

# Adoption of D for genomics bioinformatics

## Dconf 2018, Munich

Vang Le

Aalborg University Hospital  
<http://www aalborguh.dk>

May 4, 2018



AALBORG UNIVERSITY HOSPITAL

# Adoption roadmap

- ➊ Do scripting as Bash alternative (lib: Scriptlike)
- ➋ Build analysis pipeline (Python, Bash, Java/Scala are doing fine)
- ➌ Analyze data in BAM, VCF format (variant calling, chromosome structural variation, count BAM statistics).
- ➍ Port and develop new tools in D (to learn and compare)
- ➎ Develop GUI and commandline applications for end users
- ➏ Make D the main language to power big data analysis

# Agenda

- ① Introduction
- ② Characteristics of genomic bioinformatics
- ③ Bioinformaticians vs Programming languages
- ④ Relevance of D for genomic bioinformatics
- ⑤ Current status of adoption
- ⑥ How D community and bioinformatics can help each other
- ⑦ Take-home messages
- ⑧ Coding challenges for fun

# Ultra Short CV

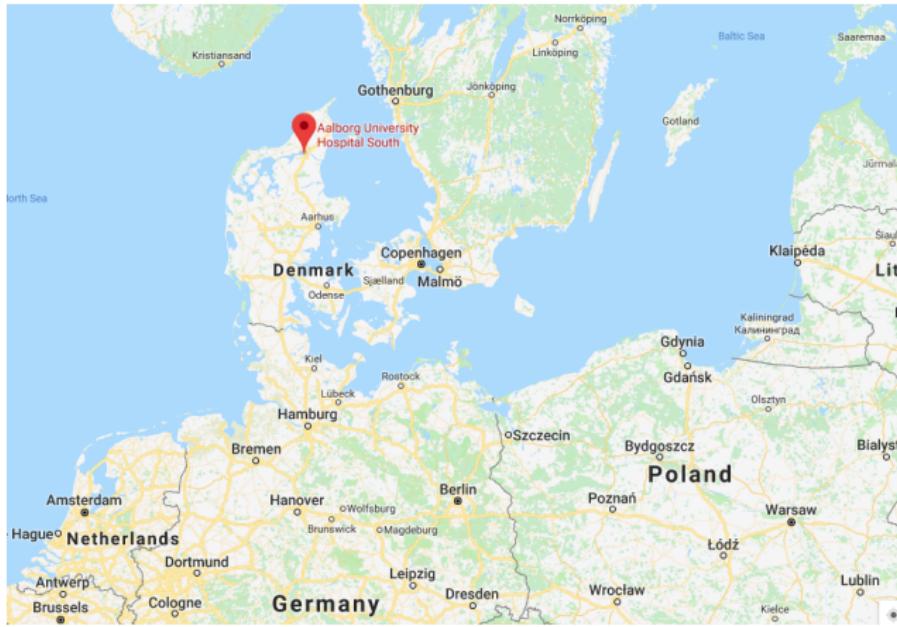
- 20 yrs biologist, 12 yrs active linux admin/user, 9 yrs bioinformatician. Bash, Python, R, C, C++, Java, Scala, D.

# Ultra Short CV

- 20 yrs biologist, 12 yrs active linux admin/user, 9 yrs bioinformatician. Bash, Python, R, C, C++, Java, Scala, D.
- Perfect opportunity of D-bioinformatics: interest, collaborations, jobs <-> tools, libraries, learning resources

# Aalborg University Hospital

- 755 beds. 1020 doctors. 6000 employees.



# Section of Molecular Diagnostics



# Section of Molecular Diagnostics

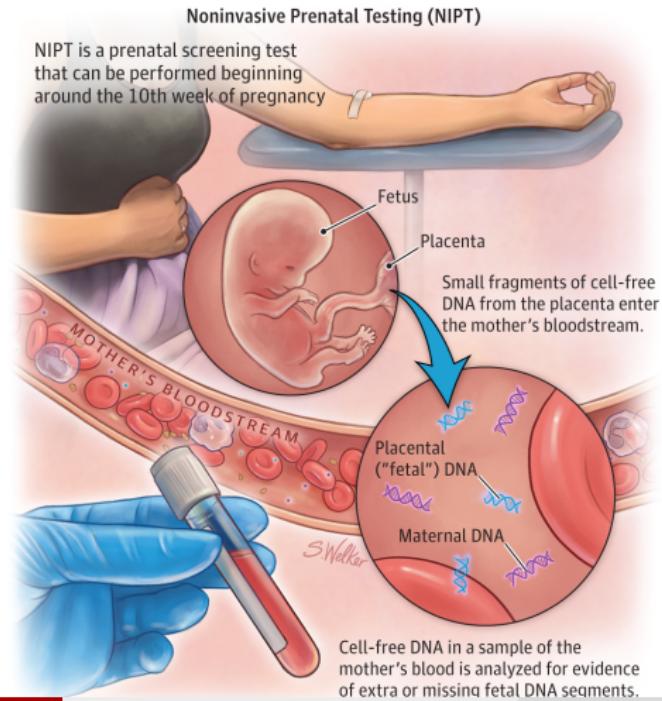
- Infrastructure: 10 Linux servers/clusters (total: 300TB storage, 1.2TB RAM, 3.5TFLOPS, 2 Tesla K20c, 6 M2070)

# Section of Molecular Diagnostics

- Infrastructure: 10 Linux servers/clusters (total: 300TB storage, 1.2TB RAM, 3.5TFLOPS, 2 Tesla K20c, 6 M2070)
- Data volume: 8015 (2017) including Gene Panel, Exome, Whole Genome. Generate ~4TB/month raw data. Keep ~1TB.

# Section of Molecular Diagnostics

- Applications: NIPT, PGS, genetic disorders, cancer, virus genotyping



# Bioinformatics in one slide

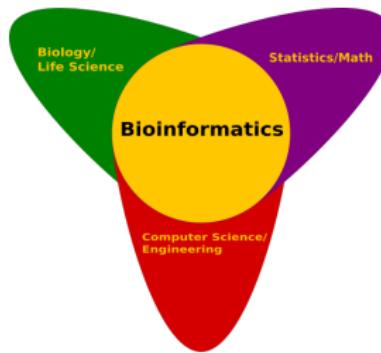
Bioinformatics ?

# Bioinformatics in one slide

Bioinformatics ?

# Bioinformatics in one slide

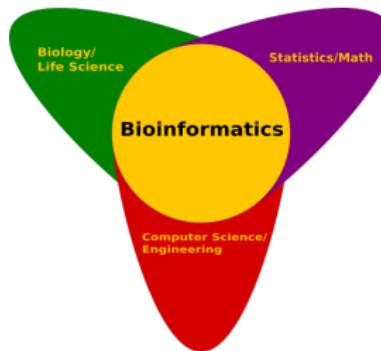
Bioinformatics ?



Computation Tasks:

# Bioinformatics in one slide

Bioinformatics ?

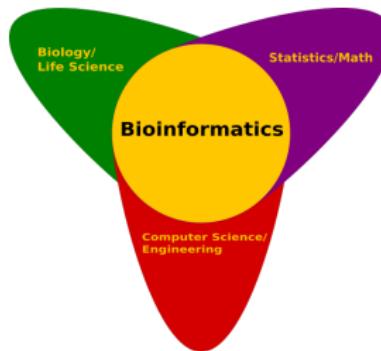


Computation Tasks:

- Input: Text, Image, Database

# Bioinformatics in one slide

Bioinformatics ?

