1 Elongation of DNA in MspA

1.1 Stretching Measurement

We hypothesize that the observed voltage-dependent shift in DNA position relative to MspA is due primarily to the elongation of the section of ssDNA between the enzyme and the pore constriction in response to the force generated by the applied voltage. To confirm that DNA stretching is the main effect responsible for the position shift and that other effects (i.e. Brownian motion of the enzyme above MspA or deformation within the enzyme or pore under force) are less important, we compare our shift vs. voltage data to the extensible Freely Jointed Chain (ex-FJC) model of ssDNA elongation in response to force. The ex-FJC is an experimentally validated model (Smith et al. Science 1992) of the elastic response of ssDNA to applied force which predicts the average end-to-end distance of the DNA (x) as a function of the force (F) applied to one end as

$$x = L_c(\coth(\frac{Fb}{k_BT}) - \frac{k_BT}{Fb})(1 + \frac{F}{S}) \tag{1}$$

where L_c is the DNA contour length, k_B is the Boltzmann constant, T is the temperature, b is the Kuhn length of ssDNA, and S is the stretching modulus of ssDNA.

In the high force regime in which we operate our variable-voltage experiments $Fb >> k_BT$, so the coth term can be well-approximated as identically equal to 1. With this approximation, the force-extension relation simplifies to

$$x = L_c(1 - \frac{k_BT}{Fb})(1 + \frac{F}{S}) \tag{2}$$

The Kuhn length of ssDNA is known to depend upon salt concentration. From Bosco *et al.* (Bosco et al. Nucleic Acids Res. 2014), we expect a Kuhn length of around 1.50 nm for the 400 mM KCl conditions in our variable-voltage experiments. We take a reasonable value of the stretching modulus S to be 800 pN (Bosco et al. Nucleic Acids Res. 2014).

Following the analysis in Derrington et al. (Nat. Biotechnol. 2015), we observe that in our system the end-to-end extension x is fixed as the distance between the constriction and the point where the DNA is anchored within the enzyme. With x fixed, it is the contour length L_c that changes with applied force. Assuming that the force on the DNA is proportional to the applied voltage as $F = \alpha V$ (α some proportionality constant) gives

$$x = L_c(1 - \frac{k_BT}{\alpha Vb})(1 + \frac{\alpha V}{S}) \tag{3}$$

Changing the applied voltage from V to βV will change the contour length of the DNA within the pore from L_c to ωL_c:

$$x = \omega L_c(1 - \frac{k_BT}{\beta\alpha Vb})(1 + \frac{\beta\alpha V}{S}) \tag{4}$$

Here, the elongation ratio ω is the ratio between the contour length of DNA in the pore at the two voltages V and βV. Solving equations 3 and 4 for ω gives us a model predicting the elongation ratio ω as a function of the voltage ratio β as

$$\omega_{model} = \beta[\frac{(b\alpha V - k_BT)(S + \alpha V)}{(b\beta\alpha V - k_BT)(S + \beta\alpha V)}] \tag{5}$$

We compare this ω_{model} to the measured elongation ratio results (ω_{meas}) as a function of voltage. The measured elongation ratio is calculated from the position shift data as

$$\omega_{meas}(\beta) = \frac{N_{ref} + \delta(\beta)}{N_{ref}} \tag{6}$$

where δ is the measured position shift from 180 mV (Fig S1) and N_{ref} is the number of nucleotides between the last hold point within the enzyme and the constriction at the reference voltage of 180 mV. Position shift is calculated as described in Supplemental Note 1.2.

From Bhattacharya *et al.* (Bhattacharya et al. ACS Nano, 2016), we estimate $N_{ref} = 12$ nt. Fitting equation 5 to our data shows that a single parameter fit with $\alpha = 1.47 \pm 0.08 \frac{e^-}{nm}$ describes the data well (Fig S1). Uncertainties here are based on uncertainties in the DNA shift at different voltages and an assumed 0.5 nt uncertainty in N_{ref}. As the position shift can be well modeled by a reasonable single parameter model of DNA elongation, we are confident attributing the shift observations to this effect.

The fitted α parameter corresponds to a force of $\sim$42 pN at 180 mV. This force estimate has larger uncertainties than the uncertainty in α as the estimate is critically dependent on the choices of ex-FJC parameters and ignores secondary effects contributing to the stretching. Potentially relevant secondary effects not accounted for by the ex-FJC model could include effects from the confinement of the ssDNA within the pore vestibule, voltage dependence fo the position of the enzyme relative to MspA, and voltage-induced deformation of the enzyme or the pore.

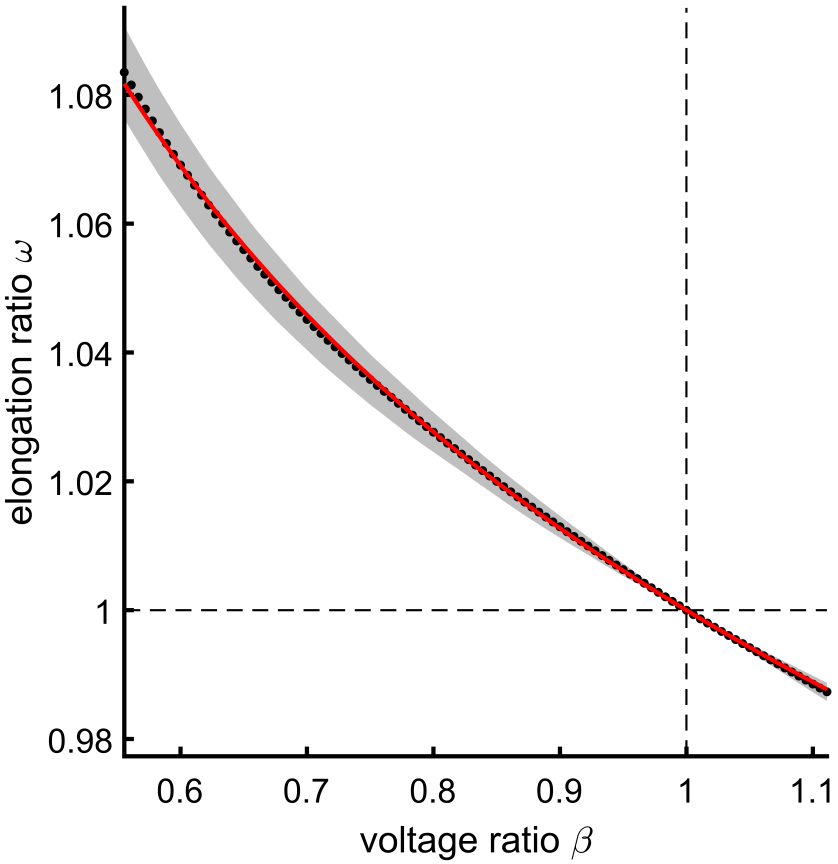


Figure S 1: DNA stretching in MspA. The fractional DNA elongation relative to DNA position at 180 mV (ω) is plotted (black points) relative to the fractional applied voltage relative to 180 mV (β). The gray shading shows the 1 standard deviation interval. The uncertainty was determined using 10 bootstraps, each consisting of a random distinct subset of the 18 read consensus used for the stretching measurement. The DNA stretch was calculated individually for each bootstrapped consensus, and the overall standard deviation in the stretch was taken as the standard deviation over these bootstrapped measurements. The red line shows the ex-FJC fit with $\alpha = 1.47 \pm 0.8 \frac{e^-}{nm}$.

1.2 Position Shift Calculation

The position shift calculation was done using the consensus conductance measurements for a known DNA sequence (stretching read strand, Table S2). The consensus conductances were determined from 18 separate measurements of the DNA sequence using standard variable-voltage sequence conditions (Table S3).

The position shift between two voltages V_1 and V_2 is determined as the shift that best places the conductance profile measurements from each of the two voltages along a single spline. The shift is calculated between consecutive voltages (e.g. 101 mV to 102 mV) because the conductance profile changes over large changes in voltages. As the DNA is more elongated at higher voltages, the conductance measured at each DNA position is a function of a smaller number of nucleotides. Consequently, features of the conductance profile that are blurred out at low voltages due to averaging can become apparent at higher voltages. This affect is accounted for during our normalization procedure, discussed in Supplemental Note 4.

The shift best placing the conductance measurements from the two voltages along a single spline is calculated as follows. After first normalizing the conductances (Supplemental Note 4), we have the conductance profiles for each of the two voltages, G_1 and G_2. We then calculate cubic spline interpolations (spG_1 and spG_2) to both of the transformed current profiles. The two splines are shifted left and right relative to each other in increments of $\frac{1}{1000}^{th}$ nt. For each shift position ϕ, we calculate a match score $\mathcal{M}$ by taking the sum-square difference between the two splines for the given shift, and dividing out by the total number of compared points:

$$\mathcal{M}(\phi) = \frac{\sum_{i=1}^{N_{pts}} (spG_1^{(i)} - spG_2^{(i+\phi)})^2}{N_{pts} - \phi} \tag{7}$$

The shift ϕ_0 giving the best (smallest) match score gives us the shift that makes the two splines most similar. This ϕ_0 is taken as the position shift between V_1 and V_2. The uncertainty $\delta\phi$ was determined by measuring the shift using subsets of the entire dataset. From the longer measured sequence, we extracted 9 distinct subsets, and measured the shift function for each. We took $\delta\phi$ ad the standard deviation of these separate measurements.

