2003

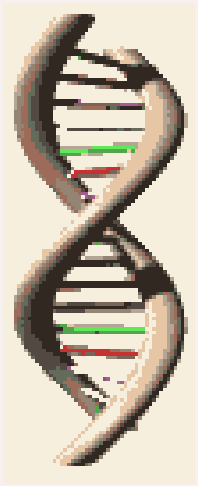
Sequence Alignments and Database Searches

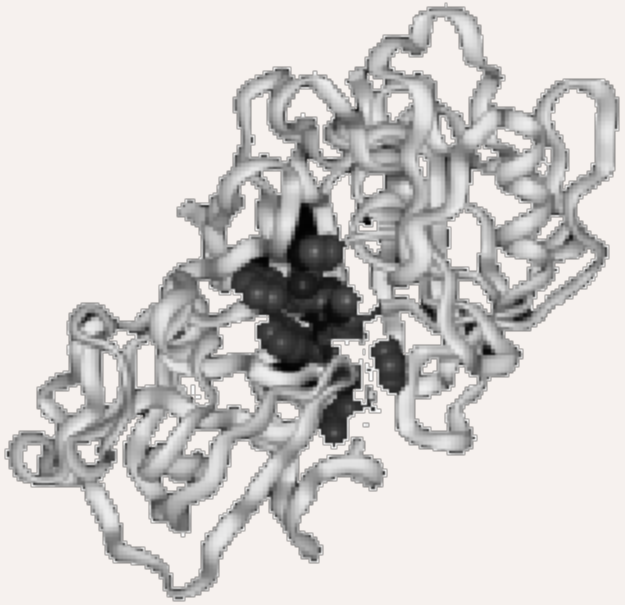
Dan E. Krane
Wright State University - Main Campus, dan.krane@wright.edu

Michael L. Raymer
Wright State University - Main Campus, michael.raymer@wright.edu

