Universidade de Brasília - UnB
Instituto de Ciências Exatas - IE
Departamento de Estatística - EST

Implementação Computacional de Algoritmos para Agrupamento de Dados HDLSS e HDLLSS

Rafael da Silva Lins

Orientador: Professor George von Borries

Brasília

2018

Rafael da Silva Lins

Implementação Computacional de Algoritmos para Agrupamento de Dados HDLSS e HDLLSS

Relatório final apresentado à disciplina de Trabalho de Conclusão de Curso II de graduação em Estatística, Instituto de Exatas, Universidade de Brasília, como parte dos requisitos necessários para obtenção do grau de Bacharel em Estatística.

Orientador: Professor George von Borries

Brasília

2018

Rafael da Silva Lins
Implementação Computacional de Algoritmos para Agrupamento de Dados HDLSS e HDLLSS/ Rafael da Silva Lins. – Brasília, 2018-
47 p. : il. (algumas color.) ; 30 cm.

Orientador: Professor George von Borries

Relatório Final – Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Estatística
Trabalho de Conclusão de Curso de Graduação, 2018.

1. algoritmos de agrupamento. 2. dados HDLSS. 3. dados HDLLSS. 4. análise de microarranjo.

Rafael da Silva Lins

Implementação Computacional de Algoritmos para Agrupamento de Dados HDLSS e HDLLSS

Relatório final apresentado à disciplina de Trabalho de Conclusão de Curso II de graduação em Estatística, Instituto de Exatas, Universidade de Brasília, como parte dos requisitos necessários para obtenção do grau de Bacharel em Estatística.

Trabalho aprovado. Brasília, 4 de julho de 2018:

Professor George von Borries
Orientador

Professora Joanlise Marco de Leon Andrade
Membro da Banca

Professor André Luiz Fernandes Cançado
Membro da Banca

Brasília
2018

A Deus
A meus pais
A meus irmãos

Agradecimentos

Primeiramente, agradeço a Deus, por me guiar nessa caminhada.

Aos meus pais, por todo o apoio, atenção e dedicação.

Aos meus irmãos, por todo o carinho e amizade.

Aos professores que fizeram parte da minha formação, em especial ao Prof. George von Borries, meu orientador, pela atenção, pela compreensão, pela confiança e conhecimentos compartilhados comigo.

À ESTAT Consultoria, pelas experiências proporcionadas, crescimento profissional e pessoal.

A toda equipe do Departamento de Estatística da UnB, pela dedicação aos alunos da graduação.

Por fim, agradeço a todos os meus amigos que acompanharam, ajudaram e me apoiaram durante esses anos de estudo.

Resumo

Este trabalho implementa em um pacote R um conjunto de algoritmos destinados a agrupar variáveis em uma estrutura de dados superdimensionada com amostras pequenas (HDLSS, *High Dimensional Low Sample Size*) e dados com estrutura superdimensionada, longitudinal com amostras pequenas (HDLLSS, *High Dimensional Longitudinal Low Sample Size*). Esses algoritmos utilizam como medida de similaridade o p-valor obtido a partir de dois testes estatísticos distintos: um não-paramétrico, que testa a ausência de efeito simples de grupo e outro, que avalia a ausência de efeito simples de grupo em um delineamento fatorial com medidas repetidas no tempo. Aplicações em dados de microarranjo apresentam resultados promissores. Os estudos de simulação sugerem que os algoritmos de agrupamento implementados tiveram um desempenho interessante ao detectar grupos em dados HDLSS e HDLLSS.

Palavras-chave: algoritmos de agrupamento, PPCLUST, PPCLUSTEL, dados HDLSS, dados HDLLSS, análise de microarranjo.

Abstract

This study implements a R package with a set of algorithms to cluster variables in high dimensional low sample size (HDLSS) data and high dimensional longitudinal low sample size (HDLLSS) data. These algorithms are based on the use of a p-value from two diferents statistical tests as a similarity measure for the clustering procedure: a nonparametric rank test of homogeneous distribution between groups of variables and a test of no simple effect on factorial design with measures observed over time. Applications on microarray data show promising results. The simulation studies reveal that the implemented clustering algorithms show an interesting performance to detecting groups in HDLSS and HDLLSS data structures.

Keywords: clustering algorithms, PPCLUST, PPCLUSTEL, HDLSS data, HDLLSS data, microarray analysis.

Sumário

Introdução

Os algoritmos de agrupamento emergiram como uma poderosa ferramenta para detecção de padrões subjacentes dos dados. Existem vários métodos propostos na literatura, tais como o agrupamento baseado em modelo, agrupamento hierárquico, k-médias e outros. A mistura de distribuições gaussianas é um método clássico tido como estado da arte e vem sendo aplicado com sucesso em muitos problemas práticos.

O agrupamento de dados superdimensionados e com amostras pequenas exerce um papel importante em várias aplicações modernas, em especial na bioinformática em análise de microarranjo, imagens médicas, sinais biopotenciais e assim por diante. Esses dados são referidos na literatura como superdimensionados, com amostras pequenas (HDLSS, *High Dimensional Low Sample Size*) e longitudinal superdimensionado com amostras pequenas (HDLLSS, *High Dimensional Longitudinal Low Sample Size*).

As principais dificuldades associadas ao agrupamento ou classificação de dados superdimensionados resultam da alta dimensionalidade (ou seja, grande número de variáveis) e da disponibilidade de apenas um pequeno número de amostras, ou seja, o tamanho da amostra n é fixo ou $n/d \to 0$ sendo a dimensão dos dados $d \to \infty$; e, ainda, com a solução de algumas questões básicas tais como definir o número de grupos e lidar com *outliers*, sendo necessária a seleção de modelo ou análise posterior para determinar o agrupamento mais apropriado.

von Borries (2008) desenvolveu o algoritmo PPCLUST[1] e sua extensão, PPCLUSTEL[2] que contornam esses obstáculos, os quais estão implementados no *software* estatístico SAS[3]. Ambos os algoritmos são adeptos do paradigma de agrupamento sólido e adotam como medida de dissimilaridade o p-valor obtido de testes de Análise de Variância (ANOVA) para detectar grupos (ou *clusters*) por meio da avaliação do desvio da hipótese nula.

O PPCLUST tem como base a metodologia de ANOVA não-paramétrica e se destina a agrupar variáveis com a mesma distribuição de dados com o formato HDLSS. Já o PPCLUSTEL permite agrupar variáveis que se comportam de maneira similar ao longo do tempo de dados no formato HDLLSS. Em outras palavras, essa extensão identifica grupos de variáveis com base na metodologia de ANOVA com delineamento fatorial e medidas repetidas no tempo. A ideia é que, ao final desses procedimentos, os grupos tenham variância alta entre si e dentro de um mesmo grupo essa variância seja pequena sem a necessidade de uma prévia especificação do número de grupos.

---

[1] Do inglês: *P*-values Based Partitional Clustering
[2] Do inglês: *P*-values Based Partitional Clustering of Longitudinal Data
[3] Linguagem Macro SAS Versão 9.2

O SAS é um *software* pago, isso limita o acesso ao uso desses algoritmos por parte da comunidade científica. Nesse sentido a disponibilidade desses procedimentos em um *software* livre como o **R** (R Core Team, 2017) torna acessível o uso desses métodos para o desenvolvimento e reprodução de trabalhos científicos.

Além disso, há um requisito para o desenvolvimento de algoritmos que sejam rápidos e agrupem dados superdimensionados mantendo-se a qualidade do agrupamento. O uso de processamento paralelo possibilita o aproveitamento dos recursos computacionais disponíveis ampliando a escalabilidade dos métodos sem perder na qualidade do agrupamento.

O MCLUST (Fraley and Raftery, 1999) é um pacote popular do **R** e amplamente utilizado em produções científicas no contexto de agrupamento. Os trabalhos de von Borries (2008) e Silva (2012) utilizaram esse pacote em estudos de simulação e o mesmo se destacou pelos resultados competitivos. Dessa maneira, o MCLUST será utilizado neste trabalho para a comparação com os algoritmos implementados.

Nesse contexto, este trabalho implementará em pacote **R** os algoritmos PPCLUST e PPCLUSTEL, assim como o PPCLUST-H e o PPCLUSTEL-H, duas versões reprojetadas dos procedimentos originais, para empregar o processamento paralelo. A avaliação dos algoritmos será feita por meio do índice ARI (*Adjusted Rand Index*) e do tempo de processamento em dados simulados.

Para mostrar a eficácia dos algoritmos, serão utilizados dois conjuntos de dados de microarranjo: expressão gênica de adenoma e câncer de cólon; leveduras expostas a variações de temperatura.

A estrutura deste trabalho está dividida da seguinte forma: no Capítulo 1, é realizada uma breve revisão da literatura que aborda o desenvolvimento de pacotes e processamento paralelo em **R**, uma visão geral sobre os algoritmos de agrupamento, o método de mistura de distribuições gaussianas presente no pacote MCLUST e o índice ARI. No Capítulo 2, são descritos os testes contidos nos algoritmos. O Capítulo 3 discorre sobre a implementação dos algoritmos no **R**. Por fim, no Capítulo 4, o desempenho dos algoritmos é comparado a partir de estudos de simulações e, em dados de microarranjo.

1 Revisão de Literatura

1.1 Desenvolvimento de pacotes em R

Definição de um pacote

O **R** é um *software* e linguagem para análises estatísticas e gráficas de código aberto, desenvolvido nos laboratórios Bell por John Chambers e colegas originado a partir da linguagem **S**. Um dos motivos para que o **R** seja tão bem-sucedido é que há uma chance alta de que alguém já tenha disponibilizado um pacote que resolva o problema em que se está trabalhando, e todos o usuários podem se beneficiar com isso baixando tal pacote.

Um pacote agrupa códigos, dados e documentação que podem ser difundidos universalmente para usuários do **R** por meio da distribuição CRAN (*Comprehensive R Archive Network*), que é amparada por uma comunidade ativa de usuários comprometidos com o desenvolvimento da própria linguagem e que contava com mais de 6.000 pacotes em 2015 (Wickham, 2015) para propósitos diversos desde o cálculo de estatísticas simples até ferramentas poderosas modernas, como para análise de dados complexos e apresentações gráficas.

A instalação e utilização de um pacote é muito simples. Por exemplo, pode-se

- Instalar o pacote do CRAN: `install.packages("mclust")`;
- Carregar o pacote: `library(mclust)`;
- Obter manuais de ajuda do pacote: `?mclust` e `help(package = "mclust")`.

O *software* **R** pode ser executado em vários sistemas operacionais tais como Windows, Linux e Mac OS. A sua distribuição principal já vem com oito pacotes padrões inclusos. O **R** está disponível para download em <http://cran.us.r-project.org/>.

Estrutura de um pacote

A criação de um pacote precisa atender a requisitos mínimos para ser utilizável. Cada arquivo e subdiretório tem um respectiva função. A Figura 1 ilustra a estrutura de um pacote **R**.

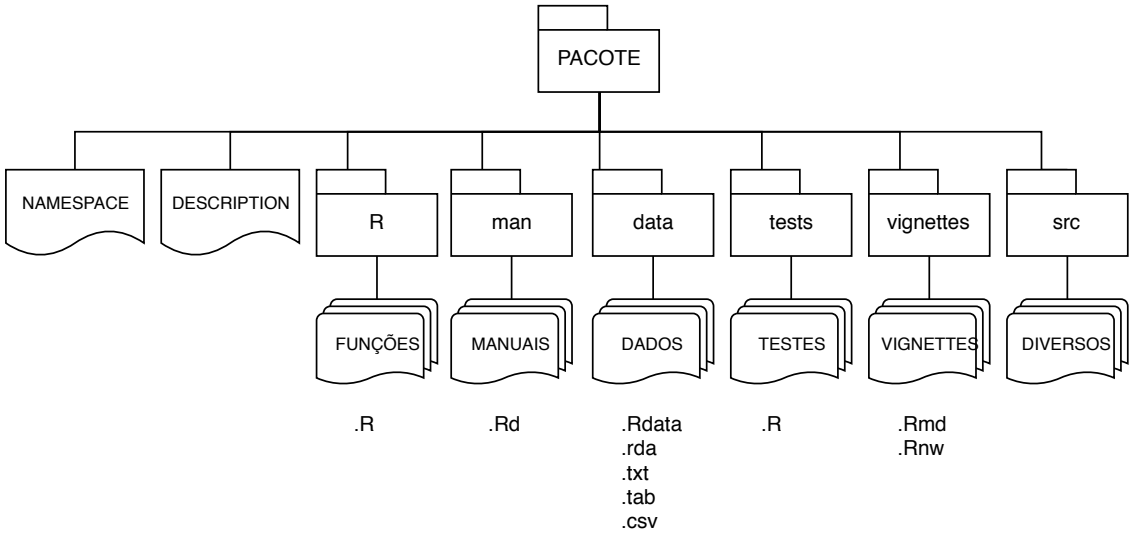


Figura 1 – Estrutura de um pacote R (Adaptado de <methodsblog.wordpress.com>).

- O arquivo NAMESPACE contém informação sobre as funções inclusas no pacote, diferenciando métodos e funções (Wickham, 2015, Cap. 8).
- O arquivo DESCRIPTION basicamente determina como o pacote funcionará com outros pacotes. Nele há descrições do pacote, autor e condições da licença em um formato de texto legível para pessoas e computadores.
- **R**: códigos em R, funções básicas que vão gerenciar e calcular as operações do pacote.
- **man**: arquivos para documentação de ajuda.
- **data**: conjuntos de dados.
- **tests**: testes de validação para identificar possíveis inconsistências nos códigos.
- **vignettes**: documentos que ensinam seus usuários a resolver problemas reais com as ferramentas disponíveis no pacote.
- **src**: conteúdos diversos como códigos em outra linguagem, imagens etc.

Todos os subdiretórios, exceto o arquivo DESCRIPTION, são opcionais, embora qualquer pacote útil tenha um subdiretório **man**/, pelo menos um **R**/ e um **data**/.

Criando e submetendo o pacote

Serão apresentadas as etapas básicas necessárias para criar um pacote R usando o RStudio, interface do **R** que tráz uma série de facilidades para a criação de um pacote tais como caminhos *point-and-click* para criar os subdiretórios, a compilação do pacote e outros.

O primeiro passo na criação de um pacote envolve a instalação do pacote devtools executando o comando: `install.packages ("devtools")`, esse pacote fornece um séries de

funções que simplificam muitas tarefas comuns do processo de desenvolvimento de um pacote.