2 Hel308 Step Durations

The distribution of step durations for Hel308 in our sequencing experimental conditions is shown in Fig S2. We need 3 complete voltage cycles (15 ms) to accurately estimate the covariance of the 3 principal components for a state (Supplemental Note 3), otherwise we must take a default value for the covariance for the state. Over 90% of states are long enough to accurately estimate the covariance (blue). A small fraction (<10%, red) are too short and are assigned a default covariance value.

States shorter than a full voltage cycle (5 ms) will not be detected by the change-point detection algorithm and ultimately manifest as skips in the final data. However, for these conditions such short states should make up only a small minority of the total Hel308 states. Long term, it will be desirable to use a faster enzyme (or experimental conditions in which Hel308 steps faster) in to increase throughput and decrease the per-read time. To accomplish this in variable-voltage setting, we will need to increase the variable-voltage cycle frequency. The primary limitation to increasing the cycle frequency is that the capacitive current (Supplemental Note 12) increases with increasing rate of change in the voltage. If the capacitive current becomes too large, it could rail the amplifier (rail is ± 1 nA), resulting in a loss of signal.

This issue can be addressed in multiple ways. First, reducing bilayer capacitance is a straightforward way of reducing the capacitive current. A commercial sequencing device will need to be dramatically miniaturized compared to the experiments we run in our lab, and will require an automated method of bilayer formation. These automatically-formed bilayers can in principle be much smaller (lower capacitance) than the hand-painted bilayers used in this work. Second, we have some room to reduce the range of the voltage sweep without compromising the efficacy of the variable-voltage signal. The 100-200 mV swing currently in use gives us more than enough state-to-state overlap to identify and correct enzyme missteps. A smaller voltage range should still provide adequate overlap, while reducing the rate of voltage change and thus the size of the capacitive current. Finally, on-line methods can be used to compensate for the capacitive signal. The injection of an in-phase square wave current into the system to counteract the square wave contribution of the capacitance would allow us to take the variable-voltage method to much higher cycle frequencies.

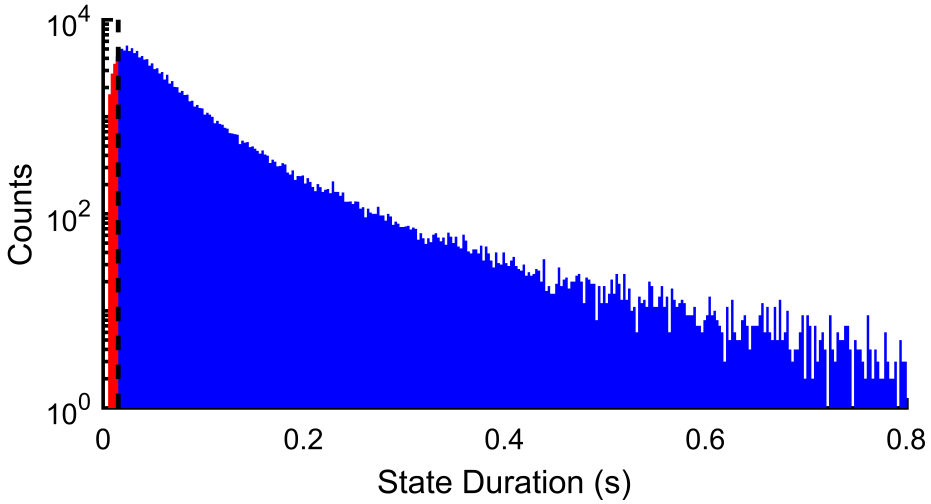


Figure S 2: Hel308 step durations in variable-voltage sequencing conditions. Marked in red are states too short to accurately estimate the covariance of the 3 principal components. States of sufficient length for this estimation are marked in blue.

3 Feature Extraction

Following change point detection (Supplemental Note 11) and capacitance compensation (Supplemental Note 12), the sequencing data is in the form of a series of time-ordered ionic current-vs-voltage ($I-V$) curves. These $I-V$ curves are converted to conductance-vs-voltage ($G-V$) curves by dividing out the voltage from the ionic current. Going forward from here, variable-voltage sequencing analysis is conducted using conductance in lieu of voltage.

Each $G-V$ curve characterizes one enzyme step along the DNA, as determined during change point detection. Each $G-V$ curve is made up of 101 conductance measurements taken at voltages between 110 and 190 mV, represented by a 101-dimensional feature (column) vector, $\boldsymbol{g}$. The sampled voltage points are chosen so that the shift in DNA registration between each consecutive pair of points is uniform–we sample the conductance uniformly over DNA position, but non-uniformly over voltage. Uniform sampling over position ensures maximum independence between the sampled conductances.

The 101 elements (features) in $\boldsymbol{g}$ are largely not independent. Many of the features provide redundant information and serve only to introduce noise into our characterization of the states. We used principal component analysis (PCA) to reduce the dimensionality of the feature vectors describing each state. PCA revealed that the top 3 principal components explain nearly 98% of the variance between $G-V$ curves. In light of this, we reduce the dimensionality of the feature vectors from 101 to 3 by replacing the 101 sampled conductances with the coefficients of the top 3 principal components (Fig S3).

We calculate the reduced 3-dimensional feature vector $\boldsymbol{p}_i$ for state i as

$$\boldsymbol{p}_i = [\boldsymbol{\pi}_1; \boldsymbol{\pi}_2; \boldsymbol{\pi}_3]^T * \boldsymbol{g}_i \tag{8}$$

where $\boldsymbol{\pi}_j$ is the j^{th} principal component (column) vector. This dimensional reduction allows us to satisfactorily characterize each state while dramatically de-noising our description (Fig S4).

Additionally, we are much better able to estimate the covariance amongst the features for these smaller feature vectors. Each full voltage cycle j (200 Hz) completed during a given state i provides two measurements $\boldsymbol{g}_i^j$ of the state's conductance feature vector $\boldsymbol{g}_i$, one from the voltage up-swing, one from the voltage down-swing. Similarly, we can treat the 3 principal component coefficients for each half cycle $\boldsymbol{p}_i^j$ as distinct measurements of the overall principal component feature vector $\boldsymbol{p}_i$. Given t half-cycle measurements, we can estimate the covariance in the state's conductance ($\boldsymbol{\Sigma}_i^{\boldsymbol{g}}$) and principal component features ($\boldsymbol{\Sigma}_i^{\boldsymbol{p}}$) as

$$\boldsymbol{\Sigma}_i^{\boldsymbol{g}} = \mathop{\mathbb{E}}_{j \in 1:t}[(\boldsymbol{g}_i^j - \mathop{\mathbb{E}}_{j \in 1:t}[\boldsymbol{g}_i^j])(\boldsymbol{g}_i^j - \mathop{\mathbb{E}}_{j \in 1:t}[\boldsymbol{g}_i^j])^T] \tag{9}$$

and

$$\boldsymbol{\Sigma}_i^{\boldsymbol{p}} = \mathop{\mathbb{E}}_{j \in 1:t}[(\boldsymbol{p}_i^j - \mathop{\mathbb{E}}_{j \in 1:t}[\boldsymbol{p}_i^j])(\boldsymbol{p}_i^j - \mathop{\mathbb{E}}_{j \in 1:t}[\boldsymbol{p}_i^j])^T] \tag{10}$$

The estimators $\boldsymbol{\Sigma}_i^{\boldsymbol{g},\boldsymbol{p}}$ are only well defined if we have at least as many measurements as there are independent entries elements in the covariance matrix. As covariance matrices are symmetric, $\boldsymbol{\Sigma}_i^{\boldsymbol{g},\boldsymbol{p}}$ has $\frac{d}{2} * (d-1)$ independent entries, where d is the dimensionality of the associated $\boldsymbol{g}$ or $\boldsymbol{p}$ feature vector. So, in order to get a good estimate of the covariance of the conductance features $\boldsymbol{g}_i$ for a given state, we require 5050 half-cycle measurements, representing over 12.5 seconds spent in that state–far longer than the typical state duration. Conversely, we can estimate the covariance of the principal component features $\boldsymbol{p}_i$ from just 6 half-cycle measurements, or 15 ms of data. Using the principal component dimensional reduction, we are thus able to accurately estimate the feature covariance for nearly all ($> 90\%$) of the observed states (Supplemental Note 2). For the $< 10\%$ of states for which the covariance is not well estimated, we fill in the covariance with the 90th percentile largest (by value of the determinant) well-estimated covariance.

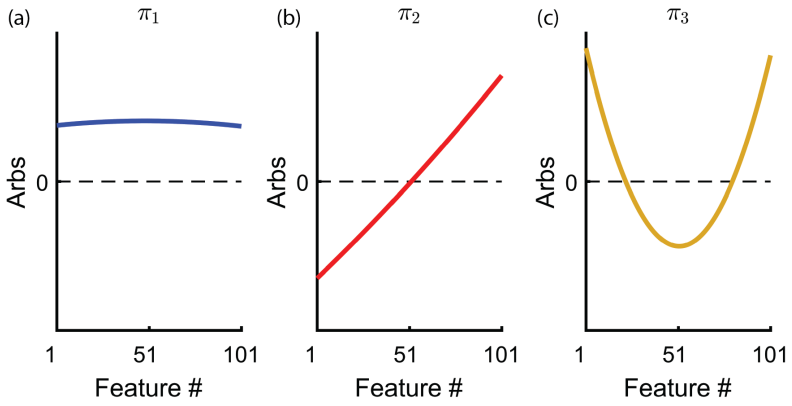


Figure S 3: Principal component vectors for feature extraction. **(a)**, **(b)**, and **(c)** show the first, second, and third principal component vectors for the variable voltage data, respectively. Linear combinations of these three vectors can describe all observed conductance vs. DNA position states. The three vectors roughly represent an offset (a), slope (b), and curvature (c) and thus primarily describe the states as quadratic curves.

