Analyzing Genomic Data with PyEnsembl and Varcode

Alex Rubinsteyn

SciPy - July 9th, 2015

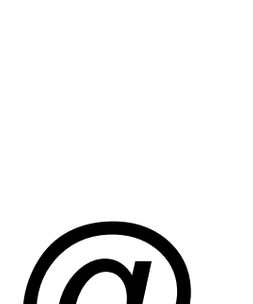
Mount Sinai

HammerLab @ Mount Sinai

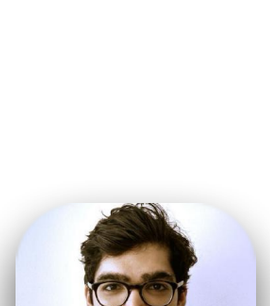
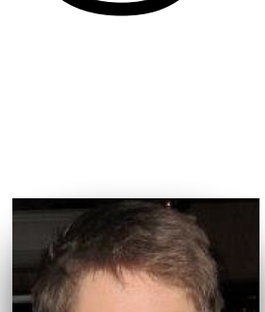
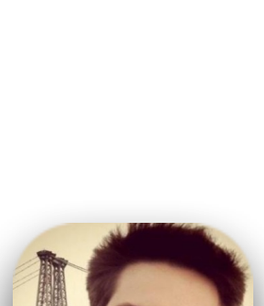
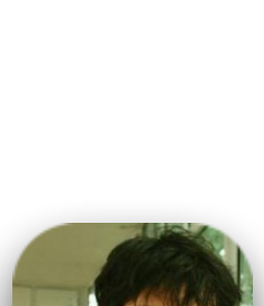








http://www.hammerlab.org/research/

Not biologists!

Most lab members have a background in Computer Science and programming:

- Distributed data management
- Machine learning/statistics
- Programming languages
- Data visualization

