Software: the good, the bad and the ugly

Festival of Genomics 2017

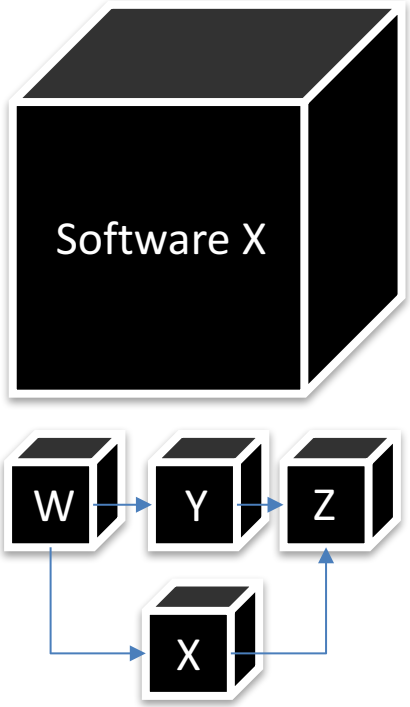
Russell Hamilton
rsh46@cam.ac.uk

Babraham Institute

UNIVERSITY OF CAMBRIDGE

Babraham Bioinformatics

Bioinformatics Software



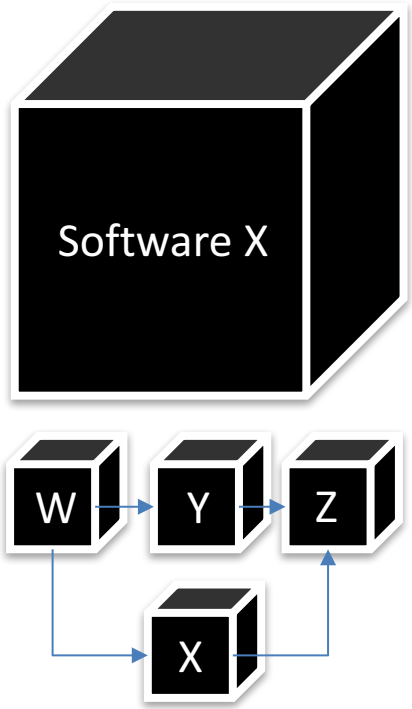
All software contains bugs:

- *Industry Average: "about 15 - 50 errors per 1000 lines of delivered code"*
  Steve McConnell (author of Code Complete and Software Estimation: Demystifying the Black Art)
- Range from spelling mistake in error message to completely incorrect results
- Most software will process the input to produce an output without errors or warnings

Never blindly trust software or pipelines

- Always **test** and **validate** results
- Avoid **black box** software (*definition: produces results, but no one knows how*)

Different classes of bioinformatics software



Class	Examples	Description
Processing	TopHat2, Bowtie2	Performing computationally intensive task, applying mathematical models
Evaluation	FastQC, BamQC	Deriving QC metrics from output files
Converters	SamToFastq (Picard Tools)	Simply converting between file formats. Generally stable no regular updates
Pipelines	Galaxy, ClusterFlow	The glue for joining software to create an automated pipeline

12 Step Guide for evaluating and selecting bioinformatics software tools

1. Finding Software to do the job
2. Has the software been published?
3. Software Availability
4. Documentation Availability
5. Presence on user groups
6. Installation and Running
7. Errors and Log Files
8. Use standard file formats
9. Evaluating Commercial Software
10. Bugs in scripts / pipelines to run software
11. Writing your own software
12. Using and creating pipelines

1. Finding software to do the job

Identify the required task

Alignment of **methylation** sequencing data to reference genome

Are there related studies performing similar analysis?
Publication / posters / talks

Developmental Cell
Resource
Global Landscape and Regulatory Principles of DNA Methylation Reprogramming for Germ Cell Specification by Mouse Pluripotent Stem Cells

Kenjiro Shirane,[1,2] Kazuki Kurimoto,[3,4] Yukihiro Yabuta,[3,4] Masashi Yamaji,[3,4,10] Junko Satoh,[5] Shinji Ito,[5] Akira Watanabe,[6,7] Katsuhiko Hayashi,[3,8,9] Mitinori Saitou,[3,4,6,7,*] and Hiroyuki Sasaki[1,11,*]

