Combinatorial approaches to RNA folding
Part II: Energy minimization via dynamic programming

Matthew Macauley

Department of Mathematical Sciences
Clemson University
http://www.math.clemson.edu/~macaule/

Math 4500, Fall 2016

Overview

Overview

Main question

Given a raw sequence of RNA, can we predict how it will fold?

Overview

Main question

Given a raw sequence of RNA, can we predict how it will fold?

There are two main approaches to this problem:

1. Energy minimization. Calculate the "free energy" of a folded structure. The "most likely" structures tend to be those where free energy is minimized.

Overview

Main question

Given a raw sequence of RNA, can we predict how it will fold?

There are two main approaches to this problem:

1. Energy minimization. Calculate the "free energy" of a folded structure. The "most likely" structures tend to be those where free energy is minimized.

   The free energy is computed recursively using dynamic programming.

Overview

Main question

Given a raw sequence of RNA, can we predict how it will fold?

There are two main approaches to this problem:

1. Energy minimization. Calculate the "free energy" of a folded structure. The "most likely" structures tend to be those where free energy is minimized.

   The free energy is computed recursively using dynamic programming.

2. Formal language theory. Use a formal grammar to algorithmically generate secondary structures: production rules convert symbols into strings according to the langauge's syntax.

Overview

Main question

Given a raw sequence of RNA, can we predict how it will fold?

There are two main approaches to this problem:

1. Energy minimization. Calculate the "free energy" of a folded structure. The "most likely" structures tend to be those where free energy is minimized.

   The free energy is computed recursively using dynamic programming.

2. Formal language theory. Use a formal grammar to algorithmically generate secondary structures: production rules convert symbols into strings according to the langauge's syntax.

   If we assign probabilities to the rules, then the "most likely" structure is the one that ocurrs with the highest probability.

Overview

Main question

Given a raw sequence of RNA, can we predict how it will fold?

There are two main approaches to this problem:

1. Energy minimization. Calculate the "free energy" of a folded structure. The "most likely" structures tend to be those where free energy is minimized.

   The free energy is computed recursively using dynamic programming.

2. Formal language theory. Use a formal grammar to algorithmically generate secondary structures: production rules convert symbols into strings according to the langauge's syntax.

   If we assign probabilities to the rules, then the "most likely" structure is the one that ocurrs with the highest probability.

In this lecture, we will study the energy minimization approach.

Dynamic programming

Dynamic programming (DP) is a method for solving complex problems by solving simplier subproblems and combining their solutions to obtain the overall solution.

The CG pairing uses 3 hydrogen bonds, whereas AU and UG each use 2.

The wobble pair UG is unstable, having roughly half the strength of an AU bond.

Given an RNA sequence $\mathbf{b} = b_1 b_2 \cdots b_n$, define the energy function of a pair:

$$e(i,j) = \begin{cases} 3 & \{b_i, b_j\} = \{\mathrm{C}, \mathrm{G}\} \text{ and } i \leq j-4 \\ 2 & \{b_i, b_j\} = \{\mathrm{A}, \mathrm{U}\} \text{ and } i \leq j-4 \\ 1 & \{b_i, b_j\} = \{\mathrm{G}, \mathrm{U}\} \text{ and } i \leq j-4 \\ 0 & \text{otherwise} \end{cases}$$

Assume energy is additive:

$$E(\mathbf{b}, S) = \sum_{\text{bp's } (i,j)} e(i,j)\,.$$

Dynamic programming

Dynamic programming (DP) is a method for solving complex problems by solving simplier subproblems and combining their solutions to obtain the overall solution.

The CG pairing uses 3 hydrogen bonds, whereas AU and UG each use 2.

The wobble pair UG is unstable, having roughly half the strength of an AU bond.

Given an RNA sequence $\mathbf{b} = b_1 b_2 \cdots b_n$, define the energy function of a pair:

$$e(i,j) = \begin{cases} 3 & \{b_i, b_j\} = \{\mathrm{C}, \mathrm{G}\} \text{ and } i \leq j-4 \\ 2 & \{b_i, b_j\} = \{\mathrm{A}, \mathrm{U}\} \text{ and } i \leq j-4 \\ 1 & \{b_i, b_j\} = \{\mathrm{G}, \mathrm{U}\} \text{ and } i \leq j-4 \\ 0 & \text{otherwise} \end{cases}$$

Assume energy is additive:

$$E(\mathbf{b}, S) = \sum_{\text{bp's } (i,j)} e(i,j)\,.$$

Goal

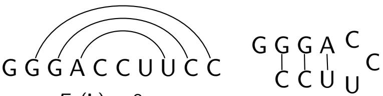
Maximize the energy score $E(\mathbf{b}, S)$ over all possible secondary structures S into which $\mathbf{b}$ can fold.

