

Usando o **plantR** para atualizar informações entre duplicatas

Renato A. F. de Lima (raffima at usp.br)

23 June 2021

Sumário

1	Introdução	1
2	Obtendo os registros	2
2.1	Baixando os registros do speciesLink e GBIF	2
2.2	Unindo os registros	2
3	Padronização e validação de campos relevantes	2
3.1	Padronização das informações	3
3.2	Validação das informações padronizadas	3
4	Preparando os registros para a busca de duplicatas	5
4.1	Definindo os identificadores únicos	5
4.2	Definindo os campos a serem usados na busca	5
5	Obtendo as indicações de duplicatas	6
5.1	Inspecionando os resultados de número de duplicatas encontradas	6
6	Homogeneizando as informações entre grupos de duplicatas	6
6.1	Inspecionando os resultados da homogeneização	7
7	Preparando os dados para a atualização da coleção	7
7.1	Filtrando campos para atualização de coleções ou de sua base de dados	7
7.1.1	Qual são os registros da coleção que podem/devem ser atualizadas?	7
7.2	Atualização taxonômica	8
7.2.1	Quais campos podem ser usados na atualização taxonômica?	8
7.2.2	Algum nome não encontrado ou com notas em relação à Flora do Brasil?	8
7.2.3	Alguma atualização taxonômica feita nas duplicatas em outras coleções?	8
7.2.4	Salvando os resultados da atualização taxonômica	11
7.3	Atualização de coordenadas geográficas	11
7.4	Atualização de informações de localidade	11

1 Introdução

Esse tutorial foi pensado para auxiliar curadores de herbário que pretendem atualizar suas coleções usando informações oriundas de duplicatas depositadas em outras coleções. Como realizar esse processo para a coleção inteira demandaria obter as bases de dados de todas as coleções biológicas, os códigos abaixo foram preparados para serem executados por famílias individualmente.

Agradecimentos especiais ao André L. de Gasper pelos testes do pacote e pela sugestão de criar esse tutorial mais dirigido aos curadores de herbário.

2 Obtendo os registros

Antes de começar precisamos carregar o pacote usando a função `library()`, assumindo que você tem o pacote instalado junto ao seu R. Se você ainda não tem o pacote instalado em seu computador, siga as instruções do tutorial geral do pacote (i.e. `'plantr_tutorial.html'`).

```
library("plantR")
```

2.1 Baixando os registros do speciesLink e GBIF

Iremos usar a família *Blechnaceae* para esse exemplo, e baixando dados tanto do speciesLink (INCT) quanto do GBIF. O GBIF possui mais de 430,000 registros para a família no mundo todo. Assumindo que o curador seja de uma coleção do Brasil, iremos baixar apenas os coletas realizadas no Brasil (i.e. campo `country` igual a `'BR'`).

Para downloads do GBIF, o limite do número de registros baixados é controlado pelo argumento `n.records` e o padrão é baixar até 5000 registros por item buscado. Então o limite foi modificado para baixar mais registros. Contudo, há um limite do número máximo de 100,000 registros que podem ser baixados de um só vez. Por isso, iremos baixar apenas os registros da família para o Brasil ($<100,000$). Veja o help da função `rgbif2()` para sugestões sobre como fazer para famílias com mais de 100,000 registros no GBIF.

Note que o campo `species` da função `rgbif2()` aceita não apenas nomes ao nível de espécie, mas em qualquer nível taxonômico. Para a função `rspeciesLink()`, há um campo específico para família (i.e. `family`).

```
familia <- "Blechnaceae"
occs_splink <- rspeciesLink(family = familia)
occs_gbif <- rgbif2(species = familia,
                    country = "BR",
                    n.records = 450000)
dim(occs_splink)
```

```
## [1] 24156    51
```

```
dim(occs_gbif)
```

```
## [1] 23003    164
```

Foram encontrados aproximadamente 24,000 registros para a família no speciesLink e aprox. 23,000 registros para o Brasil no GBIF.

