Atlas-SNP2 DOCUMENTATION

V1.0 January 20, 2010

Contact: Jin Yu (jy2@bcm.tmc.edu), and Fuli Yu (fyu@bcm.tmc.edu)

Human Genome Sequencing Center (HGSC) at Baylor College of Medicine (BCM)

Houston TX, USA

1. Introduction

Atlas-SNP2 is aimed at detecting single nucleotide polymorphisms (SNPs) from whole-genome *resequencing* data sets produced by second generation sequencing technologies. In the current version (V1.0), Atlas-SNP2 supports data from 454 Life Sciences (Roche) — both GS FLX and Titanium — and Illumina platforms.

Atlas-SNP2 is SAM friendly. It takes the standard SAM format (H. Li. 2009) alignment files and reference sequence files in FASTA format as the inputs, and outputs putative SNP sites, the estimated posterior SNP probabilities and various read context related information. Release V1.0 only uses the required fields in SAM, and has been tested on SAM files from Human Genome Sequencing Center at Baylor College of Medicine (BCM-HGSC), Wellcome Trust Sanger Institute, Washington University Genome Sequencing Center (WUGSC), which were generated using BLAT+cross_match (and later converted to SAM), MAQ, Mosaik respectively. We have also tested SAM generated by Bowtie.

Atlas-SNP2 keeps evolving. In the previous draft release (V0.1), we included several modules that facilitate the data processing pipelines (for example, the "454 raw read-BLAT/crossmatch-SAM" workflow). Their respective documentation is included in the Appendix. The official release site for the latest version and documentation is at the BCM-HGSC website (http://www.hgsc.bcm.tmc.edu/cascade-tech-software_atlas_snp-ti.hgsc). You are welcome to check back for periodic updates.

2. System requirements

- Ruby 1.9.1: http://www.ruby-lang.org/en/downloads/
- UNIX-like operating system
- If needed, the external mapping tools - BLAT and *cross_match* - can be obtained from http://users.soe.ucsc.edu/~kent/src/ and http://www.phrap.org respectively.

3. Using Atlas-SNP2

ruby Atlas-SNP2.rb -i [in.SAM] -r [reference.fa] -o [output]

Options:

-i	FILE	SAM format alignment file (Required)
-r	FILE	FASTA format reference sequence file (Required)
-o	STR	name of output result file (Required)

Choosing Platform: (Default is 454 FLX)

-s	Illumina
-x	454 Titanium

Setting up priori and filters:

-e	FLT	Prior(error|c) when variant coverage number is above 2 (Default is 0.1)
-l	FLT	Prior(error|c) when variant coverage number is 1 or 2 for 454 data (Default is 0.9)
-m	FLT	maximum percentage of substitution bases in the query (Default is 5.0)
-g	FLT	maximum percentage of insertion and deletion bases in the query (Default is 5.0)

Altas-SNP2 is flexible on inputting reference sequence files. Users could either use one specified chromosome at a time or slurp in the whole genome, based on the research purpose or the memory limitation of the computing system. It is recommended to process chromosome by chromosome to get the results faster when run Atlas-SNP2 on huge SAM files.

Please do check the SAM files you use and pay attention to the version of the reference genome used when produce the SAM. It is users' responsibility to keep the version of the reference genome that you feed into Atlas-SNP2 consistent with that indicated in SAM files.

4. Output file format

The output of Atlas-SNP2 is a list of putative SNPs, each has detailed information described in 18 tab delimited fields. Headers are shown below:

refName<tab>coordinate<tab>refBase<tab>variantBase<tab>oriQual<tab>variantReadCov<tab>numAlte
rnativeReads<tab>totalCoverage<tab>Pr(error)j<tab>Pr(SNP)j<tab>Pr(Sj|error,c)<tab>Pr(Sj|SNP,c)<tab>
Prior(error|c)<tab>Prior(SNP|c)<tab>Pr(SNP|Sj,c)j<tab>refEnv<tab>homopolymer<tab>readsInfo

Table 1. Explanations for different fields in the output.