Warm-up Exercise

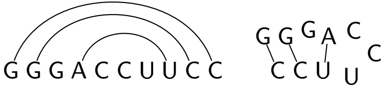
Consider the RNA sequence $\mathbf{b} = \mathrm{G\,G\,G\,A\,C\,C\,U\,U\,C\,C}$. Find all possible ways that it can fold into a secondary structure S, without leaving any "allowed" unpaired bases. Draw the arc diagram and a "realistic sketch" of the folded RNA strand. Compute the energy score $E(\mathbf{b}, S)$ of each.

Warm-up Exercise

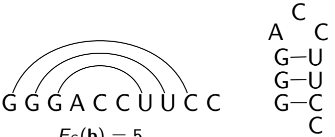
Consider the RNA sequence $\mathbf{b} = \text{G G G A C C U U C C}$. Find all possible ways that it can fold into a secondary structure S, without leaving any "allowed" unpaired bases. Draw the arc diagram and a "realistic sketch" of the folded RNA strand. Compute the energy score $E(\mathbf{b}, S)$ of each.



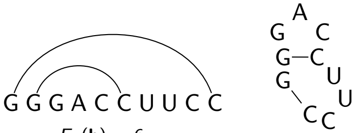
$E_S(\mathbf{b}) = 8$



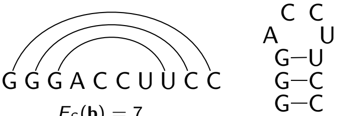
$E_S(\mathbf{b}) = 8$



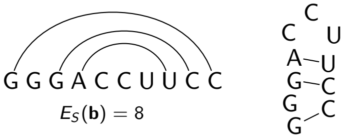
$E_S(\mathbf{b}) = 5$



$E_S(\mathbf{b}) = 6$



$E_S(\mathbf{b}) = 7$



$E_S(\mathbf{b}) = 8$

Dynamic programming

Dynamic Programming (DP) process

1. Use the optimal energy score of subsequences of **b** to determine the optimal (maximum) energy score of **b**.
2. Traceback to reconstruct the actual secondary structure that realizes this maximum.

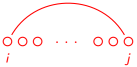
Dynamic programming

Dynamic Programming (DP) process

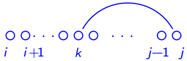
1. Use the optimal energy score of subsequences of **b** to determine the optimal (maximum) energy score of **b**.
2. Traceback to reconstruct the actual secondary structure that realizes this maximum.

We compute $E(i,j)$ recursively. If $i < j-4$, then we return $E(i,j)=0$. Otherwise, there are four ways to recurse on the sequence $\mathbf{b}_{i,j}$:

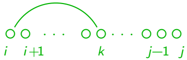
1. *(i,j) forms a basepair*: Recurse on the sequence $\mathbf{b}_{i+1,j-1}$.



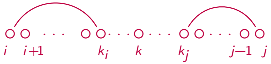
2. *i is unpaired but (k,j) is a basepair*: Recurse on the sequence $\mathbf{b}_{i+1,j}$.



3. *(i,k) is a basepair but j is unpaired*: Recurse on the sequence $\mathbf{b}_{i,j-1}$.



4. *(i,k_i) and (k_j,j) are paired for some $k_i < k_j$.* Recurse on the two subsequences $\mathbf{b}_{i,k}$ and $\mathbf{b}_{k+1,j}$, for some $k_i \leq k < k_j$. *We need to consider all possible values of $k = i+4,\ldots,j-4$.*



Dynamic programming

Taking the maximum energy score over each of four ways to recurse on the subsequence $\mathbf{b}_{i,j}$ yields a recurrence for the the maximum score $E(i,j)$:

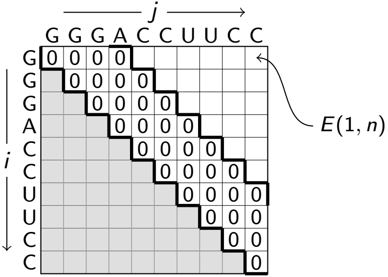
$$E(i,j) = \max \begin{cases} E(i+1,j-1) + e(i,j) \\ E(i+1,j) \\ E(i,j-1) \\ \max\limits_{i<k<j} E(i,k) + E(k+1,j). \end{cases}$$

Dynamic programming

Taking the maximum energy score over each of four ways to recurse on the subsequence $\mathbf{b}_{i,j}$ yields a recurrence for the the maximum score $E(i,j)$:

$$E(i,j) = \max \begin{cases} E(i+1,j-1) + e(i,j) \\ E(i+1,j) \\ E(i,j-1) \\ \max\limits_{i<k<j} E(i,k) + E(k+1,j). \end{cases}$$

The values of $E(i,j)$ can be arranged in a table. *The optimal energy score $E(\mathbf{b}, S)$ is simply $E(1,n)$.*