2.2 Unindo os registros

Em seguida, combinamos os registros dos dois repositórios em apenas um objeto usando a função `formatDwc()` e removendo colunas desnecessárias para a busca de duplicatas:

```
occs <- formatDwc(splink_data = occs_splink,
                  gbif_data = occs_gbif, drop = TRUE)
dim(occs)
```

```
## [1] 47159    47
```

Veja que o objeto `occs` contém agora a soma dos registros vindos do speciesLink e GBIF (i.e. aprox. 47,000 em Abril de 2021).

3 Padronização e validação de campos relevantes

Mesmo que você esteja interessado apenas em atualizar um tipo de informação (e.g. nome da espécie), é importante padronizar a notação e formato de campos que são importantes para a busca de duplicatas. Essa

padronização não é perfeita, mas ela aumenta as chances de recuperar duplicatas com pequenas diferenças de notação em campos como nome e número do coletor, a localidade da coleta, nome da família e a grafia e/ou formato do nome da espécie.

3.1 Padronização das informações

O pacote **plantR** possui funções que executam várias dessas edições de uma só vez. As funções `formatOcc()`, `formatLoc()`, `formatCoord()` e `formatTax()` fazem a padronização dos campos relacionados à coleta em si (e.g. código da coleção, nome e número do coletor), à localidade da coleta, às coordenadas geográficas associadas à coleta e à informação taxonômica.

A verificação ortográfica e de sinônimos via o The Plant List é relativamente demorada. Portanto, iremos fazer a conferência e validação dos nomes apenas usando a Flora do Brasil (i.e. campo `db` igual a `'fbo'`)

```
occs <- formatOcc(occs)
occs <- formatLoc(occs)
occs <- formatCoord(occs)
occs <- formatTax(occs, db = "fbo")
```

```
## The following family names were automatically replaced:
```

```
##
## |Genus          |Old fam.    |New fam.      |
## |:-----|:-----|:-----|
## |Acrostichum    |Blechnaceae |Pteridaceae   |
## |Blechnum       |Blechnaceae |Acanthaceae   |
## |Didymoglossum  |Blechnaceae |Hymenophyllaceae |
## |Phyllitis      |Blechnaceae |Aspleniaceae  |
## |Pteris         |Blechnaceae |Pteridaceae   |
```

Note que as funções não editam as colunas originais dos registros. Elas criam novas colunas que armazenam as informações editadas que em geral têm o mesmo nome da coluna original seguido do sufixo `'new'`.

3.2 Validação das informações padronizadas

Para cada conjunto de duplicatas, a tomada de decisão de qual espécimen será usado para obter a informação válida ou a mais atual sobre localidade, coordenadas geográficas ou identificação taxonômica depende de um processo de validação. Esse processo de validação não é essencial para encontrar as duplicatas entre coleções, mas ela ajuda no processo de tomada de decisão sobre qual é a melhor informação disponível.

Assim como para a padronização dos campos, o pacote **plantR** oferece funções que executam essas validações de uma só vez. As funções `validateLoc()` e `validateCoord()` fazem a validação dos campos de localidade e coordenadas geográficas, respectivamente.

```
occs <- validateLoc(occs)
```

```
## [1] "Locality resolution in the original data vs. edited data:"
##
##               original
## edited
## country      4502      234      8      0      34
## locality      10     6086      0      0      0
## municipality   2     23455    3664      0      0
## no_info        7        7      2     74      0
## stateProvince  0     7526     213      0    1335
```

```
occs <- validateCoord(occs)
```

A função `validateTax()` faz uma classificação da confiabilidade da identificação taxonômica dos registros. Essa avaliação é feita em uma base de dados global de nomes de taxonomistas por família (veja o help da

função digitando `?validateTax` no seu console do R).

