

NUPACK3.1

User Guide

Brian R. Wolfe, Nicholas J. Porubsky, Justin S. Bois, and Niles A. Pierce

California Institute of Technology

February 22, 2017

Contents

1	Overview	2
1.1	Terminology, notation, and physical model	2
1.2	Conventions	3
1.3	Versions	5
1.4	License	6
2	Complex Analysis	7
2.1	pfunc: calculate the partition function	7
2.2	pairs: calculate base-pairing observables	8
2.3	mfe: find the minimum free energy (MFE) secondary structure(s)	9
2.4	subopt: find all secondary structures within a specified free energy gap of the MFE	9
2.5	count: count the number of secondary structures in the ensemble	9
2.6	energy: calculate the free energy of a secondary structure	9
2.7	prob: calculate the equilibrium probability of a secondary structure	10
3	Complex Design	11
3.1	complexdefect: calculate the complex ensemble defect	11
3.2	complexdesign: design the equilibrium base-pairing properties of a complex	11
4	Test Tube Analysis	14
4.1	complexes: calculate the partition function and equilibrium base-pairing properties for each complex in a test tube	14
4.2	concentrations: calculate the equilibrium concentration for each complex in a test tube	16
4.3	distributions: calculate the equilibrium population distribution and expected value for a few complexes in a small box	17
5	Test Tube Design	20
5.1	tubedefect: calculate the test tube ensemble defect	20
5.2	tubedesign: design the equilibrium base-pairing properties of a test tube	22
6	Getting Started	25
6.1	Compilation and installation	25
6.2	Examples	26
7	Acknowledgments	27
8	References	27

1 Overview

NUPACK is a growing software suite for the analysis and design of nucleic acid molecules, devices, and systems. The package currently enables analysis or design of the equilibrium base-pairing properties of a test tube containing an arbitrary number of strand species interacting to form an arbitrary number of complex species. NUPACK algorithms are formulated in terms of nucleic acid secondary structure. In most cases, pseudoknots are excluded from the structural ensemble. Most of this software may be conveniently run using the NUPACK web application at nupack.org (Zadeh et al., 2011a).

Please direct questions, comments, features requests, and bug reports to support@nupack.org.

1.1 Terminology, notation, and physical model

The sequence, ϕ , of one or more interacting RNA strands is specified as a list of bases $\phi^a \in \{A, C, G, U\}$ for $a = 1, \dots, |\phi|$ (T replaces U for DNA). A *secondary structure*, s , of one or more interacting RNA strands is defined by a set of base pairs (each a Watson–Crick pair [A·U or C·G] or a wobble pair [G·U]). A *polymer graph* representation of a secondary structure is constructed by ordering the strands around a circle, drawing the backbones in succession from 5' to 3' around the circumference with a *nick* between each strand, and drawing straight lines connecting paired bases. A secondary structure is *unpseudoknotted* if there exists a strand ordering for which the polymer graph has no crossing lines. A secondary structure is *connected* if no subset of the strands is free of the others. A *complex* of L interacting strands with strand ordering, π , has *structural ensemble* containing all connected polymer graphs with no crossing lines (Dirks et al., 2007). (We dispense with our prior convention (Dirks et al., 2007; Zadeh et al., 2011a,b) of calling this entity an *ordered complex*.)

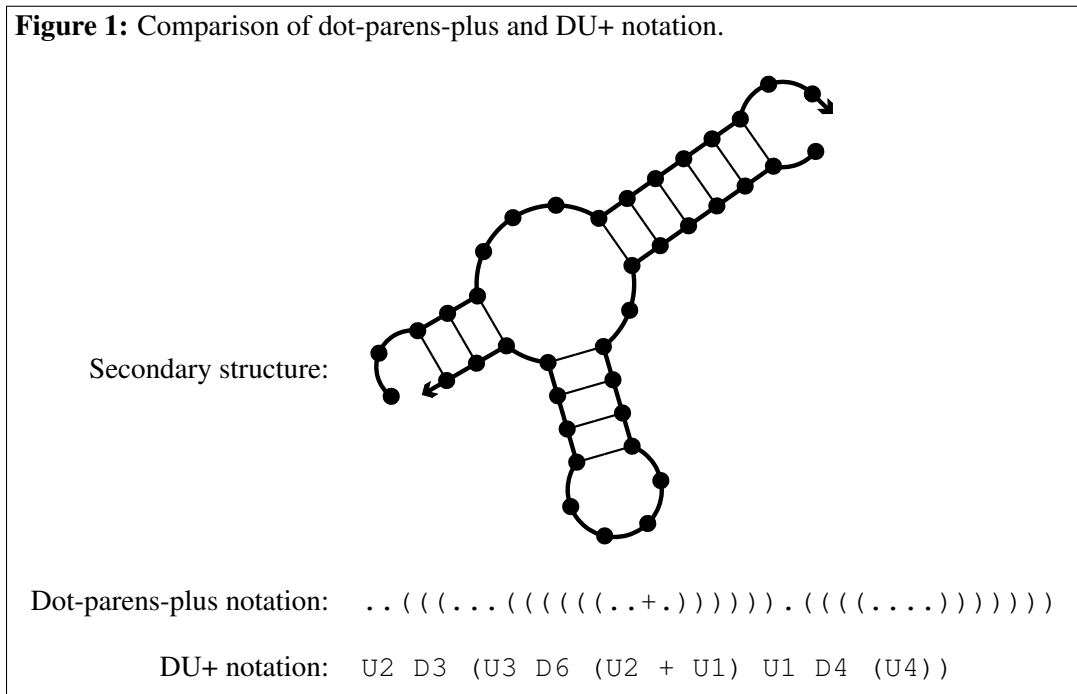
If a complex contains multiple strands with the same sequence, subtleties arise in the definition of the structural ensemble and in the calculation of experimental observables (Dirks et al., 2007). Let Γ denote the structural ensemble in which each strand is treated as distinct (i.e., each strand has a unique identifier in $\{1, \dots, L\}$) and let Γ' denote the ensemble in which strands with the same sequence are treated as indistinguishable. Two secondary structures are indistinguishable if their polymer graphs can be rotated so that all strands are mapped onto indistinguishable strands, all base pairs are mapped onto base pairs, and all unpaired bases are mapped onto unpaired bases; otherwise the structures are distinct (Dirks et al., 2007). The ensemble $\Gamma' \subseteq \Gamma$ is a maximal subset of distinct secondary structures for strand ordering π .

A *test tube* may contain an arbitrary number of strand species interacting to form an arbitrary number of complex species in a dilute solution. Let Ψ^0 denote the set of strand species that interact in a test tube to form the set of complex species Ψ . It is often convenient to define Ψ to contain all complexes of up to some size L_{\max} . Each complex $j \in \Psi$ corresponds to a distinct strand ordering π_j of L strands for $L \in \{1, \dots, L_{\max}\}$. L distinct strands can be ordered around a circle in $(L - 1)!$ distinct ways (e.g., strands A , B , and C can be ordered ABC and ACB). If some of the L strands are of the same species, there will be fewer than $(L - 1)!$ distinct strand orderings (e.g., strands A , A , and B can only be ordered AAB). For a given set of L strands, each unpseudoknotted connected secondary structure is found in the structural ensemble, Γ_j , corresponding to exactly one strand ordering, π_j (i.e., exactly one complex $j \in \Psi$) (Dirks et al., 2007).

