NextGen Sequencing and Perl

Eric Ross
2016-10-11

Sequencing Technologies

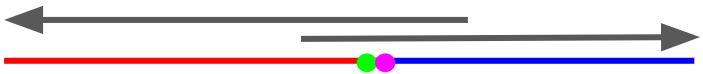
- Sanger
  - 700bp reads
  - High quality
  - Expensive
- Illumina
  - Short reads 50-250bp (HiSeq) and up to 300bp (MiSeq)
  - 250M paired-reads per lane on a Hiseq 2500.
  - Can be barcoded
  - Relatively cheap
  - Very cheap on NextSeq, but at the cost of some accuracy.
- PacBio
  - Long reads (averaging > 10 kb)
  - Low seq quality (before correction)
  - Relatively Expensive

Library Types

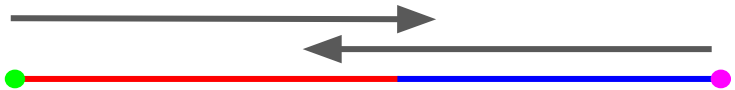
- Single-end

- Paired-end

- Mate-pair



- Jump



File Types

- FASTA
- FASTQ
- SAM/BAM
- GFF
- BED
- VCF

https://genome.ucsc.edu/FAQ/FAQformat.html

FASTA (FAST-AII)

>lcl|KP090060.2_cds_AMM76166.1_10 [protein=cytochrome c oxidase subunit II]
ATGCATATAAAAGGAAAATTTCATTTATTTTATTGATTTCAACATTCAGCTTCTACTCATATGGACCATA
TTTACAGTTTACATATAAAGTTAATGTTAATAATTATTTTTATAGCAGCTTTTGTTGCTTCAATTTTGAT
ATTTGTTTATTTCAATCCTTATTATTTTTCATCTCCTTTAGATAATCCTTATTTAGAATGTATTTGAACT
CTTCTTCCAGGAATTTTATTATTATTTGTTTCTGGTCCAAGTCTTTATGCTTTATATTTTATAGATTCTC
CATTTCTTATTAGTCCTAAGGCAACATTAAATGTTATAGCTCACCAATGGTATTGAGAATATATTTATAT
TTTACACAATCGTATAAGAAGATTAAAAATTAAAACTGATGCATATATGCAAAAGTTTTCAACTAAGAAA
CACATTACTCGAATACTAGAACCTAATAAGAGAGTTATACTTTGTAATCGAAATTTAGAAGTATTATTTC
CGTTAGGTATTCAAGTAGATACTCCAACTCGATTTATGATTAGATCTGCAGATGTTATACATAGTTTTGC
ATTACCTGGAATGGGGGTAAAGGTAGATGCAATTCCAGGTCGAAATAATCAAGCTCAAATTTTAGCATAC
CGATGTGGAAAGTATTTTGGCCAATGTTCAGAAATGTGTGGAACTTATCATTCTTTTATGCCAATTGCTT
TAGAGGTATTGAATTATATTAAAATTCCAAAGCCAATTATACCAGAAAAACCATTAATCGAACAAGGACT
TACTACACGTGGACAATTTATTCTTTTTGGTAGTATTGCATTGGGTATCCTAATTTGATTTGGAGTAATA
TATCCTACGTATTTAGATCAATATGTTGTTATTATATCTAATCCTTGACCTTAA
>lcl|KP090060.2_cds_AMM76167.1_11 [protein=NADH dehydrogenase subunit 3]
ATTTTATTATTACTTTTTTTTCTTATAGTTTTTTGAATAAAGATCCAAAGAAAAAACCTAGATCGTGAAA
GAACTTCAAAATATGAATGTGGATTTGATATAAAGGATTCTGCACGATTACCATTTTCTATTCGATTTTT
TTTAATTTTAATAATATTTATAATTTTTGATGTTGAAATTTGTTTTTTATTACAAGTAGTGTTTGAAAAT
CCATTTTATGGTACTCGAATTTGATATGTGTTATTTTTAAAATTTATATTAATCGGGGTGTTGGAAGAAA
ATCGTCGAGGGGCTTTTTTATGAAAGATTTAA