Contudo, essa base de dados não é completa. Assim, pode ser importante executar a função de maneira preliminar para avaliar se não há nomes de taxonomistas que sejam relevantes à essa avaliação e que não estejam na base de dados. Vamos portanto visualizar os 15 nomes com mais determinações para a família usado no exemplo, e que não estão listados na base de dados como especialistas da família:

```
temp <- validateTax(occs, top.det = 15)
## Top people with many determinations but not in the taxonomist list:
##
## |Identifier          | Records|
## |-----|-----:|
## |Gonzatti, F.        |    1138|
## |Matos, F.B.         |     962|
## |Almeida, T.E.       |     845|
## |Pietrobon, M.R.     |     841|
## |Kazmirczak, C.      |     677|
## |Sehnem, A.          |     499|
## |Prado, J.           |     470|
## |(uc, A.R.S.         |     466|
## |Senna, R.M.         |     442|
## |Smith, A.R.         |     431|
## |Moran, R.C.         |     391|
## |Mickel, J.T.        |     381|
## |Pietrobon-Silva, M.R. |     378|
## |Athayde Filho, F.P. |     333|
## |Michelon, C.       |     320|
rm(temp)
```

Note que entre os nomes listados há vários especialistas do grupo, mas que não são exatamente especialistas da família em questão (e.g. 'Gonzatti, F.'). Outros nomes podem estar listados porque a base de dados usada está incompleta.

Para incluir nomes faltantes de taxonomistas, é possível usar o campo `miss.taxonomist`. Nesse exemplo vamos incluir dois nomes 'Kazmirczak, C.' e 'Prado, J.', que têm que ser fornecidos em um formato específico: nome da 'família + underline + nome do determinador.

```
falta.tax <- paste(familia, c("Kazmirczak, C.", "Prado, J."), sep = "_")
occs <- validateTax(occs, miss.taxonomist = falta.tax,
                    top.det = 15)
## Top people with many determinations but not in the taxonomist list:
##
## |Identifier          | Records|
## |-----|-----:|
## |Gonzatti, F.        |    1138|
## |Matos, F.B.         |     962|
## |Almeida, T.E.       |     845|
## |Pietrobon, M.R.     |     841|
## |Sehnem, A.          |     499|
## |(uc, A.R.S.         |     466|
## |Senna, R.M.         |     442|
## |Smith, A.R.         |     431|
## |Moran, R.C.         |     391|
## |Mickel, J.T.        |     381|
## |Pietrobon-Silva, M.R. |     378|
## |Athayde Filho, F.P. |     333|
```

```
## |Michelon, C.          |      320|
## |Caxambu, M.G.        |      292|
## |Barros, I.C.L.       |      289|
```

Note que agora os nomes incluídos já não são mais listados entre os nomes com mais identificações, mas que não são especialistas da família.

4 Preparando os registros para a busca de duplicatas

4.1 Definindo os identificadores únicos

Para a busca de duplicatas, precisamos de algumas informações adicionais. Uma delas é ter um identificador único para cada registro (i.e. número de tombo). Para isso podemos concatenar o código padronizado da coleção e seu número de registro ou tombo na coleção correspondente. Isso pode ser feito usando a função `getTombo()`:

```
occs$numTombo <- getTombo(occs[, "collectionCode.new"],
                          occs[, "catalogNumber"])
```

Como muitas coleções têm suas bases de dados tanto no speciesLink/INCT quanto no GBIF é normal termos números de tombo repetidos nos dados, as chamadas duplicatas virtuais. Como nosso interesse é pelas informações contidas nas duplicatas físicas entre coleções, é indicado remover números de tombo repetidos. Isso não afeta a busca por duplicatas, mas simplifica a sua checagem final, isto é, as duplicatas virtuais não estarão entre as duplicatas encontradas.

Como alguns registros não têm número de tombo (tombo igual a 'US_NA'), usamos o código abaixo para realizar a remoção apenas dos números de tombo repetidos:

```
dim(occs)[1]

## [1] 47159

occs <- occs[!duplicated(occs$numTombo) |
              grepl("_NA$", occs$numTombo, perl = TRUE),]
dim(occs)[1]

## [1] 31546
```

Note que cerca de um terço dos registros representavam duplicatas virtuais.

Note também que existe a possibilidade de coleções terem um mesmo número de tombo para registros diferentes, devido a algum tipo de erro. Mas após o código acima, não estamos mais considerando esse tipo de erro.