Caso o pacote use código binário (por exemplo, C ou C ++) será necessário instalar um conjunto de ferramentas de desenvolvimento. No sistema operacional Windows, faça o *download* e instale a ferramenta **Rtools**; no Mac, instale as ferramentas de linha de comando do **Xcode**; no Linux, instale o pacote de desenvolvimento **R**, geralmente chamado de **r-devel** ou **r-base-dev**.

Os passos para a criação dos diretórios são como se segue:

1. Clicar em Arquivo → "Novo Projeto".
2. Escolher "Novo Diretório".
3. Selecionar"Pacote R".
4. Finalmente, nomear o pacote e clicar em "Criar Projeto".

Ou, alternativamente, executar a seguinte linha de comando no console:

```
devtools::create("diretorio/para/pacote/nomepacote").
```

A partir deste ponto as próximas etapas consistirão no desenvolvimento do pacote em si.

- Criar/Editar dos arquivos .R;
- Criar dos arquivos .Rd (manuais);
- Editar do arquivo DESCRIPTION e NAMESPACE;
- Realizar testes de funcionamento do pacote;
- Compilar o pacote.

Os manuais no formato de documentação **R** possuem referências e exemplos para cada uma das funções e conjunto de dados; os testes de funcionamento servem para checar se o pacote será instalado, se rodará os exemplos, se a documentação está completa e se executa corretamente as operações.

Então, o pacote pode ser opcionalmente submetido ao CRAN. Para mais informações, consultar Creating R Packages: A Tutorial (Leisch, 2018) e Writing R Extensions (R Core Team, 2007).

1.2 Processamento Paralelo no R

O **R** não foi projetado para explorar o paralelismo de forma nativa. Em vez disso, ele depende de bibliotecas de pacotes externos especificamente implementados para permitir o uso de estruturas de processamento paralelo de determinadas funções.

A tecnologia de computação paralela no **R** está em expansão desde da versão 2.14 (fevereiro de 2012) com o pacote **parallel** instalado por padrão. Este pacote inclui uma série de mecanismos diferentes para explorar o paralelismo utilizando múltiplos núcleos no(s) processador(es) de uma máquina, ou estender os recursos para um *cluster* de máquinas unindo as funcionalidades de dois pacotes: **snow** e **multicore**.

O pacote parallel cria novos processos para serem executados em diferentes núcleos; mas de maneiras muito diferentes (PSOCK, FORK e MPI). Este trabalho se concentra no tipo de cluster PSOCK (*cluster* de soquetes) para trabalhar no contexto de uma única máquina; serão criadas várias instâncias do **R** para executar ao mesmo tempo em uma máquina com vários núcleos.

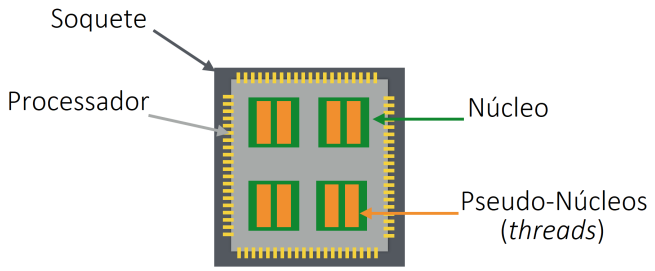


Figura 2 – Esquema de um processador *multicore* (github.com/ljdursi).

Tomaremos como exemplo, a tarefa de encontrar o valor mínimo em um vetor de dados com muitos valores numéricos. A função min() é executada em série - usando um núcleo do processador - para iterar sobre o vetor e encontrar o menor valor. Em especial, essa tarefa pode ser dividida em tarefas menores que podem ser executadas independentemente. Daí vem a ideia de usar o processamento paralelo para resolver problemas como esse em um tempo menor. Se um processador contém quatro núcleos como ilustra a Figura 2, cada núcleo pode receber tarefas que podem ser executadas simultaneamente. Dessa forma, o vetor de dados pode ser dividido em quatro partições de mesmo tamanho em que cada núcleo fica responsável por achar o menor valor em cada partição. Logo após, os resultados são reunidos e se computa o mínimo global.

Um conjunto de processos paralelos em **R** é possibilitado usando a função makeCluster(). Em seguida, utiliza-se este cluster ao executar uma função em paralelo. Existe uma vantagem principal para essa abordagem. Um exemplo de função paralela é parLapply que tomando um objeto tipo lista, distribuirá as tarefas para todos os processos de trabalho do **R** uniformemente e, em seguida, reunirá os resultados de volta. O código a seguir realiza um *bootstrap* computando um milhão de médias sem e com o processamento em dois núcleos do processador. O tempo é medido e mostrado em seguida.

```
dados <- rnorm(100)
media <- function(X, d) mean(X[d])
```

```
library(boot)

system.time({
b1 <- boot(dados, statistic = media, R = 1000000)
})
# usuario   sistema  decorrido
# 24.51      0.28      25.05

library(parallel)

cl <- makeCluster(2)
clusterEvalQ(cl, library(boot))
clusterExport(cl, c("dados","media"))
r <- vector('list', 2)

system.time({
b2 <- parLapply(cl, r, function(x) boot(dados, media, R = 500000))
})
# usuario   sistema  decorrido
# 0.02      0.02      12.78
stopCluster(cl)
```

Basicamente, a quantidade de um milhão de cálculos da média foi dividida para dois núcleos do processador. Com isso, houve um ganho significativo na velocidade de processamento. Verifica-se que o tempo decorrido[1] inicial foi 25,05 segundos, enquanto utilizando o processamento paralelo esse tempo caiu para 12,78 segundos.

1.3 Algoritmos de Agrupamento

Análises de agrupamento consistem em um grupo de técnicas computacionais cujo propósito é identificar grupos ou padrões de comportamento dos objetos em estudo. A ideia básica consiste no uso de procedimentos que unem um método e/ou um algoritmo e uma medida de dissimilaridade com o intuito colocar em um mesmo grupo objetos que sejam similares entre si. Além do mais, análises de agrupamento se mostram ferramentas úteis na compreensão de estruturas multivariadas de dados. Além disso, são utilizadas na redução dos dados, no estabelecimento e validação de hipóteses, e, na classificação baseada no grupos formados (Theodoridis and Koutroumbas, 2008).

De forma resumida, os procedimentos dividem-se em sólidos (*hard*) e difusos (*fuzzy*). No agrupamento sólido cada objeto pertence a somente um grupo, enquanto que no

[1] Diferença entre o tempo inicial e final.

agrupamento difuso um objeto pode pertencer a mais de um grupo, mas com determinados graus de pertinência. Os métodos de agrupamento tradicionais podem ser classificados como mostra a Figura 3.

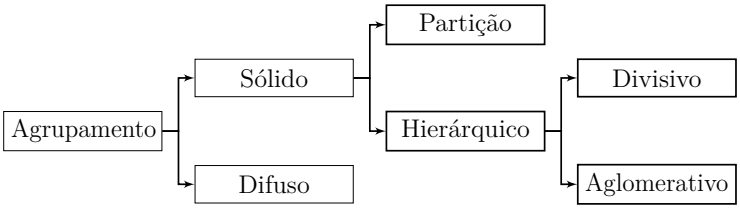


Figura 3 – Diagrama dos algoritmos de agrupamento.

Na abordagem hierárquica há dois ramos diferentes: divisivo e aglomerativo. No primeiro caso, a princípio os dados são vistos como pertencentes a um único grupo (*cluster*). Os elementos são então divididos pela sua dissimilaridade em grupos distintos até que cada elemento seja representado por um conglomerado. Isso significa que, ao final dessa etapa, a quantidade de elementos será igual à de conglomerados. Já na abordagem hierárquica aglomerativa acontece o processo contrário, a princípio, cada elemento representa um conglomerado. A partir deste ponto os conglomerados são combinados em passos sucessivos dando origem a outros até que todos os elementos pertençam ao mesmo conglomerado.

O agrupamento hierárquico é convencionalmente apresentado em um dendrograma. Uma inconveniência com a utilização desse tipo de representação é que quanto maior o tamanho da amostra mais sua análise passa a ficar prejudicada. Na análise de microarranjo, por exemplo, essa representação poderá ser muito dificultosa devido ao tamanho do conjunto de dados que pode conter informação de milhares de genes. A Figura 4 exibe o andamento do processo de agrupamento hierárquico. De maneira didática, considere que o agrupamento hierárquico divisivo tem o sentido de cima para baixo, enquanto agrupamento hierárquico aglomerativo de baixo para cima.

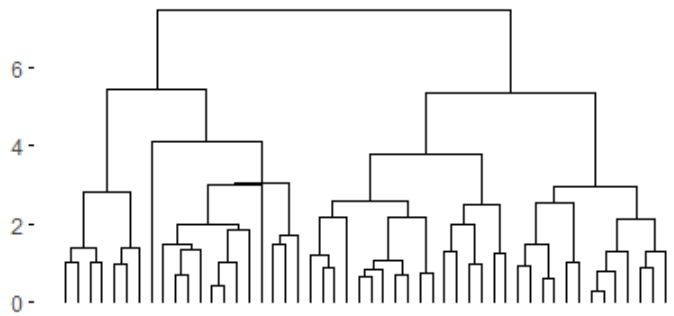


Figura 4 – Dendograma (n = 50).

Alguns algoritmos hierárquicos bastante utilizados são o AGNES (*Aglomerative Nesting*) e o DIANA (*Divisive Analysis*). Ambos estão disponíveis no pacote CLUSTER do

R. Os detalhes sobre esses algoritmos podem ser encontrados no trabalho de Kaufman and Rousseeuw (2009).

Em contraste com os métodos hierárquicos, os procedimentos de agrupamento sólido são mais flexíveis, no sentido de que um objeto alocado inicialmente em um grupo possa ser realocado diversas vezes durante o processo de agrupamento.

No agrupamento sólido os objetos são agrupados sem que haja sobreposição, ou seja, cada objeto pode pertencer a apenas um grupo. A ideia básica é consiste em dividir de maneira ótima os objetos a serem alocados em um número fixo de grupos. Exemplos de algoritmos com esse tipo de abordagem são o k-médias (MacQueen, 1967), PAM (*Partitioning Around Medoids*) e CLARA (*Clustering Large Applications*). Os detalhes sobre os dois últimos algoritmos podem ser encontrados em Kaufman and Rousseeuw (2009).

1.4 Mistura de Distribuições Gaussianas

Em uma análise de agrupamento baseada em modelo, os dados são vistos como provenientes de uma mistura de distribuições de probabilidade. Essa abordagem pode ser classificada como um procedimento hierárquico aglomerativo que modela uma mistura de distribuições aos dados para fornecer a probabilidade de um objeto i pertencer ao grupo k.

O pacote MCLUST (Fraley and Raftery, 1999) fornece funcionalidade para análise de cluster combinando agrupamento hierárquico baseado em modelo. Esse assume que os dados seguem uma mistura de distribuições normais sendo a distribuição conjunta composta por G componentes, em que cada componente é um grupo. Considere a mistura

$$f(x) = \sum_{i=1}^{G} \pi_i f_i(x|\mu_i, \sum_i),$$

em que os dados são representados por x, π_i é a probabilidade de uma observação pertencer à i-ésima componente ($\pi_i \in [0, 1]$ e $\sum_{i=1}^{G} \pi_i = 1$), f_i é a função densidade de probabilidade gaussiana do i-ésimo grupo com os parâmetros μ_i (média) e $\sum_i$ (matriz de covariâncias). Os parâmetros dessa mistura de distribuições são estimados pelo algoritmo *Expectation-Maximization* (EM) e o objeto é alocado ao componente que possui a maior probabilidade de pertinência. Ao final do procedimento, os componentes da mistura que foram estimados são os grupos formados.

Raftery et al. (2016) definem um conjunto de formas para $\sum_i$ (Tabela 1) que fornece informações sobre o volume, forma e orientação dos componentes. Seja a decomposição espectral da matriz $\sum_i = \lambda_i D_i A_i D_i^T$, em que D_k é a matriz ortogonal de autovetores de $\sum_i$, A_i é uma matriz diagonal cujos elementos são proporcionais aos autovalores de $\sum_i$, e λ_i é um escalar. A orientação dos componentes principais de $\sum_i$ é determinada por D_i;

A_i determina a forma das curvas de nível da densidade do i-ésimo grupo; $\lambda_i \propto \lambda_i^d |A_i|$ especifica o volume da elipsoide, sendo d o número de dimensões dos dados.

Modelo	Distribuição	Volume	Forma	Orientação
λI	Esférica	fixo	fixa	NA
$\lambda_i I$	Esférica	variável	fixa	NA
λA	Diagonal	fixo	fixa	Eixos coord.
$\lambda_i A$	Diagonal	variável	fixa	Eixos coord.
λA_i	Diagonal	fixo	variável	Eixos coord.
$\lambda_i A_i$	Diagonal	variável	variável	Eixos coord.
$\lambda D A D^T$	Elipsoidal	fixo	fixa	fixa
$\lambda D_i A D_i^T$	Elipsoidal	fixo	fixa	variável
$\lambda_i D_i A D_i^T$	Elipsoidal	variável	fixa	variável
$\lambda_i D_i A_i D_i^T$	Elipsoidal	variável	variável	variável

Tabela 1 – Possíveis estruturas da matriz de covariância consideradas pelo pacote mclust.

O melhor modelo é selecionado usando o critério de informação bayesiano (BIC, *Bayesian Information Criterion*). Um alto BIC indica forte evidência para o modelo correspondente.

1.5 Avaliação da Qualidade do Agrupamento

A variedade de algoritmos para agrupamento de dados cresceu no passo em que o desenvolvimento tecnológico resultou no aumento dramático na capacidade de coleta e armazenamento de dados. Isso trouxe a necessidade de se desenvolver medidas capazes de comparar o desempenho desses algoritmos e avaliar qual método é o mais apropriado.

Uma medida externa utilizada neste trabalho é o Índice de Rand Ajustado, ou simplemente ARI. Um critério externo é algum resultado padrão para o agrupamento que é julgado correto, ou mesmo o resultado do agrupamento com uma metodologia diferente que alguém quer comparar com outros métodos.

Essa medida é uma correção do RI (Rand, 1971, *Rand Index*) desenvolvida por Hubert and Arabie (1985) e mede a concordância entre duas partições de um mesmo conjunto de elementos . Diferente do RI, no caso em que os grupos estejam aleatoriamente dispostos, seu valor esperado é igual a 0 e tem valor máximo igual a 1.

