`Infrared`[1]: A Modelling Framework for Targeting Complex Features (Positive Design)

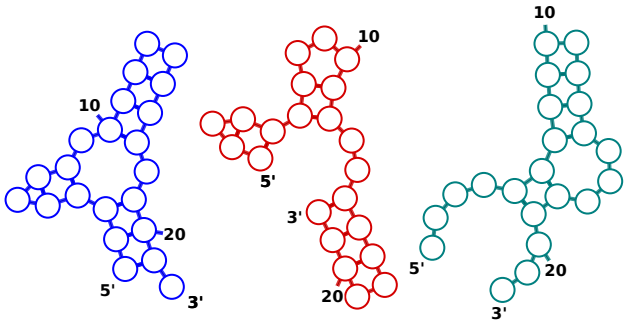
Sebastian Will
(w/ Stefan Hammer and Yann Ponty)

TBI Winterseminar in Bled 2019

[1] providing the infrastructure for RNARedPrint v2

Positive and negative RNA design

Multiple target structures




```
(((((.)).(((..))).))).
((.))((...))..(((..)))
....((((..))...))...
```

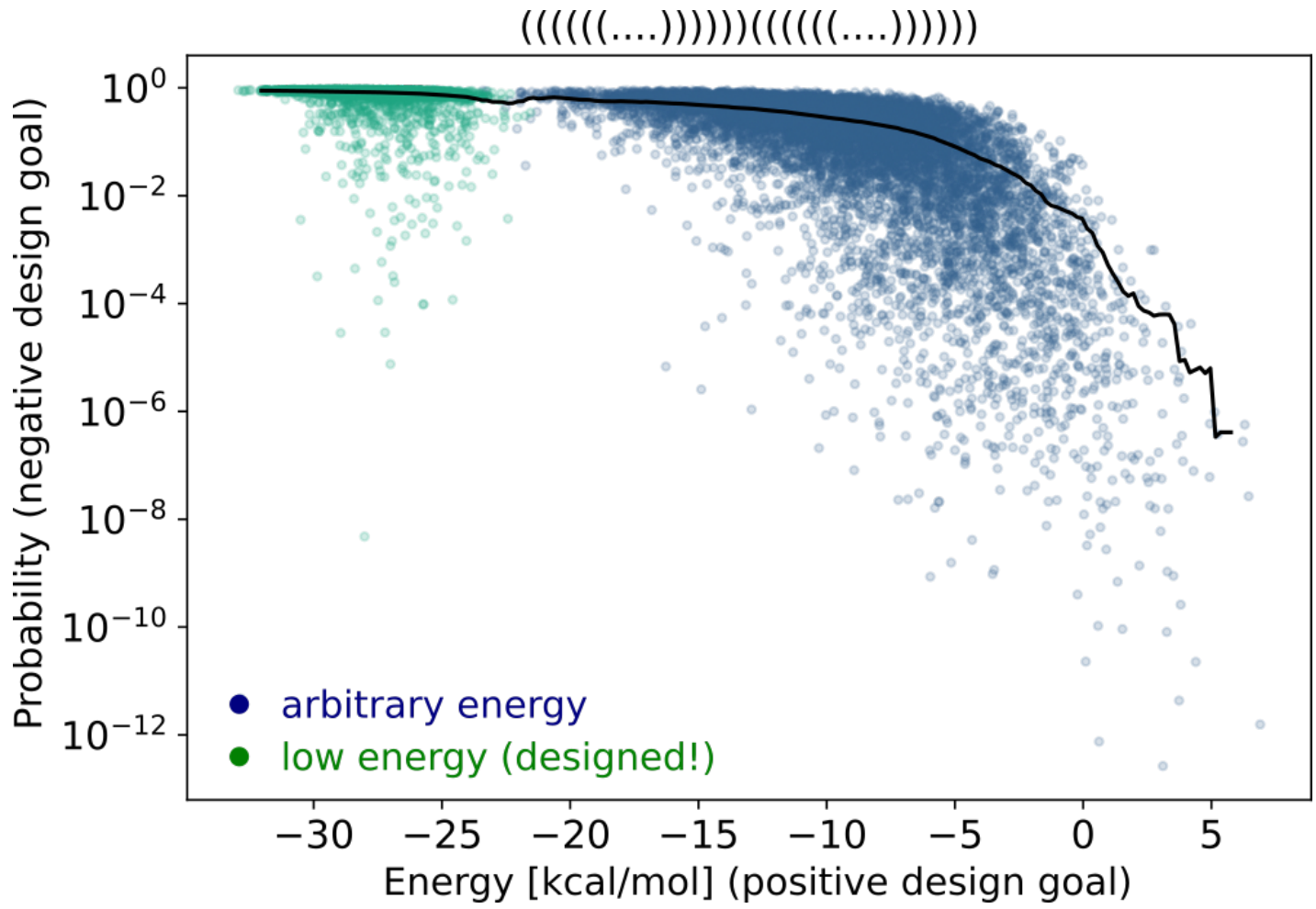
- **Negative RNA design**
  Design sequences, s.t. the target structure(s) have the best energies among **all** structures. (**Avoid** good energies for all other structures.)

  = **OUT**-design

- **Positive RNA design**
  Design sequences, s.t. **the target** structure(s) have specific (typically, good) energies.

  = **IN**-design

Positive design supports negative design



The challenge of positive design

Given is a secondary structure

$R =$ `(((((((..(((((((....)))))((((((......))..))))..((((((....((((...))))).....))))))..)))..)))))))`

- Generate 1000 uniform random compatible sequences ✓

The challenge of positive design

Given is a secondary structure

$R =$ `((((((..(((((((....)))))((((((......))..))))..((((((....((((...))))).....))))))..)))..)))))))`

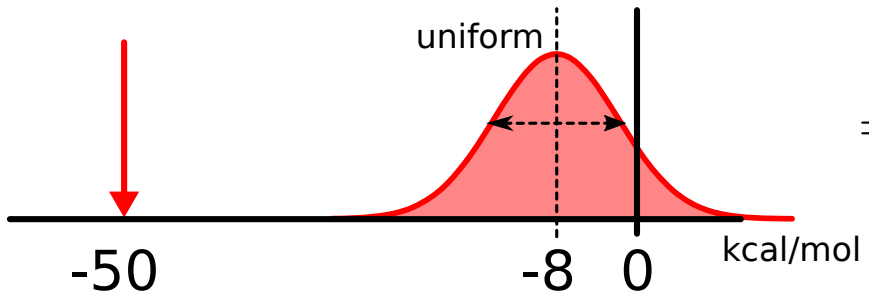
- Generate 1000 uniform random compatible sequences ✓
- Generate 1000 sequences where $E(R) \approx -50\mathrm{kcal/mol}$

The challenge of positive design

Given is a secondary structure

$R =$ ((((((..(((((((....)))))((((((......))..))))..((((((....((((...))))).....))))))..)))..)))))))

- Generate 1000 uniform random compatible sequences ✓
- Generate 1000 sequences where $E(R) \approx -50\mathrm{kcal/mol}$
  - **Uniform sampling:**



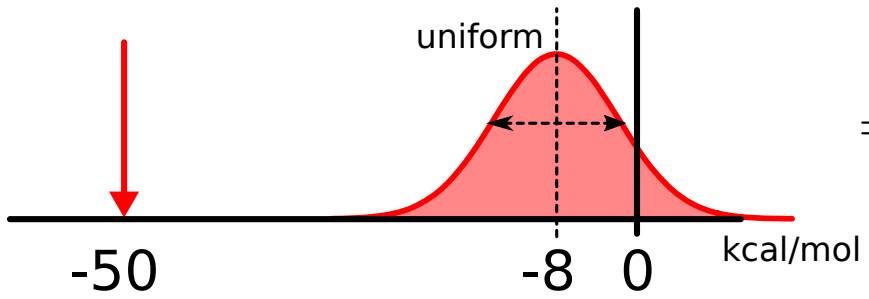
$\Rightarrow$ quasi-∞ time 

The challenge of positive design

Given is a secondary structure

$R =$ ((((((..(((((((....)))))((((((......))..))))..((((((....((((...))))).....))))))..)))..)))))))

- Generate 1000 uniform random compatible sequences ✓
- Generate 1000 sequences where $E(R) \approx -50\mathrm{kcal/mol}$
  - **Uniform sampling:**



$\Rightarrow$ quasi-∞ time 

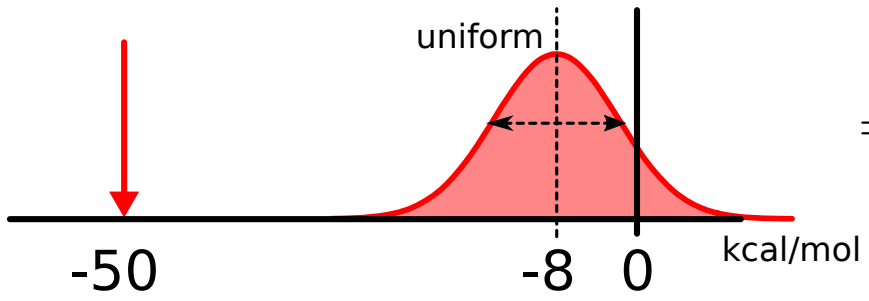
  - **Infrared** (multi-dim. Boltzmann sampling): less than 5s! 

The challenge of positive design

Given is a secondary structure

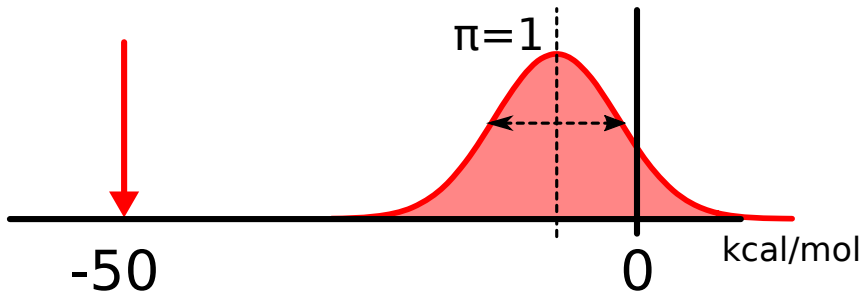
$R =$ ((((((..((((((....))))((((((......))..))))..((((((....(((((...)))).....))))))..)))..)))))))

