_test_res <- differentialGeneTest(cd[expressed_genes,],fullModelFormulaStr = "~Stage")
ordering_genes <- row.names (subset(diff_test_res, qval < 1e-5))
ordering_genes <- intersect(ordering_genes, expressed_genes)
cd <- setOrderingFilter(cd, ordering_genes)
plot_ordering_genes(cd)
```

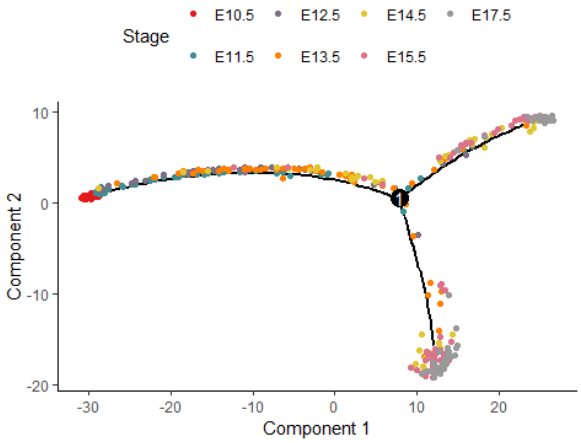


后续分析基于方法 B 的基因集合。选取的基因数目为每个细胞的维度，基于`'DDRTree'`方法进行数据降维，将数据降维至 2 维后对细胞进行排序。由于排序无法区分起点和重点，若分析所得时序于实际相反，根据 “reverse“参数进行调整，默认`reverse=F`。

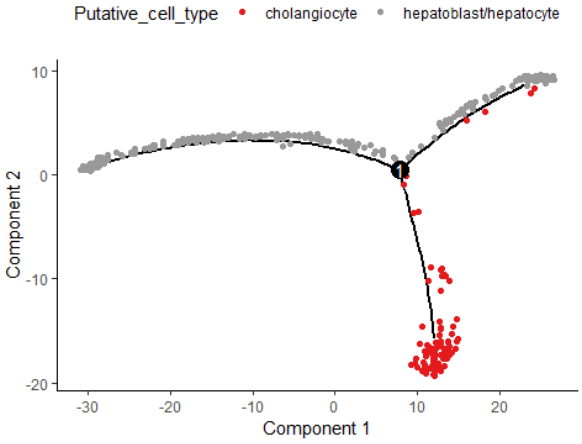
```
## Order cells by progress
cd <- reduceDimension(cd,max_components = 2, method = 'DDRTree')
cd <- orderCells(cd,reverse=F) ## The reverse flag tells Monocle to reverse the orientation of the entire process as it is being discovered from the data
```

排序好的细胞可以进行可视化，可标注细胞的各注释信息`"Stage"`, `"Putative_cell_type"`, `"Sort_by"`, `"State"`, `"Pseudotime"`等)

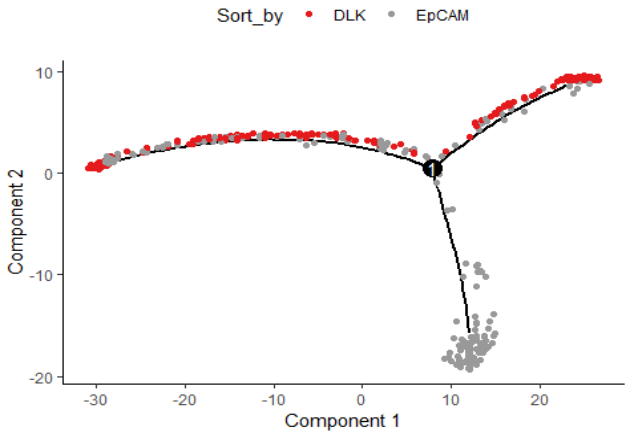
```
## generate color palette
getPalette <- colorRampPalette(brewer.pal(9, "Set1"))

## Visualize the trajectory in the reduced dimensional space.
plot(plot_cell_trajectory(cd, show_cell_names = F, color_by = "Stage")+scale_color_ma
nual(values = getPalette(length(unique(sample_sheet[,"Stage"])))))
```



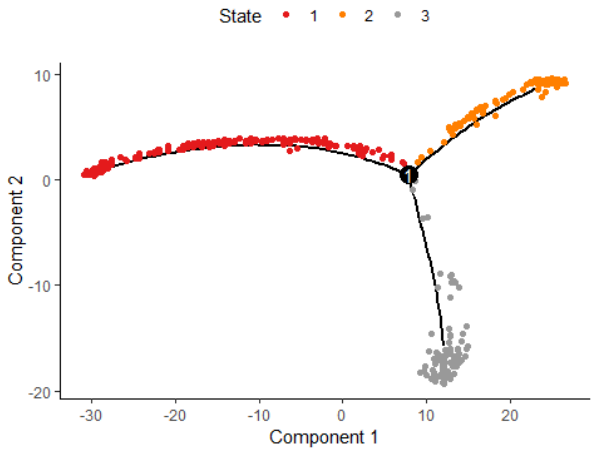