Considerando, por exemplo, uma partição $R = \{r_1, r_2, ..., r_k\}$ representando k grupos de referência que são usados para comparar os procedimentos de agrupamento e, sendo $V = \{v_1, v_2, ..., v_c\}$ uma partição de c grupos obtidos a partir de algum algoritmo de agrupamento. A Tabela 2, apresenta os resultados de ambas as partições dos dados, R e V, sendo n_{ij} o número de objetos que estão em ambos os grupos r_i e v_j, com $i = 1, ..., k$, $j = 1, ..., c$. Além disso, $n_{i.} = \sum_{j=1}^{c} n_{ij}$ e $n_{.j} = \sum_{i=1}^{k} n_{ij}$.

Grupo	v_1	v_2	...	v_c	Total
r_1	n_{11}	n_{12}	...	n_{1c}	$n_{1.}$
r_2	n_{21}	n_{22}	...	n_{2c}	$n_{2.}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
r_k	n_{k1}	n_{k2}	...	n_{kc}	$n_{k.}$
Total	$n_{.1}$	$n_{.2}$	...	$n_{.c}$	n

Tabela 2 – Tabela cruzada para o agrupamento.

O ARI pode ser calculado por:

$$\frac{\sum_i \sum_j \binom{n_{ij}}{2} - \frac{\sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2}}{\binom{n}{2}}}{\frac{1}{2}[\sum_i \binom{n_{i.}}{2} + \sum_j \binom{n_{.j}}{2}] - \frac{\sum_i \binom{n_{i.}}{2} \sum_j \binom{n_{.j}}{2}}{\binom{n}{2}}}$$

,

O ARI é igual a 1 quando as duas partições concordam exatamente e 0 quando as duas partições concordam não mais do que o esperado por acaso. No contexto de agrupamento, ARI = 0 indica que o algoritmo não foi eficaz em detectar os grupos existentes nos dados.

Supondo duas partições:

$$A = \{(1, 2, 3, 4, 5, 6), (7, 8, 9, 10, 11, 12)\} \text{ e}$$

$$B = \{(7, 8, 9, 10), (1, 2, 11, 12), (3, 4, 5, 6)\}.$$

Grupo	b_1	b_2	b_3	Total
a_1	0	2	4	6
a_2	4	2	0	6
Total	4	4	4	12

Tabela 3 – Tabela cruzada para o agrupamento.

Basicamente, essa medida se baseia no número de objetos que foram alocados juntos e separados tanto em A quanto em B. Notamos que nem todo o quarteto que está no mesmo grupo de B também está no mesmo grupo de A. O ARI calculado para esse exemplo foi igual a 0,36.

A função AdjustedRandIndex() presente no pacote mclust foi utilizada nas simulações deste trabalho para o cálculo do ARI.

2 Agrupamento de Dados HDLSS e HDLLSS com Base em p-valores

Em diversas áreas do conhecimento, o desenvolvimento tecnológico resultou em um aumento significativo na capacidade de coleta de dados. Em dados de microarranjo cada amostra fornece informações de expressões gênicas em milhares a dezenas de milhares de transcritos simultaneamente. Devido à restrições de tempo e custo, comumente é realizado um número relativamente pequeno de amostras em condições experimentais, caracterizando um conjunto de dados superdimensionado.

Os algoritmos baseados em ANOVA apresentados neste trabalho utilizam a ótica de que cada indivíduo representa uma dimensão no espaço gerado pelo dado superdimensionado. O tamanho e o volume dos conjuntos de dados genômicos são suficientes para apresentar aos algoritmos de agrupamento muitos problemas de alocação de memória e velocidade de processamento.

O algoritmo PPCLUST e sua extensão PPCLUSTEL utilizam o p-valor como uma medida de distância para o agrupamento dos dados. Uma vantagem desses procedimentos está no fato de que o p-valor é robusto a valores discrepantes. Isso possibilita um agrupamento com muita precisão e estabilidade, tarefa difícil para os algoritmos tradicionais. Tais procedimentos não enfrentam problemas com alocação de memória quando o número de variáveis no estudo é muito alto, o que representa uma vantagem adicional, visto que o mesmo trabalha exclusivamente na memória RAM (*Random Access Memory*), um recurso computacional ainda "caro". A Tabela 4 mostra o *layout* de dados superdimensionados e com amostras pequenas.

Variável	Observações				Tamanho Amostral
1	X_{11}	X_{12}	...	X_{1n_1}	n_1
2	X_{21}	X_{22}	...	X_{1n_2}	n_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
a	X_{a_1}	X_{a_2}	...	X_{an_a}	n_a

Tabela 4 – Estrutura de dados HDLSS.

O PPCLUSTEL, agrupa dados superdimensionados com amostras pequenas e ao longo do tempo, conforme ilustrado na Tabela 5.

		Tempo			
Variável	Sujeito	t_1	t_2	...	t_b
1	1	X_{111}	X_{121}	...	X_{1b1}
	2	X_{112}	X_{122}	...	X_{1b2}
	⋮	⋮	⋮	⋱	⋮
	n_1	X_{11n_1}	X_{12n_1}	...	X_{1bn_1}
2	1	X_{211}	X_{221}	...	X_{2b1}
	2	X_{212}	X_{222}	...	X_{2b2}
	⋮	⋮	⋮	⋱	⋮
	n_2	X_{21n_2}	X_{22n_2}		X_{2bn_2}
⋮	⋮	⋮	⋮	⋱	⋮
3	1	X_{a11}	X_{a21}	...	X_{ab1}
	2	X_{a12}	X_{a22}	...	X_{ab2}
	⋮	⋮	⋮	⋱	⋮
	n_a	X_{a1n_a}	X_{a2n_a}	...	X_{abn_a}

Tabela 5 – Estrutura de dados HDLLSS.

2.1 O Teste Contido no PPCLUST(-H)

Wang and Akritas (2004) desenvolvem um teste não paramétrico que detecta o efeito de grupo no caso de grande número de níveis de fator. O algoritmo PPCLUST usa o p-valor resultante desse teste como medida de dissimilaridade entre grupos de níveis de fator.

Denote R_{ij} como sendo o posto da observação X_{ij} no conjunto de todas $n_1 + n_2 + ... + n_a$ observações. Os dados observados podem ser vistos como uma matriz com elementos X_{ij}, conforme mostra a Tabela 4. A hipótese não-paramétrica de ausência de efeito de grupo pode ser escrita como

$$H_0 : F_1(x) = ... = F_a(x),$$

em que $F_i(x)$ denota uma função de distribuição de probabilidade arbitrária.

Então, sob H_0, essas classificações distribuídas uniformemente, entre 1 e $\sum_{i=1}^{a} n_i$. Agora, defina a estatística de teste

$$F_R = \frac{MST_R}{MSE_R},$$

em que MST_R é o erro quadrático médio devido aos níveis de fator e o MSE_R é a estimativa da variância amostral, também obtida por meio dos postos. Isso é,

$$MST_R = \frac{1}{a-1}\sum_{i=1}^{a}(\bar{R}_{i.} - \tilde{R}_{..})^2 \text{ e}$$

$$MSE_R = \frac{1}{a}\sum_{i=1}^{a}\frac{1}{n_i}S^2_{R,i}.$$

Note que $\bar{R}_{i.} = n_i^{-1}\sum_{j=1}^{n_i} R_{ij}$, sendo R_{ij} o posto médio de cada nível do fator, $\tilde{R}_{..} = a^{-1}\sum_{i=1}^{a}\bar{R}_{i.}$ é a média geral dos níveis de fator e $S^2_{R,i}$ é a variância amostral calculada para cada nível de fator.

Teorema 1 *((Wang & Akritas, 2004) Teste de ausência de efeito de grupo)*

Seja $H_0 : F_1(x) = ... = F_a(x)$ *satisfeita, com* $F_i(x)$ *arbitrária. Se* $n_i \geqslant 2$ *fixo, assumindo as observações independentes, os seguintes limites existem*

$$v_2^2 = \lim_{a\to\infty}\frac{1}{a}\sum_{i=1}^{a}\frac{1}{n_i}\sigma_i^2 > 0$$

$$\tau_2 = \lim_{a\to\infty}\frac{1}{a}\sum_{i=1}^{a}\frac{2\sigma_i^4}{n_i(n_i-1)}$$

.

Então, como $a \to \infty$

$$\sqrt{a}(F_R - 1) \xrightarrow{d} N(0, \tau_2/v_2^4)$$

.

A única suposição requerida por esse teste é que as observações sejam independentes. Vale ressaltar que o erro do tipo I e o poder do teste não serão apresentados, esses já foram estudados por Wang (2004).

2.2 O Teste Contido no PPCLUSTEL(-H)

von Borries (2008) desenvolveu um teste de hipótese que detecta o efeito simples de grupo considerando um grande número de níveis de fator mensurados ao longo do tempo. De forma análoga ao PPCLUST, o PPCLUSTEL usa o p-valor resultante desse teste como medida de dissimilaridade entre grupos de níveis de fator.

Considere X_{ijk} representando o sujeito k aninhado na variável i e medido do tempo j, em que $i = 1, ..., a$, $j = 1, ..., b$ e $k = 1, ..., n_i$, com n_i sendo o número de repetições da variável i. Os dados observados podem ser ilustrados como uma matriz com elementos X_{ijk}, conforme mostra a Tabela 5. A hipótese não-paramétrica de ausência de efeito simples de fator pode ser escrita como

$$H_0(\varphi) : \varphi_{ij} = \alpha_i + \gamma_{ij} = 0, \text{ para todo } i \text{ e } j.$$

Com relação à estrutura de covariância dos dados, assuma que

$$cov(X_{ijk}, X_{i'j'k'}) = \begin{cases} \sigma_{ijj'}, & \text{se } i = i', k = k'. \\ 0, & \text{se } i \neq i', k \neq k'. \end{cases}$$

Considere, ainda, a notação usual utilizada para as estatísticas em análise de variância:

$\bar{X}_{ij.} = \frac{1}{n_i} \sum_{k=1}^{n_i} X_{ijk}$ $\qquad$ $\tilde{X}_{.j.} = \frac{1}{a} \sum_{i=1}^{a} \bar{X}_{ijk}$

$\bar{X}_{...} = \frac{1}{N} \sum_{i=1}^{a} \sum_{j=1}^{b} \sum_{k=1}^{n_i} \bar{X}_{ijk}$ $\qquad$ $\bar{X}_{...} = \frac{1}{ab} \sum_{i=1}^{a} \sum_{j=1}^{b} \bar{X}_{ij.}$.

Agora, defina a estatística da soma dos quadrados médios de tratamento e a da soma dos quadrados médios do erro, respectivamente. De modo que $E(MS\varphi) = E(MSE)$ sob $H_0(\varphi)$.

$$MS\varphi = \frac{1}{(a-1)b} \sum_{i=1}^{a} \sum_{j=1}^{b} (\bar{X}_{ij.} - \tilde{X}_{.j.})^2,$$

$$MSE = \frac{1}{ab} \sum_{i=1}^{a} \sum_{j=1}^{b} \sum_{k=1}^{n_i} \frac{(X_{ijk} - \bar{X}_{ij.})^2}{n_i(n_i - 1)}.$$

Teorema 2 *((von Borries, 2008) Teste de ausência de efeito simples)*

Seja $Y_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \epsilon_{ijk}$, *em que* μ *é a média global;* $\alpha_i (i = 1, ..., a)$ *o efeito médio do fator;* $\beta_j (j = 1, ..., b)$ *o efeito do ponto no tempo;* γ_{ij} *o efeito da interação fator-tempo; e* ϵ_{ijk} *um erro aleatório com distribuição arbitrária* F_{ij} *, para todo* $k = 1, ..., n_i$. *Considere,* $H_0(\varphi) : \varphi_{ij} = \alpha_i + \gamma_{ij} = 0$ *verdadeira. Se as observações* Y_{ijk} *têm momento central finito* $(2 + \delta)(\delta > 0)$, *e o número de repetições é pequeno, com* $n_i \geq 2$ *e limitado, observado para um número fixo* b *de pontos no tempo e com* $a \to \infty$,

$$F_\varphi = \sqrt{ab}(MS\varphi - MSE) \xrightarrow{d} N\left(0, \lim_{a\to\infty} \frac{2}{ab} \sum_{i=1}^{a} \frac{1}{n_i(n_i - 1)} \sum_{j=1}^{b} \sum_{j'=1}^{a} \sigma_{ijj'}^2\right)$$

.

Vale ressaltar que o erro do tipo I e poder do teste não serão apresentados, visto que já foram estudados por von Borries (2008).

3 Algoritmo de Agrupamento por Partição

O objetivo final dos algoritmos de von Borries (2008) é concluir que os grupos (*clusters*) obtidos do conjunto de dados provêm de distribuições diferentes. Os p-valores obtidos aplicando-se cada um dos testes descritos no capítulo anterior são utilizados como medidas de similaridade em um algoritmo de agrupamento baseado em partições com dados de alta dimensão.

A ideia do algoritmo é testar iterativamente se grupos que contêm vários níveis de fator apresentam similaridade, de modo que um grupo é particionado em dois menores quando o teste não-paramétrico de efeito simples é rejeitado. Caso contrário, o grupo permanece intacto. O algoritmo finaliza quando não existir uma similaridade (p-valor) abaixo do limiar informado pelo usuário - nível de significância do teste. O algoritmo que será apresentado a seguir foi desenvolvido e implementado por von Borries (2008).

Seja g o índice do grupo em que o teste está sendo aplicado. D1 contém as observações no grupo g e nf é o número de níveis de fator em D1. O algoritmo PPCLUST por von Borries (2008) é reproduzido abaixo.

1. Seja g = 1, D1 = Dados.
2. Obtenha o Rank dos dados em D1.
3. Calcule a mediana para cada fator em D1.
4. Ordene os fatores em D1 pelas suas medianas.
5. Teste D1.
   - 5.1. Se H_0 não é rejeitada: algoritmo termina.
   - 5.2. Se H_0 é rejeitada: vá para o Passo 6.
6. Retire um subconjunto de D1 e chame de D2.
7. Calcule o número de fatores em D2 e chame de nf.
   - 7.1. Se $nf = 1$:
     - 7.1.1. Aloque fator em D2 para o grupo 0.
     - 7.1.2. Remova os fatores em D2 de D1.
     - 7.1.3. Se nf em D1 = 0, então algoritmo termina.
     - 7.1.4. Se nf em D1 > 0, então faça D2 = D1 e vá para o Passo 8.
8. Teste D2.

8.1. Se H_0 não é rejeitada:

8.1.1. Atribua fatores de D2 para o grupo g.

8.1.2. Faça g = g + 1.

8.1.3. Remova os fatores de D1 em D2.

8.1.4. Se nf em D1 = 0 então algoritmo termina.

8.1.5. Se nf em D1 > 0 então faça:

i. Teste se cada fator em D1 pertence ao novo grupo assinalado. Remova o nível de fator de D1 quando H_0 não é rejeitada e o coloque nesse novo grupo.

ii. Faça D2 = D1 para os níveis de fator remanescentes em D1 e retorne ao Passo 7.