FASTQ (like FASTA, but with Quality scores)

```
@D4ZHLFP1:34:D169JACXX:1:1101:1214:2133 1:N:0:ACTTGA
GCTTTCTTCCTCGGAAGAATGGAACGTATCTATGCCATGAGTATCTATCCAGAGAAAGTTGCTCTAATACGAGAACTTGTGCATTTGCTCA
TTTGAGTAAA
+
CCCFFFFFHHHHHJJJJJJJJJJJJJIJJJJJJJJJJJJJJHIJJIIJJJIJIIIIJJHIGIIIIIIIJIJFHGFFFFFFEDEEEEEDDDD
DEFEEBCDDD
@D4ZHLFP1:34:D169JACXX:1:1101:1141:2159 1:N:0:ACTTGA
GGACTCCATGGGAACTGGAACTGAGTTTGCGCTGGGTGGGAAGCAGCGCATGCACCTTCTTGACGCCGTCACTGAGTCCGGTGATGAGGAC
GATGATAGTG
+
CCCFFFFFHHHHHJJJJJJJJJJGJIJJJJJJJJJJFHIJIJJJJJJJJJHHHFFFFFEEEEEEDDDDBDDDDDCCCDDDD>@@CDDDDDD
DBDDDDCDAA
@D4ZHLFP1:34:D169JACXX:1:1101:1054:2200 1:N:0:ACTTGA
TACCTTACTTATCATATCCAGATGAGCGATGCTTTCTAGAACAGTACAAAAGACCCTTGGCATATGTTTCTGAAGTTGAAATGGAAAAGAG
AATAGAACTT
+
CCCFFFFFHHHHHJJIJJJJJJJJJJJJJJJJJJJJJIJIJJHIJJIJJJJJJIJJIJJIIJJHIIIJJEHHGHHFHGFFFFFEDEDE
DCCDDCCCDC
```

SAM / BAM (Sequence Alignment Map)

```
D4ZHLFP1:53:D2386ACXX:6:2115:17945:68812     0    Mle_000001     18   42   108M *    0    0
     TCCCCCTGCATGTGTCCGTCTGGCTGGATGCCATGCTCCATGCAGTATAGCTCCCAGCATGAGTTACCGATCTGGACACCTGCTTG
GCCAAGATGTACTGAGATGCAT
C@CFDFFFHHGHHHFGGBFEGGDGGGGEHGIGGGJJJJIIIGIIB9BFBBFHGGHICEAGHGEGEDHIGEEDBECCACBDDC@CCDBCDD<
?2+4>@4>>CCCCAA@@    AS:i:-5    XN:i:0    XM:i:1    XO:i:0    XG:i:0    NM:i:1    MD:Z:0A107
     YT:Z:UU
D4ZHLFP1:53:D2386ACXX:7:2110:5214:83081 0    Mle_000001     18   42   108M *    0    0
TCCCCCTGCATGTGTCCGTCTGGCTGGATGCCATGCTCCATGCAGTATAGCTCCCAGCATGAGTTACCGATCTGGACACCTGCTTGGCCAA
GATGTACTGAGATGCAT
CCCFFFFFHHHHHHGHGGEGIJIIIGJFHJJJJJIIJJIIGIJIJFHJJIIIJJIFIIIIIIJIIJJJHHFFFCEEEEDDDDDDDDDDDDD
BDCDDEEEDDDDDDDDD    AS:i:-5    XN:i:0    XM:i:1    XO:i:0    XG:i:0    NM:i:1    MD:Z:0A107
     YT:Z:UU
D4ZHLFP1:53:D2386ACXX:7:2206:9985:31556 0    Mle_000001     18   42   108M *    0    0
TCCCCCTGCATGTGTCCGTCTGGCTGGATGCCATGCTCCATGCAGTATAGCTCCCAGCATGAGTTACCGATCTGGACACCTGCTTGGCCAA
GATGTACTGAGATGCAT
CCCFFFFFHHHHHJJJJIHJJIIIIJJIJJJJJJJJJIJJJJJJJIIJJJJJJJJJJJJJJJJJIJJJGEFFFEEEEEDDDDDDDDDDDDD
DDCD@@CDCCDDCDCDC    AS:i:-5    XN:i:0    XM:i:1    XO:i:0    XG:i:0    NM:i:1    MD:Z:0A107
     YT:Z:UU
```