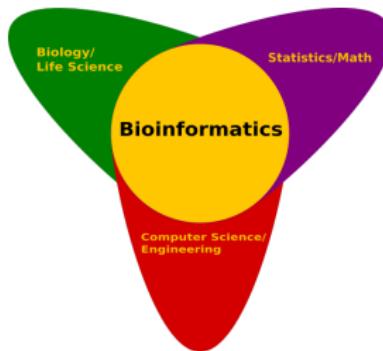


Computation Tasks:

- Input: Text, Image, Database
- Processing: Query, Pattern, Alignment, Comparison, Clustering, Classification, Statistics -> Parallel, Distributed

# Bioinformatics in one slide

Bioinformatics ?



Computation Tasks:

- Input: Text, Image, Database
- Processing: Query, Pattern, Alignment, Comparison, Clustering, Classification, Statistics -> Parallel, Distributed
- Output: Graphs, Summary -> Evaluation, Revision, Decision

# Genomics in Multi-omics context

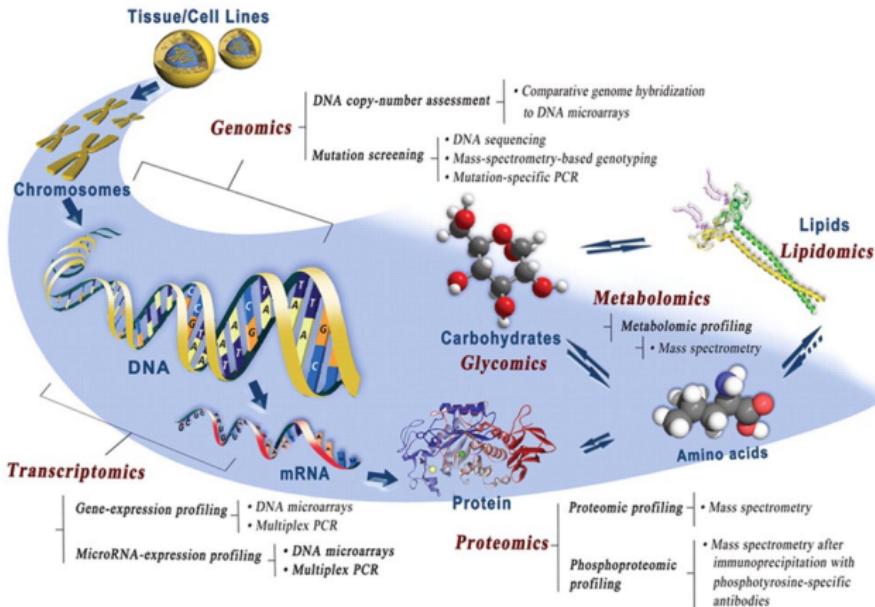


Figure: Multiomics approach, not mentioning epigenomics (Image with unknown credit)

# Genomics Data Volume

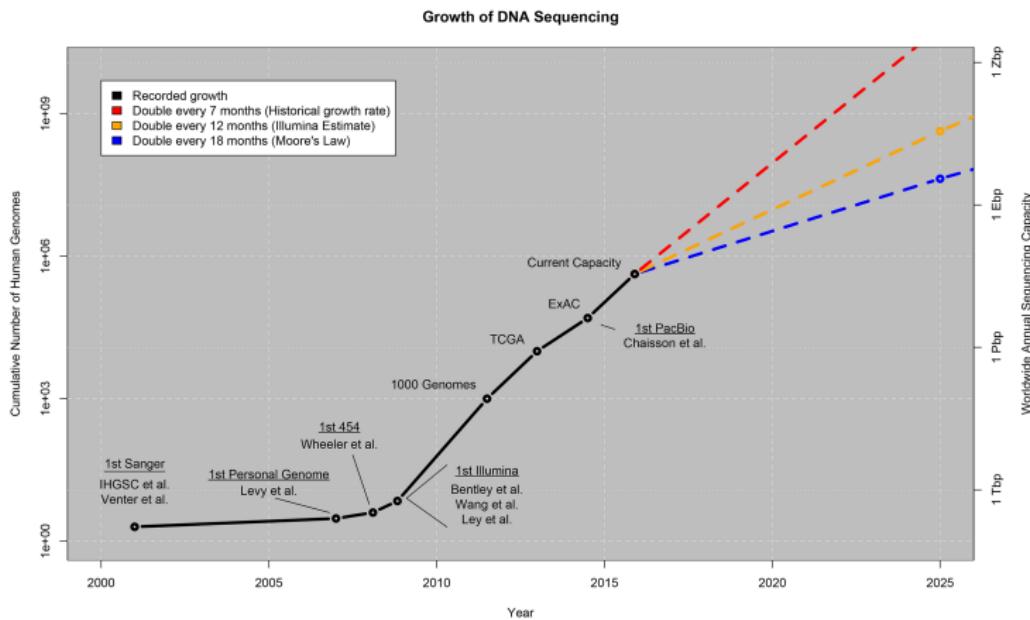


Figure: Genomics data volume (<https://doi.org/10.1371/journal.pbio.1002195>)

# Genomics Data Volume



# Applications and researchs of Bioinformatics

- Medicine

- personal medicine
- cancer research
- genetic disorder
- Non-invasive prenatal testing
- Preimplantation genetic screening

# Applications and researchs of Bioinformatics

- Medicine
  - personal medicine
  - cancer research
  - genetic disorder
  - Non-invasive prenatal testing
  - Preimplantation genetic screening
- Biology

# Applications and researchs of Bioinformatics

- Medicine
  - personal medicine
  - cancer research
  - genetic disorder
  - Non-invasive prenatal testing
  - Preimplantation genetic screening
- Biology
- Agriculture

# Applications and researchs of Bioinformatics

- Medicine
  - personal medicine
  - cancer research
  - genetic disorder
  - Non-invasive prenatal testing
  - Preimplantation genetic screening
- Biology
- Agriculture
- Astrobiology

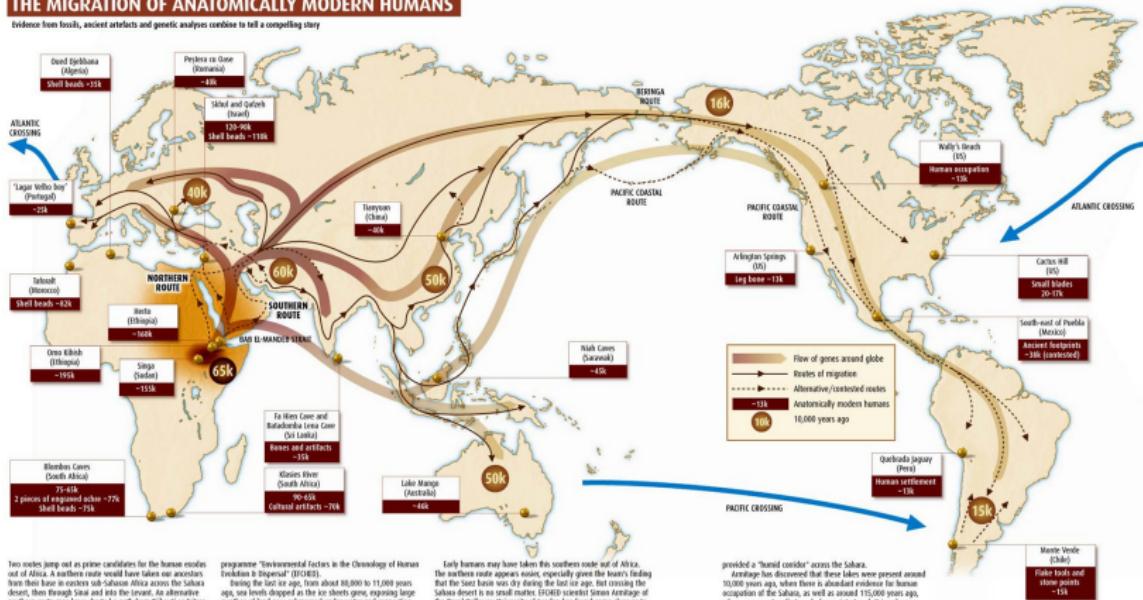
# Applications and researchs of Bioinformatics

