HIV.db: A package that provides HIV/SIV feature database and query APIs

Mike Jiang,Raphael Gottardo

January 31, 2013

Abstract

Contents

1 Introduction

The current HIV.db package provides simple API to access the HIV(HXB2) feature database that contains HIV genes,gene products,genomic structure elements and epitopes.

2 Load a feature database

The `loadFeatures` function load the feature database for the selected genome and genomic region into memory. The returned `environment` will contain the chosen reference element and its children features (i.e all the features located in the same ranges of the genome and on the same frame). The ranges of the features are given by default in amino acid coordinates relative to the reference.

2.1 Basic use

```
> HIV_env<-loadFeatures()
```

By default, the function will load the features for the enveloppe of HIV. Thus, the previous call to `loadFeatures` is equivalent to the following:

```
> HIV_env<-loadFeatures(ref="env", genome="hxb2")
```

HXB2 is the reference genome for HIV in the package.

2.2 Specifying a reference or a genome

To load a different feature, use the ref and genome arguments. Here we load the features for the Gag gene in SIV:

```
> SIV_Gag<-loadFeatures(ref="Gag", genome="mac239")
```

In this package, we use mac239 as the reference genome for SIV. With hxb2 it is at the moment the only two available options.

2.3 Changing the coordinate system

Alternately, `loadFeatures` can load features in DNA coordinates relative to the full genome.

```
> HIV_pol<-loadFeatures(ref="pol", DNA=TRUE)
```

2.4 Exploring the database

After the database is loaded, users can explore the database by `lsCategory`, which lists the types of features availabe in the database for query.

```
> lsCategory(HIV_env)
```

```
[1] "gene"     "protein"  "loop"     "RNA"      "strand"   "helix"    "membrane"
```

3 Query HIV feature database

The package provides two query methods:`getFeature` for HIV features and `getEpitope` for HIV epitopes.

3.1 Default query

`getFeature` take the result of `loadFeatures` as its first and only required argument. If no other argument is passed, the function will return the list of all the features in the selected database.

```
> getFeature(SIV_Gag)
```

```
HivFeature with 7 rows and 3 value columns across 1 space
     space     ranges |        name    category     frame
  <factor>  <IRanges> | <character> <character> <integer>
1        1 [  0, 511] |         Gag        gene         3
2        1 [  0, 135] |         p15     protein         3
3        1 [135, 364] |         p27     protein         3
4        1 [364, 381] |          p2     protein         3
5        1 [381, 433] |          p8     protein         3
6        1 [433, 447] |          p1     protein         3
7        1 [447, 511] |          p6     protein         3
```

Different genes,proteins or viral regions can be searched by their names or categories:

```
> getFeature(HIV_env, name=c("gp120","gp41"))
> getFeature(SIV_Gag, category="protein")
```

and filter the query results further by the reading frames.

```
> getFeature(HIV_env, category=c("RNA"),frame=3)
```

```
HivFeature with 4 rows and 3 value columns across 1 space
     space     ranges |                                          name    category
  <factor>  <IRanges> |                                   <character> <character>
1        1 [519, 525] |         19-base silencer RNA stem 1 side 1          RNA
2        1 [591, 597] |         19-base silencer RNA stem 1 side 2          RNA
3        1 [784, 788] |            1DUQ PDB entry chain A is bases          RNA
4        1 [788, 815] | LLP-3 Leucine zipper-like anphypathic helix         RNA
      frame
  <integer>
1         3
2         3
3         3
4         3
```

Coordinates can also be used to specify the start and end nucleotide positions. Note that the coordinates have to be of the same type as the one used in `loadFeatures` within the feature sequence. Here with SIV_Gag which has been loaded in amino-acid coordinates.

```
> getFeature(SIV_Gag,start=0,end=370)
```

```
HivFeature with 4 rows and 3 value columns across 1 space
     space     ranges |        name    category     frame
  <factor>  <IRanges> | <character> <character> <integer>
1        1 [  0, 511] |         Gag        gene         3
2        1 [  0, 135] |         p15     protein         3
3        1 [135, 364] |         p27     protein         3
4        1 [364, 381] |          p2     protein         3
```

Now with HIV_pol in DNA coordinates.

```
> getFeature(HIV_pol, start=2000, end=3000)
```

```
HivFeature with 4 rows and 3 value columns across 1 space
     space       ranges |        name    category     frame
  <factor>    <IRanges> | <character> <character> <integer>
1        1 [2253, 5096] |         pol        gene         3
2        1 [2253, 2549] |         p10     protein         3
3        1 [2550, 3869] |         p51     protein         3
4        1 [2550, 4229] |         p66     protein         3
```

It will return every feature that has a part of its sequence between the start and end argument, even if the sequence is actually longer.