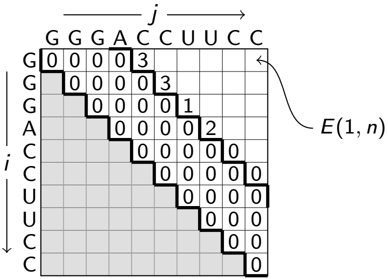
	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0						
G		0	0	0	0					
G			0	0	0	0				
A				0	0	0	0			
C					0	0	0	0		
C						0	0	0	0	
U							0	0	0	0
U								0	0	0
C									0	0
C										0

Dynamic programming

Taking the maximum energy score over each of four ways to recurse on the subsequence $\mathbf{b}_{i,j}$ yields a recurrence for the the maximum score $E(i,j)$:

$$E(i,j) = \max \begin{cases} E(i+1, j-1) + e(i,j) \\ E(i+1, j) \\ E(i, j-1) \\ \max\limits_{i<k<j} E(i,k) + E(k+1,j). \end{cases}$$

The values of $E(i,j)$ can be arranged in a table. *The optimal energy score $E(\mathbf{b}, S)$ is simply $E(1,n)$.*



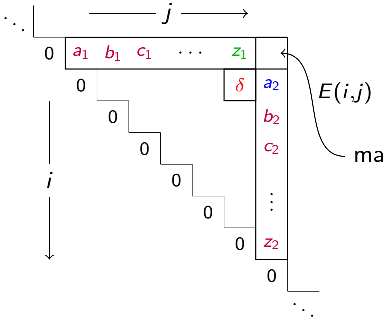
	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3					
G		0	0	0	0	3				
G			0	0	0	0	1			
A				0	0	0	0	2		
C					0	0	0	0	0	
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

Dynamic programming

Exercise

Fill out the remaining table for the sequence $\mathbf{b} = \mathrm{G\,G\,G\,A\,C\,C\,U\,U\,C\,C}$.

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3					
G		0	0	0	0	3				
G			0	0	0	0	1			
A				0	0	0	0	2		
C					0	0	0	0	0	
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



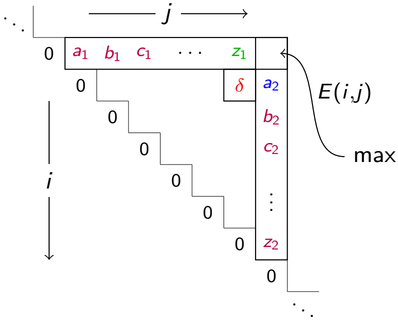
$$\max\begin{cases} E(i+1,j-1)+e(i,j)=\delta+e(i,j) \\ E(i+1,j)=a_2 \\ E(i,j-1)=z_1 \\ \max\limits_{i<k<j}\{E(i,k)+E(k+1,j)\} \\ \qquad =\max\{a_1+a_2,\,b_1+b_2,\ldots,z_1+z_2\} \end{cases}$$

Dynamic programming

Exercise

Fill out the remaining table for the sequence $\mathbf{b} = \mathrm{G\,G\,G\,A\,C\,C\,U\,U\,C\,C}$.

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3				
G		0	0	0	0	3	3			
G			0	0	0	0	1	2		
A				0	0	0	0	2	2	
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



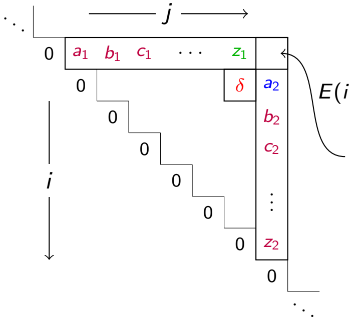
$$
\begin{cases}
E(i+1,j-1)+e(i,j)=\delta+e(i,j) \\
E(i+1,j)=a_2 \\
E(i,j-1)=z_1 \\
\max\limits_{i<k<j}\{E(i,k)+E(k+1,j)\} \\
\qquad = \max\{a_1+a_2,\, b_1+b_2, \dots, z_1+z_2\}
\end{cases}
$$

Dynamic programming

Exercise

Fill out the remaining table for the sequence $\mathbf{b} = \mathrm{G\,G\,G\,A\,C\,C\,U\,U\,C\,C}$.

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4			
G		0	0	0	0	3	3	3		
G			0	0	0	0	1	2	5	
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



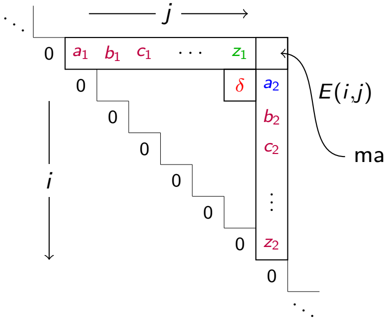
$$\max \begin{cases} E(i+1,j-1) + e(i,j) = \delta + e(i,j) \\ E(i+1,j) = a_2 \\ E(i,j-1) = z_1 \\ \displaystyle\max_{i<k<j} \{E(i,k) + E(k+1,j)\} \\ \qquad = \max\{a_1+a_2, b_1+b_2, \dots, z_1+z_2\} \end{cases}$$

Dynamic programming

Exercise

Fill out the remaining table for the sequence $\mathbf{b} = \mathrm{G\,G\,G\,A\,C\,C\,U\,U\,C\,C}$.