- Medicine
  - personal medicine
  - cancer research
  - genetic disorder
  - Non-invasive prenatal testing
  - Preimplantation genetic screening
- Biology
- Agriculture
- Astrobiology
- Archaeology

# Applications and researchs of Bioinformatics

## THE MIGRATION OF ANATOMICALLY MODERN HUMANS

Evidence from fossils, ancient artefacts and genetic analyses combine to tell a compelling story



Two routes jump out as prime candidates for the human exodus out of Africa. A northern route would have taken our ancestors northward through the Levant, crossing the Taurus and Zagros deserts, then through Anatolia and into the steppes. An alternative southern route may have charted a path from Djibouti or tributaries of the Nile down the eastern coast of Africa, through the Horn of Africa and around the Arabian peninsula. The plausibility of these two routes as gateways out of Africa has been studied as part of the UK's Natural Environment Research Council's

programme "Environmental Factors in the Chronology of Human Evolution & Dispersal" (ECHO). The Horn of Africa, with its long dry season, is not a smooth ride, especially given the terrain. In the last glacial maximum, about 30,000 to 11,000 years ago, sea levels dropped as the ice sheets grew, exposing large swathes of land now submerged under water and connecting the Red Sea to the Mediterranean. At the end of the last ice age, the ECHO team found that the Bab el-Mandeb strait, now around 30 kilometers wide and one of the world's busiest shipping lanes, was then a narrow, shallow channel.

Early humans may have taken this southern route out of Africa. The southern route appears plausible, especially given the terrain. The Sahara desert is no smooth ride, especially given the terrain. The last glacial maximum, about 30,000 to 11,000 years ago, sea levels dropped as the ice sheets grew, exposing large swathes of land now submerged under water and connecting the Red Sea to the Mediterranean. At the end of the last ice age, the ECHO team found that the Bab el-Mandeb strait, now around 30 kilometers wide and one of the world's busiest shipping lanes, was then a narrow, shallow channel.

provided a "human corridor" across the Sahara. Arribalzaga has discovered that these lakes were present around 100,000 years ago, which provides a timeline for human occupation of the Sahara, as well as around 115,000 years ago, when our ancestors first made forays into travel. It is unknown whether these early forays were successful, but between 40,000 and 50,000 years ago, the most likely time frame for the human exodus. Moreover, accumulating evidence is pointing to the southern route as the most likely jumping-off point.

# Tasks in Genomic Bioinformatics

- Manage Storage: Acquire, Process (primary), Structure, Associate metadata

# Tasks in Genomic Bioinformatics

- Manage Storage: Acquire, Process (primary), Structure, Associate metadata
- Mine Data: Query, extract, do statistics, cluster, classify, plot

# Tasks in Genomic Bioinformatics

- Manage Storage: Acquire, Process (primary), Structure, Associate metadata
- Mine Data: Query, extract, do statistics, cluster, classify, plot
- Manage metadata: metadata, tracking (materials, outputs, reports)

# Diverse tools in genomic bioinformatics (tiny portion)

- BLAST(C++), Seqan(C++), bwa(C), samtools(C), GATK(Java, Scala), FastQC(Java), Picard(Java, Scala), sambamba(D)
- Many Python (635 packages) and R(Bioconductor:1477) packages: matplotlib, pandas, numpy, limma, edgeR, biomaRt
- conda: fantastic production grade virtual environment. bioconda channel with 8400 packages
- Heavy-lifting frameworks and libraries: HTSlip(C), hail-is/hail(Scala), Cromwell (runs WDL)(Scala), bioD(D), Apache Spark(Scala), HDF5, ROOT, Boost

# Programming languages used in bioinformatics

- bio<insert<sub>anylanguage</sub>>

# Programming languages used in bioinformatics

- bio<insert<sub>anylanguage</sub>>
  - biostatistics

# Programming languages used in bioinformatics

- bio<insert<sub>anylanguage</sub>>
  - biostatistics
  - biochemistry

# Programming languages used in bioinformatics

- bio<insert<sub>anylanguage</sub>>
  - biostatistics
  - biochemistry
  - biophysics

# Programming languages used in bioinformatics

- bio<insert<sub>anylanguage</sub>>
  - biostatistics
  - biochemistry
  - biophysics
  - bioLinux

# Programming languages used in bioinformatics

- bio<insert<sub>anylanguage</sub>>
  - biostatistics
  - biochemistry
  - biophysics
  - bioLinux
  - biohazard :)

# Programming languages used in bioinformatics

- bio<insert<sub>anylanguage</sub>>
  - biostatistics
  - biochemistry
  - biophysics
  - bioLinux
  - biohazard :)
- C, Java, Python, R, Bash, C++, AWK, Scala, Go, D (in random order)

# Programming languages used in bioinformatics

- bio<insert<sub>anylanguage</sub>>
  - biostatistics
  - biochemistry
  - biophysics
  - bioLinux
  - biohazard :)
- C, Java, Python, R, Bash, C++, AWK, Scala, Go, D (in random order)
- Not personally encounter: Ruby/BioRuby, Rust/RustBio

# Bad things about C, C++, Java, and Python

- Readability, Verbosity (C++, Java )
- Boiler-plate code (Java, C++ )
- Security: Room for developer mistakes, end-user mishaps.
- Learning curve (C++, C)
- Speed (Java, Python )
- Development time (C, C++) and compile time (C++)
- Parallel and distributed computing support (C, C++)

# Good things of C, C++, Java, and Python

- IDE: code navigation, auto-completion, debugging, refactoring (Java, Python, C++, C)
- Libraries (Python, Java, C++, C)
- Readability (Python, Java)
- Speed, and light-weight (C, C++ ).
- Balance of speed, libraries, and IDE (C++, Java, Python, C).
- Have TIE

# Good things of C, C++, Java, and Python

- IDE: code navigation, auto-completion, debugging, refactoring (Java, Python, C++, C)
- Libraries (Python, Java, C++, C)
- Readability (Python, Java)
- Speed, and light-weight (C, C++ ).
- Balance of speed, libraries, and IDE (C++, Java, Python, C).
- Have TIE
  - Tools, Industry, Education

# Influencing factors to adopt a language

- CLOMPS

- ① Community -> Chance to learn, have bugfixes, find jobs and collaborations,
- ② Learning <- Real world examples(sambamba), Libraries, Books, IDE, Education programs
- ③ Opportunity <-> Community, Industry
- ④ Maintainability <- Clarity, Readability, Backward Compatibility, (Developers)
- ⑤ Productivity <- Syntax, Libraries, Maturity, and Tools

# Influencing factors to adopt a language

- CLOMPS

- ① Community -> Chance to learn, have bugfixes, find jobs and collaborations,
- ② Learning <- Real world examples(sambamba), Libraries, Books, IDE, Education programs
- ③ Opportunity <-> Community, Industry
- ④ Maintainability <- Clarity, Readability, Backward Compatibility, (Developers)
- ⑤ Productivity <- Syntax, Libraries, Maturity, and Tools
- ⑥ Speed -> Speed is not the first priority, but the first choice when correctness 'seems' to be guaranteed.