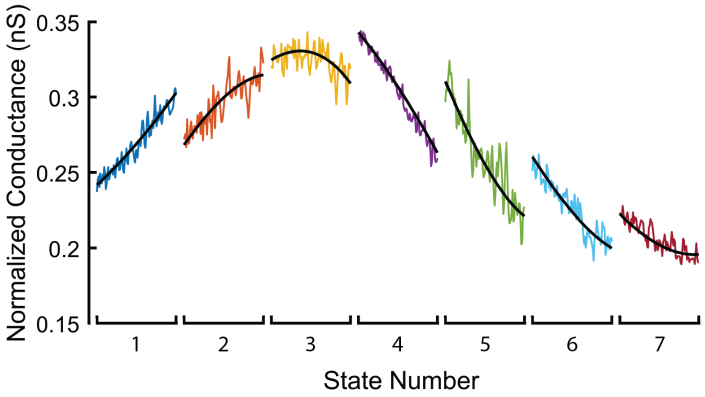


Figure S 4: Principal component description of conductance states. Linear combinations of the 3 principal components (black curves) satisfactorily describe the 101-dimensional conductance states (colored curves). The description preserves the state shape while discarding parameters describing only noise.

4 Conductance Normalization

Following capacitance compensation (Supplemental Note 12), the response to the changing voltage in the variable-voltage signal retains a nuisance component in addition to the DNA-position-dependent portion of the signal (which is the signal we are ultimately interested in for sequencing). This complicating component, dominated by the intrinsically non-ohmic character of the pore conductance when blockaded by a charged molecule, is mostly additive with the DNA-dependent portion of the signal, but is not itself affected by DNA position. We need to remove this portion of the signal in order to arrive at the purely DNA-position-dependent conductance signal that changes smoothly as a function of DNA position. We refer to the process of removing the non-position-dependent portion of the conductance as "normalization" and refer to the final smooth conductance profile as the "normalized" conductance.

To find the normalized conductance curve $\not{g}_i(V)$ of a state i, we take an average of the conductance at each voltage $g_j(V)$ over each state in a read ($j \in 1:N$ where N is the number of states), and subtract this mean conductance from the measured conductance from each state:

$$\not{g}_i(V) = g_i(V) - \frac{1}{N}\sum_{j=1}^{N} g_j(V) \tag{11}$$

In effect, this process estimates the position-independent contribution to each state's conductance curve as the portion of the curve found on average in all of the states, then removes this shared component.

Following this simple normalization, we require a further correction to fully realize the continuous conductance profile. We observe a "fraying" of the segments in the continuous curve (Fig S5). That is, at high voltage, states with normalized conductances well above the mean tend to exaggerated and take higher values than what is necessary for the curve to be continuous. Likewise, states with conductances below the mean take values lower than would be expected for the continuous curve. We attribute this effect to the stretching of the DNA at high voltages. The additional elongation of the DNA at higher voltage means that fewer bases on average will contribute to the instantaneous conductance through the pore, as fewer bases spend time near the constriction. So, the DNA-dependent signal is dominated further by a few bases at high voltage than low voltage. This effect serves to exaggerate the peaks and troughs in the normalized signal.

To correct for this effect, we note that there should be no correlation between the applied voltage and the DNA-dependent conductance. Therefore, we correct to the first order by fitting a linear model to each reduced conductance curve, obtaining from each $\not{g}_i$ a slope m_i. These slopes are then linearly fit to the voltage means of the normalized conductances,

$$\langle \not{g}_i \rangle = \frac{1}{N_V}\sum_{j=1}^{N_V} \not{g}_i(V_j) \tag{12}$$

where N_V is the number of voltages measured in each state (= 101). This fit with slope α represents the magnitude of the linear voltage response as a function of conductance. Subtracting this bias, we obtain the final normalized conductance which represents the DNA-dependent signal that will ultimately be used:

$$\not{g}_i(V) = g_i(V) - \frac{1}{N}\sum_{j=1}^{N} g_j(V) - \alpha V\left(g_i(V) - \frac{1}{N}\sum_{j=1}^{N} g_j(V)\right) \tag{13}$$

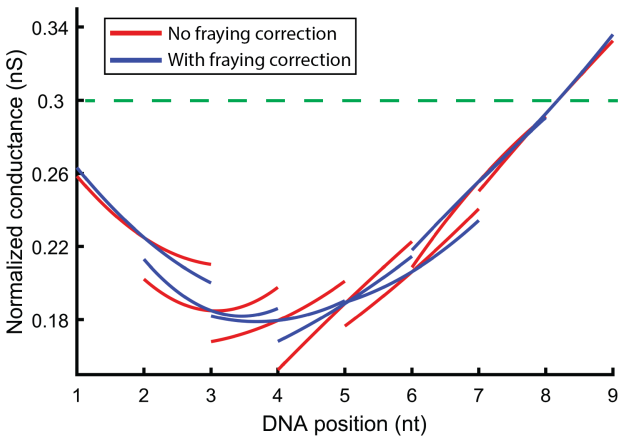


Figure S 5: Fraying correction. The linear fray correction accounts for the exaggerated effects of a few bases on the conductance at high voltage. The initial mean-only normalization (red) demonstrates systematic discontinuities around peaks (not shown) and troughs (shown here) where the high-voltage points (left on each segment) are too high (peaks) or low (troughs). The dashed green line shows the overall average conductance (for the whole read, of which only a short section is shown). The fray correction accounts for this and generates a more-continuous conductance profile (blue).

5 Variable-Voltage State Filtering

One of the primary advantages of the variable-voltage method is that it allows us to determine the correct ordering of the observed states prior to sequencing. We determine the best ordering of observed states via a three stage "state filtering" process prior to sequencing. The three stages are termed the removal filter (Supplemental Note 5.1), the recombination filter (Supplemental Note 5.2), and the reorder filter (Supplemental Note 5.3). Each stage of state filtering aims to eliminate a specific error mode common to the data.

5.1 Removal Filter

The goal of the removal filter is to find and remove states that are not informative of the DNA sequence moving through the pore. These uninformative "bad" states are common in both constant-voltage and variable-voltage sequencing data and can arise from myriad sources. Common sources of "bad" states include:

1. Pore Gating: Protein pores such as MspA are well known to exhibit transient stochastic changes in their conductance, referred to as gating. Gating can occur during DNA translocation, resulting in an abrupt drop in the observed conductance of the observed states for the duration of the gating event. Although DNA translocation continues during the gating event, the conductance states measured in this time period will not match the ionic current-to-sequence model states of the translocating DNA due to the low overall conductance.

2. Conductance Spikes: We observe occasional transient spikes up in the conductance through the pore during DNA translocation events. These spikes may be attributable to brief openings of alternative conducting pathways through the bilayer. Regardless of origin, these spike states are not indicative of the translocating DNA sequence, and are not observed at the same DNA sequence position when comparing multiple translocation events of the same DNA sequence.

3. Flickers: As discussed in Supplemental Note 14, we observe short drops in conductance within enzyme states termed "flickers" in both constant- and variable-voltage data. These drops in conductance are distinct from pore gating as they are far shorter-lived and return to the state that preceded them.

4. Over-called States: The change point detection algorithm (Supplemental Note 11) occasionally calls too many transitions, partitioning a single state into multiple. This can be caused by spontaneous changes in the electronic noise, flickers occuring faster than the variable-voltage cycling frequency, or other transient effects distorting the signal. Frequently, the over-called states exhibit higher noise than the true state. These high-noise over-called states are discarded for sequencing.

The removal filter works by iteratively assigning a "bad state probability" P_{bad} to each state in the event, then removing those where P_{bad} exceeds some threshold value for removal T_{remove}. This process is repeated until no more states are removed (algorithm S1). The process is iterated because P^i_{bad}, the bad state probability for a given state i, is a function not only of the state itself, but also of its flanking states $i+1$ and $i-1$. So, P^i_{bad} can change following the first round of removal if either of its flanking states were removed. As there is no state preceding the first state or following the last state, P_{bad} cannot be evaluated for these two cases. To cope with this, the first and last state are kept as "good" until the final iteration of removal, at which time they are discarded.

The P_{bad} values are calculated as follows. States are first evaluated using a support vector machine (SVM) with a quadratic kernel classifying between "good" states (those to be kept for sequencing) and "bad" states (those to be removed). The SVM takes as input 12-dimensional feature vectors for each state. The composition of the feature vector for state i is as follows:
Features 1-3 are the 3 principal component coefficients (Supplemental Note 3) for the previous state, $i-1$.
Features 4-6 are the 3 principal component coefficients for the state itself, i.
Features 7-9 are the 3 principal component coefficients for the subsequent state, $i+1$.
The first 9 features serve to quantify how continuous or discontinuous the state is with its neighbors. States that are discontinuous with both the previous and subsequent states are more likely to be "bad".

