Working with the RefSeq database

Jose V. Die

26/12/2017

0. Introduction

This vignette shows a tutorial of how I have been using `refseqR` to automate some common processes of my research. The package `refseqR` is built on top of `rentrez`, the excellent library written by **David Winter** to query the NCBI's API and fecth the resulting data.

In short, `refseqR` provides summary information at three different levels:

- mRNA summary
- GeneID summary
- Protein summary

First, load the library

```
library(refseqR)
```

1. mRNA summary

1.1 mRNA info

```
xm <-  "XM_020388824"
mrna <-  entrez_summary(db="nuccore", id= xm)
mrna
```

```
## esummary result with 31 items:
##  [1] uid          caption      title        extra        gi
##  [6] createdate   updatedate   flags        taxid        slen
## [11] biomol       moltype      topology     sourcedb     segsetsize
## [16] projectid    genome       subtype      subname      assemblygi
## [21] assemblyacc  tech         completeness geneticcode  strand
## [26] organism     strain       biosample    statistics   properties
## [31] oslt
```

The mRNA summary contains 31 items. You may want to check every item. I am usually interested in some of them such as id, accession, title, update, or length (bp):

```
## [1] "1150740591"
```

```
## [1] "XM_020388824"
```

```
## [1] "PREDICTED: Asparagus officinalis probable disease resistance protein At1g58602 (LOC109822593),
```

```
## [1] "2017/03/01"
```

```
## [1] 3314
```

```
## [1] "8|DH0086|male|Spear|Mature|Netherlands"
```

We can also fetch the data from NCBI. Here, the first 30 lines:

```
##  [1] "LOCUS       XM_020388824            3314 bp    mRNA    linear   PLN 01-MAR-2017"
##  [2] "DEFINITION  PREDICTED: Asparagus officinalis probable disease resistance"
##  [3] "            protein At1g58602 (LOC109822593), transcript variant X2, mRNA."
##  [4] "ACCESSION   XM_020388824"
##  [5] "VERSION     XM_020388824.1"
##  [6] "DBLINK      BioProject: PRJNA376608"
##  [7] "KEYWORDS    RefSeq."
##  [8] "SOURCE      Asparagus officinalis (garden asparagus)"
##  [9] "  ORGANISM  Asparagus officinalis"
## [10] "            Eukaryota; Viridiplantae; Streptophyta; Embryophyta; Tracheophyta;"
## [11] "            Spermatophyta; Magnoliophyta; Liliopsida; Asparagales;"
## [12] "            Asparagaceae; Asparagoideae; Asparagus."
## [13] "COMMENT     MODEL REFSEQ:  This record is predicted by automated computational"
## [14] "            analysis. This record is derived from a genomic sequence"
## [15] "            (NC_033801.1) annotated using gene prediction method: Gnomon."
## [16] "            Also see:"
## [17] "                Documentation of NCBI's Annotation Process"
## [18] "            "
## [19] "            ##Genome-Annotation-Data-START##"
## [20] "            Annotation Provider         :: NCBI"
## [21] "            Annotation Status           :: Full annotation"
## [22] "            Annotation Version          :: Asparagus officinalis Annotation"
## [23] "                                           Release 100"
## [24] "            Annotation Pipeline         :: NCBI eukaryotic genome annotation"
## [25] "                                           pipeline"
## [26] "            Annotation Software Version :: 7.3"
## [27] "            Annotation Method           :: Best-placed RefSeq; Gnomon"
## [28] "            Features Annotated          :: Gene; mRNA; CDS; ncRNA"
## [29] "            ##Genome-Annotation-Data-END##"
## [30] "FEATURES             Location/Qualifiers"
```

I am interested in some features, for example plant sex, tissue, genotype, and the CDS coordinates. To obtain that info, `refseqR` comes with some functions:

```
* `extract_from_xm`
* `extract_CDSfrom_xm`
```

```
target <-  mrna_gb
extract_from_xm(mrna_gb, feat = "tissue")
```

```
## [1] "Spear"
```

```
extract_from_xm(mrna_gb, feat = "sex")
```

```
## [1] "male"
```

```
extract_from_xm(mrna_gb, feat = "genotype")
```

```
## [1] "DH0086"
```

I usually need the coordinates of the CDS related to the mRNA molecule:

```
extract_CDSfrom_xm(target)
```

```
## $startCDS
## [1] 141
##
## $stopCDS
## [1] 2894
```

The CDS coordinates come in handy when we want to get the fasta sequence. We sometimes do not want the 5'UTR contained in the XM sequence and are interested just in the CDS.