Field Name	Explanation
refName	The name of the reference sequence, for example, chr12.
Coordinate	The physical position of the SNP site on the reference sequence.
refBase	The reference base in that SNP position.
variantBase	The variant base (if there are several variants, it takes the one with largest occurrence).
oriQual	The summation of the phred quality scores of all reads showing the variant base.
variantReadCov	The number of reads that harbor the same variant base (shown in the variantBase column).
numAlternativeReads	The total number of reads that differ from the reference sequence on the SNP site.
totalCoverage	The total number of reads covering the SNP site.
Pr(error)j	The prior error probability of the locus j when conditioning on variant read coverage.
Pr(SNP)j	The prior SNP probability of the locus j when conditioning on variant read coverage. This entire item is represented by a symbol *Sj*, which stands for signal at locus j.
Pr(Sj|error,c)	This is derived from the probability density distribution of *Sj* of errors at a specific variant read coverage *c.*
Pr(Sj|SNP,c)	This is derived from the probability density distribution of *Sj* of true SNPs at a specific variant read coverage *c.*
Prior(error|c)	Prior estimation of the substitution error rate when conditioning on variant read coverage.
Prior(SNP|c)	Prior estimation of the substitution SNP rate when conditioning on variant read coverage.
Pr(SNP|Sj,c)j	The Posterior SNP probability of the locus j when signal is Sj at a specific variant read coverage *c.*
refEnv	The reference sequence of a 13-bp window centered on the SNP site.

homopolymer	The size of the longest homopolymer within a 13-bp window centered on the SNP base on the reference sequence (legacy column, only useful for manual inspection).
*readsInfo**	Information about the reads that harbor the same variant base.

*The field "readsInfo" contains a list of semicolon-separated strings as shown in the following example. Each of the strings contains the related information of the respective variant read. The following example shows the format of one of the strings in field "readsInfo". Table 2 gives the detailed explanation about each element within the string shown below as an example.

T(15)EIXH2IB02H9BM7(16)(235.0/272)+taacccTaaccta(0.38/1.92)(0/272)snp(0.861);

Table 2. Explanations for "readsInfo"

String name	Explanation
T	The variant base
15	Raw phred-like quality score
EIXH2IB02H9BM7	The read name
16	The distance of the variant base to right end of the read
235.0/272	Smith-waterman score of match / The queried size on the reads
+	The read direction of strand
taacccTaaccta	The variant read sequence of a 13-bp window centered on the SNP site.
0.38/1.92	%subsitutions in matching region /%indels in matching region
0/272	The status of "NQS pass" / the length of the reads
Snp	This variant base is a substitution from the reference sequence to the read sequence
0.861	The prior estimated error probability of this variant base i, Pr(error)i

5. References

Shen, Y., Wan, Z., Coarfa, C., Drabek, R., Chen, L., Ostrowski, E. A., Liu, Y., Weinstock, G. M., Wheeler, D. A., Gibbs, R. A., and Yu, F. (2009) A SNP discovery method to assess variant allele probability from next generation resequencing data. Genome Research. doi:10.1101/gr.096388.109.

Li, H. (2009) Sequence Alignment/Map (SAM) format. SAMtools. Retrieved Jan 20, 2010, from http://samtools.sourceforge.net/SAM1.pdf.

Kent, W.J. (2002) BLAT—the BLAST –like alignment tool. Genome Research, 12, 656-664.

Green, P. (1993) Cross_match. Green Group. http://www.phrap.org.

6. Change log

Current release V1.0 (2010-01-20)

- added Illumina Platform support
- all calculation are based on required fields of SAM to get maximum compatibility
- added CIGAR and reference sequence test code
- used pileup number to calculate TotalCoverage
- improved performance
- migrated to Ruby 1.9
- a lot of minor improvements

Draft release V0.1 (2009-12-10)

- Initial implementation
- Initial support of SAM files

Appendix:

A “454 raw reads->BLAT/*cross_match*->SAM” workflow

1. Introduction

The whole workflow includes three steps: “Atlas-SNP-mapper”, “Atlas-SNP-splitter” and “Atlas-SNP-SAM” (Figure S1.1)

“Atlas-SNP-mapper” is a wrapper that takes the reference genomic sequence, the NGS reads (including sequence fasta and the quality files) as the inputs, and generates mapping results in *cross_match* output format.