Lab focus

Software development: building high-quality open source software for biomedicine

Research: use that software to change how a physician treats a patient

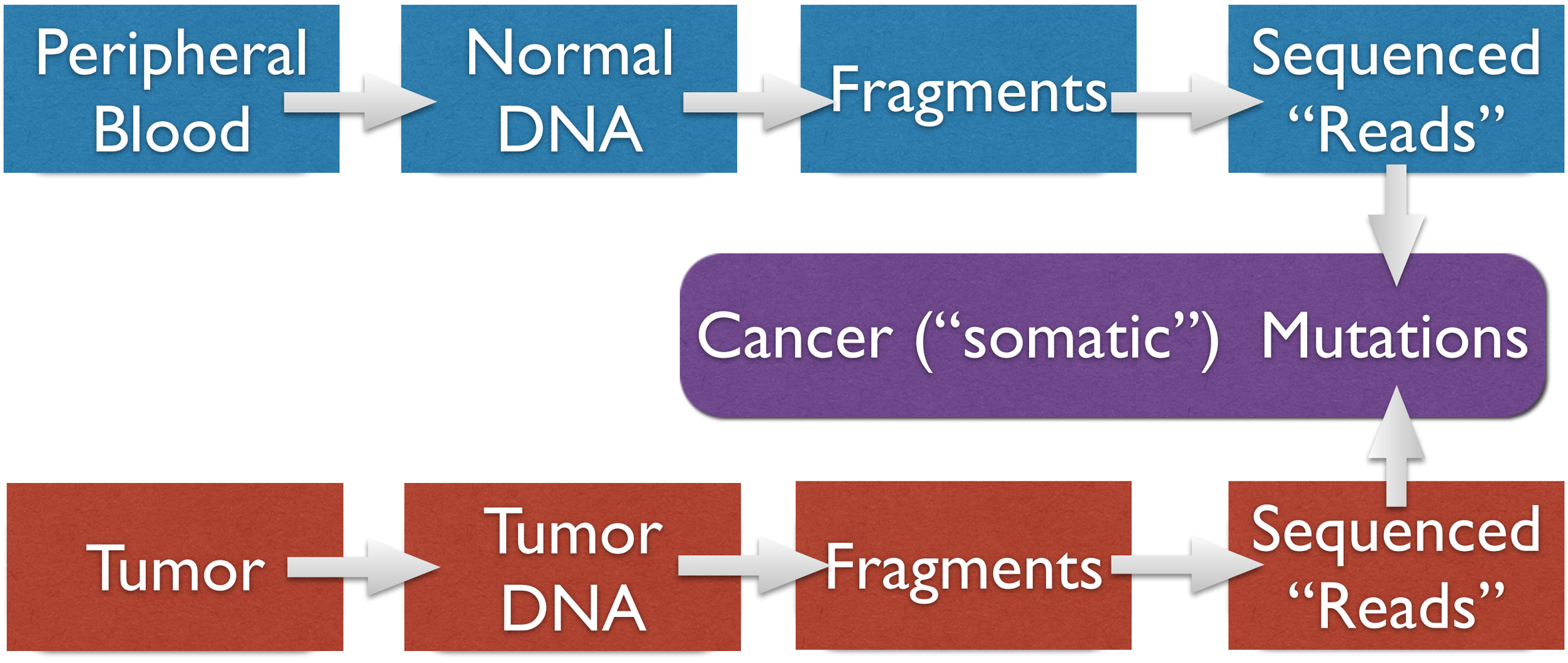
Lab research projects

- Can we predict response to immunotherapy from melanoma & lung cancer mutations? *(collaboration with Alex Snyder & Matt Hellman)*

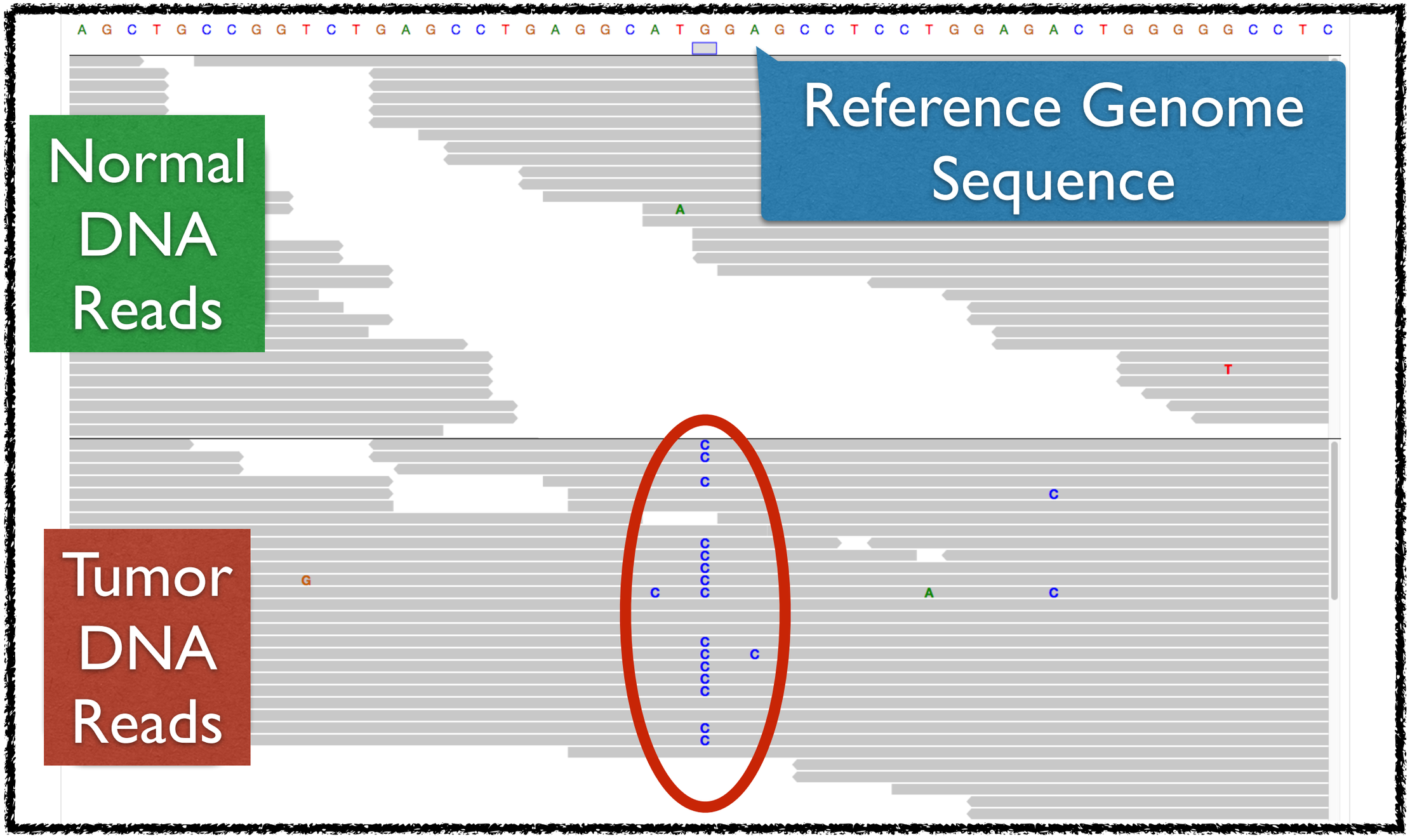
- Does chemotherapy increase heterogeneity of mutations across cells in ovarian cancer? *(collaborations with John Martignetti and Alex Snyder)*

