The 1st. D-STEP & The 36th DDBJing /

第1回 D-STEP 講習会 & 第36回 DDBJing講習会

2018 (H30) Jan. 26th. 10:00 - 12:00

Takeshi Kawashima / 川島武士 / かわしまたけし

本日の方針

参加者の皆さんには

1. NIG (国立遺伝学研究所)の大型計算機にアクセスし、
2. Unix[系]のコマンドを利用し
3. 塩基配列データを対象とした配列解析を行う

　　ための基礎を身につけてもらいます。

ほとんどの内容は、NIGの大型計算機に特化した内容ではなく、他の多くの大型計算機センターでの利用時に通じるものだと思います。ジョブ管理システム(キューイングシステム)は、NIGの大型計算機で採用しているUGE (Univa Grid Engine)を前提に説明しますが、他の大型計算機センターにおいても概ね同じようなシステムが動いていますので、参考になるはずです。

本日の目標、7項目

関連キーワード

1. **自分が利用している環境について知る** — **NIG Super Computer**
   - ls, lfs, pwd, /usr/local/seq, etc.
2. **キューイングシステムを利用する** — **ジョブ管理システム**
   - qlogin, qsub, qstat, qdel, qreport, etc.
3. **欲しいデータを取得する** — **生物データベース**
   - DDBJ, DRA / ftp, wget, tar, gzip, bzip2, etc.
4. **自分の環境を使いやすくする** — **UNIX**
   - .bashrc, ローカルへのインストール, 環境変数 etc.
5. **マッピング作業** — **Genome Informatics**
   - bowtie2, etc.
6. **キューイングシステム利用のための簡単なBASHプログラミング** — **ジョブ管理システム**
   - bash, アレイジョブ（SGE_TASK_ID）
7. **（時間に余裕があれば）描画のための簡単なRプログラミング** — **Visualize**
   - R

1-1. 自分が利用している環境について知る (1/8)

この辺りに書いてあります

© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

1. 自分が利用している環境について知る

大型計算機へのログインの方法についておさらいし、
Unixの基本的なコマンドを学びます。

1-2-1. スパコンへのログインの方法 (2/8)

```
# 自分のパソコンのターミナルにて
ls                                          #  エルエス
ls -l
# ssh <your-login-id>@gw2.ddbj.nig.ac.jp
```

```
DSTEPMac:TOSHIBA dstep$ ssh dstep@gw2.ddbj.nig.ac.jp
Last login: Thu Jan 11 13:08:03 2018 from xxx.xx.xx.xx
-----------------------------------------------------------------
Thank you for using supercomputer system.
This node is in use for login service only. Please use 'qlogin'.
-----------------------------------------------------------------
[dstep@gw2 ~]$ qlogin
```

自分のパソコンのターミナルに戻る

```
[dstep@gw2 ~]$ exit
[dstep@gw2 ~]$ exit
```

id_rsa.pubの設定がややこしくなってきた時、id_rsaのファイルをあえて指定する

```
# ssh -i <your-login-id>@gw2.ddbj.nig.ac.jp
DSTEPMac:TOSHIBA dstep$ ssh -i ~/.ssh/id_rsa.nig dstep@gw2.ddbj.nig.ac.jp
Enter passphrase for key '/Users/dstep/.ssh/id_rsa.nig':
Last login: Thu Jan 11 13:08:03 2018 from xxx.xx.xx.xx
-----------------------------------------------------------------
Thank you for using supercomputer system.
This node is in use for login service only. Please use 'qlogin'.
-----------------------------------------------------------------
[dstep@gw2 ~]$ qlogin
```

© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

1-2-2. スパコンへのログイン (3/8)

自分のパソコンのターミナルに戻る

```
[dstep@gw2 ~]$ exit
[dstep@gw2 ~]$ exit
```

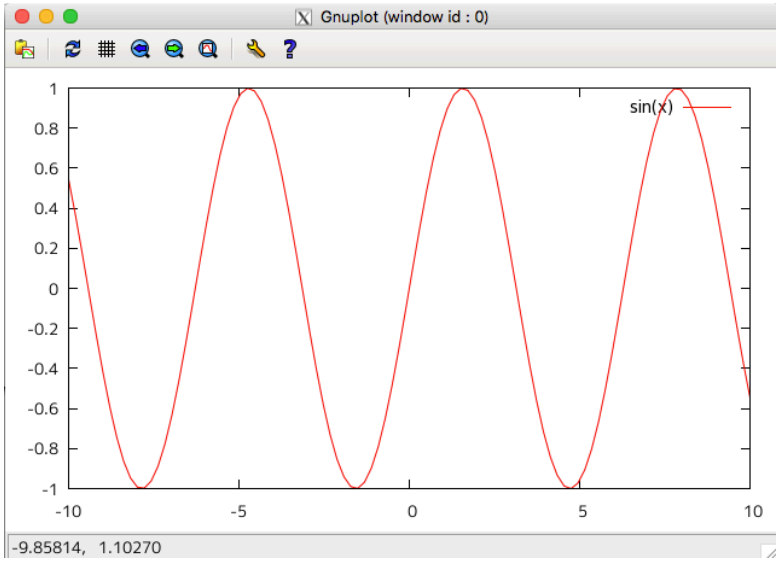
X window システムを自分のターミナルに飛ばす

```
# ssh -X <your-login-id>@gw2.ddbj.nig.ac.jp
# ssh -Y <your-login-id>@gw2.ddbj.nig.ac.jp
```

```
DSTEPMac:TOSHIBA dstep$ ssh -Y dstep@gw2.ddbj.nig.ac.jp
Last login: Thu Jan 11 13:08:03 2018 from xxx.xx.xx.xx
-----------------------------------------------------------------------
Thank you for using supercomputer system.
This node is in use for login service only. Please use 'qlogin'.
-----------------------------------------------------------------------
[dstep@gw2 ~]$ gnuplot
```

ここからはgnuplot

```
[dstep@nt098 ~]$ gnuplot

   G N U P L O T
   Version 4.4 patchlevel 4

Terminal type set to 'wxt'
gnuplot> plot sin(x)

gnuplot> quit()
```

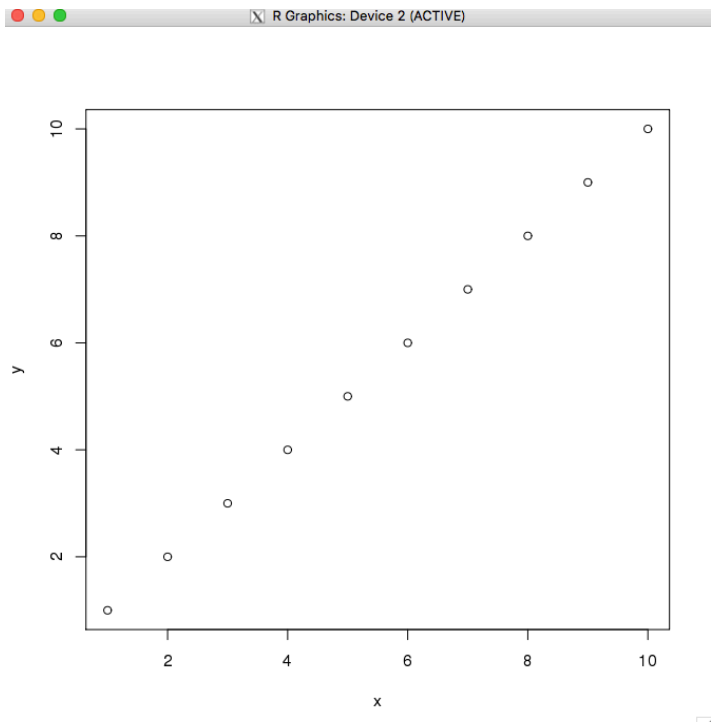


© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

1-2-3. スパコンへのログイン (4/8)

ここからはR

```
[dstep@nt094 ~]$ R

R version 2.14.1 (2011-12-22)

> x <- 1:10
> y <- 1:10
> plot(x, y)

# Rからでる
> q()
Save workspace image? [y/n/c]: n
```



© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

1-3. 利用可能なディスクスペース (5/8)

qloginしたターミナルにて

```
ls
ls -l
ls -l /home
ls -l /home | grep dstep
```

```
[dstep@nt096 ~]$ ls -l /home |grep dstep
lrwxrwxrwx 1 dstep        ma-nig          22  4月  8 10:43 2016 dstep -> /
lustre3/home/dstep
[dstep@nt096 ~]$
```

この場合、自分のホームディレクトリが /lustre3 にあることがわかる。

なおphase2の利用ユーザーは、lustre1からlustre5までのどれかに振り分けられている。

```
[dstep@nt096 ~]$ lfs quota -u dstep /lustre3
Disk quotas for user dstep (uid 3907):
     Filesystem  kbytes   quota   limit   grace   files   quota   limit   grace
       /lustre3 200000000       0 1000000000       - 7000000       0       0       -
```