“Atlas-SNP-splitter” next takes the outputs and NGS reads quality fasta files (that need to be partitioned) as its inputs. After splitting, its outputs are a series of smaller batches. This module is designed for the resource management purpose when NGS data size is large.

“Atlas-SNP-SAM” takes the outputs, read file and the read quality files and reference sequence file as the inputs to generate alignment files that are in SAM format. This is only a simple file reformatting step.

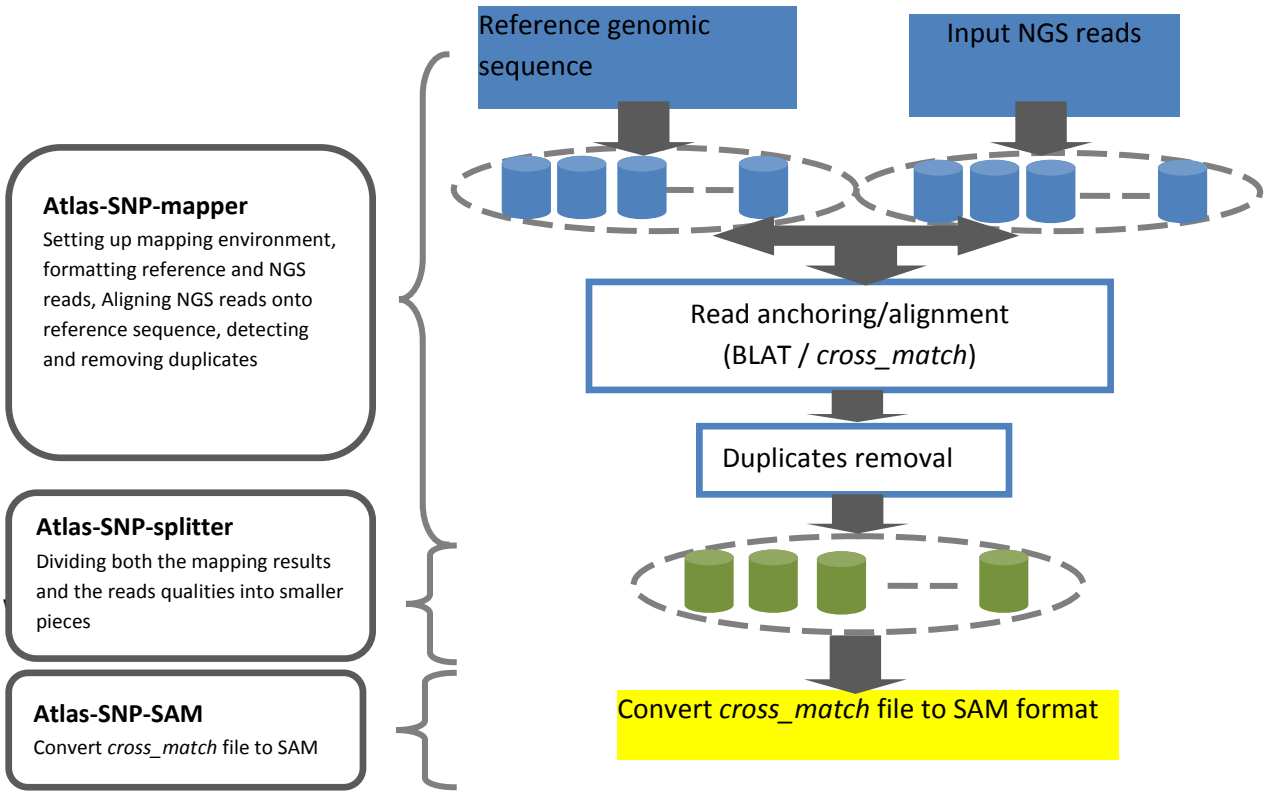

Figure S1.1 The overall “454 raw reads->BLAT/Cross-match->SAM” workflow.

Step 1: Atlas-SNP-mapper (Figure S1.2)

Atlas-SNP-mapper splits reference genomic sequences to smaller regions with the size ranging from 10Kb to 10Mb per region. It also divides NGS read files and read quality files into smaller pieces, ranging from 5Mb to 10Mb in size for the purpose of computational resource management. It uses the divided pieces from both the genomic reference sequence and NGS reads as inputs to attempt to anchor and align all the NGS sequences onto the reference sequence. A few further steps such as “duplicates removal” can be used to remove experimental artifacts from sequencing.