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4		
G		0	0	0	0	3	3	3	5	
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



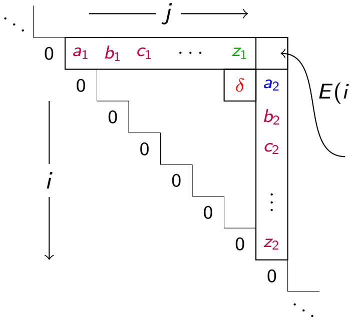
$$\max\begin{cases} E(i+1,j-1)+e(i,j) = \delta + e(i,j) \\ E(i+1,j) = a_2 \\ E(i,j-1) = z_1 \\ \max\limits_{i<k<j}\{E(i,k)+E(k+1,j)\} \\ \qquad = \max\{a_1+a_2, b_1+b_2, \dots, z_1+z_2\} \end{cases}$$

Dynamic programming

Exercise

Fill out the remaining table for the sequence **b** = G G G A C C U U C C.

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



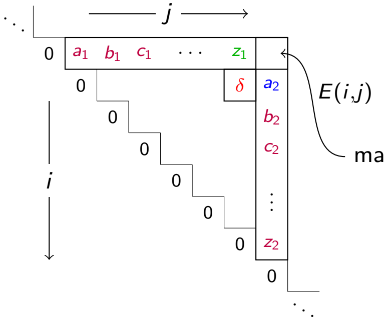
$$\max\begin{cases} E(i+1,j-1)+e(i,j)=\delta+e(i,j) \\ E(i+1,j)=a_2 \\ E(i,j-1)=z_1 \\ \max\limits_{i<k<j}\{E(i,k)+E(k+1,j)\} \\ \qquad =\max\{a_1+a_2,b_1+b_2,\dots,z_1+z_2\} \end{cases}$$

Dynamic programming

Exercise

Fill out the remaining table for the sequence $\mathbf{b} = \mathrm{G\,G\,G\,A\,C\,C\,U\,U\,C\,C}$.

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



$$\max \begin{cases} E(i+1,j-1)+e(i,j) = \delta + e(i,j) \\ E(i+1,j) = a_2 \\ E(i,j-1) = z_1 \\ \max\limits_{i<k<j}\{E(i,k)+E(k+1,j)\} \\ \qquad = \max\{a_1+a_2, b_1+b_2, \ldots, z_1+z_2\} \end{cases}$$

Dynamic programming: traceback step

Goal

Now that we know the optimal energy score, we need to determine which secondary structure realizes that score.

Exercise

"Trace back" the following table to find a minimal free energy secondary structure for $\mathbf{b} = \text{G G G A C C U U C C}$.

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

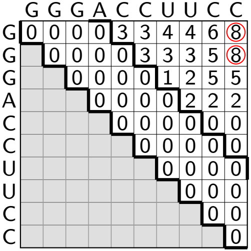
Dynamic programming: traceback step

Goal

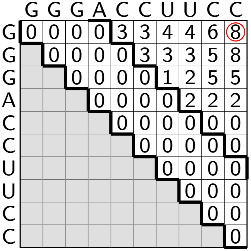
Now that we know the optimal energy score, we need to determine which secondary structure realizes that score.

Exercise

"Trace back" the following table to find a minimal free energy secondary structure for $\mathbf{b} = \text{G G G A C C U U C C}$.



	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

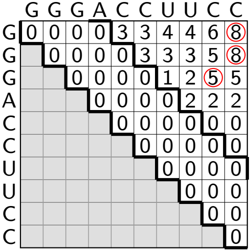
Dynamic programming: traceback step

Goal

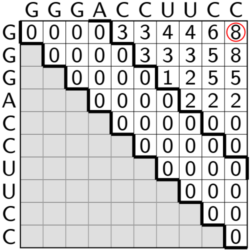
Now that we know the optimal energy score, we need to determine which secondary structure realizes that score.

Exercise

"Trace back" the following table to find a minimal free energy secondary structure for $\mathbf{b} = \mathrm{G\,G\,G\,A\,C\,C\,U\,U\,C\,C}$.



	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	⑧
G		0	0	0	0	3	3	3	5	⑧
G			0	0	0	0	1	2	⑤	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	⑧
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

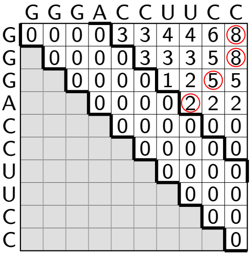
Dynamic programming: traceback step

Goal

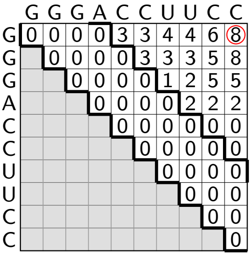
Now that we know the optimal energy score, we need to determine which secondary structure realizes that score.