4.2 Definindo os campos a serem usados na busca

Uma informação essencial para a busca de duplicatas é a definição de quais campos serão usados na busca por duplicatas. No **plantR** é possível usar qualquer combinação dos campos contendo a informação do nome da família ou espécie, nome ou sobrenome do coletor, número do coletor, ano de coleta e localidade da coleta.

Como há muita variação de notação entre coleções no nome do coletor, é possível usar apenas o seu sobrenome. Além disso, como o uso apenas do sobrenome e número de coletor pode retornar falsas duplicatas para nomes muito comuns e números de coletor baixos (e.g. Lima 10), é aconselhado incluir outros campos na busca. Neste exemplo iremos usar três combinações de campos, que são geradas usando a função `prepDup()`:

```
dups <- prepDup(occs,
                comb.fields = list(c("species", "col.name", "col.number"),
                                   c("col.last.name", "col.number", "col.loc"),
                                   c("col.last.name", "col.number", "col.year")))
```

Note que salvamos os strings concatenados de busca em uma tabela à parte, pois essa informação não será necessária após a busca, evitando aumentar ainda mais o número de novas colunas.

Note também que o padrão da função é não incluir na busca por duplicatas registros com informação faltante de coletor, data ou localidade, etc. Se você quiser incluir esses registros o campo `ignore.miss` deverá ser modificado de `TRUE` para `FALSE`. Mas tome cuidado pois isso pode resultar em muitas falsas duplicatas!!

5 Obtendo as indicações de duplicatas

Finalmente, podemos usar a nova tabela gerada para buscar os registros com alguma indicação de serem um mesmo espécimen com material depositado em diferentes coleções.

```
dups <- getDup(dups)
head(dups[order(dups$dup.ID),], 4)
```

##	numTombo	dup.srch.str1	dup.srch.str2
## 8181	ALCB_56777	Blechnum_Loureiro, D.M._501	Loureiro_501_ilheus
## 8237	ALCB_1518	Blechnum occidentale_Loureiro, D.M._501	Loureiro_501_ilheus
## 2443	JPB_49351	Blechnum binervatum_Gomes, L.A._95	Gomes_95_capela
## 2473	ASE_21027	Neoblechnum brasiliense_Gomes, L.A._95	Gomes_95_capela

##	dup.srch.str3	dup.numb	dup.prop	dup.ID
## 8181	Loureiro_501_2001	2	0.666666666666667	[ALCB_1518 ALCB_56777]
## 8237	Loureiro_501_2001	2	0.666666666666667	[ALCB_1518 ALCB_56777]
## 2443	Gomes_95_2011	2	0.666666666666667	[ASE_21027 JPB_49351]
## 2473	Gomes_95_2011	2	0.666666666666667	[ASE_21027 JPB_49351]

5.1 Inspeccionando os resultados de número de duplicatas encontradas

Novas colunas foram criadas. A coluna ‘dup.prop’ contém a força de indicação das duplicatas (i.e. quanto mais próximo de 1 mais segurança na indicação). Vamos inspecionar as categorias geradas:

```
table(dups$dup.prop)
```

##	0	0.333333333333333	0.666666666666667	1
##	11800	1080	3450	10182
##	cc			
##	5034			

Em seguida, precisamos reunir as colunas de interesse da busca de duplicatas com a nossa planilha geral:

```
occs <- cbind.data.frame(occs,
  dups[,c("dup.ID", "dup.numb", "dup.prop")],
  stringsAsFactors = FALSE)
```

Para esses dados, cerca de 27% dos registros possuem forte indicação de duplicatas e cerca de 38% não tiveram nenhuma indicação (i.e. prováveis unicatas). Além desses, 16% foram classificados com ‘dup.prop’ > 0.5 (e.g. 2 sobre 3 combinações iguais) e como ‘cc’ de ‘cannot check’ por falta de informações associadas aos campos escolhidos. Para registros com ‘dup.prop’ < 0.5 (apenas 1 combinação em comum) sugere-se um ‘pente-fino’, pois esses casos podem representar problemas ou falsas duplicatas.