Algorithm S 1 Removal Filter

assume we start with N states $\{\boldsymbol{x}^i\}_{i\in 1:N}$
$stop \leftarrow false$
while $\sim stop$ **do**
    $anyremoved \leftarrow false$
    calculate the SVM feature vectors $\{\boldsymbol{\xi}^i\}_{i\in 2:N-1}$ **from** the states $\{\boldsymbol{x}^i\}_{i\in 1:N}$
    calculate the bad state probabilities $\{P^i_{bad}\}_{i\in 2:N-1}$ **from** the SVM feature vectors $\{\boldsymbol{\xi}^i\}_{i\in 2:N-1}$
    for $j \in 2 : N-1$ **do**
        if $P^i_{bad} > T_{thresh}$ **then**
            remove $\boldsymbol{x}^j$ from $\{\boldsymbol{x}^i\}$ ▷ Remove states above the removal threshold
            $anyremoved \leftarrow true$ ▷ Stop iterating if nothing is removed
        end if
    end for
    if $\sim anyremoved$ **then**
        $stop \leftarrow true$
    end if
end while
remove $\boldsymbol{x}^1$ and $\boldsymbol{x}^N$ from $\{\boldsymbol{x}^i\}$ ▷ Remove the first and last states

Feature 10 is the value of the single conductance measurement in the state's conductance curve that most deviates from the overall mean conductance in the event. This helps to identify levels with short, extreme deviations from typical conductance values. Such deviations can indicate that a noise spike occurred during the state, likely causing an over-calling during change point detection.
Feature 11 is the average mean square difference between the state's 101-dimensional measured conductance curve and its 3-dimension principal component description. This quantifies how well the state is described by the principal components. States poorly described by the principal components are more likely to be "bad".
Feature 12 is the score of these state's best match against the 6-mer model (Supplemental Note 7). States that do not have any high scoring match within the 6-mer model are unlikely to represent good measurements of the DNA's conductance profile and should be labeled "bad".

To train the SVM, we hand-labeled states taken from the reads used for map building (Supplemental Note 7) as either "good" or "bad". The SVM was trained on a sample of 800 labeled "good" states and 800 labeled "bad" states. We then passed a hold-out validation set consisting of 400 labeled "good" and 400 labeled "bad" states to the trained SVM. The validation set showed that the SVM correctly classifies 97.3% of "good" states, 86.5% of "bad" states, and 91.9% of validation states overall.

To generate the "bad state" probabilities P_{bad}, we looked at the scores output by the SVM, rather than the labels. The SVM score $\mathbb{S}$ of a state is the distance of that state's SVM feature vector from the decision boundary (Fig S6a). This score serves as a proxy for how good (negative scores) or bad (positive scores) a state is. We want to assign higher P_{bad} to states with higher scores. We do this by plotting the true state labels (0 for good, 1 for bad) as a function of the state scores $\mathbb{S}$ (Fig S6b). These data are then fit by the logit function

$$f(\mathbb{S}|\alpha,\beta) = \frac{1}{1-e^{-(\alpha\mathbb{S}+\beta)}} \tag{14}$$

using a global likelihood maximization fit (Fig 6).

Together, the SVM and the fit logit function give us a way to calculate P^i_{bad} for any state i. First, the 12 features are evaluated for this state. Then, the SVM is used to score the feature vector relative to the decision boundary, yielding a score $\mathbb{S}^i$. Finally, we evaluate $f(\mathbb{S}^i|\alpha,\beta)$, yielding P^i_{bad} for the state.

5.2 Recombination Filter

The goal of the recombination filter is to find instances where multiple observed states represent repeated measurements of the same DNA position. Repeated state measurements can arise from two potential sources.

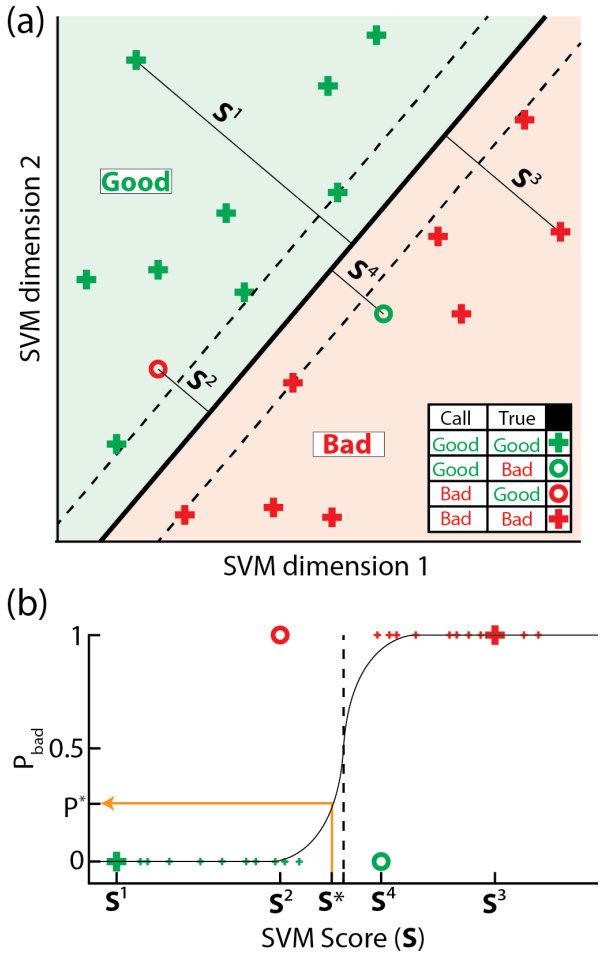


Figure S 6: Converting SVM outputs to P_{bad} probabilities. **(a)** In this classifier (contrived data), points occupying the space (shaded green) above the decision boundary (solid black line) are classified as good while those below (shaded red) are classified as bad. Points marked with a "plus" are classified correctly, while "circles" are classified incorrectly. Green markers denote truly good states and red markers denote truly bad states. Each point i has an associated score $\mathbb{S}^i$, which is its distance from the decision boundary. **(b)** Each state in the validation set is plotted by its good (0) or bad (1) label as a function of its assigned SVM score $\mathbb{S}$. The dashed black vertical line is at $\mathbb{S} = 0$, representing points lying exactly on the decision boundary. The solid black curve shows the logit function fit to the validation states using a global likelihood maximization procedure. The SVM scores $\mathbb{S}$ are converted into probabilities that the state is bad using a logit function. During removal filtering, an unknown state is assigned a score $\mathbb{S}^*$ by the SVM. This score is then converted into a probability it is bad (P^*_{bad}) using the logit function (orange arrow).

First, over-called transitions during change point detection result in consecutive states representing the same DNA position. If these over-called states are not removed by the removal filter, they show up in this stage as "holds". The second source of duplicate states is enzyme missteps in which the enzyme moves backwards (in the 3' direction) along the DNA. These "back steps" result in non-consecutive duplicate states.

The recombination filter works by aligning an event against itself (self-alignment). Repeated states will match to their duplicates within the event nearly as well as they match to themselves. We conduct a Needleman–Wunsch-style alignment of the states $\{\boldsymbol{x}^i\}$ with themselves, $\mathbb{A}(\{\boldsymbol{x}^i\}, \{\boldsymbol{x}^i\})$ (algorithm S2). In this alignment, alignment of a state i to itself $\mathbb{A}(\boldsymbol{x}^i, \boldsymbol{x}^i)$ can be thought of as establishing $\boldsymbol{x}^i$ as a unique, previously unobserved state. Conversely, alignment of a state i to a different (previous) state j, $\mathbb{A}(\boldsymbol{x}^i, \boldsymbol{x}^j), i \neq j$ means that states i and j are repeated measurements of the same DNA position and should be recombined into a single state.

A state i will always match best with itself. However, we bias the alignment against aligning states to themselves by applying a "self-alignment penalty" P_{SA} to such cells in the alignment matrix (Fig S7). Statistically, the self-alignment penalty is a penalty for adding parameters (states) to our model of the observed event and the self-alignment penalty is thus taken $\frac{1}{2}$ the number of added parameters (3, for the 3 principal component coefficients characterizing each state (Supplemental Note 3)).

With these considerations, we conduct a Needleman-Wunsch-style alignment of the measured states against themselves with the following modifications. First, to reduce the computational load and avoid recombining distant states that may look similar but too far apart to represent a likely duplication, we limit ourselves to a fixed lookback distance L, where we only consider matches for state i within states $i - L$ to $i - 1$.

Secondly, we assign unique step-type probabilities at each transition based on the conductance curve overlap information. At each transition between two states m and n, we calculate the relative probabilities that the transition between the two occurred via a single half-nucleotide step (P_S^{mn}), skip (P_K^{mn}), backstep (P_B^{mn}), or hold (P_H^{mn}). To calculate these probabilities, we use an ensemble of 3 SVMs (quadratic kernel), $\boldsymbol{S}_{SK}$, $\boldsymbol{S}_{SB}$, and $\boldsymbol{S}_{SH}$. These three SVMs are all trained on labeled transitions generated from the ΦX-174 data used to build the 6-mer model (Supplemental Note 7.3.1). In the same manner as was described above for the good/bad SVM classifier (Supplemental Note 5.1), we first trained these classifiers to determine their decision boundary, then conducted a global likelihood maximization fit to tune a logit function (characterized by two parameters, α and β) to their output scores on a held-out validation set. This fit logit function allows us to convert the output scores from the SVMs (distances from the decision boundary) into probabilities. All three SVMs take as input a 6-dimensional feature vector composed of the 3 principal component coefficients of state m and from state n.