For sequence ϕ and secondary structure, s , the *free energy*, $\Delta G(\phi, s)$, is calculated using nearest-neighbor empirical parameters for RNA (Serra and Turner, 1995; Mathews et al., 1999; Zuker, 2003) in 1M Na⁺ or for DNA (SantaLucia, 1998; Zuker, 2003) in user-specified concentrations of Na⁺ and Mg⁺⁺ (SantaLucia and Hicks, 2004; Koehler and Peyret, 2005). Additional parameters are employed for pseudoknotted secondary structures (Dirks and Pierce, 2003), which may be included in the structural ensemble only when analyzing a single RNA strand. The zero free energy reference state for all calculations is a system where all relevant strands are present with no base pairs (Dirks et al., 2007).

1.2 Conventions

- Sequences are listed 5' to 3'. The bases in a complex are indexed starting with 1 at the 5'-most base of the first strand and ending at the 3'-most base of the last strand. For example, if a complex has three strands of length 15, 20, and 13, respectively, the fifth base of the third strand has index 40.
- Valid bases are A, C, G, T, and U. For RNA calculations, T is automatically converted to U, and vice versa for DNA calculations.
- Secondary structures are specified in one of three ways:
 - *dot-parens-plus notation*: each unpaired base is represented by a dot, each base pair by matching parentheses, and each nick between strands by a plus (Zadeh et al., 2011a). For example, $((...))$ specifies that bases 1 and 2 are paired to bases 7 and 6, respectively, while bases 3, 4, and 5 are unpaired. $((+. . .))$ specifies that bases 1 and 2 of strand 1 are paired to bases 5 and 4 of strand 2. Four types of “parentheses” are accepted: $()$, $[\]$, $\{\}$, and $\langle \rangle$. Within a specified structure, each type of parentheses must satisfy a nesting property but different types need not be nested, allowing specification of pseudoknotted structures (though highly nested pseudoknots may not be specifiable with only four types of parentheses).
 - *pair list notation*: each line consists of two whitespace-separated integers $[i\ j]$, $i < j$, specifying that base i is paired to base j . Any secondary structure, including highly-nested pseudoknots, may be specified in this way.
 - *DU+ notation*: Using DU+ notation, a duplex of length \underline{x} base pairs is represented by $D\underline{x}$ and an unpaired region of length \underline{x} nucleotides is represented by $U\underline{x}$ (Zadeh, 2010). Each duplex is followed immediately by the substructure (specified in DU+ notation) that is ‘enclosed’ by the duplex. If this substructure includes more than one element, parentheses are used to denote scope. A nick between strands is specified by a '+'. See Figure 1 and Table 1 for examples.



The following option flags are recognized by multiple NUPACK executables:

`-material parameters`

The parameter files defining the nucleic acid material are specified via the argument `parameters` which

Table 1: Examples of dot-parens-plus and DU+ notation.

Dot-parens-plus notation	DU+ notation
(((((.....))))))	D12 U10
(((((.....(+)))))).....	D12 + U10
(((((.....(+.....))))))	D12 (+ U10)

represents either a filename prefix or a shorthand identifier for an included parameter set. If the filename does not contain a relative or absolute path, then the program will look for the files first in the current directory, and then in the directory `$NUPACKHOME/parameters`. Available filename prefixes currently include:

- `rna1995` (default; shorthand: `rna`)
Parameter files `*.dG` and `*.dH` for RNA allowing calculations at different temperatures (Serra and Turner, 1995; Zuker, 2003); includes pseudoknot parameters from (Dirks and Pierce, 2003).
- `dna1998` (shorthand: `dna`)
Parameter files `*.dG` and `*.dH` for DNA allowing calculations at different temperatures (SantaLucia, 1998; Zuker, 2003); there are no pseudoknot parameters.
- `rna1999`
Parameter file `*.dG` for RNA for calculations at 37 °C (Mathews et al., 1999; Zuker, 2003); includes pseudoknot parameters from (Dirks and Pierce, 2003).

DNA/RNA hybrids are not allowed.

`-sodium concentration`

The Na^+ concentration of the solution in units of molar (default: 1.0, range: [0.05,1.1]) is specified by `concentration` (SantaLucia and Hicks, 2004). This flag is only valid when the `-material dna` is also selected because no RNA salt correction parameters are available.

`-magnesium concentration`

The Mg^{++} concentration of the solution in units of molar (default: 0.0, range: [0.0,0.2]) is specified by `concentration` (Koehler and Peyret, 2005). This flag is only valid when the `-material dna` is also selected.

`-dangles treatment`

The way in which dangle energies are incorporated is specified by `treatment`, which may have the following values:

- `none`: No dangle energies are incorporated.
- `some`: (default) A dangle energy is incorporated for each unpaired base flanking a duplex (a base flanking two duplexes contributes only the minimum of the two possible dangle energies).
- `all`: A dangle energy is incorporated for each base flanking a duplex regardless of whether it is paired.

`-T temperature`

Temperature specified in °C (default: 37).

`-multi`

Specify a calculation involving complexes of multiple interacting strands.

`-pseudo`

Augment the structural ensemble Γ with a class of pseudoknots (Dirks and Pierce, 2003, 2004). This option is only available for single-stranded RNA calculations. An error message is returned if `-pseudo` is specified in combination with either `-multi` or `-material dna`.

1.3 Versions

- NUPACK 3.0:
 - Features:
 - * complex analysis
 - * complex design
 - * test tube analysis
 - Executables:
 - * `pfunc`, `pairs`, `mfe`, `subopt`, `count`, `energy`, `prob`, `pairs`, `defect`, `complexes`, `concentrations`, `distributions`, `design`
 - * These executables read input files containing comment lines preceded by `%`; blank lines are not permitted.
 - Terminology and notation:
 - * details in (Dirks et al., 2007)
- NUPACK 3.1:
 - New features:
 - * test tube design
 - New executables:
 - * `tubedefect` and `tubedesign`
 - * These executables read `*.np` script files written in a preliminary version of the NUPACK scripting language (see “Future Version: Script files” below).
 - * In `*.np` script files, a comment begins with `#` and continues for the rest of the line; blank lines are permitted.
 - Changes to existing executables:
 - * Name of executable `defect` changed to `complexdefect`.
 - * Name of executable `design` changed to `complexdesign`.
 - * Updates to the default options and output file formats for executables `complexes`, `concentrations`, and `distributions`. Use option `-v3.0` to revert to NUPACK 3.0 behavior using NUPACK 3.1.
 - Terminology and notation:
 - * details in Section 1.1
- Future Version:
 - Script files:
 - * In a future major release, all NUPACK jobs will be specified using a NUPACK scripting language in `*.np` script files. This approach will be more flexible and robust than the current approach of using a tailored input file format for each NUPACK executable.
 - Output files:
 - * In a future major release, the format of NUPACK output files will be changed so that they can be loaded using a generic parser. This approach will be more convenient and robust than the current approach of using a tailored output file format for each NUPACK executable.

1.4 License

NUPACK Software License Agreement

Copyright © 2015. California Institute of Technology. All rights reserved.