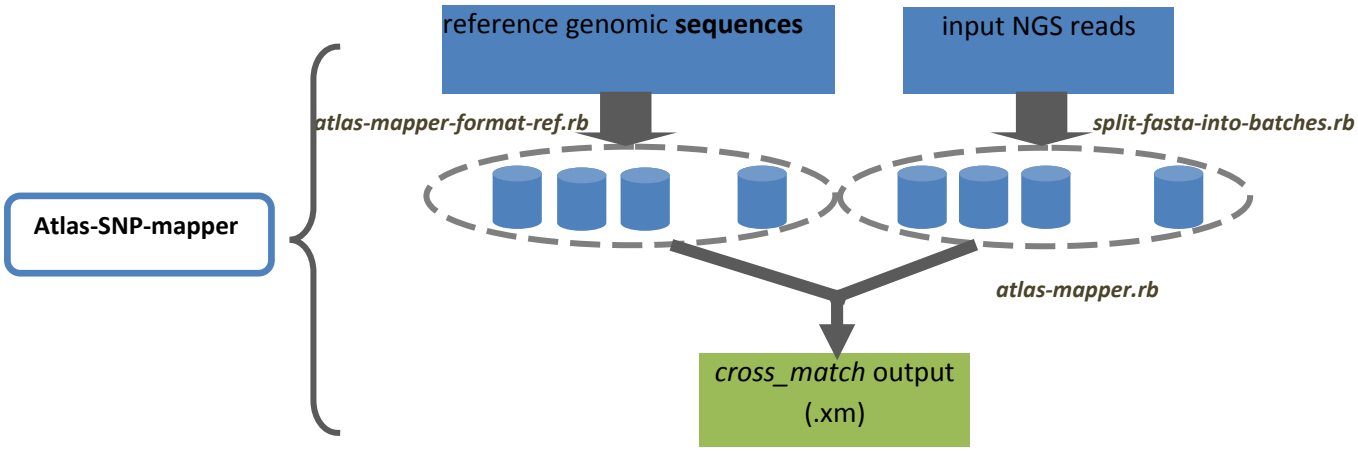

Figure S1.2 Atlas-SNP-mapper

Step 2: Atas-SNP-splitter (Figure S1.3)

Atlas-SNP-splitter splits mapping results and respective quality files of the successfully mapped reads into smaller batches. The program first splits the mapping result by chromosomes into batches, and further splits the batches within each chromosome into smaller batches that roughly have the same size. Next, two more steps are applied to categorize the successfully mapped reads into batches by looking up the read name from the original mapping batches.