Sequence Alignments
and
Database Searches

Genes encode the recipes for proteins

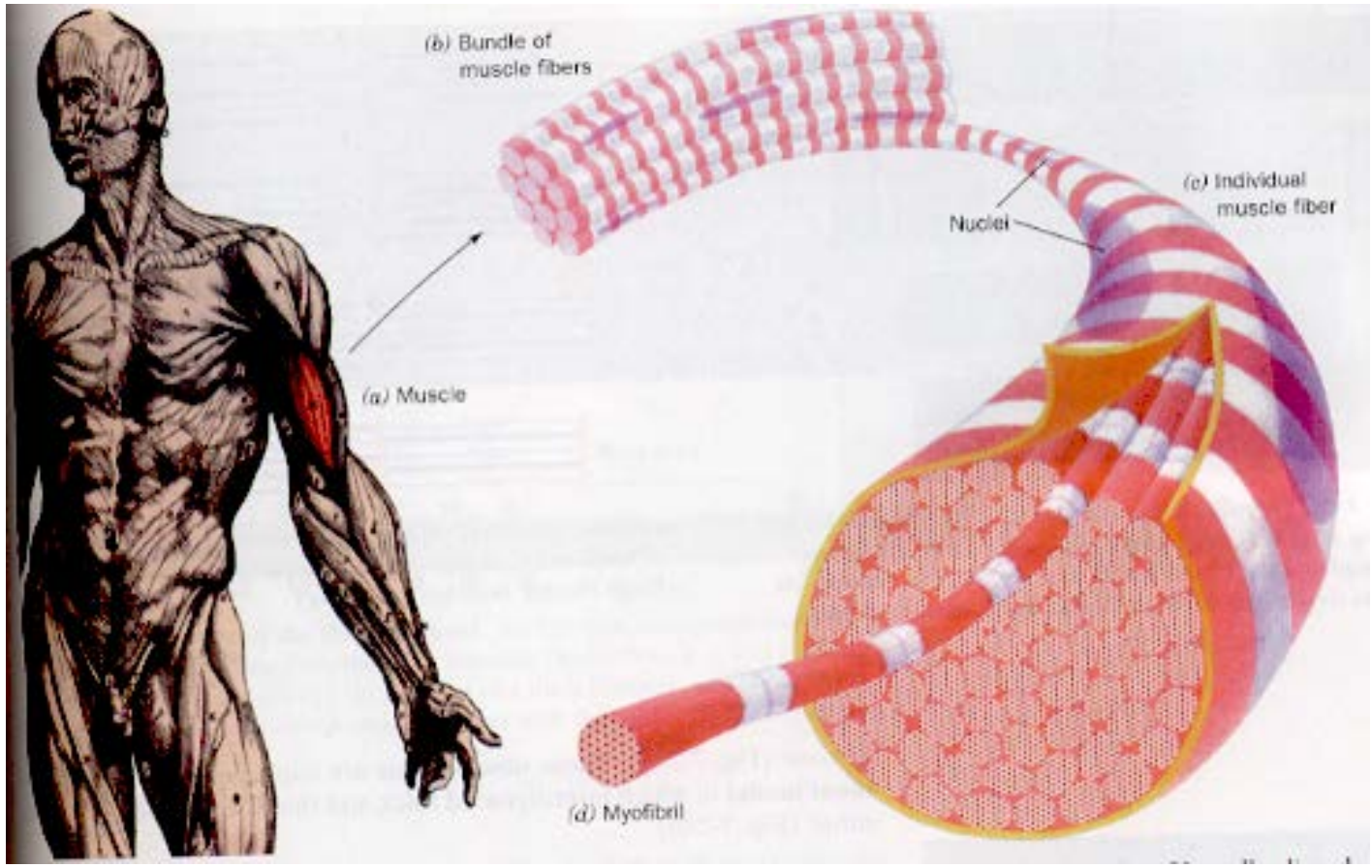




Proteins: Molecular Machines

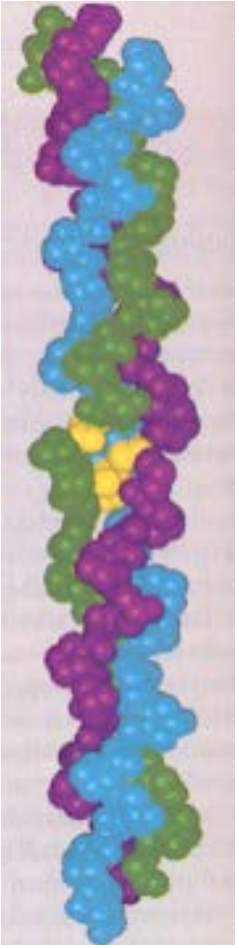
- Proteins in your muscles allows you to move:

myosin
and
actin



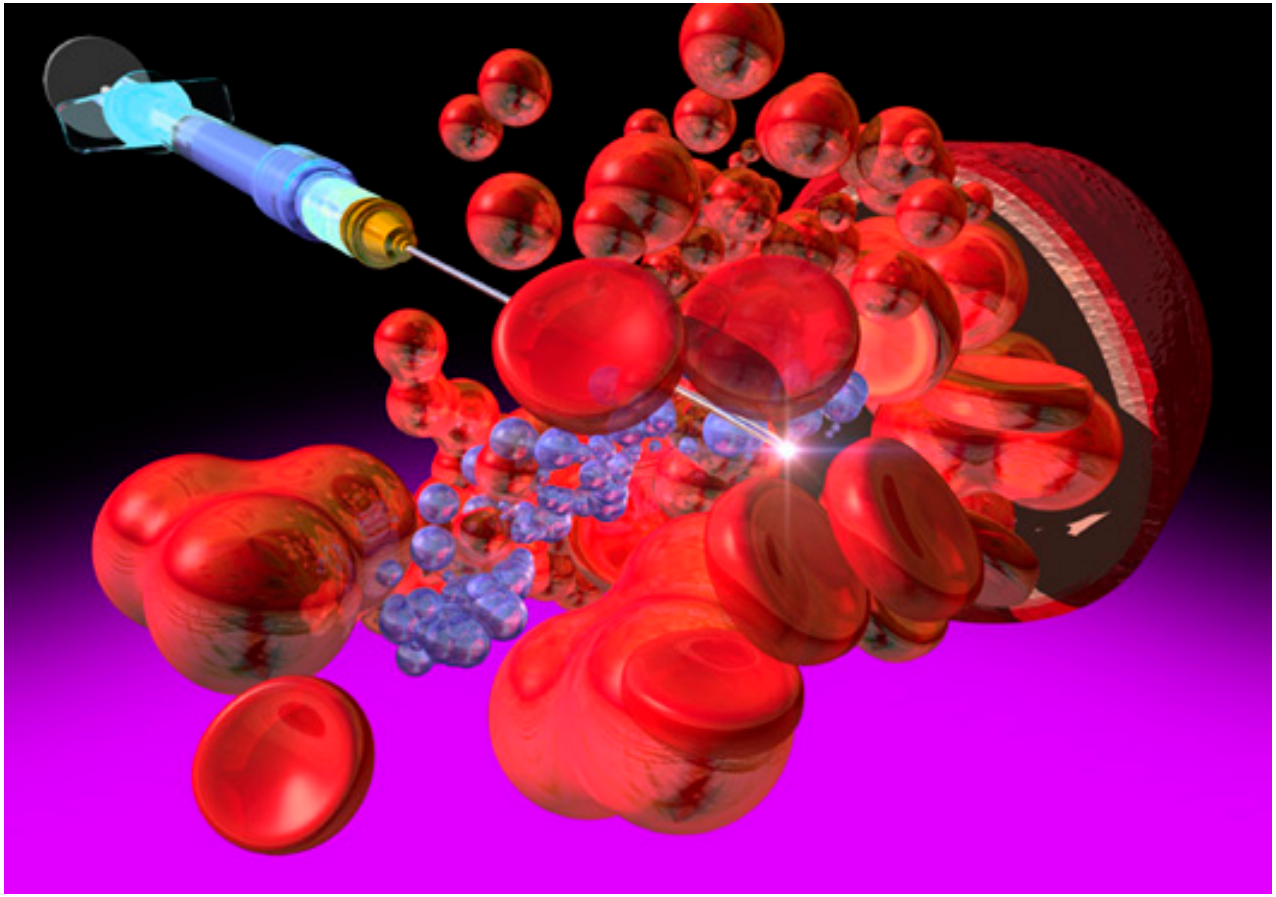
Proteins: Molecular Machines

- Enzymes
  (digestion, catalysis)
- Structure (collagen)