8.2. Se H_0 é rejeitada:

8.2.1. Retire um subconjunto de D2 e chame de D3.

8.2.2. Retorne para D1 todos os níveis de fator que não estão em D3.

8.2.3. Faça D2 = D3 e remova D3.

8.2.4. Retorne ao Passo 7.

Ao se utilizar as medidas de similaridade provenientes dos dois testes que foram descritos no capítulo anterior juntamente com o algoritmo de agrupamento baseado em partições descrito acima, obtém-se o procedimento PPCLUST capaz de agrupar dados superdimensionados com amostras pequenas (HDLSS, *High Dimensional Low Sample Size*). A extensão desse procedimento, PPCLUSTEL, é destinada a agrupar dados longitudinais superdimensionado com amostras pequenas (HDLLSS, *High Dimensional Longitudinal Low Sample Size*).

A escolha do nível de significância dos testes (limiar) fica a cargo do usuário. Quanto menor for esse limiar (próximo de 0), mais conservador o algoritmo será em detectar diferenças entre grupos, e, consequentemente, a tendência é a formação de menos grupos ao final do procedimento. Por outro lado, quanto maior o limiar escolhido (próximo de 1), menos conservador será o algoritmo, levando à formação de um número maior de grupos. O diagrama da Figura 5 (fornecido por von Borries) resume as etapas do algoritmo de forma mais clara.

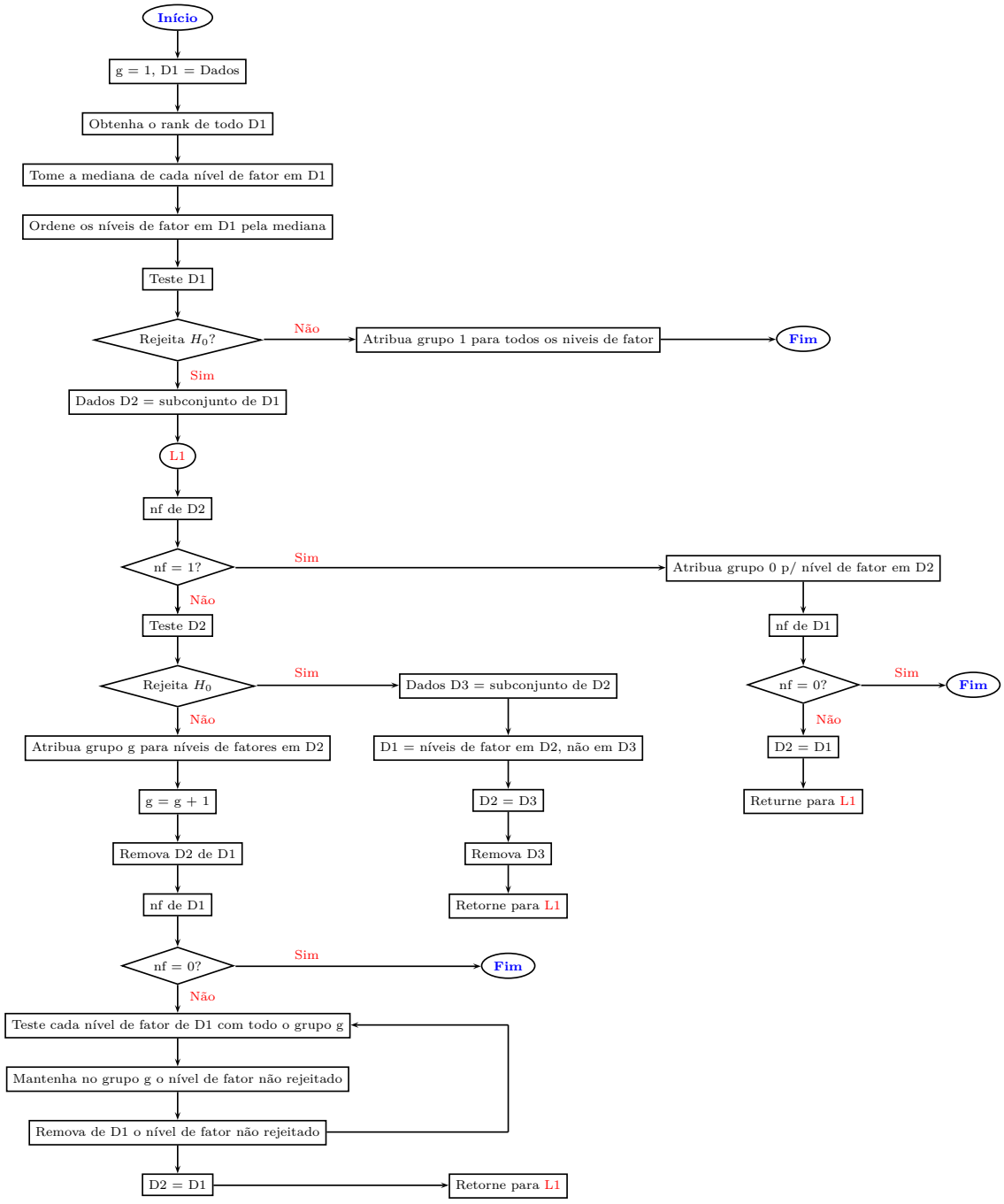


Figura 5 – Diagrama PPCLUST (von Borries, 2008), em que nf significa "número de níveis de fator" e g significa "nome do grupo". O grupo 0 é reservado aos níveis de fator que não foram alocados em nenhum outro grupo criado.

Uma vantagem do método de partição é que ele permite reduzir o número de comparações a serem feitas, o que evita a necessidade de lidar com o problema de comparações múltiplas, conhecido na literatura como taxa de falsos positivos (FDR, *False Discovery Rate*). Outras propriedades dos algoritmos são descritas por von Borries (2009).

A implementação fiel desses algoritmos no **R** encontra algumas limitações. A sub-rotina sortcenter é responsável por rearranjar os níveis de fator centrais (fundo vermelho, Figura 6) que são agrupados primeiro. Para isso o conjunto de dados é reordenado pelo centro utilizando uma variável auxiliar em que o valor 0 indica o "centro" e o valor 1 as extremidades. A partir daí ordena-se o banco por esse vetor.

O SAS não disponibiliza o algoritmo interno usado para a ordenação. Isso impossibilita que essa sub-rotina seja reproduzida fielmente no **R** com os mesmos resultados. Como exemplo considere a aplicação dessa sub-rotina apresentada na Figura 6.

indice	mediana	centro
1	1.0	1
2	1.4	1
3	1.9	1
4	2.6	1
5	2.9	1
6	3.7	0
7	5.1	0
8	6.5	0
9	6.8	0
10	7.2	0
11	8.3	1
12	8.5	1
13	8.8	1
14	9.2	1
15	9.7	1

(a) Dados ordenados pela mediana.

indice	mediana	centro
9	6.8	0
6	3.7	0
7	5.1	0
8	6.5	0
10	7.2	0
1	1.0	1
3	1.9	1
4	2.6	1
5	2.9	1
2	1.4	1
11	8.3	1
12	8.5	1
13	8.8	1
14	9.2	1
15	9.7	1

(b) Dados ordenados pelo centro.

Figura 6 – Resultados pré(a) e pós(b) aplicação da sub-rotina **sortcenter**. Resultados obtidos no SAS 9.4.

O algoritmo interno utilizado pelo SAS para ordenar os dados determina o resultado final da sub-rotina sortcenter[1]. É possível notar que, na prática, exitem várias ordenações possíveis considerando apenas valores 0's e 1's. Logo, diferentes algoritmos de ordenação resultarão em ordenações diferentes da base de dados. Não obstante, o SAS não revela o seu algoritmo responsável pela ordenação. A solução para a implementação dessa sub-rotina no **R** foi utilizar a ordenação pelo centro e pela mediana conjuntamente, assim, o seu resultado se tornou replicável. Feito esse ajuste nas implementações dos algoritmos em ambos os *softwares*, realizou-se 50 simulações para verificar a concordância dos resultados entre as implementações dos dois *softwares*.

[1] sortndx call

Algoritmo	ARI	
	Média	D.P.
PPCLUST	0,991	0,003
PPCLUSTEL	0,707	0,079

Tabela 6 – Resumo do Índice Rand Ajustado (ARI) entre os agrupamentos das implementações do SAS e do R.

Verifica-se a partir da Tabela 6 que o agrupamentos obtidos pelo PPCLUST no **R** são quase idênticos ao obtidos pela implementação original, no SAS. No entanto, o mesmo não acontece para o PPCLUSTEL. Isso pode ser explicado pela diferença de precisão do cálculos entre as duas linguagens. Essa diferença irá influenciar diretamente no resultado do agrupamento. Para mostrar que isso ocorre computamos o resultado de $\sqrt{3}$ em ambas as linguagens. No **R** obtém-se 1,7320508075688**72** enquanto no SAS 1,7320508075688**00**. Além disso, o **R** não consegue representar fielmente o resultado do SAS; ao tentar armazenar esse valor obtém-se 1,7320508075688**01**. Outras divergências aparentemente insignificantes como essa podem incidir no resultado final do agrupamento, visto que o processo de agrupamento envolve milhares de operações.

Daqui em diante apresentaremos um algoritmo alternativo. A remodelagem do algoritmo visa obter um procedimento mais rápido, aproveitando os benefícios do processamento paralelo, e promissor, com relação a qualidade de agrupamento. A esse algoritmo será dado o sufixo H (do inglês, *Hybrid*).

O oitavo passo do PPCLUST é o mais custoso computacionalmente. Neste passo cada nível de fator restante no conjunto de dados é testado com o grupo recém-formado. Se esse nível de fator não é rejeitado o mesmo passa a fazer parte do grupo, caso contrário, permanece de fora e assim sucessivamente para os níveis de fator restantes. Há uma relação de dependência entre os sucessivos testes entre um grupo formado e um dado nível de fator. Isso torna inviável o uso de processamento paralelo para o PPCLUST.

Seja g o índice do grupo em que o teste está sendo aplicado, D1 contém as observações no grupo g, nf é o número de níveis de fator em D1, np o número de subconjuntos e gt uma variável indicadora de grupo temporário. O algoritmo PPCLUST-H é descrito abaixo.

1. Seja g = 1, D1 = Dados, limiar = α.
2. Obtenha o Rank dos dados em D1.
3. Teste D1.
   - 3.1. Se H_0 é não rejeitada: algoritmo termina.
   - 3.2. Se H_0 é rejeitada: continue.
4. Faça $i = 1$, $np = 2$ e $gt = 0$.

5. Faça np subconjuntos de D1.
6. Enquanto $i <= np$:
   6.1. Teste o subconjunto i.
      6.1.1. Se H_0 é não rejeitada: atribua um grupo temporário aos níveis de fator do subconjunto i e remova-os de D1.
      6.1.2. Faça $gt = 1$.
      6.1.3. Se H_0 é rejeitada: continue.
   6.2. Faça $i = i + 1$.
7. Se nf em D1 > 0, então continue, caso contrário, vá para o passo 9.
8. Se $gt = 0$, então faça $np =$ próximo número da sequência de Fibonacci, caso contrário, faça $np = 2$.
   8.1. Retorne para o passo 5.
9. Enquanto $mp =$ máx p-valor entre dois grupos temporários $>$ limiar:
   9.1. Faça os dois grupos temporários como um grupo único.

Seguindo o mesmo principio do *bootstrap* exemplificado no Capítulo 1, no PPCLUST-H (e sua extensão PPCLUSTEL-H) o processamento paralelo atua no cálculo de milhares de estatísticas independentes, número proporcional a quantidade de níveis de fator (variáveis), que participam como insumo no cálculo dos p-valores nos passos subsequentes.

No quinto passo do algoritmo os subconjuntos são criados de tal forma que sejam homogêneos interiormente e heterogêneos entre si. A ideia por trás disso é reduzir a quantidade de testes consideravelmente durante o processo de agrupamento, diminuindo ainda mais o tempo de execução.

A sequência de Fibonacci[2] é uma série de números inteiros em que cada elemento subsequente é a soma dos dois anteriores. A característica de interesse nessa sequência é o seu crescimento rápido: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55 e assim por diante.

Em agrupamento de dados não há conhecimento prévio do número de grupos (*clusters*) ou ainda, se esse número é pequeno ou grande. Nesse sentido, a sequência de Fibonacci entra no algoritmo PPCLUST-H para determinar o número de subconjuntos

[2] A sequência de Fibonacci foi descoberta pelo matemático italiano Leonardo de Pisa (1170-1250). Essa é uma série de números bastante famosa e esconde em si a proporção áurea, ou número de ouro, uma constante bastante conhecida por estar implícita em muitas circunstâncias na natureza como no formato das ondas, dos furações, na geometria das flores e até nas medidas do corpo humano. Uma das principais aplicações dos algoritmos implementados neste trabalho é o agrupamento de expressão gênica, algo essencialmente natural.

(grupos em potencial) a serem formados dos dados. Isso possibilita que o algoritmo tenha mais versatilidade e escalabilidade.

O último passo do algoritmo consiste em verificar iterativamente a existência de dois grupos temporários que tenham p-valor acima do limiar. O algoritmo termina quando todos os p-valores de dois a dois grupos estão abaixo do limiar.

O diagrama da Figura 7 ilustra as etapas do algoritmo de forma reduzida.