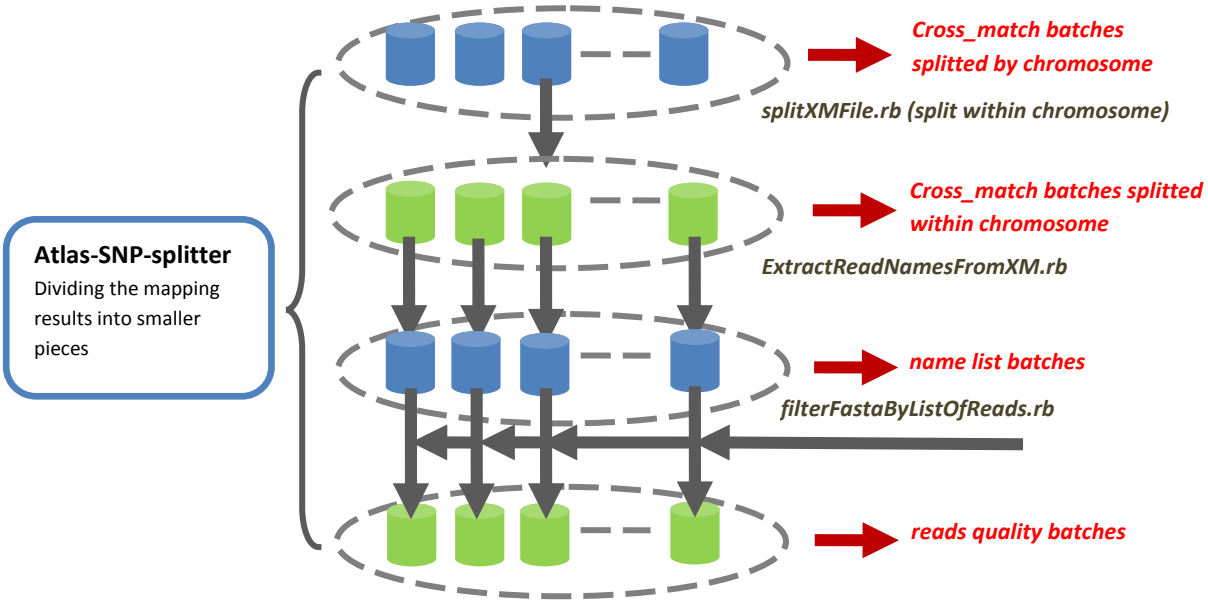

Figure S1.3 Atlas-SNP-splitter

Step 3: crossmatch2SAM (Figure S1.4)

crossmatch2SAM.rb is designed to convert *cross_match* format alignment files into SAM format.

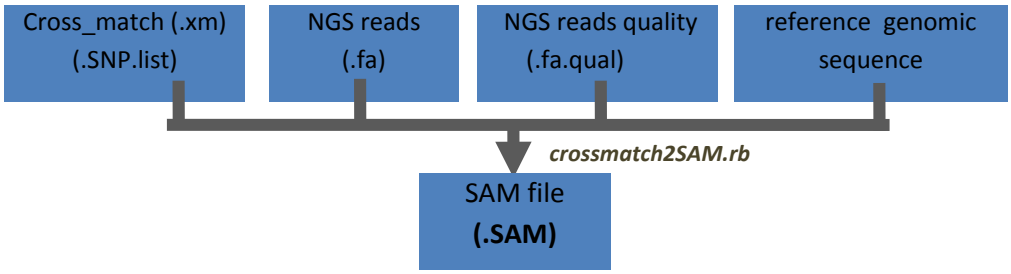

Figure S1.4 cross_match2SAM

2. Usage

2.1 Atlas-SNP-mapper

ruby atlas-mapper-format-ref.rb [-r reference] [-l length of each piece] [-f frequency Cutoff of 11mer] [--help]

"atlas-mapper-format-ref.rb" aims to divide the reference genome into pieces to meet the computational requirements when running "BLAT" and "*cross_match*" respectively. The output will be a newly created directory named "*referenceName*.Environment", under which reference fragments, the fragment index and the relative information about the splitting are placed. Under this directory, there are two sub directories: "ref-pieces" and "ref-divisions". The longer divided reference pieces (900Mb in size) are placed under the sub directory "ref-pieces", which will actually be used for "BLAT" to anchor reads back to respective local regions; the shorter reference pieces (100Kb in size) are stored under the sub directory named as "ref-divisions", which are used to precisely compare the bases of read sequences with those on the reference sequence using "*cross_match*". This program takes two required parameters, "--reference " ("-r"), which introduces the reference sequence, and "--length" ("-l") which is associated with the size of bases each reference piece may contain.

Options:

-r FILE reference genome (required)
-l INT bases a splitted reference piece may contain. With default value 1000000 (required)
-f FLT the cutoff of 11-mer frequency deemed as over-represented in the reference so BLAT will ignore it during seeding step, with default value 1024. 1024 is optimized for mammalian genomes. 100~200 is best for smaller or less complex genomes.
-h usage information

ruby atlas-mapper.rb [-r reference] [-q reads fasta file] [-m minScore] [-c cutoff] [-b blat] [-c crossMatch] [-n oneOff] [-i minIdentity] [-t minMatchRatio] [-l mastLevel] [-z xmOnly] [-a blatOnly] [-e noParse] [-s short]

"atlas-mapper.rb" takes two required parameters. The first is "--reference" ("-r"), which must be followed by the reference sequence name. The second is "--query" ("-q") which is associated with the NGS reads name. By running this wrapping program, users don't have to take efforts in any intermediate steps, such as parsing "BLAT" results, picking the best hits and format conversion between the two tools. Also, the program provides flexibility that allows users to control the pathway. For example, once the option "-z" is appended, the program will only run cross_match, the same will happen for the option "-a" for running "BLAT" only.