- Here, the first 500 nucleotides of the mRNA:

```
mrna_fasta = entrez_fetch(db="nuccore", id=xm, rettype="fasta")
# take a look at the first 500 chars.
cat(strwrap(substr(mrna_fasta, 1, 500)), sep="\n")
```

```
## >XM_020388824.1 PREDICTED: Asparagus officinalis probable disease
## resistance protein At1g58602 (LOC109822593), transcript variant
## X2, mRNA
## TCTAACGGTTTGATATCGATGATTTTCTTATGCCGCAAATGATACGTCTACCGCTACCGAAATATGAGCC
## GTCGAAAACCAATAGAAAATGATGATCCCATTCACACTGGAAACGGAACGGAAACTTCAATAGAGACCAA
## ATGTCAACGCGGCGAGTTCGAAAAACAAAAGGAAAAATCCCCAAAAAAAAAATTTCAGTGGAGAAGCTAG
## GGCAGCTTTTGATTCAGGAAACCAAGTTCTTGTCCGAAATTGGAGGTGAAATCGAGTGGCTTCGAACTGA
## GCTTCGATGGATGGAGAGCTTCCTCAAAGATGCAGATGCTAAGAGGAGGAAAGGGGATGAGAGGGTCAAG
## AACTGG
```

- Here, the first 500 nucleotides of the CDS:

```
coord <- extract_CDSfrom_xm(target)
cds <- entrez_fetch(db="nuccore", id=xm, rettype="fasta",
              seq_start = coord$startCDS, seq_stop = coord$stopCDS)
# CDS fasta sequence (look at the first 500 chars)
cat(strwrap(substr(cds, 1, 500)), sep="\n")
```

```
## >XM_020388824.1:141-2894 PREDICTED: Asparagus officinalis probable
## disease resistance protein At1g58602 (LOC109822593), transcript
## variant X2, mRNA
## ATGTCAACGCGGCGAGTTCGAAAAACAAAAGGAAAAATCCCCAAAAAAAAAATTTCAGTGGAGAAGCTAG
## GGCAGCTTTTGATTCAGGAAACCAAGTTCTTGTCCGAAATTGGAGGTGAAATCGAGTGGCTTCGAACTGA
## GCTTCGATGGATGGAGAGCTTCCTCAAAGATGCAGATGCTAAGAGGAGGAAAGGGGATGAGAGGGTCAAG
## AACTGGGTACGAGACGTTGCATACCAAGCCGAAGACGTTGTTGACCTCTTCTTCCTGCAGAATGATAGTA
## AGCAAGGAGCAATAGCAGAGTTCTTCCGGAGCTACATTTGCTTTCTCTCTGATCTAGTGGGCCTCCAT
```

The function `save_CDSfasta_from_xms` uses the CDS coordinates to fetch the NCBI data, extract the CDS sequence and save it in a fasta file.

Save nucleotide sequences into a FASTA file

```
save_CDSfasta_from_xms(cds, "myfasta")
```

The function `save_CDSfasta_from_xms` can create a single- or multi-fasta file as well.

```
xms = c("XM_020386193", "XM_020389493", "XM_020394534")
save_CDSfasta_from_xms(xms, "myfastas")
```

```
## 2. GeneID summary
### 2.1 GeneID info
From the mRNA sequence we can move forward. For example, we could get a number of features from the gene
```

```
mrna_links <- entrez_link(dbfrom = "nuccore", id = xm, db = "all")

mrna_links$links
```

```
## elink result with information from 8 databases:
## [1] nuccore_bioproject            nuccore_mrna_nuccore
## [3] nuccore_nuccore_mrnaonly      nuccore_protein
## [5] nuccore_taxonomy              nuccore_bioproject_transcript
## [7] nuccore_gene                  nuccore_sparcle_mrna
```

In this example, the accession XM_020388824 has 8 links to NCBI databases. One interesting database (db) is the Protein db. The URL link to connect with the protein db is:

Protein id link

```
mrna_links$links$nuccore_protein
```

```
## [1] "1150740592"
```

We will use that id to take a look at the info contained at the protein db later in this tutorial. But first, let's go through another very interesting db: Gene db.