Exercise

"Trace back" the following table to find a minimal free energy secondary structure for $\mathbf{b} = \mathrm{G\,G\,G\,A\,C\,C\,U\,U\,C\,C}$.



	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	⑧
G		0	0	0	0	3	3	3	5	⑧
G			0	0	0	0	1	2	⑤	5
A				0	0	0	0	②	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	⑧
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

Dynamic programming: traceback step

Goal

Now that we know the optimal energy score, we need to determine which secondary structure realizes that score.

Exercise

"Trace back" the following table to find a minimal free energy secondary structure for $\mathbf{b} = GGGACCUUCC$.

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	⑧
G		0	0	0	0	3	3	3	5	⑧
G			0	0	0	0	1	2	⑤	5
A				0	0	0	0	②	2	2
C					0	0	⓪	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	⑧
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

Dynamic programming: traceback step

Goal

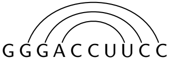
Now that we know the optimal energy score, we need to determine which secondary structure realizes that score.

Exercise

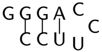
"Trace back" the following table to find a minimal free energy secondary structure for $\mathbf{b} = \text{G G G A C C U U C C}$.

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



$E_S(\mathbf{b}) = 8$



Dynamic programming: traceback step

Goal

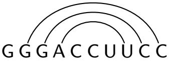
Now that we know the optimal energy score, we need to determine which secondary structure realizes that score.

Exercise

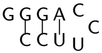
"Trace back" the following table to find a minimal free energy secondary structure for $\mathbf{b} = \text{G G G A C C U U C C}$.

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	⑧
G		0	0	0	0	3	3	3	5	⑧
G			0	0	0	0	1	2	⑤	5
A				0	0	0	0	②	2	2
C					0	0	⓪	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	⑧
G		0	0	0	0	3	3	3	⑤	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



$E_S(\mathbf{b}) = 8$



Dynamic programming: traceback step

Goal

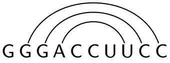
Now that we know the optimal energy score, we need to determine which secondary structure realizes that score.

Exercise

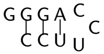
"Trace back" the following table to find a minimal free energy secondary structure for $\mathbf{b} = \text{G G G A C C U U C C}$.

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



$E_S(\mathbf{b}) = 8$



Dynamic programming: traceback step

Goal

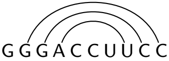
Now that we know the optimal energy score, we need to determine which secondary structure realizes that score.

Exercise

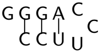
"Trace back" the following table to find a minimal free energy secondary structure for $\mathbf{b} = \mathrm{G\,G\,G\,A\,C\,C\,U\,U\,C\,C}$.

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



$E_S(\mathbf{b}) = 8$



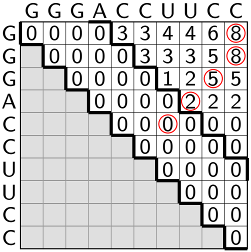
Dynamic programming: traceback step

Goal

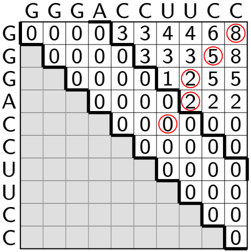
Now that we know the optimal energy score, we need to determine which secondary structure realizes that score.

Exercise

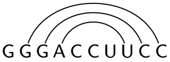
"Trace back" the following table to find a minimal free energy secondary structure for $\mathbf{b} = \mathrm{G\,G\,G\,A\,C\,C\,U\,U\,C\,C}$.



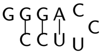
	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0



$E_S(\mathbf{b}) = 8$



Dynamic programming: traceback step

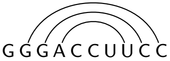
Goal

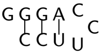
Now that we know the optimal energy score, we need to determine which secondary structure realizes that score.

Exercise

"Trace back" the following table to find a minimal free energy secondary structure for $\mathbf{b} = \text{G G G A C C U U C C}$.

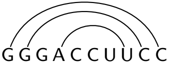
	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0

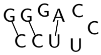




$E_S(\mathbf{b}) = 8$

	G	G	G	A	C	C	U	U	C	C
G	0	0	0	0	3	3	4	4	6	8
G		0	0	0	0	3	3	3	5	8
G			0	0	0	0	1	2	5	5
A				0	0	0	0	2	2	2
C					0	0	0	0	0	0
C						0	0	0	0	0
U							0	0	0	0
U								0	0	0
C									0	0
C										0





$E_S(\mathbf{b}) = 8$

Model validity and weaknesses

Model validity and weaknesses

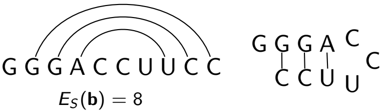
Question

Did we *really* find the most thermodynamically stable folds of the RNA sequence $\mathbf{b} = \mathrm{G\,G\,G\,A\,C\,C\,U\,U\,C\,C}$?