Options:

-r FILE reference genome (required)
-q FILE reads fasta file (required)
-m INT parameter of "crossmatch", minimum score with default value 30
-c FLT % for the best hit cutoff with default value 0.99
-b inferring blat executive file
-c inferring crossmatch executive file
-n oneOff set to run "BLAT" with default value 0
-i INT minimum identities set when running "BLAT" with default value 90
-t FLT minimum match ratio when grouping the best hit reads generated by "BLAT" with default value 0.85. Definition: matched bases / reads length
-l INT mask level set when running "crossmatch" with default value 20
-z causing program to run "crossmatch" only
-a causing program to run "BLAT" only
-e causing program to stop right after "BLAT" without parse out the best hit reads.
-s option for aligning short reads using "crossmatch"

Note: for more details on the parameter usage and output format, please refer to the "*cross_match*" and "BLAT" documentation.

2.2 Atlas-SNP-splitter

ruby splitXMFile.rb [-x crossmatch file] [-C split by chromosome] [-n number of parts of the input with mapping on genomic regions of the same size] [-o output file root] [-h help]

"splitXMFile.rb" is a flexible data splitting tool, designed especially for splitting the *cross_match* file to meet the computational requirements for running "Atlas-SNP-core". Besides the regular input *cross_match* file introduced by the parameter "--xmFile" ("-x"), the tool provides other two important parameters—"--splitByChrom" ("-C") and "--numberOfParts" ("-n"). If "-C" is appended, the *cross_match* file will be divided by chromosomes. "-C" is a MUST for large dataset with multiple reference regions like the human genome. Another option "--numberOfParts" ("-n"), is associated with a number defined by users. This option permits the program to further divide the *cross_match* file into the size determined by users. We have used "-n 20" for processing the human genome, for example. Please note that the option "-C" must be used whenever "-n" is used.

Options:

-x FILE crossmatch file (required)
-C causing the program to split crossmatch file by chromosome

-n	INT	number of parts that the crossmatch file will be divided within one chromosome
-o	STR	output file root (required)
-h		usage information

ruby extractReadNamesFromXM.rb [crossmatch file] [output file]

"extractReadNamesFromXM.rb" extracts all the read names from the *cross_match* files and places a list of these names into an output file. This name list is used as a small trace id database to be queried by the program to pick the corresponding reads. The program takes two arguments, the *cross_match* file name and an output file name.

ruby filterFastaByListOfReads.rb [-f fasta file of reads] [-l file containing the list of reads to be selected] [-o outputPrefix] [-c fasta file chunk size, in number of sequences]

"filterFastaByListOfReads.rb" is used to filter those NGS reads by choosing only the reads in the trace ids database created by "extractReadNamesFromXM.rb". By only considering the aligned reads, the time and memory usage for running "atlasNQSPASS.rb" can be largely reduced. The program takes three required parameters: "--fastaFiles" ("-f") for the reads fasta file; "--readListFile" ("-l") that is followed by the "trace id database" file created by "extractReadNamesFromXM.rb"; and "--outFile" ("-o") that specifies the output file name. The program will generate a .gz fasta file containing only the sequences of the aligned reads.

Options:

-f	FILE	reads fasta file (required)
-l	STR	the output file of "extractReadNamesFromXM.rb" (required)
-o	STR	the output file prefix (required)
-c	INT	fasta file chunk size (optional)
-h		usage information

2.3 Atlas-SNP-SAM

ruby crossmatch2SAM.rb [-i cross_match] [-o output file] [-r read sequence] [-q read quality file] [-f reference] [-o output]

For each aligned read, the program parses out the mismatched substitutions and indels from *cross_match* results. In the meantime, the program keeps querying subsequences from both reference sequence and raw reads file (sequence and quality file). The program takes the cross_match file ("-i"), read sequences ("-r"), the read quality file ("-q") and the reference sequence file ("-f") as input files, and then produce a SAM format mapping file as the output file (see Table4).

Option

-i	FILE	cross_match file (required)
-o	STR	output file indicated by users (required)
-r	FILE	NGS read sequence file (required)
-q	FILE	NGS read quality file (required)
-f	FILE	reference sequence file (required)
-h		usage information