

# Community developed variant calling pipelines

Brad Chapman

Bioinformatics Core, Harvard School of Public Health

<https://github.com/chapmanb>

22 October 2013

# Complex, rapidly changing pipelines

## Whole genome, deep coverage v1

---

Warning: the material on this page is considered out of date by the GSA team.

## Best Practice Variant Detection with the GATK v2

---

Warning: the material on this page is considered out of date by the GSA team.

## RETIRED: Best Practice Variant Detection with the GATK v3

---

## Best Practice Variant Detection with the GATK v4, for release 2.0



**Mark\_DePristo** Posts: 150 Administrator, GSA Official Member admin  
July 2012 edited April 24 in [Methods and Workflows](#)

### HaplotypeCaller now so sensitive, it cries at the movies

We know you don't want to miss a single true variant, so for this release, we've put a lot of effort into making the HaplotypeCaller more sensitive. And it's paying off: in our tests, the HaplotypeCaller is now more sensitive than the UnifiedGenotyper for calling both SNPs and indels when run over whole genome datasets.

# Large number of specialized dependencies

```
#####  
# HugeSeq                                     #  
# The Variant Detection Pipeline             #  
#####
```

```
-- DEPENDENCIES
```

```
+ ANNOVAR version 20110506  
+ BEDtools version 2.16.2  
+ BreakDancer version 1.1  
+ BreakSeq Lite version 1.3  
+ BWA version 0.6.1  
+ CNVnator version 0.2.2  
+ GATK version 1.6-9  
+ JDK version 1.6.0_21  
+ Modules Release 3.2.8  
+ Perl  
+ Picard Tools version 1.64  
+ Pindel version 0.2.2  
+ Plantation version 2  
+ pysam version 0.6  
+ Python version 2.7  
+ Simple Job Manager version 1.0  
+ Tabix version 0.1.5  
+ VCFtools version 0.1.5
```

<https://github.com/StanfordBioinformatics/HugeSeq>

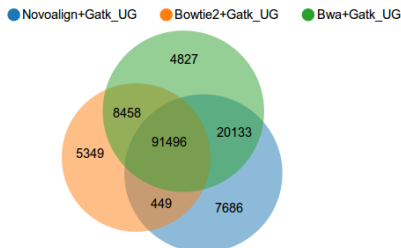
# Quality differences between methods

## Variant Calling Test

Discuss

We compare combinations of variant calling pipelines across different data sets. Browse our public facing reports to see how various aligner + variant caller combinations perform against each other. Test your own combination of tools by creating your own report. Below is a sample concordance view on our "Illumina 100bp Paired End 30x Coverage" data set.

