# Inferring Parameters for an Elementary Step Model of DNA Structure Kinetics with Locally Context-Dependent Arrhenius Rates
# *Appendix*


Sedigheh Zolaktaf[1], Frits Dannenberg[2], Xander Rudelis[2], Anne Condon[1], Joseph M Schaeffer[3], Mark Schmidt[1], Chris Thachuk[2], and Erik Winfree[2](✉)

[1] University of British Columbia, Vancouver, BC, Canada
[2] California Institute of Technology, Pasadena, CA, USA
winfree@caltech.edu
[3] Autodesk Research, San Francisco, CA, USA


## Table of Contents

## A Local Context

In this document, we use 0-based numbering for numbering bases, i.e, in a multi-strand complex, for each strand of length $l$, the first nucleotide at the 5' end of the strand is numbered 0 and the last nucleotide at the 3' end of the strand is indexed $l-1$.

To determine the local context of a base pair forming or breaking in a pseudoknot-free structure, we use Algorithm 1. This algorithm uses *dot-parens-plus* notation to represent a secondary structure (state), which consists of '(', ')', '.', and '+'. Matching parentheses represent bases which have formed a base pair, a dot represents a free base pair, and a plus represents a break between strands. For example, '((((((+))))))' means that bases 0, 1, 2, 3, 4, and 5 of the first strand are paired to bases 5, 4, 3, 2, 1, and 0 of the second strand, respectively. When all base pairs between strands break, we replace the plus sign by a space. For example, '...... .......' means that no base pair is formed.

---

**Algorithm 1:** Find the local context of a base pair forming or breaking

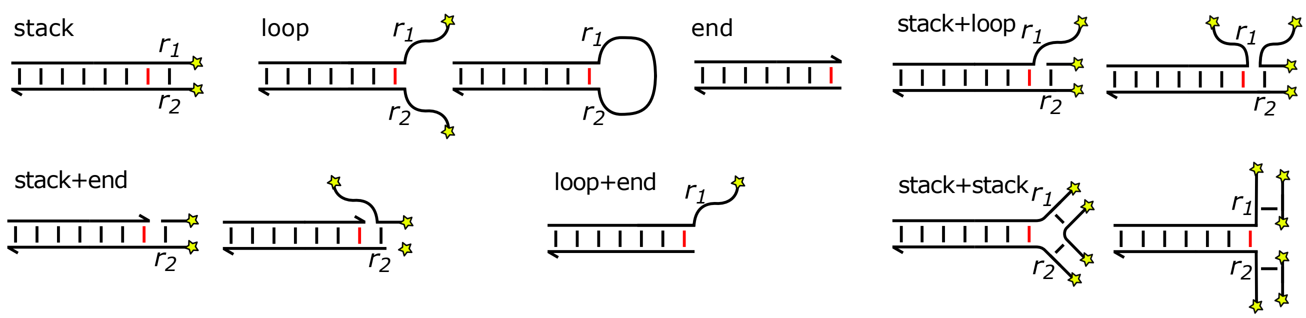
---

**Function** `LocalContext(`$s_i, s_j$`)`

- **Input**: States $s_i$ and state $s_j$, which differ in exactly one base pair. (Either of the states can have an extra base pair compared to the other.)
- **Output**: $\langle l, r\rangle$, which is the local context of the base pair breaking or forming.
- $d_i \leftarrow$ dot-parens-plus notation of $s_i$
- $d_j \leftarrow$ dot-parens-plus notation of $s_j$
- Insert '*' at the start and end of $d_i$ and $d_j$, before and after all '+' signs, and before and after every space
- $(p_1, p_2) \leftarrow$ the first and second positions where $d_i$ and $d_j$ differ, respectively
- $l \leftarrow$ `HalfContext` $(d_i, p_1 - 1, p_2 + 1)$ `// Algorithm 2`
- $r \leftarrow$ `HalfContext` $(d_i, p_1 + 1, p_2 - 1)$
- return $\langle l, r\rangle$

---

**Algorithm 2:** Find the half context on one side of a base pair forming or breaking




**Function** `HalfContext`($d, f_1, f_2$)

**Input**: $d$ is a dot-parans-plus notation, $f_1$ and $f_2$ represent one side of the base pair forming or breaking. ($f_1$ and $f_2$ are adjacent to the base pair forming or breaking).

**Output**: The half context appearing in positions $f_1$ and $f_2$.

$c_1 \leftarrow d[f_1]$
$c_2 \leftarrow d[f_2]$
**if** $c_1 =$ '(' and $c_2 =$ ')' **then**
  counter $\leftarrow 0$
  **for** $k$ in $[f_1, f_2]$ **do**
    **if** $d[k] =$ '(' **then** counter $\leftarrow$ counter + 1
    **else if** $d[k] =$ ')' **then** counter $\leftarrow$ counter - 1
    **if** counter $= 0$ **then**
      **if** $k = p_j$ **then** return stack
      **else** return stack+stack
**if** ($c_1 =$ '(' and $c_2 =$ '(' ) or ($c_1 =$ ')' and $c_2 =$ ')' ) or ($c_1 =$ ')' and $c_2 =$ '(' ) **then** return stack+stack
**if** ( $c_1 =$ '(' and $c_2 =$ '.' ) or ( $c_1 =$ ')' and $c_2 =$ '.' ) or ( $c_1 =$ '.' and $c_2 =$ ')' ) or ( $c_1 =$ '.' and $c_2 =$ '(' ) **then** return stack+loop
**if** $c_1 =$ '(' and $c_2 =$ '*' ) or ( $c_1 =$ ')' and $c_2 =$ '*' ) or ( $c_1 =$ '*' and $c_2 =$ ')' ) or ( $c_1 =$ '*' and $c_2 =$ '(' ) **then** return stack+end
**if** ( $c_1 =$ '.' and $c_2 =$ '*' ) or ( $c_1 =$ '*' and $c_2 =$ '.' ) **then** return loop+end
**if** $c_1 =$ '*' and $c_2 =$ '*' **then** return end
**if** $c_1 =$ '.' and $c_2 =$ '.' **then** return loop

# B Interacting DNA Strands State Space

To generate the state space we use Algorithm 3 in combination with a reaction-specific set of initial states $\mathcal{S}_{\text{init}}$ and final states $\mathcal{S}_{\text{final}}$ and a reaction-specific function that returns neighboring states for a state $s$, `NeighborStates`$(s)$. Algorithm 3 uses a breadth-first search approach: initially, the queue $Q$ and candidate state space $\mathcal{S}_{\text{init}}$ are composed of just the initial states. For every state in the queue, unexplored successor states are added to the candidate state space and then queued for exploration. In this paper, we use only one initial state and one final state per reaction.

Next, we explain the state space we used specific to different types of reactions in our dataset.

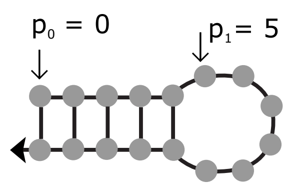
**Algorithm 3:** Generate state space

```
Function GenerateStateSpace
    S ← S_init, Q ← S_init
    while Q ≠ ∅ do
        N ← ∅
        foreach s ∈ Q do
            foreach s_p ∈ NeighborStates(s) do
                if s_p ∉ S and s_p ∉ S_final then
                    S ← S ∪ s_p
                    N ← N ∪ s_p
        Q ← N
    S ← S ∪ S_final
    return S
```

### B.1 Hairpin Closing and Opening

Let $l$ be the length of the hairpin strand and let $m < l/2$ be the length of the stem in the fully closed position. Each state is represented by a tuple $\langle p_i, p_1 \rangle$, where $0 \leq p_0 \leq p_1 \leq m$. The tuple indicates that the bases $p_0$ to $p_1 - 1$ are paired with bases $l - p_1$ to $l - p_0 - 1$ respectively, and no other base pairs are formed. Algorithm 4 describes the neighbors of a hairpin state $s$. The state space of hairpin opening and closing are equal, except that the initial and final states are swapped. In hairpin closing, in the initial state ($\mathcal{S}_{\text{init}} = \{\langle 0, 0 \rangle\}$), no base pairs have formed. In the final state ($\mathcal{S}_{\text{final}} = \{\langle 0, m \rangle\}$), all base pairs have formed.

**Algorithm 4:** Hairpin state $s = \langle p_0, p_1 \rangle$

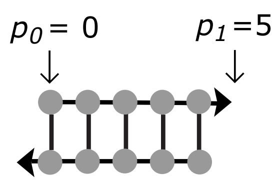



**Function** `NeighborStates`($s = \langle p_0, p_1 \rangle$)
    `// This function returns possible neighbors of state` $s$
    $\mathcal{N} \leftarrow \emptyset$
    `// Consider possibly invalid new states, then remove the invalid ones`
    **if** $\langle p_0, p_1 \rangle = \langle 0, 0 \rangle$ **then**
        **for** $p \in [0, m-1]$ **do**
            $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p, p+1 \rangle$
    **else**
        $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 1, p_1 \rangle \cup \langle p_0 + 1, p_1 \rangle \cup \langle p_0, p_1 - 1 \rangle \cup \langle p_0, p_1 + 1 \rangle$
    **for** $s' \in \mathcal{N}$ **do**
        `// The state in which no base pair has formed is shown by` $\langle 0, 0 \rangle$
        **if** $p_0 = p_1$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle 0, 0 \rangle$
    **for** $s' \in \mathcal{N}$ **do**
        **if** !`AllowedState`($s'$) **then** $\mathcal{N} \leftarrow \mathcal{N} \setminus s'$ `// Remove invalid states`
    return $\mathcal{N}$
**Function** `AllowedState`($s' = \langle p_0, p_1 \rangle$)
    **if** $!(0 \leq p_0 \leq p_1 \leq m)$ **then** return False
    return True

### B.2 Helix Association and Dissociation

Let $l$ be the length of a strand in the helix. Each state is represented with a tuple $\langle p_0, p_1\rangle$, where $0 \leq p_0 \leq p_1 \leq l$. The tuple indicates that all bases numbered $p_0$ to $p_1 - 1$ in one strand have paired with bases numbered $l - p_1$ to $l - p_0 - 1$ in the other strand, respectively, and there are no other base pairs in the state. Algorithm 5 describes the neighbors of a helix state $s$. The state space of helix association and dissociation are equal, except that the initial and final states are swapped. In helix association, in the initial state ($\mathcal{S}_{\text{init}} = \{\langle 0, 0\rangle\}$), no base pairs have formed between the two strands. In the final state ($\mathcal{S}_{\text{final}} = \{\langle 0, l\rangle\}$), all base pairs have formed between the two strands.