$\boldsymbol{S}_{SK}$ differentiates between steps and skips (88.7% correct on the validation set), $\boldsymbol{S}_{SB}$ differentiates between steps and backsteps (98.2% correct on the validation set), and $\boldsymbol{S}_{SH}$ differentiates between steps and holds (95.4% correct on the validation set). The scores of these SVMs ($\mathbb{S}_{SX}$, X one of K, B, H), converted to probabilities through their associated logit functions, give us relative likelihoods between the different step types. The relative likelihoods of a step vs. a skip between states m and n is given by

$$\frac{P_S^{mn}}{P_K^{mn}} = \frac{\text{logit}(\mathbb{S}_{SK}, \alpha_{SK}, \beta_{SK})}{1 - \text{logit}(\mathbb{S}_{SK}, \alpha_{SK}, \beta_{SK})}$$

with similar relations for step vs. back and step vs. hold. These three relations, along with the overall normalization condition that

$$P_S^{mn} + P_B^{mn} + P_H^{mn} + P_K^{mn} = 1$$

give us a system of four equations for the four unknowns, allowing us to solve for the various step type probabilities. Skips longer than two half-steps and backsteps longer than a single half-step backwards are treated as independent processes, with their probability given as the product of the correct number of P_K's or P_B's. For example, the probability of a backstep of 3 half-steps P_{B3} is given as

$$P_{B3} = P_B^3$$

In the language of affine probabilities, the extension probability is set to be equal to the basic probability,

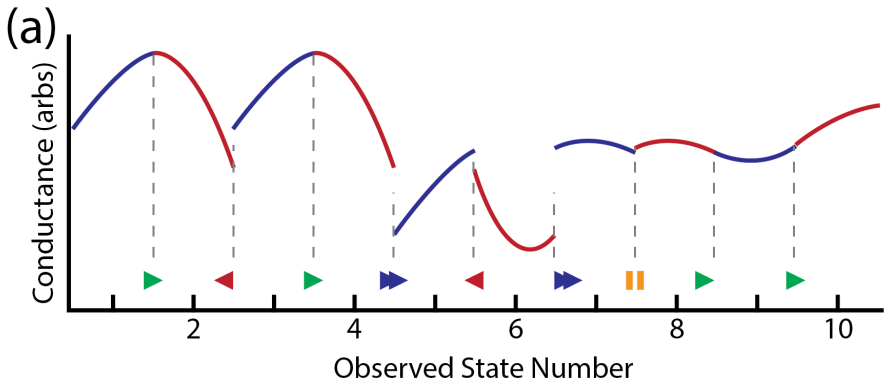
$$P_{B+} = P_B$$

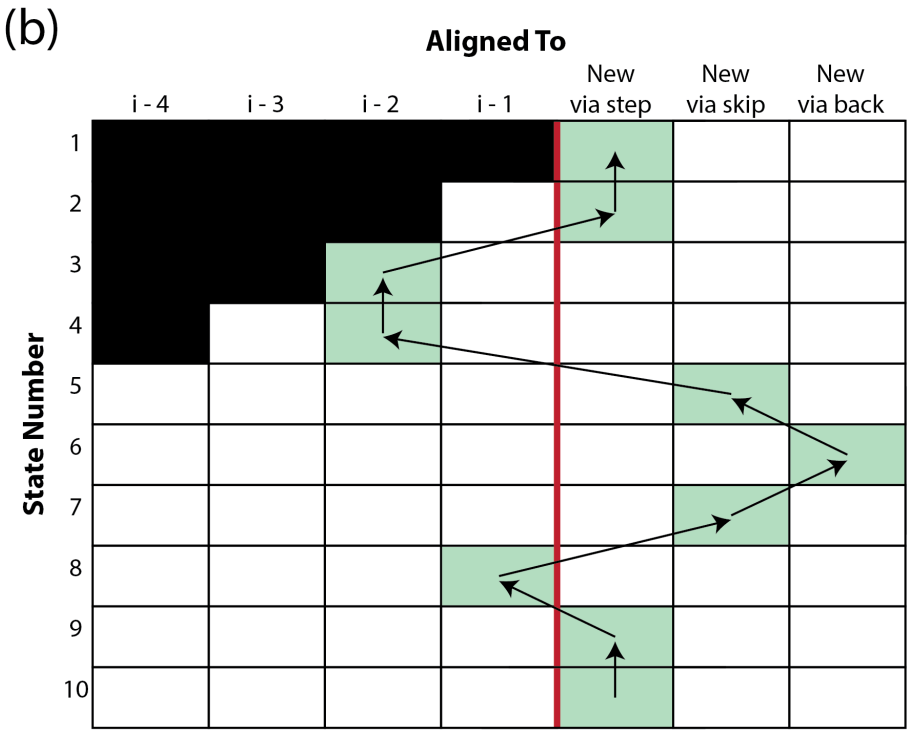
We can enter a previously unmeasured (new) state through one of three transitions: a step, a skip, or a backstep. Consequently, our alignment matrix has dimensions N x $(L+3)$ where N is the number of measured states. The columns $1:L$ represent alignment of a state to the state $L:1$ states before it. The final 3 columns represent the creation of a new state via alignment of the state to itself, entered into via a step, skip, or backstep, respectively.

The final modification made in our self-alignment method is the above-discussed assessment of an additional self-alignment penalty $P_{SA} = -\frac{3}{2}$ to these newly created states. The full matrix of transition penalties (penalties taken as log probabilities, $S = \log(P_S)$, etc.) is summarized in Fig 8. Using this self-alignment method to identify repeated states, we conduct recombination filtering as described in algorithm 2.

Algorithm S 2 Recombination filter

Input: start with N observed states $\{\boldsymbol{x}^i\}$, $i \in 1:N$ ▷ States are passed in after removal filter
function StepProbs($\{\boldsymbol{x}^i\}$) ▷ Function to calculate the transition-by-transition step-type probabilities
    Calculate Get the scores $\mathbb{S}_{SK}$, $\mathbb{S}_{SB}$, and $\mathbb{S}_{SH}$ from the SVMs $\boldsymbol{S}_{SK}$, $\boldsymbol{S}_{SB}$, and $\boldsymbol{S}_{SH}$
    Calculate Convert SVM scores into relative likelihoods using the attached logit functions
    Solve Use the resulting system of 4 equations to find P_S, P_B, P_H, and P_K for each transition
    Output Transitions matrix $\mathcal{T}$ contains the step-type probabilities for each transition
end function
function SelfAlign($\{\boldsymbol{x}^i\}$)
    $\mathcal{T} \leftarrow$ StepProbs($\{\boldsymbol{x}^i\}$)
    $P_{SA} \leftarrow -\frac{3}{2}$
    Calculate Alignment $\mathcal{A}$ is the alignment of $\{\boldsymbol{x}^i\}$ to $\{\boldsymbol{x}^i\}$ subject to the self-alignment penalty P_{SA} and the transition penalties $\mathcal{T}$ ▷ $\mathcal{A}$ is a 1 x N array, where $\mathcal{A}_i = j$ means that the i^{th} measured state is the j^{th} recombined state
    Output: $\mathcal{A}$
end function
Initialize:
$changed \leftarrow TRUE$
$\{\boldsymbol{x}^i_{new}\} \leftarrow \{\boldsymbol{x}^i\}$
while *changed* **do**
    $\{\boldsymbol{x}^i_{old}\} \leftarrow \{\boldsymbol{x}^i_{new}\}$ ▷ Store the existing $\{\boldsymbol{x}^i_{new}\}$ in a new variable
    $\mathcal{A} \leftarrow$ SelfAlign($\{\boldsymbol{x}^i_{old}\}$) ▷ Conduct self-alignment
    if $\max(\mathcal{A}) = \text{length}(\{\boldsymbol{x}^i_{old}\})$ **then** ▷ We have the same number of recombined states as initial states, meaning nothing has been recombined
        $\{\boldsymbol{x}^i_{new}\} \leftarrow \{\boldsymbol{x}^i_{old}\}$ ▷ no states have changed so just pass on the old ones
        $changed \leftarrow FALSE$ ▷ Our recombination has converged, exit the while loop
    else
        Initialize: $\{\boldsymbol{x}^i_{new}\}$ as a empty holder of size 1 x $\max(\mathcal{A})$ ▷ Storage for the set of recombined states
        for $i \in 1:\max(\mathcal{A})$ **do** ▷ Loop over old states and recombine into new states based on alignment
            $\boldsymbol{x}^i_{new} \leftarrow \text{mean}(\{\boldsymbol{x}^{\{j\}}_{old}\})$ where $\{j\}$ is such that $\mathcal{A}^j = i$ for all $j \in \{j\}$
        end for
        $changed \leftarrow TRUE$ ▷ As long as things have changed, continue recombination
    end if
end while
Output $\{\boldsymbol{x}^i_{new}\}$ ▷ Final output is the new set of recombined states


(a)
Conductance (arbs)
2
4
6
8
10
Observed State Number


(b)
Aligned To
i - 4
i - 3
i - 2
i - 1
New via step
New via skip
New via back
State Number
1
2
3
4
5
6
7
8
9
10

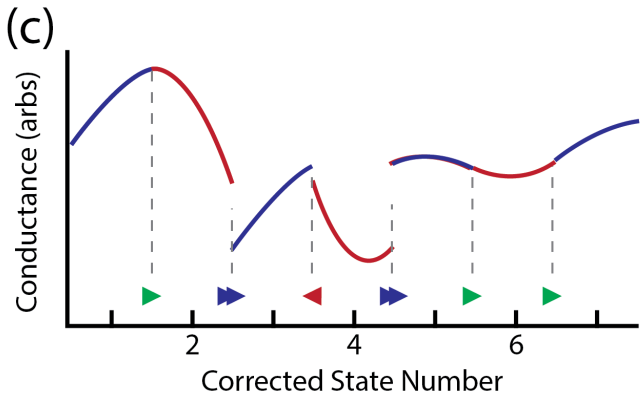

(c)
Conductance (arbs)
2
4
6
Corrected State Number

Figure S 7: Self-alignment procedure for recombination filter. **(a)** The recombination filter seeks to find repeated instances of the same k-mer conductance state in a sequencing read. Shown here is a toy example of a sequencing read with various missteps. Toy data is modeled on a single-nucleotide-stepping enzyme for simplicity. **(b)** The self-alignment of the above states to themselves reveals repeated k-mer states. States 1, 2, 5, 6, 7, 9, and 10 are unique states. States 3 (= state 1), 4 (= state 2), and 8 (= state 7) are repeated measurements of previously observed states. **(c)** Recombining the repeated measurements into single states dramatically reduces the errors in the signal. The remaining misordered states (3 and 4 should be swapped) will be treated by the reordering filter.

