計算機(Linux)使用法

[利用の手引：UNIXコマンド集]

阪大基礎工： 草部浩一・長柄一誠

CMD Workshop (２００８年３月)

```
**************************************************************************
*    Linuxは PC-UNIX と言われるものの一つでコマンドは UNIX と             *
*    99%まで同じです。もちろん FreeBSD等の BSD系 PC-UNIX とも             *
*    同じです。                                                           *
**************************************************************************
```

==========

Linux でのプログラミングはCUIが基本です。画面にコマンドと呼ばれる文字列を打たなければ何も出来ません。まずいくつかの基本コマンドを覚えて、操作になれコマンド集等の本を購入して、使えるコマンドを増やしていくと、自由に UNIX を使いこなせるようになります。MS-Windows はGUIが基本です。CUI ははじめは使いにくいと感じても慣れれば GUI よりずっと早い操作が出来ることがわかりより快適になります。

GUI : Graphical User Interface
CUI : Character (Based) User Interface

==========

ネットワーク環境での UNIX の利用形態

UNIX は開発の初期からネットワーク上にマシンを設置し、便利にリモートからの利用が出来るよう設計されていました。今日の高速ネットの発達により、マシンが実際にどこに設置されているかを殆んど意識せず、高性能マシンを利用出来る環境が整いました。

MS-Windowsは基本的にプログラムは手元のマシン上でのみ実行するよう設計されています。そのためリモートマシンが MS-Windowsで動いている限り、そのマシン以外のコンピュータから接続してプログラムを実行することは基本的には出来ません。これだと科学計算のように高性能マシンを必要とする計算では、

個々のユーザーにそれぞれ高性能マシンを準備する必要があり、
無駄も多く金銭的にも大変です。

現在では多くの計算センターをはじめほとんどの科学計算用マシン
は UNIX のもとで運転されており、リモート実行環境を通し
て多くのマシン上で同時にプログラムを実行させる並列計算等
にも簡単に利用できるようになっています。かつ UNIX コマンド
は世界中同じです。(日本語さえ、扱わなければ世界中操作方法も
同じです。計算で日本語ファイル名を付けたりする人はない
と思いますが。) UNIX は科学計算には必須とも言えます。

まず手元のマシンからリモートの UNIX(Linux) マシンを呼び出
して接続する方法を覚える必要があります。そのためには
ssh, telnet, rlogin 等の命令を使います。これらの命令は

UNIX: kterm (or xterm) の窓を開いて次のように打つ。

```
ssh -X xxx.yyy.osaka-u.ac.jp
telnet xxx.yyy.osaka-u.ac.jp
rlogin xxx.yyy.osaka-u.ac.jp
```

(xxx.yyy の情報は事前に得ておく。-X は X-window が
利用できれば、これを使用する際に付けるオプション)

Windows: teraterm 等のソフトの窓から、同様の命令を送ります。

ユーザー ID とパスワードを聞いてくるので、与えられたそれらを
入力すれば接続が完了します。

注１)kterm, xterm については下の基本コマンド集を参照。
注２)telnet, rlogin はパスワードが暗号化されておらず、通信路での
盗用をさけるため、現在ではこれによるアクセスは、内部ネット
ワーク内等のみでしか許可していないところが多くなっています。
ssh はパスワードが暗号化されています (secure shell)。
==========

==========

コマンドの基本形は

command [-option] [filename]

です。括弧内は必要に応じて付け加える。
全てのコマンドの説明は man コマンドでマニュアルを見れば
説明が書かれている。

(以下の０) - ７) のコマンド群は kterm あるいは xterm の中で
実際に実行してみるとよい。)

最初の窓 kterm (or xterm) はどこかにアイコンがあるので、
それをクリックして開く。開いた窓を閉じるには exit と打つ。

==========

次の UNIX 基本コマンドを覚えて下さい。

*** UNIX 基本コマンド集 ***

０) 入力操作、１文字消去、行末消去：
---
kterm 上でとにかくキーボードを打てば文字が表示される。
消去にはコントロールキー (ctrl) を押えながら d 又は k

ctrl+d　(カーソルの位置の文字を消去)
ctrl+k　(カーソルの以後の文字を消去)