**Algorithm 5:** Helix state $s = \langle p_0, p_1\rangle$

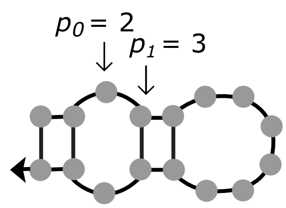



**Function** `NeighborStates` ($s = \langle p_0, p_1\rangle$)
    `// This function returns possible neighbors of state` $s$
    $\mathcal{N} \leftarrow \emptyset$
    `// Consider possibly invalid new states, then remove the invalid ones`
    **if** $\langle p_0, p_1\rangle = \langle 0, 0\rangle$ **then**
        **for** $p$ in $[0, l-1]$ **do**
            $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p, p+1\rangle$
    **else**
        $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 1, p_1\rangle \cup \langle p_0 + 1, p_1\rangle \cup \langle p_0, p_1 - 1\rangle \cup \langle p_0, p_1 + 1\rangle$
    **for** $s' \in \mathcal{N}$ **do**
        `// The state in which no base pair has formed is shown by` $\langle 0, 0\rangle$
        **if** $p_0 = p_1$ and $p_0 \neq 0$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle 0, 0\rangle$
    **for** $s' \in \mathcal{N}$ **do**
        **if** `!AllowedState(`$s'$`)` **then** $\mathcal{N} \leftarrow \mathcal{N} \setminus s'$    `// Remove invalid states`
    return $\mathcal{N}$
**Function** `AllowedState(`$s' = \langle p_0, p_1\rangle$`)`
    **if** $!(0 \leq p_0 \leq p_1 \leq l)$ **then** return False
    return True

### B.3 Bubble Closing

Let $l$ be the length of the hairpin strand, $m < l/2$ be the length of the stem in the fully closed position, and $f$ be the position where a bubble is formed. Each state is represented with a tuple $\langle p_0, p_1 \rangle$, where $0 < p_0 < p_1 < m$. The tuple indicates that all bases numbered 0 to $p_0 - 1$ have paired with bases numbered $l - p_0$ to $l - 1$, respectively, and all bases numbered $p_1$ to $m - 1$ have paired with bases numbered $l - m$ to $l - p_1 - 1$, respectively, and there are no other base pairs in the state. Algorithm 6 describes the neighbors of a bubble closing state $s$. In the initial state ($\mathcal{S}_{\text{init}} = \{\langle f-1, f\rangle\}$), all base pairs in the hairpin stem have formed except for a bubble of size 1 in the stem at position $f$. In the final state ($\mathcal{S}_{\text{final}} = \{\langle f, f\rangle\}$), all base pairs have formed.

**Algorithm 6:** Bubble state $s = \langle p_0, p_1 \rangle$




**Function** `NeighborStates` $(s = \langle p_0, p_1 \rangle)$
  `// This function returns possible neighbors of state` $s$
  $\mathcal{N} \leftarrow \emptyset$
  `// Consider possibly invalid new states, then remove the invalid ones`
  $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 1, p_1 \rangle \cup \langle p_0 + 1, p_1 \rangle \cup \langle p_0, p_1 - 1 \rangle \cup \langle p_0, p_1 + 1 \rangle$
  **for** $s' \in \mathcal{N}$ **do**
    `// The state in which all base pairs have formed is shown by` $\langle f, f \rangle$
    **if** $p_0 = p_1$ and $p_0 \neq f$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle f, f \rangle$
  **for** $s' \in \mathcal{N}$ **do**
    **if** !`AllowedState`$(s')$ **then** $\mathcal{N} \leftarrow \mathcal{N} \setminus s'$ `// Remove invalid states`
  return $\mathcal{N}$
**Function** `AllowedState`$(s' = \langle p_0, p_1 \rangle)$
  **if** $!(0 < p_0 \leq f \leq p_1 < m)$ **then** return False
  return True

### B.4 Toehold-mediated 3-way Strand Displacement

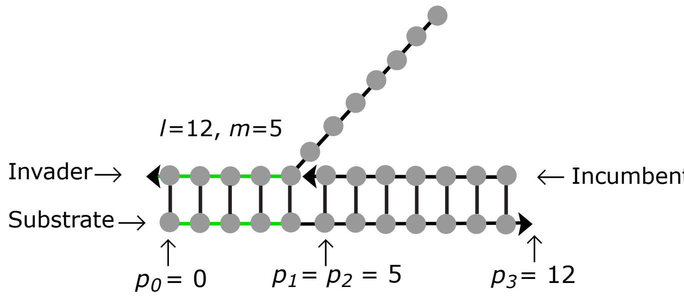
Let $l$ be the length of the substrate. For simplicity, let $l$ also be the length of the invader, $m$ be the toehold length, and $l - m$ be the length of the incumbent. Each state is represented with a tuple $\langle p_0, p_1, p_2, p_3 \rangle$, where $0 \leq p_0 \leq p_1 \leq p_2 \leq p_3 \leq l$ and $p_2 \geq m$. The tuple indicates that all bases numbered $p_0$ to $p_1 - 1$ in the substrate have paired with bases numbered $l - p_1$ to $l - p_0 - 1$ in the invader, respectively, all bases numbered $p_2$ to $p_3 - 1$ in the substrate have paired with bases numbered $l - p_3$ to $l - p_2 - 1$ in the incumbent, respectively, and there are no other base pairs in the state. Algorithm 7 describes the neighbors of a toehold-mediated 3-way strand displacement state $s$. In the initial state ($\mathcal{S}_{\text{init}} = \{\langle 0, 0, m, l \rangle\}$), the substrate is completely attached to the incumbent, but completely detached from the invader. In the final state ($\mathcal{S}_{\text{final}} = \{\langle 0, l, l, l \rangle\}$), the substrate is completely detached from the incumbent, but completely attached to the invader. Algorithm 8 adapts algorithm 7 for toehold-mediated 3-way strand displacement with mismatches between the invader and the substrate. In the algorithm, $mp$ is a pointer to the mismatch position.

Note that in both algorithms, to efficiently obtain mean first passage times with sparse matrix computations, we further heuristically prune the state space of each reaction. For example, we disallow states which have a large gap between the incumbent and the invader.

### B.5 Toehold-mediated 4-way Strand Exchange

Let *complex* be the first helix and *complex1* and *complex2* be the two strands in this helix. Let *reporter* be the second helix and *reporter1* be the strand in this helix that is complementary to *complex1* and *reporter2* be the strand in this helix that is complementary to *complex2*. Let $l$ be the length of the helices excluding their toehold. For simplicity, let $m$ be the toehold length of *complex1* and *reporter1* and let $n$ be the toehold length of *complex2* and *reporter2*. Each state is represented with a tuple $\langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle$. The tuple indicates that all bases numbered $p_0$ to $p_{0'} - 1$ in *complex1* have paired with bases numbered $l + m - p_{0'}$ to $l + m - p_0 - 1$ in *reporter1*, respectively, all bases numbered 0 to $p_2 - 1$ in *complex1* have paired with bases numbered $l + n - p_2$ to $l + n - 1$ in *complex2*, respectively, all bases numbered $p_1$ to $p_{1'} - 1$ in *reporter2* have paired with bases numbered $l + n - p_{1'}$ to $l + n - p_1 - 1$ in *complex2*, respectively, all bases numbered 0 to $p_3 - 1$ in *reporter2* have paired with bases numbered $l + m - p_3$ to $l + m - 1$ in *reporter1*, respectively, and there are no other base pairs in the state. Algorithm 9 describes the neighbors of a toehold-mediated 4-way strand exchange state $s$. In the initial state ($\mathcal{S}_{\text{init}} = \{\langle l+m, l+n, l+m, l+n, l, l \rangle\}$), *complex1* and *complex2* are completely bound except in their toeholds (have formed the *complex* helix), *reporter1* and *reporter2* are completely bound except in their toeholds (have formed the *reporter* helix), and no base pairs have formed between the *complex* helix and the *reporter* helix. Hence, each helix has two complementary strands except for their toeholds. In the final state ($\mathcal{S}_{\text{final}} = \{\langle 0, 0, l+m, l+n, 0, 0 \rangle\}$), the *reporter* and *complex* helices have completely exchanged strands and two new helices, which have complementary strands, are formed.

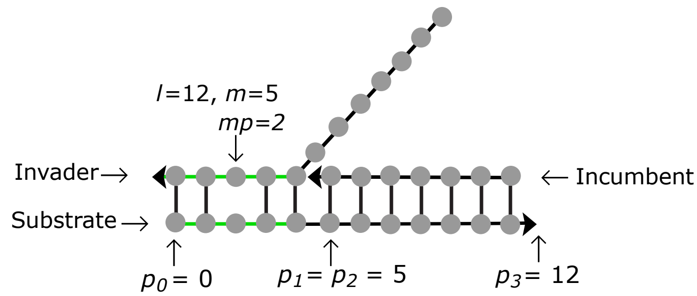
**Algorithm 7:** Toehold-mediated 3-way strand displacement state $s = \langle p_0, p_1, p_2, p_3 \rangle$




**Function** `NeighborStates` $(s = \langle p_0, p_1, p_2, p_3 \rangle)$
  `// This function returns possible neighbors of state` $s$
  $\mathcal{N} \leftarrow \emptyset$
  `// Consider possibly invalid new states, then remove the invalid ones`
  **if** $p_0 = p_1$ **then**
    `// If the invader and the substrate are detached, they can form a base pair`
    **for** $p$ in $[0, p_2 - 1]$ **do**
      $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p, p+1, p_2, p_3 \rangle$
  **else**
    `// The invader and substrate can form or break a base pair`
    $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 1, p_1, p_2, p_3 \rangle \cup \langle p_0 + 1, p_1, p_2, p_3 \rangle \cup \langle p_0, p_1 - 1, p_2, p_3 \rangle \cup \langle p_0, p_1 + 1, p_2, p_3 \rangle$
    `// The incumbent and substrate can form or break a base pair`
    $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1, p_2 - 1, p_3 \rangle \cup \langle p_0, p_1, p_2 + 1, p_3 \rangle \cup \langle p_0, p_1, p_2, p_3 - 1 \rangle \cup \langle p_0, p_1, p_2, p_3 + 1 \rangle$
  **for** $s' \in \mathcal{N}$ **do**
    `// States in which the substrate and invader are detached are shown by` $\langle 0, 0, p_2, p_3 \rangle$
    **if** $p_0 = p_1$ and $p_0 \neq 0$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle 0, 0, p_2, p_3 \rangle$
    `// States in which the substrate and incumbent are detached are shown by` $\langle p_0, p_1, l, l \rangle$
    **if** $p_2 = p_3$ and $p_2 \neq l$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle p_0, p_1, l, l \rangle$
  **for** $s' \in \mathcal{N}$ **do**
    **if** `!AllowedState`$(s')$ **then** $\mathcal{N} \leftarrow \mathcal{N} \setminus s'$ `// Remove invalid states`
  return $\mathcal{N}$

**Function** `AllowedState`$(s' = \langle p_0, p_1, p_2, p_3 \rangle)$
  **if** $!(0 \leq p_0 \leq p_1 \leq p_2 \leq p_3 \leq l$ and $p_2 \geq m)$ **then** return False
  `// Heuristically, further prune the state space to enable sparse matrix computations`
  **if** $p_0 = p_1$ and $p_2 = p_3$ **then** return False `// Disallow the complex to dissociate into three strands`
  **if** $(0 < m < p_2)$ and $(p_0 \neq 0$ or $p_1 < p_2 - 1)$ **then** return False `// When there is gap of greater than one base pair between the invader and the incumbent or the invader is not bound to the substrate, disallow the first base pair of the incumbent and the substrate to break`
  **if** $p_2 - p_1 > m + 2$ **then** return False `// Disallow states which have a large gap between the incumbent and the invader`
  return True