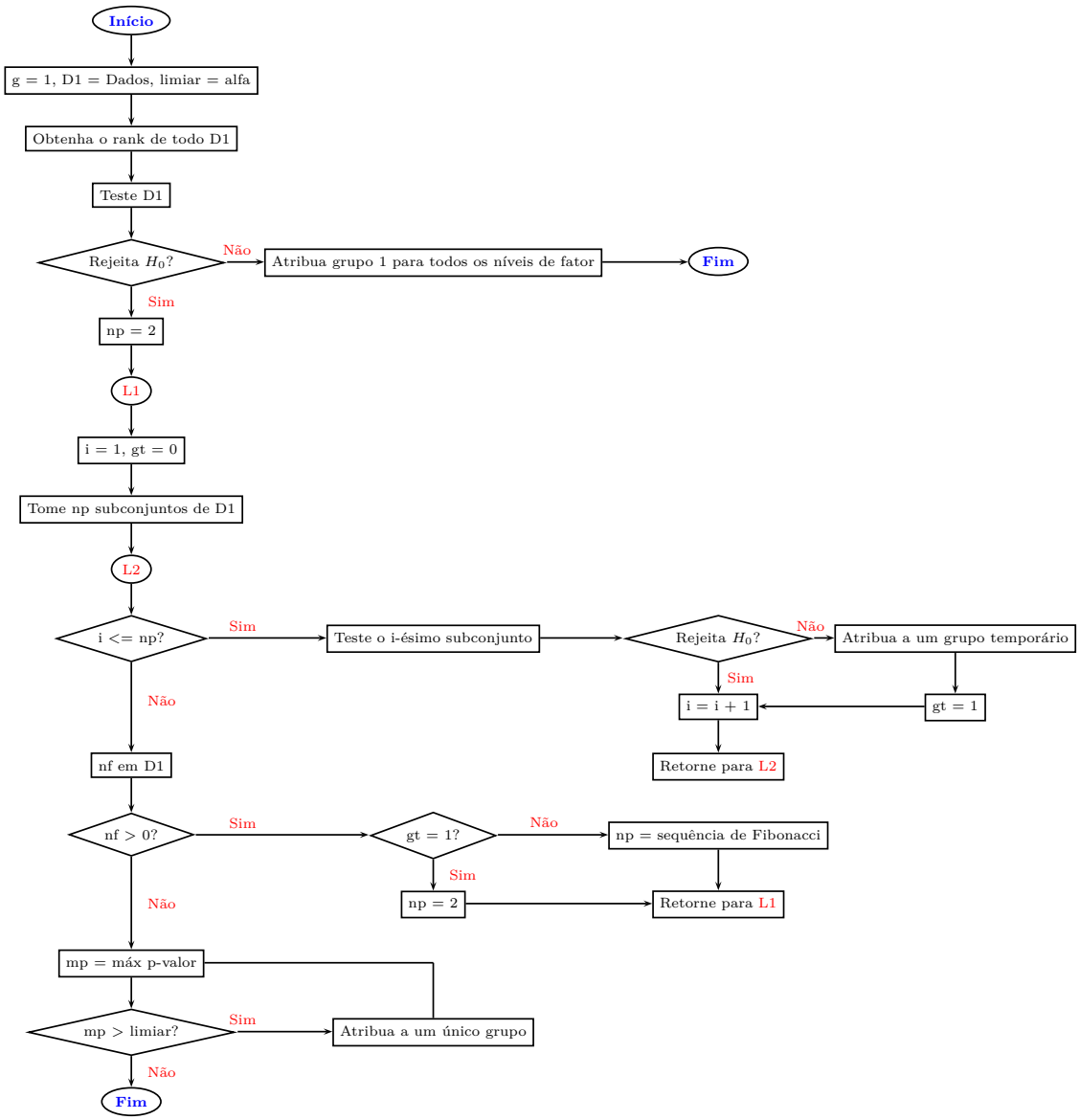


Figura 7 – Diagrama do PPCLUST-H.

4 Aplicação do Pacote R

O pacote elaborado neste trabalho, Rclust, contém os algoritmos de von Borries (2008) e suas respectivas versões alternativas apresentadas no Capítulo 3. Esses algoritmos estão disponíveis por meio das seguintes funções:

ppclust(dataset, alpha)
ppclust_h(dataset, alpha, n.cores = 1)
ppclustel(dataset_long, id, repid, alpha)
ppclustel_h(dataset_long, id, repid, alpha, n.cores = 1)

dataset(_long): matriz ou data.frame contento os dados. Os formatos dos diferentes conjuntos de dados são ilustrados abaixo.

```
> head(dataset)
    obs1     obs2     obs3     obs4     obs5
 -0.4796  -0.0997  -0.4906  -0.5081  -0.8321
 -0.4015  -0.6497  -0.6648  -0.9588  -0.4390
 -0.5938  -0.5910  -0.4898  -0.5960  -0.2735
 -0.3542  -0.7884  -0.6466  -0.4068  -0.6805
 -0.1359  -0.8393  -0.9238  -0.2069  -1.0352
 -0.6429  -0.0768  -0.6982  -0.2880  -0.7263

> head(dataset_long)
sujeito repeticao    t1       t2       t3       t4       t5
 1        1         1.0815   0.9860  -0.9490  -1.3401  -3.2225
 1        2         1.3936  -0.0216  -2.1741  -3.3784  -3.9885
 1        3         0.6245  -0.5166  -2.1773  -2.5477  -2.9283
 2        1         1.5829  -0.1851  -2.2661  -2.1858  -3.5667
 2        2         2.4562   0.4127  -2.4133  -1.8472  -4.1041
 2        3         0.4283   0.5038  -1.8206  -1.5484  -3.1500
```

id: o número da coluna que contém o identificador do objeto.
repid: o número da coluna que identifica a repetição.
alpha: limiar do algoritmo.
n.cores: número de núcleos do processador para o agrupamento (n.cores = 1 por padrão).

4.1 Resultados em Dados Simulados

Estudos de simulação foram realizados para comparar a performance dos métodos em termos da qualidade de agrupamento e do tempo de processamento. Para tanto, foi utilizado o índice ARI como a métrica da qualidade do agrupamento, cujo uso foi possibilitado por meio da função AdjustedRandIndex(), disponível no pacote MCLUST.

O MCLUST não foi projetado para agrupar dados com valores de entrada X_{ijk}, como o disposto na Tabela 5 (dados HDLLSS). Caso isso ocorra, o algoritmo não irá distinguir as repetições de um mesmo nível de fator, pois interpreta cada linha da base de dados como sendo uma unidade amostral e cada coluna como uma variável. Dessa forma, para adequar a estrutura X_{ijk} ao MCLUST, utilizou-se as médias $X_{ij.}$ como os valores de entrada do algoritmo. Esse formato é ilustrado abaixo.

```
> head(dataset_long_mclust)
sujeito    t1      t2      t3      t4      t5
1       1.0332  0.1492 -1.7668 -2.4221 -3.3798
2       1.4891  0.2437 -2.1666 -1.8605 -3.6070
3       1.2362 -0.4317 -2.2985 -2.3904 -3.4862
4       1.0666  0.3927 -1.0534 -1.2141 -2.3495
5       0.2362 -0.6645 -1.8171 -2.6169 -3.2470
6       0.0214 -0.6650 -2.7392 -1.9679 -3.2201
```

É importante destacar que, em todas as simulações, o número real de grupos foi informado ao MCLUST, o que ofereceu uma vantagem tanto em relação à qualidade de agrupamento quanto ao tempo processamento.

Inicialmente, todos os conjuntos de dados simulados foram gravados. Em virtude da grande quantidade de dados a serem gerados optou-se, por conveniência, pelo uso do SAS para essa tarefa. Em seguida, os dados foram lidos no **R** para dar sequência à realização dos estudos de simulação. A máquina utilizada tem a seguinte configuração: processador Intel Core i5, CPU @ 3.25 GHz, 8GB de memória RAM e plataforma Windows 10 com arquitetura 64 bits.

4.1.1 Agrupamento de Dados HDLSS

Qualidade de Agrupamento

A base de dados que simula a estrutura HDLSS foi construída levando-se em consideração as seguinte situações: 2000 e 6000 níveis de fator; 5, 10 e 15 observações. As variáveis foram divididas em 5 grupos que diferem apenas em relação à sua média. Além disso, outros grupos foram obtidos a partir da transformação dos grupos inicias para a distribuição log-normal. Para comparar a estabilidade dos algoritmos, foram gerados 50

replicações dos conjunto de dados. Para $X \sim \mathcal{N}(0,1)$ uma variável aleatória normalmente distribuída com média 0 e variância 1, gerou-se dados conforme os grupos descritos abaixo.

Grupo 1: 20% de níveis de fator com distribuição $0,25 \times X - 0,5$;

Grupo 2: 20% de níveis de fator com distribuição $0,25 \times X - 0,2$;

Grupo 3: 20% de níveis de fator com distribuição $0,25 \times X + 0$;

Grupo 4: 20% de níveis de fator com distribuição $0,25 \times X + 0,5$;

Grupo 5: 20% de níveis de fator com distribuição $0,25 \times X + 1$.

A Figura 8 ilustra os dados gerados com os 5 grupos (gráfico esquerdo) e os dados transformados para distribuição log-normal (gráfico direito) com o respectivos grupos.

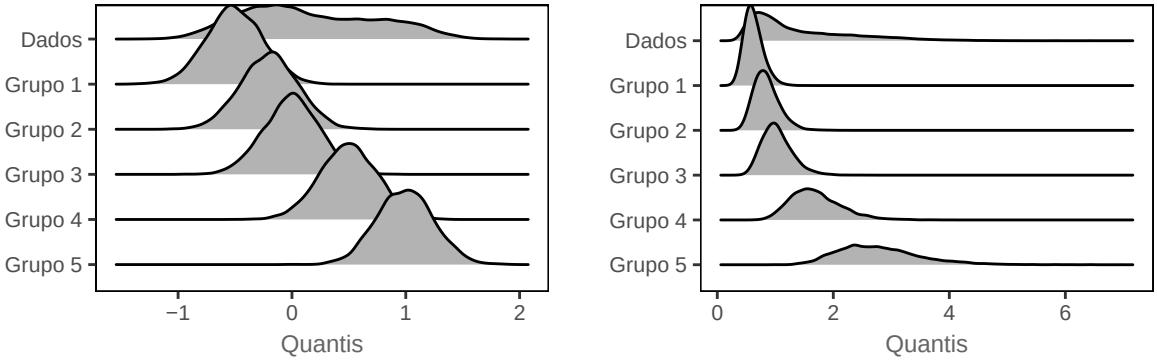


Figura 8 – Os 5 grupos gerados considerando as distribuições: normal e log-normal, nessa ordem.

A Tabela 7 apresenta as médias do índice ARI obtidos das simulações considerando: 2000 e 6000 níveis de fator para 5, 10 e 15 observações. Vale lembrar que esse índice foi calculado comparando o agrupamento obtido com a real estrutura simulada em cada um das 50 simulações realizadas.

Parâmetros		PPCLUST-H		MCLUST			
		Normal/LogN		Normal		LogN	
a	n_i	Média	D.P.	Média	D.P.	Média	D.P.
2000	5	0,741	0,013	0,746	0,014	0,691	0,014
2000	10	0,861	0,014	0,877	0,009	0,832	0,011
2000	15	0,913	0,016	0,933	0,009	0,896	0,009
6000	5	0,743	0,007	0,745	0,009	0,694	0,009
6000	10	0,863	0.006	0,879	0,004	0,833	0,006
6000	15	0,912	0,013	0,932	0,003	0,896	0,005

Tabela 7 – Média do índices de Rand ajustado (ARI) em 50 replicações dos algoritmos PPCLUST-H e MCLUST com o uso dos dados normais e log-normais. Estudo verificado com base em 50 conjuntos de dados gerados com 2000 e 6000 níveis de fator para 5, 10 e 15 observações.

A partir da Tabela 7 nota-se que a taxa de acerto do PPCLUST-H tende a crescer à medida que a quantidade de níveis de fator e de observações aumenta. Isso se deve ao fato de que essas duas quantidades fornecem mais informação para o agrupamento. De maneira geral, os algoritmos identificaram corretamente os grupos gerados.

Um ponto a se destacar foi que MCLUST não agrupou os dados log-normais com a mesma qualidade. Como vemos na Tabela 7, a taxa de acerto desse procedimento para os dados log-normais diminuiu. Tal queda fez com que o MCLUST ficasse abaixo do PPCLUST-H em termos da taxa de acerto. Isso se deve à suposição do MCLUST de que os dados são provenientes de uma mistura de distribuições normais, essencialmente simétricas. Isso ressalta a vantagem do algoritmo PPCLUST-H devido à sua propriedade de invariância a transformações monótonas tais como a logarítmica e exponencial. Além disso, esse algoritmo se destaca pela sua generalidade, podendo agrupar dados HDLSS sem a necessidade de suposição sobre a distribuição dos dados.

Percebe-se uma tendência geral de crescimento da qualidade de agrupamento à medida que a quantidade de níveis de fator e/ou de observações aumentam. Isso ocorre principalmente porque o incremento de ambos os parâmetros fornece informação útil para o agrupamento dos dados (Tabela 7). Os resultados entre os dois procedimentos são mais competitivos quando consideramos a razão entre a quantidade de observações e a quantidade de níveis de fator sendo ínfima. A Figura 9 apresenta os *boxplot's* para os ARI's obtidos considerando uma situação semelhante, com conjuntos de dados de 2000 e 6000 níveis de fator com apenas 5 observações.

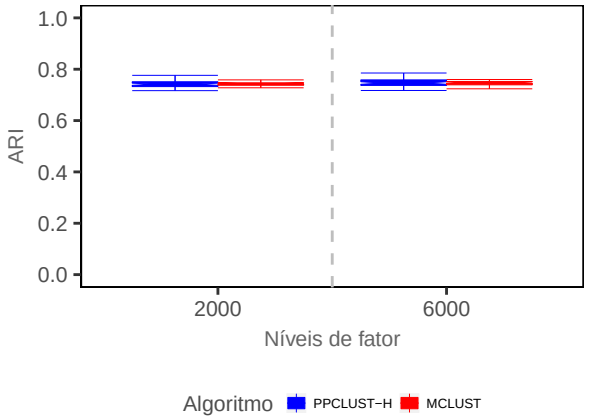


Figura 9 – Índices de Rand ajustados (ARI) para os algoritmos PPCLUST-H e MCLUST aplicados em 50 conjuntos de dados gerados com 2000 e 6000 níveis de fator para 5 observações.

Os resultados para esse cenário mostraram-se bem próximos para ambos os procedimentos. Segundo von Borries (2009), devido ao teste (Seção 2.1), a qualidade de agrupamento é ainda maior considerando conjuntos de dados com mais de 15,000 níveis de fator e 3 observações. Dessa forma, os cenários considerados nas simulações não

possibilitaram que o PPCLUST-H atingisse um desempenho ainda melhor.

Tempo de Processamento

O tempo de processamento é um quesito importante no agrupamento de dados superdimensionados. A Tabela 8 mostra o tempos de agrupamento para os dois algoritmos sendo o PPCLUST-H aplicado para um e dois núcleos do processador.

Algoritmo	Níveis de fator					
	2000		6000		10000	
	Média	D.P.	Média	D.P.	Média	D.P.
PPCLUST-H	0,7s	0,02s	1,5s	0,03s	2,3s	0,05s
PPCLUST-H+	0,8s	0,02s	1,3s	0,02s	1,9s	0,06s
MCLUST	6,4s	0,7s	11s	1,5s	16s	2,7s

Tabela 8 – Resumo dos tempos de processamento dos algoritmos em 50 replicações. Estudo verificado em uma base de dados com 2000, 6000 e 10000 níveis de fator; 10 observações. O símbolo + indica o uso de dois núcleos do processador.