注)Delete キー、← キーも使えるが動作はUNIX の設定による。

１）マニュアルを読む：(manual)

```
man man
man cd
man ls
man mkdir
man mail
man fort77
```

終るには Q をいれる。

２）ディレクトリー(Windows ではフォルダーと 呼んでいる）操作:

ディレクトリーを移動(change directory)

```
cd ..          (一つ上のディレクトリーに移る)
cd             (自分のホームディレクトリーに移る)
cd test1       (ディレクトリー  test1 に移る)
```

注）. は現在のディレクトリーを意味し .. は一つ上を意味する。

ディレクトリーを作る

```
mkdir
```

空のディレクトリーを消す(空でない場合は中を消してから)

```
rmdir
```

３）ファイル操作：

ファイル名一覧(リスト)を見る(list)

```
ls             (短いファイルリスト)
ls -alF        (長いファイルリスト:所有者、作成時間等)
```

ls test2　　（ディレクトリー　test2の中をリスト）

テキストファイルの中身を表示（concatenate）

cat test.f　　（test.f を表示）

注）バイナリーファイル（例えば a.out）を表示して窓が変になった時はその窓の上でマウスの中ボタンを押すとメニューが出る。ターミナル窓のフルリセットを選ぶ。

ファイルをコピー

cp a.out test.exe

ファイル名を変える（move）

mv a.out test.exe　（a.out を test.exe に変える）

ファイルを消す（remove）

rm a.out　　（a.out を消す）

４）アプリケーションの起動する操作：

新しくウインドウ端末を開く

xterm &　　（新しく xterm端末を開く。xtermは英語のみ表示。）

kterm &　　（新しく kterm端末を開く。ktermは漢字も表示。）

注）最後の & は新しい窓を開いて表示するため。

新しくエディターの起動、

gedit &　　（gedit は操作が MS-Windows に似ていて初心者向き。）

emacs &　　（emacs は高機能エディタ、是非これをマスターして下さい。）

vi　　（emacs と並んで利用者の多いエディター、vim は改良版）

emacs と vi の最も簡単な使用法を下に書いておきます。

これらの使用法に関する情報はインターネット上からも
得られますが使いこなすには、本を買う必要があるで
しょう。

新しい窓を開かず(no windowで)、xterm (or kterm)の中で文章を編集するには

```
emacs -nw
```

注1) コマンドの最後の & は正確にはバックグラウンドプロセスとして起動、すなわち今開いている端末の‘‘裏’’で処理という意味。
X-ウインドウ環境(UNIX のウインドウ環境)では別の窓が開く。

注2) エディターとしては xterm (kterm) の画面から emacs (or xemacs)を起動して emacs になれる事を勧める。使い方の本を買う必要があるほど多機能である。また殆んどの UNIX 系マシンに導入されており、かつ種々のコマンドを実行する機能もあるため、いろいろなマシンを使う時違和感がない。
(UNIX 使いになるには是非 emacs か vi エディタを)

注3) プロセスとは動いているプログラムのこと。

注4) バックグラウンドプロセスとは裏で動いているプログラムのこと。バックグラウンドの逆はフォアグラウンドです。今開いている端末で表示や命令が可能なプロセスのことです。

同じように窓を開かず、文章を編集するエディターに vi がある。

```
vi
```

で立ち上げる。最近では殆んどのコンピュータで改良版の vim がインストールされており、vi と打つと普通 vim が立ち上がる。

注）ネットワーク速度が遅い場合、emacs & で新しいウインドウをあげると動作が重たい。このような時 emacs -nw や vi は動作が 軽く使いやすい。

５) 現在動いているプロセス (プログラム) の表示、停止操作：

---

ps	(自分のプロセスのみ表示)
ps -aux	(全プロセスを詳しく表示)
kill 1120	(1120 番のプロセスを止める)
kill -9 1120	(強制的に 1120 番のプロセスを止める)
ctrl+c	(フォアグラウンドプロセスを止める)

注 1) フォアグラウンドプロセスとは表で今動いているプログラムのこと。

ctrl+z	(フォアグラウンドプロセスを一時停止)
bg	(一時停止したプロセスをバックグラウンドで再開)
fg	(バックグラウンドプロセスをフォアグラウンドへ移す)