6 Homogeneizando as informações entre grupos de duplicatas

Por fim, podemos usar para todas as duplicatas de cada grupo as informações mais confiáveis ou recentes sobre os registros. Atualmente, o **plantR** faz essa homogeneização para as informações de localidade, coordenadas

geográficas e/ou taxonomia. Ela é feita usando a função `mergeDup()` cujo padrão é fazer apenas o merge para registros com `'dup.prop' > 0.75`. Isso pode ser alterado usando o campo `prop` da função, como abaixo.

```
occs <- mergeDup(occs, prop = 0.65)
```

O default dessa função usa os campos já editados pelo **plantR** e cria novas colunas com um `'1'` adicionado ao final do nome da coluna (e.g. coluna `'family.new'` após a homogeneização será armazenada na nova coluna `'family.new1'`). Lembre-se, contudo, que os nomes das colunas que serão usadas podem ser definidas pelo usuário (e.g. usar a coluna `'family'` ao invés da `'family.new'`).

6.1 Inspeccionando os resultados da homogeneização

Podemos comparar as mudanças obtidas para um dos campos, como por exemplo, o campo que define o nível de confiança na identificação do registro. Note que mais de cerca de 700 registros que possuíam nível definido como `'unknown'` e `'low'` passaram a ter um nível definido como `'high'`.

```
table(occs$tax.check, occs$tax.check1, dnn = c("antes", "depois"))
##               depois
## antes      high    low unknown
##   high    5774      0         0
##   low     517 13724      0
## unknown  268      0    11263
```

7 Preparando os dados para a atualização da coleção

7.1 Filtrando campos para atualização de coleções ou de sua base de dados

Para esse tutorial, vamos supor que o André L. de Gasper, atual curador do herbário FURB, queira comparar os materiais depositados no FURB com as demais coleções. Mais especificamente, ele quer atualizar as informações relacionadas à identificação taxonômica e aos campos de determinação.

7.1.1 Qual são os registros da coleção que podem/devem ser atualizadas?

Primeiro, precisamos definir o nome da coleção focal e filtrar quais registros possuem direta ou indiretamente informações que podem ser relevantes. No caso, iremos definir o FURB como a coleção focal.

```
colecacao <- "FURB"
occs1 <- occs[grepl(colecacao, occs$numTombo, perl = TRUE) |
              grepl(colecacao, occs$dup.ID, perl = TRUE), ]
dim(occs1)
## [1] 1747 103
```

Podemos inspecionar quantos registros há por coleção:

```
table(occs1$collectionCode.new)
##
##   ALCB   ASE   BHCB BHCB-SL   CESJ   CGMS   CRI   CTBS   EAFM   EVB
##     2     1    18     18    23     1    15     1     1     6
##   FCAB   FLOR   FPS   FUEL   FURB   HAS   HCF   HEPH   HSTM   HUCS
##     7    37    20    21  1282     1    28    11     2    69
##   HURB   IRAI   JOI   LUSC   MBM   MO   NX   NY   PACA   RB
##    22     1    43     2    38     1     3     1     2    30
##   RON   SAMES   SHPR   SLUI   UEC   UPCB   VIC   VIES
##     1     4     4     5     2    13     7     4
```

Note que há cerca de 1700 registros de Blechnaceae no nosso filtro, dos quais aprox. 1300 são de registros do FURB mesmo, e o restante dos registros estão espalhados em cerca de 40 outras coleções.

Se você quiser fazer o mesmo, mas para outra coleção basta trocar o nome da coleção que é atribuída ao objeto chamado `colecão`.

7.2 Atualização taxonômica

7.2.1 Quais campos podem ser usados na atualização taxonômica?

Podemos filtrar apenas as colunas que são de interesse para a checagem alvo que o André quer realizar, a checagem taxonômica. Para cada um dos campos de interesse iremos filtrar as informações básicas de cada registro, mais a informação original (em formato DwC), a editada pelo **plantR** (sufixo `'new'`) e a homogeneizada a partir dos grupos de duplicatas (sufixo `'new1'`) relativas à identificação taxonômica dos registros.

