分子生物第３講座/構造生物学研究センター　　2009 年 2 年生コンピュータ実習
文責　成田哲博

テキスト訂正

レポートの出し方補足

１　メールの件名

メールの件名は、かならずテキスト 7 ページ目に書いてある通りにしてください。これができていないと、提出を見落とす可能性があります。

２　図の扱い

図を別ファイルにして送ってくる人がいますが、見るのが大変です。全てワードファイルのなかに貼り付けて、ワードファイルだけで完結するようにまとめてください。

３　ワードファイルの署名

提出レポートのワードファイルの先頭に自分の名前と学籍番号を入れておいてください。

４　ワードファイルの拡張子

ワードファイルのファイル名の最後に必ず拡張子.doc をつけてください。コンピュータがワードファイルと認識できません。

５　プログラム名とワードファイル名

これはお願いですが、全員の提出プログラム名が同じだと扱いが難しいことに気がつきました。面倒ですが、プログラム名.py の.py の前に学籍番号を入れておいてください。たとえば、元のファイルが add.py で学籍番号が 60801001 なら、add60801001.py という名前にしてください。また提出ワードファイル名にも同様に、.doc の前に学籍番号を入れてください。たとえば、課題 23.doc であれば、課題 2360801001.doc で良いです。

第一回

p14 ターミナル

実習用コンピュータのターミナル上の emacs(本実習で用いるテキストエディタ)に不具合が見つかったので、ターミナルの代わりに X window を用いることにする。

Dock の中にある X11 のアイコン(白地に X のマーク、前ページ図青矢印) をクリックしてみよう。左上に前ページ図のようなウィンドウ(黒矢印)が一つ立ち上がるはずである。これは xterm と呼ばれる X window アプリケーションで、ターミナルの X window 版である。xterm の扱いはコピーペースト(後述)をのぞいてターミナルとほぼ同じである。アップルキーと N を同時に押すと、別の xterm のウィンドウが立ち上がるのも同じである。

p25 コピー、ペースト補足。

5-1 X window のコピー、ペースト準備

xterm 等の X window アプリケーションでは、マウスの中ボタンでペーストを行うが、初期状態では、この中ボタンは dashboard という別のアプリケーションにわりあてられている。xterm でペーストをするためには、これを解除する必要がある。まず、画面左上のアップルマーク(左上図矢印)をクリック。メニューからシステム環境設定を選択する。

システム環境設定のウィンドウの中のキーボードとマウスをクリックする(右上図矢印)。すると、キーボードとマウスの設定画面がでてくるのでこの画面のマウスタブをクリック。すると次ページ左図のようなマウスの絵がでてくる。青矢印で示した中ボタンの設定は、現在 Dashboard になっている。これをクリックし、次ページ右図のようにボタン 3 に変更する。このウィンドウを閉じて、設定は終了である。

5-2 X window のコピー、ペースト

xtermなどのX windowアプリケーションにおいては、アップルキー+vでペーストはできない。そのかわり、中ボタン(ボタン３)を押すだけでペーストができる。これは、X window以外のMacOSX標準アプリケーション(safariやwordなど)からコピーした内容を、xtermにペーストする場合でも同様である。実習用コンピュータにおいてはマウス中央の小さなボタンが中ボタンで、その左側が左ボタン、右側が右ボタンの役割をする。

また、X windowアプリケーションにおいては、左ボタンでテキストを選択すると、それだけでその内容はクリップボードにコピーされる。この内容は、X windowアプリケーションにペーストするときは中ボタンを押せばペーストできる。しかし、X windowアプリケーションからそれ以外のアプリケーションにペーストしたい場合は、標準通りにアップルキー+cでコピーし、アップルキー+vでペーストする。まとめると、