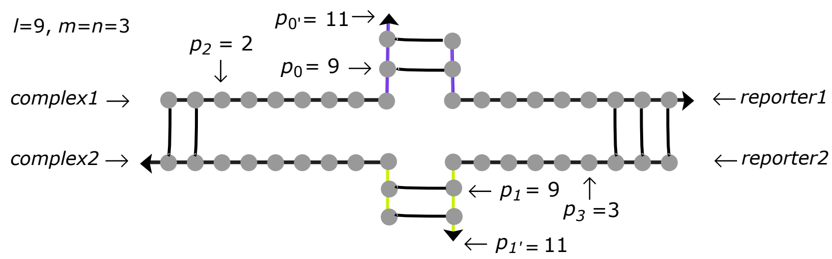
**Algorithm 8:** Toehold-mediated 3-way strand displacement state $s = \langle p_0, p_1, p_2, p_3 \rangle$ that has a mismatch between the invader and the substrate




**Function** `NeighborStates` ($s = \langle p_0, p_1, p_2, p_3 \rangle$)
- `// This function returns possible neighbors of state` $s$
- $\mathcal{N} \leftarrow \emptyset$
- `// Consider possibly invalid new states, then remove the invalid ones`
- **if** $p_0 = p_1$ **then**
  - `// If the invader and the substrate are detached, they can form a base pair that is not located at the mismatch position`
  - **for** $p$ in $[0, p_2 - 1]$ **do**
    - **if** $p \neq mp$ **then**
      - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p, p+1, p_2, p_3 \rangle$
- **else**
  - `// The incumbent and substrate can form or break a base pair`
  - $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1, p_2 - 1, p_3 \rangle \cup \langle p_0, p_1, p_2 + 1, p_3 \rangle \cup \langle p_0, p_1, p_2, p_3 - 1 \rangle \cup \langle p_0, p_1, p_2, p_3 + 1 \rangle$
  - `// The invader and substrate can form or break a base pair not located at the mismatch position`
  - **if** $p_0 - 1 \neq mp$ **then** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 1, p_1, p_2, p_3 \rangle$
  - **else** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 - 2, p_1, p_2, p_3 \rangle$
  - **if** $p_0 + 1 \neq mp$ **then** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 + 1, p_1, p_2, p_3 \rangle$
  - **else** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0 + 2, p_1, p_2, p_3 \rangle$
  - **if** $p_1 - 1 \neq mp$ **then** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1 - 1, p_2, p_3 \rangle$
  - **else** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1 - 2, p_2, p_3 \rangle$
  - **if** $p_1 + 1 \neq mp$ **then** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1 + 1, p_2, p_3 \rangle$
  - **else** $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1 + 2, p_2, p_3 \rangle$
- **for** $s' \in \mathcal{N}$ **do**
  - `// States in which the substrate and invader are detached are shown by` $\langle 0, 0, p_2, p_3 \rangle$
  - **if** $p_0 = p_1$ and $p_0 \neq 0$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle 0, 0, p_2, p_3 \rangle$
  - `// States in which the substrate and incumbent are detached are shown by` $\langle p_0, p_1, l, l \rangle$
  - **if** $p_2 = p_3$ and $p_2 \neq l$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle p_0, p_1, l, l \rangle$
- **for** $s' \in \mathcal{N}$ **do**
  - **if** `!AllowedState`($s'$) **then** $\mathcal{N} \leftarrow \mathcal{N} \setminus s'$ `// Remove invalid states`
- return $\mathcal{N}$

**Function** `AllowedState`($s' = \langle p_0, p_1, p_2, p_3 \rangle$)
- **if** $!(0 \leq p_0 \leq p_1 \leq p_2 \leq p_3 \leq l$ and $p_2 \geq m)$ **then** return False
- `// Heuristically, further prune the state space to enable sparse matrix computations`
- **if** $p_0 = p_1$ and $p_2 = p_3$ **then** return False `// Disallow the complex to dissociate into three strands`
- **if** $(0 < m < p_2)$ and $(p_0 \neq 0$ or $p_1 < p_2 - 5)$ **then** return False `// When there is gap of greater than five base pairs between the invader and the incumbent or the invader is not bound to the substrate, disallow the first base pair of the incumbent and the substrate to break`
- **if** $p_2 - p_1 > m + 4$ **then** return False `// Disallow states which have a large gap between the incumbent and the invader`
- return True

**Algorithm 9:** Toehold-mediated 4-way strand exchange state $s = \langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle$




**Function** `NeighborStates` ($s = \langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle$)

  `// This function returns possible neighbors of state` $s$
  $\mathcal{N} \leftarrow \emptyset$
  `// Consider possibly invalid new states, then remove the invalid ones`
  **if** $p_0 = p_{0'}$ **then**
    `// If complex1 and` *`reporter1`* `are detached, they can form a base pair`
    **for** $p$ in $[\max\{p_2, p_3\}, l+m-1]$ **do**
      $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p, p_1, p+1, p_{1'}, p_2, p_3 \rangle$
  **else**
    `// complex1 can form or break a base pair with` *`reporter1`*
    $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0-1, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle \cup \langle p_0+1, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle \cup \langle p_0, p_1, p_{0'}-1, p_{1'}, p_2, p_3 \rangle \cup \langle p_0, p_1, p_{0'}+1, p_{1'}, p_2, p_3 \rangle$
  **if** $p_1 = p_{1'}$ **then**
    `// If reporter2 and` *`complex2`* `are detached, they can form a base pair`
    **for** $p$ in $[\max\{p_2, p_3\}, l+n-1]$ **do**
      $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p, p_1, p+1, p_2, p_3 \rangle$
  **else**
    `// reporter2 can form or break a base pair with` *`complex2`*
    $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1-1, p_{0'}, p_{1'}, p_2, p_3 \rangle \cup \langle p_0, p_1+1, p_{0'}, p_{1'}, p_2, p_3 \rangle \cup \langle p_0, p_1, p_{0'}, p_{1'}-1, p_2, p_3 \rangle \cup \langle p_0, p_1, p_{0'}, p_{1'}+1, p_2, p_3 \rangle$
  **if** $(p_0 != p_{0'}$ or $p_1 != p_{1'})$ or $(m = 0$ or $n = 0)$ **then**
    `// If` *`complex1`* `and` *`reporter1`* `have attached or` *`complex2`* `and` *`reporter2`* `have attached or a toehold does not exist, then` *`complex1`* `and` *`complex2`* `can form or break a base pair`
    $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1, p_{0'}, p_{1'}, p_2-1, p_3 \rangle \cup \langle p_0, p_1, p_{0'}, p_{1'}, p_2+1, p_3 \rangle$
    `// If` *`complex1`* `and` *`reporter1`* `have attached or` *`complex2`* `and` *`reporter2`* `have attached or a toehold does not exist, then` *`reporter1`* `and` *`reporter2`* `can form or break a base pair`
    $\mathcal{N} \leftarrow \mathcal{N} \cup \langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3-1 \rangle \cup \langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3+1 \rangle$
  **for** $s' \in \mathcal{N}$ **do**
    `// States in which` *`complex1`* `and` *`reporter1`* `are detached are shown by` $\langle l+m, p_1, l+m, p_{1'}, p_2, p_3 \rangle$
    **if** $p_0 = p_{0'}$ and $0 \le p_0 < l+m$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle l+m, p_1, l+m, p_{1'}, p_2, p_3 \rangle$
    `// States in which` *`reporter2`* `and` *`complex2`* `are detached are shown by` $\langle p_0, l+n, p_{0'}, l+n, p_2, p_3 \rangle$
    **if** $p_1 = p_{1'}$ and $0 \le p_1 < l+n$ **then** $\mathcal{N} \leftarrow (\mathcal{N} \setminus s') \cup \langle p_0, l+n, p_{0'}, l+n, p_2, p_3 \rangle$
  **for** $s' \in \mathcal{N}$ **do**
    **if** !`AllowedState`($s'$) **then** $\mathcal{N} \leftarrow \mathcal{N} \setminus s'$ `// Remove invalid states`
  return $\mathcal{N}$

**Function** `AllowedState`($s' = \langle p_0, p_1, p_{0'}, p_{1'}, p_2, p_3 \rangle$)

  **if** $!(p_3 \le p_0$ and $p_3 \le p_1$ and $p_2 \le p_0$ and $p_2 \le p_1$ and $0 \le p_2 \le l$ and $0 \le p_3 \le l$ and $0 \le p_0 \le p_{0'} \le l+m$ and $0 \le p_1 \le p_{1'} \le l+n)$ **then** return False
  `// Heuristically, further prune the state space to enable sparse matrix computations`
  **if** $(p_0 = p_{0'}$ or $p_1 = p_{1'})$ and $(p_2 = 0$ or $p_3 = 0)$ **then** return False `// Disallow the complex to dissociate into three or four complexes`
  **if** $(m = 0$ or $n = 0)$ and $(p_0 = p_{0'}$ or $p_1 = p_{1'})$ and $(l - p_2 > 3 - m/3$ or $l - p_3 > 3 - n/3)$ **then** return False `// When one of the toeholds does not exist and the reporter and complex have not attached from both sides, disallow the complex and the reporter to break more than` $3 - m/3$ `and` $3 - n/3$ `base pairs, respectively`
  **if** $(m \ne 0$ and $n \ne 0)$ and $(p_0 = p_{0'}$ or $p_1 = p_{1'})$ and $(p_2 < l-1$ or $p_3 < l-1)$ **then** return False `// When neither of the toeholds is 0 and the reporter and the complex haven not attached from both sides, disallow the complex and the reporter to break more than one base pair`
  **if** $p_{0'} < l$ or $p_{1'} < l$ **then** return False
  **if** $(p_0 \ne p_{0'}$ and $p_1 \ne p_{1'})$ and $(|p_0 - p_3| + |p_0 - p_2| + |p_1 - p_3| + |p_1 - p_2| > 8 - n/3 - m/3)$ **then** return False
  `// When the complex and reporter are attached from both sides, disallow large loops`
  return True

## C Half Context Frequency

Fig. 1 shows the fraction of all unimolecular elementary steps that involve a given half context, i.e., $\frac{\#l}{\sum_{l \in \mathcal{C}} \#l}$, where $\mathcal{C} = \{\text{stack}, \text{loop}, \text{end}, \text{stack+loop}, \text{stack+end}, \text{loop+end}, \text{stack+stack}\}$ is the set of half contexts and $\#l$ is the number of all unimolecular elementary steps that involve the half context $l$. Analogously, Fig. 2 shows the fraction of all bimolecular elementary steps that involve a given half context.

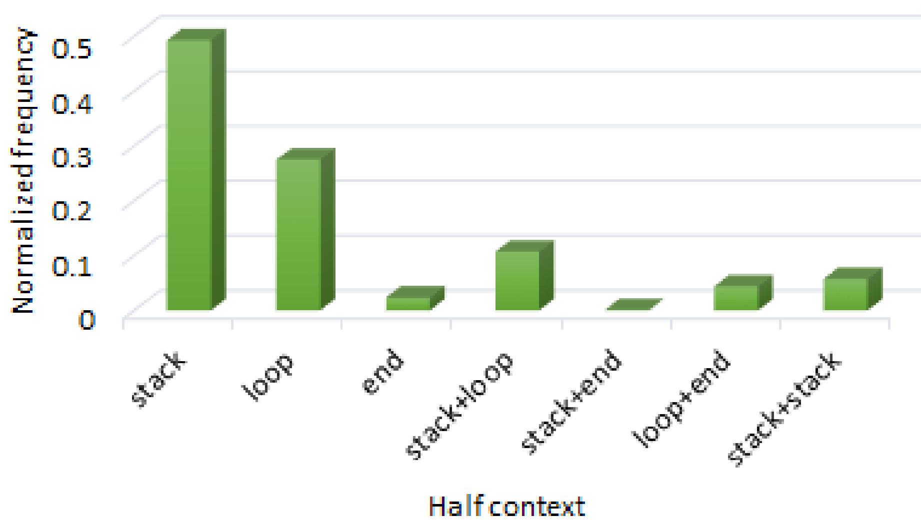



Fig. 1: Normalized frequency of the half contexts in unimolecular transitions.