```
plot(plot_cell_trajectory(cd, show_cell_names = F, color_by = "Putative_cell_type")+s
cale_color_manual(values = getPalette(length(unique(cd@phenoData@data[,"Putative_cell
_type"])))))
```



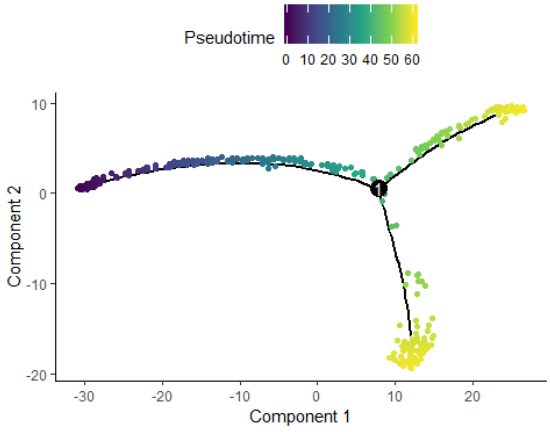

```
plot(plot_cell_trajectory(cd, show_cell_names = F, color_by = "Sort_by")+scale_color_
manual(values = getPalette(length(unique(cd@phenoData@data[,"Sort_by"])))))
```




```
plot(plot_cell_trajectory(cd, show_cell_names = F, color_by = "State")+scale_color_manual(values = getPalette(length(unique(cd@phenoData@data[,"State"])))))
```

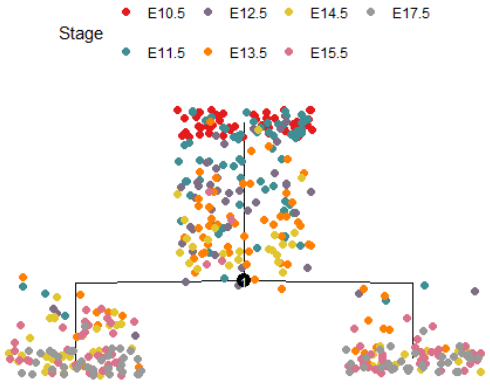



```
plot(plot_cell_trajectory(cd, show_cell_names = F, color_by = "Pseudotime")+scale_color_viridis_c())
```

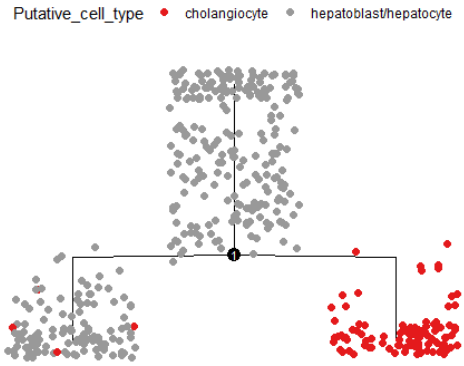


树状分支的可视化。

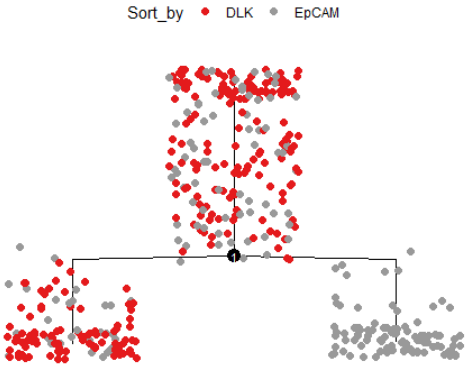
```
## Plot complicate tree structure
plot_complex_cell_trajectory(cd, color_by = 'Stage', show_branch_points = T, cell_size = 2, cell_link_size = 0.5) + scale_color_manual(values = getPalette(length(unique(sample_sheet[,"Stage"]))))
```



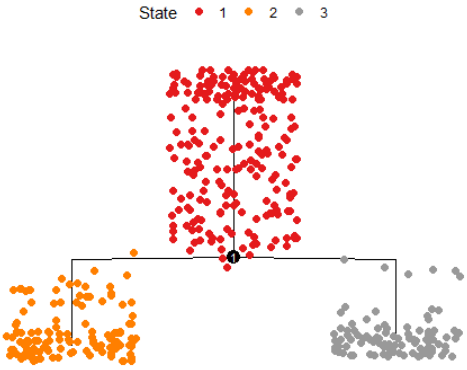

```
plot_complex_cell_trajectory(cd, color_by = 'Putative_cell_type', show_branch_points = T, cell_size = 2, cell_link_size = 0.5) + scale_color_manual(values = getPalette(length(unique(cd@phenoData@data[,"Putative_cell_type"]))))
```



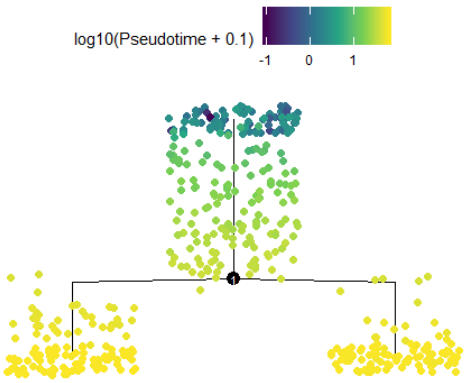