VCF (Variant Call Format)

```
20     14370   rs6054257 G      A       29   PASS   NS=3;DP=14;AF=0.5;DB;H2
GT:GQ:DP:HQ 0|0:48:1:51,51 1|0:48:8:51,51 1/1:43:5:.,.
20     17330   .         T      A       3    q10    NS=3;DP=11;AF=0.017
GT:GQ:DP:HQ 0|0:49:3:58,50 0|1:3:5:65,3   0/0:41:3
20     1110696 rs6040355 A      G,T     67   PASS   NS=2;DP=10;AF=0.333,0.667;AA=T;DB
GT:GQ:DP:HQ 1|2:21:6:23,27 2|1:2:0:18,2   2/2:35:4
20     1230237 .         T      .       47   PASS   NS=3;DP=13;AA=T
GT:GQ:DP:HQ 0|0:54:7:56,60 0|0:48:4:51,51 0/0:61:2
20     1234567 microsat1 GTCT   G,GTACT 50   PASS   NS=3;DP=9;AA=G
```

BED (Browser Extensible Data)

```
Chromosome_I    29205    30824    JUNC00000001    1    -    29205    30824    255,0,0    2
     29,72    0,1547
Chromosome_I    61202    61452    JUNC00000002    576    +    61202    61452    255,0,0    2
     100,100    0,150
Chromosome_I    61434    65332    JUNC00000003    638    +    61434    65332    255,0,0    2
     100,100    0,3798
Chromosome_I    61460    65330    JUNC00000004    3    +    61460    65330    255,0,0    2
     74,86    0,3784
Chromosome_I    79724    80426    JUNC00000005    3    -    79724    80426    255,0,0    2
     27,83    0,619
```

GFF (General Feature Format)

```
##gff-version 3
ChrI .         contig 1       9179 .    .    .    ID=v31.022151;Name=v31.022151
ChrI maker     gene   1684    7298 .    +    .    ID=gene1;Name=gene1
ChrI maker     mRNA   1684    7298 .    +    .    ID=mRNA1;Parent=gene1;Name=mRNA1
ChrI maker     exon   1684    1798 .    +    .    ID=mRNA1:exon:726;Parent=mRNA1
ChrI maker     exon   2917    3069 .    +    .    ID=mRNA1:exon:727;Parent=mRNA1
```

Perl Example #1

Convert a BED file into a GFF file.

```perl
#!/usr/bin/perl
# Convert tophat junctions.bed into GFF3 format
# ejr - 2016-10-12
use strict;
use warnings;

# write out header
print "##gff-version 3\n";

# read in junctions file
# throw away first line of the bed file
open(IN, "<", "junctions.bed") or die "Cannot open file: $!\n";
my $junk = <IN>;

while (my $line = <IN>) {
    chomp $line;
    # split into fields, fields we don't care about are in @other
    my ($chr, $start, $end, $name, $score, $strand, @other) = split /\t/, $line;

    $start = $start + 1;
    # print GFF to STDOUT.  fields separated by tabs.
    print join("\t", $chr, "tophat", "splice_site", $start, $end, $score, $strand, ".",
"ID=" . $name . ";Name=" . $name), "\n";

}
```

Quality Control

- Quality Scores
- FASTQC
- Clipping and Trimming

Quality Scores

Q = -10log10(E)