この場合、自分には /lustre3の元で1000000000 kbyte (=1Tbyte)割り当てられており

そのうち200000000 kbyte (=200Gbyte)利用していることが分かる。

© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

1-4. 環境変数 (6/8)

```
# qloginしたターミナルにて
echo “TEST”

printenv
printenv | wc
printenv | sed -e “s/=.*//"

echo $HOME
echo $SHELL
echo $PWD
echo $USER
echo $PATH

time
which time
```

```
[dstep@nt098 ~]$ which time
/usr/bin/time
```

time コマンドは　/usr/binの下においてある

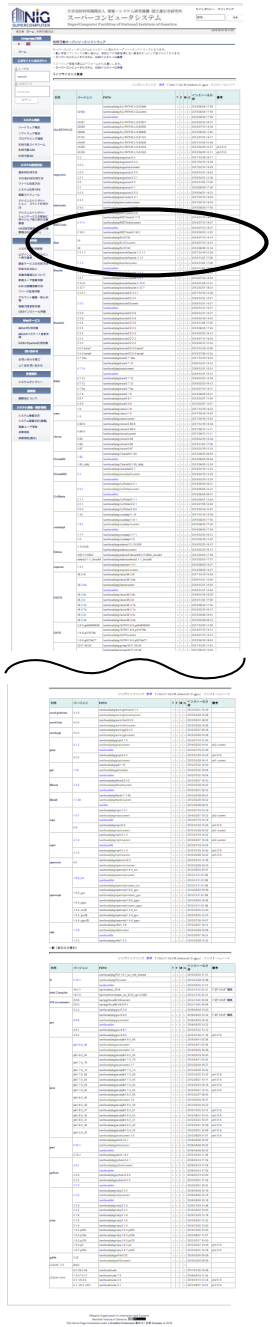
```
echo $PATH
```

/usr/binというディレクトリは確かに環境変数PATHの中に記載されている

```
[dstep@nt098 ~]$ echo $PATH
/home/geadmin2/UGER/bin/lx-amd64:/usr/lib64/qt-3.3/bin:/opt/pgi/linux86-64/current/bin:/
usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/pkg/java/current/bin:/opt/intel/xe_2016/
compilers_and_libraries_2016.1.150/linux/bin/intel64:/opt/intel/xe_2016/
compilers_and_libraries_2016.1.150/linux/mpi/intel64/bin:/opt/intel/xe_2016/debugger_2016/
gdb/intel64_mic/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/opt/
ibutils/bin:/opt/intel/itac/8.1.3.037/bin:/home/dstep/local/bin:/home/dstep/bin
```

© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

1-5. 利用可能OSS (7/8)



	0.5.9	/usr/local/pkg/bwa/0.5.9	○	○	○	○	2012/02/07 11:07
canu	1.6	/usr/local/pkg/canu/v1.6	○	○	○	○	2017/10/19 14:22
	1.5	/usr/local/pkg/canu/v1.5	○	○	○	○	2017/05/18 15:42
		/usr/local/pkg/canu/current	○	○	○	○	2017/05/18 15:42
	0.69-6	/usr/local/pkg/circos/0.69-6	○	○	○	○	2017/10/19 14:36

/usr/local/pkg/canu/current

```
[dstep@nt094 ~]$ /usr/local/pkg/canu/current/bin/canu --version
```

```
Canu snapshot v1.5 +43 changes (r8243 f05c81531c19887459ca53ac3702509ee70eac22)
```

full pathを打たないで使えるようにしたい

```
[dstep@nt094 ~]$ canu
```

```
-bash: canu: コマンドが見つかりません
```

環境変数 PATHを設定することで、利用可能になります

© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

1-6. 自分が利用している環境について知る (8/8)

ここまでで、

適切にログインし、
自分の環境を確認し、
適切なプログラムへPATHを通す、ことができるようになりました。

© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

2. キューイングシステムを利用する

大型計算機利用時に特有のジョブ管理システム（キューイングシステム）
について学びます。

ただし**最低限**知っていなければならないことは、すでに1.で学んだように、
ログイン時にすぐqloginコマンドを打つことです。

とはいえ、ゲノム解析などを実際に行うには、**キューイングシステム**について、
正しい理解が必要です。ここでは実際にqsubコマンドを利用してみましょう。

2. キューイングシステムを利用する　(1/6)

qlogin, qsub, qstat, qdel, qreport, etc.

qsub経由で利用するプログラムを作成する

参考用に、time.qsubというプログラムを作成しました。

下記リンクから取得してください。

(gitを普段利用している方へ：

ここでは説明のため、あえてgitの機能を利用せずにwgetで取得しています。)

```
wget https://raw.githubusercontent.com/tkwsm/DSTEP20180126/master/time.qsub
wget https://raw.githubusercontent.com/tkwsm/DSTEP20180126/master/time.sh
```

```
[dstep@nt096 tmp]$ wget https://raw.githubusercontent.com/tkwsm/DSTEP20180126/master/time.qsub

--2018-01-11 16:55:38--  https://raw.githubusercontent.com/tkwsm/DSTEP20180126/master/time.qsub
100%[=====================================>] 236         --.-K/s 時間 0s

2018-01-11 16:55:38 (24.8 MB/s) - `time.qsub' へ保存完了 [236/236]
```

gitを普段利用されている方は下記の方が楽です。

```
DSTEPMac:tmp dstep$ git clone git@github.com:tkwsm/DSTEP20180126.git
Cloning into 'DSTEP20180126'...

Checking connectivity... done.

DSTEPMac:tmp dstep$ ls
DSTEP20180126
DSTEPMac:tmp dstep$ cd DSTEP20180126/
DSTEPMac:DSTEP20180126 dstep$ ls
README.md time.qsub time.sh
```

2. キューイングシステムを利用する　(2/6)

time.sh というプログラムが取得できたかどうかをlsで確認する

```
[dstep@nt096 tmp]$ ls
time.sh
```

time.sh というプログラムの中身をcatで確認する

```
[dstep@nt096 tmp]$ cat time.sh
```

```
#!/bin/sh

echo "start program"
echo "start sleep 15 sec"
sleep 15
echo "end sleep"
echo "program done"
```

→ # shebang行

→ # プログラム本体

→ # time.sh
というプログラムの中身

time.sh というプログラムを実行してみる

```
[dstep@nt094 dstep20180126]$ bash time.sh
start program
start sleep 15 sec
end sleep
program done
[dstep@nt094 dstep20180126]$
```

2. キューイングシステムを利用する　(3/6)

time.qsub というプログラムが取得できたかどうかをlsで確認する

```
[dstep@nt096 tmp]$ ls
time.qsub
```

time.qsub というプログラムの中身をcatで確認する

```
[dstep@nt096 tmp]$ cat time.qsub
```

```
#!/bin/sh
#$ -S /bin/sh
#$ -cwd
#$ -l s_vmem=1G,mem_req=1G
#$ -l short
### #$ -l debug
#$ -pe def_slot 1
#$ -o ./
#$ -e ./

source ~/.bashrc

echo "start program"
echo "start sleep 15 sec"
sleep 15
echo "end sleep"
echo "program done”
```

→ # shebang行

→ # qsubコマンド設定

→ # 普段利用している
設定ファイルの読み込み

→ # プログラム本体

→ # time.qsub
というプログラムの中身

```
[dstep@nt096 tmp]$
```

2. キューイングシステムを利用する　(4/6)

time.qsub というプログラムを実行してみる

```
[dstep@nt094 dstep20180126]$ qsub time.qsub
Your job 10384431 ("time.qsub") has been submitted

[dstep@nt094 dstep20180126]$ qstat

job-ID     prior   name       user         state submit/start at     queue                          jclass           slots ja-task-ID
-----------------------------------------------------------------------------------------------------------------------------------
  10384220 0.25036 QLOGIN     dstep        r     01/16/2018 11:36:44 login.q@nt094i                                                1
  10384431 0.00000 time.qsub  dstep        qw    01/16/2018 12:35:25                                                               1

[dstep@nt094 dstep20180126]$ qstat
job-ID     prior   name       user         state submit/start at     queue                          jclass           slots ja-task-ID
-----------------------------------------------------------------------------------------------------------------------------------
  10384220 0.25040 QLOGIN     dstep        r     01/16/2018 11:36:44 login.q@nt094i                                                1

[dstep@nt094 dstep20180126]$

[dstep@nt094 dstep20180126]$ ls

time.sh
time.qsub

time.qsub.e10384431
time.qsub.pe10384431
time.qsub.o10384431
time.qsub.po10384431
```

→ # 標準出力と標準エラー出力

2. キューイングシステムを利用する　(5/6)

time.qsub というプログラムの中身をcatで確認する

```
[dstep@nt096 tmp]$ cat time.qsub
```

```
#!/bin/sh
#$ -S /bin/sh
#$ -cwd
#$ -l s_vmem=1G,mem_req=1G
#$ -l short
### #$ -l debug
#$ -pe def_slot 1
#$ -o ./
#$ -e ./

source ~/.bashrc

echo "start program"
echo "start sleep 15 sec"
sleep 15
echo "end sleep"
echo "program done”
```

→ # shebang行

→ # qsubコマンド設定

→ # time.qsub
というプログラムの中身

利用可能バイオツール
利用可能OSS
利用可能DB

システム使用方法

基本的利用方法
その他UGE利用方法
ファイル転送方法
システム利用TIPS
稼働スケジュール
アドバンスドリザベーション　コマンド利用方法
アドバンスドリザベーション