```
plot_complex_cell_trajectory(cd, color_by = 'Sort_by', show_branch_points = T, cell_size = 2, cell_link_size = 0.5) + scale_color_manual(values = getPalette(length(unique(cd@phenoData@data[,"Sort_by"]))))
```




```
plot_complex_cell_trajectory(cd, color_by = 'State', show_branch_points = T, cell_siz
e = 2, cell_link_size = 0.5) + scale_color_manual(values = getPalette(length(unique(c
d@phenoData@data[,"State"]))))
```




```
plot_complex_cell_trajectory(cd, color_by = 'Pseudotime', show_branch_points = T, cel
l_size = 2, cell_link_size = 0.5) + scale_color_viridis_c()
```

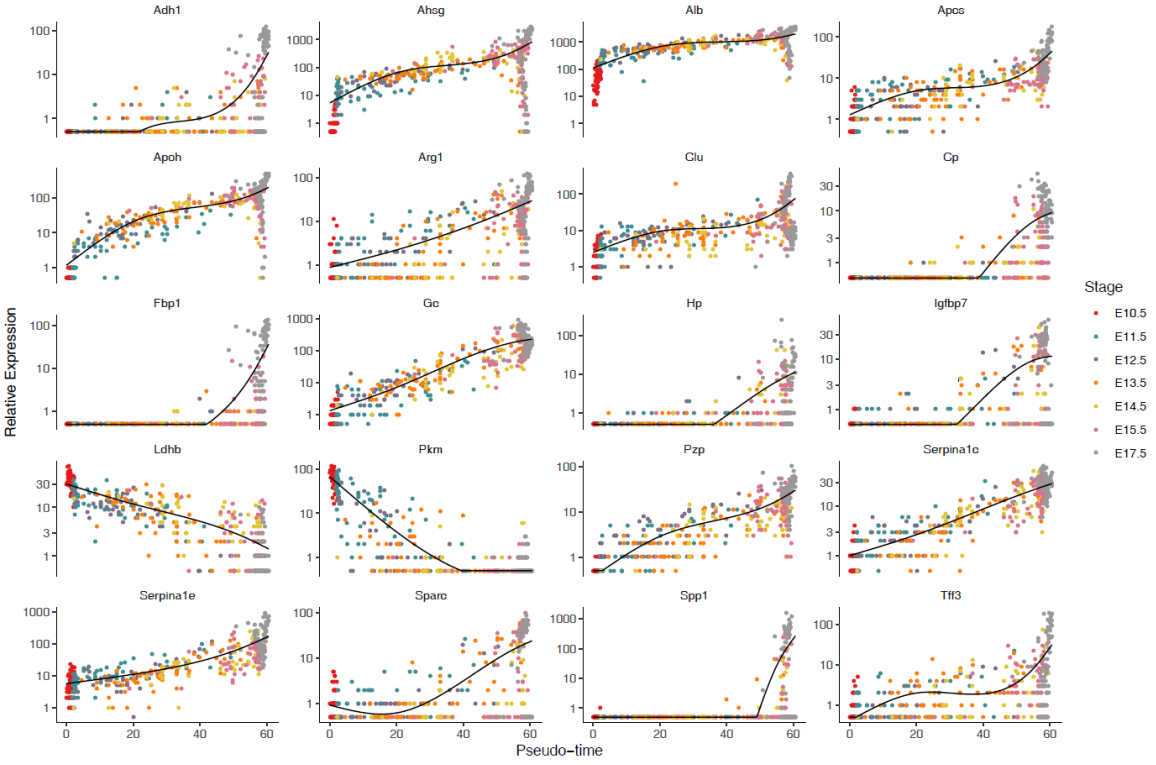


确定了细胞时序后，可分析随着细胞发育进程的基因表达变化。与选取基因集合进行降维的方法 B 相似同样采用 `differentialGeneTest` 函数，只是指定`"~sm.ns(Pseudotime)"`模型进行差异表达分析。

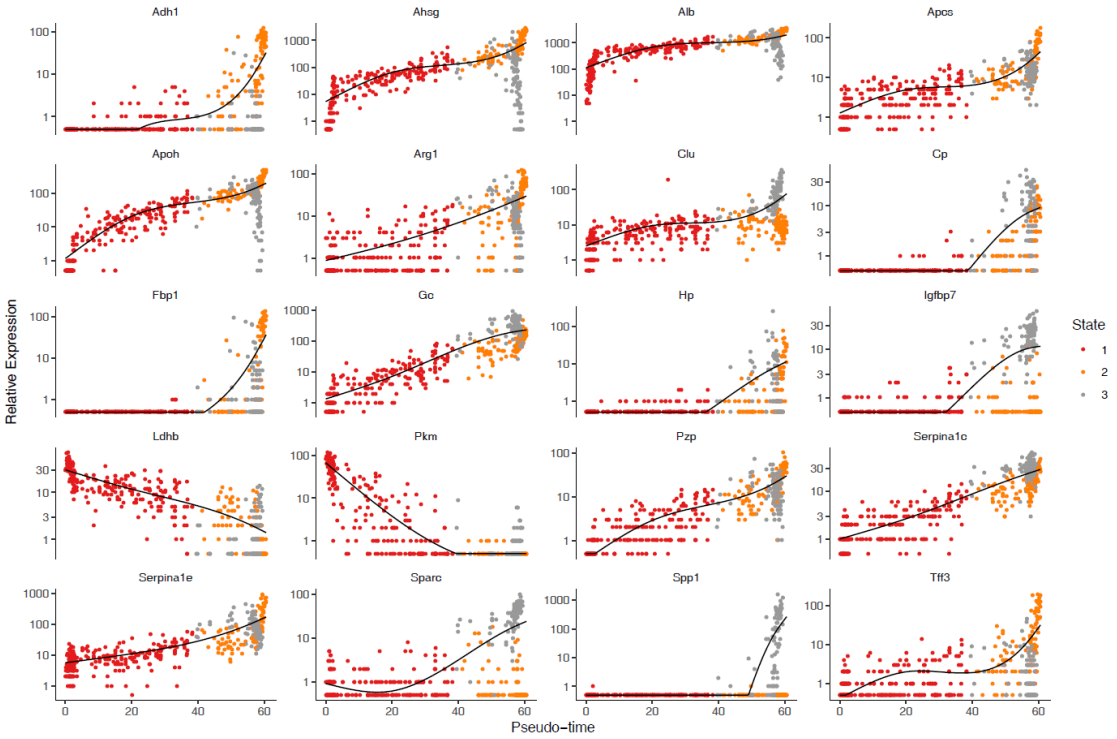
```
## Finding genes that change as a function of pseudotime
diff_test_res <- differentialGeneTest(cd[expressed_genes,],fullModelFormulaStr ="~sm.
ns(Pseudotime)")
diff_test_res <- diff_test_res[order(diff_test_res$qval),]
```

根据 `qval` 选取 top 20 差异基因进行表达量随 pseudotime 动态变化的可视化，用颜色标记 `Stage` 或 `State` 信息。