Required features

Must Have
1. INPUT standard FASTQ format files
2. OUTPUT standard BAM alignments
3. OUTPUT Compatible with methylKit

Like to have
1. Perform methylation calls
2. Must make use of multi processors for large numbers of samples

2. Has the software been published?

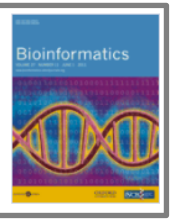
- ✓ Published in a peer reviewed journal
  As stand alone software or part of study
- ✓ Cited by other peer reviewed papers
- ✓ Has the software been benchmarked
  (by other people than the authors)

Short read mapping is "generally solved problem"
Informative for run times

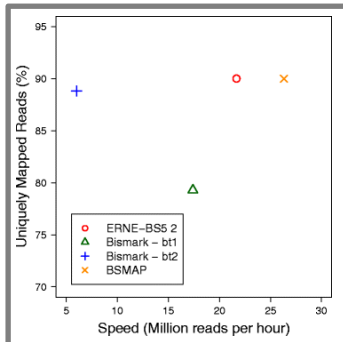
Bismark: a flexible aligner and methylation caller for Bisulfite-Seq applications

Felix Krueger; Simon R. Andrews

Bioinformatics (2011) 27 (11): 1571-1572.
DOI: https://doi.org/10.1093/bioinformatics/btr167



Bismark: a flexible aligner and methylation caller for Bisulfite-Seq applications
F Krueger, SR Andrews - Bioinformatics, 2011 - Oxford Univ Press
Summary: A combination of bisulfite treatment of DNA and high-throughput sequencing (BS-Seq) can capture a snapshot of a cell's epigenomic state by revealing its genome-wide cytosine methylation at single base resolution. Bismark is a flexible tool for the time-efficient
Cited by 586 Related articles Cite Save More