X window アプリケーション間

左ボタンでテキスト選択のみでコピー、中ボタンでペースト

MacOSX標準アプリケーションからコピー、X window アプリケーションでペースト

アップルキー+cでコピー、中ボタンでペースト

X window アプリケーションからコピー、MacOSX標準アプリケーションにペースト

アップルキー+c でコピー、アップルキー+v でペースト

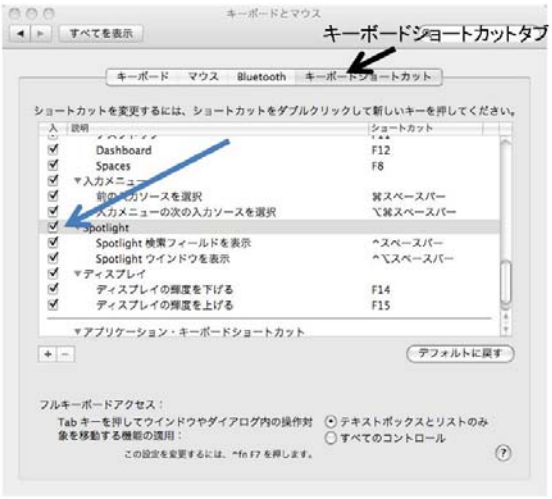
MacOSX 標準アプリケーション間

アップルキー+c でコピー、アップルキー+v でペースト

p25 テキストエディタ

６－０　Emacs 使用準備

Emacs では、文字列のカットにおいて、６－５のように control キーを押しながらスペースキーを押すという動作が必要とされるが、これは MacOSX では Spotlight という別のアプリケーションの起動に割り振られている。このままでは不便なので、MacOSX の設定を変更する。マウス中ボタンの設定を変えたときと同様に、システム環境設定のウィンド



ウの中の”キーボードとマウス”をクリックする。すると、キーボードとマウスの設定画面がでてくるのでキーボードショートカットタブ(左上図)をクリック。キーボードショートカットのリストがでてくるので、スクロールして、Spotlight の項を探す。そのチェックを解除して(右上図)、ウィンドウを閉じれば終了。

６－５　文字列をカット&ペーストする。

カットしたい文字列の最初にカーソルを合わせ、control キーを押しながらスペースキーを押す。次にカットしたい文字列の最後の文字の次の位置にカーソルを合わせ、C-w
カットした内容をペーストする場合は、ペーストしたい場所にカーソルを合わせて、C-y。ただし、挿入場所はマウスポインタがある場所ではなく、カーソルがある場所になる。また、別のウィンドウにカットした内容をペーストすることはできない。そのときは、6-6, 6-7 のコピー&ペーストを行う。

６－６　文字列をコピーする。

xterm 上では、マウスで文字列を選択するだけでよい。

６－７　文字列をペーストする。

xterm 上では中ボタン。ただし、挿入場所はマウスポインタがある場所ではなく、カーソルがある場所になる。

６－１１　特定の行に飛ぶ

テキストに書いた方法は間違い。ESC を押した後、g を二回押す。ミニバッファに Goto line:

と表示されるので、そのあとに、行きたい行番号を入力する。

p28 .cshrc.local 補足

補足１　.で始まるファイルの表示

.tschrc や、.cshrc.local のような”.”(ドット)で始まる名前を持つファイルはオプションなしの ls では表示できない。この場合は、ls の代わりに ls –a を用いる。たとえば、

```
% mkdir test
% cd test
% touch .tcshrc
% ls
%                    #何も表示されない。
% ls –a
.          ..          .tcshrc     #.tcshrc が表示された。
                              #カレントディレクトリ”.”、親ディレクトリ”..”も表示されている。
```

補足２　.cshrc.local の設定がうまくいかないとき

綴り等内容に間違いがない場合、以下の二つのどちらかが原因である場合が多い。

１　.cshrc.local の最後に改行が入っていない。

本文で書くのを忘れていたが、.cshrc.local の最後には改行が必要である。これが入っていないと、最後の行の内容が反映されない。たとえば、以下のように cat で.cshrc.local の中身を表示したときに、

```
% cat .cshrc.local
set PATH=~/com:$PATH%
```

