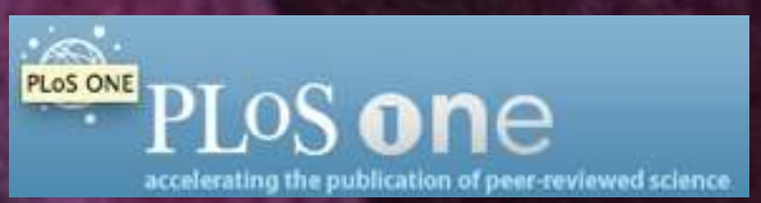
De Novo Illumina Assemblies with Velvet

Velvet

- Velvet: algorithms for *de novo* short read assembly using de Bruijn graphs. D.R. Zerbino and E. Birney. Genome Research 18:821-829. (2008)

- Pebble and Rock Band: Heuristic Resolution of Repeats and Scaffolding in the Velvet Short-Read de Novo Assembler. Zerbino DR, McEwen GK, Margulies EH, Birney E. PLoS One. 2009 Dec 22;4(12):e8407.



- Using the Velvet de Novo Assembler for short-read sequencing technologies. Zerbino DR. Curr Protoc Bioinformatics. Chap 11:Unit 11.5

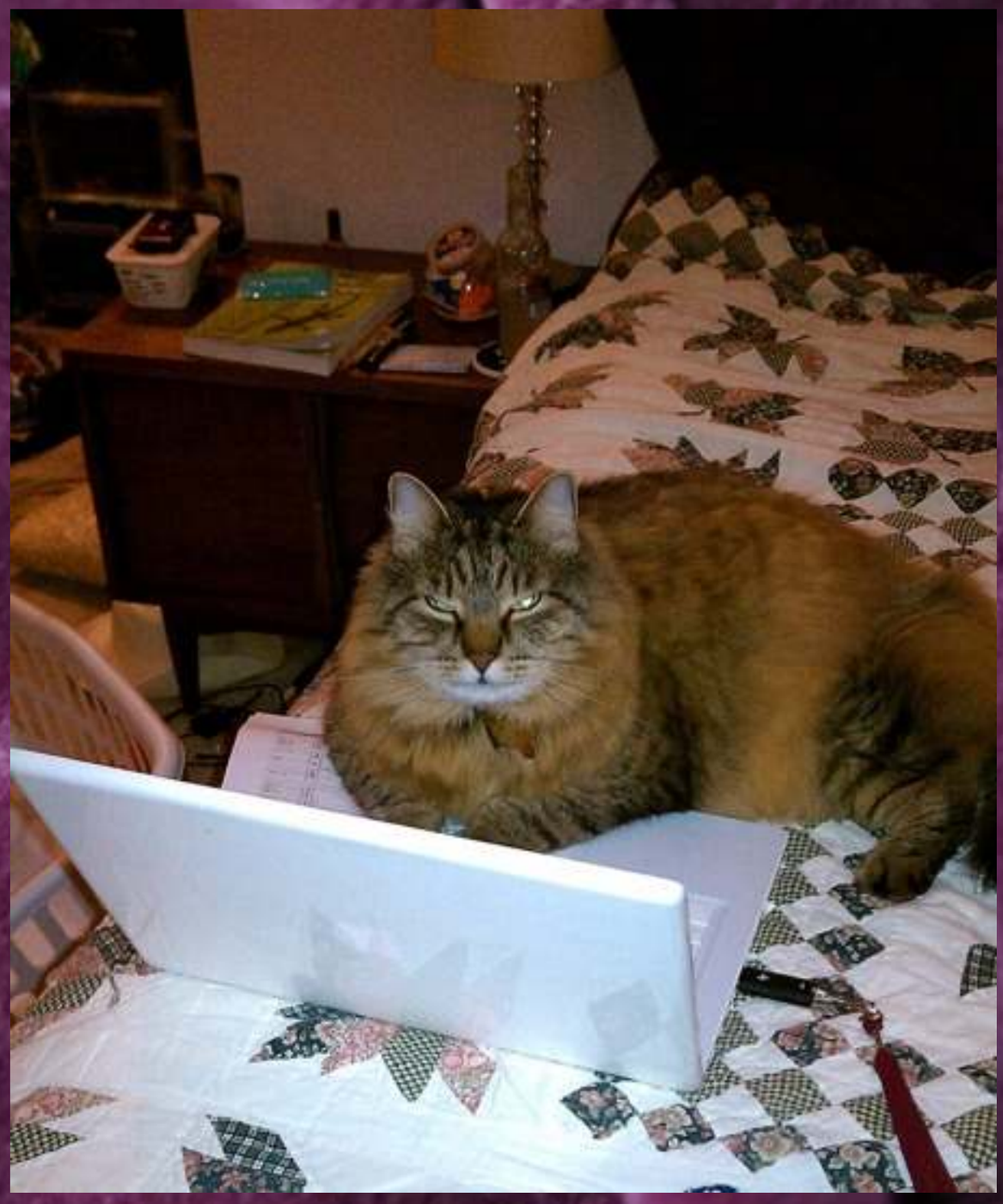
Overview

- Run velveth
- Background
- Run velvetg
- More background
- Details on select velvetg parameters
- Summary and Practical considerations

Disclaimer, re: quality

- Velvet does not use quality values for assembly. Period.
- You should trim your reads prior to assembly
- High coverage can make up for things

Disclaimer, re: kittens



Getting started

- **CAREFULLY** READ THROUGH THE GETTING STARTED BEFORE YOU BEGIN THE ACTIVITY

Getting started

- **CAREFULLY** READ THROUGH THE GETTING STARTED BEFORE YOU BEGIN THE ACTIVITY
- http://www.molecularevolution.org/resources/activities/velvetbowtie_activity
- Complete exercise 1 and start exercise 2.

Overlap-layout-consensus

- Traditional paradigm for Sanger sequencing assemblies
- Overlap : compute all overlaps between reads and construct a graph
- Layout : simplify graph by removing redundancy
- Consensus : build an alignment of all the reads covering the genome
- “Hamiltonian path”: identifying a path through the graph that contains all the nodes

Eulerian paths on de Bruijn graphs

- Pevsner 2001(PNAS)
- k-mers instead of whole reads

Creating a hashtable

- If k=31 and we have a 36bp read...