# Influencing factors to adopt a language

- CLOMPS

- ① Community -> Chance to learn, have bugfixes, find jobs and collaborations,
- ② Learning <- Real world examples(sambamba), Libraries, Books, IDE, Education programs
- ③ Opportunity <-> Community, Industry
- ④ Maintainability <- Clarity, Readability, Backward Compatibility, (Developers)
- ⑤ Productivity <- Syntax, Libraries, Maturity, and Tools
- ⑥ Speed -> Speed is not the first priority, but the first choice when correctness 'seems' to be guaranteed.

- PS

# Influencing factors to adopt a language

- CLOMPS

- ① Community -> Chance to learn, have bugfixes, find jobs and collaborations,
- ② Learning <- Real world examples(sambamba), Libraries, Books, IDE, Education programs
- ③ Opportunity <-> Community, Industry
- ④ Maintainability <- Clarity, Readability, Backward Compatibility, (Developers)
- ⑤ Productivity <- Syntax, Libraries, Maturity, and Tools
- ⑥ Speed -> Speed is not the first priority, but the first choice when correctness 'seems' to be guaranteed.

- PS

- Platform Portability (not a big issue with bioinformatics)

# Influencing factors to adopt a language

- CLOMPS

- ① Community -> Chance to learn, have bugfixes, find jobs and collaborations,
- ② Learning <- Real world examples(sambamba), Libraries, Books, IDE, Education programs
- ③ Opportunity <-> Community, Industry
- ④ Maintainability <- Clarity, Readability, Backward Compatibility, (Developers)
- ⑤ Productivity <- Syntax, Libraries, Maturity, and Tools
- ⑥ Speed -> Speed is not the first priority, but the first choice when correctness 'seems' to be guaranteed.

- PS

- Platform Portability (not a big issue with bioinformatics)
- Security:

# Influencing factors to adopt a language

- CLOMPS

- ① Community -> Chance to learn, have bugfixes, find jobs and collaborations,
- ② Learning <- Real world examples(sambamba), Libraries, Books, IDE, Education programs
- ③ Opportunity <-> Community, Industry
- ④ Maintainability <- Clarity, Readability, Backward Compatibility, (Developers)
- ⑤ Productivity <- Syntax, Libraries, Maturity, and Tools
- ⑥ Speed -> Speed is not the first priority, but the first choice when correctness 'seems' to be guaranteed.

- PS

- Platform Portability (not a big issue with bioinformatics)
- Security:
  - No insecure language, just insecure applications made by unbeaten programmers, and used by inexperienced users



# Influencing factors to adopt a language

- CLOMPS

- ① Community -> Chance to learn, have bugfixes, find jobs and collaborations,
- ② Learning <- Real world examples(sambamba), Libraries, Books, IDE, Education programs
- ③ Opportunity <-> Community, Industry
- ④ Maintainability <- Clarity, Readability, Backward Compatibility, (Developers)
- ⑤ Productivity <- Syntax, Libraries, Maturity, and Tools
- ⑥ Speed -> Speed is not the first priority, but the first choice when correctness 'seems' to be guaranteed.

- PS

- Platform Portability (not a big issue with bioinformatics)
- Security:
  - No insecure language, just insecure applications made by unbeaten programmers, and used by inexperienced users



# What a bioinformatician is looking for (BEEPS)

- Beautiful presentation of the end product (plot, diagram)
- Expandable and reusable. Versatile: commandline, GUI, web, cloud, (mobile)
- Easy to understand/customize and learn (the language, tools, community, libraries)
- Productivity
- Speed -> Parallel/distributed/cloud support -> Money, Turnaround time.

# Why D is good for bioinformatics

- B(eautiful)

# Why D is good for bioinformatics

- B(eautiful)
  - ggplotD, gtkD, qtD

# Why D is good for bioinformatics

- B(eautiful)
  - ggplotD, gtkD, qtD
- E(xandable), E(asy) to understand/customize learn,  
P(roductivity)

# Why D is good for bioinformatics

- B(eautiful)
  - ggplotD, gtkD, qtD
- E(xandable), E(asy) to understand/customize learn,  
P(roductivity)
  - Dlang's modernness (syntactic sugar) ~ python, not Applescript :)

# Why D is good for bioinformatics

- B(eautiful)
  - ggplotD, gtkD, qtD
- E(xpandable), E(asy) to understand/customize learn,  
P(roductivity)
  - Dlang's modernness (syntactic sugar) ~ python, not Applescript  
:)
  - Built-in unit test: learn to write unit test right from start

# Why D is good for bioinformatics

- B(eautiful)
  - ggplotD, gtkD, qtD
- E(xandable), E(asy) to understand/customize learn,  
P(roductivity)
  - Dlang's modernness (syntactic sugar) ~ python, not Applescript :)
  - Built-in unit test: learn to write unit test right from start
  - Easy switching commandline, GUI, Web

# Why D is good for bioinformatics

- B(eautiful)
  - ggplotD, gtkD, qtD
- E(xandable), E(asy) to understand/customize learn,  
P(roductivity)
  - Dlang's modernness (syntactic sugar) ~ python, not Applescript :)
  - Built-in unit test: learn to write unit test right from start
  - Easy switching commandline, GUI, Web
  - Multi-paradigms and multi modes (scripting, compiled program, library)

# Why D is good for bioinformatics

- B(eautiful)
  - ggplotD, gtkD, qtD
- E(xpandable), E(asy) to understand/customize learn,  
P(roductivity)
  - Dlang's modernness (syntactic sugar) ~ python, not Applescript :)
  - Built-in unit test: learn to write unit test right from start
  - Easy switching commandline, GUI, Web
  - Multi-paradigms and multi modes (scripting, compiled program, library)
  - Strong interface to C and C++

# Why D is good for bioinformatics

- B(eautiful)
  - ggplotD, gtkD, qtD
- E(xpandable), E(asy) to understand/customize learn,  
P(roductivity)
  - Dlang's modernness (syntactic sugar) ~ python, not Applescript :)
  - Built-in unit test: learn to write unit test right from start
  - Easy switching commandline, GUI, Web
  - Multi-paradigms and multi modes (scripting, compiled program, library)
  - Strong interface to C and C++
- Speed

# Why D is good for bioinformatics

- B(eautiful)
  - ggplotD, gtkD, qtD
- E(xandable), E(asy) to understand/customize learn,  
P(roductivity)
  - Dlang's modernness (syntactic sugar) ~ python, not Applescript :)
  - Built-in unit test: learn to write unit test right from start
  - Easy switching commandline, GUI, Web
  - Multi-paradigms and multi modes (scripting, compiled program, library)
  - Strong interface to C and C++
- Speed
  - Built-in parallel support: foreach (e; [1,2,3].parallel)

# Why D is good for bioinformatics

- B(eautiful)
  - ggplotD, gtkD, qtD
- E(xandable), E(asy) to understand/customize learn,  
P(roductivity)
  - Dlang's modernness (syntactic sugar) ~ python, not Applescript :)
  - Built-in unit test: learn to write unit test right from start
  - Easy switching commandline, GUI, Web
  - Multi-paradigms and multi modes (scripting, compiled program, library)
  - Strong interface to C and C++
- Speed
  - Built-in parallel support: foreach (e; [1,2,3].parallel)
  - Compile code and standalone (-static): Both speed and end-users' peace

# Why D is good for bioinformatics

- B(eautiful)
  - ggplotD, gtkD, qtD
- E(xandable), E(asy) to understand/customize learn,  
P(roductivity)
  - Dlang's modernness (syntactic sugar) ~ python, not Applescript :)
  - Built-in unit test: learn to write unit test right from start
  - Easy switching commandline, GUI, Web
  - Multi-paradigms and multi modes (scripting, compiled program, library)
  - Strong interface to C and C++
- Speed
  - Built-in parallel support: foreach (e; [1,2,3].parallel)
  - Compile code and standalone (-static): Both speed and end-users' peace
  - Performance on par with C, C++. Embedded Assembly code!