計算ジョブの投入

計算ノードに計算ジョブを投入するには、qsubコマンドを利用します。qsubコマンドでジョブを投入する為には、投入するジョブスクリプトを作成する 必要が有ります。簡単な例ですが以下のように記述します。

```
#!/bin/sh
#$ -S /bin/sh
pwd
hostname
date
sleep 20
date
echo  “to stderr”  1>&2
```

この時、行の先頭に、"#$"が指定されている行が、UGEへのオプション指示行になります。オプション指示行をシェルスクリプトにする、またはqsub コマンド 実行時のオプションとして指定することでUGEに動作を指示します。主なオプションとしては以下のものがあります。

指示行の記述	コマンドラインオプション指示	指示の意味
#$ -S インタプリタパス	-S インタプリタパス	コマンドインタプリタをパス指定する。シェル以外のスクリプト言語も指定可能。このオプションは指定必要
#$ -cwd	#$ -cwd	ジョブ実行のカレントワーキングディレクトリを指定する。これによりジョブの標準出力、エラー出力もcwd上に出力される。 指定しない場合はホームディレクトリがカレントワーキングディレクトリとなりジョブ実行される。
#$ -N ジョブ名	-N ジョブ名	ジョブの名前を指定する。この指定が無ければスクリプト名がジョブ名になる。
#$ -o ファイルパス	-o ファイルパス	ジョブの標準出力の出力先を指定する。
#$ -e ファイルパス	-e ファイルパス	ジョブの標準エラー出力の出力先を指定する。

qsubには上記の他にも指定可能なオプションが多数ありますので、詳細についてはシステムにログイン後"man qsub"として、qsubコマンドの オンラインマニュアルを参照して確認してください。

NIG スパコンのシステム使用方法のあたりに詳しく書いてあります。

2. キューイングシステムを利用する　(6/6)

ここまでで、

適切にログインし、
UGEキューイングシステム(qsubコマンドなど)を使って、
計算機を利用できるようになりました。

3. 欲しいデータを取得する

大型計算機によるゲノム解析の特徴の一つは、事前に大量の生物学データを自分の計算機環境にアップロードする必要があることです。そのような生物学データは多くの場合、既存の配列データベースに登録されているものや利用者自身が取得したNGS等による配列データでしょう。NIGの大型計算機では、主要な生物学データベースがすでに相当数整備され、常時アップデートされています。これらのデータベースの利用方法を学びましょう。またNIGの大型計算機に整備されていないデータベースについてはご自身で最新のデータをダウンロードしてこなくてはなりません。そのために必要な知識としてftp, wgetなどのコマンドの利用方法を学びましょう。

3-1. NIGのスパコンにすでに整備してあるデータベースを利用する

3-2. その他のゲノムの基礎データの登録サイト

3-2. DRAから取得

3. 欲しいデータを取得する

NIGのスパコンにすでに整備してあるデータベースを利用する

スーパーコンピュータシステム　利用可能ＤＢ

スーパーコンピュータシステムでは、各計算ノード、各ログインノードから各種バイオ系DBが利用可能です。

1.DDBJ,NCBI,EBI等の公共DBを利用したい場合

スーパーコンピュータシステムにて利用可能なDBおよびパスは 利用可能DB一覧をご覧下さい。

2.DRAを含むその他のDDBJ DBを利用したい場合

上記利用可能DB以外のDDBJ DBについては下記方法にてデータをコピーしてご利用下さい。

利用可能DB一覧

DB名	パス (/usr/local/seq/)		設置されているファイルの詳細	更新頻度
DDBJ-unified-all	-	ddbj-unified-all/	-	毎日
	fasta/		ftp://ftp.ddbj.nig.ac.jp/ddbj_database/ddbjnew/unified-all/fasta/以下を解凍したFASTA形式ファイル	
	blast/		ftp://ftp.ddbj.nig.ac.jp/ddbj_database/ddbjnew/unified-all/blastdb/以下を解凍したBLASTデータベース	
DDBJ-unified-new	-	ddbj-unified-new/	-	毎日
	fasta/		ftp://ftp.ddbj.nig.ac.jp/ddbj_database/ddbjnew/unified-new/fasta/以下を解凍したFASTA形式ファイル	
	blast/		ftp://ftp.ddbj.nig.ac.jp/ddbj_database/ddbjnew/unified-new/blastdb/以下を解凍したBLASTデータベース	
GenBank	flat/	genbank/	ftp://ftp.ncbi.nih.gov/genbank/以下の*.seqファイル、およびWGSファイル	随時
	fasta/		上記のFASTA形式ファイル	
	blast/		上記FASTA形式ファイルのBLASTデータベース	
GenBank-UPD	flat/	genbank-upd/	ftp://ftp.ncbi.nih.gov/genbank/daily-nc/以下の全ファイル	毎日
	fasta/		上記ファイルの内*.flatのFASTA形式ファイル	
	blast/		上記FASTA形式ファイルのBLASTデータベース	
GenPept	-	genpept/	-	随時
	fasta/		GenBankのseqファイル内の翻訳可能なエントリを翻訳したFASTA形式ファイル	
	blast/		上記FASTA形式ファイルのBLASTデータベース	
GenPept-UPD	-	genpept-upd/	-	毎日
	fasta/		GenBank-UPDの*.flatファイル内の翻訳可能なエントリを翻訳したFASTA形式ファイル	
	blast/		上記FASTA形式ファイルのBLASTデータベース	

3. 欲しいデータを取得する

NIGのスパコンにすでに整備してあるデータベースを利用する

```
[dstep@nt098 dstep20180126]$ ls /usr/local/seq/
blast  chemicaldb  entity  fasta  flat  igenome  old  taxonomy  work
[dstep@nt098 dstep20180126]$ ls /usr/local/seq/blast
ddbj-unified-all  embl      genbank      genpept      ncbi    refseq-upd
ddbj-unified-new  embl-upd  genbank-upd  genpept-upd  refseq  uniprot
```

```
[dstep@nt098 dstep20180126]$ ls
hemoglobin_b.fa
DRR016430.fastq  README.md  time.qsub  time.sh
```

```
[dstep@nt098 dstep20180126]$ cat hemoglobin_b.fa
>NP_000509.1 hemoglobin subunit beta [Homo sapiens]
MVHLTPEEKSAVTALWGKVNVDEVGGEALGRLLVVYPWTQRFFESFGDLSTPDAVMGNPKVKAHGKKVLG
AFSDGLAHLDNLKGTFATLSELHCDKLHVDPENFRLLGNVLVCVLAHHFGKEFTPPVQAAYQKVVAGVAN
ALAHKYH
```

```
# blastp -query hemoglobin_b.fa
#        -db /usr/local/seq/blast/uniprot/swissprot
#        -num_alignments 1
#        -outfmt 6
```

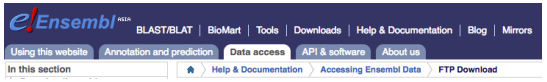
```
[dstep@nt098 dstep20180126]$ blastp -query hemoglobin_b.fa -db /usr/local/seq/blast/uniprot/
swissprot -num_alignments 1 -outfmt 6
```

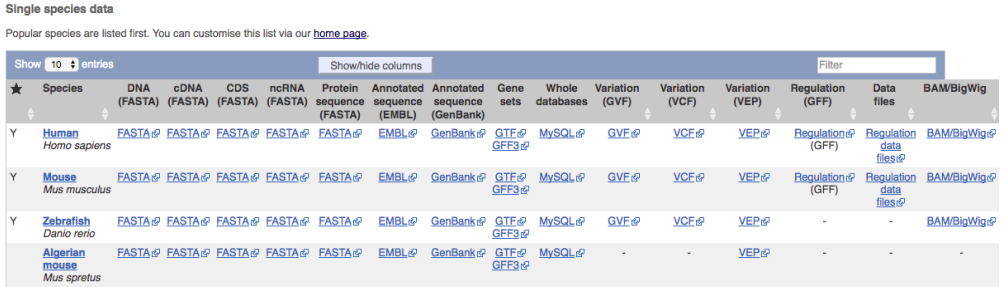
```
NP_000509.1  sp|P68873|HBB_PANTR100.00014700114711473.25e-104301
```

3. 欲しいデータを取得する

3-3. その他のゲノムの基礎データの登録サイト (オススメいくつか)

https://asia.ensembl.org/index.html





★	Species	DNA (FASTA)	cDNA (FASTA)	CDS (FASTA)	ncRNA (FASTA)	Protein sequence (FASTA)	Annotated sequence (EMBL)	Annotated sequence (GenBank)	Gene sets	Whole databases	Variation (GVF)	Variation (VCF)	Variation (VEP)	Regulation (GFF)	Data files	BAM/BigWig
Y	Human *Homo sapiens*	FASTA	FASTA	FASTA	FASTA	FASTA	EMBL	GenBank	GTF GFF3	MySQL	GVF	VCF	VEP	Regulation (GFF)	Regulation data files	BAM/BigWig
Y	Mouse *Mus musculus*	FASTA	FASTA	FASTA	FASTA	FASTA	EMBL	GenBank	GTF GFF3	MySQL	GVF	VCF	VEP	Regulation (GFF)	Regulation data files	BAM/BigWig
Y	Zebrafish *Danio rerio*	FASTA	FASTA	FASTA	FASTA	FASTA	EMBL	GenBank	GTF GFF3	MySQL	GVF	VCF	VEP	-	-	BAM/BigWig
	Algerian mouse *Mus spretus*	FASTA	FASTA	FASTA	FASTA	FASTA	EMBL	GenBank	GTF GFF3	MySQL	-	-	VEP	-	-	-

https://phytozome.jgi.doe.gov/pz/portal.html

https://gold.jgi.doe.gov/

http://www.genome.jp/kegg/catalog/org_list.html

3. 欲しいデータを取得する (ftp経由)

EnsemblからcDNAファイルを取得する方法を試しましょう

https://asia.ensembl.org/info/data/ftp/index.html

Drosophila

Click!!