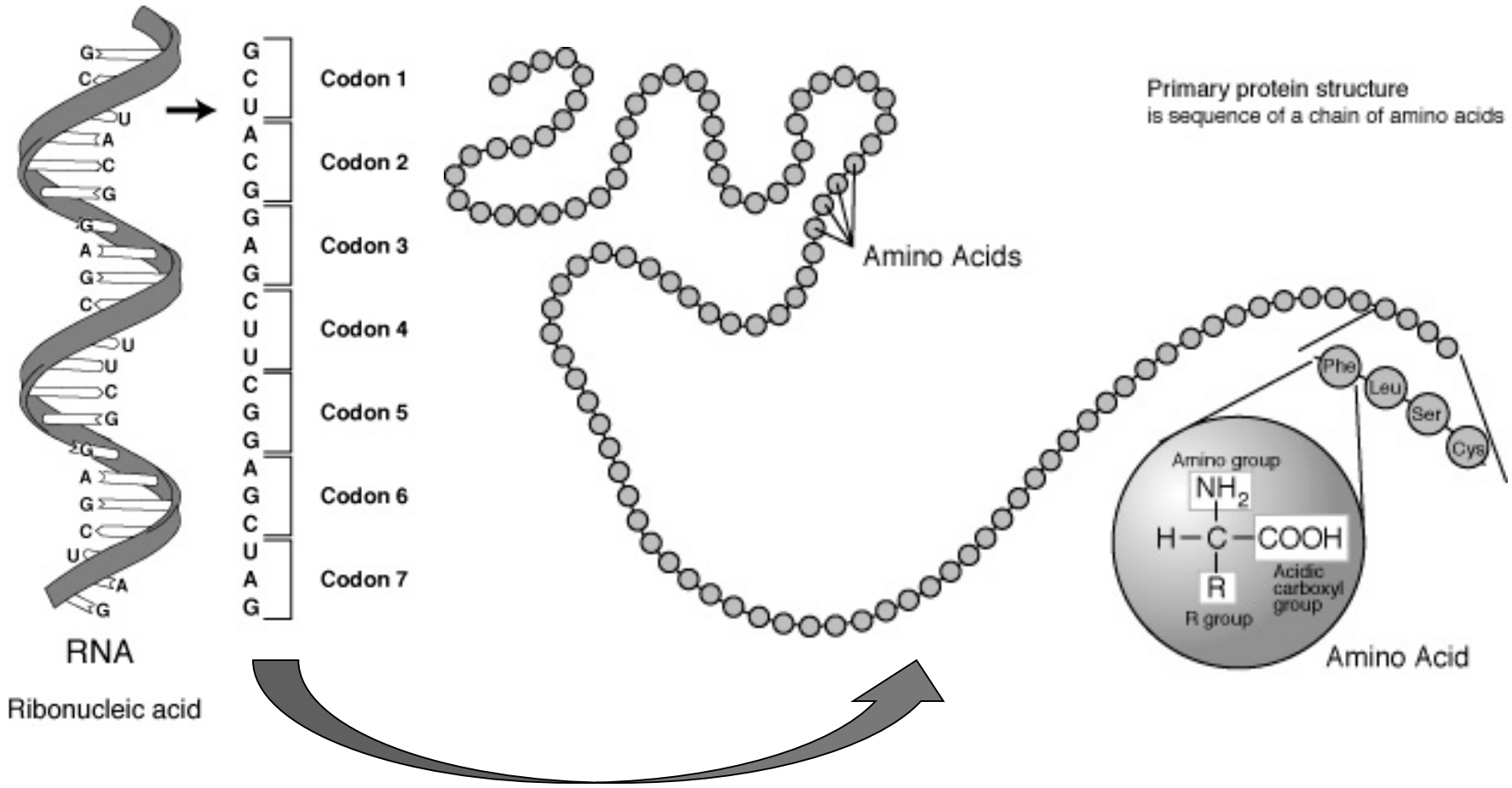


Proteins: Molecular Machines

- Signaling (hormones, kinases)
- Transport (energy, oxygen)

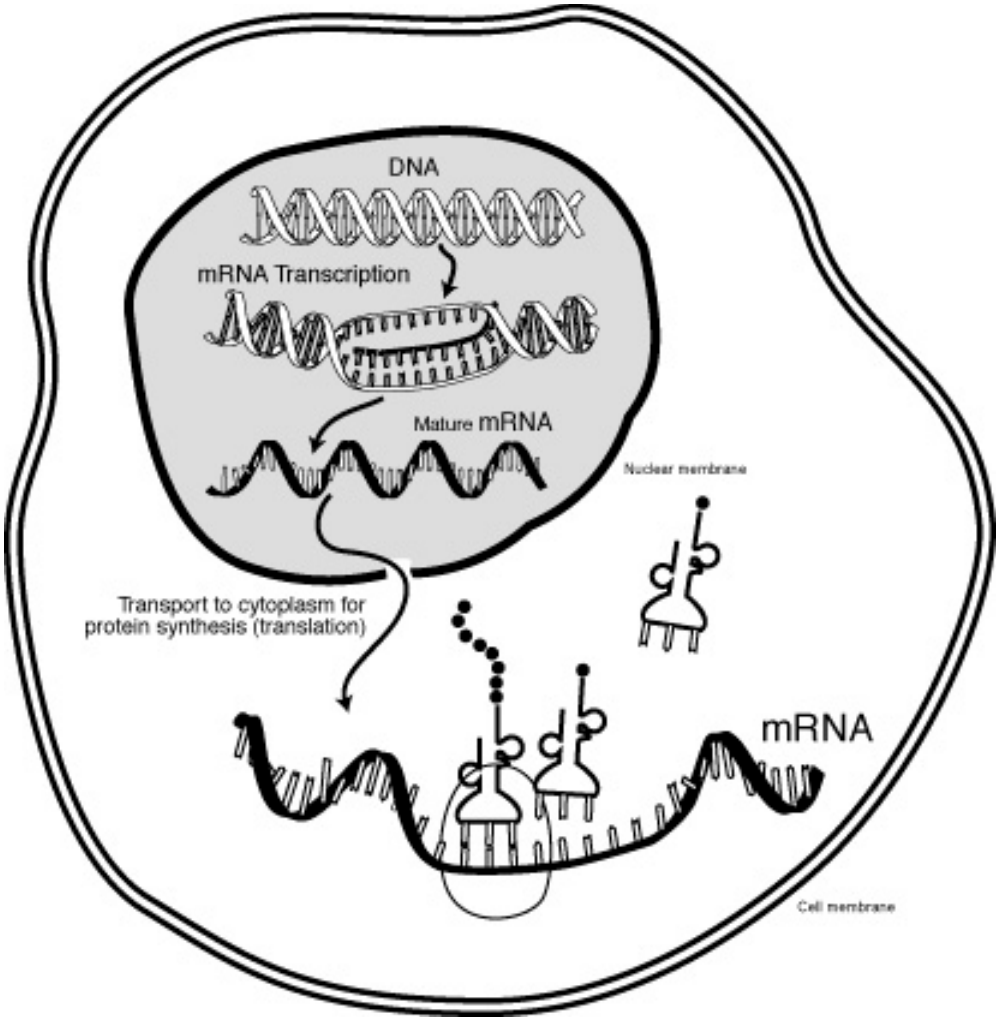


Proteins are amino acid *polymers*



Messenger RNA

- Carries instructions for a protein outside of the nucleus to the *ribosome*
- The ribosome is a protein complex that synthesizes new proteins