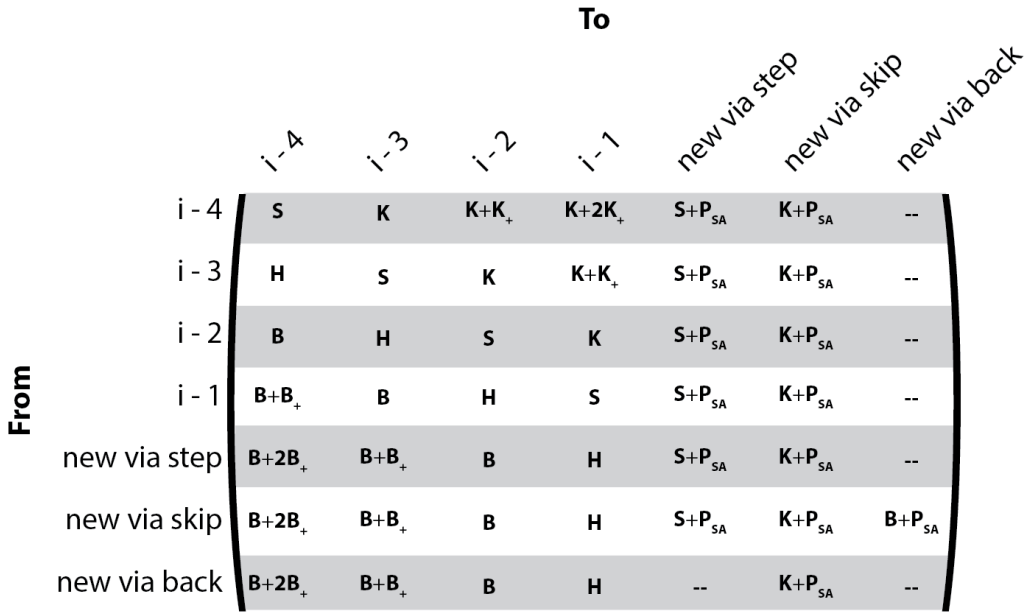


Figure S 8: Self-alignment transition penalties. During self-alignment, the transition from a starting point in the alignment (rows) into a final point in the alignment matrix (columns) takes an additive penalty. Alignments of a state to a previously measured state take a penalty equal to the log probability of the enzyme step required to generate the states in that order. Alignments of a state to itself take a step-type penalty as well as the self-alignment penalty P_{SA}. Certain transitions (marked $--$) are not allowed.

5.3 Reordering Filter

Some enzyme misstep errors can persist in the signal even after removal and recombination filtering. Particularly, complex error modes involving successive enzyme missteps (e.g. a skip, then backstep, then skip as in Fig S7c) can result in out-of-order states even after bad states are removed and duplicate states are recombined. The reordering filter–the last of the three filters involved in the state filtering process–aims to identify and correct these out-of-order states prior to sequencing.

The reordering filter works by using by using an ensemble of SVMs with associated logit functions (as in the recombination filter, Supplemental Note 5.2) to assign a probability that each transition was a single step ("S"), a skip ("K"), or a backstep ("B"). A dynamic programming algorithm is then used to find the most likely set of allowed transitions linking the observed states.

The calculation of step-type probabilities for the reordering filter uses the same SVMs and logits as the recombination filter. The only change here is we are no longer looking for holds (holds by definition result

in duplicate states, and so should be entirely treated by the recombination filter) so we only use two of the three SVMs: $\boldsymbol{S}_{SK}$ to decide between steps and skips, and $\boldsymbol{S}_{SB}$ to decide between steps and backsteps. Using the same procedure as in the recombination filter, we use these two SVMs to calculate the probability that each state-to-state transition represents a step, skip, or backstep. In our notation, the n^{th} transition, from state n to state $n+1$ has a step probability P_S^n, skip probability P_K^n, and backstep probability P_B^n. For a read of N states, the step-type probabilities are summarized in the $N-1$ x 3 matrix $\mathcal{P}$:

$$\mathcal{P} = \begin{bmatrix} P_S^1 & P_K^1 & P_B^1 \\ P_S^2 & P_K^2 & P_B^2 \\ ... & ... & ... \\ P_S^{N-1} & P_K^{N-1} & P_B^{N-1} \end{bmatrix}$$

For convenience, we convert these probabilities to log-probabilities (denoted $\mathbb{P}$) for further use:

$$\mathbb{P} = \log(\mathcal{P})$$

Now with the step-type log-probabilities calculated, we use a dynamic programming algorithm (algorithm S3) to find the most likely path of transitions through the states, subject to certain constraints. Namely, we must choose a set of transitions reflective of a state ordering not requiring any repeated visits to the same state. For example, we cannot choose to take a step from state 1 to 2, then a backstep from 2 to 3. This hypothetical path would imply that state 3 is a repeated measurement of state 1. If this were the case, these states would have been recombined during the previous filtering step. As they were not recombined, this transition pathway must be ruled out, and is not allowed during reordering. To implement this "no repeated states" condition, we consider 4 "transition states": steps (S), backsteps (B), skips where the previous transition was a step or a skip ($K|SK$), and skips where the previous transition was a backstep ($K|B$). The allowed linkages between these transition states are summarized as an allowed linkage matrix $\mathbb{L}$:

$$\mathbb{L} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

where a linkage from transition state i to transition state j is allowed if $\mathbb{L}_{ij} = 1$ and is not allowed if $\mathbb{L}_{ij} = 0$. The 1st row and column in $\mathbb{L}$ represents the step "transition state" S, the 2nd represents B, the 3rd represents $K|SK$, and the 4th represents $K|B$. So, for example $\mathbb{L}_{1,2} = 0$ tells us that we cannot jump from a step into a backstep (as discussed above). Subject to these allowed transitions, we compute an alignment matrix $\mathbb{A}$ and traceback matrix $\mathbb{B}$ that provide us the most likely pathway through the allowed transitions. This pathway tells us what type of step was most likely taken at each transition. With this step-type information, we can optimally reorder the observed states to finally reconstruct the most likely sequence order, completing the filtering process.

Algorithm S 3 Reordering filter

Input:
$\mathbb{P}$ ▷ step-type log-probabilities for each of the $N-1$ transitions
$\mathbb{L}$ ▷ matrix of allowed "transition type" linkages
Initialize:
$\mathbb{A} \leftarrow \text{ones}(N-1, 4)$ ▷ alignment matrix, 1st column is S, 2nd is B, 3rd is $K|SK$, 4th is $K|B$
$\mathbb{B} \leftarrow \text{ones}(N-1, 4)$ ▷ traceback matrix
$\mathbb{A}_{1,1:3} \leftarrow [\mathbb{P}_{1,1}, \mathbb{P}_{1,2}, \mathbb{P}_{1,3}, -\infty]$ ▷ Fill first row of alignment matrix
for $i \in 2 : (N-1)$ **do** ▷ Loop over the rest of the transitions, filling the alignment and traceback matrices
    $A \leftarrow [\mathbb{P}_{i,1}, \mathbb{P}_{i,3}, \mathbb{A}_{i,2}, \mathbb{A}_{i,2}]$ ▷ Begin filling next row in alignment matrix
    $T \leftarrow [\mathbb{A}_{i-1,1}, \mathbb{A}_{i-1,2}, \mathbb{A}_{i-1,3}, \mathbb{A}_{i-1,4}]$ ▷ T stores the scores of possible cells we can transition in from
    for $j \in 1 : 4$ **do** ▷ Loop over the 4 transition types
        $t \leftarrow T$ ▷ make a copy of t as we fill this particular cell
        $t_k \leftarrow -\infty$ for $\forall k$ where $\mathbb{L}_{k,j} = 0$ ▷ turn off disallowed transitions
        $t_* \leftarrow \max(t)$ ▷ take the best scoring path in
        $b_* \leftarrow k$ such that $t_k = t_*$ ▷ record which transition had the best score
        $\mathbb{A}_{i,j} \leftarrow A_j + t_*$ ▷ fill alignment matrix cell
        $\mathbb{B}_{i,j} \leftarrow b_*$ ▷ fill traceback matrix cell
    end for
end for
Intialize $\mathbb{R} \leftarrow []$ ▷ inialize storage for the best path through the alignment matrix as we conduct traceback
$\mathbb{R} \leftarrow [r\mathbb{R}]$ where r is such that $\mathbb{A}_{n-1,r} = \max(\mathbb{A}_{n-1,:}$ ▷ start traceback at best scoring cell in bottom row of $\mathbb{A}$
for do$i \in (n-1) : -1 : 2$ ▷ conduct traceback over most likely pathway
    $r \leftarrow \mathbb{B}_{i,r}$ ▷ $\mathbb{B}$ tells us where we came from to get to the max cell in $\mathbb{A}$
    $\mathbb{R} \leftarrow [r\mathbb{R}]$ ▷ append the location of the best score in the row to the start of the traceback
end for
Output: r contains the type of step ($1 = S$, $2 = B$, 3 and $4 = K$) taken at each transition