のように中身のすぐあとに改行なくプロンプトが出る場合は、最後に改行が入っていない。この場合は、emacs で.cshrc.local を開き、最後の行の最後にカーソルを持って行って、リターンキーを押して保存すればよい。

```
% cat .cshrc.local
set PATH=~/com:$PATH
%
```

のように、cat で表示した際に、プロンプトの前に改行が入っていれば大丈夫。

2 全角文字が入っている。

一般に UNIX の設定ファイルには全角文字を入れてはいけない。ことえり等漢字変換ソフトが起動したまま入力すると、全角文字として入力される。必ず英数キーを押して、半角文字で入力すること。

第二回

p36 abs

P36 の下から５行目で複素数の絶対値を求める関数として、cmath.abs を挙げたが、

```
>>> cmath.abs(complex)     #絶対値
3.1415926535897931
```

これは、abs の間違い。正しくは、

```
>>> abs(complex)     #絶対値
3.1415926535897931
```

cmath モジュールに abs 関数は無く、標準の abs 関数が複素数もサポートしている。

p40 課題３補足

課題３　文字列

変数 c に”No, I don’t.”, he said. を代入せよ。実際に入力したすべてのコマンドと結果をレポートせよ。

この代入がうまくいった場合、以下のようになる。

```
>>> c
'No, I don\'t.", he said.'   #\が表示される。
>>> print c
No, I don't.", he said.    #\が表示されない。
>>>
```

最初の例では\が表示されるが、print 文で表示すると、\は表示されない。対話モードで、前者のようにただ変数名だけで表示させる場合には、’を入力するために入れた、’の前のバックスラッシュが表示されることがある。print 文で表示される内容が実際に入力された文字列であるとみなしてよい。

p43 sort 訂正

p43 の最後の行

```
>>> b.sort
```

は、

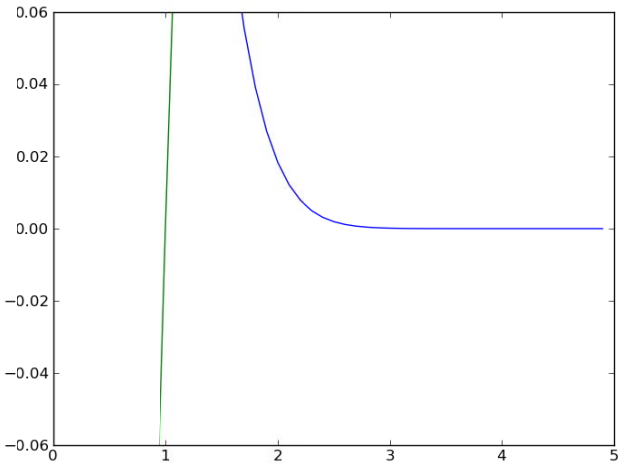
```
>>> b.sort()
```

の間違い。()が無いとうまくいかない。

第三回

pylab.show()

MacOSX の対話モードでは、pylab.show()は必要ないので、**テキスト中の pylab.show()はすべて無視すること**。pylab.show()を入力すると、python に次のコマンドを入力できなくなる。そのときは、グラフが表示されているウィンドウを閉じればよい。



補足:
Linux、windows の対話モードにおいては、pylab.show()を入力しないとグラフは表示されない。この場合もグラフが表示されているウィンドウを閉じないと、python に次のコマンドを入力できない。

p49, p53 pylab.arange(0,5,0.1)

下から 8 行目の
>>> x=pylab.arange(0,5,0.1)
は、
>>> x=pylab.arange(0.1,5,0.1)
のまちがい。そうしないと、結果が p49 の図のようにならず、右のようになる。これは log(0)が負の無限大になってしまうため。P53 の”最初の例をもう一度”の二行目も同様。

p52 a=pylab.array([3,5,6,7])