６) 圧縮ファイルの作成、解凍操作：

UNIX 上でのソフトは通常 tar.gz, .tgz 等の拡張子を持ったファイルで転送されます。

ファイル xxx, あるいはディレクトリー xxx 以下を全て圧縮

```
tar czvf yyy.tgz xxx
```

解凍するには次で xxx 以下のディレクトリーが再現される。

```
tar xzvf yyy.tgz
```

オプション　czvf, xzvf の意味は次のとうり。

(c:create, x: extract, z: zipped, v:verbose, f:file)

注)UNIX の zip は MS-Wondows の zip ファイルと違います
ので MS-Windows の winzip 等では圧縮解凍出来ません。

7) コマンドの間違い訂正操作 (コマンドヒストリー (履歴)):

↑キーを押すと一つ前のコマンドが表示されるので、
修正して CR(改行キー) を押せばよい。前に打ったコマンド
をもう一度打つ必要がなくなる。

==========

*** X-Wondows 環境を快適に使うには ***

X-Window (UNIX のウインドウ) ではウインドウをいくつも
開いて仕事をするのが能率があがる。
例えば以下の作業のように。
fortran では xterm (or kterm) と gedit (or emacs, vi)
を常に開いておいて、gedit (or emacs) の画面で
プログラムを修正しては、xterm の画面でコンパ
イル、実行をする。(ヒストリー機能を使いながら。)
またグラフを書く場合なども、グラフの画面を出した
ままで、kterm の画面や gedit(or emacs, vi) の
画面でデータの修正をしながら、グラフの画面
で再表示する。

==========

==========

*** emacs エディターの最も簡単な使用法 ***

CTRL は Ctrl キー
ESC は Esc キー
SPACEbar はスペースキー（横長の）
+ は２つのキーを同時に押す
--> は順次キーを押す

[起動と終了]

スタート

emacs [file name]&　　(新しいウインドウを立ち上げて
編集開始)
emacs -nw [file name] (今のウインドウ内で開始)
[file name] は無くてもよい。

ファイルへの書き込み

CTRL+s

終了

CTRL+xc

[編集]

文字列の入力修正

任意の位置にカーソルを移動して入力修正

文字列の検索

CTRL+s --> 検索文字列　（繰り返しは CTRL+s）

コピーと切り取り

（始め）CTRL+SPACEbar --> （終り）カーソル移動
--> ESC --> w
（始めと終りの間がコピーされる）
（始め）CTRL+SPACEbar --> （終り）カーソル移動
--> CTRL+w
（始めと終りの間が切り取られる）
このあと次の貼りつけ操作が可能

貼りつけ（コピーまたは切り取り内容を）
カーソル移動 --> CTRL+y
コマンドモードの取消
CTRL+g

==========

*** vi or vim エディターの最も簡単な使用法 ***

３つのモード：vi エディターには テキスト入力モード、
コマンドモード、 ex モードの３つのモードがあります。

これらの間を往ったり来たりして編集するのですが、
今はこのことを気にしないで、ただファイルに文字を
入力したり保存したりする方法と考えて使ってみて
下さい。-->詳しくは本やマニュアルを参照。

ESC は Esc キー
ENTER は Enter キー
--> はキーを順番に押す。

[起動と終了]
スタート
vi [file name]　　（[file name]は無くてもよい）

ファイルへの書き込み（write）
ESC --> : --> w!　　（w file-name でファイル名も
指定可）
終了（quit）
ESC --> : --> q!　　（wq! で書き込み終了）

[編集]
文字列の入力修正（input, append, or open）
カーソル移動 --> i（or a）--> 文字入力
先頭に行を加えるには i --> ENTER

最後尾に行を加えるには o

文字列の検索

ESC --> / --> 検索文字列

文字消去

ESC --> x

行削除

ESC --> dd

単語コピー

ESC --> yw

行コピー（yank）

ESC --> yy

貼りつけ（コピーまたは削除内容の paste）

ESC --> p

ESC --> P

編集内容を捨てて全て元に戻す

ESC --> : --> e!