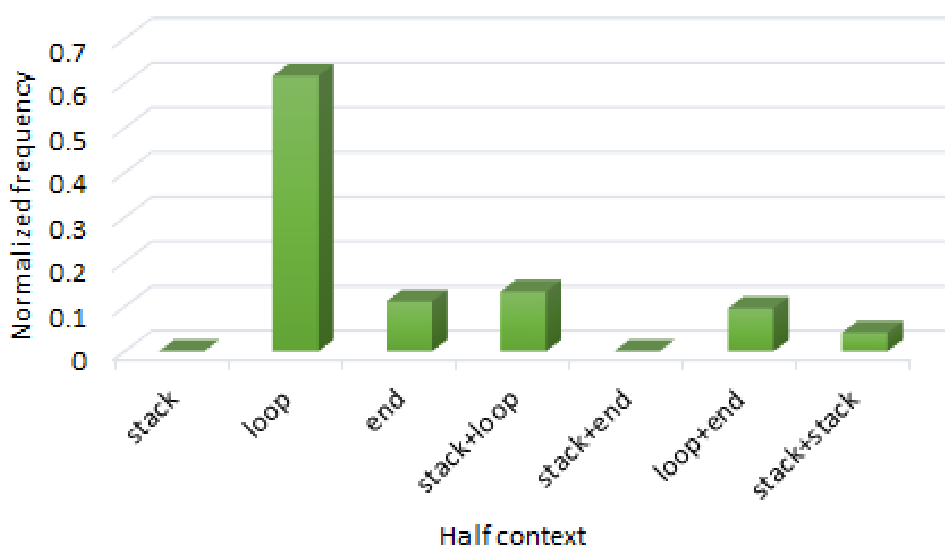



Fig. 2: Normalized frequency of the half contexts in bimolecular transitions.

# D Experimental Plot Reproduction

The following plots show how the performance of the Metropolis and the Arrhenius models on the training and testing datasets. Dashed lines indicate model fits and predictions and solid lines indicate experimentally determined values. For the MCMC ensemble method, error bars indicate the range (minimum to maximum) of predictions.

## D.1 Training Set ($\mathcal{D}_{\text{train}}$)

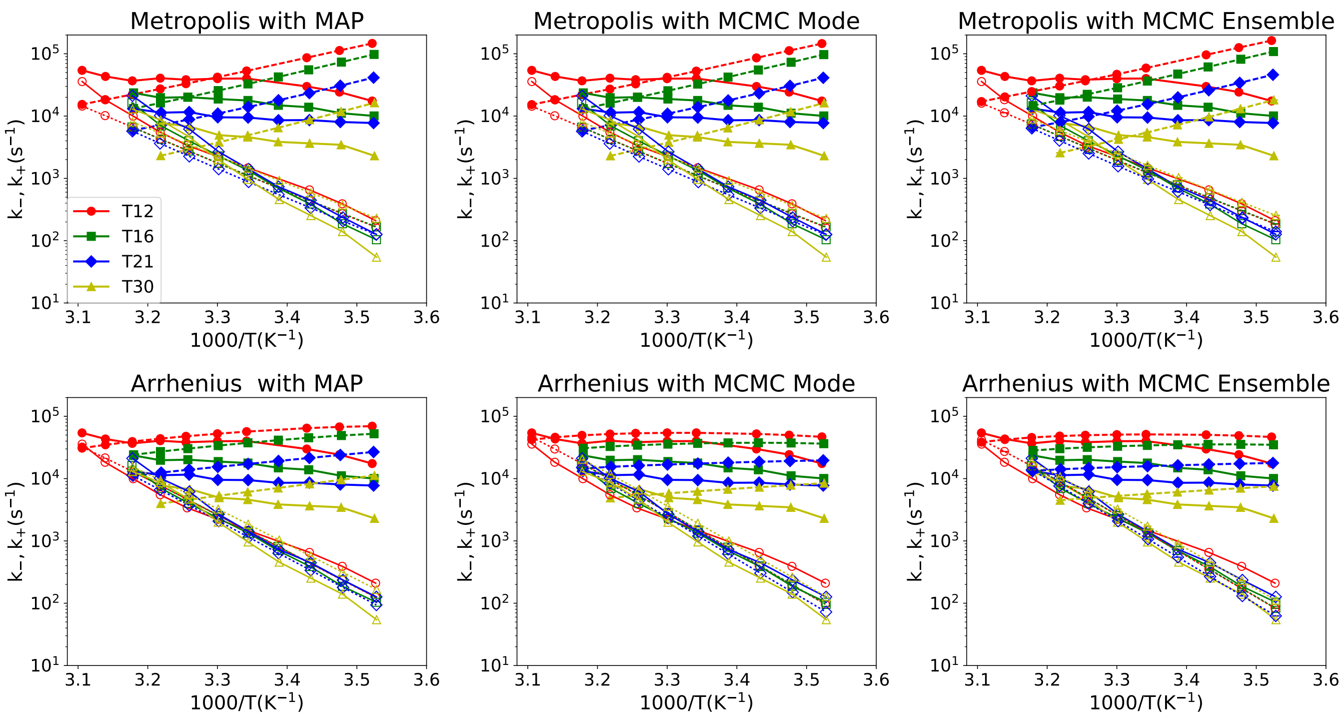



Fig. 3: Model fitting (dashed lines) of reaction rate constants (y axis) for hairpin closing (solid) and opening (open) with sequence $5'$-$CCCAA$-$(T)_n$-$TTGGG$-$3'$ where $n$ is 12,16, 21, or 30, experimental data (solid lines) from Fig. 4 of Bonnet et al. [3].

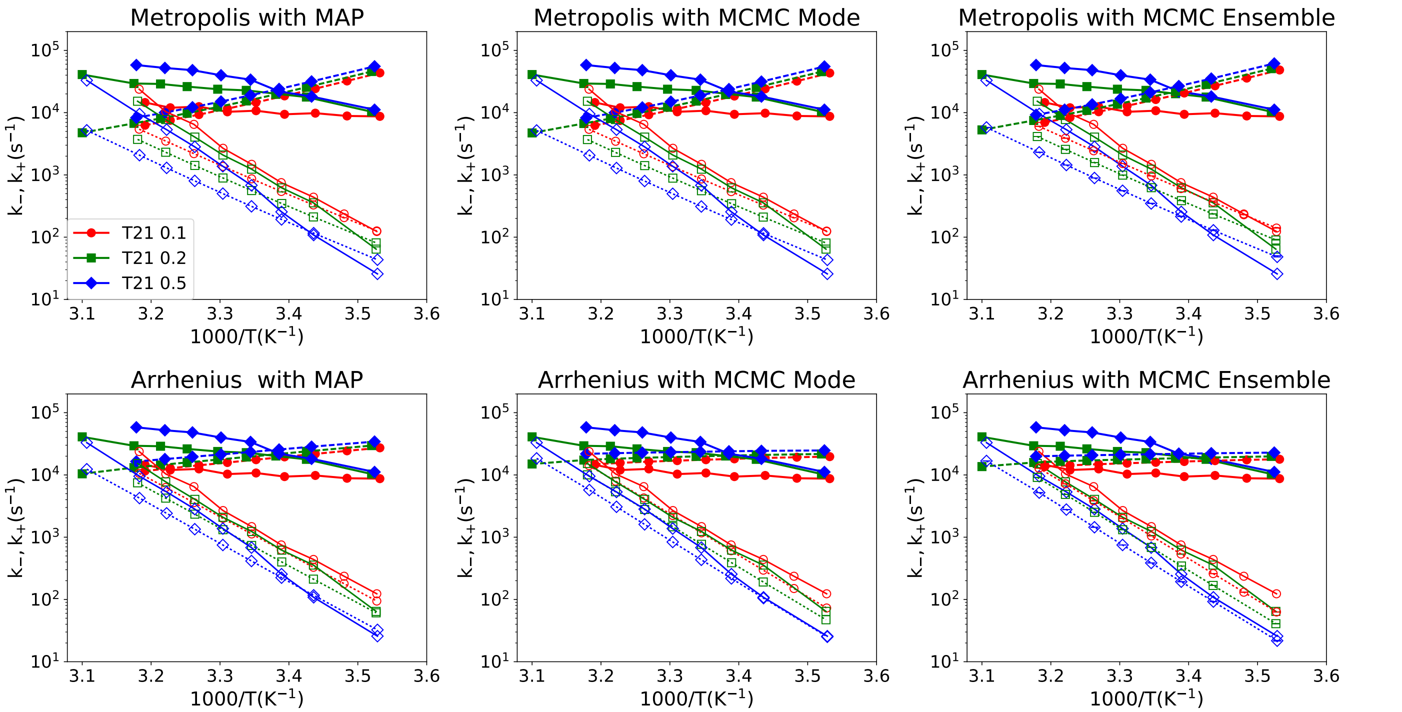



Fig. 4: Model fitting (dashed lines) of reaction rate constants (y axis) for hairpin opening (open) and closing (solid) with sequence $5'$-$CCCAA$-$(T)_{21}$-$TTGGG$-$3'$ at different salt concentrations, Fig. 6 from Bonnet et al. [3]. experimental data (solid lines) from Fig.6 of Bonnet et al. [3] wrongfully notes the use of a poly-A instead of a poly-T hairpin loop, which becomes evident in comparison to Fig. 5 of the same work (private communication with the authors).