Use and redistribution in source form and/or binary form, with or without modification, are permitted for non-commercial academic purposes only, provided that the following conditions are met:

1. Redistributions in source form must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation provided with the distribution.
3. Web applications that use the software in source form or binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in online documentation provided with the web application.
4. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote derivative works without specific prior written permission.

Disclaimer

This software is provided by the copyright holders and contributors “as is” and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the copyright holder or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

2 Complex Analysis

Analyze the equilibrium base-pairing properties of a complex of interacting nucleic acid strands.

2.1 pfunc: calculate the partition function

Command: pfunc [-T temperature] [-multi] [-pseudo] [-material parameters]
[-dangles treatment] prefix

Description: Computes the partition function, $Q(\phi)$, over the ensemble Γ' .

Input: Input is read from the file prefix.in, where prefix is a command line argument. Comment lines are preceded by % and may be interspersed with data in input files; blank lines are not permitted. For single-stranded calculations, the input file contains the strand sequence specified on a single line. If `-multi` is specified, the input file must contain the following entries on separate lines:

- The number of distinct strand species, $|\Psi^0|$.
- The sequence for each distinct strand species (each on a separate line). Note that strand species defined on different lines are treated as distinct even if they have the same sequence.
- L integers from the range 1 to $|\Psi^0|$ representing the strand ordering π of the L strands in the complex.

Output:

Following header comments, the free energy of the complex ($\Delta G \equiv -kT \log Q$) and the partition function are written to the screen.

Example 1:

Calculate the partition function for a single RNA strand at 37 °C including a class of pseudoknots.

Input file contents:

```
GGGUCGUUUUUCUCGCGUGACUUUCAGCCCCAAACAAAAAUGUCAGCA
```

Command: pfunc -pseudo

```
$NUPACKHOME/doc/examples/complex-analysis/pseudoknot/input/telomerase
```

Example 2:

Calculate the partition function for a complex of four DNA strands at 23 °C, two of which are indistinguishable.

Input file contents:

```
3
AGTCTAGGATTCGGCGTGGGTAA
TTAACCCACGCCGAATCCTAGACTCAAAGTAGTCTAGGATTCGGCGTG
AGTCTAGGATTCGGCGTGGGTAAACACGCCGAATCCTAGACTACTTTG
1 2 2 3
```

Command: pfunc -T 23 -multi -material dna

```
$NUPACKHOME/doc/examples/complex-analysis/advanced/input/hcr
```

2.2 pairs: calculate base-pairing observables

Command: `pairs [-T temperature] [-multi] [-pseudo] [-material parameters]
[-dangles treatment] [-cutoff cutoffvalue] prefix`

Description: Computes *pair probabilities* $p(i_n \cdot j_m)$ for the complex corresponding to the specified strand ordering π . When `-multi` is selected, also computes the *expected number of base pairs* $E(i_{\{A\}} \cdot j_{\{B\}})$.

Additional option:

`-cutoff cutoffvalue`

Only probabilities and expected values at or above cutoffvalue (default: 0.001) are saved in the output file(s).

Input: Same format as for the executable `pfunc`.

Output: The output is written to the files:

- `prefix.ppairs`

Contains the probability of each type of base pair in the complex. The relevant quantities are $p(i_n \cdot j_m)$, the probability that base i of strand n is paired to base j of strand m in the complex corresponding to strand ordering π . All strands in the complex are considered to be distinct. For example, the two strands labeled 2 in Example 2 are considered distinct. One might think of them as strand $2a$ and $2b$, and a given base of strand $2a$ may have different pair probabilities than the corresponding one in strand $2b$. The total number of bases in the complex is $N = \sum_{l=1}^L N_l$, so indexing bases from 1 to N , the pair probabilities can be stored in a symmetric $N \times N$ matrix. Augmentation by an $N + 1$ st column containing the probability that each base is unpaired causes the rows to sum to unity.

By default, the file is formatted as follows. Following header comments, the first entry is the integer N . The remaining entries come in triplets of the form $[i \ j \ p]$, where $1 \leq i \leq N$ and $1 \leq j \leq N + 1$ are base numbers and p is the probability of the corresponding pair. Values corresponding to $j = N + 1$ represent the probability that base i is unpaired. If `-pseudo` is selected, each row is augmented by two additional columns. The first is the probability that bases i and j form a nested pair and the second is the probability that bases i and j form a non-nested pair. In the case of $j = N + 1$, these additional columns store the probability that bases i and j do not form a nested pair and the probability that they do not form a non-nested pair, respectively.

- `prefix.epairs`

Generated when `-multi` is selected. Similar to `prefix.ppairs` except strands of the same species are considered to be indistinguishable. The relevant quantities are $E(i_{\{A\}} \cdot j_{\{B\}})$, the expected number of base i of strand species A that are paired to base j of strand species B in the complex corresponding to strand ordering π . The number of distinct bases in the complex is $N_{\text{distinct}} \equiv \sum_{k \in \Psi^0} N_k$, representing the total number of bases in all $|\Psi^0|$ strand species. Numbering the distinct bases from 1 to N_{distinct} , the distinct base pairs may be represented as a symmetric $N_{\text{distinct}} \times N_{\text{distinct}}$ matrix; by augmenting the matrix with an extra column that contains the expected number of base i of strand species A that are unpaired, each row sums to the number of base i of strand species A in the complex. Note that this numbering system is used even if some sequences listed in the input file are absent from the specified complex.

The file is formatted as follows. Following header comments, the first entry is the integer N_{distinct} , and the remaining entries come in triplets of the form $[i \ j \ E]$, analogously to the `.ppairs` file, except E is the expected number of the corresponding pair. Information is stored only for bases included in the specified complex.

2.3 mfe: find the minimum free energy (MFE) secondary structure(s)

Command: mfe [-T temperature] [-multi] [-pseudo] [-material parameters]
[-dangles treatment] [-degenerate] prefix

Description: Determine the minimum free energy secondary structure(s), $s^{\text{MFE}}(\phi)$, of sequence ϕ over the ensemble of the complex, Γ . If the `-degenerate` flag is selected, all secondary structures with the minimum free energy are determined; otherwise only one MFE structure is returned.

Input: Same format as for the executable `pfunc`.

Output: Output is written to the file `prefix.mfe`. After header comments, each entry describes one of the possibly many degenerate MFE structures. The entries are separated by comment lines (repeated % signs). The first line in each entry is the number of bases in the ordered complex. The second line is the minimum free energy. The third line is the dot-parens-plus representation of the MFE structure. Subsequent lines contain the MFE structure in pair list notation.

2.4 subopt: find all secondary structures within a specified free energy gap of the MFE

Command: subopt [-T temperature] [-multi] [-pseudo] [-material parameters]
[-dangles treatment] prefix

Description: Similar to `mfe` except that all secondary structures in Γ with free energies within the specified (non-negative) free energy gap of the MFE are calculated and stored. This can be very slow and the output very large if the specified gap is too large. The output is sorted by increasing free energy.

Input: Same format as for the executable `pfunc`, plus one additional row containing the energy gap.

Output: Output is written to the file `prefix.subopt` with the same format as for the executable `mfe`.

2.5 count: count the number of secondary structures in the ensemble

Command: count [-multi] [-pseudo] prefix

Description: Calculates the number of secondary structures, $|\Gamma|$, in the ensemble of the complex, treating all strands as distinct.

Input: Same format as for the executable `pfunc`.

Output: The number of secondary structures is written to the screen, preceded by header comments.