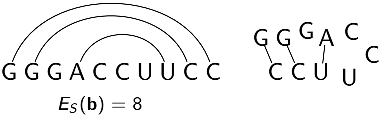
Model validity and weaknesses

Question

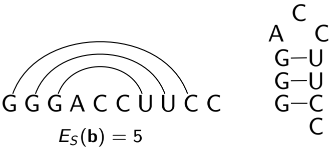
Did we *really* find the most thermodynamically stable folds of the RNA sequence $\mathbf{b} = \mathrm{G\,G\,G\,A\,C\,C\,U\,U\,C\,C}$?



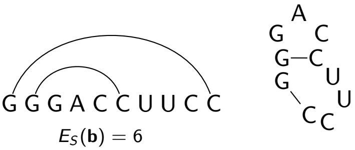
$E_S(\mathbf{b}) = 8$



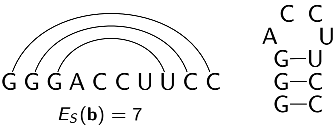
$E_S(\mathbf{b}) = 8$



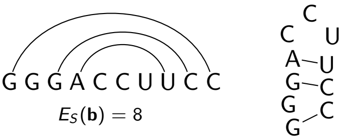
$E_S(\mathbf{b}) = 5$



$E_S(\mathbf{b}) = 6$



$E_S(\mathbf{b}) = 7$



$E_S(\mathbf{b}) = 8$

Model validity and weaknesses

Model validity and weaknesses

One major weakness of this DP algorithm is that it does not take loop structure into account.

Model validity and weaknesses

One major weakness of this DP algorithm is that it does not take loop structure into account.

The free energy of a secondary structure is the amount of energy needed to maintain its structural integrity.

Model validity and weaknesses

One major weakness of this DP algorithm is that it does not take loop structure into account.

The free energy of a secondary structure is the amount of energy needed to maintain its structural integrity.

Structures with positive free energy are unstable, and structures with negative free energy are stable.

Model validity and weaknesses

One major weakness of this DP algorithm is that it does not take loop structure into account.

The free energy of a secondary structure is the amount of energy needed to maintain its structural integrity.

Structures with positive free energy are unstable, and structures with negative free energy are stable.

Big idea

The free energy of a secondary structure is determined by two factors:

Model validity and weaknesses

One major weakness of this DP algorithm is that it does not take loop structure into account.

The free energy of a secondary structure is the amount of energy needed to maintain its structural integrity.

Structures with positive free energy are unstable, and structures with negative free energy are stable.

Big idea

The free energy of a secondary structure is determined by two factors:

1. its base pairs, which are stabilizing (contribute negative free energy)

Model validity and weaknesses

One major weakness of this DP algorithm is that it does not take loop structure into account.

The free energy of a secondary structure is the amount of energy needed to maintain its structural integrity.

Structures with positive free energy are unstable, and structures with negative free energy are stable.

Big idea

The free energy of a secondary structure is determined by two factors:

1. its base pairs, which are stabilizing (contribute negative free energy)
2. its loops, which are destabilizing (contribute positive free energy)

Model validity and weaknesses

One major weakness of this DP algorithm is that it does not take loop structure into account.

The free energy of a secondary structure is the amount of energy needed to maintain its structural integrity.

Structures with positive free energy are unstable, and structures with negative free energy are stable.

Big idea

The free energy of a secondary structure is determined by two factors:

1. its base pairs, which are stabilizing (contribute negative free energy)
2. its loops, which are destabilizing (contribute positive free energy)

The most likely secondary structure is the one with the minimal free energy.

Model validity and weaknesses

One major weakness of this DP algorithm is that it does not take loop structure into account.

The free energy of a secondary structure is the amount of energy needed to maintain its structural integrity.

Structures with positive free energy are unstable, and structures with negative free energy are stable.

Big idea

The free energy of a secondary structure is determined by two factors:

1. its base pairs, which are stabilizing (contribute negative free energy)
2. its loops, which are destabilizing (contribute positive free energy)

The most likely secondary structure is the one with the minimal free energy.

In our previous DP model, we were only taking the base pairs into consideration – not loops.

Model validity and weaknesses

One major weakness of this DP algorithm is that it does not take loop structure into account.

The free energy of a secondary structure is the amount of energy needed to maintain its structural integrity.

Structures with positive free energy are unstable, and structures with negative free energy are stable.

Big idea

The free energy of a secondary structure is determined by two factors:

1. its base pairs, which are stabilizing (contribute negative free energy)
2. its loops, which are destabilizing (contribute positive free energy)

The most likely secondary structure is the one with the minimal free energy.

In our previous DP model, we were only taking the base pairs into consideration – not loops.

(And technically, our energy scores should have been negative, with our goal being to *minimize* free energy.)