Como pode-se perceber o PPCLUST-H apresentou os melhores tempos de agrupamento. Os incrementos de velocidade utilizando dois núcleos foram -14,2%, 13,3% e 17,4%. Isso mostra que os benefícios do processamento paralelo mostraram-se promissores uma quantidade maior de níveis de fator. Isso permite que o PPCLUST-H apresente uma característica importante no contexto de agrupamento de dados superdimensionados - a escalabilidade. Vale pontuar que o processamento paralelo traz consigo um custo computacional para que determinadas tarefas sejam distribuídas para os núcleos e os resultados computados. Apesar da vantagem dada ao MCLUST, que fornece o número de grupos dos dados (algo que não acontece no agrupamento de dados reais), este não foi tão rápido quanto o PPCLUST-H.

4.1.2 Agrupamento de Dados HDLLSS

Qualidade de Agrupamento

Nesta subseção, o desempenho dos algoritmos PPCLUSTEL-H e MCLUST é comparado no agrupamento de dados longitudinais superdimensionados gerados. A base de dados que simula a estrutura HDLLSS foi construída levando-se em consideração as seguinte situações: 2000 e 6000 níveis de fator; 5, 10 e 15 pontos no tempo; 3 repetições. Os grupos foram gerados a partir de distribuições normais multivariadas. Além disso, esses grupos foram utilizados para dar origem a outros por meio da transformação para a distribuição log-normal multivariada. Com o intuito de avaliar a estabilidade dos algoritmos, foram geradas 50 replicações dos conjuntos de dados descritos a seguir.

Grupo 1: 25% de curvas com média $\mu_j^1 = min\Big(\Big(\frac{2-5j}{2}\Big), \Big[\frac{5j-2^2}{3} + sen\Big(\frac{5\pi j}{2}\Big)\Big]\Big)$;

Grupo 2: 25% de curvas com média $\mu_j^2 = \mu_j^1$;

Grupo 3: 25% de curvas com média $\mu_j^3 = cos(2\pi j)$;

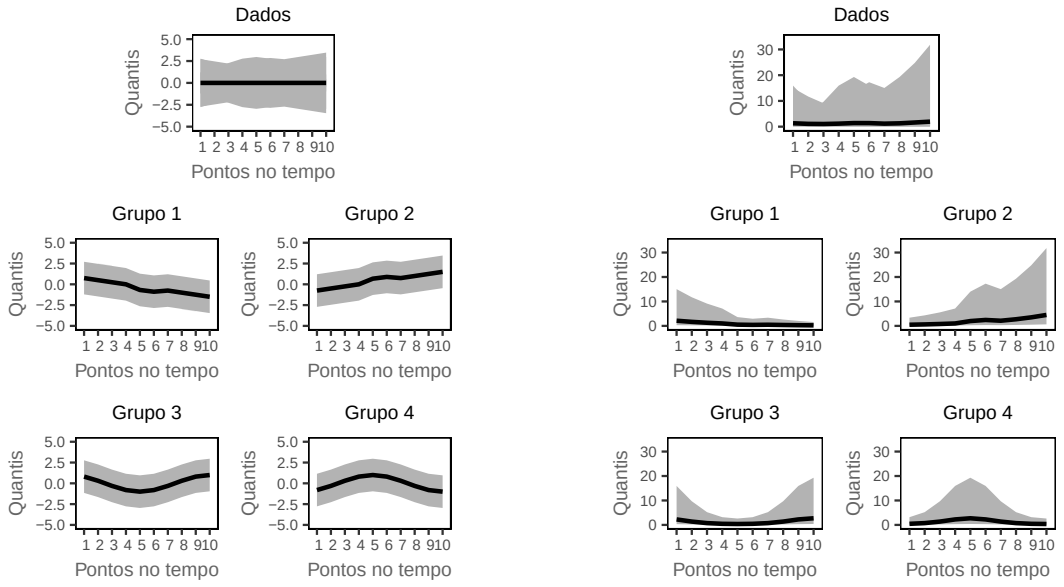
Grupo 4: 25% de curvas com média $\mu_j^4 = -\mu_j^4$.

Estes quatro grupos também foram usados por Serban and Wasserman (2005). Contudo, esses autores consideraram observações independentes no tempo, ou seja, $\sum_{b\times b} = I$, em que I é a matriz identidade e b, o número de pontos no tempo. Neste trabalho, será considerado o caso correlacionado. Seguindo a estrutura de covariância utilizada nas simulações dos trabalhos de Fan and Zhang (2000) e Wu and Chiang (2000), temos:

$$\sum_{b\times b} = cov(X_{ijk}, X_{i'j'k'}) = \begin{cases} \sigma_j \sigma_{j'} e^{-|j-j'|/b}, & \text{se } i = i', k = k'. \\ 0, & \text{se } i \neq i', k \neq k'. \end{cases}$$

em que b é o número de pontos no tempo do experimento e $\sigma_j = 1$ escolhido. De acordo com essa estrutura de covariância, a correlação entre pontos sucessivos é alta e decresce à medida que o intervalo de tempo aumenta.

A Figura 10 ilustra os dados com as misturas juntas e os 4 grupos gerados separadamente (painel esquerdo); no painel direito os dados e respectivos 4 grupos transformados para a distribuição log-normal multivariada.



(a) Grupos originais. (b) Grupos após a transformação.

Figura 10 – Média, 5^o e 95^o percentil para os dados agregados e para os 4 grupos simulados considerando as distribuições: normal multivariada (a) e log-normal multivariada (b), com 10 pontos no tempo.

Uma característica interessante desses grupos é a semelhança com o comportamento de processos biológicos. Por exemplo, na análise de dados de microarranjo longitudinais em que estamos interessados em agrupar genes que mostram perfis semelhantes tais como: monótono crescente (decrescente), ascendente (descendente) ou cíclico.

A Tabela 9 apresenta a média do índice de Rand ajustado (ARI) calculado para os procedimentos PPCLUSTEL-H e MCLUST aplicados em 50 conjuntos de dados simulados com 2000 e 6000 níveis de fator; 5, 10 e 15 pontos no tempo para 3 repetições.

Parâmetros		PPCLUSTEL-H		MCLUST	
a	b	Média	D.P.	Média	D.P.
2000	5	0,572	0,056	0,994	0,002
2000	10	0,757	0,092	0,998	0,001
2000	15	0,762	0,130	0,998	0,001
6000	5	0,612	0,062	0,994	0,001
6000	10	0,793	0,109	0,998	0,001
6000	15	0,844	0,086	0,998	0,001

Tabela 9 – Média dos índices de Rand ajustados (ARI) para os algoritmos PPCLUSTEL-H e MCLUST com o uso dos dados normais. Estudo verificado com base em 50 conjunto de dados gerados com 2000 e 6000 variáveis; 5, 10 e 15 pontos no tempo para 3 repetições.

Um ponto interessante a se observar na Tabela 9 é que o PPCLUSTEL-H se beneficia da alta dimensionalidade dos dados. A taxa de acerto do PPCLUSTEL-H aumentou na medida em que a quantidade de níveis de fator e de pontos no tempo cresceram. De forma análoga ao estudo de simulação para PPCLUSTEL-H os cenários aqui considerados não foram os mais favoráveis, visto que o teste utilizado pelo algoritmo requer teoricamente um número de níveis de fator infinitamente grande. Nesse sentido, considerando o cenário de dados com 15000 níveis de fator, 10 pontos no tempo e três repetições foram gerados 50 conjuntos. A Figura 11 apresenta os *boxplot's* do índice Rand ajustado para os bases de dados geradas.

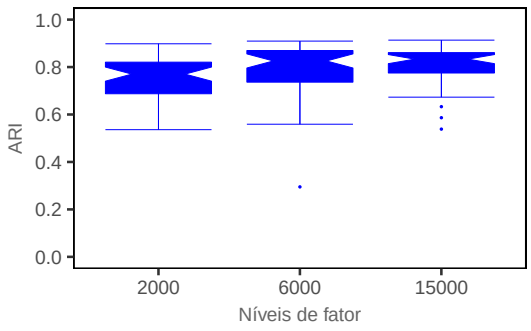


Figura 11 – Índices de Rand ajustados (ARI) para o algoritmo PPCLUSTEL-H aplicado em 50 conjuntos de dados gerados com 2000, 6000 e 15000 níveis de fator; 10 pontos no tempo para 3 repetições.

Na Figura 11 nota-se uma tendência crescente do ARI obtido pelo PPCLUSTEL-H na medida em que o número de níveis de fator aumenta. Nesse contexto, o algoritmo não evidenciou todo o seu potencial no agrupamento dos dados. Vale ressaltar que para um número maior de repetições também há o mesmo efeito crescente do ARI.

Embora o PPCLUSTEL-H não tenha apresentado uma qualidade de agrupamento tão refinada quanto a do MCLUST, esse cumpre o seu papel na detecção de grupos estatisticamente relevantes em um conjunto de dados HDLLSS. Uma maneira de aproveitar essa capacidade é o uso do PPCLUSTEL-H como ferramenta para determinar o número real de grupos. Feito isso, a aplicação do MCLUST tomando esse número como o de componentes da mistura de distribuições.

O teste presente no PPCLUSTEL-H é baseado nos dados originais, com isso, a qualidade de agrupamento pode deixar a desejar quando os dados apresentarem grande variabilidade no tempo. Essa situação pode ser contornada considerando os dados em postos em vez dos originais. A demonstração de convergência da estatística F do teste para dados em postos não está formalizada na literatura. Ainda assim, Silva (2012) conclui que o teste baseado em postos apresenta a mesma convergência que o baseado nas observações originais a partir da convergência obtida em diferentes cenários de simulação. Com base nisso, calculamos o índice ARI dos agrupamento obtidos pelos procedimentos PPCLUSTEL-H e MCLUST aplicados em 50 conjuntos de dados log-normais.

Parâmetros		PPCLUSTEL-H		MCLUST	
a	b	Média	D.P.	Média	D.P.
2000	5	0,593	0,068	0,503	0,068
2000	10	0,739	0,114	0,485	0,051
2000	15	0,747	0,122	0,477	0,055
6000	5	0,651	0,081	0,513	0,064
6000	10	0,805	0,099	0,484	0,054
6000	15	0,807	0,133	0,480	0,053

Tabela 10 – Média dos índices de Rand ajustados (ARI) para os algoritmos PPCLUSTEL-H e MCLUST com o uso dos dados em postos e log-normais, respectivamente. Estudo verificado com base em 50 conjunto de dados gerados com 2000 e 6000 níveis de fator; 5, 10 e 15 pontos no tempo para 3 repetições.

Em aplicações em que há um indício que o dados HDLLSS são normalmente distribuídos, em termos de qualidade de agrupamento, o PPCLUSTEL-H com a utilização dos dados em postos se sobressai frente ao MCLUST. Observando a Tabela 10 vemos que o PPCLUSTEL-H apresentou resultados significativamente melhores que o MCLUST. Com isso o uso dos dados em postos é uma alternativa para que o PPCLUSTEL-H tenha um desempenho estabilizado em dados tal como os log-normais, visto que o uso dos dados em postos pode agregar mais robustez ao teste presente nos algoritmo.

Tempo de Processamento

A Tabela 11 apresenta o tempo que os algoritmos levam para realizar o agrupamento em uma base de dados com 2000, 6000 e 10000 níveis de fator dispostas em 10 pontos no tempo com 3 repetições.

Algoritmo	Níveis de fator					
	2000		6000		10000	
	Média	D.P.	Média	D.P.	Média	D.P.
PPCLUSTEL-H	1,9s	0,1s	5s	0,2s	8,3s	0,2s
PPCLUSTEL-H+	1,9s	0,1s	4,5s	0,2s	7,4s	0,7s
MCLUST	3,1s	0,1s	3,7s	0,7s	4,2s	1,5s

Tabela 11 – Resumo dos tempos de processamento dos algoritmos em 50 replicações de bases de dados com 2000, 6000 e 10000 níveis de fator (variáveis) com 10 pontos no tempo e 3 repetições. O símbolo + indica o uso de dois núcleos do processador.

Os resultados do PPCLUSTEL-H e MCLUST mostraram-se competitivos. No tocante ao PPCLUSTEL-H, percebe-se que o processamento paralelo agregou velocidade no agrupamento, os incrementos de velocidade utilizando dois núcleos foram 0%, 10% e 0,11%. Esses resultados sugerem que o PPCLUSTEL-H tem flexibilidade para agrupar conjuntos de dados maiores de forma eficiente. Vale ressaltar a vantagem dada ao MCLUST fornecendo o número de grupos dos dados (algo que não acontece no agrupamento de dados reais), caso contrário, o MCLUST teria de computar uma série de modelos considerando várias quantidades de componentes (grupos) o que tomaria um tempo bem maior e um custo computacional mais elevado.

4.2 Resultados em Dados Reais

4.2.1 Análise de Microarranjo

A medição da expressão gênica fornece uma visão de como um organismo responde a uma determinada circunstância como doença, infecção, lesão ou tratamento. O desenvolvimento da tecnologia de microarranjo permitiu que a expressão de milhares de genes fossem medidas simultaneamente.

Um gene é um segmento específico de uma molécula de ácido desoxirribonucleico (DNA). O DNA humano se encontra no núcleo de cada célula biológica e contém cerca de 25000 genes. Essa molécula serve como molde para gerar uma outra molécula chamada ácido ribonucleico (RNA) que, por sua vez, tem a função de expressar o código genético do DNA fora do núcleo celular criando uma sequencia específica de aminoácidos que serão ligados para formar uma proteína.

Isso nos remete a um conceito chamado Dogma Central da Biologia Molecular, representado na Figura 12. Esse dogma diz que a informação genética passa do DNA para a proteína em um caminho de informação unidirecional, e, de forma simplificada compreende três processos: replicação, transcrição e translação.

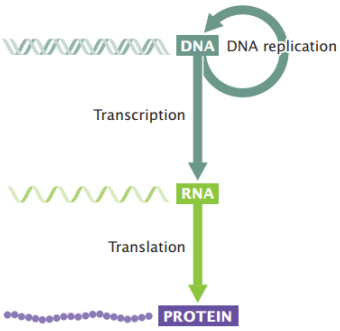


Figura 12 – Dogma Central da Biologia Molecular (retirado de Pierce (2012)).