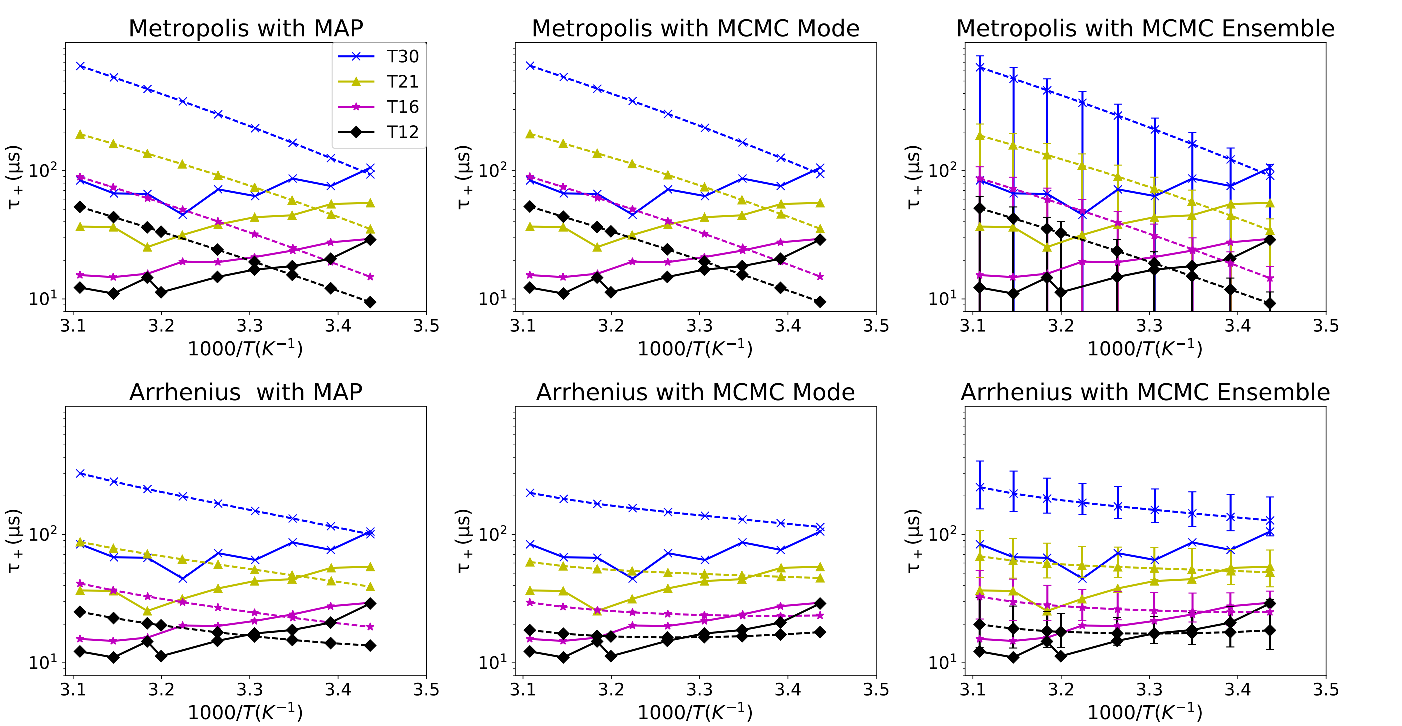



Fig. 5: Model fitting (dashed lines) of reaction timescales (y axis) for hairpin closing with sequence $5'$-$CCCAA$-$(T)_n$-$TTGGG$-$3'$ where $n$ is 12,16, 21, or 30, experimental data (solid lines) from Fig. 3.28 of Bonnet [2].

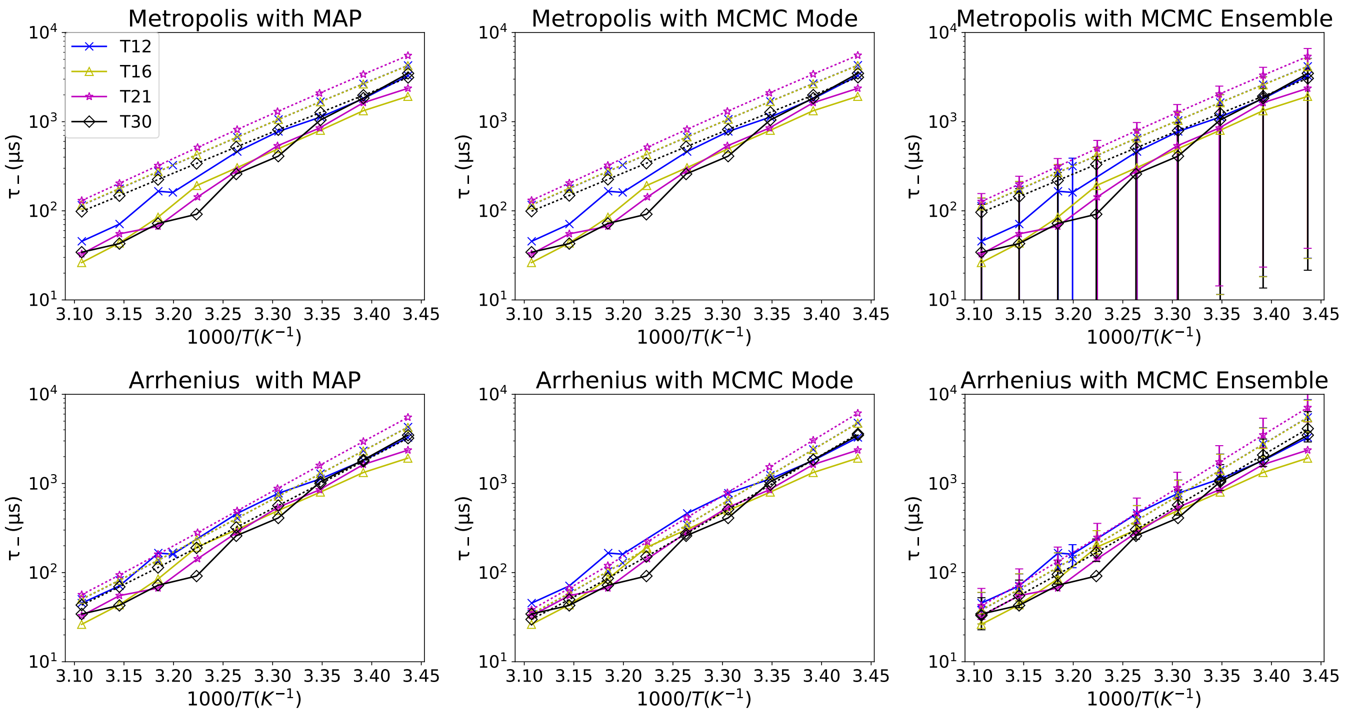



Fig. 6: Model fitting (dashed lines) of reaction timescales (y axis) for hairpin opening with sequence $5'$-$CCCAA$-$(T)_n$-$TTGGG$-$3'$ where $n$ is 12,16, 21, or 30, experimental data (solid lines) from Fig. 3.28 of Bonnet [2].

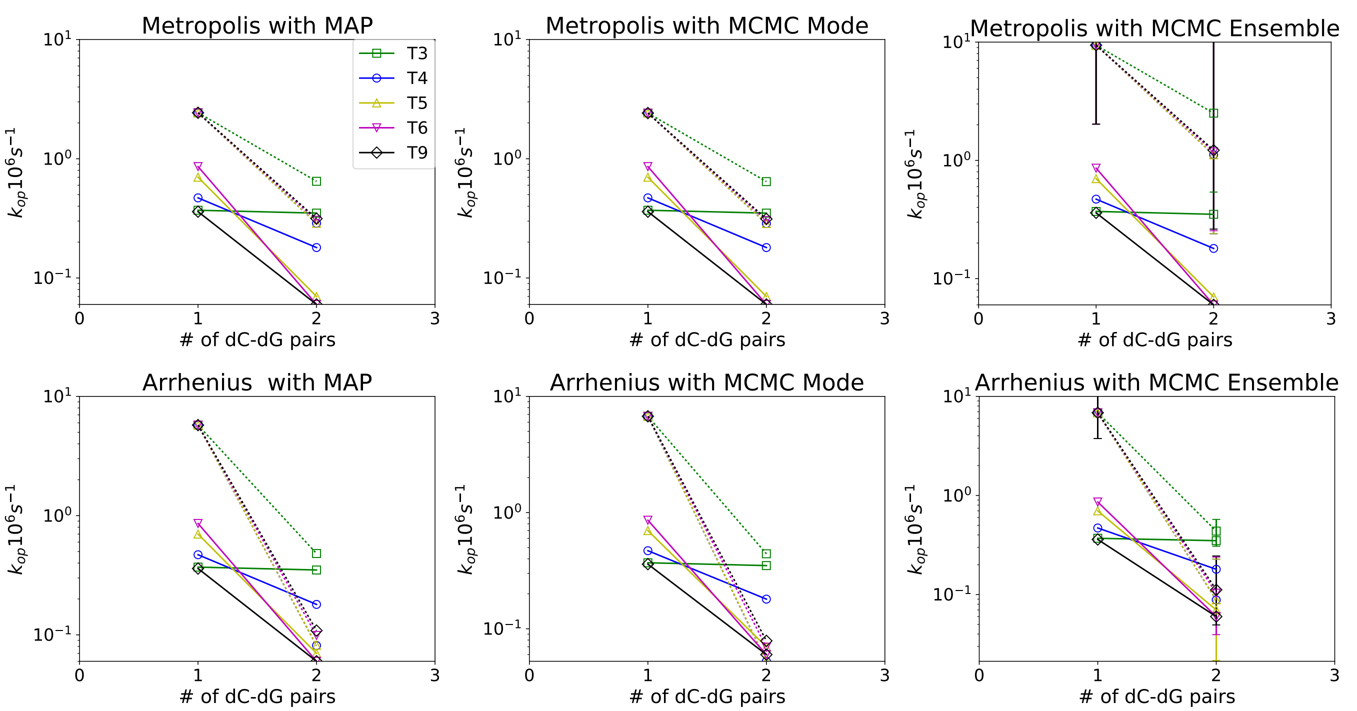



Fig. 7: Model fitting (dashed lines) of reaction rate constants (y axis) for hairpin opening with sequence $F$-$(dC)_y$-$(dT)_x$-$(dG)_y$ (x ranging from 3 to 9) as a function of $dC$-$dG$ pairs ($y$ ranging from 1 to 2), experimental data (solid lines) from Table 1 (Fig. 3b) of Kim et al. [5].

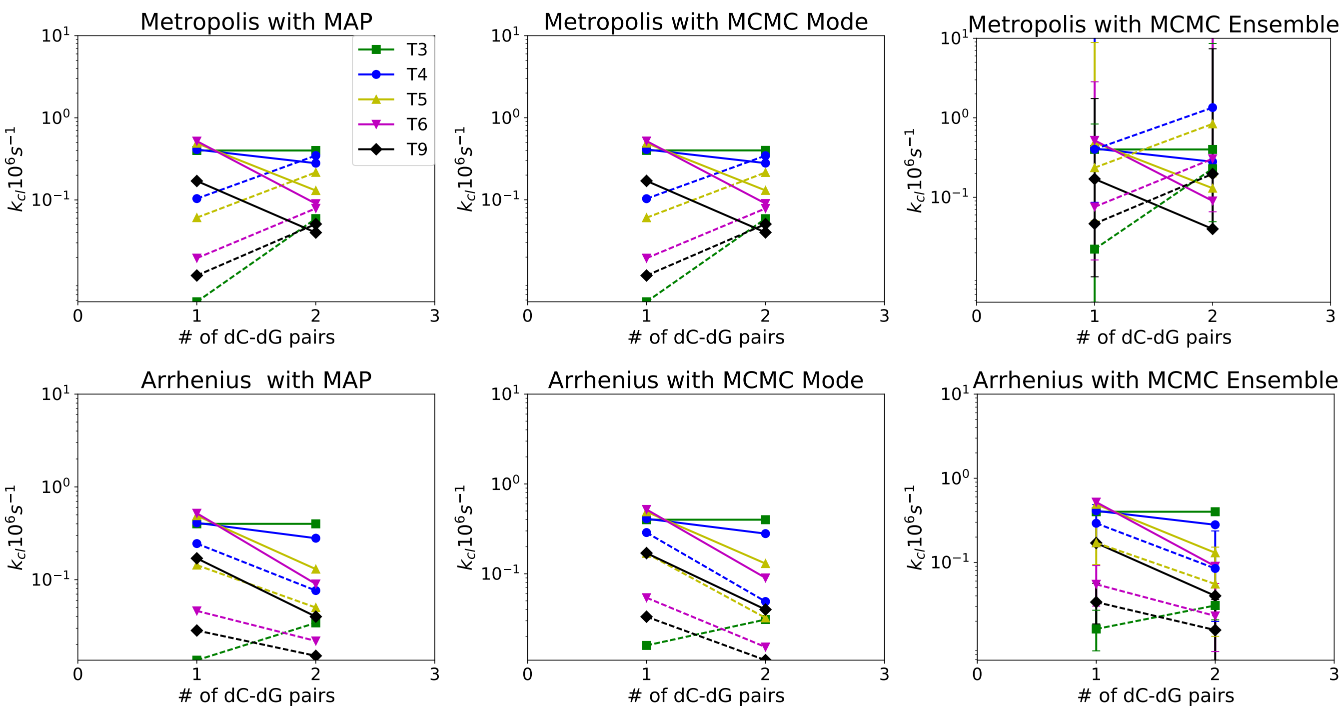



Fig. 8: Model fitting (dashed lines) of reaction rate constants (y axis) for hairpin closing with sequence $F$-$(dC)_y$-$(dT)_x$-$(dG)_y$ ($x$ ranging from 3 to 9) as a function of $dC$-$dG$ pairs ($y$ ranging from 1 to 2), experimental data (solid lines) from Table 1 (Fig. 3b) of Kim et al. [5].