Transcription

The Central Dogma

DNA

transcription

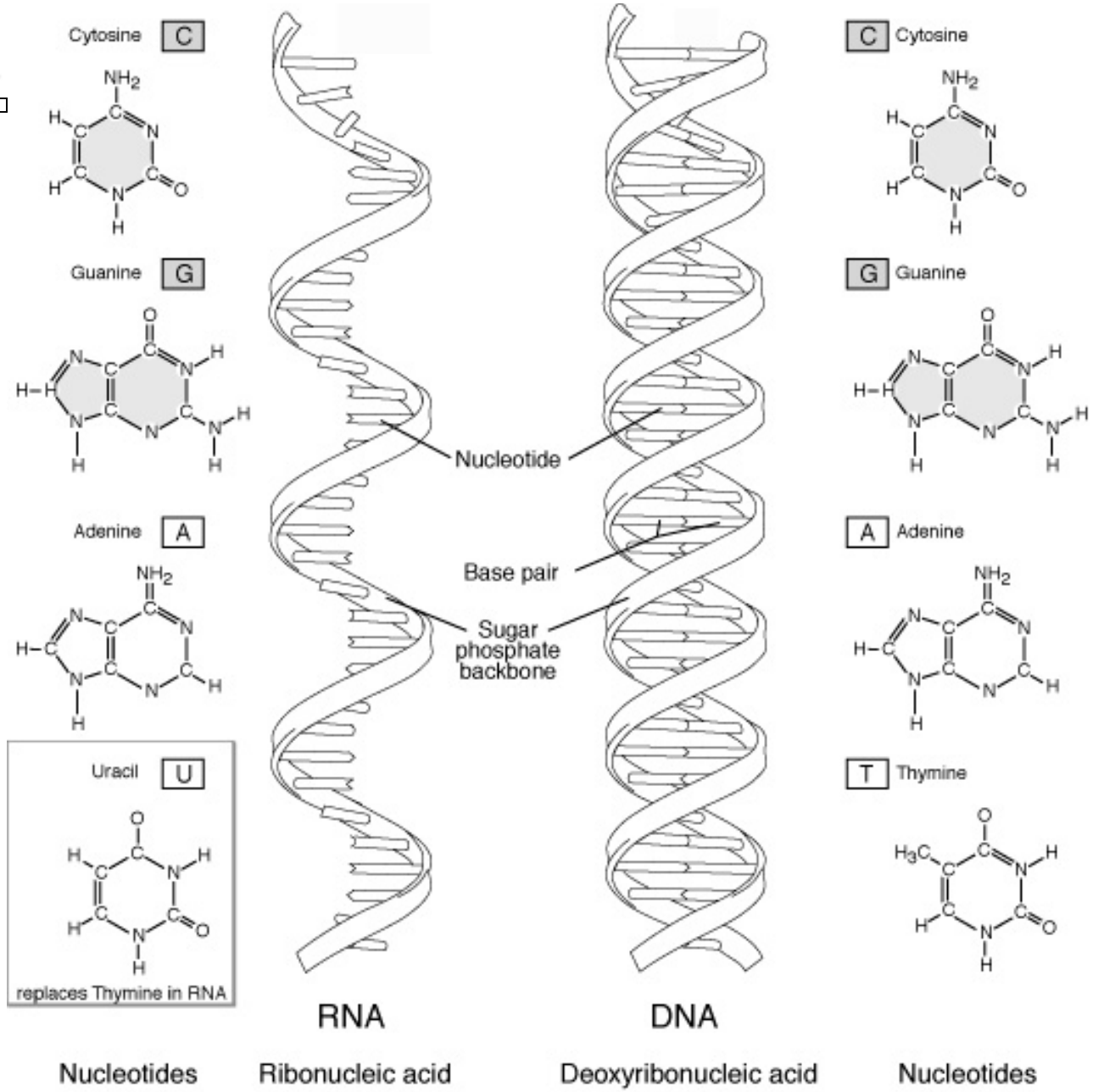
↓

RNA

translation

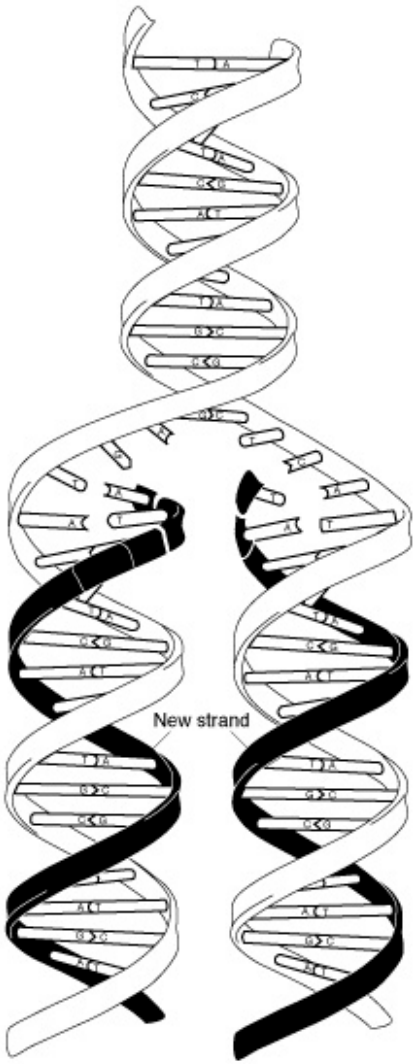
↓

Proteins



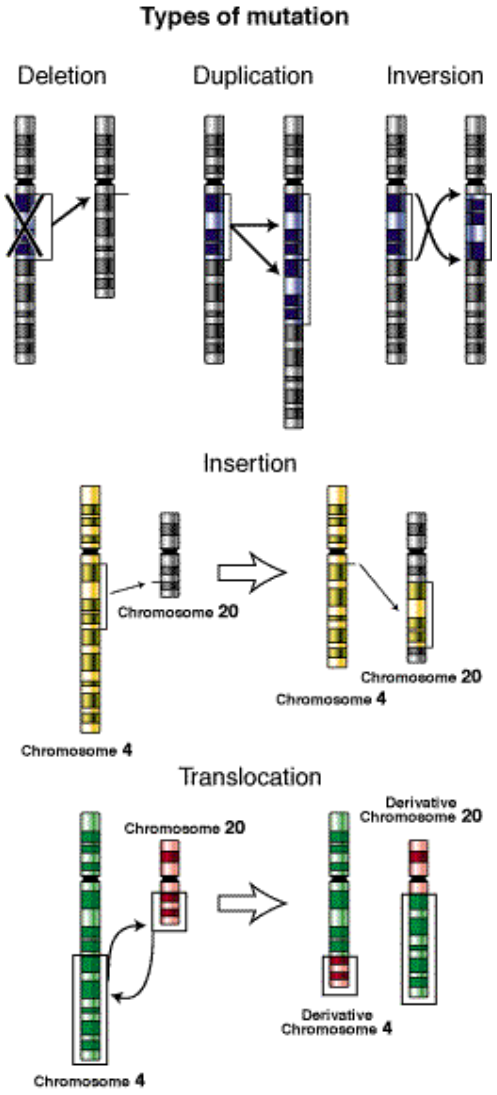
DNA Replication

- Prior to cell division, all the genetic instructions must be "copied" so that each new cell will have a complete set
- DNA polymerase is the enzyme that copies DNA
  - Reads the old strand in the 3´ to 5´ direction



Over time, genes accumulate *mutations*

- Environmental factors
  - Radiation
  - Oxidation
- Mistakes in replication or repair
  - Deletions, Duplications
  - Insertions
  - Inversions
  - Point mutations



Deletions

- Codon deletion:
  ACG ATA GCG ~~TAT~~ GTA TAG CCG…
  - Effect depends on the protein, position, etc.
  - Almost always deleterious
  - Sometimes lethal
- Frame shift mutation:
  ACG ATA GCG ~~T~~AT GTA TAG CCG…
  ACG ATA GCG ATG TAT AGC CG?…
  - Almost always lethal

Indels

- Comparing two genes it is generally impossible to tell if an *indel* is an insertion in one gene, or a deletion in another, unless ancestry is known:

```
ACGTCTGATACGCCGTATCGTCTATCT
ACGTCTGAT---CCGTATCGTCTATCT
```

The Genetic Code

First position (5' end)	U		C		A		G		Third position (3' end)
U	UUU	Phe	UCU	Ser	UAU	Tyr	UGU	Cys	U
	UUC		UCC		UAC		UGC		C
	UUA	Leu	UCA		UAA	Stop	UGA	Stop	A
	UUG		UCG		UAG	Stop	UGG	Trp	G
C	CUU	Leu	CCU	Pro	CAU	His	CGU	Arg	U
	CUC		CCC		CAC		CGC		C
	CUA		CCA		CAA	Gln	CGA		A
	CUG		CCG		CAG		CGG		G
A	AUU	Ile	ACU	Thr	AAU	Asn	AGU	Ser	U
	AUC		ACC		AAC		AGC		C
	AUA		ACA		AAA	Lys	AGA	Arg	A
	AUG		ACG		AAG		AGG		G
G	GUU	Val	GCU	Ala	GAU	Asp	GGU	Gly	U
	GUC		GCC		GAC		GGC		C
	GUA		GCA		GAA	Glu	GGA		A
	GUG		GCG		GAG		GGG		G