# Simple task: Count base nucleotide frequencies

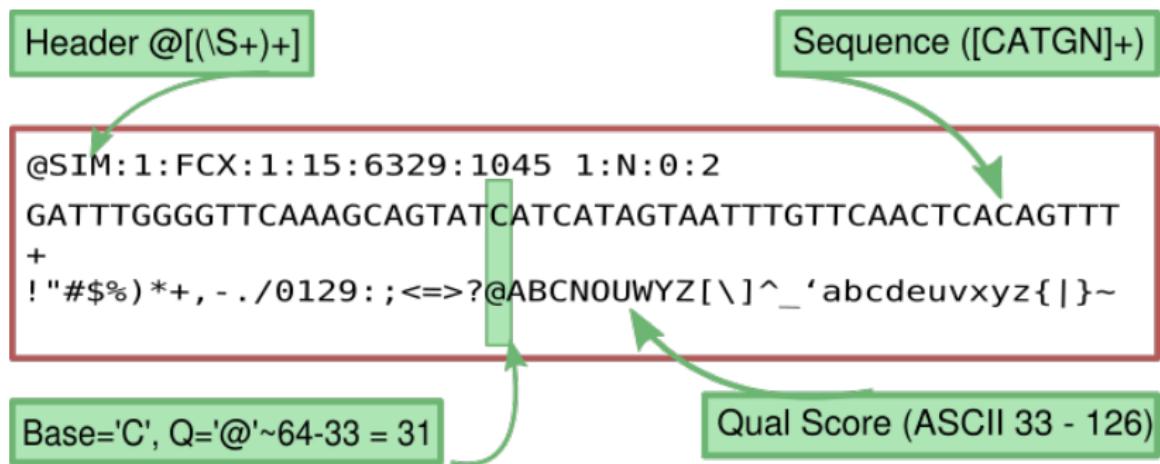
```
cat rosalind_dna.txt
```

```
CGACTAACGGGACAATCCAAGGCGGTGT..(contains CATGN or newline)
```

# Simple task: Count base nucleotide frequencies

```
time (repeat 10000 {<runcommand> >/dev/null})  
  
gcc -O3 cdnacount 4.92s user 1.49s system 105% cpu 6.098 total  
ldc2 -O3 ldccount 9.39s user 1.90s system 102% cpu 11.003 total  
dnacount.py 85.16s user 22.23s system 100% cpu 1:46.96 total  
. ./dnacount.d 25.60s user 5.91s system 106% cpu 29.584 total dmd  
-O dmdcount 42.17s user 6.82s system 100% cpu 48.528 total
```

# A more complex task: parsing FastQ file, and extract info



# Real work: Processing BAM file

- Indexing 71 GB of 5 BAM files around 14GB each

```
conda create -n benchmark picard samtools sambamba ncurses  
source activate benchmark  
time (for ...; do samtools index -@10 $f;done;)  
time (for ...; do sambamba index -t10 $f;done;)  
samtools 423% cpu 5:37.61 total  
sambamba 1068% cpu 4:16.97 total  
time (picard BuildBamIndex I=$f O=${f/.bam/.bai};done;)  
143% cpu 36:59.24 total
```

# Adoption roadmap

- ① Do scripting as Bash alternative (lib: Scriptlike)
- ② Build analysis pipeline (Python, Bash, Java/Scala are doing fine)
- ③ Analyze data in BAM, VCF format (variant calling, chromosome structural variation, count BAM statistics).
- ④ Port and develop new tools in D (to learn and compare)
- ⑤ Develop GUI and commandline applications for end users
- ⑥ Make D the main language to power big data analysis

# Difficulties

- Double mission: bioinformatics, which Python is leading, and D!

# Difficulties

- Double mission: bioinformatics, which Python is leading, and D!
- Learn to use D concepts

# Difficulties

- Double mission: bioinformatics, which Python is leading, and D!
- Learn to use D concepts
  - How do I implement ranges for FASTQ/BAM records

# Difficulties

- Double mission: bioinformatics, which Python is leading, and D!
- Learn to use D concepts
  - How do I implement ranges for FASTQ/BAM records
- Learn to express common tasks and algorithms in D  
(Important!) For example, fastq parser -> performance tuning  
(ranges, auto-vectorization, @nogc, @fastmath, std.mmfile)

# Difficulties

- Double mission: bioinformatics, which Python is leading, and D!
- Learn to use D concepts
  - How do I implement ranges for FASTQ/BAM records
- Learn to express common tasks and algorithms in D  
(Important!) For example, fastq parser -> performance tuning  
(ranges, auto-vectorization, @nogc, @fastmath, std.mmfile)
- Few real-world examples for bioinformatics

# Difficulties

- Double mission: bioinformatics, which Python is leading, and D!
- Learn to use D concepts
  - How do I implement ranges for FASTQ/BAM records
- Learn to express common tasks and algorithms in D  
(Important!) For example, fastq parser -> performance tuning  
(ranges, auto-vectorization, @nogc, @fastmath, std.mmfile)
- Few real-world examples for bioinformatics
  - Parser: FASTA/FASTQ GFF3, BED, BAM, VCF

# Difficulties

- Double mission: bioinformatics, which Python is leading, and D!
- Learn to use D concepts
  - How do I implement ranges for FASTQ/BAM records
- Learn to express common tasks and algorithms in D  
(Important!) For example, fastq parser -> performance tuning  
(ranges, auto-vectorization, @nogc, @fastmath, std.mmfile)
- Few real-world examples for bioinformatics
  - Parser: FASTA/FASTQ GFF3, BED, BAM, VCF
- Lack libraries

# Difficulties

- Double mission: bioinformatics, which Python is leading, and D!
- Learn to use D concepts
  - How do I implement ranges for FASTQ/BAM records
- Learn to express common tasks and algorithms in D  
(Important!) For example, fastq parser -> performance tuning  
(ranges, auto-vectorization, @nogc, @fastmath, std.mmfile)
- Few real-world examples for bioinformatics
  - Parser: FASTA/FASTQ GFF3, BED, BAM, VCF
- Lack libraries
- Small community

# Motivating Factors

Be among the pioneers.

# Motivating Factors

Be among the pioneers.

Mastering a powerful language to deal with problems that require performance, security, ease of development

# Motivating Factors

Be among the pioneers.

Mastering a powerful language to deal with problems that require performance, security, ease of development

Inherit and enhance C/C++ to get more productive of quality work  
(BetterC, Tool: Calypso, Dpp)

# Motivating Factors

Be among the pioneers.