```
## Plot top 20 genes that change as a function of pseudotime
cds_subset <- cd[rownames(diff_test_res)[1:20],]
plot_genes_in_pseudotime(cds_subset, ncol = 4, color_by = 'Stage')+scale_color_manual
(values = getPalette(length(unique(sample_sheet[,'Stage']))))
```

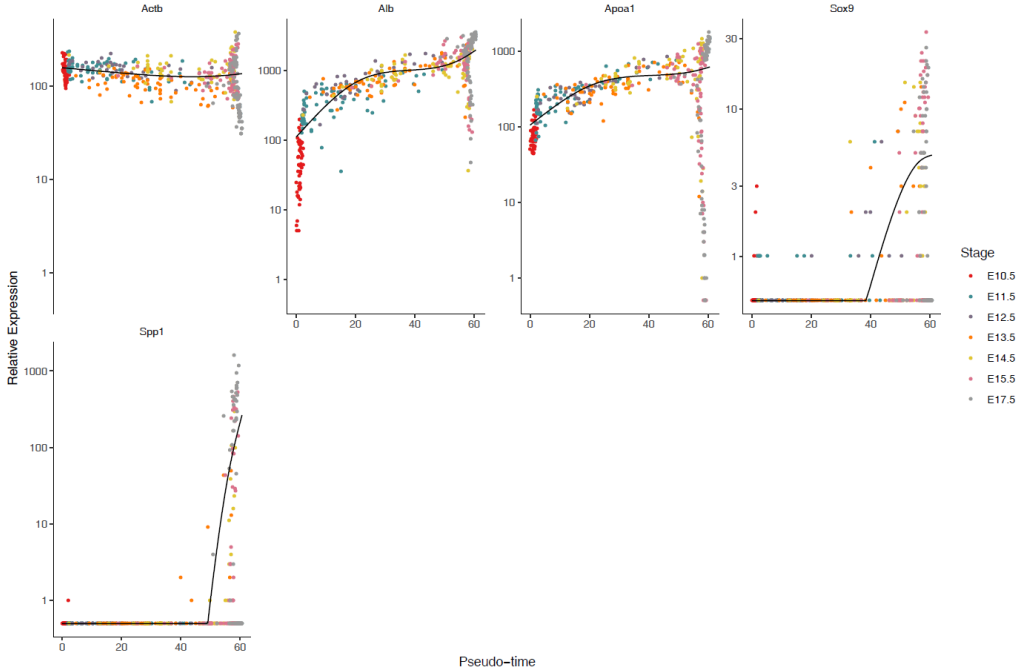