Na replicação, a informação passa de uma molécula de DNA para outras moléculas de DNA. Esse processo possibilita que a divisão celular dê origem a outras células com exatamente o mesmo material genético. No caso de uma célula já cancerígena o resultado disso pode ser a divisão desregulada e a proliferação de células defeituosas, uma das marcas registradas do câncer.

O processo de transcrição consiste basicamente na transferência de instruções genéticas do DNA para uma molécula de RNA. Na tradução, a informação passa do RNA para a proteína. O RNA transfere a informação genética para uma proteína com uma sequência de aminoácidos específica.

Os genes tem uma relação direta com a função de uma célula. Dessa forma, iterações atípicas de expressões gênicas podem dar origem a sequências impróprias de aminoácidos que por sua vez produzirão proteínas de formato irregular. Isso pode comprometer a função de uma célula de forma a dar origem a doenças como o câncer.

Com o intuito de entender os mecanismos biológicos de diversos organismos em determinadas circunstâncias, uma aplicação que tem tido bastante destaque é a análise de dados de perfil de expressão gênica, ou, simplesmente, análise de microarranjo. Os principais objetivos dessa análise envolve o agrupamento e a classificação de genes, individuais ou em grupos, de acordo com a sua expressão.

A análise de microarranjo é largamente utilizada em experimentos de genômica funcional projetados para estudar as funções e interações dos genes dentro do contexto global do genoma de diversas espécies animais e vegetais. Por exemplo, determinação do ciclo celular de fungos (Gillespie et al., 2010) e a identificação dos genes causadores da leucemia aguda (Yu et al., 2017), linfoma (Alizadeh et al., 2000), câncer de mama (Perou et al., 1999), entre outros.

Pierce (2012) apresenta um exemplo prático da análise de microarranjo. Nesse exemplo foram identificados setenta genes cujos padrões de expressão previram com precisão a recorrência de câncer de mama dentro de 5 anos de tratamento. Cada pequeno quadrado da Figura 15 representa a expressão de um gene no tumor de um paciente em comparação com a expressão desse gene em suas células não cancerígenas. A cor verde indica que o gene se expressou mais em células não cancerígenas do que em células cancerígenas, já a cor vermelha é exatamente o oposto. De uma outras forma, considere a Figura 13 como uma matriz composta de valores numéricos que estão associados e que as linhas representam os pacientes e as colunas os genes. Os tumores acima da linha amarela vieram principalmente de pacientes que permaneceram livres do câncer por pelo menos 5 anos e os tumores abaixo da linha amarela sólida vieram principalmente de pacientes nos quais o câncer se espalhou dentro de 5 anos após o diagnóstico.

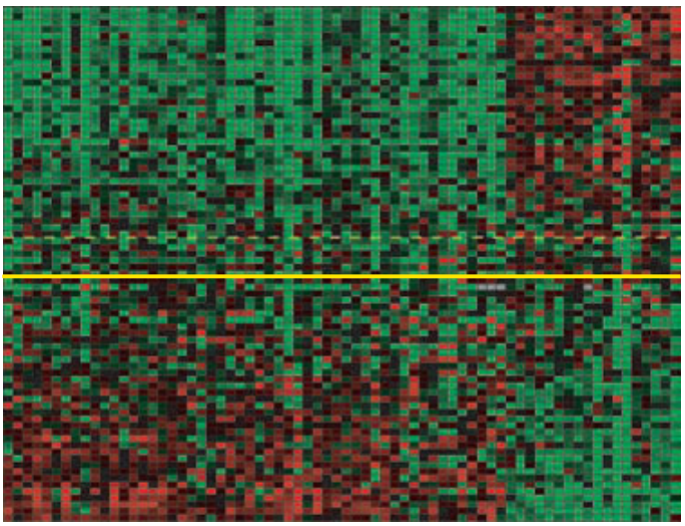


Figura 13 – Expressão gênica de 70 genes que apresentaram padrões de expressão para a detecção da recorrência de câncer de mama dentro de 5 anos de tratamento.

Nota-se o contraste entre os níveis de expressão desses genes acima e abaixo da linha amarela. Identificar padrões de expressão como esse é o propósito final dos procedimentos PPCLUST(-H).

Os dados de microarranjo podem se dividir em: não longitudinais, em que as expressões gênicas são observadas um ponto fixo no tempo que tem como foco permitir a detecção de padrões de genes diferentemente expressos de um caso controle; e os longitudinais que possibilitam estudar os padrões de expressão gênica através de uma série de pontos no tempo.

Em experimentos de microarranjo longitudinais a expressão gênica é monitorada ao longo do tempo e o interesse é agrupar genes que mostram perfis temporais semelhantes. Isso ocorre porque as informações ao longo do tempo fornecem evidências sobre os mecanismos dos processos biológicos observados.

Os dados de expressão gênica são organizados em uma matriz onde cada linha corresponde a um gene e cada coluna a uma condição. As condições podem ser representadas por exemplo por pontos no tempo ou tecidos. O tamanho e o volume dos conjuntos de dados genômicos são suficientes para apresentar aos algoritmos de agrupamento muitos problemas de alocação de memória e velocidade de agrupamento.

4.2.2 Expressão Gênica de Câncer de Cólon

A variação da expressão gênica das células do intestino pode ajudar no desenvolvimento de métodos para o diagnóstico de câncer. Nesta subseção, o PPCLUST-H é aplicado a um conjunto de expressão gênica pré-processado por Corrada Bravo et al. (2012). Os autores realizam um estudo sobre o perfil de expressão gênica em alguns dos principais tipos de doença no intestino grosso (cólon), dentre elas o carcinoma colorretal. O dados pré-processados contêm a expressão gênica de 5339 genes das biópsias de 15 pacientes com adenoma, 15 pacientes com câncer de cólon e de 8 pacientes saudáveis (controles). Esses dados podem ser obtidos no *Bioconductor* por meio do download do pacote **R** chamado antiProfilesData com identificador GSE4183.

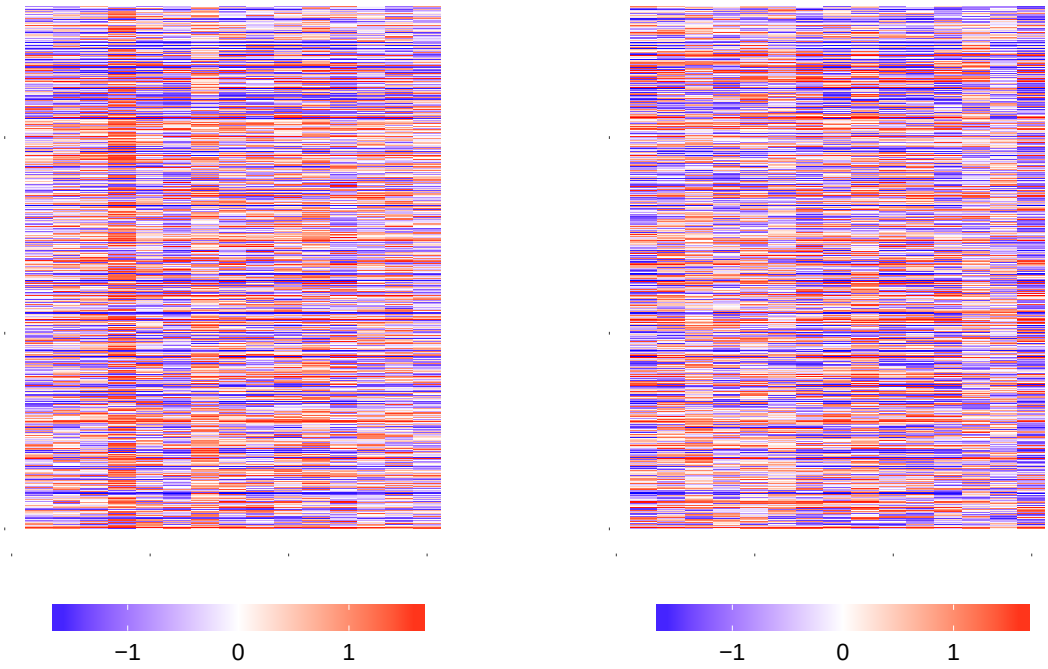
Para se obter a variabilidade da expressão gênica devido ao câncer, as amostras de carcinoma foram padronizadas pela mediana das amostras de tecido normais. A ideia é que os genes não relacionados à doença não devem ter uma mudança nos níveis de expressão entre tecidos normais e com câncer. No entanto, os genes que têm uma mudança significativa no nível de expressão devem produzir grupos (*clusters*) significativos com a aplicação dos algoritmos. Para visualizar os dados de expressão gênica apresentamos na Figura 14 os *heatmaps* para os dados de expressão antes do agrupamento de adenoma (adenoma-normal) e de câncer (câncer-normal) em que as amostras são representadas como colunas e os genes como linhas.

Aparentemente, os *heatmaps* não apresentam uma concentração de zero a níveis diferentes de expressão para os dados padronizados, sem existência clara de quaisquer grupos de genes diferentemente expressos. Foi utilizado como limiar 10^{-8}, valor próximo à precisão da máquina.

A partir dos *heatmaps* após o agrupamento podemos visualizar claramente diferentes grupos de genes diferentemente expressos. Os grupos inferiores tiveram genes menos expressos, enquanto os grupos superiores tem diferenças positivas entre os níveis de expressão de tecidos normais e câncer. A Figura 15 apresenta os *heatmaps* após o agrupamento obtido pelo algoritmo PPCLUST-H.

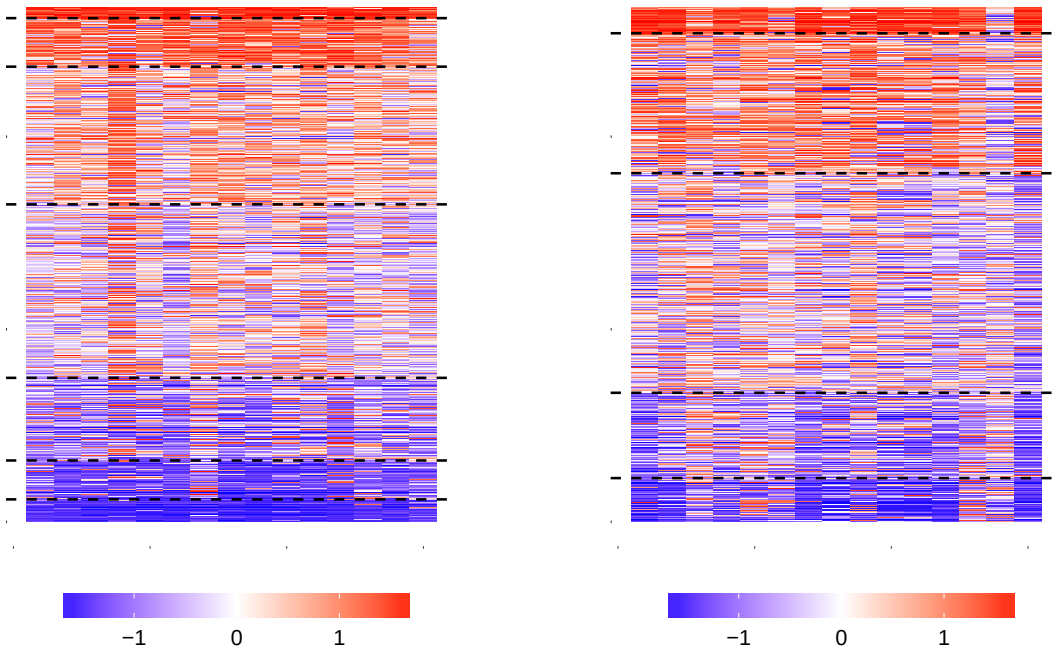
Os dois grupos extremos indicando valores expressos mais baixos em vermelho/-branco e maiores valores expressos em azul são mais evidentes do que outros grupos. Esses grupos em especial são formados por genes que têm um padrão de expressão destoante do

normal. A ideia é que grupos de genes como esses possam ser usados como um indicativo de câncer de cólon.



(a) Adenoma. (b) Carcinoma.

Figura 14 – *Heatmaps* para os dados de expressão gênica.



(a) Adenoma. (b) Carcinoma.

Figura 15 – *Heatmaps* após o agrupamento realizado pelo PPCLUST-H

Grupo	Qtd. genes	% genes	p-valor
1	108	2	0,003
2	504	9,4	0,049
3	1430	26,8	0,661
4	1802	33,7	0,703
5	858	16	0,521
6	404	7,6	0,999
7	233	4,4	0,999

Tabela 12 – Grupos de genes diferentemente expressos em tecidos tumorais pelo PPCLUST-H.

Grupo	Qtd. genes	% genes	p-valor
1	265	4,9	$4{,}9\times10^{-6}$
2	1454	27,3	0,999
3	2280	42,7	0,999
4	886	16,6	0,969
5	454	8,5	0,999

Tabela 13 – Grupos de genes diferentemente expressos em tecidos cancerígenos pelo PPCLUST-H.

Verifica-se que todos os p-valores dos grupos são altos (com p-valores entre cada par de grupos abaixo do limiar), o que indica que o algoritmo foi efetivo em identificar os padrões de expressão dos genes.

Considerando os grupos extremos foi encontrado um conjunto de 124 genes tiveram co-expressão forte tanto no adenoma quanto no carcinoma. De forma análoga, outro conjunto com 55 genes tiveram co-expressão bem abaixo do nível de expressão normal.

4.2.3 Análise de Microarranjo Longitudinal

Os dados de expressão gênica utilizado nesta subseção foram coletados por Gillespie et al. (2010) e utilizado por Silva (2012). Esses dados consistem em três repetições de uma cepa contendo leveduras do tipo selvagem (*wildtype*) e uma cepa contendo leveduras portadoras de mutação sensível a variações de temperatura (*cdc13-1*). As repetições das leveduras foram amostradas inicialmente a 23^{o}C, e, então, 1, 2, 3 e 4 horas após um aumento de temperatura para 30^{o}C. Ao todo foram observados 10,928 genes de leveduras em 5 pontos no tempo, com três repetições e dois tipos de condições experimentais: cepas de leveduras sem acréscimo de temperatura (controle) e cepas com alteração de temperatura (tratamento). Esses dados estão disponíveis na plataforma *ArrayExpress* com número de acesso: E-MEXP-1551.

A hipótese é de que a expressão gênica das leveduras *cdc13-1* sofreram variações ao longo do tempo em decorrência da temperatura. Nesse sentido, o objetivo desse estudo foi identificar grupos de genes que se expressaram de forma distinta entre si e de forma semelhante ao longo do tempo. Gillespie et al. (2010) reduziu a dimensão do estudo selecionando os 50 genes que apresentaram uma maior variabilidade em sua expressão ao longo do tempo. Isso permitiu a utilização das técnicas de agrupamento tradicionais. Os algoritmos implementados neste trabalho tiram proveito da multidimensionalidade, portanto, não há necessidade de reduzir a dimensão dos dados evitando a perda desnecessária de informações.