BMC Bioinformatics; 2016; 17(Suppl 4):69
DOI: 10.1186/s12859-016-0910-3

Babraham Institute

UNIVERSITY OF CAMBRIDGE

Babraham Bioinformatics

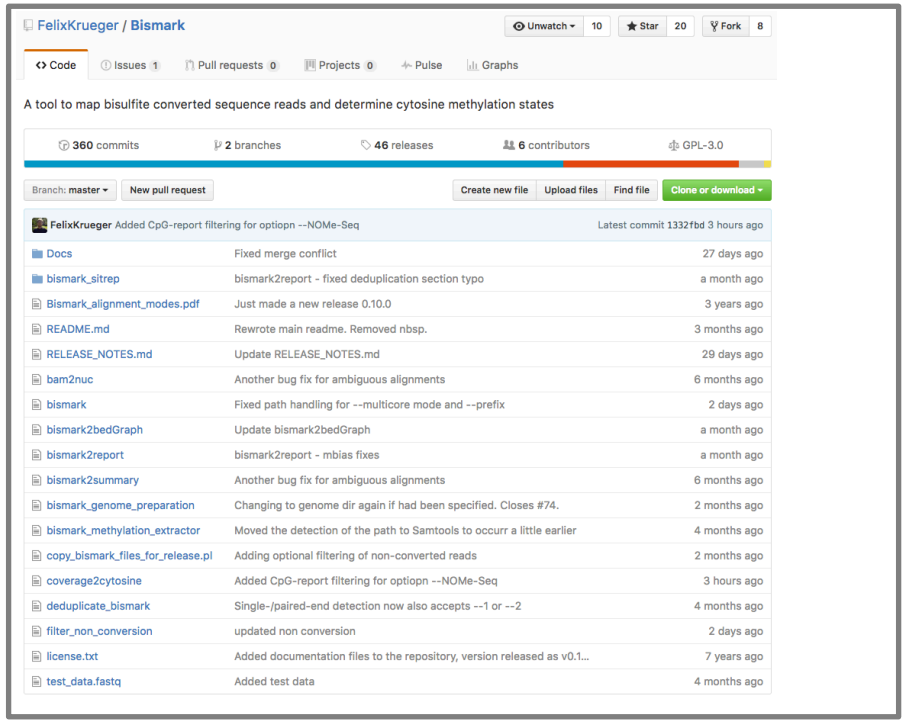
3. Software Availability

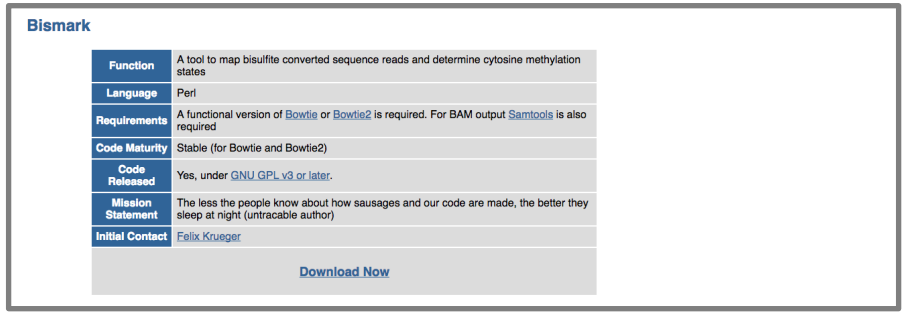
✓ Software available for download
Hosted on a recognised software repository
e.g. GitHub, BitBucket, SourceForge

✓ Software regularly updated / bugs fixed / releases
More than one developer (e.g. group account)

Permanent archive of software releases
e.g. zenodo.org, figshare.com

✓ University / Institute / Company Web site
Software is the responsibility of a group not just an individual





Babraham Institute

UNIVERSITY OF CAMBRIDGE

Babraham Bioinformatics

3. Software Availability

✓ Bugs are reported and fixed

New feature requests are added

Check for truncated BAM files #50

Closed **ewels** opened this issue on Jun 10, 2016 · 4 comments

ewels commented on Jun 10, 2016 Contributor

Truncated BAM files are super scary - not only is data lost, but because the read pairing gets messed up, incorrect methylation calls can be generated.

BAM files should have EOF tags, making it possible to detect when they have been truncated. Could Bismark deduplication / methextraction steps check for this and either bail or throw a big scary warning if they're not found?

FelixKrueger commented on Sep 6, 2016 Owner

Added EOF detection for `deduplicate_bismark` (`ace0f35`) and `bismark_methylation_extractor` (`3901385`). Hope it will stop at least some of the most serious truncations.

4. Documentation Availability

✓ User Documentation

✓ Release Documentation
Versions – aids reproduceability

Bismark Bisulfite Mapper

User Guide - v0.16.3

14 December, 2016

This User Guide outlines the Bismark suite of tools and gives more details for each individual step. For troubleshooting some of the more commonly experienced problems in sequencing in general and bisulfite-sequencing in particular please browse through the sequencing section at QCFail.com.

Changelog

- 18-01-2017: 0.17.0 released (click here for the Release Notes hosted on Github)
- 25-07-2016: 0.16.3 released (click here for the Release Notes hosted on Github)
  - Bismark: Essential fixes (2 in total) to address a bug for Bowtie 2 alignments where reads that should be considered ambiguous were incorrectly assigned to the first alignment thread. These errors had crept in during releases 0.16.0 and 0.16.2). More info available on Github
  - Bismark: Added support for large Bowtie (1) index files ending in .ebwtl which had been added in Bowtie v1.1.0
  - Changed the Shebang in all scripts of the Bismark suite to #!/usr/bin/env perl instead of #!/usr/bin/perl
  - deduplicate_bismark: Does now bail with a useful error message when the input files are empty
  - bismark_genome_preparation: Added new option '--genomic_composition' so that the genomic composition can be calculated and written right at the genome preparation stage rather than by using bam2nuc
  - bam2nuc: Now also calculates a fold coverage for the various (di-)nucleotides. The changes in the nucleotide_stats text file are also picked up and plotted by bismark2report
  - bam2nuc: Added a new option '--genomic_composition_only' to just process the genomic sequence without requiring any data files
  - bismark2summary: Added option -o/--basename FILENAME to specify a certain filename. If not specified the name will remain bismark_summary_report.txt/html
  - bismark2summary: Added documentation and the options '--help' and '--version' to be consistent with the rest of Bismark
  - bismark2summary: Added option '--title STRING' to give the HTML report a different title