```
plot_genes_in_pseudotime(cds_subset, ncol = 4)+scale_color_manual(values = getPalette
(length(unique(cd@phenoData@data[,"State"]))))
```

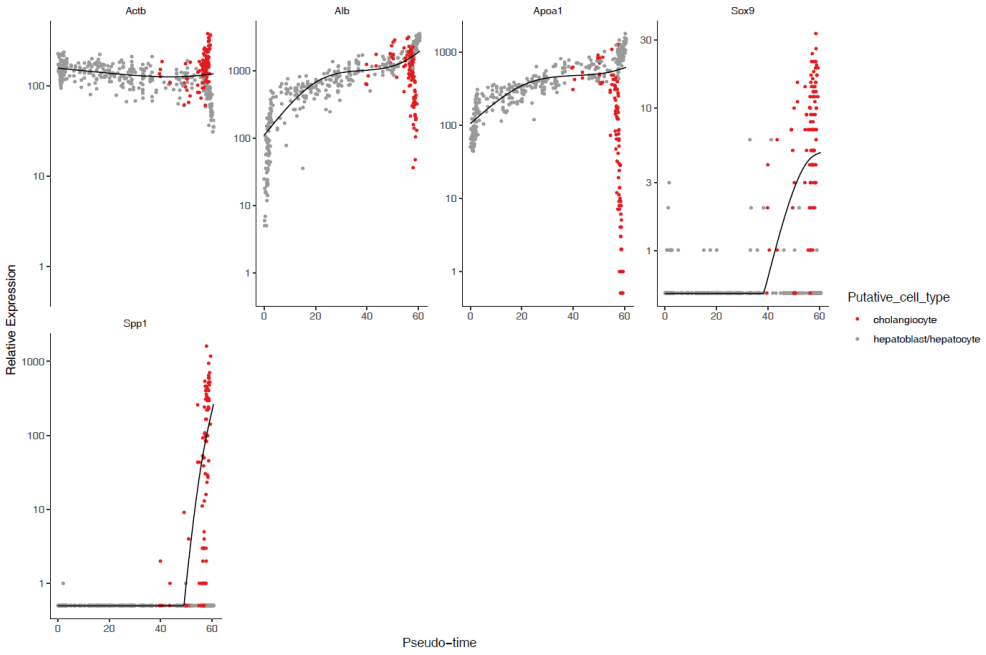


同时，可选取一些感兴趣的基因进行可视化。下图展示的是管家基因 ***Actb***，肝实质细胞 marker: ***Alb***, ***Apoa1*** 和胆管上皮 marker: ***Sox9***, ***Spp1***

```
## Select a couple of markers of housekeeping genes, hepatocyte and cholangiocyte
plot_genes <- row.names(subset(fData(cd), gene_short_name %in% c("Actb","Alb","Spp1",
"Sox9","Apoa1")))
plot_genes_in_pseudotime(cd[plot_genes,], ncol = 4, color_by = 'Stage')+scale_color_m
anual(values = getPalette(length(unique(sample_sheet[,'Stage']))))
```

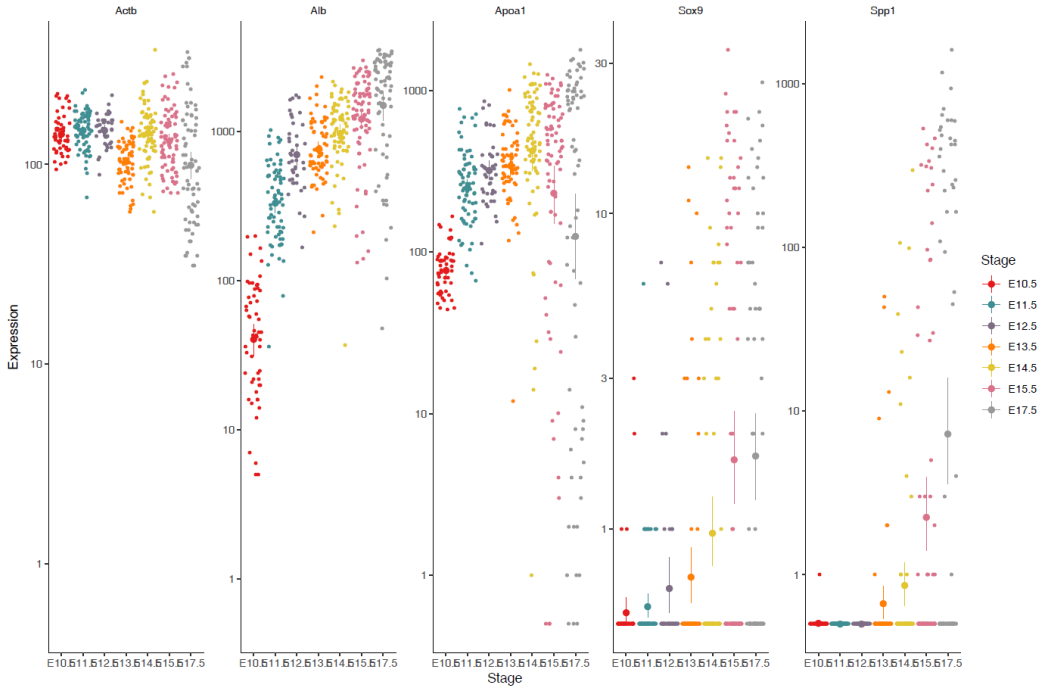