- Generate 1000 uniform random compatible sequences ✓
- Generate 1000 sequences where $E(R) \approx -50\mathrm{kcal/mol}$
  - **Uniform sampling:**



$\Rightarrow$ quasi-∞ time

  - **Infrared** (multi-dim. Boltzmann sampling): less than 5s!

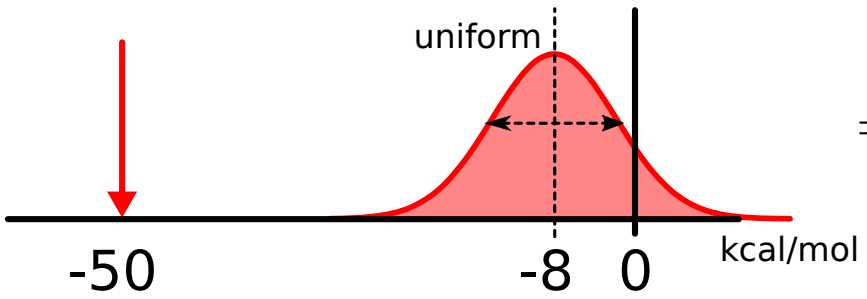


The challenge of positive design

Given is a secondary structure

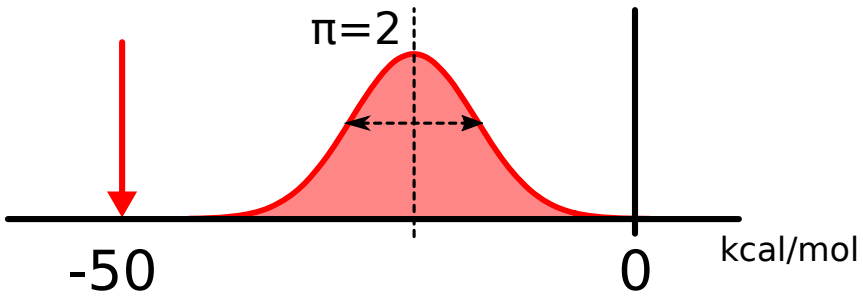
$R =$ ((((((..((((((....))))((((((......))..))))..((((((....((((...))))).....))))))..)))..)))))))

- Generate 1000 uniform random compatible sequences ✓
- Generate 1000 sequences where $E(R) \approx -50\text{kcal/mol}$
  - **Uniform sampling:**



$\Rightarrow$ quasi-∞ time

  - **Infrared** (multi-dim. Boltzmann sampling): less than 5s!

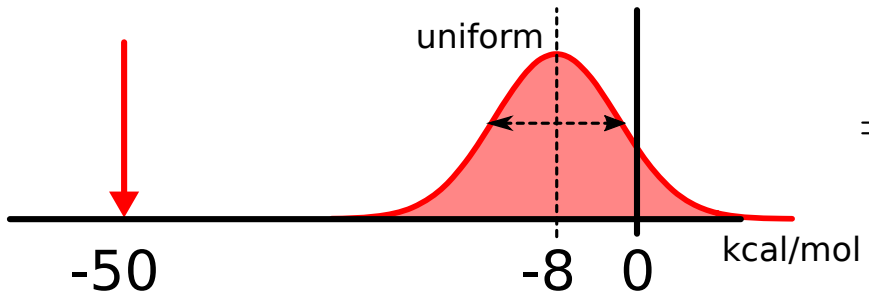


The challenge of positive design

Given is a secondary structure

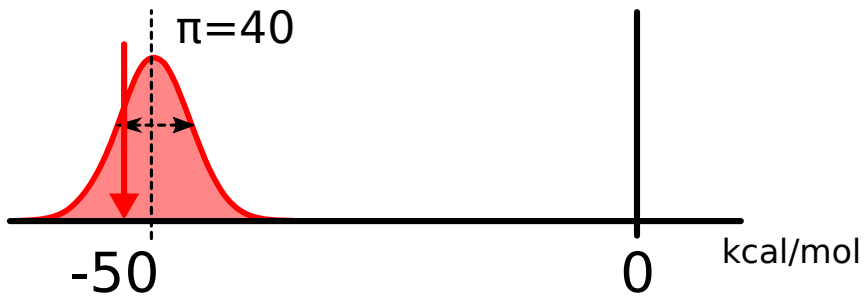
$R =$ ((((((..((((((....))))((((((......))..))))..((((((....(((((...)))).....))))))..)))..)))))))

- Generate 1000 uniform random compatible sequences ✓
- Generate 1000 sequences where $E(R) \approx -50\mathrm{kcal/mol}$
  - **Uniform sampling:**



$\Rightarrow$ quasi-∞ time 

  - **Infrared** (multi-dim. Boltzmann sampling): less than 5s! 

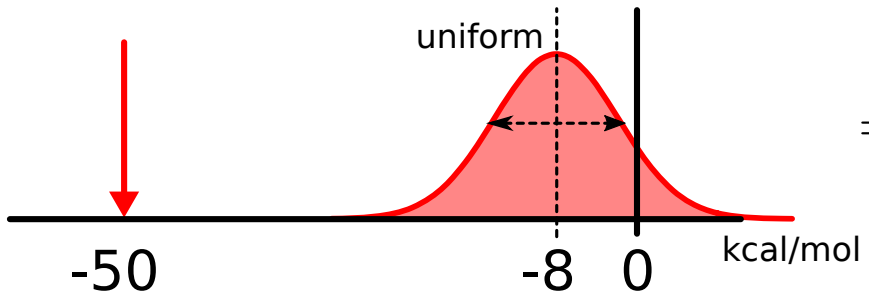


The challenge of positive design

Given is a secondary structure

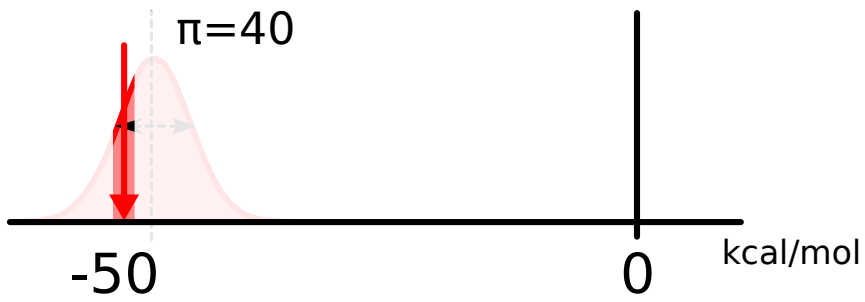
$R =$ ((((((..((((((....))))((((((......))..))))..((((((....(((((...)))).....))))))..)))..)))))))

- Generate 1000 uniform random compatible sequences ✓
- Generate 1000 sequences where $E(R) \approx -50\text{kcal/mol}$
  - **Uniform sampling:**



$\Rightarrow$ quasi-∞ time

  - **Infrared** (multi-dim. Boltzmann sampling): less than 5s!

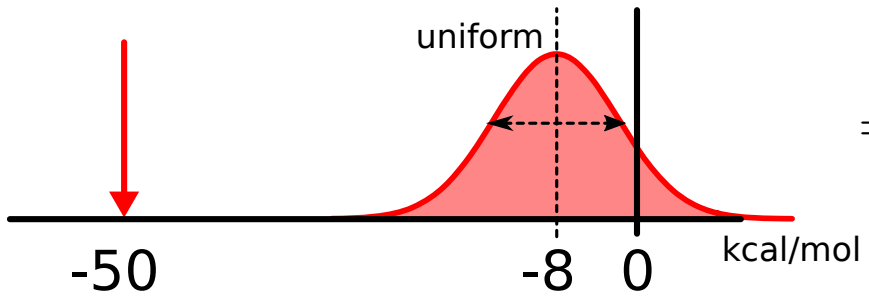


The challenge of positive design

Given is a secondary structure

$R =$ ((((((..(((((((....)))))((((((......))..))))..((((((....(((((...)))))....))))))..)))..)))))))

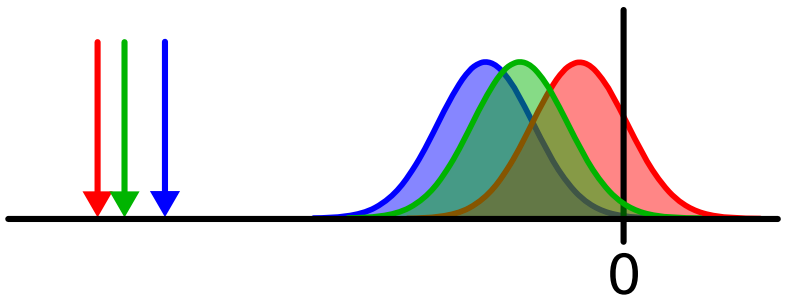
- Generate 1000 uniform random compatible sequences ✓
- Generate 1000 sequences where $E(R) \approx -50\mathrm{kcal/mol}$
  - **Uniform sampling:**



$\Rightarrow$ quasi-∞ time

  - **Infrared** (multi-dim. Boltzmann sampling): less than 5s!

- Now: 1000 samples with very low energies for **three** targets

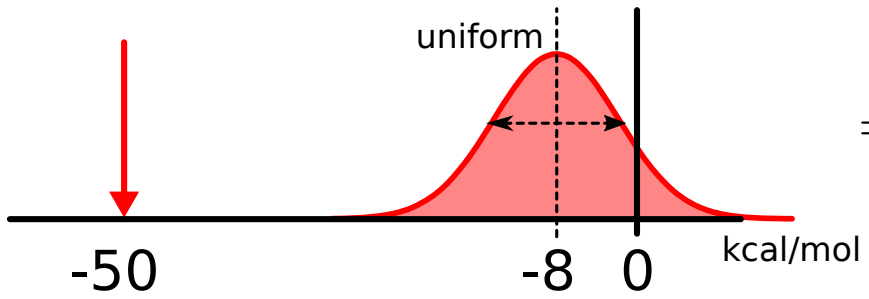


The challenge of positive design

Given is a secondary structure

$R =$ ((((((..(((((((....)))))((((((......))..))))..((((((....(((((...)))))....))))))..)))..)))))))