Where E = estimated probability of the base call being wrong

Q	Probability of incorrect base call	Inferred base call accuracy
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1000	99.9%

http://www.illumina.com/science/education/sequencing-quality-scores.html

Quality Scores

S - Sanger Phred+33, raw reads typically (0, 40)
X - Solexa Solexa+64, raw reads typically (-5, 40)
I - Illumina 1.3+ Phred+64, raw reads typically (0, 40)
J - Illumina 1.5+ Phred+64, raw reads typically (3, 40)
with 0=unused, 1=unused, 2=Read Segment Quality Control Indicator (bold)
(Note: See discussion above).
L - Illumina 1.8+ Phred+33, raw reads typically (0, 41)

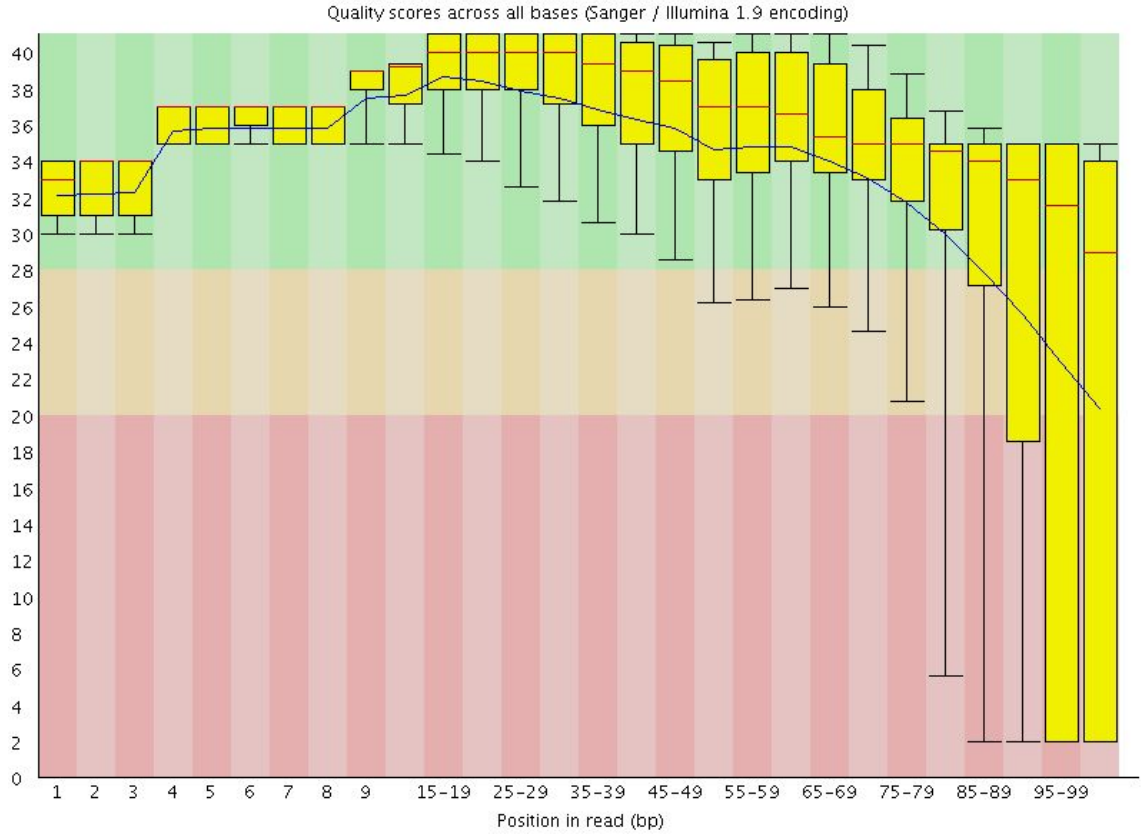
https://en.wikipedia.org/wiki/FASTQ_format