```
plot_genes_in_pseudotime(cd[plot_genes,], ncol = 4, color_by = 'Putative_cell_type')+
scale_color_manual(values = getPalette(length(unique(sample_sheet[,'Putative_cell_type'])))
```

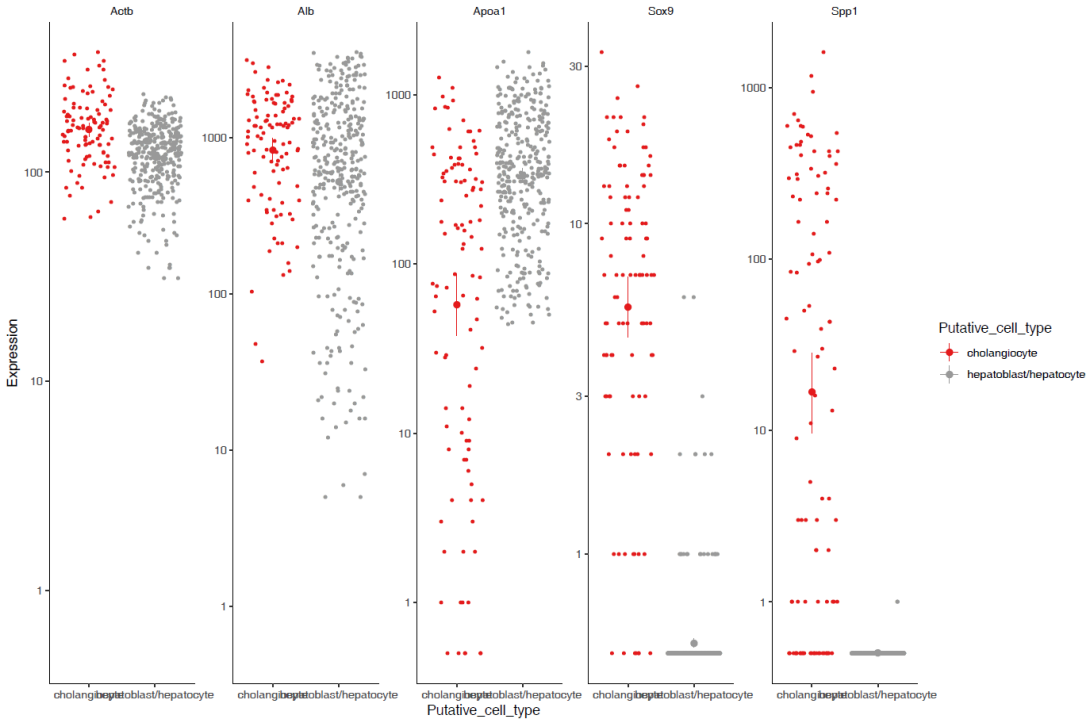


除了可对基因进行表达量随 pseudotime 动态变化的可视化外，还可根据不同的细胞分组进行可视化。