3. 欲しいデータを取得する (ftp経由)

EnsemblからanonymousFTPで、cDNAファイルを取得する方法を試しましょう

ftp://ftp.ensembl.org/pub/release-91/fasta/drosophila_melanogaster/cdna/ の一覧

上位のディレクトリーへ移動

名前	サイズ	最終更新日時
CHECKSUMS	1 KB	2017/11/29 13:33:00 JST
Drosophila_melanogaster.BDGP6.cdna.all.fa.gz	16973 KB	2017/11/20 23:37:00 JST
README	3 KB	2017/11/20 23:37:00 JST

ftpコマンド

```
[dstep@nt093 dat]$ ftp -i ftp.ensembl.org
Name (ftp.ensembl.org:dstep): anonymous
331 Please specify the password.
Password: <your-email-address>
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd pub/release-91/fasta/drosophila_melanogaster/cdna
ftp> ls
227 Entering Passive Mode (193,62,193,8,246,104)
150 Here comes the directory listing.
-rw-r--r--    1 ftp      ftp            76 Nov 29 13:33 CHECKSUMS
-rwxrwxr-x    1 ftp      ftp      17379879 Nov 20 23:37 Drosophila_melanogaster.BDGP6.cdna.all.fa.gz
-rwxrwxr-x    1 ftp      ftp          2521 Nov 20 23:37 README
226 Directory send OK.
ftp> get Drosophila_melanogaster.BDGP6.cdna.all.fa.gz
local: Drosophila_melanogaster.BDGP6.cdna.all.fa.gz remote: Drosophila_melanogaster.BDGP6.cdna.all.fa.gz
227 Entering Passive Mode (193,62,193,8,94,215)
150 Opening BINARY mode data connection for Drosophila_melanogaster.BDGP6.cdna.all.fa.gz (17379879 bytes).
226 Transfer complete.
17379879 bytes received in 5.3 seconds (3.2e+03 Kbytes/s)
ftp> bye
221 Goodbye.
[dstep@nt093 dat]$
```

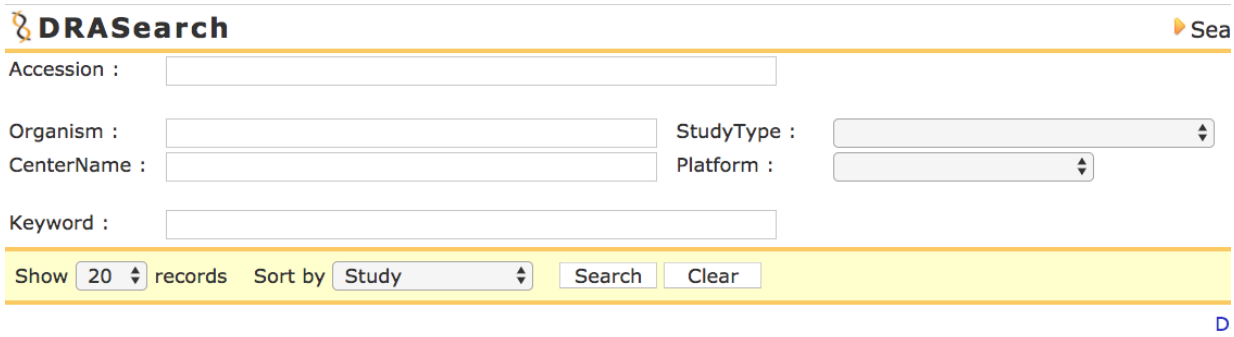
3. 欲しいデータを取得する (ftp経由)

```
ftp> bye
221 Goodbye.
[dstep@nt093 dat]$

[dstep@nt093 dat]$ ls
DRR016430.fastq  Drosophila_melanogaster.BDGP6.cdna.all.fa.gz

[dstep@nt093 dat]$ gunzip Drosophila_melanogaster.BDGP6.cdna.all.fa.gz
[dstep@nt093 dat]$ ls
DRR016430.fastq  Drosophila_melanogaster.BDGP6.cdna.all.fa
```

今回は拡張子が *.gzだったので、gunzipで解凍しました。
拡張子が*bz2だった場合は、bunzip2で解凍しましょう

3. 欲しいデータを取得する (DRA経由)

http://trace.ddbj.nig.ac.jp/dra/index.html



3. 欲しいデータを取得する (DRAから)

danio rerio

Transcriptome Analysis

ILLUMINA

DRASearch

Search Home | DRA Home

Accession :

Organism : danio rerio

StudyType : Transcriptome Analysis

CenterName :

Platform : ILLUMINA

Keyword :

Show 20 records Sort by Study Search Clear

Search Results (262 studies)

<< < 1 / 14 Page >

#	STUDY	SUBMISSION	STUDY_TITLE	STUDY_TYPE	ORGANISM
1	ERP000016	ERA000092	Sequencing the Zebrafish transcriptome form a range of tissues and developmental stages using the Illumina Genome Analyzer	Transcriptome Analysis	Danio rerio

DRASearch

Search Home | DRA Home

ERP000016

Study Detail	
Title	Sequencing the Zebrafish transcriptome form a range of tissues and developmental stages using the Illumina Genome Analyzer
Study Type	Transcriptome Analysis
Abstract	Paired-end sequence data has been generated using polyA selected RNA from a range of zebrafish tissues and developmental stages using the Illumina Genome Analyzer. These data have been used to improve the gene annotation of the zebrafish genome
Description	Zebrafish total RNA was extracted from embryonic and adult tissue, then polyA selected. After fragmentation and reverse transcription Illumina sequencing libraries were prepared. Paired-end sequence runs were performed with 36, 37, 54 and 76 base reads on the Illumina Genome Analyzer.. This data is .. [more]
Center Name	SC

Navigation			
Submission	ERA000092	FTP	
Experiment	ERX000398	FASTQ	SRA
	ERX000399	FASTQ	SRA
	ERX000400	FASTQ	SRA
	ERX000401	FASTQ	SRA
	ERX000402	FASTQ	SRA
	ERX000403	FASTQ	SRA
	ERX000404	FASTQ	SRA
	ERX000405	FASTQ	SRA
	ERX000406	FASTQ	SRA
	ERX000407	FASTQ	SRA
	ERX000408	FASTQ	SRA
	ERX000409	FASTQ	SRA

"ftp://ftp.ddbj.nig.ac.jp/ddbj...0092/ERX000398/"へ接続中...

サーバ"ftp.ddbj.nig.ac.jp"用の名前とパスワードを入力してください。

ユーザの種類: ゲスト / 登録ユーザ

キャンセル 接続

ERX000398

ERR003990_1.fastq.bz2 ERR003990_2.fastq.bz2 ERR003990.fastq.bz2

ERR003990

ERR003990.sra

3. 欲しいデータを取得する (DRAから)

Control key を押しながらクリック

ftp://ftp.ddbj.nig.ac.jp/ddbj_database/dra/fastq/ERA000/ERA000092/ERX000398/ERR003990_1.fastq.bz2

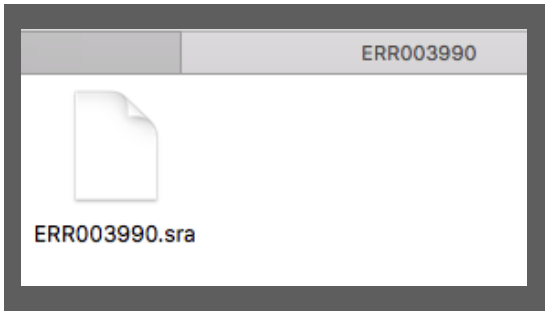
3. 欲しいデータを取得する (DRAから)

後の作業のために下記4つのファイルをダウンロードしておく

```
[dstep@nt094 dstep20180126]$ wget ftp://ftp.ddbj.nig.ac.jp/ddbj_database/dra/fastq/ERA000/ERA000092/ERX000398/ERR003991_1.fastq.bz2
[dstep@nt094 dstep20180126]$ wget ftp://ftp.ddbj.nig.ac.jp/ddbj_database/dra/fastq/ERA000/ERA000092/ERX000398/ERR003991_2.fastq.bz2
[dstep@nt094 dstep20180126]$ wget ftp://ftp.ddbj.nig.ac.jp/ddbj_database/dra/fastq/ERA000/ERA000092/ERX000398/ERR003992_1.fastq.bz2
[dstep@nt094 dstep20180126]$ wget ftp://ftp.ddbj.nig.ac.jp/ddbj_database/dra/fastq/ERA000/ERA000092/ERX000398/ERR003992_2.fastq.bz2
[dstep@nt094 dstep20180126]$ bunzip2 ERR00399*bz2
```