Gene id link

```
mrna_links$links$nuccore_gene
```

```
## [1] "109822593"
```

We access the content in two steps:

- get the database link
- get the info summary for that link

```
# get the geneID for the mRNA
gene_id <- mrna_links$links$nuccore_gene
```

```
# use the geneID to connect with the Gene db and get the summary
gene <- entrez_summary(db = "gene", id = gene_id)
gene
```

```
## esummary result with 20 items:
##  [1] uid                 name                description
##  [4] status              currentid           chromosome
##  [7] geneticsource       maplocation         otheraliases
## [10] otherdesignations   nomenclaturesymbol  nomenclaturename
## [13] nomenclaturestatus  mim                 genomicinfo
## [16] geneweight          summary             chrsort
## [19] chrstart            organism
```

2.2 Acessing the GeneID info

Now, there is a number of items that I want to keep for my records:

- Info related to the LOC symbol and gene description.

```
gene$name
```

```
## [1] "LOC109822593"
```

```
gene$description
```

```
## [1] "probable disease resistance protein At1g58602"
```

- Info related to the chromosome, start/end coordinates and exon number.

```
gene$chromosome
```

```
## [1] "8"
```

```
gene$genomicinfo
```

```
##   chrloc   chraccver chrstart chrstop exoncount
## 1      8 NC_033801.1  9494840 9499026         3
```

- Info related to the species: scientific/common name, and taxon ID.

```
gene$organism$scientificname
```

```
## [1] "Asparagus officinalis"
```

```
gene$organism$commonname
```

```
## [1] "garden asparagus"
```

```
gene$organism$taxid
```

```
## [1] 4686
```

3. Protein summary

3.1 Protein info

Earlier in the tutorial (section 2.1) we got the protein id for the mRNA sequence accession XM_020388824.

```
protein_id <- mrna_links$links$nuccore_protein
```

Now, we can use that id to extract the info summary for the link.

```
## esummary result with 30 items:
##  [1] uid          caption      title        extra        gi
##  [6] createdate   updatedate   flags        taxid        slen
## [11] biomol       moltype      topology     sourcedb     segsetsize
## [16] projectid    genome       subtype      subname      assemblygi
## [21] assemblyacc  tech         completeness geneticcode  strand
## [26] organism     strain       statistics   properties   oslt
```

3.2 Acessing the Protein info

The protein summary contains 30 items. You may want to check every item. I am usually interested in some of them such as :

- Protein description

```
protein$title
```

```
## [1] "putative disease resistance protein At1g50180 isoform X2 [Asparagus officinalis]"
```

- Protein accession

```
protein$caption
```

```
## [1] "XP_020244413"
```

- Protein update

```
protein$updatedate
```

```
## [1] "2017/03/01"
```

- Protein length (aa)

```
protein$slen
```

```
## [1] 917
```

- Database

```
protein$sourcedb
```

```
## [1] "refseq"
```

We can also fetch the data from NCBI. Here, the first 30 lines:

```r
protein_gb <- entrez_fetch(db= 'protein', id = protein_id, rettype = 'gp')
strsplit(protein_gb, "\n")[[1]][1:30]
```

```
##  [1] "LOCUS       XP_020244413             917 aa            linear   PLN 01-MAR-2017"
##  [2] "DEFINITION  putative disease resistance protein At1g50180 isoform X2 [Asparagus"
##  [3] "            officinalis]."
##  [4] "ACCESSION   XP_020244413"
##  [5] "VERSION     XP_020244413.1"
##  [6] "DBLINK      BioProject: PRJNA376608"
##  [7] "DBSOURCE    REFSEQ: accession XM_020388824.1"
##  [8] "KEYWORDS    RefSeq."
##  [9] "SOURCE      Asparagus officinalis (garden asparagus)"
## [10] "  ORGANISM  Asparagus officinalis"
## [11] "            Eukaryota; Viridiplantae; Streptophyta; Embryophyta; Tracheophyta;"
## [12] "            Spermatophyta; Magnoliophyta; Liliopsida; Asparagales;"
## [13] "            Asparagaceae; Asparagoideae; Asparagus."
## [14] "COMMENT     MODEL REFSEQ:  This record is predicted by automated computational"
## [15] "            analysis. This record is derived from a genomic sequence"
## [16] "            (NC_033801.1) annotated using gene prediction method: Gnomon."
## [17] "            Also see:"
```

```
## [18] "                Documentation of NCBI's Annotation Process"
## [19] "            "
## [20] "            ##Genome-Annotation-Data-START##"
## [21] "            Annotation Provider         :: NCBI"
## [22] "            Annotation Status           :: Full annotation"
## [23] "            Annotation Version          :: Asparagus officinalis Annotation"
## [24] "                                           Release 100"
## [25] "            Annotation Pipeline         :: NCBI eukaryotic genome annotation"
## [26] "                                           pipeline"
## [27] "            Annotation Software Version :: 7.3"
## [28] "            Annotation Method           :: Best-placed RefSeq; Gnomon"
## [29] "            Features Annotated          :: Gene; mRNA; CDS; ncRNA"
## [30] "            ##Genome-Annotation-Data-END##"
```