FASTQC

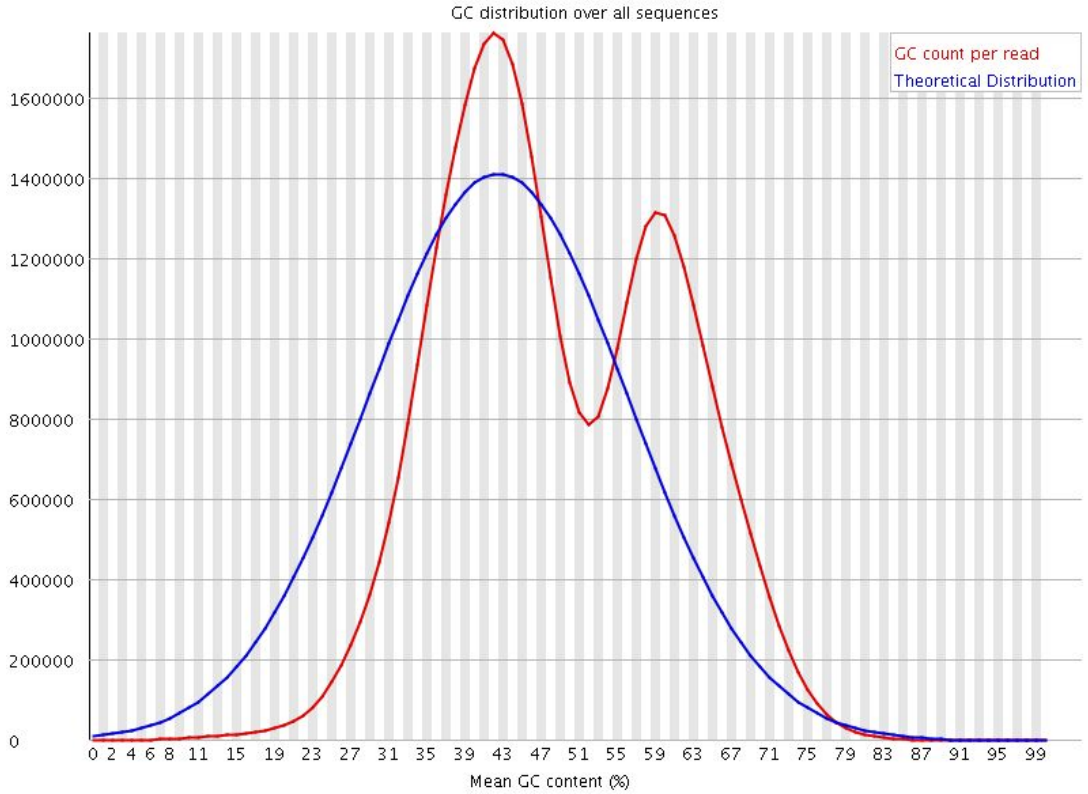
- Samples FASTQ file to assess quality quickly.
- Independent of reference sequence.