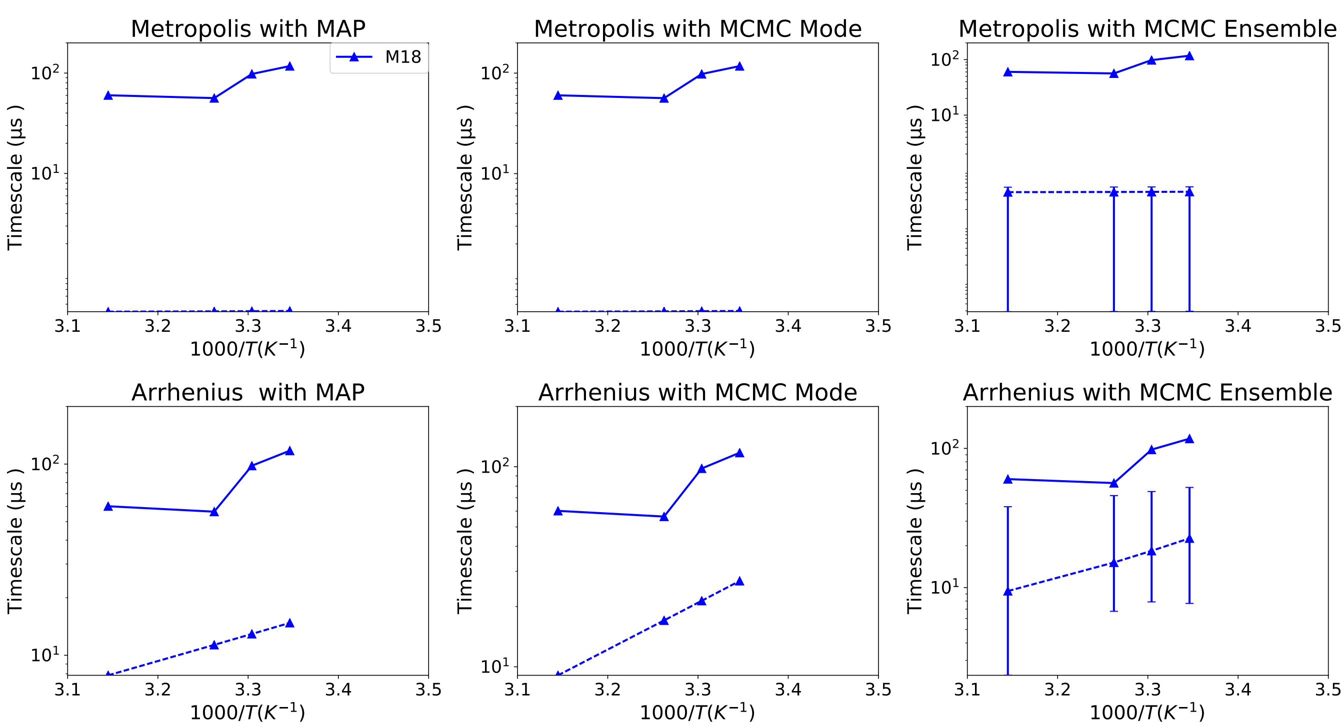



Fig. 9: Model fitting (dashed lines) of reaction timescales (y axis) for bubble closing with sequence $M_{18}$, experimental data (solid lines) from Fig. 4 of Altan-Bonnet et al. [1].

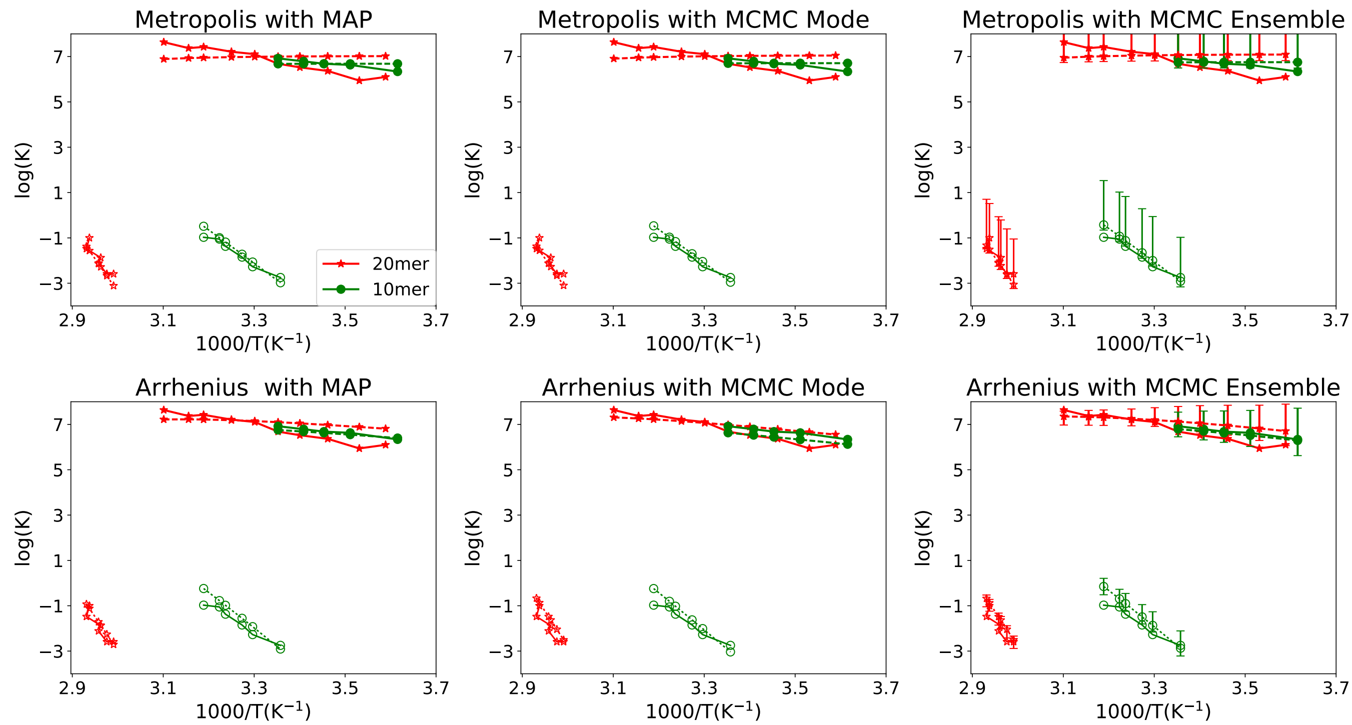



Fig. 10: Model fitting (dashed lines) of reaction rate constants (y axis) for helix association (solid) and disassociation (solid), experimental data (solid lines) from Fig. 6 of Morrison and Stols [7]. 10mer and 20mer are variation in the length of the strand.

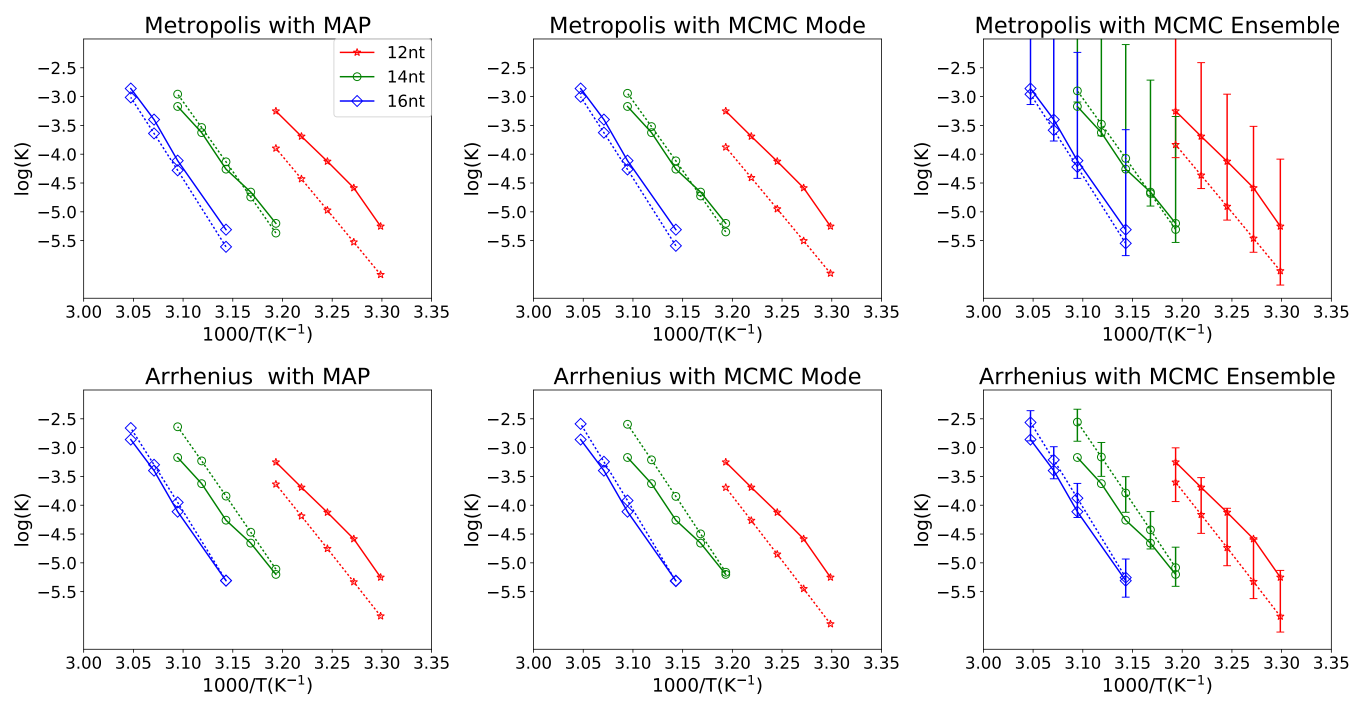



Fig. 11: Model fitting (dashed lines) of of reaction rate constants (y axis) for helix disassociation, experimental data (solid lines) from Fig. 6 of Reynaldo et al. [8]. 12nt, 14nt, and 16nt are variations in the length of the strand.