下から 12 行目の、
>>> a=pylab.array([3,5,6,7])
は、
>>> a=pylab.array([3,4,5,6])
のまちがい。同様に、下から 9 行目
e^3, e^5, e^6,e^7
は、
e^3, e^4, e^5,e^6
のまちがい。下から 6 行目

log(3), log(5), log(6), log(7)は、
log(3), log(4), log(5), log(6)のまちがい。

p55 pylab.xlin([0,0.5])

下から 8 行目　pylab.xlin([0,0.5])は、
pylab.xlim([0,0.5])のまちがい。

第四回

p63 環境変数

第一回では、.cshrc.local に環境変数 PATH を

```
set PATH=~/com:$PATH
```

と書いて設定したが、これだとうまくいかないことがわかった。正しくは、

```
setenv PATH ~/com:$PATH
```

である。申し訳ないが、各自直しておいてほしい。なぜ set だとうまくいかないかについてはここでは詳しく述べないが、set はシェル変数の設定、setenv は環境変数の設定に使われる。知りたい人はシェル変数と環境変数について調べてみること。

p64 add.py 一行目、p65 if.py 一行目など

```
#!/usr/bin/env /python
```

は、全て

```
#!/usr/bin/env python
```

のまちがい。Python プログラムの一行目は常に

```
#!/usr/bin/env python
```

とすること。そうしないとエラーになる。

p71 “True と False, if の真実”の最後の段落

つまり if 文は評価式が 0 であれば真、そうでなければ偽
は、
つまり if 文は評価式が 0 であれば偽、そうでなければ真
のまちがい。

補足１　必ずプログラムを実行可能にすること。

プログラムを最初に実行可能にする際には、かならず hello.py のときのように、
chmod u+x プログラム名
を実行して、実行可能にすること。

補足２　デバッグ追加

Permission denied

権限がない。プログラム実行時にでる場合は、プログラムが実行可能になっていない場合が多い。たとえば、実行可能でないプログラム env.py を実行しようとすると、以下のように、

```
% ./env.py
./env.py: Permission denied.
```

のように、Permission denied が起こる。この場合は、
chmod u+x プログラム名
を実行して、実行可能にすること。
また、プログラム一行目が間違っている場合にも起こる。本来 python プログラムの一行目は、

```
#!/usr/bin/env python
```

にならなくてはならないが、これが、

```
#!/usr/bin env python
```

になっていてりすると、プログラムが実行可能であっても Permission denied が起こる。

No such file or directory

プログラム名が間違っている場合が大半だが、プログラム一行目が間違っている場合にも起こる。たとえば、python プログラム一行目が、

```
#!/usr/bin/env pythn
```

や、

```
#!/usr/bin/env /python
```

になっていたりすると、たとえプログラム名があっていても以下のようにこのエラーになる。env.py の一行目が以上のようなものであれば、

```
% ./env.py
env: /python: No such file or directory
```

のようにエラーになる。この場合は一行目を正しく、

```
#!/usr/bin/env python
```

とすれば良い。

第五回

p89 二行目

$ prime.py 100
は、
% prime.py 100
の間違い

第六回

第六回テキスト補遺、統計学とエラー超入門のダウンロード

今回は平均と標本偏差の計算を行うのだが、その意味を習っていない人もいるようなので、ごく簡単に統計学とエラー扱いの入門テキストを書いた。興味がある人はホームページからダウンロードして読んでほしい。
現在、ホームページの立ち上げ中であるが、立ち上げ中のホームページにアクセスできる。
http://133.6.129.33:8080/Plone
の一番下の講義資料をクリックし、二年生コンピュータ実習をクリック。
第六回補遺統計学とエラー超入門
をクリック。開いたページの第六回補遺統計学とエラー超入門.pdf からダウンロードすること。また、このサイトに、いままでの訂正をまとめた pdf ファイルも置いておくので、必要に応じてダウンロードできる。このサイトはまだ立ち上げ途中なので、停止することがある。毎週木曜日は必ず動かしておくので、必要なダウンロードは木曜日中に行うこと。