- Can we reprogram the immune system to attack cancer cells making mutated proteins? *(collaboration with Nina Bhardwaj's lab)*
  - Personalized cancer vaccine pipeline written in Python
  - Phase I clinical trial (~20 patients) starting soon (!!)

tl;dr of high throughput DNA sequencing for cancer



How do we find mutations?



Variant file format: VCF

HEADER

```
##fileformat=VCFv4.1
##fileDate=20090805
##tcgaversion=1.1
##vcfProcessLog=<InputVCF=<file1.vcf>,InputVCFSource=<caller1>,InputVCFVer=<1.0>,InputVCFParam=<a1,b>,InputVCFgeneAnno=<anno1.gaf>>
##reference=ftp://ftp.ncbi.nih.gov/genbank/genomes/Eukaryotes/vertebrates_mammals/Homo_sapiens/GRCh37/special_requests/GRCh37-lite.fa
##contig=<ID=20,length=62435964,assembly=B36,md5=f126cdf8a6e0c7f379d618ff66beb2da,species="Homo sapiens",taxonomy=x>
##phasing=partial
##INFO=<ID=NS,Number=1,Type=Integer,Description="Number of Samples With Data">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Total Depth">
##INFO=<ID=AF,Number=A,Type=Float,Description="Allele Frequency">
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=DB,Number=0,Type=Flag,Description="dbSNP membership, build 129">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
```

INFO meta-information

```
##FILTER=<ID=q10,Description="Quality below 10">
##FILTER=<ID=s50,Description="Less than 50% of samples have data">
```

FILTER meta-information

```
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##FORMAT=<ID=HQ,Number=2,Type=Integer,Description="Haplotype Quality">
```

FORMAT meta-information

```
##SAMPLE=<ID=NORMAL,Individual=TCGA-01-1000,File=TCGA-01-1000-1.bam,Platform=Illumina,Source=dbGAP,Accession=1234>
##SAMPLE=<ID=TUMOR,Individual=TCGA-01-1000,File=TCGA-01-1000-2.bam,Platform=Illumina,Source=dbGAP,Accession=4567>
##PEDIGREE=<Name_0=TUMOR,Name_1=NORMAL>
```