Mastering a powerful language to deal with problems that require performance, security, ease of development

Inherit and enhance C/C++ to get more productive of quality work  
(BetterC, Tool: Calypso, Dpp)

Friendly and supportive community

# Library development

- bioD (<https://github.com/biod>)

# Library development

- bioD (<https://github.com/biod>)
- bioslaD (<https://github.com/bioslaD>)

# Library development

- bioD (<https://github.com/biod>)
- bioslaD (<https://github.com/bioslaD>)
- statistics

# Library development

- bioD (<https://github.com/biod>)
- bioslaD (<https://github.com/bioslaD>)
- statistics
- plotting (<https://github.com/BlackEdder/ggplotd>)

# Library development

- bioD (<https://github.com/biod>)
- bioslaD (<https://github.com/bioslaD>)
- statistics
- ploting (<https://github.com/BlackEdder/ggplotd>)
- parallel/distributed computing (mir, dcompute)

# Library development

- bioD (<https://github.com/biod>)
- bioslaD (<https://github.com/bioslaD>)
- statistics
- plotting (<https://github.com/BlackEdder/ggplotd>)
- parallel/distributed computing (mir, dcompute)
- machine learning  
(<https://github.com/Netflix/vectorflow>)

# Port popular tools, and make new ones

- Top 3 tools in C/C++: bwa, khmer, deepvariant
- aligners (C: clustalw, bwa(DNA), STAR(RNA))
- sequence assemblers (C++: abyss; Python + C++: Spades)
- BAM file tools: samtools, htslib -> partially handled by sambamba
- variant calling (Java + Scala: GATK; Python + C++: Deepvariant)
- structural variant (C++: BreakDancer; Java + Scala + R: svtoolkit)
- gene expression counting (C++: Tophat, Cufflink, salmon)
- primer design (C + Perl: Primer3)

# More Showcases of small programs

- [rosalind.info](http://rosalind.info) problems -> submit your solutions to [github.com/bioslaD/rosalind](https://github.com/bioslaD/rosalind)

The screenshot shows the Rosalind website with a navigation bar for About, Problems, Statistics, Glossary, search, and social media links (Facebook, Twitter). It features a main heading "Locations" and a brief introduction about the platform.

**Rosalind** is a platform for learning bioinformatics and programming through problem solving. [Take a tour](#) to get the hang of how Rosalind works. If you don't know anything about programming, you can start at the [Python Village](#). For a collection of exercises to accompany Bioinformatics Algorithms book, go to the [Textbook Track](#). Otherwise you can try to storm the [Bioinformatics Stronghold](#) right now.

**Python Village**: Python logo icon. A portrait of Rosalind Franklin with DNA helixes. Text: "completely new to try these initial learn a few basics about programming language. familiar with the operations of solving bioinformatics the Stronghold."

**Bioinformatics Stronghold**: Red blob logo icon. Text: "Discover the algorithms underlying a variety of bioinformatics topics: computational mass spectrometry, alignment, dynamic programming, genome assembly, genome rearrangements, phylogeny, probability, string algorithms and others."

**Bioinformatics Armory**: Computer monitor icon with an X. Text: "Ready-to-use software tools abound for bioinformatics analysis. Whereas in the Stronghold you implement algorithms on your own, in the Armory you solve similar problems by using existing tools."

**Bioinformatics Textbook Track**: Text: "A collection of exercises to accompany Bioinformatics Algorithms: An Active-Learning Approach by Phillip Compeau & Pavel Pevzner. A full version of this text is hosted on [stepic.org](#)"

**Algorithmic Heights**: Three stacked squares icon. Text: "A collection of exercises in introductory algorithms to accompany 'Algorithms', the popular textbook by Dasgupta, Papadimitriou, and Vazirani."

# MS, PhD projects: Bioinformatic Algorithms, Big Data applications



# Support other languages

- R(Rcpp, rJava), Python(Cython) -> Be the most popular second language

# Support other languages

- R(Rcpp, rJava), Python(Cython) -> Be the most popular second language
- So, RnD?!

# Bioinformatics for Bio + Informatics

- Status of the adoption

# Bioinformatics for Bio + Informatics

- Status of the adoption
  - It's still in early stages, but very exciting and promising.

# Bioinformatics for Bio + Informatics

- Status of the adoption

- It's still in early stages, but very exciting and promising.
- There are some difficulties for biologist to take up D but it's worth to try.

# Bioinformatics for Bio + Informatics

- Status of the adoption

- It's still in early stages, but very exciting and promising.
- There are some difficulties for biologist to take up D but it's worth to try.
- It will be much better with more libraries and more people using D for bioinformatics.

# Bioinformatics for Bio + Informatics

- Status of the adoption
  - It's still in early stages, but very exciting and promising.
  - There are some difficulties for biologist to take up D but it's worth to try.
  - It will be much better with more libraries and more people using D for bioinformatics.
- Computer programmers/scientists

# Bioinformatics for Bio + Informatics

- Status of the adoption
  - It's still in early stages, but very exciting and promising.
  - There are some difficulties for biologist to take up D but it's worth to try.
  - It will be much better with more libraries and more people using D for bioinformatics.
- Computer programmers/scientists
  - Bioinformatics is interesting. Lot of ground for you to build your careers.

# Bioinformatics for Bio + Informatics

- Status of the adoption
  - It's still in early stages, but very exciting and promising.
  - There are some difficulties for biologist to take up D but it's worth to try.
  - It will be much better with more libraries and more people using D for bioinformatics.
- Computer programmers/scientists
  - Bioinformatics is interesting. Lot of ground for you to build your careers.
- Bioinformaticians/Biologists

# Bioinformatics for Bio + Informatics

- Status of the adoption
  - It's still in early stages, but very exciting and promising.
  - There are some difficulties for biologist to take up D but it's worth to try.
  - It will be much better with more libraries and more people using D for bioinformatics.
- Computer programmers/scientists
  - Bioinformatics is interesting. Lot of ground for you to build your careers.
- Bioinformaticians/Biologists
  - D is very good for bioinformatics (speed, unittest, parallel, C/C++ interoperability, C++ alternative)

# Bioinformatics for Bio + Informatics

- Status of the adoption
  - It's still in early stages, but very exciting and promising.
  - There are some difficulties for biologist to take up D but it's worth to try.
  - It will be much better with more libraries and more people using D for bioinformatics.
- Computer programmers/scientists
  - Bioinformatics is interesting. Lot of ground for you to build your careers.
- Bioinformaticians/Biologists
  - D is very good for bioinformatics (speed, unittest, parallel, C/C++ interoperability, C++ alternative)
- Join [github.com/bioslaD](https://github.com/bioslaD), and see you at the hackathon!

# Future of D and Bioinformatics

- D will enjoy the same level of popularity with C/C++, Java/Scala, Python, and R?

# Future of D and Bioinformatics

- D will enjoy the same level of popularity with C/C++, Java/Scala, Python, and R?
- D and bioinformatics need each other to develop and grow

# Future of D and Bioinformatics

- D will enjoy the same level of popularity with C/C++, Java/Scala, Python, and R?
- D and bioinformatics need each other to develop and grow
- Perfect opportunity of D-bioinformatics: interest, collaborations, jobs <-> tools, libraries, learning resources

# Future of D and Bioinformatics

- D will enjoy the same level of popularity with C/C++, Java/Scala, Python, and R?
- D and bioinformatics need each other to develop and grow
- Perfect opportunity of D-bioinformatics: interest, collaborations, jobs <-> tools, libraries, learning resources
- D CLOMPS and everyone BEEPS!