Creating a hashtable

- **GGTTTCGCTAGATAGCAGCTAGGGACAGTGG**GAATC

Creating a hashtable

- GGTTTCGCTAGATAGCAGCTAGGGACAGTGGGAATC

Creating a hashtable

- GG**TTTCGCTAGATAGCAGCTAGGGACAGTGGGA**ATC

Creating a hashtable

- GGT**TTCGCTAGATAGCAGCTAGGGACAGTGGGAA**TC

Creating a hashtable

- GGTT**TCGCTAGATAGCAGCTAGGGACAGTGGGAAT**C

Creating a hashtable

GGTTTCGCTAGATAGCAGCTAGGGACAGTGGGAATC

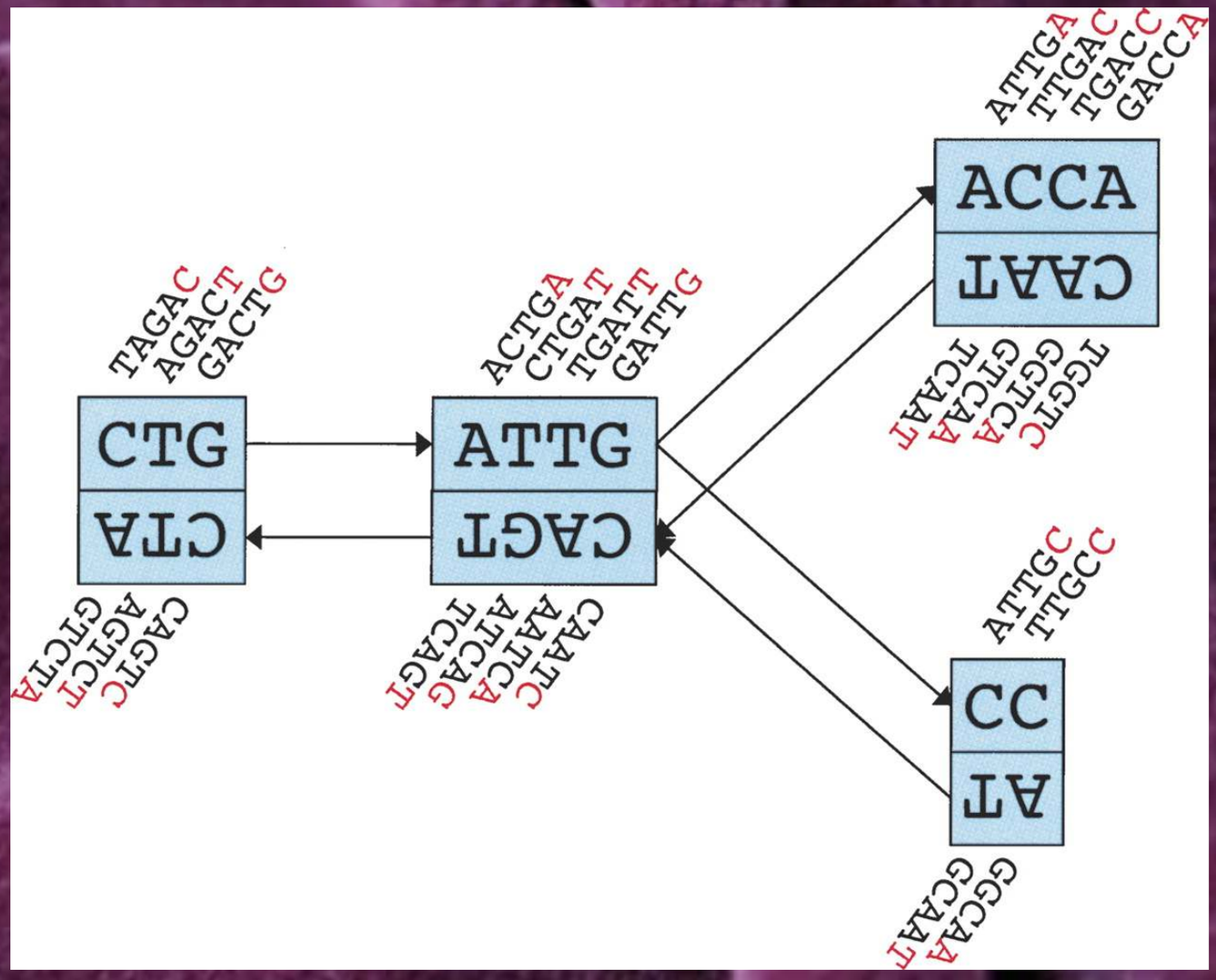
Creating a hashtable

- GGTTT**CGCTAGATAGCAGCTAGGGACAGTGGGAATC**
- A simple graph is created by overlapping these hashed kmers (Roadmap)

Continue exercises

- Finish exercise 2 and start exercise 3.

de Bruijn graphs



Primary velvetg parameters

- k
- exp_cov
  - = Total sequence bases / Genome length
- ins_length

Contig sizes

- NODE_3_length_277033_cov_81.263977
- k-mer length = 277033
- actual length = k-mer length + (k-1)

n50 and n90

- "Contig or scaffold N50 is a weighted median statistic such that 50% of the entire assembly is contained in contigs or scaffolds equal to or larger than this value" -seqanswers.com

k-mer coverage (C_k)

- $C_k = C * (L–k+1)/L$
- k = *kmer* length
- L = read length
- C = nucleotide coverage

