

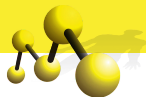
# File Parsing

Luis Pedro Coelho

Programming for Scientists

October 22, 2012





## What's a File?

A sequence of bytes.

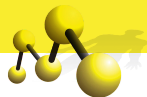
(and meta-data).



## ASCII

- 65: A
- 66: B
- ...
- 48: 0
- 49: 1
- ...
- ...

127 code points taken.



> qqq

```
ACTTTGTTATATATACTATCTGTATTTTC
CTGGGTGAGAGAGTGGTTGAGAGGGGAA
CCCCCAACCACATTTCCCCACACCCCCTG
ACTTTCCTATATGTCCATTTTTATAATC
```

> ppp

```
TTTTTGGTATCTATTTTTCCACTCATTCTTTAT
TACCCAGTCATCACAAAACACACACAACC
ATTATCTCTAATATATAATTTTACCTTT
```

# Parsing FASTA



```
sequences = []
curseq = ''
for line in file('input.fsa'):
    if line[0] == '>':
        sequences.append(curseq)
    else:
        curseq += line.strip()
sequences.append(curseq)
```

# File Format Examples (II): GenBank



LOCUS SCU49845 5028 bp DNA PLN 21-JUN-1999  
DEFINITION Saccharomyces cerevisiae TCP1-beta gene, partial cds, and A  
(AXL2) and Rev7p (REV7) genes, complete cds.  
ACCESSION U49845  
VERSION U49845.1 GI:1293613  
KEYWORDS .  
SOURCE Saccharomyces cerevisiae (baker's yeast)  
ORGANISM Saccharomyces cerevisiae  
Eukaryota; Fungi; Ascomycota; Saccharomycotina; Saccharomycetes;  
Saccharomycetales; Saccharomycetaceae; Saccharomyces.

...

## ORIGIN

```
1 gatcctccat atacaacggt atctccacct caggtttaga tetcaacaac ggaaccattg
61 ccgacatgag acagttaggt atcgtcgaga gttacaagct aaaacgagca gtagtcagct
121 ctgcatctga agccgctgaa gtttactaa ggggtggataa catcatccgt gcaagaccaa
181 gaaccgcaa tagacaacat atgtaacata tttaggatat acctcgaaaa taataaacg
241 ccacactctc attattataa ttagaacaag aacgcacaaa ttatccacta tataattcaa
```



- Unix: LF (line feed)
- Windows: CRLF (carriage return, line feed)
- (Old Mac OS: CR)

The extra carriage returns will often show up as `^M` in Unix.  
Some unix text files will show up as a single ultra-long line on Windows.

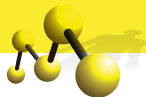
# What About International Characters?



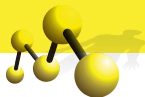
- Such as á or ç?
- Or  $\mu$ ?
- Or Asian characters?
- Or —?

It's a mess!





- Traditional (latin-1,latin-9,latin-15,...)
- Unicode (16-bits, or 32-bits)
- UTF-8



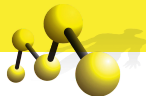
## Unicode

Use 16 bits for (almost) all possible possible characters.

Use 32 bits for all possible characters.

## Byte Order

If you have a 2 byte number, which byte do you write first?



Emerging standard (at some levels).



Emerging standard (at some levels).

You still see errors

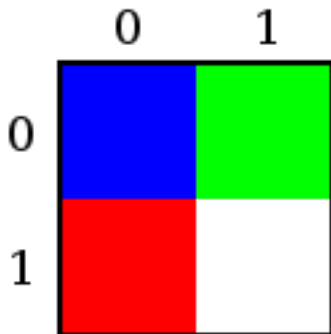
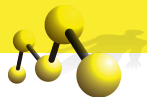
I saw this in an ATM receipt the other day

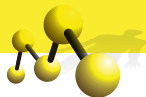
confirmaÃ§Ã£o (which is confirmação in UTF-8, but printed out in Latin-15)

Size	Hex Value	Value	Meaning
2	42 4D	"BM"	Magic Number (66, 77)
4	46 00 00 00	70 Bytes	Size of Bitmap
2	00 00	Unused	Application Specific
2	00 00	Unused	Application Specific
4	36 00 00 00	54 bytes	The offset of data.
4	28 00 00 00	40 bytes	Size of header.
4	02 00 00 00	2 pixels	The width in pixels
4	02 00 00 00	2 pixels	The height in pixels
2	01 00	1 plane	Number of color planes.
2	18 00	24 bits	The bits/pixel.
4	00 00 00 00	0	No compression used
4	10 00 00 00	16 bytes	The size of the raw BMP data
4	13 0B 00 00	2,835 pixels/m	The horizontal resolution
4	13 0B 00 00	2,835 pixels/m	The vertical resolution
4	00 00 00 00	0	Number of colors in the palette
4	00 00 00 00	0	Means all colors are important

Size	Hex Value	Value	Meaning
3	00 00 FF	0 0 255	Red, Pixel (0,1)
3	FF FF FF	255 255 255	White, Pixel (1,1)
2	00 00	0	Padding for 4 bytes/row
3	FF 00 00	255 0 0	Blue, Pixel (0,0)
3	00 FF 00	0 255 0	Green, Pixel (1,0)
2	00 00	0	Padding for 4 bytes/row

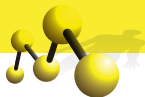
(Wikipedia)



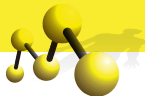


When possible, prefer text formats.  
They are simpler.

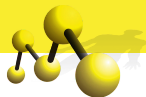




- Text format is not only for text
- Your Python files are text files
- Anything can be represented as a text-file



- Easier to parse
- Easier to debug
- Easier to transfer from one machine to the next
- Larger files (but you can compress them)

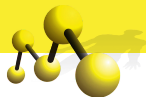


- Comma- or TAB- separated files are flexible
- Can be used to transfer from and to spreadsheet software
- Unfortunately, these are often not 100% well specified

# Example: SAM Format



```
@HD    VN:1.0 SO:unsorted
@SQ    SN:    LN:9732
@PG    ID:bowtie2    PN:bowtie2    VN:2.0.0-beta7
FC5:1101:1441:2131    4    *    0    0    *    *    0    0    TNTCTATT
FC5:1101:1371:2227    4    *    0    0    *    *    0    0    TTGTCTCT
FC5:1101:1650:2171    4    *    0    0    *    *    0    0    GTGTAATT
FC5:1101:1599:2183    4    *    0    0    *    *    0    0    GTATATAC
FC5:1101:1539:2201    4    *    0    0    *    *    0    0    CCTTCTCA
```



FC5:1101:1441:2131\t4\t\*\t0\t0...

# Parsing This File

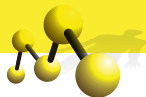


```
infile = open('input.sam')
matched = 0
nreads = 0
for line in infile:
    if line[0] == '@':
        continue
    nreads += 1
    tokens = line.strip().split('\t')
    if tokens[1] != '4': # Four is *unmatched*
        matched += 1

print 'Matched {0} reads out of {1} ({2:.2%})' \
      .format(matched, nreads, matched/float(nreads))
```



- Use UTF-8
- Prefer text-based formats (use generic compression on top)



- Download FastQ file from course webpage
- Plot (for each base pair position) average quality & std. dev.
- Write a sequence trimmer



# Homework I