Fixed fields

Optional: FORMAT field specifying data type + Per-sample genotype data

BODY

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO	FORMAT	NORMAL	TUMOR
20	14370	rs6054257	G	A	29	PASS	NS=3;DP=14;AF=0.5;DB;H2	GT:GQ:DP:HQ	0|0:48:1:51,51	1|0:48:8:51,51
20	17330	.	T	A	3	q10	NS=3;DP=11;AF=0.017	GT:GQ:DP:HQ	0|0:49:3:58,50	0|1:3:5:65,3
20	1110696	rs6040355	A	G,T	67	PASS	NS=2;DP=10;AF=0.333,0.667;DB	GT:GQ:DP:HQ	1|2:21:6:23,27	2|1:2:0:18,2
20	1230237	.	T	.	47	PASS	NS=3;DP=13;AA=T	GT:GQ:DP:HQ	0|0:54:7:56,60	0|0:48:4:51,51
20	1234567	microsat1	GTC	G,GTCTC	50	PASS	NS=3;DP=9;AA=G	GT:GQ:DP	0/1:35:4	0/2:17:2

Entry for variant "chr20:1235678 GTC>G"

Source: wiki.nci.nih.gov

Cancer-specific variant file format: MAF

```
Hugo_Symbol Entrez_Gene_Id  Center  NCBI_Build  Chromosome  Start_position
End_position    Strand  Variant_Classification Variant_Type
Reference_Allele    Tumor_Seq_Allele1   Tumor_Seq_Allele2   dbSNP_RS
dbSNP_Val_Status    Tumor_Sample_Barcode    Matched_Norm_Sample_Barcode
Match_Norm_Seq_Allele1 Match_Norm_Seq_Allele2 Tumor_Validation_Allele1
Tumor_Validation_Allele2    Match_Norm_Validation_Allele1
Match_Norm_Validation_Allele2  Verification_StatusValidation_Status
Mutation_Status Sequencing_Phase    Sequence_SourceValidation_Method  Score
BAM_file    Sequencer   Tumor_Sample_UUID  Matched_Norm_Sample_UUID

AGL 178 genome.wustl.edu    37  1   100349684   100349684   +
Missense_Mutation   SNP G   G   A           TCGA-13-1405-01A-01W-0494-09
TCGA-13-1405-10A-01W-0495-09   G   G   G   A   G   G   Unknown Valid   Somatic
4   WXS 454_PCR_WGA 1   dbGAP   Illumina GAIIx
c0d1de72-4cce-4d74-93f0-29c462dc1426   89f04056-0478-4305-b1ce-486ae469b4dd

SASS6   163786  genome.wustl.edu    37  1   100573197   100573197   +
Missense_Mutation   SNP G   G   A           TCGA-04-1542-01A-01W-0553-09
TCGA-04-1542-10A-01W-0553-09   G   G   G   A   G   G   Unknown Valid   Somatic
4   WXS 454_PCR_WGA 1   dbGAP   Illumina GAIIx 317a63af-
e862-43df-8ef5-7c555b2cb678    b94052a8-c3d2-4e47-81e2-62242bc0841a
```

Working With Genomic Mutation Data in Python

PyEnsembl

- Python interface to the Ensembl genome annotations
- *Where is each gene/transcript/exon?*
- *What's the "biotype" of each transcript?* (e.g. long noncoding RNA, protein coding)

Varcode

- Read VCF (& MAF) files
- Filter variants (e.g. only protein coding genes)
- Compare collections of variants (e.g. do these two patients' cancers share mutations?)
- Predict effect of mutations on protein sequence

Getting Started

Download reference genome annotation data:

```
$ pip install pyensembl varcode
$ pyensembl install --release 75
```

…and wait for ~10 minutes for download and indexing of several GB of reference sequence and annotation data.