http://www.bioinformatics.babraham.ac.uk/projects/fastqc/

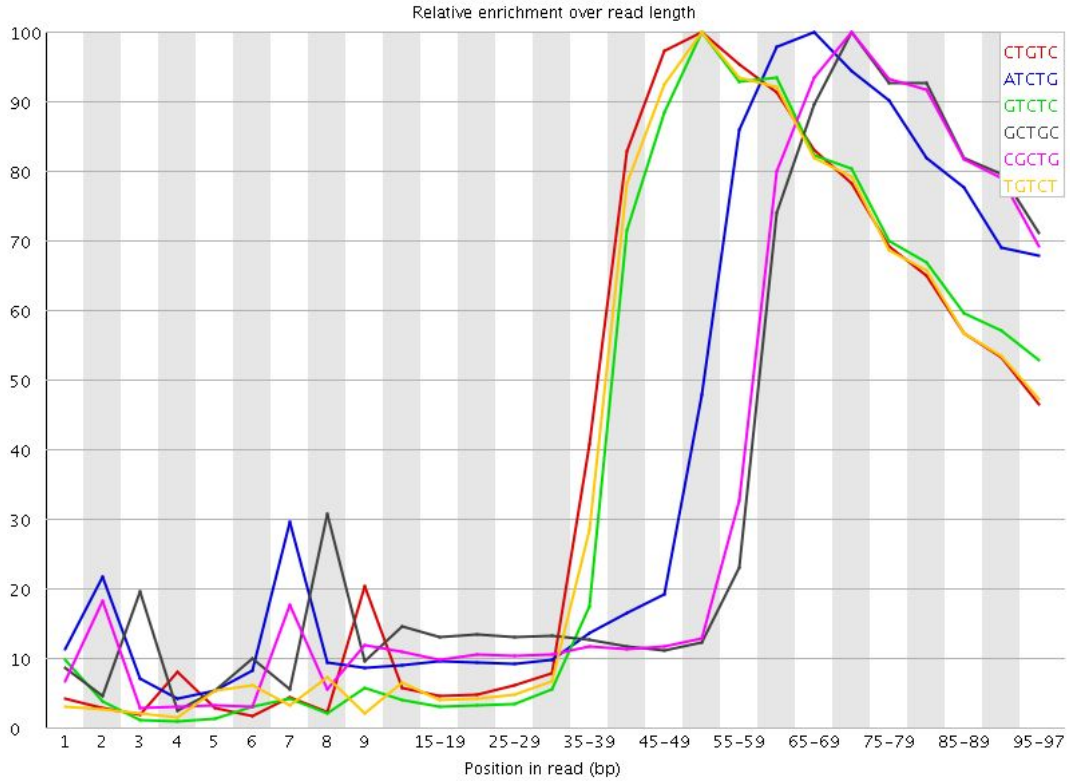
FASTQC - base quality



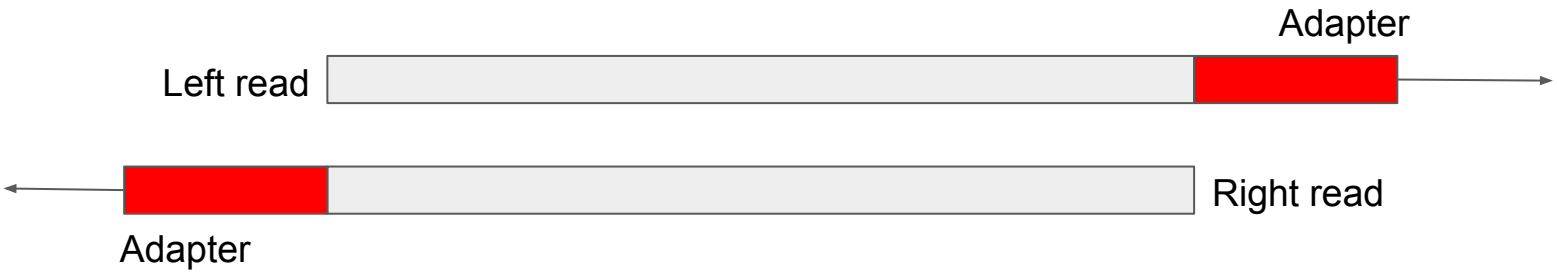
FASTQC - GC distribution



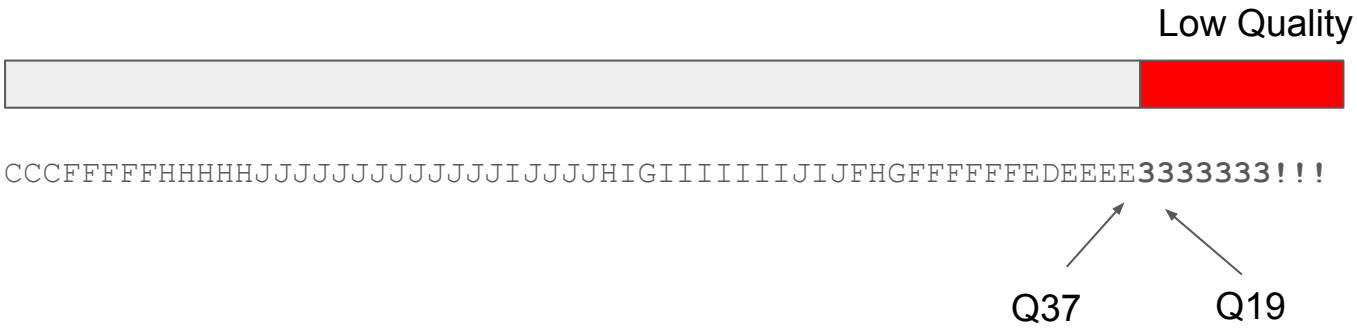
FASTQC - Over-represented kmers



Adapter Clipping



Quality Trimming



Short Read Alignment

- There are many short read aligners: Bowtie, BWA, STAR etc.
- Designed to be very fast, because of the enormous number of reads.
- This speed comes at the expense of accuracy.
- For variant calling the original alignments at variant sites are realigned to improve accuracy.

Useful Tools

- samtools - manipulate SAM and BAM files
- bedtools - manipulate and compare BED and GFF files.
- bcftools / vcftools - manipulate VCF files
- FASTX toolkit - FASTA and FASTQ manipulation
- sickle / Trimmomatic - Quality trimming
- GATK - variant discovery

Perl Example #2

Process a vcf file and calculate the number of each type of nucleotide substitutions.

```
ejr@compute:~/ngs_example$ bcftools query -f '%CHROM  %POS  %REF
%ALT{0}\n' ALL.chr22.vcf.gz | head
[W::bcf_hdr_check_sanity] GL should be declared as Number=G
22  16050408  T  C
22  16050612  C  G
22  16050678  C  T
22  16050984  C  G
22  16051107  C  A
22  16051249  T  C
22  16051347  G  C
22  16051453  A  C
22  16051477  C  A
22  16051480  T  C

ejr@compute:~/ngs_example$ bcftools query -f '%CHROM  %POS  %REF
%ALT{0}\n' ALL.chr22.vcf.gz > ALL.chr22.txt
[W::bcf_hdr_check_sanity] GL should be declared as Number=G

ejr@compute:~/ngs_example$ wc -l ALL.chr22.txt
494328 ALL.chr22.txt
```