4. Documentation Availability

Example: RNA-Seq Differential Gene analysis using DESeq2 Discover limitations

Name	*fileName*	*genotype*
WT1	wt1.htseq_counts.txt	WT
WT2	wt2.htseq_counts.txt	WT
WT3	wt3.htseq_counts.txt	WT
KO1.1	ko1.1.htseq_counts.txt	KO1
KO1.2	ko1.2.htseq_counts.txt	KO1
KO1.3	ko1.3.htseq_counts.txt	KO1
KO2.1	ko2.1.htseq_counts.txt	KO2
KO2.2	ko2.2.htseq_counts.txt	KO2
KO2.3	ko2.3.htseq_counts.txt	KO2
KO3.1	ko3.1.htseq_counts.txt	KO3
KO3.2	ko3.2.htseq_counts.txt	KO3
KO3.3	ko3.3.htseq_counts.txt	KO3

	WT	KO1	KO2	KO3
WT		X	X	**X**
KO1			X	X
KO2				X
KO3				

DESeq2 Manual

"The `results` function without any arguments will automatically perform a contrast of the last level of the last variable in the design formula over the first level."

```
count.data      <- DESeqDataSetFromMatrix(sampleTable=smplTbl,
                                          design= ~ genotype)
count.data      <- DESeq(count.data)
```

✗ `binomial.result <- results(count.data)`

✓ `binomial.result <- results(count.data, contrast=c("genotype","K01","K02"))`

5. Presence on user groups

✓ Evidence for support questions being answered e.g. FAQ, searchable public support group

http://seqanswers.com

https://www.biostars.org

GitHub Issues

Google Groups

Is there someone near by you can ask for help

Bioinformatics Core Facility

Research group down the corridor

6. Installation and Running

✓ Will run on standard architecture

✓ Easy to install

✓ Release versions

✓ Default parameters

✓ Source code available
Binaries can simplify installation

Docker/Galaxy/BaseSpace

Installation notes

Bismark is written in Perl and is executed from the command line. To install Bismark simply copy the bismark_v0.X.Y.tar.gz file into a Bismark installation folder and extract all files by typing:

```
tar xzf bismark_v0.X.Y.tar.gz
```

```
$ bismark --version
        Bismark - Bisulfite Mapper and Methylation Caller.
                Bismark Version: v0.16.3_dev
    Copyright 2010-15 Felix Krueger, Babraham Bioinformatics
        www.bioinformatics.babraham.ac.uk/projects/
```

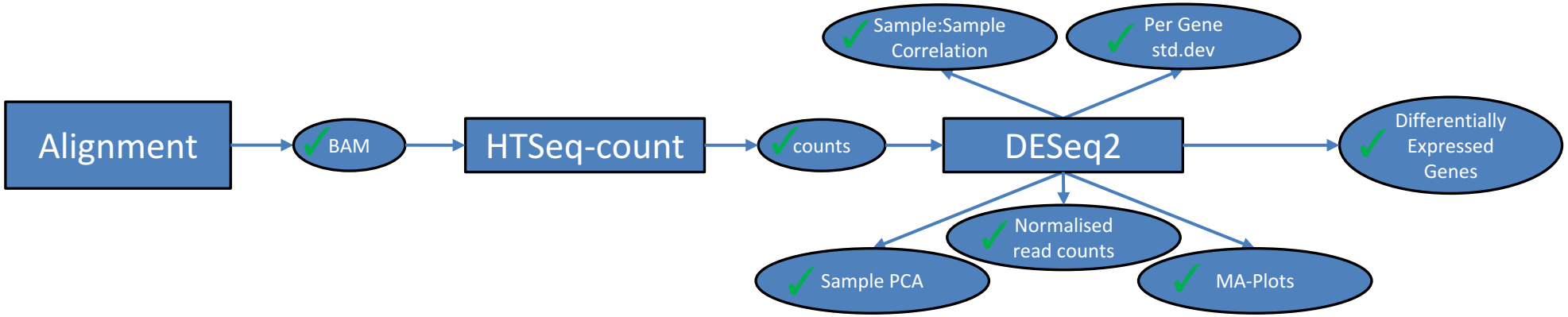
A sensible set of default parameters that are likely to produce a good first pass at the results