P93 copy.py の名前

copy.py というファイルがあると、課題 3 で問題が起こることがわかった。copy.py は copytest.py にすること。

P98 課題 2 空行

データファイルに空行があるとうまくいかない。データの中の空行をなくすこと。

課題 3,4,5

matplotlibrc があるディレクトリからプログラムを実行する。プログラムがホームにない場合は、プログラムがあるディレクトリに以下のようにコピーすればよい。
cp ~/matplotlibrc .
また、課題３、４、５では、グラフの表示のために、最後に pylab.show()を入れないと表示されない。

課題 4　hist.py のダウンロード

前項で示したホームページの二年生コンピュータ実習のページから、、hist.txt をクリックして、開いたページの上のほうにある hist.txt のリンクからダウンロードすること。
テキストの hist.dat は hist.txt のまちがい。

課題 5 改行文字の修正

改行文字は、プログラム中ではどの OS でも\n で統一されているが、その実態は OS ごとに異なる。このことは、Wikipedia の改行コードに詳しく載っている。同じ OS を使っている限りでは問題にならないが、MacOS においては、ver9 以前と ver10 でこの改行文字が異なる。過去の経緯により、本実習用コンピュータは、エクセルやワードなどの MacOSX のアプリケーションが、MacOS9 の改行コードを出力するように特別な設定がなされている。そのため、エクセルが出力した csv ファイルは、xterm やターミナル上ではうまく見えない。これに対処するため、改行文字を MacOSver9 以前のものから、MacOSX 用のものに変更するプログラム textchange.py を書いた。課題 5 をする際には、解析するファイルを、このプログラムで変換してほしい。
MacOS9 以前のテキストファイルの改行文字はプログラム中では、'\r'で表される。このプログラムは、'\r'を探し、'\n'に変換するだけのプログラムである。

textchange.py

```
#!/usr/bin/env python

import sys
infile=sys.argv[1]
outfile=sys.argv[2]
chout=""
fin=open(infile,'r')
```

```
fout=open(outfile,'w')

for line in fin:
    for ch in line:
        if ch=='\r':
            ch='\n'
        chout = chout + ch
fout.write(chout)
fin.close
fout.close
```

使い方は、

% textchange.py 変換元ファイル名　変換先ファイル名

である。たとえば、エクセルで出力したbook.csvを変換し、その結果をbook2.csvに保存したい場合、

% textchange.py book.csv book2.csv

とすれば良い。この場合、解析はbook2.csvに対して行う。

課題５ “課題４の plot3.py”

課題４の plot3.py”は、”課題３の plot3.py”のまちがい。

第七回

P105 def average(inlist)の中最後から二行目

sum=sum/len(i)

は、 sum=sum/len(inlist)のまちがい。

P108 avrmulti.py の実行結果

Avrall: 1.12777777 が正しい結果。

p109 下から 9 行目　set PYTHONPATH=~/pythonlib

これは、第四回 p63 と同様に、

setenv PYTHONPATH ~/pythonlib
の間違いである。

補足: モジュールの検索順

モジュールの検索順は、呼び出し側のプログラムがあるディレクトリ->PYTHONPATH。プログラムがあるディレクトリに liststat.py がある状態で、pythonlib の中の liststat.py だけを書き換えると、プログラムがあるディレクトリの中の古い liststat.py が呼び出される。

第八回

rabit.dat

文中の rabit.dat は rabit.txt の間違い。

課題２

第八回は課題２が抜けていた。課題２は以下の通り。

課題２　タンパク質の分子量

与えられたシークエンスからタンパク質の分子量を計算する、proteinweight.py を作成せよ。各アミノ酸の一文字表記とその分子量は、ホームページ http://133.6.129.33:8080/Plone の講義データから、aminoweight.txt をダウンロードすれば、その中に書いてある。このテキストファイルをプログラムに読み込むようにしても良いし、プログラムの中にこれらの値を書き込んでしまっても良い。ただし、アミノ酸が重合するさいには、水分子(分子量18.02)が抜けることを考慮せよ。