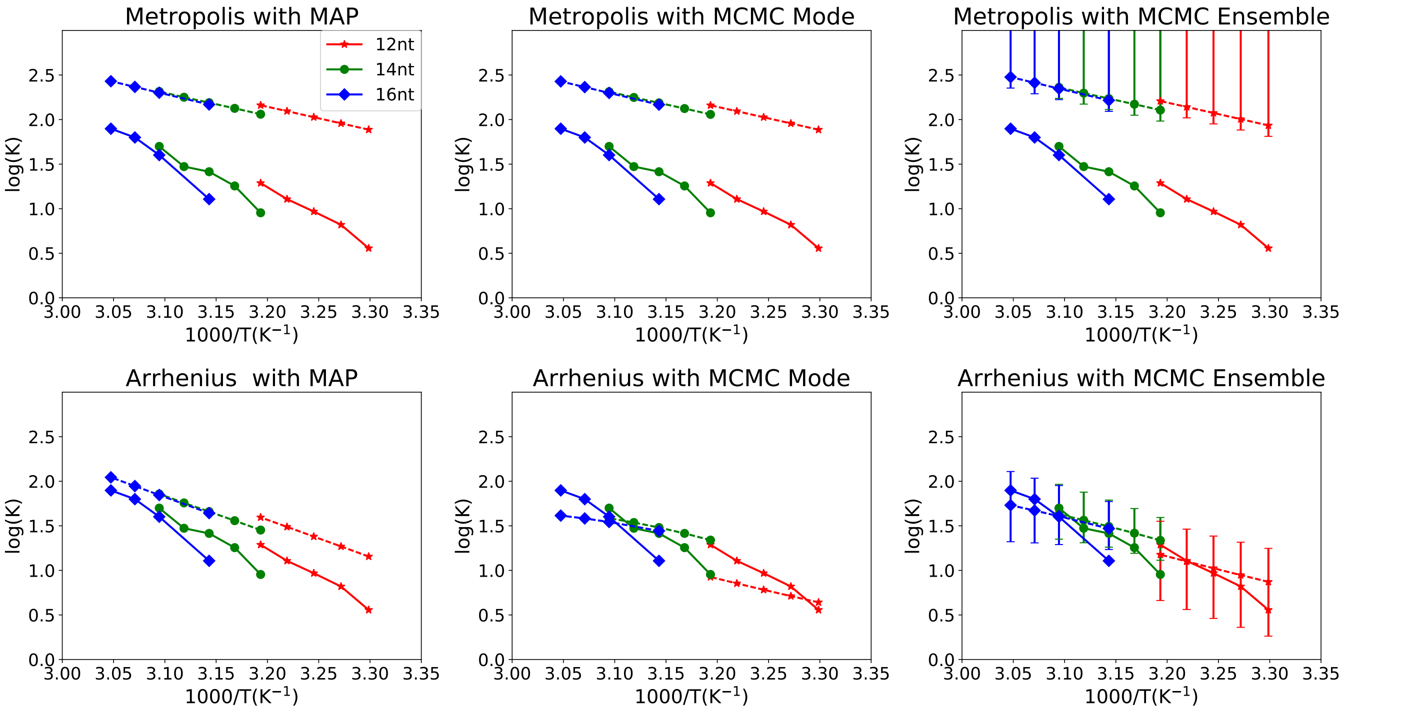



Fig. 12: Model fitting (dashed lines) of reaction rate constants (y axis) for toehold-mediated 3-way strand displacement, experimental data (solid lines) from Fig. 6 of Reynaldo et al. [8]. 12nt, 14nt, and 16nt are variations in the length of the strand.

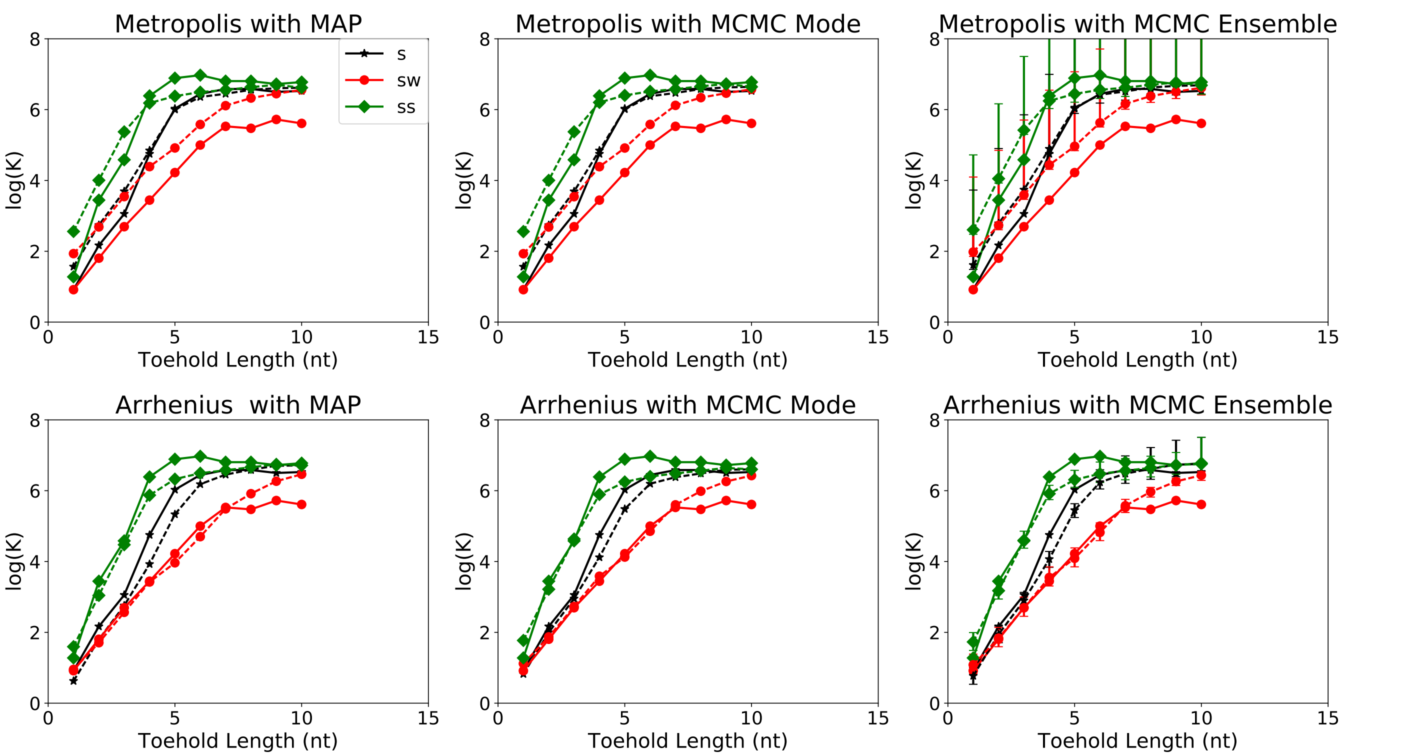



Fig. 13: Model fitting (dashed lines) of reaction rate constants (y axis) for toehold-mediated 3-way strand displacement, experimental data (solid lines) from Fig. 3b of Zhang and Winfree [9]. The toehold is varied between strong (ss), regular (s) and weak (sw) binding strength by varying the G/C content of the toehold sequence.

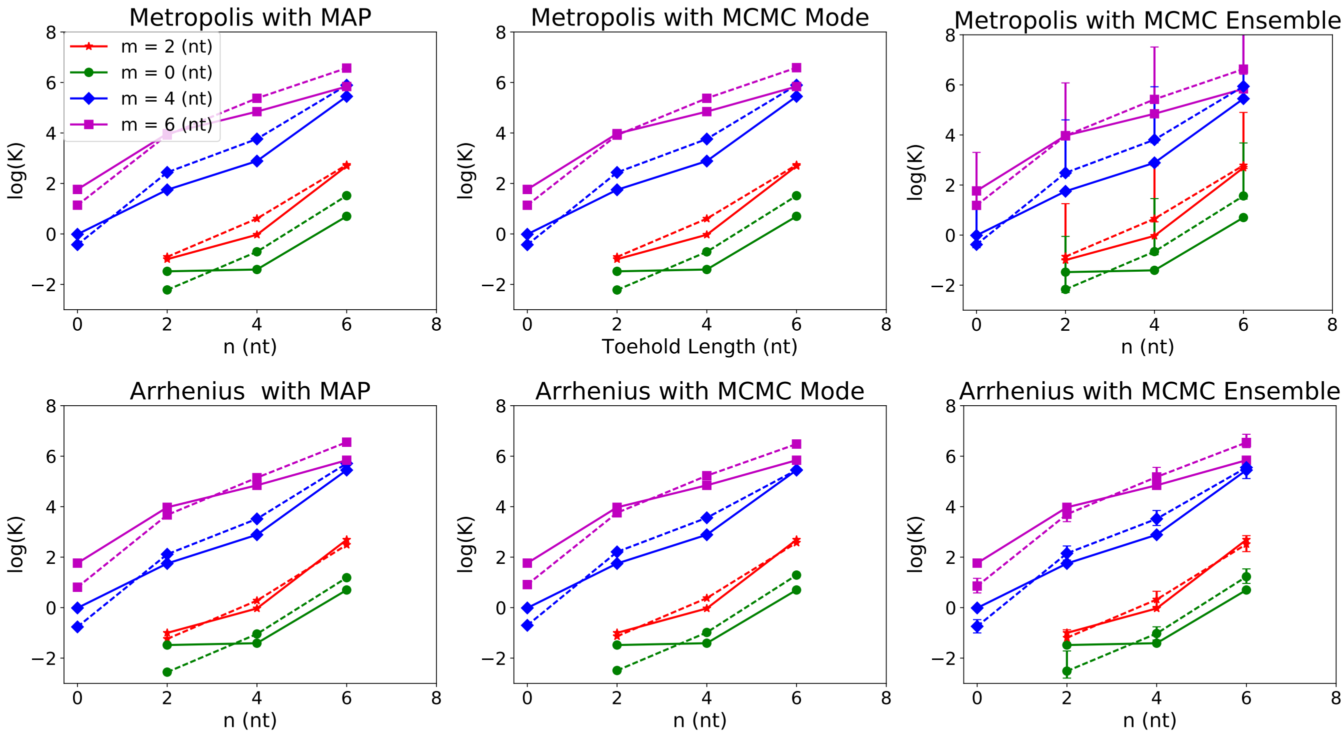



Fig. 14: Model fitting (dashed lines) of reaction rate constants (y axis) for toehold-mediated 4-way strand exchange, experimental data (solid lines) from Table 5.2 of Dabby [4]. m (shown on the legend) and n (shown on the x-axis) are variations in the length of the toehold domains (see Appendix B.5).