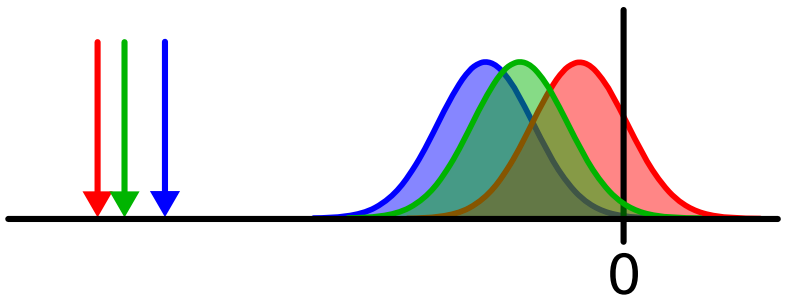
- Generate 1000 uniform random compatible sequences ✓
- Generate 1000 sequences where $E(R) \approx -50\mathrm{kcal/mol}$
  - **Uniform sampling:**



$\Rightarrow$ quasi-∞ time

  - **Infrared** (multi-dim. Boltzmann sampling): less than 5s!

- Now: 1000 samples with very low energies for **three** targets

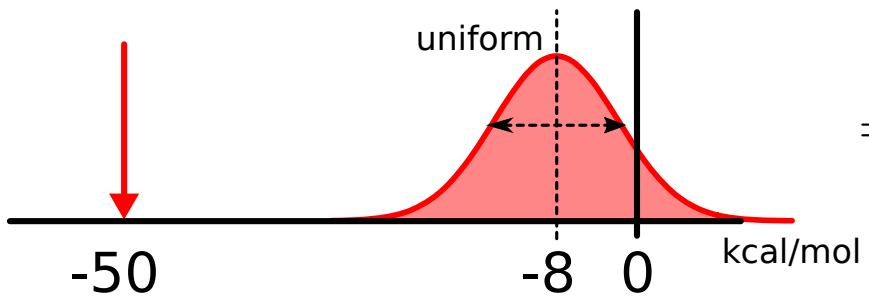


The challenge of positive design

Given is a secondary structure

$R =$ ((((((..((((((....))))((((((......))..))))..((((((....((((...))))).....))))))..)))..)))))))

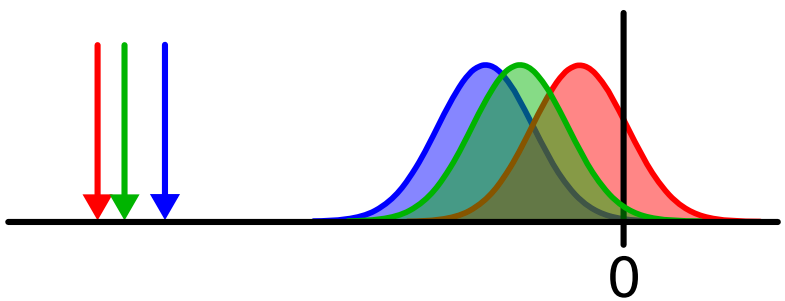
- Generate 1000 uniform random compatible sequences ✓
- Generate 1000 sequences where $E(R) \approx -50\text{kcal/mol}$
  - **Uniform sampling:**



$\Rightarrow$ quasi-∞ time

  - **Infrared** (multi-dim. Boltzmann sampling): less than 5s!

- Now: 1000 samples with very low energies for **three** targets



- **Uniform sampling:**
- **Infrared:** [a]

[a] involves serious CS-fu: CNs, TD, FPT DP,...

What is Infrared?

- Infrared is a **C++/Python-hybrid**[2] library for positive design
- generic **C++ engine** for efficient Boltzmann sampling
- **Python classes** to support
  the modeling of specific (positive design) problems
  e.g. RNA Design in `RNARedPrint v2`

[2] using Boost.Python

What is Infrared?

- Infrared is a **C++/Python-hybrid**[2] library for positive design
- generic **C++ engine** for efficient Boltzmann sampling
- **Python classes** to support
  the modeling of specific (positive design) problems
  e.g. RNA Design in `RNARedPrint v2`

Thus: new functionality and entire tools can be conveniently implemented in Python

[2] using Boost.Python

The world (of positive design) according to Infrared

General Task: Generate *things*$^{\text{TM}}$ with very specific properties

1) Define features, e.g.

- GC content
- #occurrences of the dinucleotide XY
- #occurrences of some k-mer (motif)
- energy of the ith target structure

2) Constrain features to **values:** *specific* GC%, energies, dinucl. freq's, forbid and enforce motifs, ...

3) Sample things$^{\text{TM}}$ that satisfy constraints **"Feature $\approx$ Value"**

Modelling complex constraints
"Feature $\approx$ Value"

Examples:

- **GC content** $\Rightarrow$ per position i, register one contribution

$$\texttt{GCContrib}(i, \pi_{GC}) = \begin{cases} 1 & \text{if } S_i \ in \{G, C\} \\ 0 & \text{otherwise} \end{cases}$$

Define feature: $\texttt{GCContent}(\pi_{GC}) = \sum_i \texttt{GCContrib}(i, \pi_{GC})$
Then constrain to specific value [filter; to be effective, learn π_{GC}]

Modelling complex constraints
"Feature ≈ Value"

Examples:

- **GC content** ⇒ per position i, register one contribution

$$\texttt{GCContrib}(i, \pi_{GC}) = \begin{cases} 1 & \text{if } S_i \ in \{G, C\} \\ 0 & \text{otherwise} \end{cases}$$

  Define feature: $\texttt{GCContent}(\pi_{GC}) = \sum_i \texttt{GCContrib}(i, \pi_{GC})$
  Then constrain to specific value [filter; to be effective, learn π_{GC}]

- **Base pair energy**
  ⇒ register contribution $\texttt{BPEnergy(i,j,}\pi_k\texttt{)}$
  per base pair (i, j) in structure k
  Then constrain [again: filter; learn π_k]

Modelling complex constraints
"Feature ≈ Value"

Examples:

- **GC content** ⇒ per position i, register one contribution

$$\texttt{GCContrib}(i, \pi_{GC}) = \begin{cases} 1 & \text{if } S_i \ in \{G, C\} \\ 0 & \text{otherwise} \end{cases}$$

  Define feature: $\texttt{GCContent}(\pi_{GC}) = \sum_i \texttt{GCContrib}(i, \pi_{GC})$
  Then constrain to specific value [filter; to be effective, learn π_{GC}]

- **Base pair energy**
  ⇒ register contribution $\texttt{BPEnergy(i,j,}\pi_k\texttt{)}$
  per base pair (i, j) in structure k
  Then constrain [again: filter; learn π_k]

Modelling complex constraints
"Feature ≈ Value"

Examples:

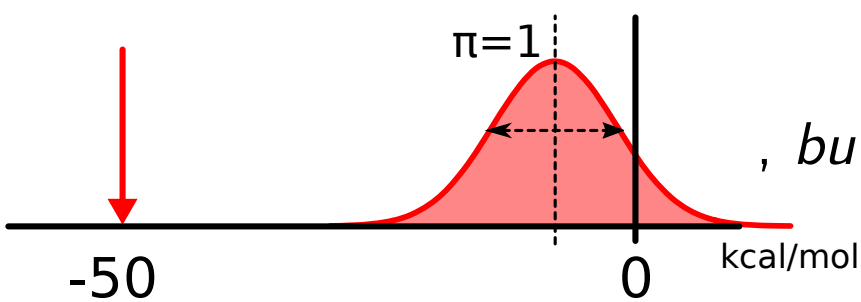
- **GC content** ⇒ per position i, register one contribution

$$\texttt{GCContrib}(i, \pi_{GC}) = \begin{cases} 1 & \text{if } S_i \textit{ in} \{G, C\} \\ 0 & \text{otherwise} \end{cases}$$

  Define feature: $\texttt{GCContent}(\pi_{GC}) = \sum_i \texttt{GCContrib}(i, \pi_{GC})$
  Then constrain to specific value [filter; to be effective, learn π_{GC}]

- **Base pair energy**
  ⇒ register contribution $\texttt{BPEnergy(i,j,}\pi_k\texttt{)}$
  per base pair (i, j) in structure k
  Then constrain [again: filter; learn π_k]

Idea:



, *but multi-dimensional*

Modelling complex constraints
"Feature ≈ Value"

Examples:

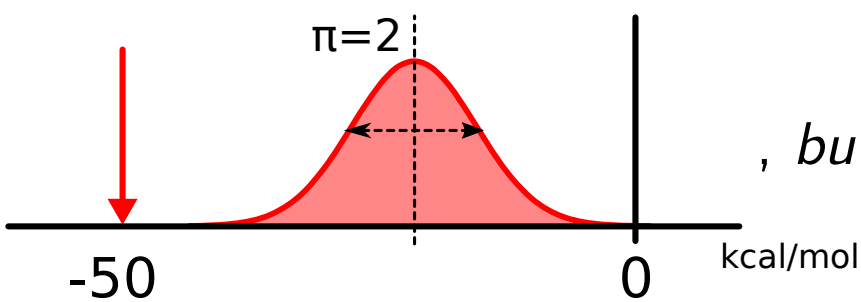
- **GC content** ⇒ per position i, register one contribution

$$\texttt{GCContrib}(i, \pi_{GC}) = \begin{cases} 1 & \text{if } S_i \text{ in} \{G, C\} \\ 0 & \text{otherwise} \end{cases}$$

  Define feature: $\texttt{GCContent}(\pi_{GC}) = \sum_i \texttt{GCContrib}(i, \pi_{GC})$
  Then constrain to specific value [filter; to be effective, learn π_{GC}]

- **Base pair energy**
  ⇒ register contribution $\texttt{BPEnergy(i,j,}\pi_k\texttt{)}$
  per base pair (i, j) in structure k
  Then constrain [again: filter; learn π_k]

Idea:



, *but multi-dimensional*

Modelling complex constraints
"Feature ≈ Value"

Examples:

- **GC content** ⇒ per position i, register one contribution

$$\texttt{GCContrib}(i, \pi_{GC}) = \begin{cases} 1 & \text{if } S_i \text{ in} \{G, C\} \\ 0 & \text{otherwise} \end{cases}$$

  Define feature: $\texttt{GCContent}(\pi_{GC}) = \sum_i \texttt{GCContrib}(i, \pi_{GC})$
  Then constrain to specific value [filter; to be effective, learn π_{GC}]