Note que como usamos os defaults do **plantR** os filtros das colunas também seguem esse padrão. Caso você tenha selecionado outras colunas, os nomes abaixo têm que ser adaptados.

```
colunas <- c("numTombo", "recordedBy.new", "recordNumber.new",
            "identifiedBy", "identifiedBy.new", "identifiedBy.new1",
            "yearIdentified", "yearIdentified.new", "yearIdentified.new1",
            "family", "family.new", "family.new1",
            "scientificName", "scientificName.new", "scientificName.new1",
            "scientificNameStatus", "tax.notes",
            "tax.check", "tax.check1", "dup.ID")
occs1.tax <- occs1[, colunas]
```

7.2.2 Algum nome não encontrado ou com notas em relação à Flora do Brasil?

```
table(occs1[grepl("FURB_", occs1$numTombo, fixed = TRUE) &
            !occs1$tax.notes %in% "", c("scientificName", "tax.notes")])
```

	scientificName	tax.notes
##	not found	was misspelled
##	<i>Blechnum xcaudatum</i> Cav.	0 14
##	<i>Blechnum xleopoldense</i> (Dutra) V.A.O.Dittrich & Salino	0 1
##	<i>Blechnum australe</i> L.	2 0
##	<i>Lomaridium acutum</i> (Desv.) Gasper & V.A.O.Dittrich	1 0

Sem grandes problemas para o FURB! Algumas notas em relação à diferença como os híbridos são anotados na Flora do Brasil e duas espécies que não ocorrem no Brasil.

7.2.3 Alguma atualização taxonômica feita nas duplicatas em outras coleções?

Para realizar essa operação, podemos realizar uma filtragem mais, para ficarmos apenas com os registro com indicações de duplicata (i.e. coluna `'dup.ID'` não vazia):

```
occs2.tax <- occs1.tax[!is.na(occs1.tax$dup.ID), ]
occs2.tax <- occs2.tax[order(occs2.tax$dup.ID), ]
```

Para facilitar o trabalho do curador, podemos marcar as duplicatas com indicação de alteração da identificação:

```
occs2.tax$checar.taxon <-
  !is.na(occs2.tax$scientificName.new) &
  occs2.tax$scientificName.new != occs2.tax$scientificName.new1 &
  grepl("FURB_", occs2.tax$numTombo, fixed = TRUE)
occs2.tax[occs2.tax$checar.taxon,]
```

	numTombo	recordedBy.new	recordNumber.new	identifiedBy
##	10516 FURB_41754	Falkenberg, D.B.	6416	A.L. de Gasper
##	10543 FURB_6461	Falkenberg, D.B.	6416	Gasper, AL de


```
## 2499 FURB_35038 Schmitt, J.L. 3169 A.L. de Gasper
## 2538 FURB_6656 Schmitt, J.L. 3169 Gasper, A.L. de
## 2281 FURB_6463 Gonzatti, F. 597 Dittrich, V.A.O.
## identifiedBy.new identifiedBy.new1 yearIdentified yearIdentified.new
## 10516 Gasper, A.L. Gasper, A.L. <NA> 2010
## 10543 Gasper, A.L. Gasper, A.L. <NA> 2010
## 2499 Gasper, A.L. Gasper, A.L. <NA> 2016
## 2538 Gasper, A.L. Gasper, A.L. <NA> n.d.
## 2281 Dittrich, V.A.O. Dittrich, V.A.O. <NA> 2018
## yearIdentified.new1 family family.new family.new1
## 10516 2010 Blechnaceae Blechnaceae Blechnaceae
## 10543 2010 Blechnaceae Blechnaceae Blechnaceae
## 2499 2017 Blechnaceae Blechnaceae Blechnaceae
## 2538 2017 Blechnaceae Blechnaceae Blechnaceae
## 2281 2018 Blechnaceae Blechnaceae Blechnaceae
## scientificName scientificName.new
## 10516 Blechnum auriculatum Cav. Blechnum auriculatum
## 10543 Blechnum auriculatum Cav. Blechnum auriculatum
## 2499 Lomaria spannagelii (Rosenst.) Gasper & V.A.O.Dittrich Lomaria spannagelii
## 2538 Lomaria spannagelii (Rosenst.) Gasper & V.A.O.Dittrich Lomaria spannagelii
## 2281 Blechnum L. Blechnum
## scientificName.new1 scientificNameStatus tax.notes tax.check tax.check1
## 10516 Blechnum australe name_w_authors high high
## 10543 Blechnum australe name_w_authors high high
## 2499 Parablechnum cordatum name_w_authors high high
## 2538 Parablechnum cordatum name_w_authors high high
## 2281 Blechnum xleopoldense indet/name_w_authors high high
## dup.ID
## 10516 BHCBSL_28997/BHCBSL_28997/FLOR_23621/FUEL_19780/FURB_41754/FURB_6461
## 10543 BHCBSL_28997/BHCBSL_28997/FLOR_23621/FUEL_19780/FURB_41754/FURB_6461
## 2499 FLOR_17332/FURB_35038/FURB_6656/JOI_11850
## 2538 FLOR_17332/FURB_35038/FURB_6656/JOI_11850
## 2281 FUEL_53109/FURB_39824/FURB_6463/HUCS_39468/SLUI_4389/VIC_37877/VIES_27594
## checar.taxon
## 10516 TRUE
## 10543 TRUE
## 2499 TRUE
## 2538 TRUE
## 2281 TRUE
```