1000 Genomes data from ftp://ftp-trace.ncbi.nih.gov/1000genomes/ftp/release/20110521/

```perl
#!/usr/bin/perl
# Calculate the number of each type of nucleotide substitution.
# ejr - 20161012
use strict;
use warnings;

my %subs;

open(IN,"<", "ALL.chr22.txt") or die "cannot open file:$!\n";

while( my $line = <IN>) {
    chomp $line;
    my ($chr, $pos, $ref, $var) = split / +/, $line;
    # we only count substitutions, not insertions or deletions
    if (length($var) == 1 and length($ref) == 1) {
        my $type = $ref . " to " . $var;
        $subs{$type}++;
    }
}

# output counts of each substitution type
foreach my $type (sort keys %subs) {
    $subs{$type} = add_commas($subs{$type});
    printf("%s\t%8s\n", $type, $subs{$type});
}
```

```perl
sub add_commas {
    my $number = shift;

    # Find the integer part of the number. By default we assume
    # that the number is an integer, but if we observe a '.' in
    # the number, then its a decimal and we must start from there
    my $integer = length($number);
    my $decimal = index($number, '.');
    if ($decimal >= 0) {
        $integer = $decimal;
    }
    # Don't want comma at decimal or end of integer, subtract 3
    # from initial value.
    for(my $i = $integer - 3; $i > 0; $i -= 3) {
        substr($number,$i,0,',');
    }
    return $number;
}
```

```
ejr@compute:~/tmp/example$ ./calc_sub.pl
A to C     13,759
A to G     55,603
A to T     11,997
C to A     20,481
C to G     21,970
C to T    113,628
G to A    114,818
G to C     21,568
G to T     20,552
T to A     11,711
T to C     55,770
T to G     13,514
```