<- 時間がかかる

3. 欲しいデータを取得する (DRAから)



fastq-dump ERR003990.sra

```
/usr/local/pkg/sra-tools/2.5.7/bin/fastq-dump.2.5.7 --split-files ERR003990.sra
```
<- やってはいけない

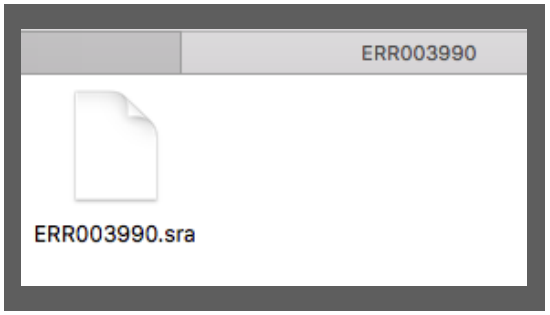
qsub コマンドを利用する。必ず！！

```
[dstep@nt094 dstep20180126]$ cat sra2fastq.sh
```
<- qsub用のスクリプトを作成して、

```
#!/bin/sh
#$ -S /bin/bash
#$ -cwd
#$ -l short
#$ -l s_vmem=1G,mem_req=1G
#$ -o ./
#$ -e ./

source ~/.bashrc

FASTQDUMP="/usr/local/pkg/sra-tools/2.5.7/bin/fastq-dump.2.5.7"
Dbdir="/home/dstep/tanomare/dstep20180126"

$FASTQDUMP --split-files ERR003990.sra
```

```
[dstep@nt094 dstep20180126]$ qsub sra2fastq.qsub
```
<- qsubコマンドを利用して計算機に投入

3. 欲しいデータを取得する (DRAから)



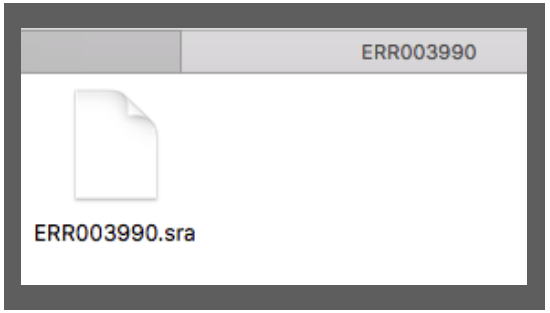

```
[dstep@nt094 dstep20180126]$ qsub sra2fastq.qsub
```

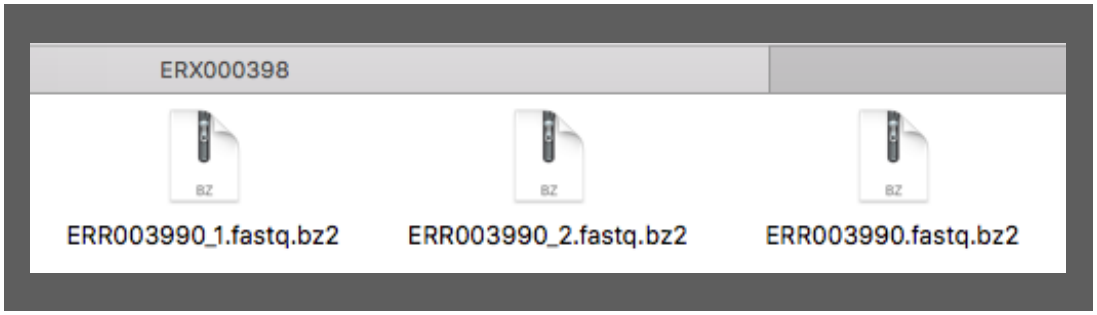
<- qsubコマンドを利用して計算機に投入

```
[dstep@nt094 dstep20180126]$ ls ERR*
ERR003990.sra  ERR003990_1.fastq  ERR003990_2.fastq
```

SRAファイルから、ペアエンドのfastqファイルが取得できた

3. 欲しいデータを取得する (DRAから)






```
[dstep@nt094 dstep20180126]$ ls ERR*
ERR003990.sra  ERR003990_1.fastq  ERR003990_2.fastq
```

もともとdraに登録されていた
fastqファイル

*.sraファイルから自分でfastq-dump
を利用してfastqを作成した

両者は少し違う！

↓

*.sraから作成したfastqファイルは
クオリティー等でtrimingされていない(ことが多い)

↓

NGSデータを利用する場合は、
低クオリティー部分やアダプター等の除去してから使いましょう
(この後fastQCやTrimmomaticの利用を体験しましょう。)

3. 欲しいデータを取得する

以上で、

3-1. NIGのスパコンにすでに整備してあるデータベースを利用する

3-2. その他のゲノムの基礎データの登録サイト (いくつか)

3-3. DRAから取得

これらの典型的なデータの取得方法を学びました。

Webブラウザからクリックしてダウンロードする場合、

wgetやcurlコマンドを利用する場合、

ftpコマンドを利用する場合、などがありました。

またDRAやSRAに特徴的なこととしてfastq-dumpの利用方法も学びました。

なお、DRAから *.sraファイルを取得すると、cfq フォーマット(solidのcolor space)の配列という場合があります。cfqからfqへの変換方法を知っておきたい方ははおられますか？

4. 自分の環境を使いやすくする

UNIX系の計算機環境では、自分の利用している環境を使いやすくするために、環境変数や各種設定ファイル の編集方法を学んでいなくてはなりません。本来最初に学ぶべきこれらの事柄をここまでは避けて説明してきました。

NIGの大型計算機には主要なゲノム解析様のプログラムがかなりインストールされていますが、これらを利用するためにも環境変数について学ばなければなりません。

また利用者が利用したいプログラムがインストールされていない場合は、利用者ご自身でご自身の計算機環境にそのプログラムをインストールする必要があるでしょう。

4-1. 環境変数について学びましょう

4-2. ローカルに好みのプログラムをインストールすることを学びましょう。

4. 1-4. 環境変数 (6/8)のおさらい (2/7)

```
# qloginしたターミナルにて
echo “TEST”

printenv
printenv | wc
printenv | sed -e “s/=.*//"

echo $HOME
echo $SHELL
echo $PWD
echo $USER
echo $PATH

time
which time
```

```
[dstep@nt098 ~]$ which time
/usr/bin/time
```

time コマンドは　/usr/binの下においてある

```
echo $PATH
```

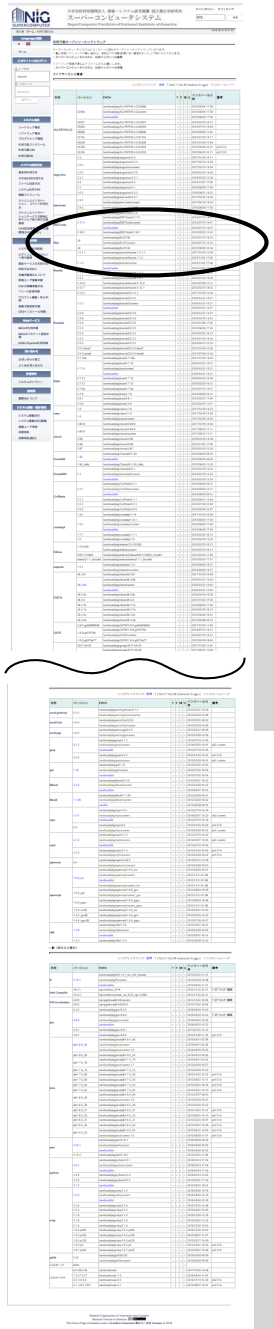
/usr/binというディレクトリは確かに環境変数PATHの中に記載されている

```
[dstep@nt098 ~]$ echo $PATH
/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/opt/ibutils/bin:/opt/intel/
itac/8.1.3.037/bin:/home/dstep/local/bin
```

実際の表示とは変えてあります

© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

4. 1-5. 利用可能OSS (7/8) のおさらい　(3/7)



	0.5.9	/usr/local/pkg/bwa/0.5.9	○	○	○	○	2012/02/07 11:07
canu	1.6	/usr/local/pkg/canu/v1.6	○	○	○	○	2017/10/19 14:22
	1.5	/usr/local/pkg/canu/v1.5	○	○	○	○	2017/05/18 15:42
		/usr/local/pkg/canu/current	○	○	○	○	2017/05/18 15:42
	0.69-6	/usr/local/pkg/circos/0.69-6	○	○	○	○	2017/10/19 14:36

/usr/local/pkg/canu/current

```
[dstep@nt094 ~]$ /usr/local/pkg/canu/current/bin/canu --version

Canu snapshot v1.5 +43 changes (r8243 f05c81531c19887459ca53ac3702509ee70eac22)
```

full pathを打たないで使えるようにしたい

```
[dstep@nt094 ~]$ canu

-bash: canu: コマンドが見つかりません
```