Amino acid names:

Ala = alanine	Gln = glutamine	Leu = leucine	Ser = serine
Arg = arginine	Glu = glutamate	Lys = lysine	Thr = threonine
Asn = asparagine	Gly = glycine	Met = methionine	Trp = tryptophan
Asp = aspartate	His = histidine	Phe = phenylalanine	Tyr = Tyrosine
Cys = cysteine	Ile = Isolevcine	Pro = proline	Val = valine

Substitutions are mutations accepted by natural selection.

Synonymous:
CG<u>C</u> ⇨ CG<u>A</u>

Non-synonymous:
GA<u>U</u> ⇨ GA<u>A</u>

Comparing two sequences

- Point mutations, easy:
  ACGTCTGAT**A**CGCC**G**TAT**A**GTCTATCT
  ACGTCTGAT**T**CGCC**C**TAT**C**GTCTATCT
- Indels are difficult, must *align* sequences:
  ACG**TC**T**GAT**A**CGCCG**TAT**AGTCTATCT**
  CTG**AT**T**CGC**A**TCGTC**TAT**CT**

  ACGTCTGAT**A**CGCCGTAT**A**GTCTATCT
  ----CTGAT**T**CGC---AT**C**GTCTATCT

Why align sequences?

- The draft human genome is available
- Automated gene finding is possible
- Gene: `AGTACGTATCGTATAGCGTAA`
  - *What does it do?*
- One approach: Is there a similar gene in another species?
  - Align sequences with known genes
  - Find the gene with the “best” match

Scoring a sequence alignment

- Match score: +1
- Mismatch score: +0
- Gap penalty: –1

```
ACGTCTGATACGCCGTATAGTCTATCT
     ||||| |||    || ||||||||
----CTGATTCGC---ATCGTCTATCT
```

- Matches: 18 × (+1)
- Mismatches: 2 × 0
- Gaps: 7 × (– 1)

Score = +11

Origination and length penalties

- We want to find alignments that are *evolutionarily likely.*
- Which of the following alignments seems more likely to you?

```
ACGTCTGATACGCCGTATAGTCTATCT
ACGTCTGAT-------ATAGTCTATCT
```

①

```
ACGTCTGATACGCCGTATAGTCTATCT
AC-T-TGA--CG-CGT-TA-TCTATCT
```

②

- We can achieve this by penalizing more for a new gap, than for extending an existing gap

Scoring a sequence alignment (2)

- Match/mismatch score: +1/+0
- Origination/length penalty: –2/–1

```
ACGTCTGATACGCCGTATAGTCTATCT
     ||||| |||    || ||||||||
----CTGATTCGC---ATCGTCTATCT
```

- Matches: 18 × (+1)
- Mismatches: 2 × 0
- Origination: 2 × (–2)
- Length: 7 × (–1)

Score = +7

How can we find an optimal alignment?

- Finding the alignment is computationally hard:
  ACGTCTGATACGCCGTATAGTCTATCT
  CTGAT---TCG—CATCGTC--T-ATCT
- C(27,7) gap positions = ~888,000 possibilities
- It's possible, as long *as we don't repeat our work*!
- Dynamic programming: The Needleman & Wunsch algorithm

What is the optimal alignment?

- ACTCG
  ACAGTAG
- Match: +1
- Mismatch: 0
- Gap: –1

Needleman-Wunsch: Step 1

- Each sequence along one axis
- Mismatch penalty multiples in first row/column
- 0 in [1,1] (or [0,0] for the CS-minded)

		A	C	T	C	G
	0	-1	-2	-3	-4	-5
A	-1	1				
C	-2					
A	-3					
G	-4					
T	-5					
A	-6					
G	-7					

Needleman-Wunsch: Step 2

- Vertical/Horiz. move: Score + (simple) gap penalty
- Diagonal move: Score + match/mismatch score
- Take the **MAX** of the three possibilities