6 Sequencing Algorithm

DNA sequencing is performed using a hidden Markov model (HMM) solver as described below. Simply, we decode the series of k-mers most likely to have generated the observed series of conductance states by conducting an alignment between the observed states and the 6-mer model states. In standard sequence-to-sequence (or conductance-to-conductance) alignment, the alignment proceeds from left-to-right in both sequences. In contrast, this sequencing alignment proceeds left to right in the measured states, but jumps around in the model states based on the allowed k-mer transitions. For example, $AAAAAT$ (the 4th k-mer) can transition to $AAAATG$ (the 15th k-mer) via a single nucleotide step, but requires a 6 nucleotide jump to reach the 5th k-mer, $AAAACA$. Our full adaptation and calculation of the measured-to-model alignment is described in detail below.

6.1 Match Scores

We first compute an score matrix S of match likelihoods S_{nj} between measured state n and reference 6-mer model state j. The measured state and model state are each characterized by their d principal component amplitudes and the associated uncertainty (for measured states) or covariance (for model states) covariance matrix. The measured state is written as $\boldsymbol{x}_n$ with uncertainty matrix $\Sigma_{\boldsymbol{x}_n}$, and the reference state is written as $\boldsymbol{y}_j$ with covariance matrix $\Sigma_{\boldsymbol{y}_j}$. The match score between these states is given by

$$S_{nj} = \frac{1}{\sqrt{(2\pi)^d \frac{|\Sigma_{\boldsymbol{x}_n}^{-1}||\Sigma_{\boldsymbol{y}_j}^{-1}|}{|\Sigma_{\boldsymbol{x}_n}^{-1}+\Sigma_{\boldsymbol{y}_j}^{-1}|}}} \exp\left[-\frac{1}{2}\left(\Sigma_{\boldsymbol{x}_n}^{-1}\boldsymbol{x}_n - \Sigma_{\boldsymbol{y}_j}^{-1}\boldsymbol{y}_j\right)^T \left(\Sigma_{\boldsymbol{x}_n}^{-1}+\Sigma_{\boldsymbol{y}_j}^{-1}\right)\left(\Sigma_{\boldsymbol{x}_n}^{-1}\boldsymbol{x}_n - \Sigma_{\boldsymbol{y}_j}^{-1}\boldsymbol{y}_j\right)\right] \tag{15}$$

The corresponding array of log-likelihoods is the natural logarithm of this,

$$\begin{aligned} s_{nj} = \log S_{nj} = \frac{1}{2}\Bigg[& d\log 2\pi + \log|\Sigma_{\boldsymbol{x}_n}^{-1}| + \log|\Sigma_{\boldsymbol{y}_j}^{-1}| - \log|\Sigma_{\boldsymbol{x}_n}^{-1}+\Sigma_{\boldsymbol{y}_j}^{-1}| \\ & - \left(\Sigma_{\boldsymbol{x}_n}^{-1}\boldsymbol{x}_n - \Sigma_{\boldsymbol{y}_j}^{-1}\boldsymbol{y}_j\right)^T \left(\Sigma_{\boldsymbol{x}_n}^{-1}+\Sigma_{\boldsymbol{y}_j}^{-1}\right)\left(\Sigma_{\boldsymbol{x}_n}^{-1}\boldsymbol{x}_n - \Sigma_{\boldsymbol{y}_j}^{-1}\boldsymbol{y}_j\right)\Bigg]. \end{aligned} \tag{16}$$

6.2 Hel308 Backstep Kinetics

We use the known backstep kinetics of the Hel308 enzyme to inform our sequencing. Specifically, previous work found that Hel308 is far more likely to backstep when in its ATP-independent state (the "pre" states in our 6-mer model) than when in its ATP-dependent state (the "post" states in our 6-mer model) (Craig et al. Proc. Natl. Acad. Sci., 2017). Consequently, measured states determined to have backstepped during enzyme step correction (Supplemental Note 5.2) are more likely to have been generated with ATP-independent states in our 6-mer model. To use this information, we label which measured states backstepped, then incorporate the independent/dependent state probabilities into the score matrix S as follows.

We estimate the probability that a state that backstepped was an ATP-independent state $P_{ind|b} = 0.975$ from Hel308 kinetics data (Craig et al. Proc. Natl. Acad. Sci., 2017). The overall probability that a state will backstep is also estimated from kinetics data as $P_b = 0.025$ (Craig et al. Proc. Natl. Acad. Sci., 2017). From these, we can calculate the probability $P_{ind|\sim b}$ that an ATP-independent state will *not* backstep as

$$P_{ind|\sim b} = \frac{\frac{1}{2} - P_b * P_{ind|b}}{P_{ind|\sim b}}$$

The probability that a given state is ATP-dependent given that it did ($P_{dep|b}$) or did not ($P_{dep|\sim b}$) backstep is simply 1 minus the complementary independent probability:

$$P_{dep|b} = 1 - P_{ind|b}$$

$$P_{dep|\sim b} = 1 - P_{ind|\sim b}$$

We incorporate these probabilities into the score matrix S by first converting them to log-probabilities: $p = log(P)$. The odd-numbered columns in the score matrix (S_{ij} where j is odd) represent matches to ATP-independent states and the even-numbered columns (S_{ij} where j is even) are matches to ATP-dependent states. For every measured state i where we observed a backstep, we update the row $S_{i:}$ as

$$S_{ij} \leftarrow S_{ij} + p_{ind|b} \text{ if } j \text{ is odd}$$

and

$$S_{ij} \leftarrow S_{ij} + p_{dep|b} \text{ if } j \text{ is even}$$

Likewise, for every measured state i where we did not observe a backstep, we update the row $S_{i:}$ as

$$S_{ij} \leftarrow S_{ij} + p_{ind|\sim b} \text{ if } j \text{ is odd}$$

and

$$S_{ij} \leftarrow S_{ij} + p_{dep|\sim b} \text{ if } j \text{ is even}$$

This accounting tells our sequencer to preferentially call states for which a backstep was observed as ATP-independent states.

6.3 Transition Probabilities

We also determine transition probabilities between each pair of states. In the case of constant-voltage sequencing, the relative probabilities of different transitions (step, skip1, skip2, ...) between any two given states are fixed for all states. In variable-voltage sequencing, we can use the overlap between two states' conductance curves in order to get a more informed estimate of the relative probabilities. Two states whose conductance curves overlap well are likely to be separated by a single step, whereas states whose conductance curves do not overlap are more likely to be skips. We find that we can differentiate effectively between steps and non-steps (88.9% correct calls on the labeled validation set), as well as between single skips and larger skips (79.1% correct calls on the validation set).

We use an ensemble of SVMs (similar to those described in Supplemental Note 5) to assign each transition its own set of probabilities of being a step, skip1, or a larger skip. The SVMs take as input the principal components of the two measured states m and n to assign probabilities to the different types of transitions between m and n. The ensemble of SVMs is made up of two classifiers ($\boldsymbol{S}_1$ and $\boldsymbol{S}_2$), trained on labeled examples of steps and variously sized skips from the ΦX-174 data collected during the 6-mer model construction (Supplemental Note 7.3.1). The scores $\mathbb{S}_i$ output by the SVMs $\boldsymbol{S}_i$ are converted into probabilities using the same logit procedure as described previously in Supplemental Note 5.

$\boldsymbol{S}_1$ differentiates between steps and non-steps, and assigns the probability that the transition from state m to state n was a single step as

$$P_{mn}^{(1)} = \text{logit}(\mathbb{S}_1, \alpha_1, \beta_1)$$

with the logit function as defined previously and using logit parameters α_1 and β_1 determined from global likelihood maximization over a labeled validation set (Supplemental Note 5).

Similarly, $\boldsymbol{S}_2$ differentiates between single skips (involving two half-nucleotide steps) and larger skips (involving more than two half-nucleotide steps). $\boldsymbol{S}_2$ gives us the probability that the transition between states m and n was a single skip given that it was not a step:

$$P_{mn}^{(2|\sim 1)} = \text{logit}(\mathbb{S}_2, \alpha_2, \beta_2)$$

The overall probability then that the transition between m and n was a single skip is then

$$P_{mn}^{(2)} = P_{mn}^{(2|\sim 1)} * P_{mn}^{(\sim 1)} = \text{logit}(\mathbb{S}_2, \alpha_2, \beta_2) * \big(1 - \text{logit}(\mathbb{S}_1, \alpha_1, \beta_1)\big)$$

We set the probabilities of larger skips by an affine probability $P_{mn}^{(+)}$ such that

$$P_{mn}^{(k)} = P_{mn}^{(k-1)} * P_{mn}^{(+)}$$

$P_{mn}^{(+)}$ is set so that the summed probability of all possible steps and skips sums to 1.