Ensembl releases specific to the version of the human genome variants are aligned against:

- *GRCh37/NCBI37/hg19* = Ensembl release `75`
- *GRCh38* = Ensembl release `80` (newer releases coming)

PyEnsembl Example: Looking up info about the TP53 gene

```
In [1]: import pyensembl

In [2]: pyensembl.ensembl_grch38.genes_by_name("TP53")
        INFO:root:Cached file Homo_sapiens.GRCh38.79.gtf from URL ftp://ftp.ensembl.org/pub/release-79/gtf/homo_sapiens/Hom
        o_sapiens.GRCh38.79.gtf.gz
Out[2]: [Gene(id=ENSG00000141510, name=TP53, biotype=protein_coding, location=17:7661779-7687550)]

In [3]: gene = _[0]

In [4]: gene
Out[4]: Gene(id=ENSG00000141510, name=TP53, biotype=protein_coding, location=17:7661779-7687550)

In [5]: gene.biotype
Out[5]: 'protein_coding'

In [6]: gene.transcripts
Out[6]: [Transcript(id=ENST00000413465, name=TP53-018, gene_name=TP53, biotype=protein_coding, location=17:7661779-7676594),
         Transcript(id=ENST00000359597, name=TP53-019, gene_name=TP53, biotype=protein_coding, location=17:7666086-7676594),
         Transcript(id=ENST00000504290, name=TP53-006, gene_name=TP53, biotype=protein_coding, location=17:7668402-7675493),
         Transcript(id=ENST00000510385, name=TP53-007, gene_name=TP53, biotype=protein_coding, location=17:7668402-7675493),
         Transcript(id=ENST00000504937, name=TP53-008, gene_name=TP53, biotype=protein_coding, location=17:7668402-7675493),
         Transcript(id=ENST00000619186, name=TP53-023, gene_name=TP53, biotype=protein_coding, location=17:7668402-7675493),
         Transcript(id=ENST00000618944, name=TP53-024, gene_name=TP53, biotype=protein_coding, location=17:7668402-7675493),
         Transcript(id=ENST00000610623, name=TP53-026, gene_name=TP53, biotype=protein_coding, location=17:7668402-7675493),
         Transcript(id=ENST00000610292, name=TP53-022, gene_name=TP53, biotype=protein_coding, location=17:7668402-7687481),
         Transcript(id=ENST00000269305, name=TP53-001, gene_name=TP53, biotype=protein_coding, location=17:7668402-7687538),
         Transcript(id=ENST00000620739, name=TP53-021, gene_name=TP53, biotype=protein_coding, location=17:7668402-7687538),
         Transcript(id=ENST00000617185, name=TP53-202, gene_name=TP53, biotype=protein_coding, location=17:7668402-7687550),
```

PyEnsembl Example: What's the sequence of TP53-001?

```
In [8]: tp53_001 = pyensembl.ensembl_grch38.transcript_by_id("ENST00000269305")

In [9]: tp53_001
Out[9]: Transcript(id=ENST00000269305, name=TP53-001, gene_name=TP53, biotype=protein_coding, location=17:7668402-7687538)
```

```
In [13]: tp53_001.sequence
Out[13]: Seq('GTTTTCCCCTCCCATGTGCTCAAGACTGGCGCTAAAAGTTTTGAGCTTCTCAAA...GTG', SingleLetterAlphabet())

In [14]: tp53_001.protein_id
Out[14]: 'ENSP00000269305'

In [15]: tp53_001.protein_sequence
Out[15]: Seq('MEEPQSDPSVEPPLSQETFSDLWKLLPENNVLSPLPSQAMDDLMLSPDDIEQWF...DSD', SingleLetterAlphabet())
```