- **Base pair energy**
  ⇒ register contribution $\texttt{BPEnergy(i,j,}\pi_k\texttt{)}$
  per base pair (i, j) in structure k
  Then constrain [again: filter; learn π_k]

Idea:

, *but multi-dimensional*

Modelling complex constraints
"Feature ≈ Value"

Examples:

- **GC content** ⇒ per position i, register one contribution

$$\texttt{GCContrib}(i, \pi_{GC}) = \begin{cases} 1 & \text{if } S_i \text{ in} \{G, C\} \\ 0 & \text{otherwise} \end{cases}$$

  Define feature: $\texttt{GCContent}(\pi_{GC}) = \sum_i \texttt{GCContrib}(i, \pi_{GC})$
  Then constrain to specific value [filter; to be effective, learn π_{GC}]

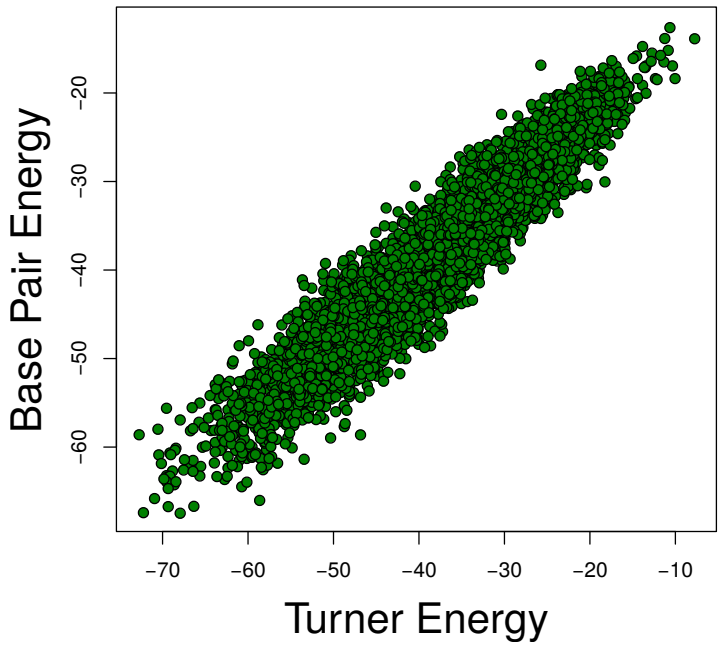
- **Base pair energy**
  ⇒ register contribution $\texttt{BPEnergy(i,j,}\pi_k\texttt{)}$
  per base pair (i, j) in structure k
  Then constrain [again: filter; learn π_k]

Idea:

, *but multi-dimensional*

Base pair energies approximate Turner energies

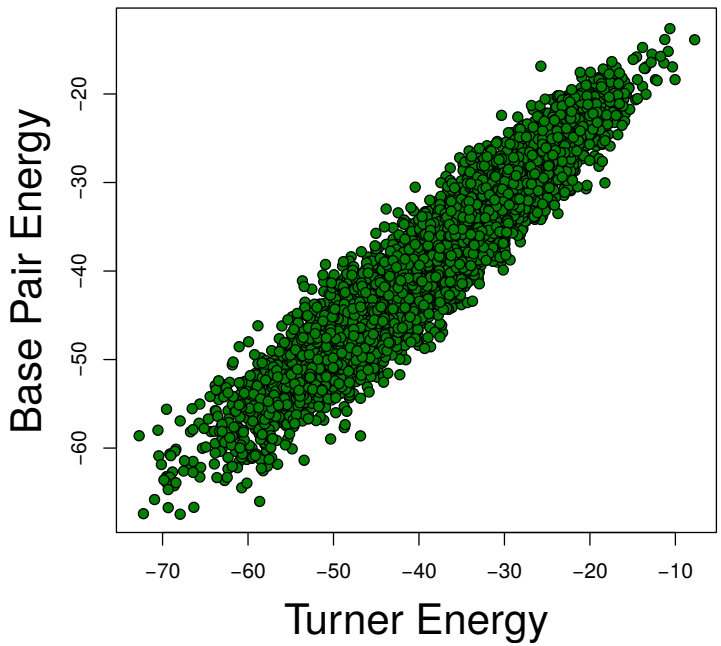
base pair energy model:	non-stacked			stacked		
	AU	CG	GU	AU	GC	GU
	1.27	-0.09	0.79	-0.52	-2.10	-0.88



$R^2 = 0.99$, BUT:
This approximation suffices for positive design, not prediction!

Base pair energies approximate Turner energies

base pair energy model:	non-stacked			stacked		
	AU	CG	GU	AU	GC	GU
	1.27	-0.09	0.79	-0.52	-2.10	-0.88

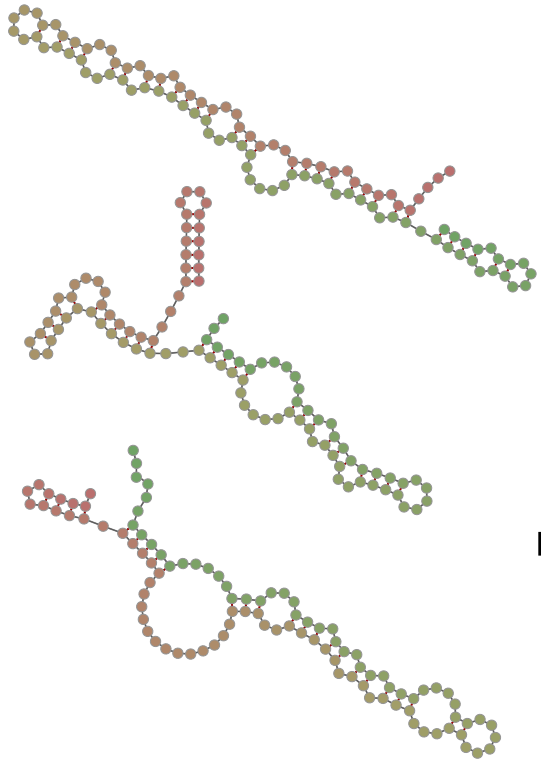


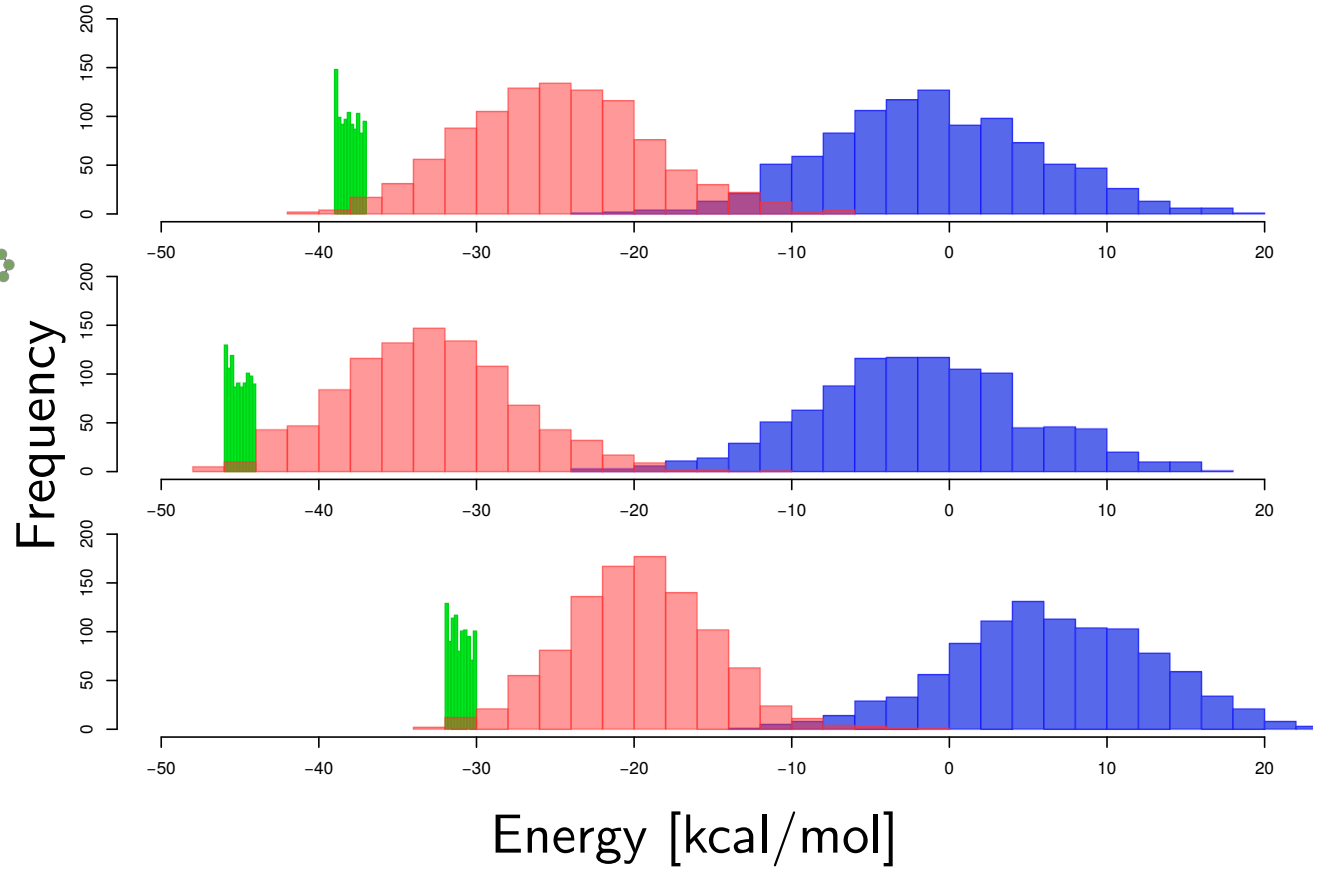
$R^2 = 0.99$, BUT:
This approximation suffices for positive design, not prediction!