Perl

6. Installation and Running

Example: Traceability of results though the steps in the analysis

Intermediate results are excellent check points



RNA-Seq Differential Gene Expression Analysis

7. Errors and Log Files

Keep and read log files for software run

Warnings

Don't ignore warnings, they may be telling you something crucial about your data

Errors

Problem severe enough for the program to stop and produce an error

8. Use standard file formats

Bioinformaticians spend an embarrassing amount of time converting between file formats

✓ Standard Input Files

FASTA, FASTQ

Converting between formats could introduce errors

✓ Standard Output Files

BAM

Compatible with downstream tools

9. Evaluating Commercial Software

Should you use commercial software to do RNA-Seq DGE analysis?

Lots of good commercial software available e.g. Partek

Pros	Cons
Graphical Interface – no command line	Run analysis without understanding the steps
Single application for all steps	Harder to trace back step by step
Dedicated Customer Support	Limited user group activity
	Less transparency (methods / bugs fixed)
	Expensive
	License required to reproduce analysis (e.g. reviewers)

10. Bugs in scripts / pipelines to run software

Often written specifically for each analysis or project and are prone to bugs

Examples of accidentally missing out samples

1. Bash Script for running fastQC

```
for file in *_1.fq.gz;
do
  fastqc $file
done

multiqc .
```

2. RNA-Seq DESeq2 Sample Table

```
genotype <- data.frame(
            'WT', 'wt', 'Wt', 'K01','k01','K01',
            'K02','Ko2','K02','K03','K03','K03')
...
results(dds, contrast=c("genotype","WT","K02"))
```

11. Utilising dedicated Pipeline tools

The bad and ugly

- Home made "glue" scripts for running software can be bug prone
- "dark script matter" isn't reviewed or assesses and rarely released in methods sections
- In a 3000 sample study, errors are propagated 3000 times!

The good

- Purpose build pipeline tools
- Premade pipelines for e.g. RNA-Seq differential gene expression
- Job queuing - Load balancing across hardware (laptop to cluster farm)
- Log files track a samples progress through pipeline

11. Utilising dedicated Pipeline tools

Galaxy https://usegalaxy.org/	Interaction via a web browser Public and private server installs Many pre-built pipelines Large user community
CF Cluster Flow http://clusterflow.io/	Command line interface Many pre-built pipelines
CWL Common Workflow Language https://github.com/common-workflow-language	A language for building your own pipelines Utilised by other pipeline tools e.g. NextIO

12. Developing your own software

If you are sure a great piece of software doesn't already exist or can be modified for the task

Developing your own tools gives an appreciation of how difficult it can be

PLOS | COMPUTATIONAL BIOLOGY

EDITORIAL

Ten Simple Rules for Developing Usable Software in Computational Biology

Markus List[1]*, Peter Ebert[1,2]*, Felipe Albrecht[1,2]

1 Computational Biology and Applied Algorithmics, Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany, 2 Graduate School of Computer Science, Saarland Informatics Campus, Saarbrücken, Germany

Rule 1: Identify the Missing Pieces
Rule 2: Collect Feedback from Prospective Users
Rule 3: Be Ready for Data Growth
Rule 4: Use Standard Data Formats for Input and Output
Rule 5: Expose Only Mandatory Parameters
Rule 6: Expect Users to Make Mistakes
Rule 7: Provide Logging Information
Rule 8: Get Users Started Quickly
Rule 9: Offer Tutorial Material
Rule 10: Consider the Future of Your Tool