Note que há cinco registros cujo nome encontrado do FURB é diferente do nome encontrado nas demais coleções. Vamos imprimir um exemplo para entender o que as funções estão fazendo. Iremos usar como exemplo o identificador de duplicatas ('dup.ID') 'FUEL_53109|FURB_39824|FURB_6463|HUCS_39468|SLUI_4389|VIC_37877|VIES_27594'.

```
occs2.tax[grepl('FURB_6463', occs2.tax$dup.ID),]
## numTombo recordedBy.new recordNumber.new identifiedBy identifiedBy.new
## 2238 FURB_39824 Gonzatti, F. 597 V.A.O. Dittrich Dittrich, V.A.O.
## 2239 HUCS_39468 Gonzatti, F. 597 Gonzatti, F. Gonzatti, F.
## 2244 VIES_27594 Gonzatti, F. 597 Gonzatti, F. Gonzatti, F.
## 2245 SLUI_4389 Gonzatti, F. 597 <NA> s.n.
## 2269 VIC_37877 Gonzatti, F. 597 <NA> s.n.
## 2273 FUEL_53109 Gonzatti, F. 597 Dittrich, VAO Dittrich, V.A.O.
## 2281 FURB_6463 Gonzatti, F. 597 Dittrich, V.A.O. Dittrich, V.A.O.
## identifiedBy.new1 yearIdentified yearIdentified.new yearIdentified.new1
```