# Acknowledgements

- For friendly introduction to D and helpful discussion online/offline
  - Ali Çehreli
  - Mike Parker
  - Dentcho Pankov
  - Nicholas Wilson
  - John Loughran Colvin
  - Stefan Koch
  - Simen Kjærås
- Petar Kirov and Steven Schveighoffer for being my D mentors
- Sebastian Wilzbach for on going discussions about bioinformatics

# Acknowledgements

- For friendly introduction to D and helpful discussion online/offline
  - Ali Çehreli
  - Mike Parker
  - Dentcho Pankov
  - Nicholas Wilson
  - John Loughran Colvin
  - Stefan Koch
  - Simen Kjærås
- Petar Kirov and Steven Schveighoffer for being my D mentors
- Sebastian Wilzbach for on going discussions about bioinformatics
- Nina A H Madsen (AUH colleague) for discussion about this talk

# Acknowledgements

- For friendly introduction to D and helpful discussion online/offline
  - Ali Çehreli
  - Mike Parker
  - Dentcho Pankov
  - Nicholas Wilson
  - John Loughran Colvin
  - Stefan Koch
  - Simen Kjærås
- Petar Kirov and Steven Schveighoffer for being my D mentors
- Sebastian Wilzbach for on going discussions about bioinformatics
- Nina A H Madsen (AUH colleague) for discussion about this talk
- Aalborg University Hospital for funding

# Implement Needleman–Wunsch and Smith–Waterman algorithm

- Alignment problems: <http://bit.do/seqalign> (notes) and <http://bit.do/alignslides> (slides)

# Implement Needleman–Wunsch and Smith–Waterman algorithm

- Alignment problems: <http://bit.do/seqalign> (notes) and <http://bit.do/alignslides> (slides)
- Submit to <https://github.com/bioslaD/algorithms.bio>

# Implement Needleman–Wunsch and Smith–Waterman algorithm

- Alignment problems: <http://bit.do/seqalign> (notes) and <http://bit.do/alignslides> (slides)
- Submit to <https://github.com/bioslaD/algorithms.bio>
- Implement in D.

# Implement Needleman–Wunsch and Smith–Waterman algorithm

- Alignment problems: <http://bit.do/seqalign> (notes) and <http://bit.do/alignslides> (slides)
- Submit to <https://github.com/bioslaD/algorithms.bio>
- Implement in D.
- Output in MSA (global alignment) or BAM (local alignments)

# Implement Needleman–Wunsch and Smith–Waterman algorithm

- Alignment problems: <http://bit.do/seqalign> (notes) and <http://bit.do/alignslides> (slides)
- Submit to <https://github.com/bioslaD/algorithms.bio>
- Implement in D.
- Output in MSA (global alignment) or BAM (local alignments)
- Writing criteria: Combination of readability, reusability, scalability and speed

# Implement Needleman–Wunsch and Smith–Waterman algorithm

- Alignment problems: <http://bit.do/seqalign> (notes) and <http://bit.do/alignslides> (slides)
- Submit to <https://github.com/bioslaD/algorithms.bio>
- Implement in D.
- Output in MSA (global alignment) or BAM (local alignments)
- Winning criteria: Combination of readability, reusability, scalability and speed
- License: GPLv3

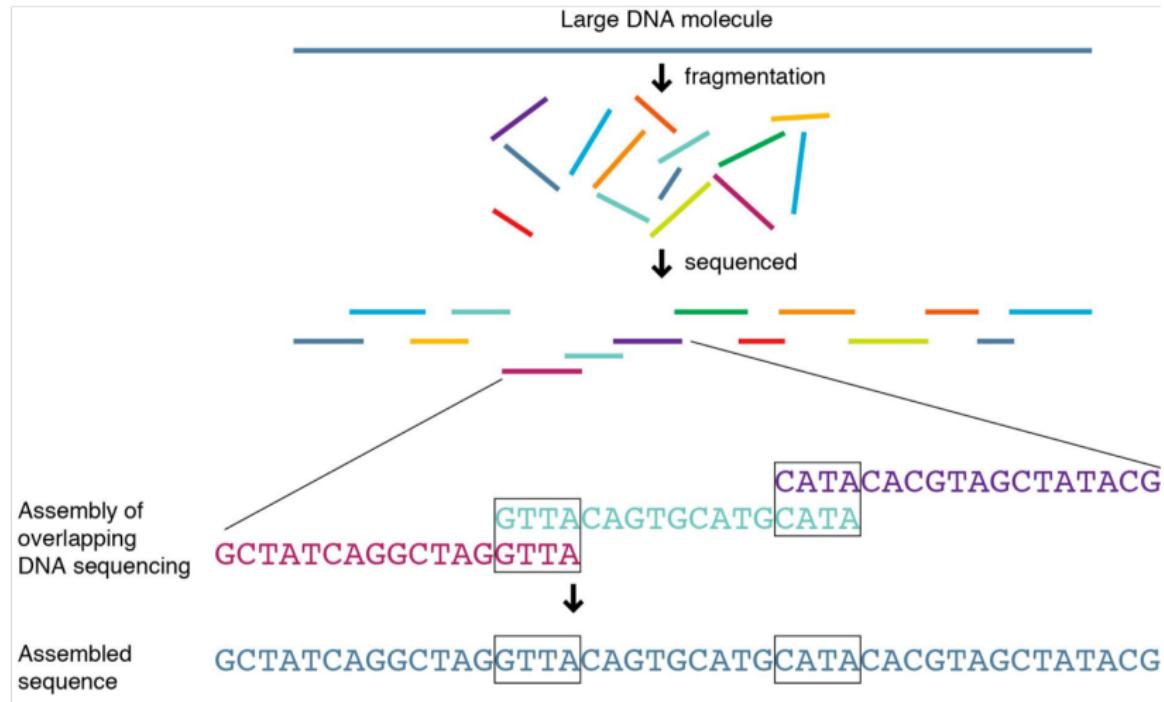
# Implement Needleman–Wunsch and Smith–Waterman algorithm

- Alignment problems: <http://bit.do/seqalign> (notes) and <http://bit.do/alignslides> (slides)
- Submit to <https://github.com/bioslaD/algorithms.bio>
- Implement in D.
- Output in MSA (global alignment) or BAM (local alignments)
- Winning criteria: Combination of readability, reusability, scalability and speed
- License: GPLv3
- Prize: USD100+ (more sponsorship?) for each problem (global or local alignment).