How to **target Turner energies**:

- start with initial weight $\pi_k = 1$ of *base pair energy* (!) of kth structure
- generate samples and estimate mean *Turner energy* (!) of kth structure
- adapt weight π_k and iterate

Multi-target design to three RNA structures





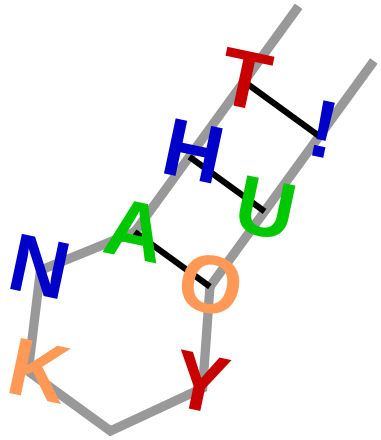
Uniform sample: 1000 sequences; generated in **seconds**

Boltzmann sample: 1000 good sequences; generated in **seconds**

Targeted sample: 1000 highly specific sequences; in **minutes**

https://github.com/s-will/Infrared

- framework based on **multi-dim. Boltzmann sampling**: effectively satisfies multiple (complex) constraints
- Promising application to **RNA design**: **`RNARedprint`**
- Generic modeling system to extend RNA design ...
  ... and develop novel sampling-based tools
- Supports construction of fancy **background models**: e.g. sample RNA alignments with fixed phylo-distances and energies of multiple structures
- Makes extensions easy (in Python) due to **C++/Python-hybrid programming**



Collaborators

Stefan Hammer

Yann Ponty

Team TBI (Ivo Hofacker) at universität wien

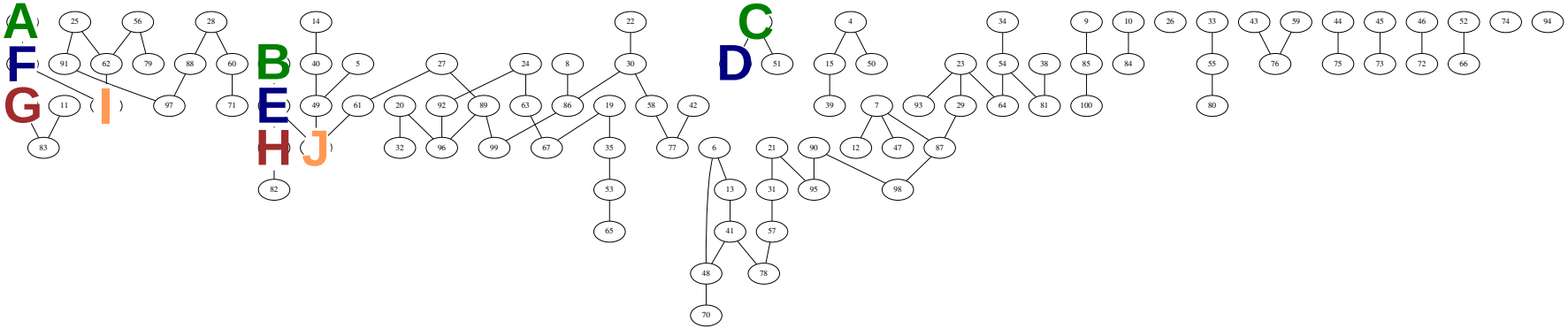
Funding ANR · Federal Ministry of Education and Research · FWF · Seventh Framework Programme

Read more RECOMB 2018 PARIS 'abstract' · `https://arxiv.org/abs/1804.00841`

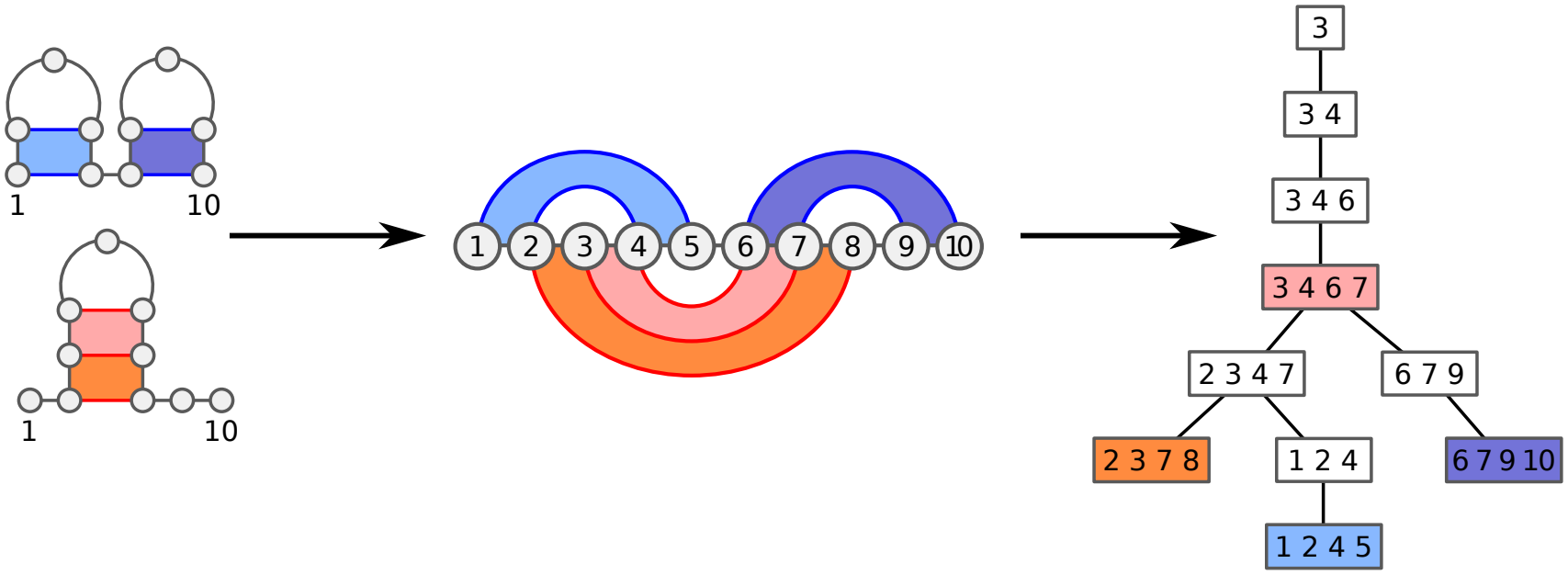
APPENDIX

Dependency graphs

```
(((.((....)).)))..((.(((.((((.....(((..((((((.((..(((.(.....).))).))).)).))))..)))..)))).))).....
..(((((.....(((.(((((((.....))))..))).)))).....)))))..((((((((((...))).).....))))))...((((((....))))))
......((((.....((...(((.((.((.(((....((......))...))).)).)))))..)))............)))))..(((...)))).
ABC             DEF                 GH                              IJ
```



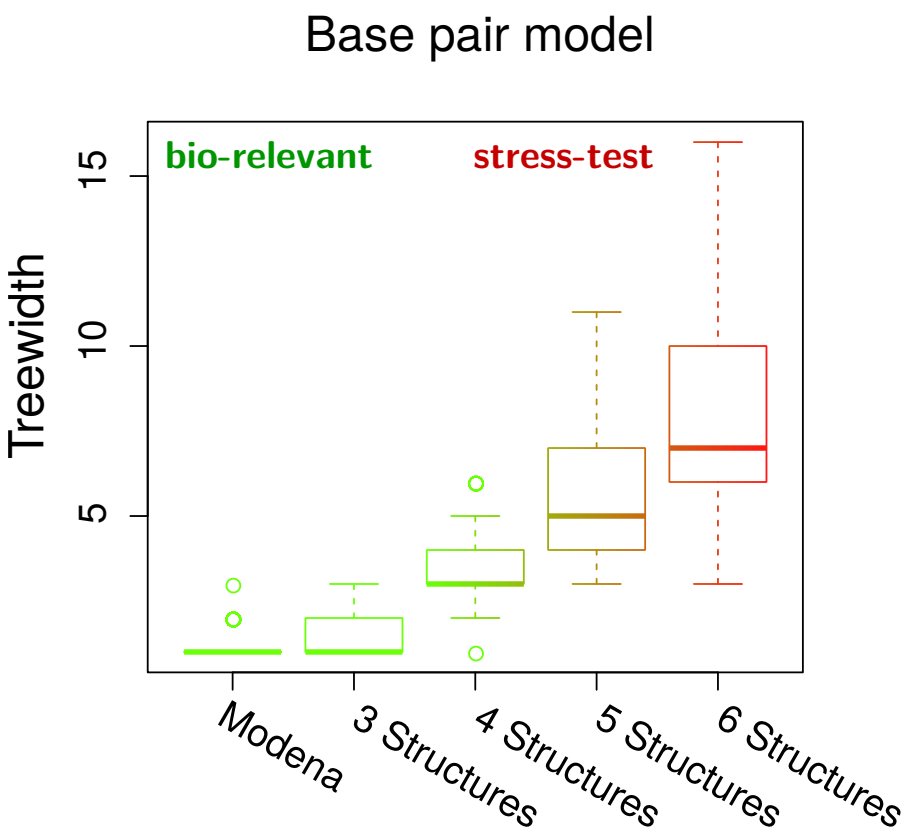
Tree decomposition

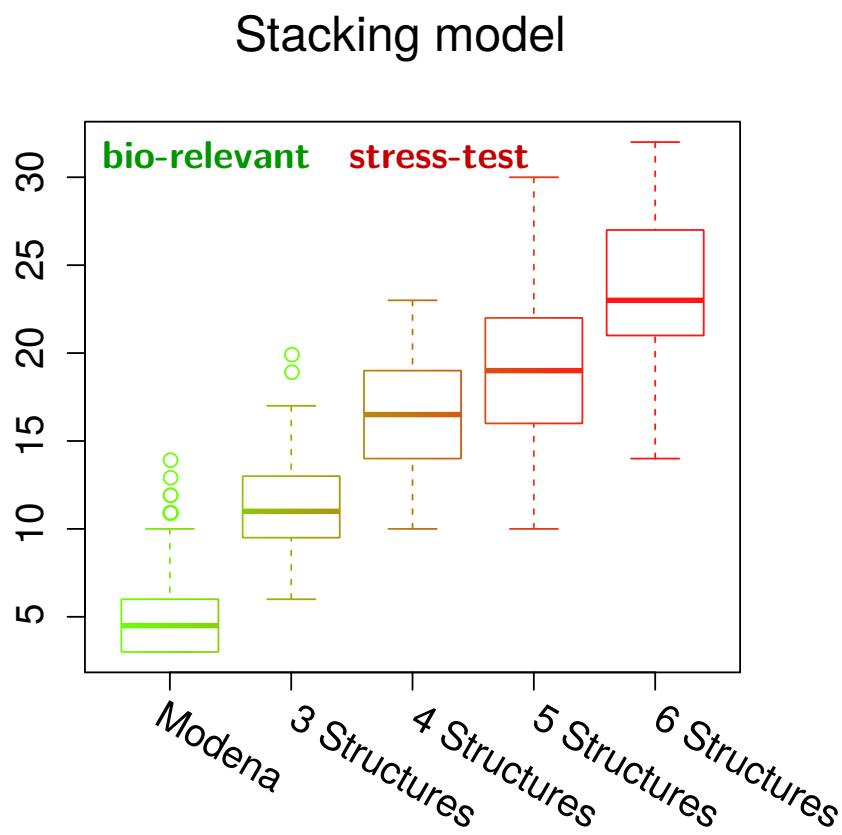


The tree decomposition . . .