2.6 energy: calculate the free energy of a secondary structure

Command: energy [-T temperature] [-pseudo] [-multi] [-material parameters]
[-dangles treatment] prefix

Description: Calculate the free energy, $\Delta G(\phi, s)$, of sequence ϕ in secondary structure s .

Input: Same format as for executable `pfunc`, plus one additional row specifying the secondary structure in dot-parens-plus notation. Alternatively, the structure may be represented in pair list notation.

Output: The free energy is written to the screen, preceded by header comments.

2.7 **prob**: calculate the equilibrium probability of a secondary structure

Command: `prob [-T temperature] [-pseudo] [-multi] [-material parameters]
[-dangles treatment] prefix`

Description: Calculates the equilibrium probability, $p(\phi, s)$, that sequence ϕ adopts secondary structure s within the ensemble of the complex, Γ' .

Input: Same format as for the executable `energy`.

Output: The probability is written to the screen, preceded by header comments.

3 Complex Design

Design the equilibrium base-pairing properties of a complex of interacting nucleic acid strands.

3.1 `complexdefect`: calculate the complex ensemble defect

Command: `complexdefect [-T temperature] [-pseudo] [-multi] [-material parameters] [-dangles treatment] [-mfe] prefix`

Description: Calculate the complex ensemble defect, $n(\phi, s)$, representing the average number of incorrectly paired nucleotides at equilibrium evaluated over the ensemble of the complex, Γ (Dirks et al., 2004; Zadeh et al., 2011b). Here, ϕ is the sequence and s is the target secondary structure.

Additional option:

`-mfe`

Instead, calculate the complex MFE defect, $\mu(\phi, s)$, representing the number of incorrectly paired nucleotides in the MFE structure s^{MFE} (Zadeh et al., 2011b).

Input: Same format as for the executable energy.

Output: Following header comments, the complex ensemble defect, $n(\phi, s)$, and the normalized complex ensemble defect, $n(\phi, s)/|\phi|$, are written to the screen. If the `-mfe` flag is selected, the complex MFE defect, $\mu(\phi, s)$, and the normalized complex MFE defect, $\mu(\phi, s)/|\phi|$, are written to the screen.

3.2 `complexdesign`: design the equilibrium base-pairing properties of a complex

Command: `complexdesign [-init initmode] [-loadinit] [-outputinit] [-loadseed] [-outputseed] [-fstop fstopvalue] [-prevent file] [-mleafopt mleafoptvalue] [-mreopt mreoptvalue] [-pairs] [-cutoff cutoffvalue] prefix`

Description: Perform sequence design over the ensemble of a complex of interacting nucleic acid strands. The user specifies a target secondary structure. Sequence design is formulated as an optimization problem with the goal of reducing the complex ensemble defect below a user-specified stop condition (Zadeh et al., 2011b).

Additional options:

`-init initmode`

The argument `initmode` selects the sequence initialization method from one of the following (using a sequence that satisfies the base-pairing requirements of the target secondary structure with Watson-Crick pairs):

AU: Initial sequences are randomly selected from A and T/U bases only.

CG: Initial sequences are randomly selected C and G bases only.

RND: (default) Initial sequences are randomly selected from A, C, G, T/U.

SSM: Initial sequences are generated using sequence symmetry minimization (Seeman, 1982; Dirks et al., 2004).

`-loadinit`

Initialize the sequence from the file `prefix.init`.

`-outputinit`

Output the initial sequence to `prefix.init`.

-loadseed

Initialize the random number generator with the seed specified in `prefix.seed`. This can be used to duplicate design execution.

-outputseed

Output the random number generator's seed to `prefix.seed`.

-fstop `fstopvalue`

Set the stop condition for the design algorithm to `fstopvalue` (default: 0.01). The design algorithm seeks to achieve $n(\phi, s) \leq \text{fstopvalue} |\phi|$.

-prevent `preventfile`

The file `preventfile` contains patterns to be prevented from appearing in the sequence design.

-mleafopt `mleafoptvalue`

Leaf optimization is restarted from new initial conditions up to `leafoptvalue` times (default: 3) before terminating unsuccessfully (Zadeh et al., 2011b).

-mreopt `mreoptvalue`

The elimination of emergent defects in a parent node by defect-weighted child sampling and reoptimization is attempted up to `mreoptvalue` times (default: 10) (Zadeh et al., 2011b).

-pairs

Save the pair probabilities in a `.ppairs` file.

-cutoff `cutoffvalue`

Only probabilities at or above `cutoffvalue` are saved in the `.ppairs` file (default: 0.001).

Input: The target structure and sequence constraints are read from `prefix.fold`. The first line of the file is the target structure in dot-parens-plus notation. The second line of the file contains the sequence constraints (if any) specified using IUPAC nucleic acid codes (Table 2). If no sequence constraints are specified for a given base, it is assumed to be unconstrained. Comment lines are preceded by % and may be interspersed with data in input files; blank lines are not permitted.

Table 2: IUPAC Degenerate Nucleotide Codes for RNA*

Code	Nucleotides
M	A or C
R	A or G
W	A or U
S	C or G
Y	C or U
K	G or U
V	A, C, or G
H	A, C, or U
D	A, G, or U
B	C, G, or U
N	A, C, G, or U

*T replaces U for DNA

Optional inputs are specified in the following files:

`prefix.init`: Used when `-loadinit` is specified. The first line in the file is the initial sequence.

`prefix.seed`: Used when `-loadseed` is specified. The first line is an integer random seed for the design algorithm (unique seeds are in the range $[0, 2^{32} - 1]$).

`preventfile`: Specifies patterns to be prevented from appearing in the designed sequences. The file must contain one pattern per line using standard nucleic acid codes. No design will be produced if the sequence constraints cannot be satisfied.

Output: Output is written to the files:

- `prefix.summary`
The header of this file includes comments about thermodynamic and design parameters used in the design process. The first line below the header contains the strand sequences separated by + symbols.
- `prefix.init`
Generated if `-outputinit` is specified. It contains the initial sequence on the first line of the file.
- `prefix.seed`
Generated if `-outputseed` is specified. It contains the random seed on the first line of the file.
- `prefix.ppairs`
Generated if `-pairs` is specified. This file specifies the base pairing probabilities for the complex in the same format as the file generated by the executable `pairs`.

Example 3:
Design a sequence for a complex of three DNA strands intended to adopt a target secondary structure at 23 °C. The first 24 nucleotides are constrained to nucleotide H (corresponding to a 3-letter alphabet) and the specified list of patterns are prevented throughout. A seed is set to make the design repeatable.

Input file contents:

```
((((( ((((((((((+(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
.+))))))))))))))))))))))))))))))))))))))))))))))))))))))
HHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHH
```

Prevent file contents:

```
AAAA
CCCC
GGGG
UUUU
KKKKKK
MMMMMM
RRRRRR
SSSSSS
WWWWWW
YYYYYY
```

Seed file contents:

```
93
```

Command: `complexdesign -T 23 -material dna -pairs -loadseed -prevent $NUPACKHOME/doc/examples/complex-design/advanced/input/hcr-design.prevent $NUPACKHOME/doc/examples/complex-design/advanced/input/hcr-design`

4 Test Tube Analysis

Analyze the equilibrium base-pairing properties of a test tube of interacting nucleic acid strands.