# Implement Needleman–Wunsch and Smith–Waterman algorithm

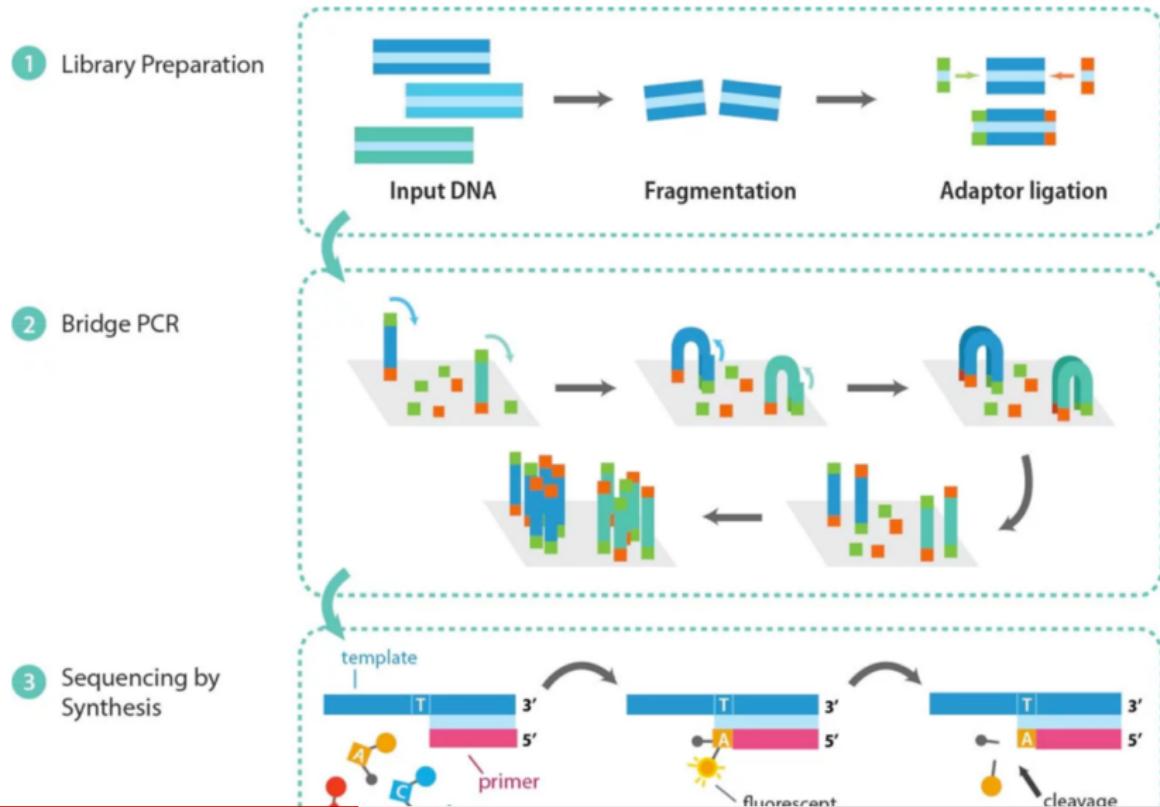
- Alignment problems: <http://bit.do/seqalign> (notes) and <http://bit.do/alignslides> (slides)
- Submit to <https://github.com/bioslaD/algorithms.bio>
- Implement in D.
- Output in MSA (global alignment) or BAM (local alignments)
- Winning criteria: Combination of readability, reusability, scalability and speed
- License: GPLv3
- Prize: USD100+ (more sponsorship?) for each problem (global or local alignment).
- Deadline: At least 3 solutions until 10:00 GMT+2 May 4, or 24:00 GMT+2, Saturday 12 May 2018

# NGS principle



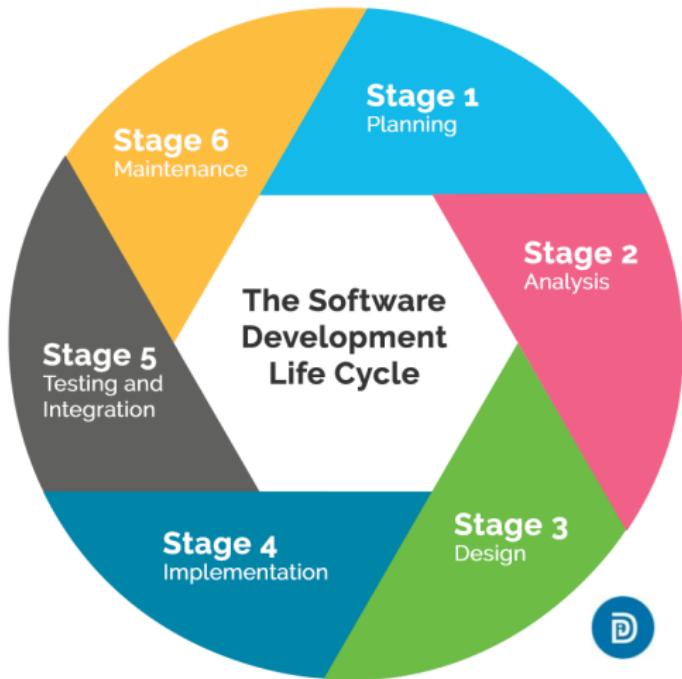
<http://wiremea.com/>

# Illumina Sequencing principle



# Ultra Short CV

- 20 yrs biologist, 12 yrs active linux admin/user, 9 yrs bioinformatician. Bash, Python, R, C, C++, Java, Scala, D



# Ultra Short CV

- 20 yrs biologist, 12 yrs active linux admin/user, 9 yrs bioinformatician. Bash, Python, R, C, C++, Java, Scala, D



# Ultra Short CV

- 20 yrs biologist, 12 yrs active linux admin/user, 9 yrs bioinformatician. Bash, Python, R, C, C++, Java, Scala, D

# Ultra Short CV

- 20 yrs biologist, 12 yrs active linux admin/user, 9 yrs bioinformatician. Bash, Python, R, C, C++, Java, Scala, D
- What can we make out of this?

# Ultra Short CV

- 20 yrs biologist, 12 yrs active linux admin/user, 9 yrs bioinformatician. Bash, Python, R, C, C++, Java, Scala, D
- What can we make out of this?
  - May provide experience, maybe bias toward bioinformatics

# Ultra Short CV

- 20 yrs biologist, 12 yrs active linux admin/user, 9 yrs bioinformatician. Bash, Python, R, C, C++, Java, Scala, D
- What can we make out of this?
  - May provide experience, maybe bias toward bioinformatics
  - Can write unit tests :)

# Ultra Short CV

- 20 yrs biologist, 12 yrs active linux admin/user, 9 yrs bioinformatician. Bash, Python, R, C, C++, Java, Scala, D
- What can we make out of this?
  - May provide experience, maybe bias toward bioinformatics
  - Can write unit tests :)
  - Perfect opportunity of D-bioinformatics: interest, collaborations, jobs <-> tools, libraries, learning resources

# Software version for benchmarking

- dmd-2.078.2, ldc 1.7.0, gcc 5.4.0

# Software version for benchmarking

- dmd-2.078.2, ldc 1.7.0, gcc 5.4.0
- java jvm: OpenJDK 64-Bit Server VM 1.8.0\_121-b15

# Software version for benchmarking

- dmd-2.078.2, ldc 1.7.0, gcc 5.4.0
- java jvm: OpenJDK 64-Bit Server VM 1.8.0\_121-b15
- sambamba 0.6.6

# Software version for benchmarking

- dmd-2.078.2, ldc 1.7.0, gcc 5.4.0
- java jvm: OpenJDK 64-Bit Server VM 1.8.0\_121-b15
- sambamba 0.6.6
- samtools Version: 1.8 (using htslib 1.8)

# Software version for benchmarking

- dmd-2.078.2, ldc 1.7.0, gcc 5.4.0
- java jvm: OpenJDK 64-Bit Server VM 1.8.0\_121-b15
- sambamba 0.6.6
- samtools Version: 1.8 (using htslib 1.8)
- picard 2.18.4

# Software version for benchmarking

- dmd-2.078.2, ldc 1.7.0, gcc 5.4.0
- java jvm: OpenJDK 64-Bit Server VM 1.8.0\_121-b15
- sambamba 0.6.6
- samtools Version: 1.8 (using htslib 1.8)
- picard 2.18.4
- Linux environment: Funtoo (for rosalind problems) and Ubuntu 16.04