 # 環境変数PATHにcanuのfull pathを追加する
 # 下記は、打ち間違えると嫌な目にあいます。

```
$ export PATH=$PATH:"/usr/local/pkg/canu/current/bin"
```

確かにcanuと打つだけで、canuが使えるようになった。

```
$ which canu
/usr/local/pkg/canu/current/bin/canu
```

 # 今後もずっと環境変数PATHにcanuのfull pathを追加されるようにする
 # 下記は、打ち間違えるとひどい目にあいます。慎重に！！（もしくは試さない）

```
$ echo 'export PATH=$PATH:"/usr/local/pkg/canu/current/bin"' >>~/.bashrc
$ source ~/.bashrc
```

© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

4. 自分の環境を使いやすくする (4/7)

すでにインストールしてあるプログラムについては、PATHを通すだけで利用できる。

```
[dstep@nt098 ~]$ cat ~/.bashrc
```

```
# < 略 >

# BOWTIE2
export PATH=$PATH:"/usr/local/pkg/bowtie2/current"

# CANU
export PATH=$PATH:"/usr/local/pkg/canu/current/bin"

# < 略 >
```

→ # .bashrcファイルの中身の例

自分で好みのプログラムをインストールする方法は！？

Gblocksというプログラムのインストールを体験してみましょう。

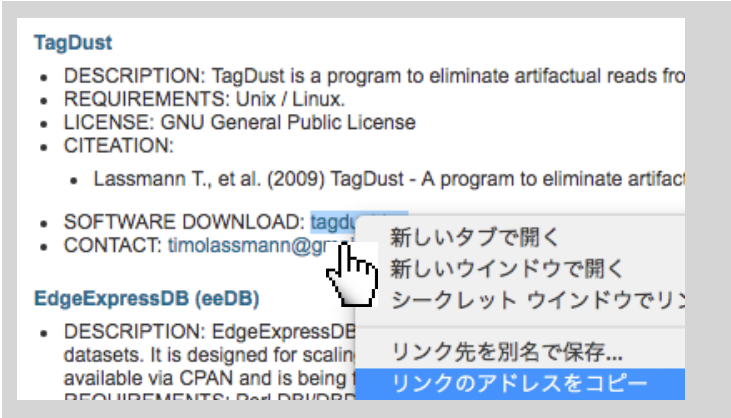
Gblocksは、SpainのCastresana Labが開発したアライメントの保存領域選択ツールです。

http://molevol.cmima.csic.es/castresana/Gblocks.html
http://genome.gsc.riken.jp/osc/english/dataresource/
http://molevol.cmima.csic.es/castresana/Gblocks/Gblocks_Linux64_0.91b.tar.Z

© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

4. 自分の環境を使いやすくする (5/7) Gblocksの install

```
# http://molevol.cmima.csic.es/castresana/Gblocks/Gblocks_Linux64_0.91b.tar.Z
# Gblocksのインストール
```




```
# ホームディレクトリにソースファイルを格納するディレクトリを作ります
[dstep@nt098 src]$ mkdir ~/src
[dstep@nt098 src]$ mkdir ~/src/gblocks
[dstep@nt098 src]$ cd ~/src/gblocks
[dstep@nt098 gblocks]$ pwd
/home/dstep/src/gblocks
# wgetでソースファイルを取得取得
[dstep@nt098 gblocks]$ wget http://molevol.cmima.csic.es/castresana/Gblocks/Gblocks_Linux64_0.91b.tar.Z
```

ダウンロード先のURLはコピペで良い

```
# gunzipと tarでソースファイルを解凍
[dstep@nt098 gblocks]$ Gblocks_Linux64_0.91b.tar.Z
[dstep@nt098 gblocks]$ ls
Gblocks_Linux64_0.91b.tar
[dstep@nt098 tmp]$ tar xvf Gblocks_Linux64_0.91b.tar

[dstep@nt098 tmp]$ ls
Gblocks_0.91b  Gblocks_Linux64_0.91b.tar
```

インストールといってもGblocksは解凍するだけですぐ使える

```
[dstep@nt098 tmp]$ cd Gblocks_0.91b
[dstep@nt098 Gblocks_0.91b]$ ls [dstep@nt098 Gblocks_0.91b]$ cat nad3.pir
Documentation  Gblocks  more_alignments  nad3.pir  paths
[dstep@nt098 Gblocks_0.91b]$ cat nad3.pir
[dstep@nt098 Gblocks_0.91b]$ ./Gblocks nad3.pir -t=p -e=-gb1 -b4=5 -d=y
[dstep@nt098 Gblocks_0.91b]$ ls
Documentation  more_alignments  nad3.pir-gb1      nad3.pir-gb1PS
Gblocks        nad3.pir          nad3.pir-gb1.htm  paths
[dstep@nt098 Gblocks_0.91b]$ cat nad3.pir-gb1
```

© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

4. 自分の環境を使いやすくする (6/7)

Gblocksを適切なディレクトリ（今回は ~/local/binを設定）に移動させる。

環境変数 PATH に ~/local/binを追加する。

```
[dstep@nt098 Gblocks_0.91b]$ mkdir ~/local
[dstep@nt098 Gblocks_0.91b]$ mkdir ~/local/bin
[dstep@nt098 Gblocks_0.91b]$ cp Gblocks ~/local/bin
```

 # 下記は、打ち間違えるとひどい目にあいます。慎重に！！（不安なら試さない）

```
[dstep@nt098 Gblocks_0.91b]$ echo 'export PATH=$PATH:"/usr/local/pkg/canu/current/bin"' >>~/.bashrc
[dstep@nt098 Gblocks_0.91b]$ source ~/.bashrc
```

Gblocksは解凍するだけですぐに使えました。

他の多くのプログラムは、

```
./configure –prefix="~/local/bin"
make
make install
```

としないと、動くプログラムが作成されません。

./configure の後ろにオプションとして、–prefix="~/local/bin"

としておくのが、ローカル環境にプログラムをインストールする時のTips(コツ)です。

© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

4. 自分の環境を使いやすくする (7/7)

以上で、自分の環境へ好みのプログラムをインストールする基礎を学びました

1. 環境変数について学びました。特にPATH環境変数を学びました。
   プログラムがどのディレクトリに存在するかはPATHに教えなければなりません。
   もし自分のプログラムを~/local/binに入れることにした場合は、~/.bashrcファイルに

   export PATH=$PATH:”~/local/bin”

   と記載します。

   ~/local/bin以外のディレクトリにプログラムが存在する場合は、そのディレクトリを
   環境変数PATHに追加しましょう。
   例えば. nigの大型計算機にすでにインストールされているbowtie2を利用したければ

   export PATH=$PATH:”/usr/local/pkg/bowtie2/current”

   などとすると利用できます。

2. nigの大型計算機に自分でプログラムをインストールする例を学びました。
   プログラムのインストール方法は、プログラム毎に様々です。
   READMEかINSTALLファイルに記載してあることが多いです。
   その手順が **./configure ; make ; make install** の３ステップのものが多いですが、
   この場合、./configureをする際に、**`./configure –prefix=“~/local/bin”`**
   などとする必要がある場合があります。

© 2014 DBCLS, Licensed under Creative Commons Attribution 4.0 International license (CC-BY 4.0)

5. short readのマッピング

ここまでで学んだことは、UNIX系OSの基本コマンドいくつか、大型計算機へのログイン方法、大型計算機へのコマンドの投入方法（キューイングシステム）、必要な生物学データの取得と自分の環境へのアップロード、必要なプログラムを自分の環境へインストールすること、などを学びました。

　これらの知識のおさらいとして、実際に配列解析の手順の一例を体験してみましょう。参照配列にshort readの配列をマッピングして見ることにします。

5. マッピング作業

すでにダウンロード済みの下記の配列ファイルをリファレンス配列にマッピングしてみましょう

```
ERR003990_1.fastq  ERR003990_2.fastq
```

手順は

ゼブラフィッシュのゲノム配列をリファレンス配列として取得しておく。(Ensembl)
今回は下記サイトから、染色体9番の配列だけを取得して利用。
ftp://ftp.ensembl.org/pub/release-91/fasta/danio_rerio/dna/Danio_rerio.GRCz10.dna.chromosome.9.fa.gz

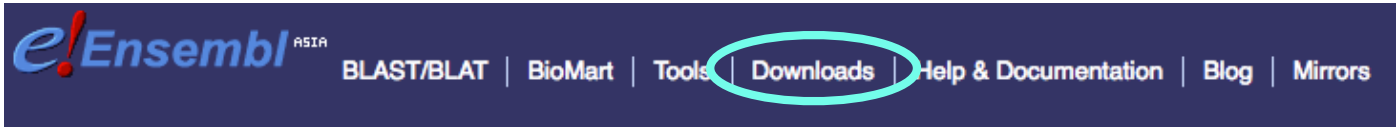
それぞれのfastqファイルのアダプターやクオリティーをfastQCでチェックする
Trimmomaticをインストールしておく
適切な値のクオリティーでフィルタリングする。Trimmomatic
得られたHigh Qualityのreadを、リファレンスにbowtie2でマッピングする。

samtools でマッピングの様子を眺める。
samtools でpileupしたファイルを作成する。

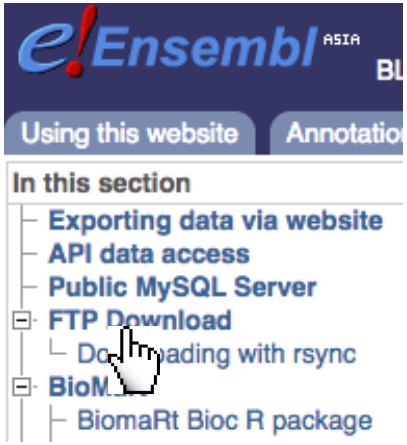
5. マッピング作業

ゼブラフィッシュのゲノム配列をリファレンス配列として取得しておく。(Ensembl)