課題３補足

課題３は使用するシークエンスの改行や空白に注意。改行や空白は一文字として数えられるので、水の質量を計算するときに真の値とずれが生じる。たとえば、

ACD
EFG

は、ACD と EFG の間に改行があるので、文字数を数えると７文字になる。プログラム側で改行文字に対応しない場合は、プログラムの最後に

print count

をした場合に、’¥n’: 1 のような表示がでれば要注意。

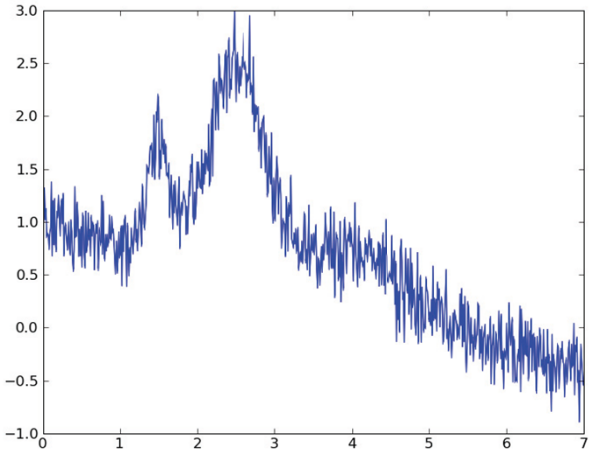
第九回

p130 の式 $\mathbf{f(t) = -A\frac{df}{dt}}$

この式は、$-Af(t) = \frac{df}{dt}$ の間違い。

ガウシアン補足

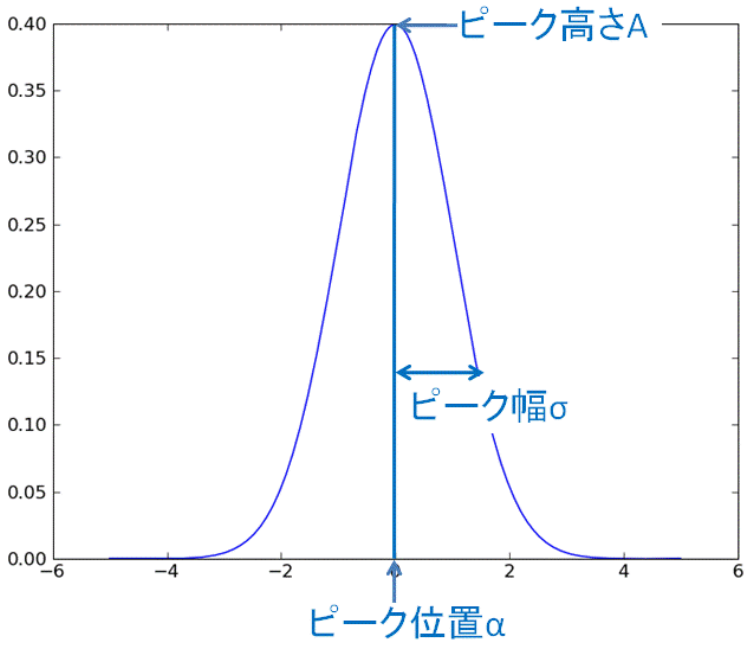
下図のような、ピークが含まれるデータを得ることは、現代生物学において、SDS ページ、カラムクロマトグラフィーなど、非常に多い。このようなピークの位置、大きさ、幅を評価する際、ガウシアンという関数を用いる。



ガウシアンは以下の式で定義される。

$$Ae^{-(\frac{x-\alpha}{\sigma})^2}$$

A はピーク高さ、α はピーク位置、σ はピーク幅を表す。



第十回

p6 2-4 行, p9 1-3 行

[2.29741379 0.8362069]

[[0.0862069 -0.20689656]

[-0.20689656 0.89655173]]

は間違い。p5 にあるように、

[0.8362069,2.29741379]

[[0.89655173, -0.20689656]

[-0.20689656,0.0862069]]

が正しい。