I am usually interested in the molecular weight of the protein. The following function will do the job:

* `extract_mol.wt_from_xp`

Now, we can get the molecular weigh (in Daltons):

```
extract_mol.wt_from_xp(protein_gb)
```

```
## [1] 104178
```

We may want to download the amino acid sequence into a fasta file.

```
## >XP_020244413.1 putative disease resistance protein At1g50180
## isoform X2 [Asparagus officinalis]
## MSTRRVRKTKGKIPKKKISVEKLGQLLIQETKFLSEIGGEIEWLRTELRWMESFLKDADAKRRKGDERVK
## NWVRDVAYQAEDVVDLFFLQNDSKQGAIAEFFRSYICFLSDLVGLHELGVEISQIKSKVLRICESRDAYG
## IVSLSESREQSSYSAVDAMLQVRRQSSPHLDDDMVVVGFDTYKQFILELLLDTNIARRCVISIVGMGGLG
## KTTLATMVYNSSEVETHFSICAWITVSQDYRVSELLKNIMKRVMGTVFTGEHYERLENLEEDELKSKLYN
## FLKQTRYLIVLDDIWAQEAWEQIKAAFPNAKNGSRVLLTTRLMDVARSADPRVPPYELPFLTHEQSWELF
## LKKAFPSDQDFTPSCPKELEELGHEIVKRCGGLPLAVVVLGGLLSRKE
```

Similarly to nucleotide sequences, we can define a function to create a fasta file with the amino acids of a vector of XP proteins:

Save aa sequences into a FASTA file:

```
xps = c("XP_020271897", "XP_020271898", "XP_020271899")
save_AAfasta_from_xps(xps, "myAAfastas")
```

4. Common operations

When working with refseq accessions, there is a number of common operations that can be performed in a programmatically way.

* Convert XM --> XP
* Convert XP --> XM
* Convert LOC --> XP
* Convert LOC --> XM

```
xm <-  "XM_020388824"
getXP(xm)
```

```
## [1] "XP_020244413"
```

```
xp <-  "XP_020244413"
getXM(xp)
```

```
## [1] "XM_020388824"
```

The Gene db at Genbank provides a symbol that is constructed with the prefix 'LOC'. You may want to read the Gene Help for more information.
Another common operation is to switch between LOC symbols, and XP, or XM accessions.

```
locIds = c("LOC101515097", "LOC101515098", "LOC101515099")
getXPfromLOC(locIds)
```

```
## [1] "XP_004495855" "XP_004515819" "XP_004515900"
```

```
getXMfromLOC(locIds)
```

```
## [1] "XM_004495798" "XM_004515762" "XM_004515843"
```

5. Concluding Remarks

This tutorial is based on `rentrez` packg. On top of it, 'refseqR'contains a number of functions to programmatically automatize some common operations.

Functions to extract features from XM Genbank format

- `extract_from_xm`
- `extract_CDSfrom_xm`
- `save_CDSfasta_from_xms`

Functions to extract features from XP Genbank format

- `extract_mol.wt_from_xp`
- `save_AAfasta_from_xps`

Common operations

- `getXP`
- `getXM`
- `getXPfromLOC`
- `getXMfromLOC`

I'd really appreciate your feedback. The whole code used in this tutorial is available from my **Github** repository. I usually **tweet** about Genomics and Coding. You can contact me by **email** or visit my **website**.

Córdoba, (Spain), 2018-03-06.