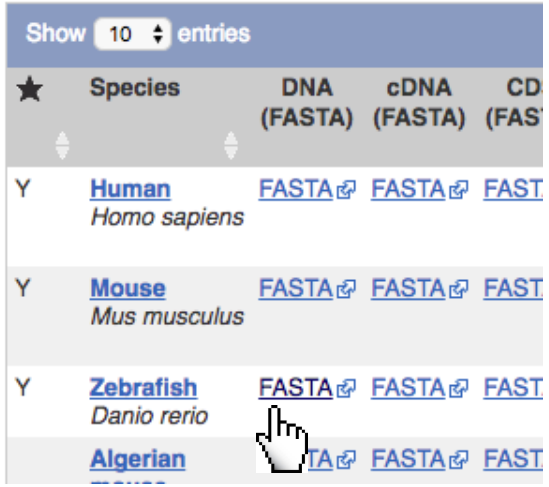
http://asia.ensembl.org/index.html



Download a sequence or region　　**# クリック**



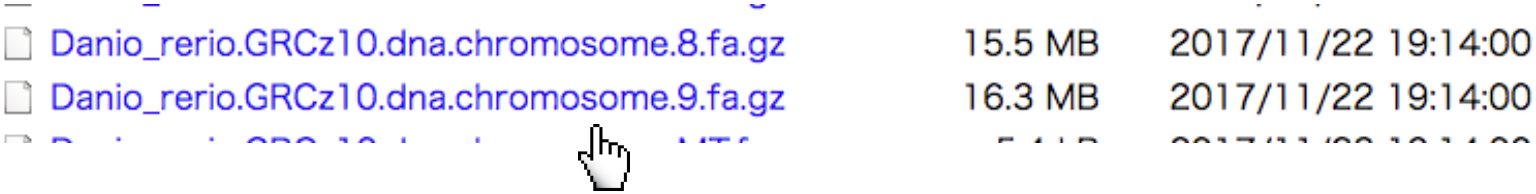
 # FTP Download クリック



http://asia.ensembl.org/info/data/ftp/index.html

 # Zebrafishの DNA (FASTA)を クリック

ftp://ftp.ensembl.org/pub/release-91/fasta/danio_rerio/dna/



ftp://ftp.ensembl.org/pub/release-91/fasta/danio_rerio/dna/Danio_rerio.GRCz10.dna.chromosome.9.fa.gz

5. マッピング作業

wget ftp://ftp.ensembl.org/pub/release-91/fasta/danio_rerio/dna/Danio_rerio.GRCz10.dna.chromosome.9.fa.gz

↑ ダウンロード先のURLはコピペで良い

```
[dtep@nt098 dstep20180126]$ ls
DRR016430.sra                                dat
Danio_rerio.GRCz10.dna.chromosome.9.fa.gz    hemoglobin_b.fa

[dtep@nt098 dstep20180126]$ gunzip Danio_rerio.GRCz10.dna.chromosome.9.fa.gz

[dtep@nt098 dstep20180126]$ ls
DRR016430.sra                                dat
Danio_rerio.GRCz10.dna.chromosome.9.fa       hemoglobin_b.fa
```

これでゼブラフィッシュのリファレンスゲノム配列を取得できた

5. マッピング作業

本来はそれぞれのfastqファイルのアダプターやクオリティーをfastQCでチェックする

fastQCはご自身のパソコンで行うほうがだいたいうまくいきます。今回はスルー
trimmomaticをインストールする

http://www.usadellab.org/cms/?page=trimmomatic

Trimmomatic: A flexible read trimming tool for Illumina NGS data

Citations

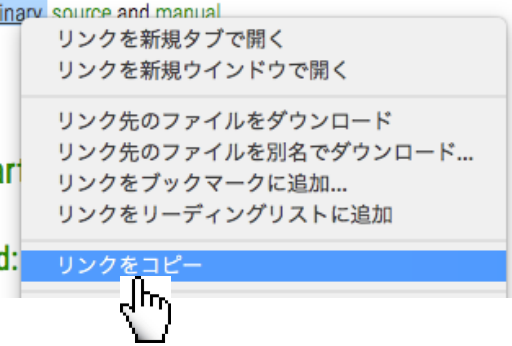
Bolger, A. M., Lohse, M., & Usadel, B. (2014). Trimmomatic: A flexible trimmer for Illumina Sequence Data. *Bioinformatics*, btu170.

Downloading Trimmomatic

Version 0.36: binary, source and manual

Downloading Trimmomatic

Version 0.36: binary, source and manual



Quick start

Paired End:

続く

5. マッピング作業

————trimmomaticのインストール、続き————

```
[dstep@nt098 ~]$ cd ~
[dstep@nt098 ~]$ mkdir src/
[dstep@nt098 ~]$ mkdir src/trimmomatic
[dstep@nt098 ~]$ cd src/trimmomatic

[dstep@nt098 trimmomatic]$ wget http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/Trimmomatic-0.36.zip
```

ダウンロード先のURLはコピペで良い

```
[dstep@nt098 trimmomatic]$ ls
Trimmomatic-0.36.zip
[dstep@nt098 trimmomatic]$ unzip Trimmomatic-0.36.zip
Archive:  Trimmomatic-0.36.zip
   creating: Trimmomatic-0.36/
  inflating: Trimmomatic-0.36/LICENSE
…
[dstep@nt098 trimmomatic]$ ls
Trimmomatic-0.36  Trimmomatic-0.36.zip
[dstep@nt098 trimmomatic]$ cd Trimmomatic-0.36
[dstep@nt098 Trimmomatic-0.36]$ ls
LICENSE  adapters  trimmomatic-0.36.jar
[dstep@nt098 Trimmomatic-0.36]$ pwd
 /home/dstep/src/trimmomatic/Trimmomatic-0.36
```

<- # このpathを覚えておく

trimmomaticもダウンロードして解凍すればそれで使える

5. マッピング作業

ここまでで、リファレンス配列とクエリーになるRNAseq配列の取得がすみ、
RNAseqのクオリティーフィルターをかけるためのTrimmomaticのインストール
が済んだ。次はTrimmomaticでRNAseqのクオリティーフィルターをかける。

```
# データのあるディレクトリに戻る
[dstep@nt098 Trimmomatic-0.36]$ cd ~/dstep20180126
```

```
# 小量データのサンプルファイルを作る
[dstep@nt098 dstep20180126] head -4000 ERR003990_1.fastq >s1.fastq
[dstep@nt098 dstep20180126] head -4000 ERR003990_2.fastq >s2.fastq

[dstep@nt098 dstep20180126] java -jar ~/src/trimmomatic/Trimmomatic-0.36/
trimmomatic-0.36.jar PE s1.fastq s2.fastq s1.tmf.fastq s1.tmf.unp.fastq
s2.tmf.fastq s2.tmp.unp.fastq TRAILING:20 MINLEN 36
```

5. マッピング作業

————trimmomaticのインストール、続き————

```
[dstep@nt098 ~]$ cd ~
[dstep@nt098 ~]$ mkdir src/
[dstep@nt098 ~]$ mkdir src/trimmomatic
[dstep@nt098 ~]$ cd src/trimmomatic

[dstep@nt098 trimmomatic]$ wget http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/Trimmomatic-0.36.zip
```

ダウンロード先のURLはコピペで良い

```
[dstep@nt098 trimmomatic]$ ls
Trimmomatic-0.36.zip
[dstep@nt098 trimmomatic]$ unzip Trimmomatic-0.36.zip
Archive:  Trimmomatic-0.36.zip
   creating: Trimmomatic-0.36/
  inflating: Trimmomatic-0.36/LICENSE
…
[dstep@nt098 trimmomatic]$ ls
Trimmomatic-0.36  Trimmomatic-0.36.zip
[dstep@nt098 trimmomatic]$ cd Trimmomatic-0.36
[dstep@nt098 Trimmomatic-0.36]$ ls
LICENSE  adapters  trimmomatic-0.36.jar
[dstep@nt098 Trimmomatic-0.36]$ pwd
 /home/dstep/src/trimmomatic/Trimmomatic-0.36
```