- . . . is computed from the dependency graph
- . . . works as a template to guide our dynamic programming sampling algorithm
- . . . allows to consider all feature contributions in the sampling
- . . . gets more complex with increasing dependencies (complexity measure: treewidth $\hat{=}$ bag size)

Treewidths can be kept low





Why multi-dim. Boltzmann sampling?

Problem

IN: structures $\mathcal{R}$, length n, d features $F_1, \cdots, F_d$;
objective values $f_1^\star, \cdots, f_d^\star$; and tolerance $\varepsilon > 0$

OUT: t random sequences S, compatible w/ $\mathcal{R}$, s.t.

$$\forall 1 \leq \ell \leq d : F_\ell(S) \in [f_\ell^\star \cdot (1-\varepsilon), f_\ell^\star \cdot (1+\varepsilon)]$$

Possible approaches:

- **Multi-dim. Boltzmann sampling (+ rejection step)**

- **Classified Dynamic Programming**

Why multi-dim. Boltzmann sampling?

Problem

IN: structures $\mathcal{R}$, length n, d features $F_1, \cdots, F_d$;
objective values $f_1^\star, \cdots, f_d^\star$; and tolerance $\varepsilon > 0$

OUT: t random sequences S, compatible w/ $\mathcal{R}$, s.t.

$$\forall 1 \leq \ell \leq d : F_\ell(S) \in [f_\ell^\star \cdot (1-\varepsilon), f_\ell^\star \cdot (1+\varepsilon)]$$

Possible approaches:

- **Multi-dim. Boltzmann sampling (+ rejection step)**
  works well b/c distributions are typically concentrated
  - expect $\mathcal{O}(1)$ rejections for $\varepsilon > 1/\sqrt{n}$,
  - $\Theta(n^{d/2})$ for $\varepsilon = 0$ [Bender et al., 1983; Drmota, 1997].
- **Classified Dynamic Programming**

Why multi-dim. Boltzmann sampling?

Problem

IN: structures $\mathcal{R}$, length n, d features $F_1, \cdots, F_d$;
objective values $f_1^\star, \cdots, f_d^\star$; and tolerance $\varepsilon > 0$

OUT: t random sequences S, compatible w/ $\mathcal{R}$, s.t.

$$\forall 1 \leq \ell \leq d : F_\ell(S) \in [f_\ell^\star \cdot (1-\varepsilon), f_\ell^\star \cdot (1+\varepsilon)]$$

Possible approaches:

- **Multi-dim. Boltzmann sampling (+ rejection step)**
  works well b/c distributions are typically concentrated
  - expect $\mathcal{O}(1)$ rejections for $\varepsilon > 1/\sqrt{n}$,
  - $\Theta(n^{d/2})$ for $\varepsilon = 0$ [Bender et al., 1983; Drmota, 1997].
- **Classified Dynamic Programming**
  - convolution: $\times\Theta(n^{2d})$ time / $\Theta(n^d)$ space [Cupal et al., 1996]
  - using DFT to avoid convolution allows more efficient uniform sampling over range (case $\varepsilon > 0$) [cf. Senter et al., 2012]

Complex sequence constraints

Task: forbid a set $\mathcal{W}$ of subwords of length $\leq k$

Naïve: add k-ary constraints for each k successive sequence positions

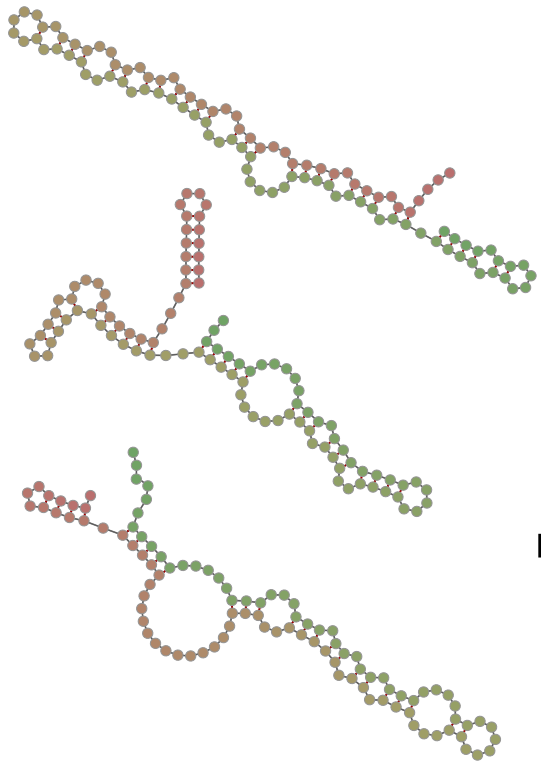
Proposed:

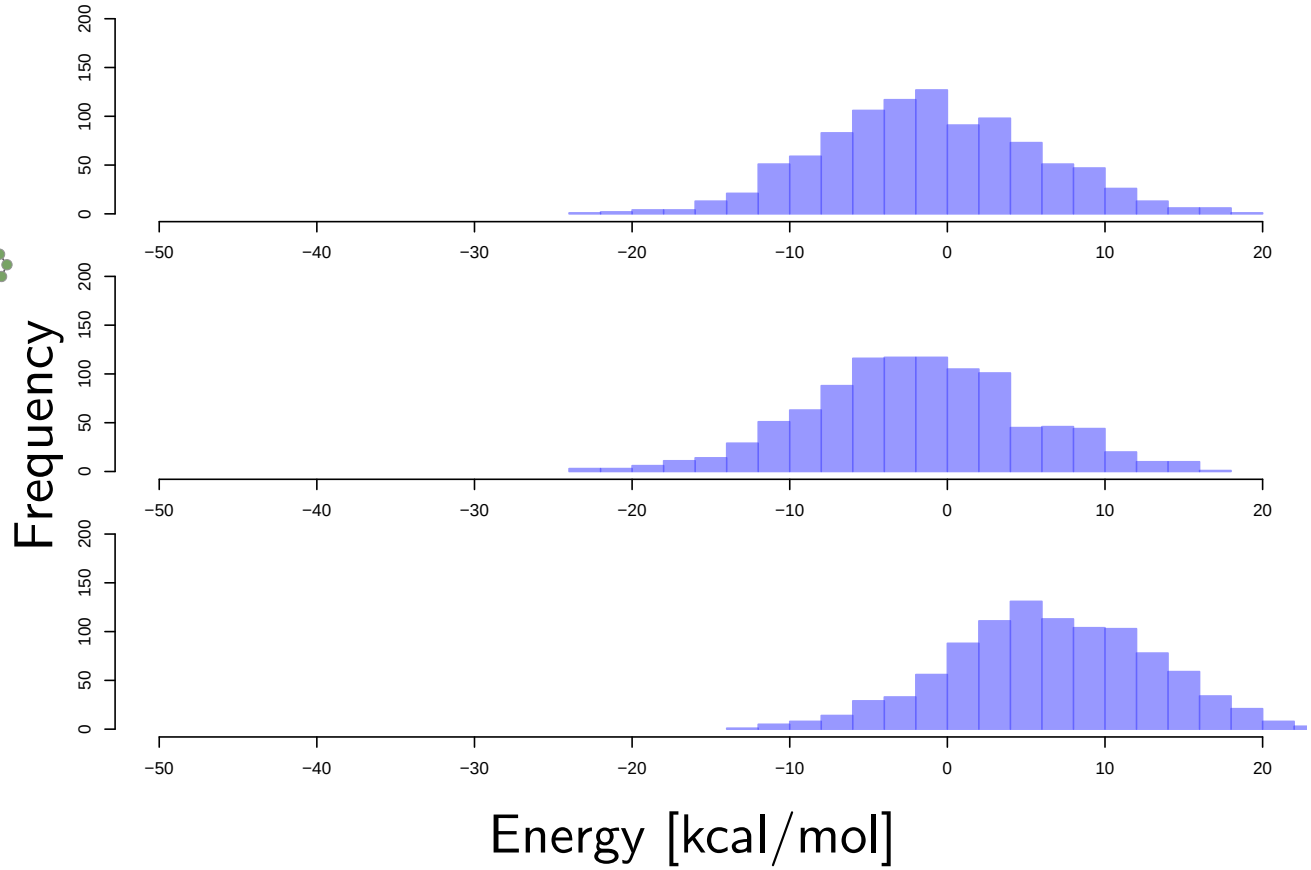
- construct Aho-Corasick automaton (states Q)
- extend alphabet from Σ to $Q \times \Sigma$
- restrict consecutive positions to transitions of the automaton (adds Hamiltonian path of binary constraints)
- new complexity $\mathcal{O}(n \cdot |\mathcal{R}| \cdot (|\Sigma| \cdot |Q|)^{w'+1})$; new tree width w' (!)