4.1 `complexes`: calculate the partition function and equilibrium base-pairing properties for each complex in a test tube

Command: `complexes` [-T temperature] [-material parameters] [-pairs]
[-mfe] [-degenerate] [-dangles treatment] [-timeonly] [-v3.0]
[-quiet] prefix

Description: For the set of strands Ψ^0 , calculate the partition function, Q_j , for each complex $j \in \Psi$, corresponding to all complexes of up to L_{\max} strands. Significant additional functionality can be specified via command line flags. The output of `complexes` can be used as the input to the executables `concentrations` and `distributions`.

Additional options:

`-pairs`

Calculate base-pairing observables as for the `pairs` executable.

`-cutoff` cutoffvalue

Only probabilities and expected values at or above cutoffvalue (default: 0.001) are saved in the output file(s) generated when the `-pairs` flag is selected.

`-mfe`

Calculate all MFE structures for each complex as for the `mfe` executable. The `-degenerate` flag is only applicable in conjunction with the `-mfe` flag.

`-timeonly`

After generating the strand orderings for all complexes in Ψ , estimate the time it would take to compute all of the partition functions. The partition function calculations are not performed, the time estimate is written to the screen, and no output files are generated.

`-v3.0`

Revert to NUPACK 3.0 behavior (see [changes](#)).

`-quiet`

Suppress output to the screen.

Input: Input is read from the file `prefix.in`, where `prefix` is a command line argument. Comment lines are preceded by `%` and may be interspersed with data in input files; blank lines are not permitted. The input file must contain the following entries on separate lines:

- The number of distinct strand species, $|\Psi^0|$.
- The sequence for each distinct strand species (each on a separate line).
- The maximum complex size, L_{\max} .

In addition to considering all complexes up to size L_{\max} , the optional file `prefix.list` can be used to manually specify complexes with more than L_{\max} strands. Each ordered complex of size $L > L_{\max}$ is specified on a separate line by:

- A list of L integers from the range 1 to $|\Psi^0|$ representing the strand ordering for the complex.

Example 4:

Calculate the partition function, equilibrium pair probabilities, and MFE structure(s) for each complex in a test tube containing three DNA strand species that interact to form all complex species of up to four strands, plus additional larger complexes specified in a `.list` file.

Input file contents:

```
3
AGTCTAGGATTCGGCGTGGGTAA
TTAACCCACGCCGAATCCTAGACTCAAAGTAGTCTAGGATTCGGCGTG
AGTCTAGGATTCGGCGTGGGTAAACACGCCGAATCCTAGACTACTTTG
4
```

List file contents:

```
1 2 2 3 3
1 2 3 2 3
2 3 2 3 2
1 2 2 2 3 3
```

Command: `complexes -T 23 -material dna -pairs -mfe -degenerate`
`$NUPACKHOME/doc/examples/tube-analysis/advanced/input/hcr`

Output: Unless the `-quiet` flag is selected, `complexes` reports progress to the screen. By default there are two output files:

- `prefix.ocx`
 Contains the strand composition and free energy of each complex $j \in \Psi$. The first and second columns are integer strand composition and strand ordering identifiers, respectively, the next $|\Psi^0|$ columns are $A_{1,j} A_{2,j} \dots A_{|\Psi^0|,j}$ defining the number of each strand type in complex j , and the final column is ΔG_j for complex j .
- `prefix.ocx-key`
 Contains the strand ordering π_j for each complex $j \in \Psi$ containing L_j strands. The first and second columns are integer strand composition and strand ordering identifiers, respectively, and the remaining L_j columns are integers from the range 1 to $|\Psi^0|$. Note that the value of L_j may be different for each complex $j \in \Psi$.

Depending on the command line options, the following output files may also be written:

- `prefix.ocx-epairs`
 Generated if `-pairs` is selected. Contains the base-pairing expectation values for each type of distinct base pair in each complex. The relevant quantities are $E(i_{\{A\}} \cdot j_{\{B\}})$, the expected number of base i of strand species A that are paired to base j of strand species B in the complex. The entries are separated by comment lines (repeated `%` symbols), and each entry begins with a comment line containing the strand composition identifier `idcomp` and strand ordering identifier `idorder` expressed as “% compositionidcomp orderingidorder”.
- `prefix.ocx-ppairs`
 Generated if `-pairs` is selected. Similar to `.epairs` except that all strands in the complex are assumed to be distinct. The data in each entry are the same as those in the `.ppairs` file produced by the executable `pairs`.
- `prefix.ocx-mfe`
 Generated if `-mfe` is selected. Contains the minimum free energy and MFE structure(s) for each complex.

Each entry is formatted the same as the output for the `mfe` executable. The entries are separated by comment lines (repeated % symbols), and each entry begins with a comment line containing the strand composition identifier `idcomp` and strand ordering identifier `idorder` expressed as “% composition`idcomp` ordering`idorder`”. If the `-degenerate` flag is selected, the degenerate MFE structures for a given entry are separated by a comment line of repeated % symbols.

Changes: Relative to NUPACK 3.0, the following changes were introduced to the `complexes` executable:

- `-ordered` is on by default
- output files `.cx` and `.cx-epairs` are not written
- the comment lines in `.ocx-epairs` and `.ocx-mfe` employ updated terminology

Use the `-v3.0` option to revert to NUPACK 3.0 behavior.

4.2 concentrations: calculate the equilibrium concentration for each complex in a test tube

Command: `concentrations [-pairs] [-sort method] [-v3.0] [-quiet] prefix`

Description: Given a user-specified concentration for each strand species, calculates the equilibrium concentration of each complex species or base pair in a dilute solution (e.g., a test tube) (Dirks et al., 2007). Partition function information is read in from output files generated with the executable `complexes`.

Additional options:

`-pairs`

Compute base-pairing information for the entire solution using results from `prefix.ocx-epairs` as output by the executable `complexes`.

`-cutoff cutoffvalue`

Only ensemble pair fractions at or above `cutoffvalue` (default: 0.001) are saved in the output file `prefix.fpairs` generated when the `-pairs` flag is selected. Note that `cutoffvalue` should not be less than that used with `complexes` to generate the input files.

`-sort method`

The argument `method` is one of the following integers:

- 0: Output is listed in the same order as in the input file.
- 1: Output is sorted by the concentration of each complex (default).
- 2: Output is sorted first by the sum of the concentrations of all complexes with each strand composition and then by the concentration of each complex with that strand composition.
- 3: Output is sorted first by the strand composition identifier and then by the strand ordering identifier.
- 4: Output is sorted first by the number of strands in each complex, then by the integers $A_{1,j}$ $A_{2,j}$... $A_{|\Psi^0|,j}$ defining the number of each strand type in complex j (with $A_{1,j}$ having the highest precedence, followed by $A_{2,j}$, and so on), and finally by the strand ordering identifier.

`-v3.0`

Revert to NUPACK 3.0 behavior (see [changes](#)).

`-quiet`

Suppress output to the screen.

Input: Input is read from the file `prefix.ocx` output from the executable `complexes`. The temperature at which the calculation is done is read from a line in the comments of the `.ocx` input file that reads “% T = temperature”, where temperature is the temperature in °C. This line is automatically included in all output files of the executable `complexes`.