Incorporating loop structure

Incorporating loop structure

Recall that every secondary structure has a unique loop decomposition.

Incorporating loop structure

Recall that every secondary structure has a unique loop decomposition.

That is, every vertex is part of some loop, which is one of the following:

0. null loop

Incorporating loop structure

Recall that every secondary structure has a unique loop decomposition.

That is, every vertex is part of some loop, which is one of the following:

0. null loop
1. hairpin loop

2a. stacked pair (stabilizing!)

2b. bulge loop

2c. interior loop

Incorporating loop structure

Recall that every secondary structure has a unique loop decomposition.

That is, every vertex is part of some loop, which is one of the following:

0. null loop
1. hairpin loop

2a. stacked pair (stabilizing!)

2b. bulge loop

2c. interior loop

3+. multiloop

Incorporating loop structure

Recall that every secondary structure has a unique loop decomposition.

That is, every vertex is part of some loop, which is one of the following:

0. null loop
1. hairpin loop

2a. stacked pair (stabilizing!)

2b. bulge loop

2c. interior loop

3+. multiloop

Note that this takes into account the negative energy contributions by the base pairs.

Incorporating loop structure

Recall that every secondary structure has a unique loop decomposition.

That is, every vertex is part of some loop, which is one of the following:

0. null loop
1. hairpin loop
2a. stacked pair (stabilizing!)
2b. bulge loop
2c. interior loop
3+. multiloop

Note that this takes into account the negative energy contributions by the base pairs.

We assume that loop energy contributions are additive, and thus the free energy is:

$$E_S(\mathbf{b}) = \sum_{\text{all loops } \mathbf{L}} e(\mathbf{L}) = e(\mathbf{L}_0) + \left(\sum_{(i,j)} e(L((i,j))) \right).$$

Incorporating loop structure

Incorporating loop structure

The energy contribution of a loop depends on a number of parameters:

Incorporating loop structure

The energy contribution of a loop depends on a number of parameters:

- its type

Incorporating loop structure

The energy contribution of a loop depends on a number of parameters:

- its type
- its size

Incorporating loop structure

The energy contribution of a loop depends on a number of parameters:

- its type
- its size
- its "closing base pairs"

Incorporating loop structure

The energy contribution of a loop depends on a number of parameters:

- its type
- its size
- its "closing base pairs"
- many other technicalities and special cases

There are many of these "special cases," which lead to energy penalities or bonuses.

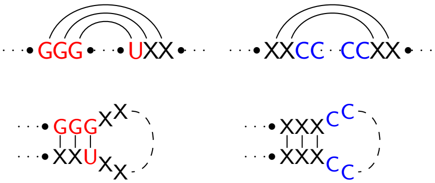
Incorporating loop structure

The energy contribution of a loop depends on a number of parameters:

- its type
- its size
- its "closing base pairs"
- many other technicalities and special cases

There are many of these "special cases," which lead to energy penalities or bonuses.

Two examples of these "special cases" are GGG-loops and poly-C hairpin loops:



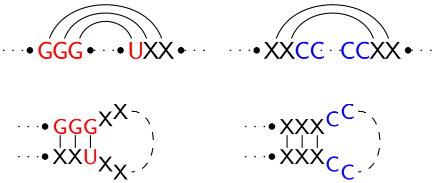
Incorporating loop structure

The energy contribution of a loop depends on a number of parameters:

- its type
- its size
- its "closing base pairs"
- many other technicalities and special cases

There are many of these "special cases," which lead to energy penalities or bonuses.

Two examples of these "special cases" are GGG-loops and poly-C hairpin loops:



These special cases result in the model having hundreds of parameters, most of which are experimentally determined.

Incorporating loop structure

In the theory of polymers, the free energy of a hairpin loop **L** of size l_s is approximately

$$e(\mathbf{L}) = 1.75 \cdot R \cdot T \cdot \ln(\ell_s)\,,$$

where T is temperature, and $R \approx 1.987$ cal/mol/K is a universal gas constant.

Incorporating loop structure

In the theory of polymers, the free energy of a hairpin loop **L** of size l_s is approximately

$$e(\mathbf{L}) = 1.75 \cdot R \cdot T \cdot \ln(\ell_s)\,,$$

where T is temperature, and $R \approx 1.987$ cal/mol/K is a universal gas constant.

However, this isn't accurate for very small (size $\ell_s = 3$ or 4), or large (size $\ell_s > 30$) hairpin loops. Extra penalities and/or bonuses need to be added for these.

Incorporating loop structure

In the theory of polymers, the free energy of a hairpin loop $\mathbf{L}$ of size l_s is approximately

$$e(\mathbf{L}) = 1.75 \cdot R \cdot T \cdot \ln(\ell_s)\,,$$

where T is temperature, and $R \approx 1.987$ cal/mol/K is a universal gas constant.