		A	C	T	C	G
	0	-1	-2	-3	-4	-5
A	-1	1				
C	-2					
A	-3					
G	-4					
T	-5					
A	-6					
G	-7					

Needleman-Wunsch: Step 2 (cont'd)

- Fill out the rest of the table likewise…

		a	c	t	c	g
	0	-1	-2	-3	-4	-5
a	-1	1	0	-1	-2	-3
c	-2					
a	-3					
g	-4					
t	-5					
a	-6					
g	-7					

Needleman-Wunsch: Step 2 (cont'd)

- Fill out the rest of the table likewise…

		a	c	t	c	g
	0	-1	-2	-3	-4	-5
a	-1	1	0	-1	-2	-3
c	-2	0	2	1	0	-1
a	-3	-1	1	2	1	0
g	-4	-2	0	1	2	2
t	-5	-3	-1	1	1	2
a	-6	-4	-2	0	1	1
g	-7	-5	-3	-1	0	2

- The optimal alignment score is calculated in the lower-right corner

But what *is* the optimal alignment

- To reconstruct the optimal alignment, we must determine of where the MAX at each step came from…

		a	c	t	c	g
	0	-1	-2	-3	-4	-5
a	-1	1	0	-1	-2	-3
c	-2	0	2	1	0	-1
a	-3	-1	1	2	1	0
g	-4	-2	0	1	2	2
t	-5	-3	-1	1	1	2
a	-6	-4	-2	0	1	1
g	-7	-5	-3	-1	0	2

A path corresponds to an alignment

- ↑ = GAP in top sequence
- ← = GAP in left sequence
- ↖ = ALIGN both positions
- One path from the previous table: ↖↖↑↑↖↖↖
- Corresponding alignment (start at the end):

```
AC--TCG
ACAGTAG
```

Score = +2

Practice Problem

- Find an optimal alignment for these two sequences:

  `GCGGTT`

  `GCGT`

- Match: +1
- Mismatch: 0
- Gap: –1

		g	c	g	g	t	t
	0	-1	-2	-3	-4	-5	-6
g	-1						
c	-2						
g	-3						
t	-4						

Practice Problem

- Find an optimal alignment for these two sequences:

  GCGGTT

  GCGT

		g	c	g	g	t	t
	0	-1	-2	-3	-4	-5	-6
g	-1	1	0	-1	-2	-3	-4
c	-2	0	2	1	0	-1	-2
g	-3	-1	1	3	2	1	0
t	-4	-2	0	2	3	3	2

GCGGTT

GCG-T-

Score = +2

What are all these numbers, anyway?

- Suppose we are aligning:

  A with A…

		a
	0	-1
a	-1	

The dynamic programming concept

- Suppose we are aligning:
  ACTCG
  ACAGTAG
- Last position choices:

G G	+1	ACTC ACAGTA
G -	-1	ACTC ACAGTAG
- G	-1	ACTCG ACAGTA

Semi-global alignment

- Suppose we are aligning:
  `GCG`
  `GGCG`
- Which do you prefer?
  `G-CG` `-GCG`
  `GGCG` `GGCG`

		g	c	g
	0	-1	-2	-3
g	-1	1	0	-1
g	-2	0	1	1
c	-3	-1	1	1
g	-4	-2	0	2

- Semi-global alignment allows gaps at the ends for free.

Semi-global alignment

- Semi-global alignment allows gaps at the ends for free.

		g	c	g
	0	0	0	0
g	0	1	0	1
g	0	1	1	1
c	0	0	2	1
g	0	1	1	3

  - Initialize first row and column to all 0's
  - Allow free horizontal/vertical moves in last row and column

Local alignment

- Global alignments – score the entire alignment
- Semi-global alignments – allow unscored gaps at the beginning or end of either sequence
- Local alignment – find the best matching subsequence
- CG**ATG**
  AA**ATG**GA
- This is achieved by allowing a 4th alternative at each position in the table: zero.

Local alignment

- Mismatch = –1 this time

		c	g	a	t	g
	0	-1	-2	-3	-4	-5
a	-1	0	0	0	0	0
a	-2	0	0	1	0	0
a	-3	0	0	1	0	0
t	-4	0	0	0	2	1
g	-5	0	1	0	1	3
g	-6	0	1	0	0	2
a	-7	0	0	2	1	1

CG**ATG**

AA**ATG**GA

CS790 Assignment #1

- Look up the *principal of optimality*, as it applies to dynamic programming. In no more than one single-spaced page, describe how dynamic programming in general, and the principal of optimality in particular apply to the Needleman-Wunsch algorithm.
- Due on Tues, 4/16.