Similarly: *enforce* subwords

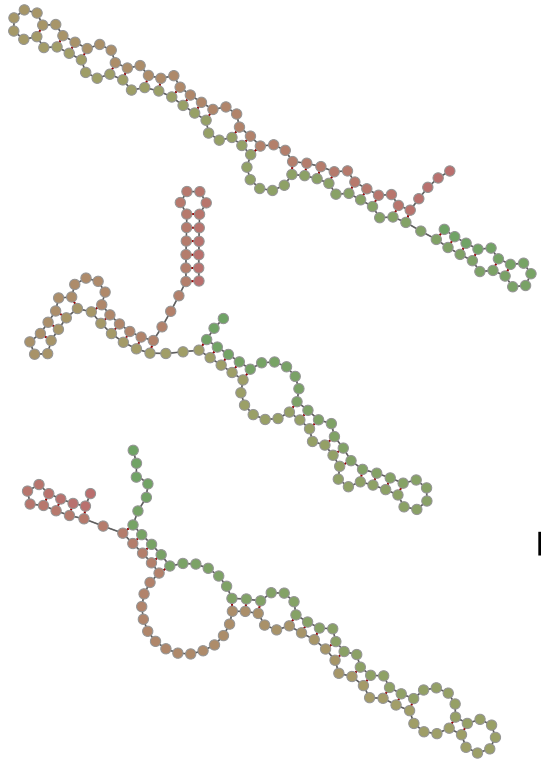
transfers ideas of [Zhou et al, 2013]

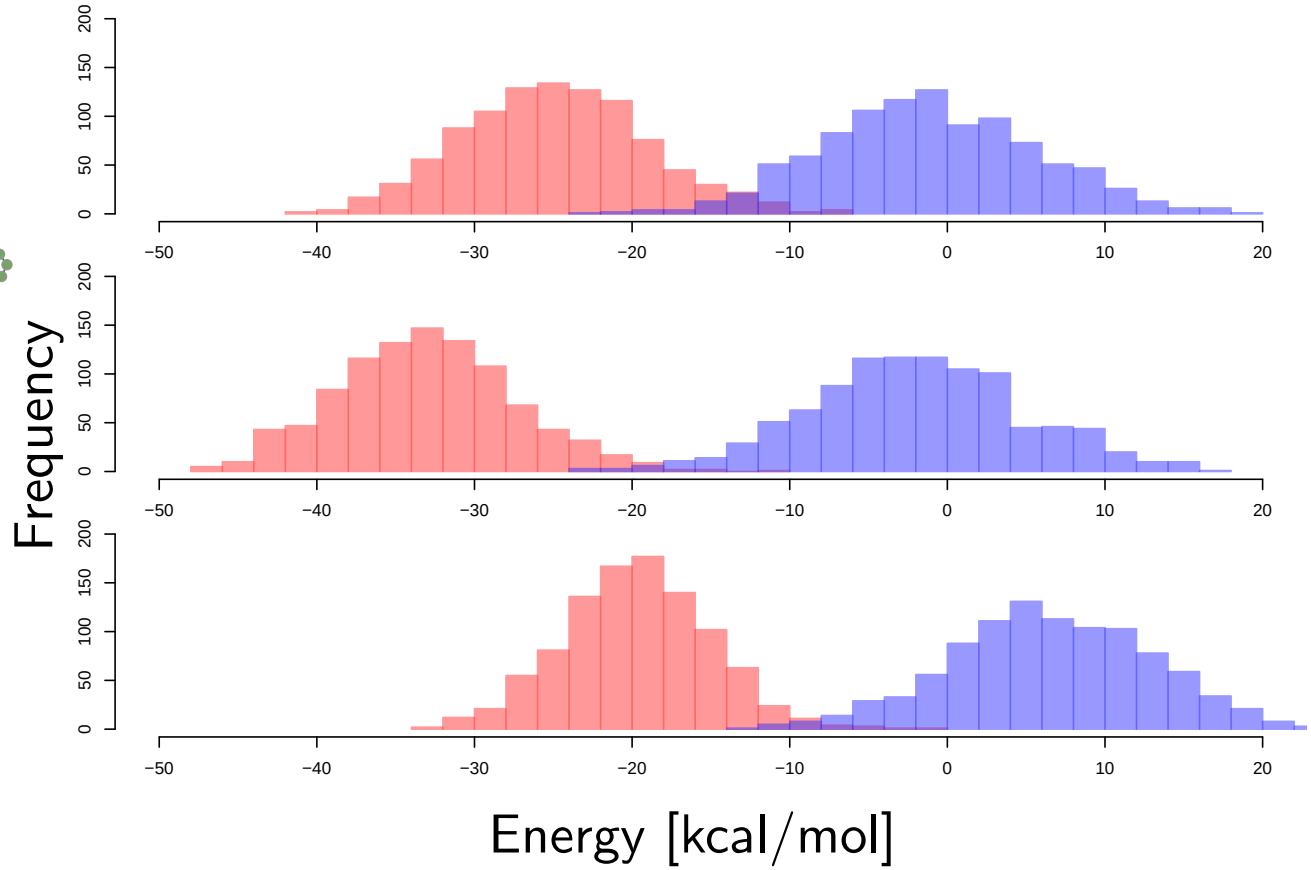
Multi-target design to three RNA structures





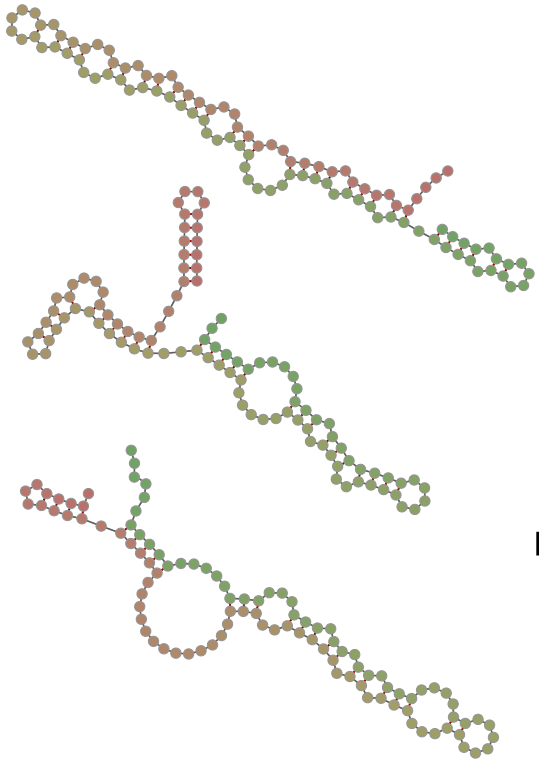
Multi-target design to three RNA structures