3.2 Querying epitopes

The same query can be done to the epitope database by `getEpitope` method.

```
> getEpitope(HIV_pol)
```

Epitope queries can also be filtered, the available filters are the ranges, the frame and the species.

```
> getEpitope(HIV_env, start=50, end=70, frame=3, species="mouse")
```

```
Epitope with 4 rows and 6 value columns across 1 space
     space    ranges |        name    category     frame              Epitope
  <factor> <IRanges> | <character> <character> <integer>          <character>
1        1  [41, 60] |         M86     Epitope         3 VPVWKEATTTLFCASDAKAY
2        1  [51, 70] |  polyclonal     Epitope         3 LFCASDAKAYDTEVHNVWAT
3        1  [60, 69] |     133/237     Epitope         3           YDTEVHNVWA
4        1  [63, 77] |      133/11     Epitope         3      EVHNVWATHACVPTD
      Species     Subtype
  <character> <character>
1       mouse           B
2       mouse           B
3       mouse           B
4       mouse
```

Alternately, a name can be specified to retrieve a specific epitope.

```
> getEpitope(HIV_pol, name="13E1")

Epitope with 1 row and 6 value columns across 1 space
     space       ranges |        name    category     frame     Epitope
  <factor>    <IRanges> | <character> <character> <integer> <character>
1        1 [2364, 2385] |        13E1     Epitope         3    LPGRWKPK
      Species     Subtype
  <character> <character>
1     hamster           B
```

`getEpitope` also takes a `HivFeature` object as input and use the HXB2 coordinates range to get the appropriate epitopes.

```
> gp41<-getFeature(HIV_env, name="gp41")
> getEpitope(gp41, species="mouse")
```

4 Query by parent/children relations

HivFeatures have the parent or children features based on the relative positions of their HXB2 coordinates. We provide two methods to query the children or parent features: `getChildren` and `getParent`.

```
> getChildren(gp41)

> getParent(gp41)

HivFeature with 1 row and 3 value columns across 1 space
     space    ranges |        name    category     frame
  <factor> <IRanges> | <character> <character> <integer>
1        1  [0, 857] |         env        gene         3
```

When `recursive` is set as `TRUE`,all the descendants or ancestors are returned besides the immediate children or parents.

```
> V1_loop<-getFeature(HIV_env, name="V1")
> getParent(V1_loop,recursive=TRUE)
> env<-getFeature(HIV_env, name="env")
> getChildren(env, recursive=TRUE)
```

5 Sequence of feature objects

We also provide two methods to extract amino acid or DNA sequence: `getAA` and `getDNA`.

5.1 From HIV features

Sequence can be extracted from `HivFeature` objects directly

```
> getAA(gp41)

  857-letter "AAString" instance
seq: MRVKEKYQHLWRWGWRWGTMLLGMLMICSATEKLWV...VAEGTDRVIEVVQGACRAIRHIPRRIRQGLERILL*

> getDNA(gp41)

  A DNAStringSet instance of length 1
    width seq                                                   names
[1]  2571 ATGAGAGTGAAGGAGAAATATCA...GCTTGGAAAGGATTTTGCTATAA HXB2_NT_Full_Genome
```

By default, if no feature name is provided, the returned sequence is the one of the reference used when loading features in the enviornment. The DNA sequence is the one coding for the reference.

When the argument name is specified, the functions will return the sequences of the feature corresponding to the name.

```
> gp_features<-getFeature(HIV_env)
> getAA(gp_features, name=c("gp41","gp120"))

$gp120
  482-letter "AAString" instance
seq: ATEKLWVTVYYGVPVWKEATTTLFCASDAKAYDTEV...RDNWRSELYKYKVVKIEPLGVAPTKAKRRVVQREKR

$gp41
  347-letter "AAString" instance
seq: RAVGIGALFLGFLGAAGSTMGAASMTLTVQARQLLS...VAEGTDRVIEVVQGACRAIRHIPRRIRQGLERILL*

> getDNA(gp_features, name=c("gp41","gp120"))

$gp120
  A DNAStringSet instance of length 1
    width seq                                                   names
[1]  1444 TACAGAAAAATTGTGGGTCACAG...GAGTGGTGCAGAGAGAAAAAAGA HXB2_NT_Full_Genome

$gp41
  A DNAStringSet instance of length 1
    width seq                                                   names
[1]  1039 AGCAGTGGGAATAGGAGCTTTGT...GCTTGGAAAGGATTTTGCTATAA HXB2_NT_Full_Genome
```

5.2 From environments

It is also possible to get sequences directly from the `environment` objects returned by `loadFeatures`.

```
> getAA(HIV_pol)

  1004-letter "AAString" instance
seq: FFREDLAFLQGKAREFSSEQTRANSPTRRELQVWGR...NSDIKVVPRRKAKIIRDYGKQMAGDDCVASRQDED*

> getDNA(HIV_pol)

  A DNAStringSet instance of length 1
    width seq                                                 names
[1]  2844 CCTCAGGTCACTCTTTGGCAACG...CAAGTAGACAGGATGAGGATTAG HXB2_NT_Full_Genome
```

Here again, names can be provided to select specific features sequences.

6 conclusion

The package allows users to query the built-in HIV features database for the important information about HIV gene and gene product as well as genomic structure elements.

7 Reference

http://www.hiv.lanl.gov/content/sequence/HIV/MAP/landmark.html