The input file `prefix.con` specifies the total molar concentration of each of $|\Psi^0|$ strand species on a separate line. The concentration may be in scientific notation (e.g., $1e-6$ for a strand species at 1 μ M concentration).

Output: Unless `-quiet` is selected, the following information is written to the screen:

- The error in conservation of mass for each strand species in molar.
- The free energy of the entire solution in kcal/L.
- The wall clock time for the calculation.

The output is written to the files:

- `prefix.eq`
The content is the same as the input file (except resorted, depending on the `-sort` option) with an extra column containing the concentration of the species in molar inserted after the free energy column.
- `prefix.fpairs`
Generated if `-pairs` is selected. Reports the fraction of each distinct base that is paired to each of the other distinct bases in solution. The relevant quantity is $f_A(i_A \cdot j_B)$, the expected fraction of strands of species A for which base i is paired to base j of strand species B (Dirks et al., 2007). The number of distinct bases in the dilute solution is $N_{\text{distinct}} \equiv \sum_{k=1}^{|\Psi^0|} N_k$, representing the total number of bases in all $|\Psi^0|$ strand species. Numbering the distinct bases from 1 to N_{distinct} , the quantity $f_A(i_A \cdot j_B)$ may be stored as an (asymmetric) $N_{\text{distinct}} \times N_{\text{distinct}}$ matrix; by augmenting the matrix with an extra column that contains the expected fraction of base i of strand species A that are unpaired, each row sums to unity. The file is formatted as follows. Following header comments, the first entry is the integer N_{distinct} . The remaining entries come in triplets of the form $[i \ j \ f]$, where $1 \leq i \leq N_{\text{distinct}}$ and $1 \leq j \leq N_{\text{distinct}} + 1$ are base numbers and f is the corresponding fraction from the augmented matrix.

Changes: Relative to NUPACK 3.0, the following change was introduced to the `concentrations` executable: the `-ordered` option is on by default. Use the `-v3.0` option to revert to NUPACK 3.0 behavior.

4.3 distributions: calculate the equilibrium population distribution and expected value for a few complexes in a small box

Command: `distributions [-maxstates big] [-writestates]
[-sort method] [-v3.0] [-quiet] prefix`

Description: The executable `distributions` calculates the partition function, Q_{box} , for a box containing a small number of strands, given user-defined populations for each strand species (Dirks et al., 2007). This is used to calculate the expected value and probability distribution of the population of each species of complex. Partition function information is read from output files generated with the executable `complexes`.

Additional options:

`-maxstates big`

The maximum number of states of the box to be enumerated (default: $1e7$). A segmentation fault will occur if the stack size on your machine is exceeded.

`-writestates`

Write a (typically large) output file describing properties for all population states of the system.

Example 5:

A cautionary tale. NUPACK calculates free energies and equilibrium concentrations of complexes as described in (Dirks et al., 2007) (in particular, if you plan to calculate equilibrium concentrations by hand, see endnote 13 regarding strand association penalties). For example, the following holds at equilibrium for a dilute solution containing strands A and B that can interact to form complex AB:

$$\begin{aligned} \frac{x_{AB}}{x_A x_B} &= \exp \left\{ -\frac{\Delta G_{AB} - \Delta G_A - \Delta G_B}{kT} \right\}, \\ &= \frac{[AB]/\rho_{H_2O}}{([A]/\rho_{H_2O})([B]/\rho_{H_2O})}, \\ &= \frac{[AB] \rho_{H_2O}}{[A][B]}, \end{aligned}$$

where for each complex, i , x_i is the *mole fraction*, $[i]$ is the concentration (e.g. in units of mol/L), ΔG_i is the free energy as reported by NUPACK, and ρ_{H_2O} (≈ 55.14 mol/L at 37°C) is the concentration of water.

Consider duplex formation for two RNA strands, A = GCGCG and B = CGCGC, present at concentrations of $[A]_0$ and $[B]_0$, respectively, in 1 M Na^+ at 37°C. The free energies given by NUPACK are

$$\Delta G_A = 0.00 \text{ kcal/mol}, \quad \Delta G_B = 0.00 \text{ kcal/mol}, \quad \Delta G_{AB} = -9.62 \text{ kcal/mol}.$$

If only these three complexes are considered, the concentration of AB is determined by finding the appropriate root of

$$\frac{[AB] \rho_{H_2O}}{([A]_0 - [AB])([B]_0 - [AB])} = \exp \left\{ -\frac{\Delta G_{AB} - \Delta G_A - \Delta G_B}{kT} \right\}.$$

For $[A]_0 = [B]_0 = 1 \mu\text{M}$, we get

$$[A] = [B] = 0.91 \mu\text{M}, \quad [AB] = 0.09 \mu\text{M}.$$

A common mistake is to forget to include the ρ_{H_2O} in the calculation. Doing so would give the erroneous result of $[AB] = 0.67 \mu\text{M}$. It is important to remember that

$$\exp \left\{ -\frac{\Delta G_{AB} - \Delta G_A - \Delta G_B}{kT} \right\} \neq \frac{[AB]}{[A][B]}!$$

Finally, note that we have artificially stipulated that only three complexes are allowed. However, the sequences of A and B are such that they may form homodimers. If we consider this possibility, we are left with a system of coupled nonlinear algebraic equations that are difficult to solve. NUPACK performs such calculations, and the resulting concentrations are

$$[A] = 0.686 \mu\text{M}, \quad [B] = 0.925 \mu\text{M}, \quad [AB] = 0.069 \mu\text{M}, \quad [AA] = 0.123 \mu\text{M}, \quad [BB] = 0.003 \mu\text{M},$$

significantly different from what we calculated neglecting the other complexes. Therefore, one must exercise caution when applying complex free energies to determination of equilibrium concentrations. It is best to directly use the `concentrations` executable or the NUPACK web application for these calculations.

-sort method

The argument method is one of the following integers:

- 1: (default) Output is sorted by the expected value of the population of each complex.
- 2: Output is sorted first by the sum of the expected values of all complexes with each strand composition and then by the expected value of each complex with that strand composition.
- 3: Output is sorted first by the complex composition identifier and then by the strand ordering identifier.
- 4: Output is sorted first by the number of strands in each complex, then by the integers $A_{1,j}$ $A_{2,j}$... $A_{|\Psi^0|,j}$ defining the number of each strand type in complex j (with $A_{1,j}$ having the highest precedence, followed by $A_{2,j}$, and so on), and finally by the strand ordering identifier.

-v3.0

Revert to NUPACK 3.0 behavior (see [changes](#)).

-quiet

Suppress output to the screen.

Input: Same format as for the executable `concentrations`, except the file `prefix.con` file is replaced by `prefix.count`, which specifies the total strand population, m_i^0 , for each strand species $i \in \Psi^0$ on a separate line. The last line of the file contains the volume of the box in liters. This may be entered in scientific notation (e.g., $1.4e-18$).

Output: Unless the `-quiet` flag is selected, the following information is written to the screen:

- The number of states of the box.
- The free energy of the entire box in units of kT and in units of kcal.
- The wall clock time for the calculation.