Weighting the evaluation criteria

	Criteria	Importance	Comments
1	Finding Software to do the job	+++++	Use the right tools for the job
2	Has the software been published?	+++	New software being released so check for improved methods. Just because its published and well used doesn't mean it's still the best
3	Software Availability	++	Openness it a good sign for finding error / bugs / suggesting feature enhancements
4	Documentation Availability	+++++	
5	Presence on user groups	+++++	
6	Installation and Running	+++	
7	Errors and Log Files	+++++	
8	Use standard file formats	++++	Conversions could add sources of error
9	Evaluating Commercial Software	+	Price Vs Open source software
10	Bugs in scripts / pipelines to run software	+++++	Pipelines standardise workflows
11	Utilising dedicated pipeline tools	+	
12	Writing your own software	+	Don't re-inventing the wheel

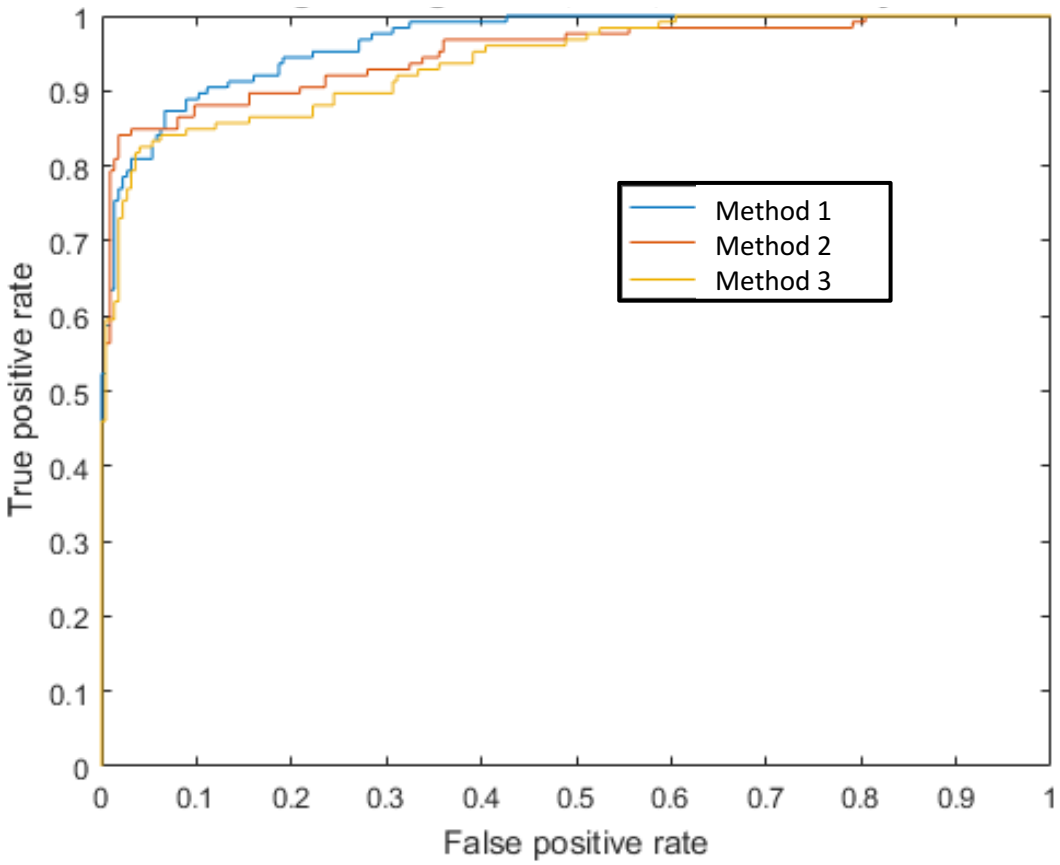
Weighting the evaluation criteria

Compromises for run time vs accuracy/sensitivity

Project A has 3000 samples vs Project B with 12 samples

Method 1: 4 hours per sample 98% accuracy
Method 2: 30 mins per sample 97% accuracy

What would you choose if
Method 1: 4 hours per sample 98% accuracy
Method 3: 15 mins per sample **90**% accuracy



Summary

Many ways to evaluate software

- Openness and engagement with users is very important
  - bugs fixed, features added, large user base

- Evaluate features, e.g. run time, against your project requirements

- If you are using pipelines, use purpose build pipelining tools