Varcode Example: Loading a MAF file

- Each mutation represented by a Variant object
- Collection of Variant objects is called a … VariantCollection!

```
In [1]: import varcode

In [2]: varcode.load_maf("tcga_ovarian_cancer.maf")

Out[2]: <VariantCollection from 'tcga_ovarian_cancer.maf' with 6428 elements>
          -- Variant(contig=1, start=69538, ref=G, alt=A, genome=GRCh37)
          -- Variant(contig=1, start=881892, ref=T, alt=G, genome=GRCh37)
          -- Variant(contig=1, start=3389714, ref=G, alt=A, genome=GRCh37)
          -- Variant(contig=1, start=3624325, ref=G, alt=T, genome=GRCh37)
          -- Variant(contig=1, start=3782259, ref=T, alt=C, genome=GRCh37)
          -- Variant(contig=1, start=6206923, ref=., alt=G, genome=GRCh37)
          -- Variant(contig=1, start=6631275, ref=G, alt=C, genome=GRCh37)
```

```
In [ ]: variants.
        variants.gene_counts
        variants.groupby
        variants.groupby_gene_id
        variants.groupby_gene_name
        variants.metadata
        variants.multi_groupby
        variants.path
        variants.read_json_file
        variants.reference_names
        variants.short_string
```

Varcode: *Which gene is most mutated in ovarian cancer?*

```
In [5]: variants.gene_counts().most_common(10)

Out[5]: [('TP53', 67),
         ('TTN', 37),
         ('TTN-AS1', 34),
         ('RP11-799N11.1', 14),
         ('CTC-297N7.11', 14),
         ('PCDHA2', 12),
         ('FLG-AS1', 12),
         ('PCDHA1', 12),
         ('PCDHGA1', 11),
         ('PCDHGB1', 10)]
```

Varcode: *What are the protein effects of mutations in TP53?*

Many effect classes corresponding to kinds of changes in protein sequence (or describing which non-coding region a mutation affects), examples:

- *Substitution* (change on amino acid into another)
- *Frameshift* (change in codon translation frame)
- *Intronic* (mutation gets spliced out of transcript)

```
In [8]: tp53_variants = variants.groupby_gene_name()["TP53"]

In [9]: tp53_effects = tp53_variants.effects()

In [10]: tp53_effects

Out[10]: <EffectCollection with 1044 elements>
  -- FrameShiftTruncation(variant=chr17 g.7573984_7573985insAGGCCTT, transcript_name=TP53-001, transcript_id=ENST0000
0269305, effect_description=p.L348fs*)
  -- Intronic(variant=chr17 g.7573984_7573985insAGGCCTT, transcript_name=TP53-019, transcript_id=ENST00000359597)
  -- Intronic(variant=chr17 g.7573984_7573985insAGGCCTT, transcript_name=TP53-018, transcript_id=ENST00000413465)
  -- ThreePrimeUTR(variant=chr17 g.7573984_7573985insAGGCCTT, transcript_name=TP53-005, transcript_id=ENST0000042024
6)
  -- FrameShiftTruncation(variant=chr17 g.7573984_7573985insAGGCCTT, transcript_name=TP53-002, transcript_id=ENST0000
0445888, effect_description=p.L348fs*)
  -- ThreePrimeUTR(variant=chr17 g.7573984_7573985insAGGCCTT, transcript_name=TP53-004, transcript_id=ENST0000045526
3)
```

Varcode: *What's the sequence of a mutant protein?*

Don't just want to know what *kind* of effect a mutation has, but specifically what will the sequence of the new protein be.

```
In [20]: worst_effect = tp53_effects.top_priority_effect()

In [21]: worst_effect
Out[21]: FrameShift(variant=chr17 g.7573995_7573995delC, transcript_name=TP53-001, transcript_id=ENST00000269305, effect_descr
         iption=p.N345fs)

In [24]: worst_effect.original_protein_sequence[worst_effect.aa_mutation_start_offset:]
Out[24]: Seq('NEALELKDAQAGKEPGGSRAHSSHLKSKKGQSTSRHKKLMFKTEGPDSD', SingleLetterAlphabet())

In [25]: worst_effect.mutant_protein_sequence[worst_effect.aa_mutation_start_offset:]
Out[25]: Seq('MRPWNSRMPRLGRSQGGAGLTPAT', SingleLetterAlphabet())

In [26]: worst_effect.short_description
Out[26]: 'p.N345fs'
```

Thanks!

PyEnsembl

www.github.com/hammerlab/pyensembl

Varcode

www.github.com/hammerlab/varcode