The output is written to the files:

- `prefix.dist`
The content is the same as the input file (with rows sorted according to `-sort`) with extra columns after the free energy column. The first extra column (for complex j) is the expected value of the population $\langle m_j \rangle$. Subsequent columns are $[p_j(0) p_j(1) \dots p_j(\max(m^0))]$. These represent the probability that complex j has population 0, 1, ..., $\max(m^0)$, at equilibrium.
- `prefix.states`
Generated when `-writestates` is selected. Each row corresponds to a population vector, m , for the box. The first column is the probability that the population vector occurs at equilibrium. The remaining entries come in triples: strand composition identifier, strand ordering identifier, nonzero population. This pattern continues for all complexes with non-zero populations.

Changes: Relative to NUPACK 3.0, the following change was introduced to the `distributions` executable: the `-ordered` option is on by default. Use the `-v3.0` option to revert to NUPACK 3.0 behavior.

5 Test Tube Design

Design the equilibrium base-pairing properties of a test tube of interacting nucleic acid strands.

5.1 `tubedefect`: calculate the test tube ensemble defect

Command: `tubedefect prefix`

Description: Calculate the test tube ensemble defect, C , representing the concentration of incorrectly paired nucleotides evaluated over the ensemble of a target test tube (Wolfe and Pierce, 2015). The target test tube is specified as a set of desired on-target complexes, each with a target secondary structure and target concentration, and a set of undesired off-target complexes, each with vanishing target concentration. The set of off-target complexes is specified to be all complexes up to a user-specified number of strands (excluding those complexes that are on-target complexes).

Input: The executable `tubedefect` reads a job description from NUPACK script file `prefix.np`. In `.np` script files, a comment begins with `#` and continues for the rest of the line; blank lines are permitted. A sample script file is shown in Example 6. See executable `tubedesign` for additional details on the scripting language.

Output: Following header comments, the test tube ensemble defect, C , and the normalized test tube ensemble defect, C/y^{nt} , are written to the screen (Wolfe and Pierce, 2015). Here, y^{nt} is the total concentration of nucleotides in the test tube.

Example 6:

Calculate the test tube ensemble defect for a specified sequence and target test tube at default temperature 37 °C. The target test tube contains 1 on-target dimer (with a target structure and target concentration) and all off-target complexes of up to 2 strands (each with vanishing target concentration).

Script file contents:

```
# set properties
material = dna

# define 2 sequence domains
domain a = CGTGAACATCGGCGTGGTTCGACCAACCCACACAAAAACCTA
domain b = TTCCCTCTATATTTCTACTCCCGACCACGCCGATGTTACG

# define 2 strands
strand leg1 = a
strand leg2 = b

# define target structure for 1 on-target complex
structure legs = (((((((((((((((((((((((.....+.....
.....))))))))))))))))))))))

# define strand ordering for 1 on-target complex
legs.seq = leg1 leg2

# define target test tube containing 1 on-target complex
tube walker = legs

# define target concentration for 1 on-target complex (molar)
# default: 1.0e-6
walker.legs.conc = 1.0e-6

# augment tube with all off-target complexes of up to 2 strands
# default: 0
walker.maxsize = 2
```

Command: tubedefect

```
$NUPACKHOME/doc/examples/tube-design/simple/input/walker-defect
```

5.2 `tubedesign`: design the equilibrium base-pairing properties of a test tube

Command: `tubedesign prefix`

Description: Perform sequence design for a test tube of interacting nucleic acid strands. The user specifies a set of desired on-target complexes, each with a target secondary structure and target concentration, and a set of undesired off-target complexes, each with vanishing target concentration. The set of off-target complexes is specified to be all complexes up to a user-specified number of strands (excluding those complexes that are on-target complexes). Sequence design is formulated as an optimization problem with the goal of reducing the test tube ensemble defect below a user-specified stop condition (Wolfe and Pierce, 2015).

Input: The executable `tubedesign` reads a job description from NUPACK script file `prefix.np`. In a `.np` file, a comment begins with `#` and continues for the rest of the line; blank lines are permitted. Specification of a simple test tube design job is illustrated in Example 7. Specification of a test tube design job that employs sequence domains to enforce complementarity constraints is illustrated in Example 8.

Output: Output is written to commented output file `prefix.out`.
See examples in `$NUPACKHOME/doc/examples/tube-design/`

Example 7:

Design a sequence for a target test tube at default temperature 37 °C. The target test tube contains 1 on-target dimer (with a target structure and target concentration) and all off-target complexes of up to 2 strands (each with vanishing target concentration).

Script file contents:

```
# set properties
material = dna
seed = 93 # set seed to make design repeatable

# define target structure for 1 on-target complex
structure legs = (((((((((((((((((((((((.....+.....
.....))))))))))))))))))))))

# define target test tube containing 1 on-target complex
tube walker = legs

# define target concentration for 1 on-target complex (molar)
# default: 1.0e-6
walker.legs.conc = 1.0e-6

# augment tube with all off-target complexes of up to 2 strands
# default: 0
walker.maxsize = 2
```

Command: `tubedesign`

`$NUPACKHOME/doc/examples/tube-design/simple/input/walker-design`

Example 8:

Design a sequence for a target test tube at 23 ° C. The target test tube contains 2 on-target monomers and 1 on-target trimer (each with a target structure and target concentration) and all off-target complexes of up to 3 strands (each with vanishing target concentration).

Script file contents:

```
# set properties
material = dna
temperature = 23
seed = 93 # set seed to make design repeatable

# define sequence domains
domain a = N6
domain b = N18
domain c = N6

# define strands
strand I = b* a*
strand A = a b c* b*
strand B = b* a* b c

# define target structure for each on-target complex
structure monomer1 = U6 D18 U6
structure monomer2 = D18 U6 U6
structure polymer = D24 (D24 (+ U24) +)

# define strand ordering for each on-target complex
monomer1.seq = A
monomer2.seq = B
polymer.seq = A B I

# define target test tube containing 3 on-target complexes
tube HCR = monomer1 monomer2 polymer

# define target concentration for each on-target complex
# default: 1.0e-6 (molar)
HCR.monomer1.conc = 1.0e-6
HCR.monomer2.conc = 1.0e-6
HCR.polymer.conc = 1.0e-7

# augment tube with all off-target complexes of up to 3 strands
# default: 0
HCR.maxsize = 3

# set stop condition for normalized test tube ensemble defect
# default: 0.01
HCR.stop = 0.02
```

Command: tubedesign

```
$NUPACKHOME/doc/examples/tube-design/advanced/input/hcr-design
```

Optional commands: In a `.np` script file, parameter values are set as follows (defaults shown):

```
# physical model parameters: see options for details
material = rna                # values: rna, dna, rna1995, rna1999, dna1998
temperature = 37.0           # °C
sodium = 1.0                 # in interval [0.05,1.1], molar
magnesium = 0.0              # in interval [0.0,0.2], molar
dangles = some               # values: none, some, all

# algorithm parameters: see (Wolfe and Pierce, 2015) for details
hsplit = 2                   # default: 2 for rna, 3 for dna
nsplit = 12                  #
fsplit = 0.99                # in interval (0,1)
fstringent = 0.99            # in interval (0,1)
dgclamp = -25.0              # kcal/mol
mbad = 300                   #
mreseed = 50                 #
mreopt = 3                   #
fpassive = 0.01              # in interval (0,1)
fredecomp = 0.03            # in interval (0,1)
frefocus = 0.03             # in interval (0,1)
allowwobble = false          # allow algorithm to introduce wobble pairs
initgc = 0.5                 # in interval [0,1], initial GC content
maxopttime = 31536000        # seconds
```