However, this isn't accurate for very small (size $\ell_s = 3$ or 4), or large (size $\ell_s > 30$) hairpin loops. Extra penalities and/or bonuses need to be added for these.

The thermodynamics of multiloops are poorly understood. Biochemists have proposed the following model:

$$e(\mathbf{L}) = a + 6b + 1.75R \cdot T \cdot \ln(\ell_s(\mathbf{L})/6) + c \cdot \ell_{\#bp}(\mathbf{L}) + e(\mathbf{L}_{\text{stack}})$$

where the error term $e(\mathbf{L}_{\text{stack}})$ accounts for stacking interactions, and a, b, and c are experimentally determined constants.

Incorporating loop structure

In the theory of polymers, the free energy of a hairpin loop **L** of size l_s is approximately

$$e(\mathbf{L}) = 1.75 \cdot R \cdot T \cdot \ln(\ell_s)\,,$$

where T is temperature, and $R \approx 1.987$ cal/mol/K is a universal gas constant.

However, this isn't accurate for very small (size $\ell_s = 3$ or 4), or large (size $\ell_s > 30$) hairpin loops. Extra penalities and/or bonuses need to be added for these.

The thermodynamics of multiloops are poorly understood. Biochemists have proposed the following model:

$$e(\mathbf{L}) = a + 6b + 1.75R \cdot T \cdot \ln(\ell_s(\mathbf{L})/6) + c \cdot \ell_{\#bp}(\mathbf{L}) + e(\mathbf{L}_{\text{stack}})$$

where the error term $e(\mathbf{L}_{\text{stack}})$ accounts for stacking interactions, and a, b, and c are experimentally determined constants.

The state-of-the-art (energy-based) RNA folding software is UNAFold (an upgraded version of mfold.

Incorporating loop structure

In the theory of polymers, the free energy of a hairpin loop **L** of size l_s is approximately

$$e(\mathbf{L}) = 1.75 \cdot R \cdot T \cdot \ln(\ell_s)\,,$$

where T is temperature, and $R \approx 1.987$ cal/mol/K is a universal gas constant.

However, this isn't accurate for very small (size $\ell_s = 3$ or 4), or large (size $\ell_s > 30$) hairpin loops. Extra penalities and/or bonuses need to be added for these.

The thermodynamics of multiloops are poorly understood. Biochemists have proposed the following model:

$$e(\mathbf{L}) = a + 6b + 1.75R \cdot T \cdot \ln(\ell_s(\mathbf{L})/6) + c \cdot \ell_{\#bp}(\mathbf{L}) + e(\mathbf{L}_{\text{stack}})$$

where the error term $e(\mathbf{L}_{\text{stack}})$ accounts for stacking interactions, and a, b, and c are experimentally determined constants.

The state-of-the-art (energy-based) RNA folding software is UNAFold (an upgraded version of mfold.

UNIFold stores these parameters in text files, which are called upon when the algorithm runs.

Energies of stacked pairs

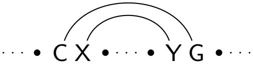
Energies of stacked pairs

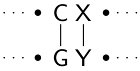
The energy value of a stacked pair (X, Y) depends not only on X and Y, but on what arc it is "stacked underneath."

Energies of stacked pairs

The energy value of a stacked pair (X, Y) depends not only on X and Y, but on what arc it is "stacked underneath."

For example, suppose (X, Y) is stacked beneath a CG bond. UNAFold compute the energy contribution of the stacked pair (X, Y) by looking it up in a preprocessed text file:



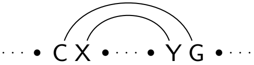


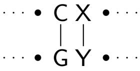
X \ Y	A	C	G	U
A	∞	∞	∞	-2.1
C	∞	∞	-3.3	∞
G	∞	-2.4	∞	-1.4
U	-2.1	∞	-2.1	∞

Energies of stacked pairs

The energy value of a stacked pair (X, Y) depends not only on X and Y, but on what arc it is "stacked underneath."

For example, suppose (X, Y) is stacked beneath a CG bond. UNAFold compute the energy contribution of the stacked pair (X, Y) by looking it up in a preprocessed text file:





X \ Y	A	C	G	U
A	∞	∞	∞	-2.1
C	∞	∞	-3.3	∞
G	∞	-2.4	∞	-1.4
U	-2.1	∞	-2.1	∞

The table above is for pairs stacked above a CG bond. Since there are 6 possibilities for what (X, Y) can be stacked under, UNAFold has 6 tables like the one above.

UNAFold

UNAFold "Unified Nucleic Acid Folding" is a comprehensive software package for nucleic acid folding and hybridization prediction.

It was developed by Michael Zuker's research group at RPI and SUNY Albany. It is freely available from the following website:

`http://mfold.rna.albany.edu/`

Also available there is extensive documentation and literature about the software and the algorithms.

It essentially runs a complicated dynamic programming algorithm, albeit with a hundreds of special cases and parameters.