```
plot_genes_jitter(cd[plot_genes,], grouping="Stage", color_by="Stage",nrow=1, ncol=NULL, plot_trend=TRUE)+scale_color_manual(values = getPalette(length(unique(sample_sheet[,'Stage'])))
```

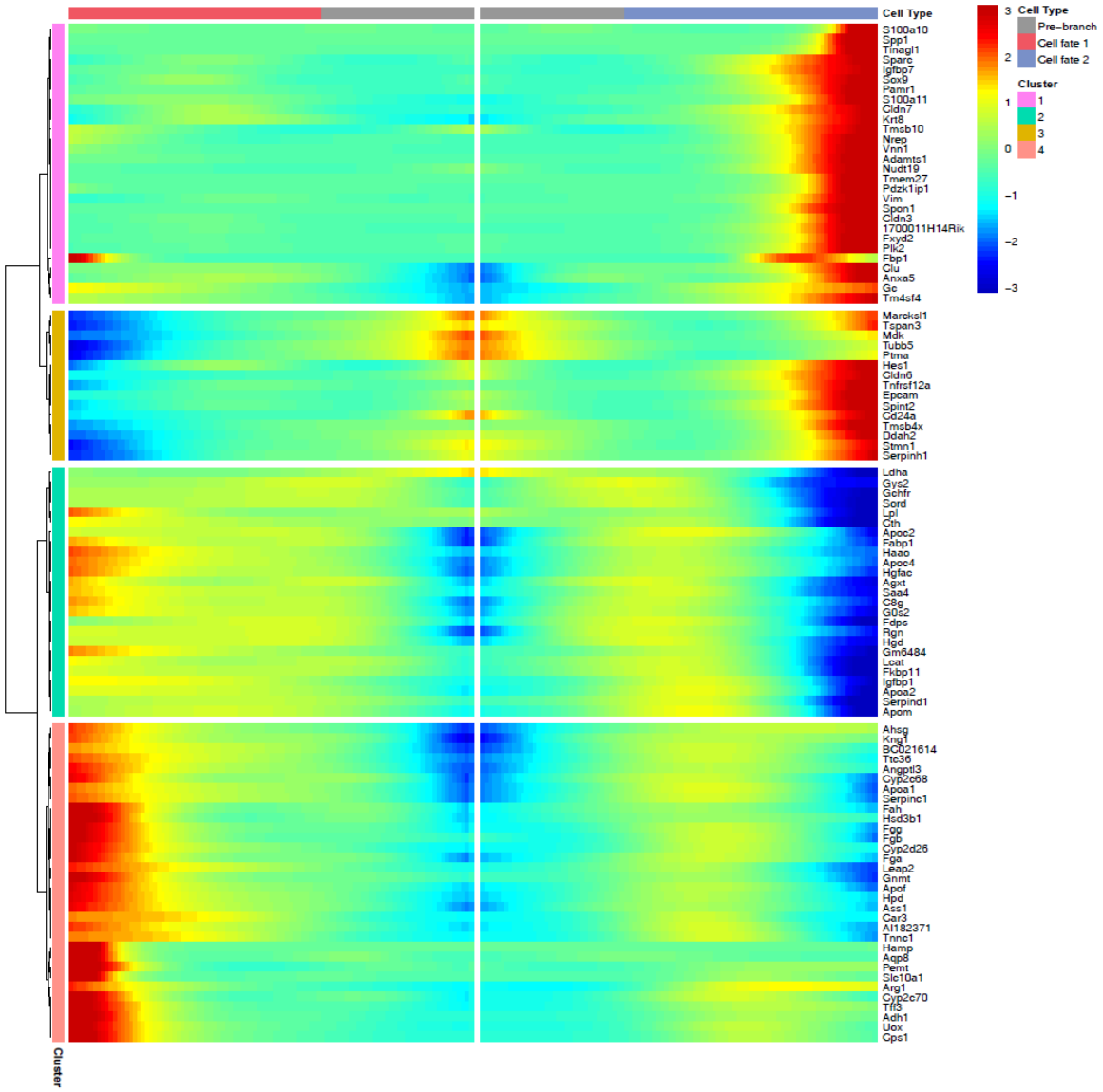


```
plot_genes_jitter(cd[plot_genes,], grouping="Putative_cell_type", color_by="Putative_cell_type",nrow=1, ncol=NULL, plot_trend=TRUE)+scale_color_manual(values = getPalette(length(unique(sample_sheet[,'Putative_cell_type']))))
```



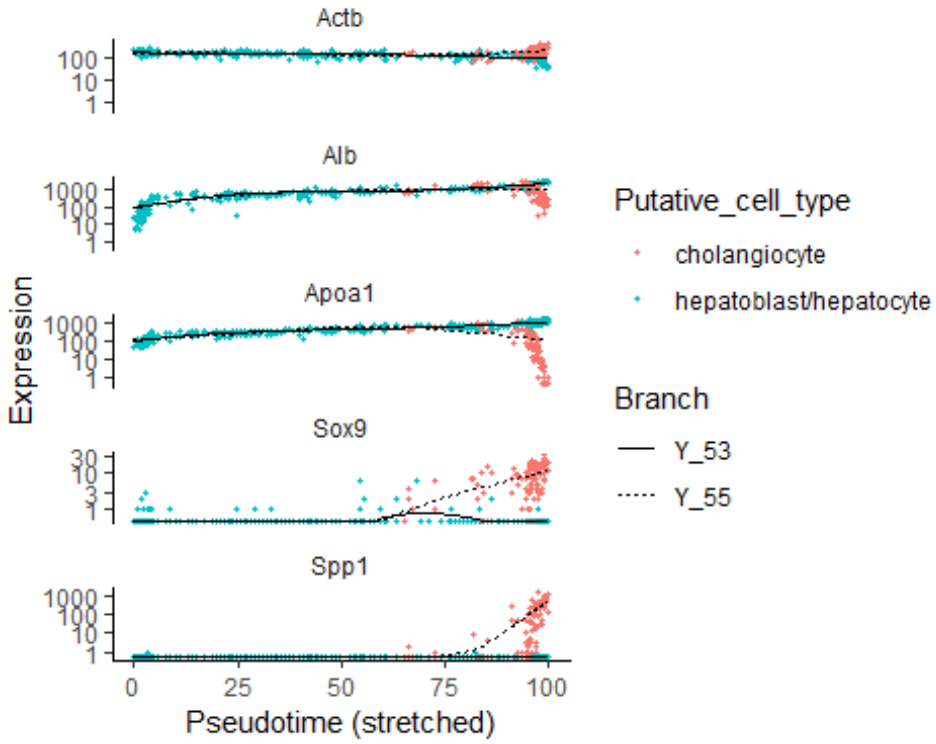
本例中的时序分析结果中可明显看到 1 个发育分支，分别对应着成肝细胞分化为肝实质细胞和肝内胆管细胞。可使用 **BEAM** 函数分析分支点中哪些基因与不同细胞命运选择的相关。