Continue activity

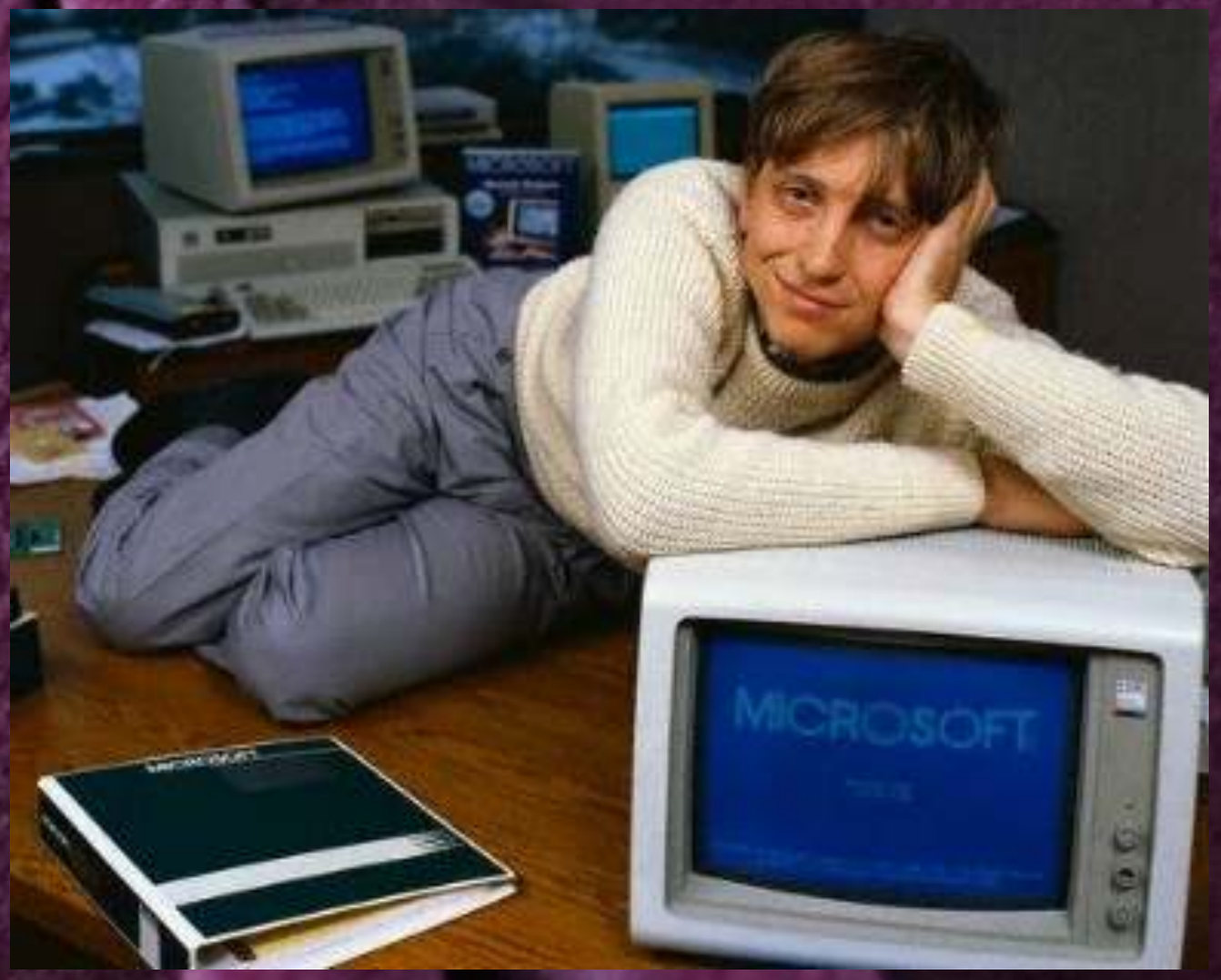
- Finish exercises 4-6

VelvetOptimiser

- Simon Gladman
- Runs velvet on a *k-mer* range you specify to determine optimum
- Estimates optimum *exp_cov* and then searches for optimum cov_cutoff
- Pros:
  - Optimization functions (n50, max contig, tbp)
  - Good job of guessing amount of RAM required
- Cons:
  - Can’t specify most velvetg parameters directly

Memory considerations

- “640Kb ought to be enough for anybody.” - Bill Gates, 1981



Memory estimation

- ~RAM (kb) = -109635 + 18977*ReadSize(bp) + 86236*GenomeSize(Mbp) + 233353*NumReads(in millions) - 51092*K
- Divide by 1048576 to get GB
- Valid for the following ranges:
  - *k* = 15-31
  - Numreads = 5 - 70 million
  - Genome size = 2 - 10 Megabases
  - Read length (size) = 20 - 75 bases
- Developed by **Simon Gladman**

Hardware considerations

- A 1GB genome project with a mixture of ~1billion reads (~73x) is using 300GB+ of RAM
- Other assemblers such as SOAP De Novo tend to use a little less memory, but have given less desirable results in MY experience
- Opt for a processor with higher L3 cache (24MB) than simply clock speed
- Lot's of disk space

Oases

- De novo transcriptome assembler for short reads
- Uses an intitial assembly produced by velvet
- Uses paired-end read information to construct transcript isoforms