Para verificar o efeito do tratamento na expressão gênica das leveduras portadoras da mutação, Silva (2012) padronizou as observações da base de dados com tratamento em relação às observações da base de dados de controle. O autor verificou que cada gene possui um nível de expressão gênico distinto, dessa forma a padronização foi feita de acordo com a mediana de cada gene, ou seja, $Y_{ijk} - \tilde{X}_{i..}$, em que Y_{ijk} é a expressão gênica da levedura do tipo mutante e $\tilde{X}_{i..}$ é a mediana do i-ésimo gene do tipo selvagem.

A Figura 16 mostra o efeito da temperatura nas expressões dos genes ao longo do tempo. Percebe-se que a maior parte dos genes concentrou-se em torno de zero durante todo o período em análise, ou seja, não reagiu ao aumento da temperatura.

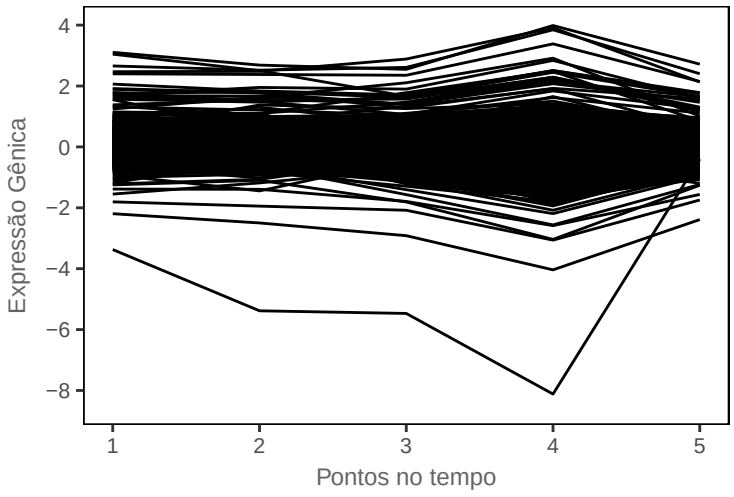


Figura 16 – Expressão gênica padronizada das leveduras.

A Figura 17 apresenta o resultado do agrupamento aplicando-se o procedimento PPCLUSTEL-H.

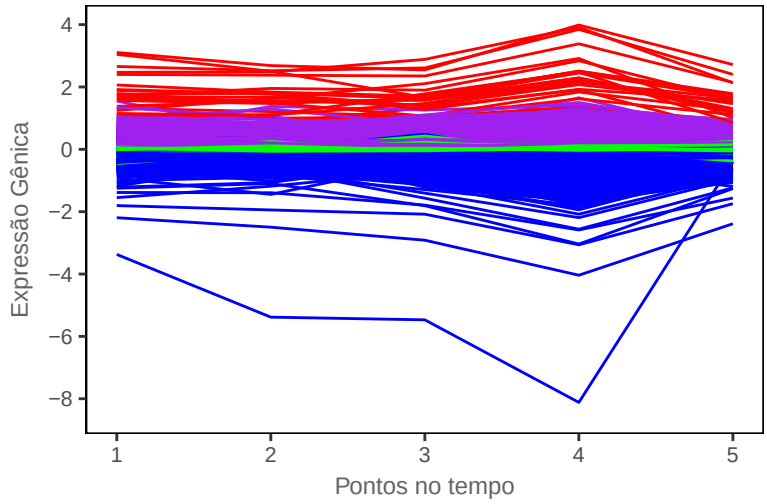


Figura 17 – Agrupamento obtido pelo algoritmo PPCLUSTEL-H com os dados originais. Foi utilizado com o limiar igual a 10^{-5}.

As linhas com coloração vermelha representam genes de maior expressão e a na cor azul representam os genes que se expressaram negativamente. Percebe-se que o algoritmo separou basicamente os genes sem mudança de expressão e outros com níveis de expressão variantes positivamente ou negativamente ao longo do tempo.

Em posse do número de grupos significativos obtido pelo algoritmo, o próximo passo foi informar ao MCLUST o número de grupos encontrado pelo PPCLUSTEL-H. A Figura 18 apresenta o resultado do agrupamento aplicando-se o procedimento MCLUST.

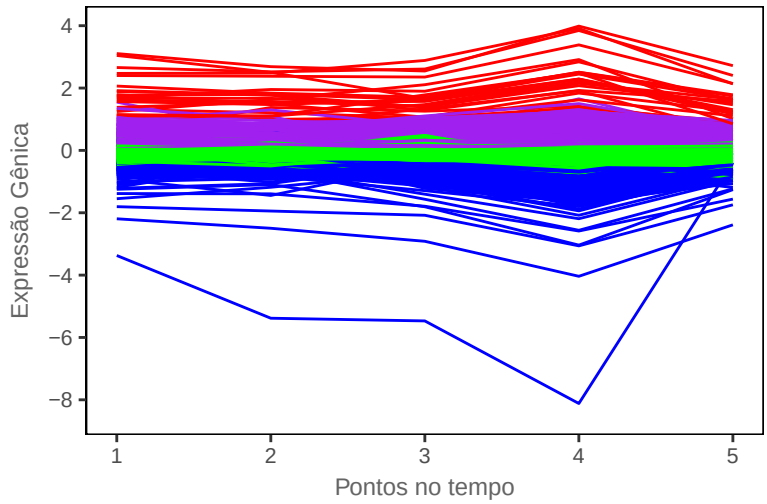


Figura 18 – Agrupamento obtido pelo algoritmo MCLUST com a especificação do número de grupos obtido pelo PPCLUSTEL-H.

Alternativamente, usando a competência do PPCLUSTEL-H em encontrar o número significativo de grupos a ideia foi usar essa informação a priori na aplicação do MCLUST para a obtenção de um agrupamento mais refinado dos dados.

Conclusão

Neste trabalho, foram implementados em um pacote **R** o algoritmo desenvolvido e implementados no *software* SAS por von Borries (2008), o PPCLUST e sua extensão PPCLUSTEL. No entanto, não foi possível a replicação exata desses procedimentos em linguagem **R**, visto que as implementações em SAS utilizam o procedimento sortndx de código fonte fechado. Ainda, foram sugeridas duas versões alternativas desses procedimentos adeptas de processamento paralelo: PPCLUST-H e PPCLUSTEL-H.

Todos os procedimentos se baseiam na utilização do p-valor resultante de dois testes estatísticos distintos: um não-paramétrico que testa a ausência de efeito simples e outro que avalia a ausência de efeito simples de grupo em um delineamento fatorial com medidas repetidas no tempo. O PPCLUST(-H) se destina a agrupar dados com um grande número de variáveis e poucas observações (dados HDLSS). O PPCLUSTEL(-H) se destina a agrupar dados longitudinais, com grande número de variáveis e um número fixo de pontos no tempo (dados HDLLSS).

Com relação ao desempenho dos procedimentos em dados simulados, verificou-se que o MCLUST tende a superar ambos os algoritmos apenas quando os dados seguem distribuições simétricas. O PPCLUST-H apresentou resultados competitivos em termos da qualidade e velocidade de agrupamento. Acrescido de suas propriedades de invariância a transformações monótonas, não necessidade de pressupostos de distribuição e robustez a valores discrepantes, esse algoritmo se torna uma opção conveniente para o agrupamento de dados HDLSS.

O PPCLUSTEL-H apresentou uma qualidade de agrupamento promissora na medida em que o número de níveis de fator e de pontos no tempo aumentou. Embora esse procedimento não tenha apresentado uma qualidade de agrupamento tão elevada quanto à do MCLUST, se mostrou um algoritmo veloz e eficiente em detectar grupos estatisticamente significantes. O MCLUST não incorpora variância devida às repetições. Essas repetições contêm informações importantes sobre o padrão de grupos dos dados. Uma maneira de aproveitar a capacidade do PPCLUSTEL-H é o seu uso como uma ferramenta para determinar o número real de grupos e, então, essa informação é passada ao MCLUST como o número de componentes da mistura de distribuições.

Ambos os procedimentos elaborados neste trabalho lograram êxito no agrupamento de dados reais: expressão gênica de câncer de cólon e de leveduras expostas a diferentes temperaturas. Ainda que em estágio exploratório, foi possível a identificação de grupos de genes significativamente diferentes quanto à sua expressão.

Esperamos que esse trabalho também sirva como incentivo a futuros estudos. Com

esse intuito, sugerimos os seguintes tópicos como propostas a trabalhos futuros:

- Comparação via simulações entre a versões originais do PPCLUST e do PPCLUSTEL implementadas no SAS e as versões ajustadas[1] para o **R** com o intuito de investigar se existe uma ordenação feita pela sub-rotina sortcenter que melhore o desempenho desses algoritmos.
- Desenvolvimento de uma metodologia para a escolha do limiar que forneça o agrupamento mais verossímil.
- O uso colaborativo do PPCLUSTEL-H com o MCLUST no agrupamento com a estrutura longitudinal superdimensionada com poucas repetições (HDLLSS).

[1] sub-rotina sortcenter

Referências

A. Alizadeh, M. B. Eisen, and R. E. Davis. Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature*, 403:503–511, 2000. Citado na página 34.

H. Corrada Bravo, V. Pihur, M. McCall, R. A. Irizarry, and J. T. Leek. Gene expression anti-profiles as a basis for accurate universal cancer signatures. *BMC Bioinformatics*, 13 (1):272, Oct 2012. ISSN 1471-2105. 10.1186/1471-2105-13-272. URL `https://doi.org/10.1186/1471-2105-13-272`. Citado na página 36.

J. Fan and J. Zhang. Two-step estimation of functional linear models with applications to longitudinal data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 62(2):303–322, 2000. 10.1111/1467-9868.00233. URL `https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/1467-9868.00233`. Citado na página 30.

C. Fraley and A. Raftery. Mclust: Software for model-based cluster analysis. 16:297–306, 01 1999. Citado 2 vezes nas páginas 2 e 9.

C. S. Gillespie, G. Lei, R. J. Boys, A. Greenall, and D. J. Wilkinson. Analysing time course microarray data using bioconductor: a case study using yeast2 affymetrix arrays. *BMC Research Notes*, 3(1):81, Mar 2010. ISSN 1756-0500. 10.1186/1756-0500-3-81. URL `https://doi.org/10.1186/1756-0500-3-81`. Citado 3 vezes nas páginas 34, 38 e 39.

P. Hubert and L. Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, Dec 1985. Citado na página 10.

L. Kaufman and P. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. 09 2009. ISBN 9780470317488. Citado na página 9.

F. Leisch. *Creating R Packages: A Tutorial*, 06 2018. Citado na página 5.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif., 1967. University of California Press. URL `https://projecteuclid.org/euclid.bsmsp/1200512992`. Citado na página 9.

C. M. Perou, S. S. Jeffrey, and M. van de Rijn. Distinctive gene expression patterns in human mammary epithelial cells and breast cancers. *Proceedings of the National Academy of Sciences*, 96(16):9212–9217, 1999. ISSN 0027-8424. 10.1073/pnas.96.16.9212. URL `http://www.pnas.org/content/96/16/9212`. Citado na página 34.

B. Pierce. *Genetics: A Conceptual Approach*. Genetics: A Conceptual Approach. W. H. Freeman, 2012. ISBN 9781429232524. URL https://books.google.com.br/books?id=z4pXRaZAkdkC. Citado na página 34.

R Core Team. *Writing R Extensions*. R Foundation for Statistical Computing, Vienna, Austria, 2007. Citado na página 5.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2017. URL http://www.R-project.org/. Citado na página 2.

A. E. Raftery, M. Fop, T. B. Murphy, and L. Scrucca. mclust 5: clustering, classification and density estimation using Gaussian finite mixture models. *The R Journal*, 8(1): 205–233, 2016. Citado na página 9.

W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971. 10.1080/01621459.1971.10482356. URL https://www.tandfonline.com/doi/abs/10.1080/01621459.1971.10482356. Citado na página 10.

N. Serban and L. Wasserman. Cats. *Journal of the American Statistical Association*, 100 (471):990–999, 2005. 10.1198/016214504000001574. URL https://doi.org/10.1198/016214504000001574. Citado na página 30.

A. P. T. Silva. *Implementação, análise e aplicação de algoritmos de agrupamento de dados superdimensionados, longitudinais e com amostras pequenas.* Dissertação (Mestrado em Estatística), Universidade de Brasília., 2012. Citado 4 vezes nas páginas 2, 32, 38 e 39.

S. Theodoridis and K. Koutroumbas. *Pattern Recognition, Fourth Edition*. Academic Press, Inc., Orlando, FL, USA, 4th edition, 2008. ISBN 1597492728, 9781597492720. Citado na página 7.

G. von Borries. *Partition clustering of High Dimensional Low Sample Size data based on P-Values*. Tese (Doutorado em Estatstica), 2008. Citado 7 vezes nas páginas 1, 2, 15, 16, 17, 25 e 41.

G. von Borries. *A SAS/JMP Integration for Implementation of a Clustering Algorithm for High Dimensional Low Sample Size Data*. SAS Global Forum. Paper 283, 2009. Citado 2 vezes nas páginas 20 e 28.

H. Wang. *Testing in Multi-factor heteroscedastic anova and repeated measures designs with large number of levels*. Tese (Doutorado em Estatistica)., 2004. Citado na página 15.

H. Wang and M. G. Akritas. Rank tests for anova with large number of factor levels. *Journal of Nonparametric Statistics*, 16(3-4):563–589, 2004. URL `https://doi.org/10.1080/10485250310001624774`. Citado na página 14.

H. Wickham. *R Packages*. O'Reilly Media, Inc., 1st edition, 2015. ISBN 1491910593, 9781491910597. Citado 2 vezes nas páginas 3 e 4.

C. Wu and C.-T. Chiang. Kernel smoothing on varying coefficient models with longitudinal dependent variable. 10:433–456, 04 2000. Citado na página 30.

Z. Yu, L. Xiaoyang, M. L. C., C. Hongbao, and C. Qiusheng. An integrative computational approach to evaluate genetic markers for chronic lymphocytic leukemia. *Journal of Computational Biology*, 24(9):942–952, 2017. 10.1089/cmb.2017.0041. URL `https://doi.org/10.1089/cmb.2017.0041`. PMID: 28570130. Citado na página 34.

Apêndice

O pacote em R desenvolvido neste trabalho pode ser obtido em:

$$< https://github.com/slrafael/Rclust > .$$