```
## Analyzing branches in single-cell trajectories
BEAM_res <- BEAM(cd, branch_point=1, cores = 1)
BEAM_res <- BEAM_res[order(BEAM_res$qval),]
BEAM_res <- BEAM_res[,c("gene_short_name", "pval", "qval")]
## Visualize changes for top 100 genes that are significantly branch dependent
plot_genes_branched_heatmap(cd[row.names(BEAM_res[1:100,]),],branch_point = 1,num_clusters = 4,cores = 1,use_gene_short_name = T,show_rownames = T)
```



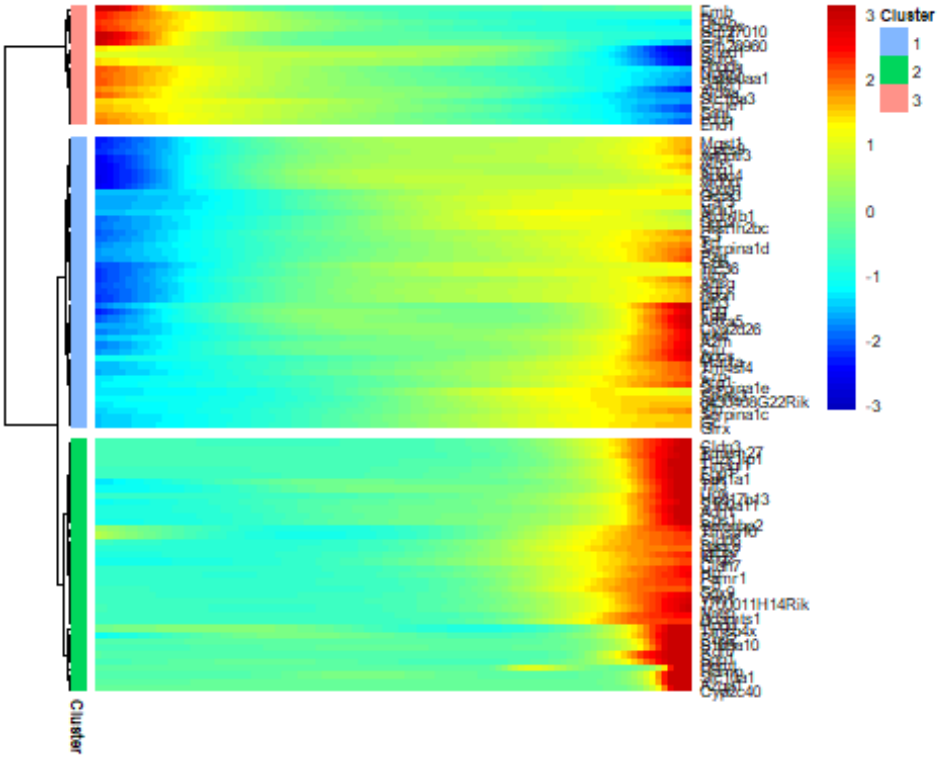
利用 `plot_genes_branched_pseudotime` 可分析基因在不同分化方向的表达变化趋势。

```
## Select a couple of markers of housekeeping genes, hepatocyte and cholangiocyte to visualize changes along brabch
plot_genes <- row.names(subset(fData(cd), gene_short_name %in% c("Actb","Alb","Spp1","Sox9","Apoa1")))
plot_genes_branched_pseudotime(cd[plot_genes,],branch_point=1,color_by="Putative_cell_type",ncol=1)
```



plot_pseudotime_heatmap 可以通过将随时间变化具有相似变化趋势的基因进行分组，可用于后续分析这些不同变化趋势的基因它们共同的生物学功能。

```
## Clustering top 100 genes by pseudotemporal expression pattern
if(nrow(diff_test_res)>100){
  sig_gene_names <- row.names(diff_test_res[1:100,]) # Select top 200 gene used to cl
uster
}else{
  sig_gene_names <- row.names(diff_test_res)
}
plot_pseudotime_heatmap(cd[sig_gene_names,],num_clusters = 3, cores = 1,show_rownames
 = T)
```



三、 其他说明

本文档主要针对 TPM 相关表达数据基于 monocle2 的时序分析和可视化操作说明。更多的 monocle2 使用方法请浏览官网上的详细使用手册：

http://cole-trapnell-lab.github.io/monocle-release/docs/