6 Getting Started

6.1 Compilation and installation

- Required packages

The following package must be installed to compile NUPACK3.1:

CMake (version 2.6.0+) - A cross-platform open-source build system

This can be installed with a standard package manager (e.g., [Homebrew](#)) or downloaded from the software developer's website. Note: *You may need root privileges to perform default installations (e.g., precede the installation command with `sudo` and provide a password when prompted).*

- NUPACK root directory

Unpack the file `nupack.tar.gz` and place the `nupack` root directory in a convenient location (e.g., `/usr/local/nupack` or `$HOME/nupack`). Set the environment variable `NUPACKHOME` to specify an absolute path to the `nupack` root directory:

```
export NUPACKHOME=/usr/local/nupack
```

- Compiling and installing NUPACK

To compile NUPACK, type the following commands from within the `nupack` root directory:

```
mkdir build
cd build
cmake ../
make
```

To install NUPACK, then type the following command:

```
make install
```

Note: *You may need root privileges to perform a default installation (e.g., type `sudo make install` and provide a password when prompted).* The default installation will put executables in `/usr/local/bin`, libraries in `/usr/local/lib`, headers in `/usr/local/include`, and parameters in `/usr/local/share`. You are now ready to run NUPACK executables. To start, you may want to run some of the [examples](#) in `$NUPACKHOME/doc/examples`.

- Custom builds

If you wish to compile multiple versions of NUPACK, simply make additional build directories in `$NUPACKHOME`. Various build configuration options can be passed to `cmake`. For example, to specify different compilers, replace the default call to `cmake` with:

```
cmake -DCMAKE_C_COMPILER=gcc -DCMAKE_CXX_COMPILER=g++ ../
```

See [CMake](#) documentation for configuration options.

- Custom installation

To install NUPACK in a custom location, replace the default call to `cmake` with:

```
cmake -DCMAKE_INSTALL_PREFIX=prefix ../
```

The `install` command will then put executables in `prefix/bin`, libraries in `prefix/lib`, headers in `prefix/include`, and parameters in `prefix/share`. For custom installations, set the `NUPACKINSTALL` environment variable:

```
export NUPACKINSTALL=prefix
```

so that NUPACK executables can find NUPACK parameter files. Also, you must add `NUPACKINSTALL/bin` to your path:

```
export PATH=$PATH:$NUPACKINSTALL/bin
```

so that the NUPACK executables can be run without specifying an absolute path.

6.2 Examples

Following the organization of the NUPACK User Guide, sample input and output files are provided in `$NUPACKHOME/doc/examples` with the following directory structure:

```
complex-analysis
  simple, advanced, pseudoknot
  runjobs, input, output, output.ref
complex-design
  simple, advanced
  runjobs, input, output, output.ref
tube-analysis
  simple, advanced
  runjobs, input, output, output.ref
tube-design
  simple, advanced
  runjobs, input, output, output.ref
```

Four subdirectories (`complex-analysis`, `complex-design`, `tube-analysis`, `tube-design`) correspond to different problem classes. Within each of these, three subdirectories (`simple`, `advanced`, `pseudoknot`) or two subdirectories (`simple`, `advanced`) correspond to different sample calculations. Within each of these, there is a shell script (`runjobs`) and three subdirectories containing job files (`input`, `output`, `output.ref`).

Run the shell script `runjobs` to call multiple NUPACK executables, reading input files from the directory `input` and writing output files to the directory `output`. The generated results in directory `output` can be compared to the reference results in directory `output.ref` to check that your local installation of NUPACK 3.1 is running properly.

Alternatively, in directory `$NUPACKHOME/doc/examples`, run the script `runall` to run all of the `runjobs` scripts (a total of 9 `runjobs` scripts). Then run the script `diffall` to compare the files in the `output` directories to the corresponding files in the `output.ref` directories.

7 Acknowledgments

NUPACK is supported by the National Science Foundation via the Molecular Programming Project (NSF-CCF-1317694), by the Gordon and Betty Moore Foundation (GBMF2809), and by the Beckman Institute at Caltech (PMTIC). NUPACK has previously been supported by the John Simon Guggenheim Memorial Foundation, by the National Science Foundation (NSF-CCF-0832824, NSF-CHE-0533064, NSF-DMS-0506468, NSF-CCF-CAREER-0448835), by the National Institutes of Health (P50 HG004071), by the Ralph M. Parsons Foundation, and by the Charles Lee Powell Foundation.

8 References

- R. M. Dirks and N. A. Pierce. A partition function algorithm for nucleic acid secondary structure including pseudoknots. *J. Comput. Chem.*, 24:1664–1677, 2003. [\(pdf\)](#)
- R. M. Dirks and N. A. Pierce. An algorithm for computing nucleic acid base-pairing probabilities including pseudoknots. *J. Comput. Chem.*, 25:1295–1304, 2004. [\(pdf\)](#)
- R. M. Dirks, M. Lin, E. Winfree, and N. A. Pierce. Paradigms for computational nucleic acid design. *Nucleic Acids Res.*, 32(4):1392–1403, 2004. [\(pdf\)](#)
- R. M. Dirks, J. S. Bois, J. M. Schaeffer, E. Winfree, and N. A. Pierce. Thermodynamic analysis of interacting nucleic acid strands. *SIAM Rev.*, 49(1):65–88, 2007. [\(pdf\)](#)
- R. T. Koehler and N. Peyret. Thermodynamic properties of DNA sequences: Characteristic values for the human genome. *Bioinformatics*, 21(16):3333–3339, 2005.
- D. H. Mathews, J. Sabina, M. Zuker, and D. H. Turner. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *J. Mol. Biol.*, 288:911–940, 1999.
- J. SantaLucia, Jr. A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbor thermodynamics. *Proc. Natl. Acad. Sci. USA*, 95(4):1460–1465, 1998.
- J. SantaLucia, Jr. and D. Hicks. The thermodynamics of DNA structural motifs. *Annu. Rev. Biophys. Biomol. Struct.*, 33:415–440, 2004.
- N.C. Seeman. Nucleic acid junctions and lattices. *J. Theor. Biol.*, 99:237–247, 1982.
- M. J. Serra and D. H. Turner. Predicting thermodynamic properties of RNA. *Methods Enzymol.*, 259:242–261, 1995.
- B. R. Wolfe and N. A. Pierce. Nucleic acid sequence design for a test tube of interacting nucleic acid strands. *ACS Synth. Biol.*, 4:1086–1100, 2015. [\(pdf\)](#)
- J. N. Zadeh. *Algorithms for Nucleic Acid Sequence Design*. PhD thesis, California Institute of Technology, 2010.
- J. N. Zadeh, C. D. Steenberg, J. S. Bois, B. R. Wolfe, M. B. Pierce, A. R. Khan, R. M. Dirks, and N. A. Pierce. NUPACK: Analysis and design of nucleic acid systems. *J. Comput. Chem.*, 32(1):170–173, 2011a. [\(pdf\)](#)
- J. N. Zadeh, B. R. Wolfe, and N. A. Pierce. Nucleic acid sequence design via efficient ensemble defect optimization. *J. Comput. Chem.*, 32:439–452, 2011b. [\(pdf\)](#)
- M. Zuker. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res.*, 31(13):3406–3415, 2003.