注）ここで３つのモードについて少し。

１）立ち上げ直後はコマンドモード

２）a: コマンドモードからテキスト入力モードへ
i, a, o 等、　戻るには　ESC

b:コマンドモードから ex モードへ
:　、　戻るには　ENTER

c：テキスト入力モードと ex モードは直接
往き来出来ない。コマンドモードを経由。

３）従って同じモードのコマンドを続けて実行する
場合移行操作ははいらない。

==========

==========

*** fortran プログラムを書き、compile, 実行 するには ***

a) xterm または kterm を起動。
b) gedit& または emacs& でエディターを起動。(別の窓が開く。)
　emacs -nw or vi でもよい。
c) プログラムを書く (あるいは修正する)
d) 書いたプログラムをセーブする。
　(例えば test.f の名前で。または上書き。)
e) xterm の画面で ifort test.f
　(test.f をコンパイル。a.out が出来る。)
　ifort の代わりに fortran77 では f77,g77, fort77 と
　入れるものもある。
f) xterm の画面で ./a.out　　(実行ファイル a.out を実行)

コンパイルエラーがあれば c)　e) を繰り返してエラーをなくす。
実行結果がおかしいときは c)　f) を繰り返して正しい結果にする。

注 1) ファイルに結果を書き込むには ./a.out > outf
　のように打つ。(outf に画面の情報が入る。)
　この操作を"リダイレクション"という。
　"リダイレクション"が使えることは UNIX に
　慣れる基本です。もうひとつ"パイプ" | も UNIX
　使いには必須、例えば ls -aux | less と打って見よ。
注 2) 動き出したプログラムを途中で止めるには ctrl+c
　を押す。
注 3) 最後に & をつけるとバックグラウンド JOB として動く
　から終るまで時間がかかるプログラムはこうして実行
　しながらその間、別のプログラムを作成、編集等出来る。
　バックグラウンド JOB を止めるには ps で番号を調べ
　kill する。
注 4) コンパイルとはコンピュータが理解できる機械語プロ
　グラムに変換すること。

注5) もとになるプログラムの入ったファイルをソースファイル
という。ソースファイルをコンパイルすると実行ファイル
(C 言語や FORTRAN では a.out) が出来る。
但し　a.out は機械語で読めない。

==========

*** グラフを書くには ***

a) kterm & (xterm &) でターミナル窓を新しく開く。
b) fortran 又はエディター (gedit or emacs or vi) で x-軸、
y-軸 のデータを並べて書いたファイル (例えば plot.data)
を作る。
(普通 fortran でデータを書くプログラルを作りコンパイルし
./a.out >plot.data 等でファイルに書く。)
c) kterm & でさらに新しい窓をひらき gnuplot と打って
作図ソフト gnuplot を立ちあげる。
d) そこで　plot "plot.data" と打つと表示される。終了は quit。

eps ファイルを作るには d) に続いて

```
set terminal postscript eps
set output "plot.eps"
plot "plot.data" with lines
```

f)plot.eps を見るには kterm の窓で ghostview plot.eps &
と打つ。印刷するなら print ボタンを押す。
(ghostview は gv でも可。)

注) ファイルに gnuplot の命令を書いておいて自動化する
方法、色々なグラフを重ねる等、gnuplot はとても多機能
です。参考書を一冊持つと役に立ちます。

==========

==========

最後に超簡単例題：

つぎのプログラムをエディターでファイル prog.f90 に
打ち込んで（! 以下は不要です。）コンパイル（ifort prog.f90）
実行（./a.out）してみてください。
input c と出たら好きな数字を打ち込んで下さい。
プログラムと結果を

```
!====== A very simple program prog.f90 ==============
   real::a,b,c,d,e,f,g
   integer::i,j,k

      a=1.0
      b=2.0
   write(*,*) 'a,b=',a,b

   print *,'input c'
   read(*,*) c

      d=a+b+c      ! addition
   write(*,*) 'a,b,c,d=',a,b,c,d

      d=d+a        ! Is this correct?  Yes.
   write(*,*) 'a,d=',a,d

      e=a/c        ! division
      f=a*c        ! multiplication
      g=a**c       ! power
   write(*,*) 'a,c,d,e,f=',a,c,d,e,f

      d=d+a        ! What is the answer? Why?
   write(*,*) 'a,d=',a,d
```

```
   i=5.5
   j=2
   k=i/j        ! What is the answer? Is your answer correct?
write(*,*) 'i,j,k=',i,j,k

   stop
   end
```