## D.2 Testing Set ($\mathcal{D}_{\text{test}}$)

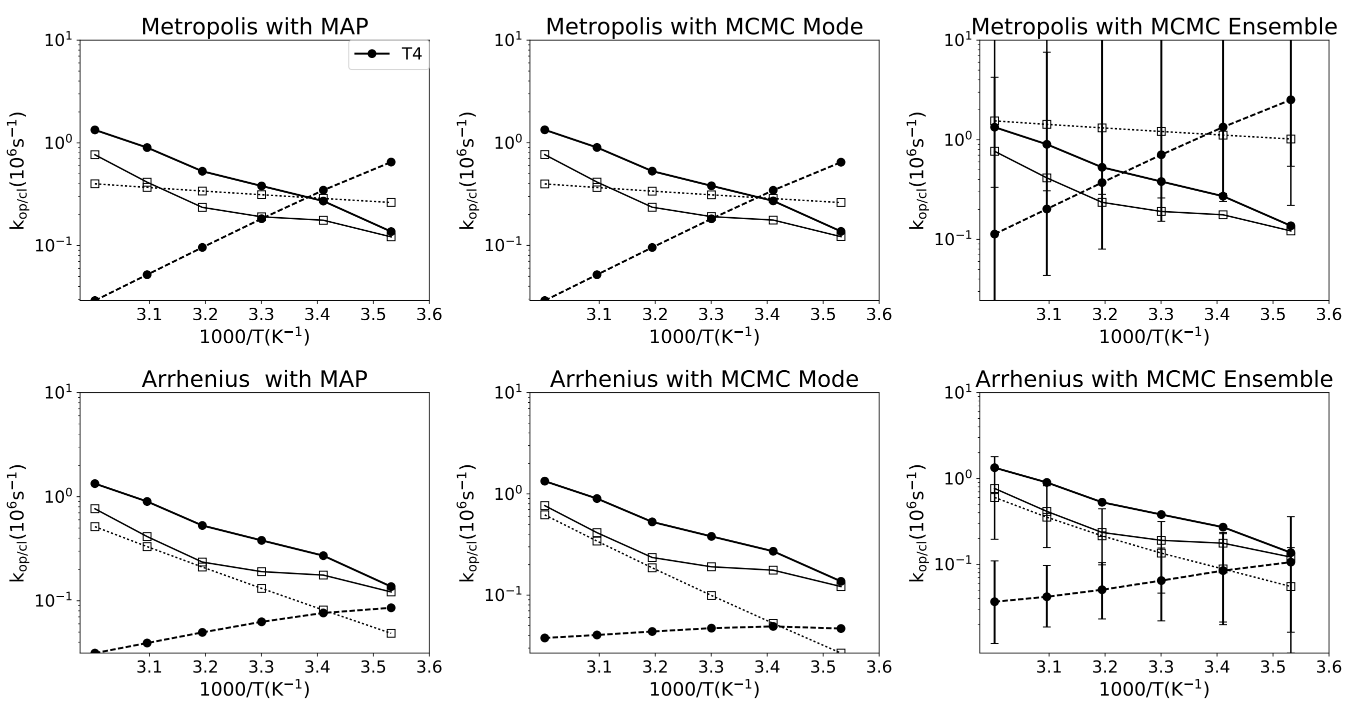



Fig. 15: Model predictions (dashed lines) of reaction rate constants (y axis) for hairpin closing (solid) and opening (open) with sequence $F$-$(dC)_2$-$(dT)_4$-$(dG)_2$, experimental data (solid lines) from Fig. 5a of Kim et al. [5].

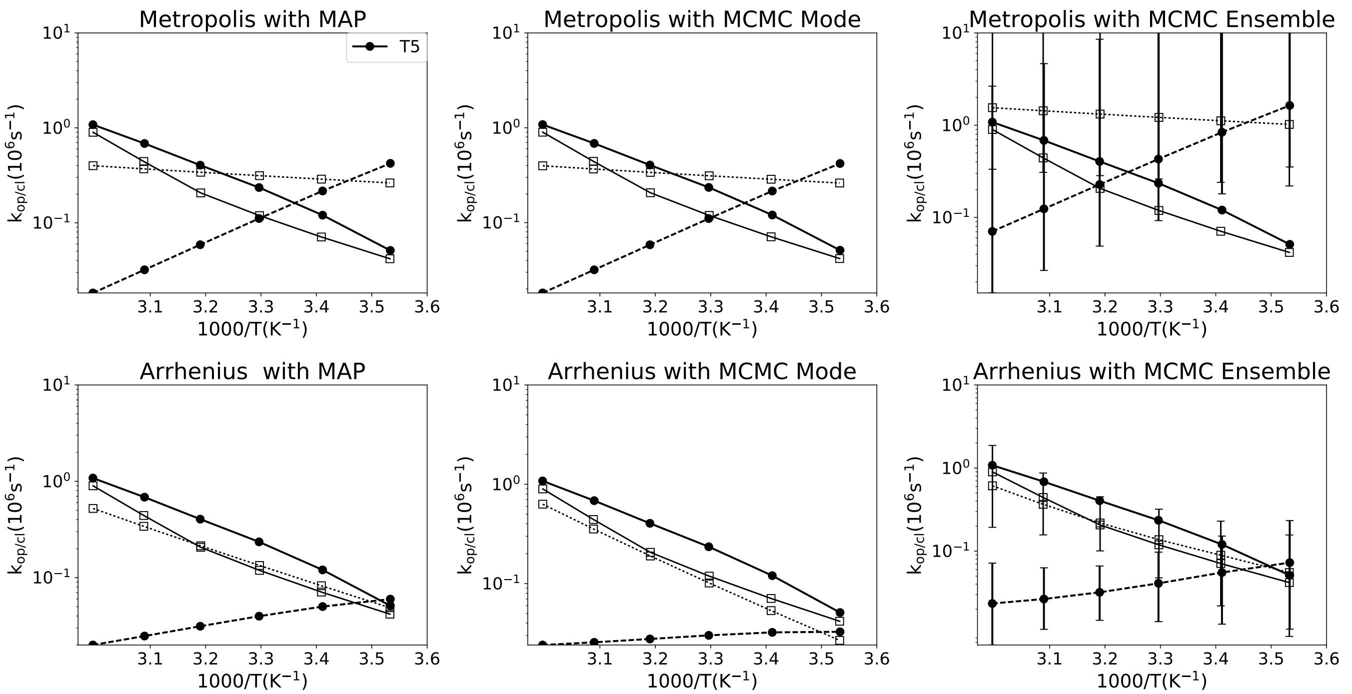



Fig. 16: Model predictions (dashed lines) of reaction rate constants (y axis) for hairpin closing (solid) and opening (open) with sequence $F$-$(dC)_2$-$(dT)_5$-$(dG)_2$, experimental data (solid lines) from Fig. 5b of Kim et al. [5].

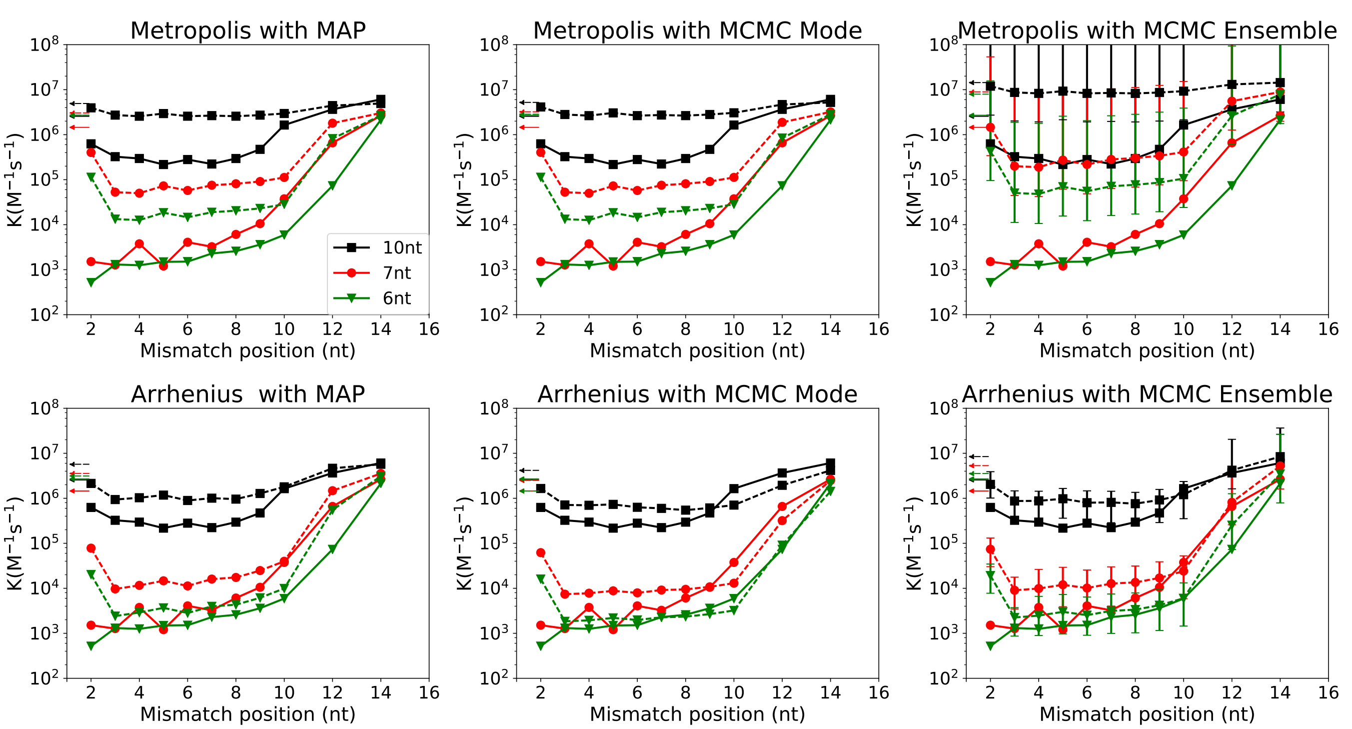



Fig. 17: Model predictions (dashed lines) of reaction rate constants (y axis) for toehold-mediated 3-way strand displacement with mismatches, experimental data (solid lines) from Fig. 2d of Machinek et al. [6]. For the MCMC ensemble method, error bars indicate the range (minimum to maximum) of predictions. Arrows indicate no mismatch. The mismatch in the invading strand affects the reaction rate. The length of the toehold domain is ten, seven, and six nucleotides long for ■, ●, and ▼, respectively.

## References

1. Altan-Bonnet, G., Libchaber, A., Krichevsky, O.: Bubble dynamics in double-stranded DNA. Physical Review Letters 90, 138101 (2003)
2. Bonnet, G.: Dynamics of DNA breathing and folding for molecular recognition and computation. Ph.D. thesis, Rockefeller University (2000)
3. Bonnet, G., Krichevsky, O., Libchaber, A.: Kinetics of conformational fluctuations in DNA hairpin-loops. Proceedings of the National Academy of Sciences 95(15), 8602–8606 (1998)
4. Dabby, N.L.: Synthetic molecular machines for active self-assembly: prototype algorithms, designs, and experimental study. Ph.D. thesis, California Institute of Technology (2013)
5. Kim, J., Doose, S., Neuweiler, H., Sauer, M.: The initial step of DNA hairpin folding: a kinetic analysis using fluorescence correlation spectroscopy. Nucleic Acids Research 34, 2516–2527 (2006)
6. Machinek, R.R., Ouldridge, T.E., Haley, N.E., Bath, J., Turberfield, A.J.: Programmable energy landscapes for kinetic control of DNA strand displacement. Nature Communications 5 (2014)
7. Morrison, L.E., Stols, L.M.: Sensitive fluorescence-based thermodynamic and kinetic measurements of DNA hybridization in solution. Biochemistry 32, 3095–3104 (1993)
8. Reynaldo, L.P., Vologodskii, A.V., Neri, B.P., Lyamichev, V.I.: The kinetics of oligonucleotide replacements. Journal of Molecular Biology 297, 511–520 (2000)
9. Zhang, D.Y., Winfree, E.: Control of DNA strand displacement kinetics using toehold exchange. Journal of the American Chemical Society 131, 17303–17314 (2009)