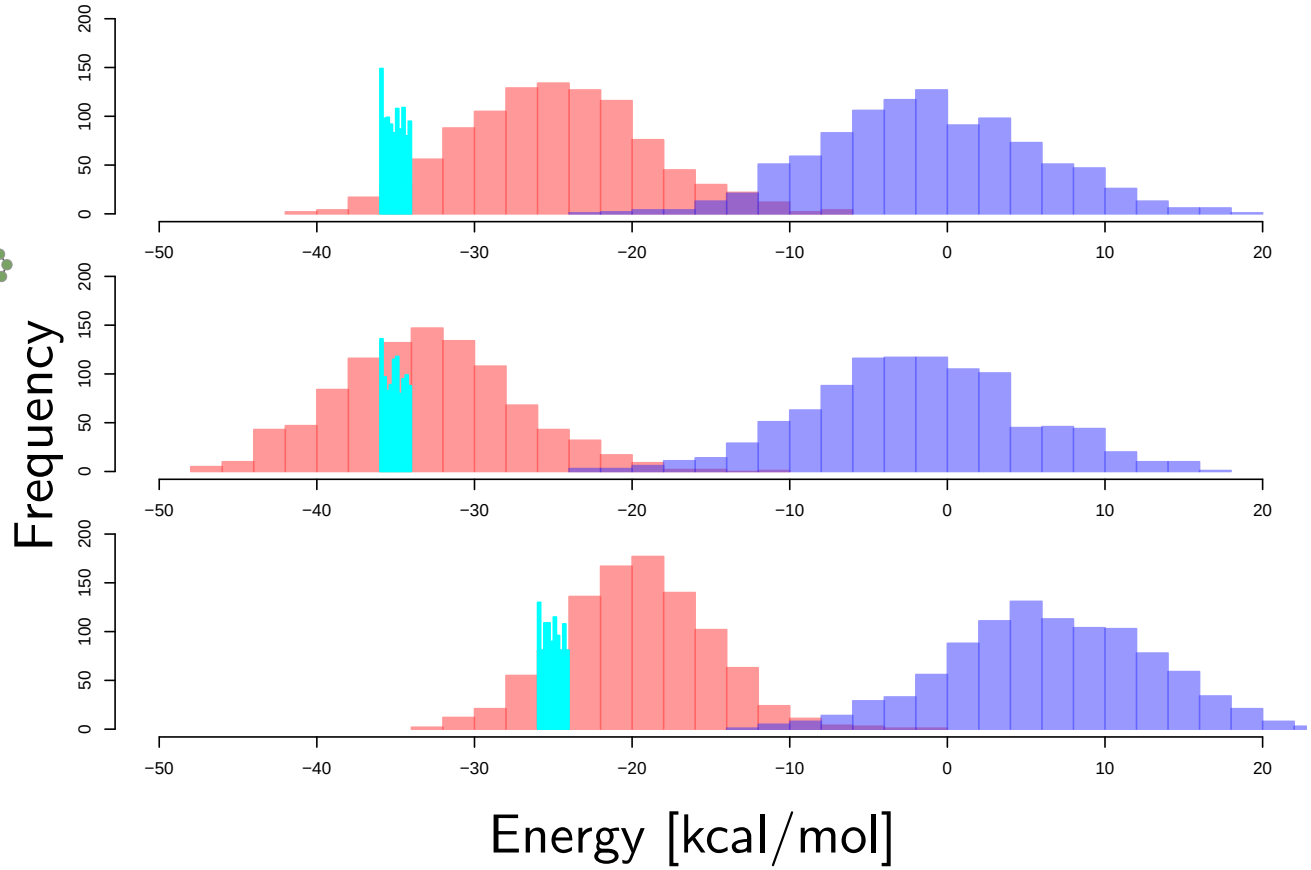




Boltzmann sample: 1000 low energy sequences; generated in seconds

Multi-target design to three RNA structures

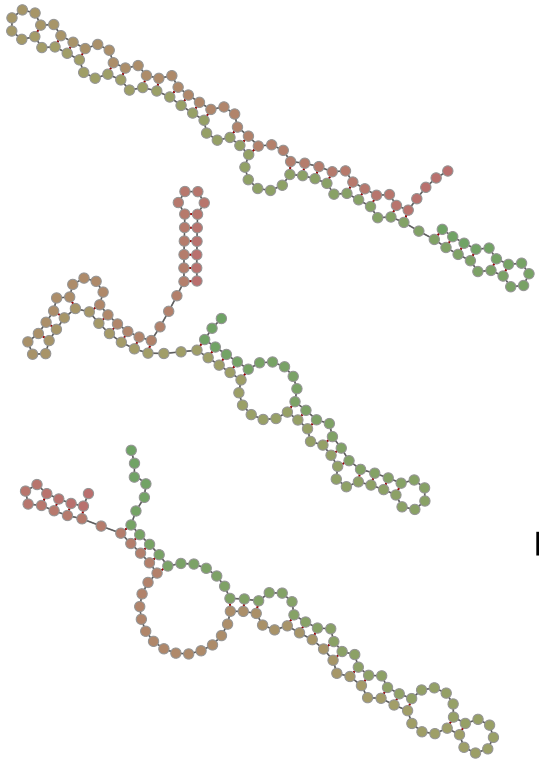


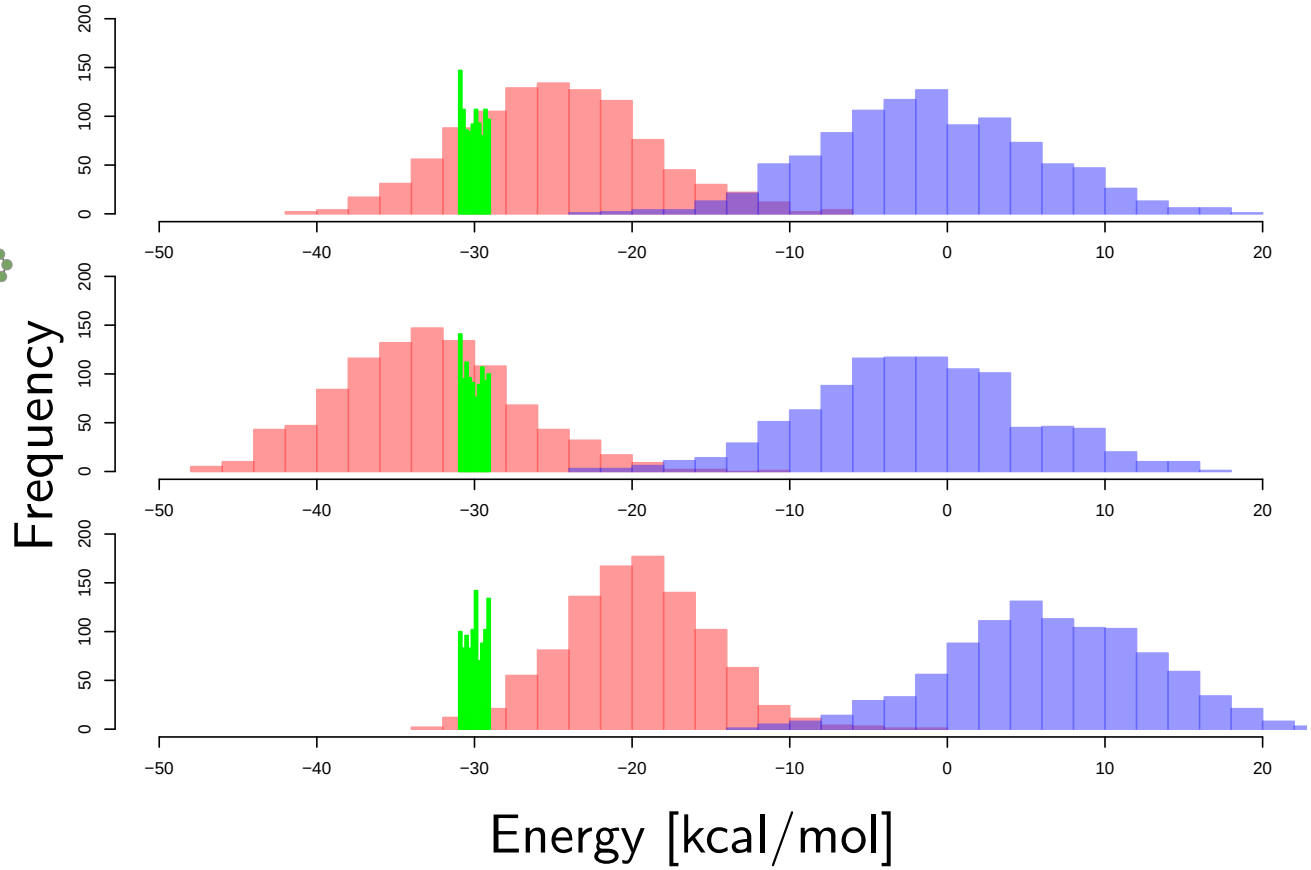


Boltzmann sample: 1000 low energy sequences; generated in seconds

Targeted samples: 1000 highly specific sequences; in minutes

Multi-target design to three RNA structures

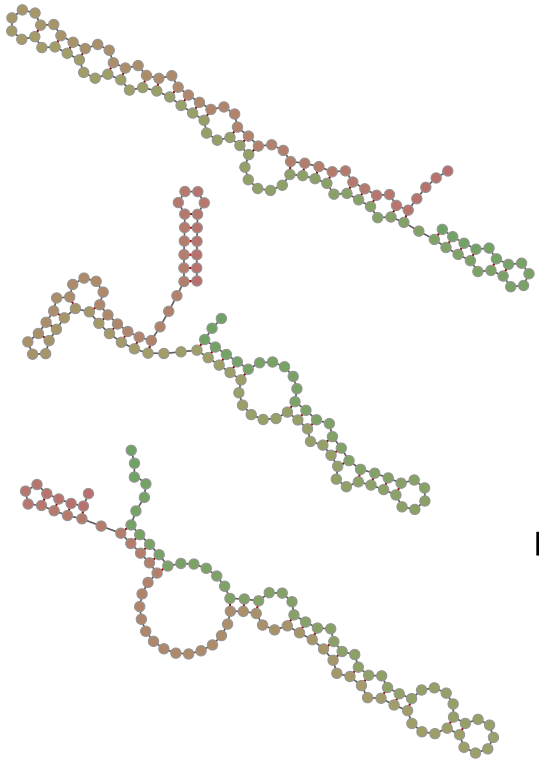


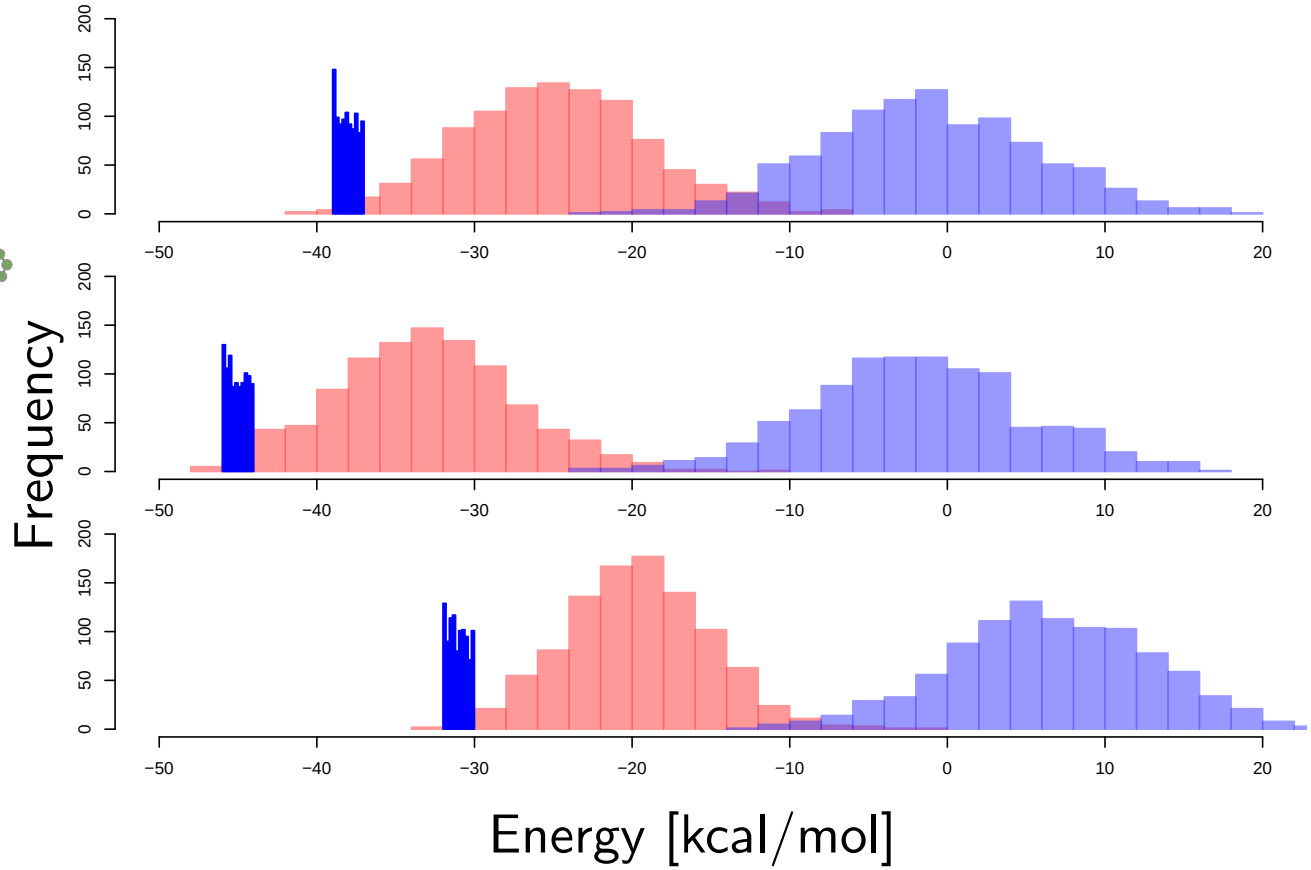


Boltzmann sample: 1000 low energy sequences; generated in seconds

Targeted samples: 1000 highly specific sequences; in minutes

Multi-target design to three RNA structures





Boltzmann sample: 1000 low energy sequences; generated in seconds
Targeted samples: 1000 highly specific sequences; in minutes