## 2238	Dittrich, V.A.O.	<NA>	2018	2018
## 2239	Dittrich, V.A.O.	<NA>	2019	2018
## 2244	Dittrich, V.A.O.	<NA>	2012	2018
## 2245	Dittrich, V.A.O.	<NA>	n.d.	2018
## 2269	Dittrich, V.A.O.	<NA>	n.d.	2018
## 2273	Dittrich, V.A.O.	<NA>	2018	2018
## 2281	Dittrich, V.A.O.	<NA>	2018	2018
##	family	family.new	family.new1	
## 2238	Blechnaceae	Blechnaceae	Blechnaceae	
## 2239	Blechnaceae	Blechnaceae	Blechnaceae	
## 2244	Blechnaceae	Blechnaceae	Blechnaceae	
## 2245	Blechnaceae	Blechnaceae	Blechnaceae	
## 2269	Blechnaceae	Blechnaceae	Blechnaceae	
## 2273	Blechnaceae	Blechnaceae	Blechnaceae	
## 2281	Blechnaceae	Blechnaceae	Blechnaceae	
##		scientificName	scientificName.new	
## 2238	Blechnum xleopoldense (Dutra) V.A.O.Dittrich & Salino	Blechnum xleopoldense		
## 2239	Blechnum xleopoldense (Dutra) V.A.O.Dittrich & Salino	Blechnum xleopoldense		
## 2244	Blechnum xleopoldense (Dutra) V.A.O.Dittrich & Salino	Blechnum xleopoldense		
## 2245		Blechnum australe L.	Blechnum australe	
## 2269		Blechnaceae	Blechnaceae sp.	
## 2273	Blechnum xleopoldense (Dutra) V.A.O.Dittrich & Salino	Blechnum xleopoldense		
## 2281		Blechnum L.	Blechnum	
##	scientificName.new1	scientificNameStatus	tax.notes	tax.check
## 2238	Blechnum xleopoldense	hybrid_species/name_w_authors was misspelled		high
## 2239	Blechnum xleopoldense	hybrid_species/name_w_authors was misspelled		low
## 2244	Blechnum xleopoldense	hybrid_species/name_w_authors was misspelled		low
## 2245	Blechnum xleopoldense	name_w_authors not found		unknown
## 2269	Blechnum xleopoldense	family_as_genus		unknown
## 2273	Blechnum xleopoldense	hybrid_species/name_w_authors was misspelled		high
## 2281	Blechnum xleopoldense	indet/name_w_authors		high
##	tax.check1			
## 2238		high		
## 2239		high		
## 2244		high		
## 2245		high		
## 2269		high		
## 2273		high		
## 2281		high		
##			dup. ID	
## 2238	FUEL_53109 FURB_39824 FURB_6463 HUCS_39468 SLUI_4389 VIC_37877 VIES_27594			
## 2239	FUEL_53109 FURB_39824 FURB_6463 HUCS_39468 SLUI_4389 VIC_37877 VIES_27594			
## 2244	FUEL_53109 FURB_39824 FURB_6463 HUCS_39468 SLUI_4389 VIC_37877 VIES_27594			
## 2245	FUEL_53109 FURB_39824 FURB_6463 HUCS_39468 SLUI_4389 VIC_37877 VIES_27594			
## 2269	FUEL_53109 FURB_39824 FURB_6463 HUCS_39468 SLUI_4389 VIC_37877 VIES_27594			
## 2273	FUEL_53109 FURB_39824 FURB_6463 HUCS_39468 SLUI_4389 VIC_37877 VIES_27594			
## 2281	FUEL_53109 FURB_39824 FURB_6463 HUCS_39468 SLUI_4389 VIC_37877 VIES_27594			
##	checar.taxon			
## 2238		FALSE		
## 2239		FALSE		
## 2244		FALSE		
## 2245		FALSE		
## 2269		FALSE		

```
## 2273      FALSE
## 2281      TRUE
```

Para esse caso há identificações feitas por diferentes pessoas e em diferentes datas. Como descrito no tutorial do **plantR**, o registro com identificação mais recente feita por um especialista da família é usado como a identificação de referência - nesse caso, a identificação feita por ‘Dittrich, V.A.O.’ em 2018.

O registro ‘FURB_6463’, assim como o ‘SLUI_4389’ e o ‘VIC_37877’, possuem identificações diferentes e estes tiveram suas informações homogeneizadas automaticamente pelo **plantR** (colunas terminando em ‘new1’) e podem ser usadas pelo curador do FURB para atualização.

7.2.4 Salvando os resultados da atualização taxonômica

Em seguida, podemos salvar os resultados em uma planilha à parte para conferência, usando a função `saveData()`:

```
nome.arquivo <- paste(colecao, familia, "taxonomia", sep = "_")
saveData(occs2.tax,
         file.name = nome.arquivo,
         dir.name = "checagem_plantR",
         file.format = "csv", compress = FALSE)
```

Esses registros estão marcados como ‘TRUE’ na coluna ‘checar.taxon’ que foi salva na pasta ‘checagem_plantR’ do seu diretório local. Por fins de organização, usamos a combinação do nome da família e da coleção no nome do arquivo ‘.csv’ salvo localmente.

7.3 Atualização de coordenadas geográficas

[EM BREVE]

7.4 Atualização de informações de localidade

[EM BREVE]