### Variant Concordance - "illumina-100bp-pe-exome-30x"



<http://www.bioplanet.com/gcat>

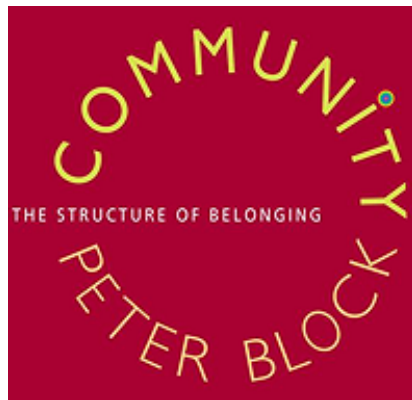
# Scaling on full ecosystem of clusters



Platform LSF

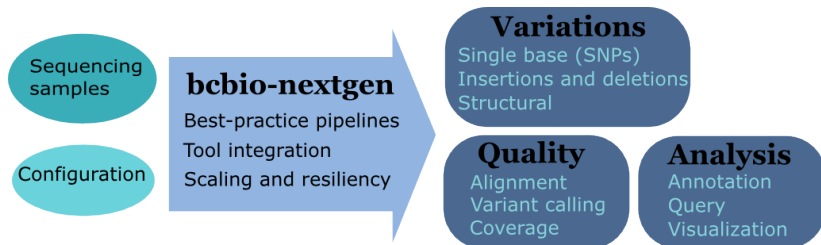
Torque

# Solution



<http://www.amazon.com/Community-Structure-Belonging-Peter-Block/dp/1605092770>

# Overview

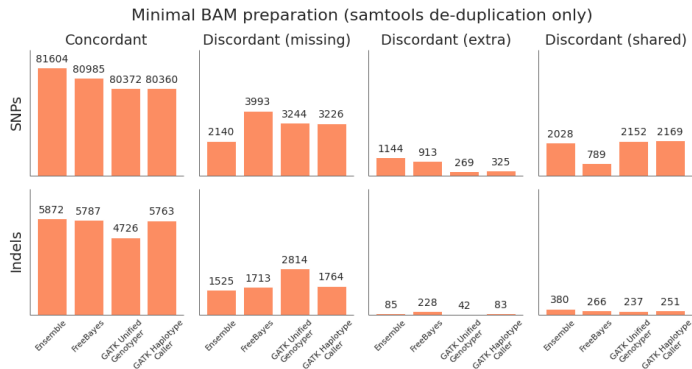


# Development goals

- Quantifiable
- Analyzable
- Reproducible
- Scalable
- Community developed
- Accessible



# Quantify quality



- Reference materials: <http://www.genomeinabottle.org/>
- Quantification details: <http://j.mp/bcbioeval2>

## Query and analyze



## Table Of Contents

### Querying the GEMINI database

- Basic queries
- Selecting sample genotype
- Filtering on genotypes
- Finding out which samples

[Previous topic](#)

### Loading a VCF file into GEMIN

[Next topic](#)

### Built-in analysis tools

### Querying the GEMINI database

The real power in the GEMINI framework lies in the fact that all of your genetic variants have been stored in a convenient database in the context of a wealth of genome annotations that facilitate variant interpretation. The expressive power of SQL allows one to pose intricate questions of one's variation data.

### Note

If you are unfamiliar with SQL, [sqlzoo](#) has a decent online tutorial describing the basics. Really all you need to learn is the SELECT statement, and the examples below will give you a flavor of how to compose basic SQL queries against the GEMINI framework.

## Basic queries

GEMINI has a specific tool for querying a gemini database that has been load''ed using the ``gemini load command. That's right, the tool is called `gemini query`. Below are a few basic queries that give you a sense of how to interact with the gemini database using the `query tool`.

1. Extract all transitions with a call rate  $\geq 95\%$ .

```
$ gemini query -q "select * from variants \
                    where sub_type = 'ts' \
                    and call_rate >= 0.95" my.db
```

GEMINI: <https://github.com/arq5x/gemini>

# Automated installation

- Single biggest software problem: running for the first time
- Bootstrap from bare machine to ready-to-go pipeline
- Builds off existing installation work: CloudBioLinux, Homebrew
- Real life benchmark example data

<http://cloudbiolinux.org>

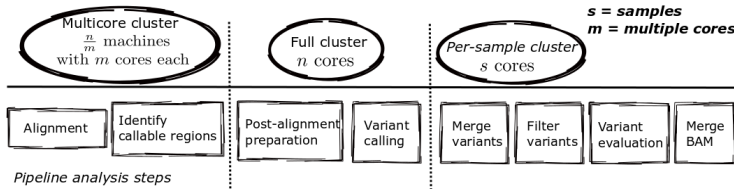
<https://bcbio-nextgen.readthedocs.org>

- Current approaches
  - Full logging – command lines and debugging
  - Third party version tracking
  - Virtual machines
- Long term plans
  - Keep
  - Metadata + traceability

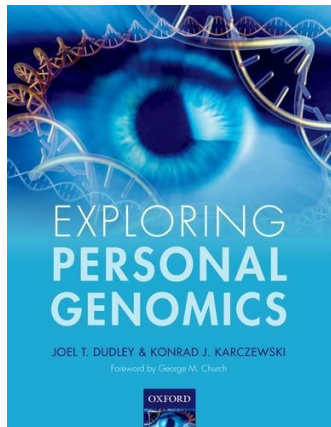
# Parallel scaling

## Heterogeneous cluster creation during variant calling

$n$  = total cores  
 $s$  = samples  
 $m$  = multiple cores



- Infrastructure details: <http://j.mp/bcbioscale>
- IPython: <http://ipython.org/ipython-doc/dev/parallel/index.html>
- Crunch: compute + distributed filesystems



<http://exploringpersonalgenomics.org/>

# Summary

- Community developed pipelines > challenges
- Focus
  - Assessing quality: good science
  - Analysis: enable exploration
  - Scalability: finish in time
  - Reproducibility: show your work
- Widely accessible

<https://github.com/chapmanb/bcbio-nextgen>