<- # このpathを覚えておく

trimmomaticもダウンロードして解凍すればそれで使える

5. マッピング作業

trimmomaticを使ってクオリティーフィルターをします

下記のコマンドで結果が得られるはずですが、
java -jar trimmomatic.jar PE s_1.fq s_2.fq s_1.filt s_1.unpair.fq s_2.filt.fq s_2.unpair.fq TRAILING:5 MINLEN:35

```
java -jar ~/src/trimmomatic/Trimmomatic-0.36/trimmomatic-0.36.jar \
PE ERR003990_1.fastq ERR003990_2.fastq \
ERR003990_1.trimmomatic.fq ERR003990_1.trimmomatic.unp.fq \
ERR003990_2.trimmomatic.fq ERR003990_2.trimmomatic.unp.fq \
TRAILING:8 MINLEN:36
```

すでに学んだ通り、qsub様のスクリプトを作成してqsubに投げる

```
[dstep@nt098 dstep20180126]$ cat trimmomatic.sh
```

```
#!/bin/sh
#$ -S /bin/sh
#$ -cwd
#$ -l s_vmem=10G,mem_req=10G
#$ -l short
#$ -pe def_slot 1
#$ -o ./log
#$ -e ./log

source ~/.bashrc
cd /home/dstep/tanomare/dstep20180126

java -jar ~/src/trimmomatic/Trimmomatic-0.36/trimmomatic-0.36.jar PE ERR003990_1.fastq
ERR003990_2.fastq ERR003990_1.trimmomatic.fq ERR003990_1.trimmomatic.unp.fq
ERR003990_2.trimmomatic.fq ERR003990_2.trimmomatic.unp.fq TRAILING:8 MINLEN:36
```

5. マッピング作業

trimmomaticを使ってクオリティーフィルターをします
すでに学んだ通り、qsub様のスクリプトを作成してqsubに投げる

```
[dstep@nt098 dstep20180126]$ mkdir log
[dstep@nt098 dstep20180126]$ qsub trimmomatic.sh
```

少し時間がかかります

```
[dstep@nt098 dstep20180126]$ qstat
```

qstat を何度か使って、コマンドの進み具合を見ます

```
[dstep@nt098 dstep20180126]$ ls log
[dstep@nt098 dstep20180126]$ cat log/trimmomatic.sh.e*
```

ERR003990_1.fastqとERR003990_2.fastq の二つの元ファイルから、
クオリティーフィルターをかけた ERR003990_1.trimmomatic.fqとERR003990_2.trimmomatic.fq
#が 得られた。

```
[dstep@nt098 dstep20180126]$ ls ERR*
ERR003990.sra                    ERR003990_2.fastq
ERR003990_1.fastq                ERR003990_2.trimmomatic.fq
ERR003990_1.trimmomatic.fq       ERR003990_2.trimmomatic.unp.fq
ERR003990_1.trimmomatic.unp.fq
[dstep@nt098 dstep20180126]$ ls ERR*

[dstep@nt098 dstep20180126]$ wc ERR003990_*.fq
  21290748   42581496  950083932 ERR003990_1.trimmomatic.fq
    377024     754048   16799544 ERR003990_1.trimmomatic.unp.fq
  21290748   42581496  950198864 ERR003990_2.trimmomatic.fq
    427972     855944   19079832 ERR003990_2.trimmomatic.unp.fq
```

ペアは同数になっている

5. マッピング作業

必要なファイルが揃いました

```
[dstep@nt098 dstep20180126]$ ls
Danio_rerio.GRCz10.dna.chromosome.9.fa
ERR003990_1.trimmomatic.fq
ERR003990_2.trimmomatic.fq
```

-h オプションをつけると使い方を表示する

```
[dstep@nt098 dstep20180126]$ /usr/local/bin/bowtie2-build -h
```

```
[dstep@nt098 dstep20180126]$ bowtie2-build -f
Danio_rerio.GRCz10.dna.chromosome.9.fa Danio_rerio.GRCz10.dna.chromosome.9.fa
```

それほど大きくない配列なので、今回はbowite2-buildをqsubに投げませんが
こういう場合も本来はqsubコマンドを作成してください。

```
[dstep@nt098 dstep20180126]$ ls
Danio_rerio.GRCz10.dna.chromosome.9.fa
Danio_rerio.GRCz10.dna.chromosome.9.fa.1.bt2
Danio_rerio.GRCz10.dna.chromosome.9.fa.2.bt2
Danio_rerio.GRCz10.dna.chromosome.9.fa.3.bt2
Danio_rerio.GRCz10.dna.chromosome.9.fa.4.bt2
Danio_rerio.GRCz10.dna.chromosome.9.fa.rev.1.bt2
Danio_rerio.GRCz10.dna.chromosome.9.fa.rev.2.bt2
```

5. マッピング作業

必要なファイルが揃いました

```
[dstep@nt098 dstep20180126]$ cat bowtie2.qsub
```

```
#!/bin/sh
#$ -S /bin/sh
#$ -cwd
#$ -l s_vmem=10G,mem_req=10G
#$ -l short
#$ -pe def_slot 1
#$ -o ./log
#$ -e ./log

source ~/.bashrc

bowtie2 -q -x Danio_rerio.GRCz10.dna.chromosome.9.fa -1
ERR003990_1.trimmomatic.fq -2 ERR003990_2.trimmomatic.fq
-S danio_ERR003990.sam
```

bowtie2.qsub

-Sオプションに与えた名前が出力

```
[dstep@nt098 dstep20180126]$ qsub bowtie2.qsub
[dstep@nt098 dstep20180126]$ wc danio_ERR003990.sam
```

5. マッピング

samtoolsでの一連の作業

```
[dstep@nt098 dstep20180126]$ samtools view -bt Danio_rerio.GRCz10.dna.chromosome.9.fa -o
danio_ERR003990.bam danio_ERR003990.sam
[dstep@nt098 dstep20180126]$ samtools sort -o danio_ERR003990.sorted.bam danio_ERR003990.bam
[dstep@nt098 dstep20180126]$ samtools index danio_ERR003990.sorted.bam
[dstep@nt098 dstep20180126]$ samtools faidx Danio_rerio.GRCz10.dna.chromosome.9.fa
[dstep@nt098 dstep20180126]$ samtools tview danio_ERR003990.sorted.bam
Danio_rerio.GRCz10.dna.chromosome.9.fa
```

キーボードのLとHで配列上を見ることができる。
キーボードのqで終了することができる

5. マッピング

以上で、

リファレンスとなるゲノム配列にshort readのクエリーをマッピングする手順を学びました。
実際には、この後さらに、マッピングの深さの分布やエラーやSNPの分布をpileupして
データを統計的に解析する (集団遺伝学的解析など)、マッピングによって得られたピークを検出する（ChIPSeq、DNase-seq、ATAC-seqなど）、マッピング領域を取り出す（RNAseqによる遺伝子予測など）、様々な解析が必要になります。
　これら個別の解析方法についてはお伝えした参考図書が参考になるでしょう。

6. キューイングシステム利用のための簡単なBASHプログラミング

ここまでで学んだqsubコマンドの利用方法は非常に初歩的なもののみです。ご自身の解析時間の短縮のためにも、大型計算機のリソースを有効利用するためにも、より効率的なshellプログラムの書き方を学びましょう。

6. キューイングシステム利用のための簡単なBASHプログラミング

```
[dstep@nt098 dstep20180126]$ cat bowtie2_multi.qsub
```

```
source ~/.bashrc
cd /home/dstep/tanomare/dstep20180126

arr=( ERR003990_1.trimmomatic.fq ERR003990_2.trimmomatic.fq ERR003991_1.fastq
ERR003991_2.fastq ERR003992_1.fastq ERR003992_2.fastq )

i=$(( $SGE_TASK_ID -1 ))                # <- $SGE_TASK_ID という変数を利用する
j=$(( $i * 2 ))
k=$(( $j + 1 ))

bowtie2 -q -x Danio_rerio.GRCz10.dna.chromosome.9.fa -1 ${arr[$j]} -2 $
{arr[$k]} -S danio.${arr[$j]}.pe.sam
```

$SGE_TASK_ID に-tで与えた数値が順に入る

```
[dstep@nt098 dstep20180126]$ qsub -t 1-3 bowtie2_multi.qsub
Your job-array 10398586.1-3:1 ("bowtie2_multi.qsub") has been submitted
```

```
[dstep@nt098 dstep20180126]$ qstat
job-ID     prior   name       user         state submit/start at     queue          jclass     slots ja-task-ID
-----------------------------------------------------------------------------------------------------------------
  10397226 0.25133 QLOGIN     dstep        r     01/19/2018 13:12:54 login.q@nt098i             1
  10398580 0.25011 QLOGIN     dstep        r     01/19/2018 16:39:45 login.q@nt099i             1
  10398586 0.25000 bowtie2_mu dstep        r     01/19/2018 16:58:54 short.q@nt115i             1 1
  10398586 0.25000 bowtie2_mu dstep        r     01/19/2018 16:58:54 short.q@nt115i             1 2
  10398586 0.25000 bowtie2_mu dstep        r     01/19/2018 16:58:54 short.q@nt115i             1 3
```

下記三つのファイルが得られた。

```
danio.ERR003990_1.trimmomatic.fq.pe.sam
danio.ERR003991_1.fastq.pe.sam
danio.ERR003992_1.fastq.pe.sam
```

7. （時間に余裕があれば）描画のための簡単なRプログラミング (1/9)

Rについては当日、講演者が十作業をお見せするのみにいたします。