6.4 Transition Matrix

For each pair of measured states we wish to consider, we compute an 8192 x 8192 transition matrix T composed of the probabilities of transitioning between map states:

$$T_{mn,ij} = P\left(\begin{array}{c}\text{state } m \text{ is a measurement} \\ \text{of true map state } i\end{array} \middle| \begin{array}{c}\text{state } n \text{ is a measurement} \\ \text{of true map state } j\end{array}\right) \tag{17}$$

To calculate the transition matrix, we first find a matrix whose elements are the probabilities of having transitioned between states conditioned on a step size of a single half-step,

$$\tau_{ij1} = P\left(\begin{array}{c}\text{state } t \text{ is a measurement} \\ \text{of true map state } i\end{array} \middle| \begin{array}{c}\text{state } t+1 \text{ is a measurement}, \\ \text{of true map state } j\end{array} \text{ step size} = 1\right)$$

$$= \begin{cases} 1 & i \text{ is a "pre" state and } j \text{ the corresponding "post" state} \\ 1/4 & i \text{ is a "post" state and } j \text{ a succeeding 6-mer} \\ 0 & \text{otherwise} \end{cases}$$

where we define two 6-mers as "successive" when they share 5 nucleotides shifted by one position, e.g. A<u>CGTAC</u> could be succeeded by <u>CGTAC</u>T. We then define a similar matrix for larger sizes of step, which is calculated by taking powers of the single half-step matrix:

$$\tau_{ijk} = P\left(\begin{array}{c}\text{state } t \text{ is a measurement} \\ \text{of true map state } i\end{array} \middle| \begin{array}{c}\text{state } t+1 \text{ is a measurement}, \\ \text{of true map state } j\end{array} \text{ step size} = k\right) = (\tau_{ij1})^k.$$

Finally, we define $\tau_{ij(12)}$ to correspond to all transitions with step size greater than or equal to 12, which could be between any two states. Therefore it has uniform entries $\tau_{ij(12)} = 1/8192$. Now, we can compute the total transition probability matrix as the sum of the probabilities of each possible step size by which the measured levels could have advanced:

$$T_{mn,ij} = P^{(1)}_{mn}\tau_{ij1} + \sum_{k=2}^{12} P^{(2)}_{mn}\left(P^{(+)}_{mn}\right)^{k-2}\tau_{ijk}. \tag{18}$$

We also define the log transition likelihood, $t_{mn,ij} = \log T_{mn,ij}$.

6.5 Markov Model

If we are sequencing a read of N measured states, we create an $N \times 8192$ alignment matrix, $\mathbb{A}$. In each element of the array $\mathbb{A}_{nj}$ we write an estimate of the log-likelihood that measured state n came from map state j, given the observation of measured states 1 through $n-1$:

$$\mathbb{A}_{1j} = s_{1j} + \log\left(1 - P_1^{(\text{bad})}\right)$$

$$\mathbb{A}_{nj} = \log \sum_{k=1}^{8192}\sum_{m=1}^{n-1} \exp\left\{s_{nj} + t_{mn,kj} + h_{mk} + \log\left(1 - P_n^{(\text{bad})}\right) + \sum_{l=m+1}^{n-1} \log P_l^{(\text{bad})}\right\}, \quad n > 1,$$

where $P_n^{(\text{bad})}$ is the probability that observed state n is an erroneous measurement that should be omitted from the sequencer. In constant-voltage sequencing, $P^{(bad)}$ is taken as a constant value for all states. In variable-voltage sequencing, we use the same bad state classifier as in the removal filter (Supplemental Note 5.1) to assign a unique $P^{(bad)}$ to each state.

This is a forwards-propagating approximation of a MAP algorithm, which in practice gives similar results to a slower forwards-backwards algorithm relying on all observations to determine likelihoods (Bahl et al. IEEE Trans. Inf. Theory, 1974; Viterbi, IEEE Trans. Inf. Theory, 1967). We take two additional steps to increase speed. Firstly, using the approximation

$$\log \sum_i e^{a_i} \approx \arg\max_i a_i,$$

which is valid when one a_i is significantly larger than the others. We replace the logarithms of sums of exponentials in our alignment matrix $\mathbb{A}$ with maxima, which are more expedient to calculate:

$$\mathbb{A}_{nj} = \max_{k,m} \left\{ s_{nj} + t_{mn,kj} + h_{mk} + \log\left(1 - P_n^{(\text{bad})}\right) + \sum_{l=m+1}^{n-1} \log P_l^{(\text{bad})} \right\}, \quad n > 1,$$

We also record a traceback array,

$$\mathbb{B}_{nj} = \arg\max_{k,m} \left\{ s_{nj} + t_{mn,kj} + h_{mk} + \log\left(1 - P_n^{(\text{bad})}\right) + \sum_{l=m+1}^{n-1} \log P_l^{(\text{bad})} \right\}, \quad n > 1.$$

Thus $\mathbb{B}nj = (m, k)$, such that $\mathbb{A}_{mk}$ is the maximum likelihood observed state-map state matching to have occurred just prior to the one described by likelihood $\mathbb{A}_{nj}$. This is a Viterbi algorithm (Viterbi, IEEE Trans. Inf. Theory, 1967) approximating the results of the MAP algorithm (Bahl et al. IEEE Trans. Inf. Theory, 1974).

Additionally, we improve speed by restricting the max over m to only cases where $m > n - q - 1$, where q is the maximum number of sequential "bad" observed states allowed by the algorithm. We found good results taking $q = 3$, as cases of more than 3 consecutive "bad" states not removed by the removal filter (Supplemental Note 5.1) are exceedingly rare. We also restrict the max over k to values of k such that $s_{nk} > \max_j s_{nj} - c$, where c is a score difference cut-off. Similarly, $\mathbb{A}_{nk}$ with k subject to the same restrictions is left uncalculated, because it will not be used by the algorithm under any circumstances. This avoids spending time calculating the probability flow into and out of states unlikely to represent the optimal alignment. Using $c = 10$ provides identical results to the full calculation in all tested cases, while dramatically reducing the computational load.

Calculation of $\mathbb{A}_{nj}$ and $\mathbb{B}_{nj}$ requires knowledge of $\mathbb{A}_{(n-1)k}$ for all k, so the array is calculated starting with the $n = 1$ elements and proceeding upwards in n.

6.6 Traceback and Sequence Construction

Once $\mathbb{B}$ has been calculated, we find sequence of map states with the maximum approximate-likelihood of having produced the observed states. We do this by starting at the maximum approximate-likelihood entry in alignment matrix, at $\mathbb{A}_{n^*j^*}$, and iteratively following the traceback array through the most likely sequence of transitions. In other words, if a is the sequence of indices of true map states and n is the sequence of indices of valid observed states,

$$(n_{\text{final}},\ a_{\text{final}}) = \arg\max_{(n,\ j)}\ \mathbb{A}_{nj},$$

$$(n_i, a_i) = \mathbb{B}_{n_{i+1}, a_{i+1}}.$$

From a we calculate the most likely DNA sequence. Between a_i and a_{i+1}, we find the most likely (the smallest) step size that could transition between those two 6-mers, and fill in bases accordingly. For example, GTACAC (pre) could transition to ACACTT (pre) with four half-nucleotide steps, moving the GT outside of the pore's constriction and the TT into it. It could also make the transition by taking eight half-nucleotide steps, moving the GTAC outside and the ACTT in, or by taking twelve half-nucleotide steps, moving the entire sequence GTACAC out of the constriction and the entire sequence of ACACTT in. The four-nucleotide step is the most likely based on our empirical model of transition probabilities. Therefore, if these two 6-mers were a_i and a_{i+1}, they would be sequenced as GTACACTT, because that is more likely than the alternative choices GTACACACTT or GTACACACACTT. By performing this step for every state in a, we arrive at a close-to-optimal-likelihood sequence for the observed states.

7 Constructing the 6-mer Model

7.1 General Considerations

To sequence the variable-voltage reads, we determine the DNA sequence most likely to have generated the observed series of conductance states. In order to decode this DNA sequence, we require a map relating the conductance signal and the DNA sequence in the pore. The nanopore signal is modeled as being generated by the k nucleotides (i.e. the k-mer, with k an integer) nearest the constriction of the pore. This model is described by a map of the 4^k possible k-mers to the conductances typically observed when they are in the pore. The k-mer model has been previously validated as an effective model for nanopore signal prediction (Manrao, et al. Nat. Biotechnol. 2012).

Previous work on constant-voltage nanopore DNA sequencing used a model with $k = 4$, but we found that a 4-mer model was insufficient for our variable-voltage signal. During variable-voltage sequencing, the nucleotides centered within the nanopore constriction at each enzyme registration are shifted forwards and backwards as the DNA is stretched by the changing voltage. This shifting means that the nucleotides both 5' and 3' of the central 4-mer have more of an effect on the observed signal when using variable-voltage than they had in the constant-voltage case. To better model this effect, we expanded the model from 4-mers to 6-mers, now including an additional nucleotide on both the 5' and 3' ends of the previous 4-mers, expanding our model from $4^4 = 256$ 4-mers to $4^6 = 4096$ 6-mers.

Each state in the variable-voltage model is characterized by more complex information than in the constant-voltage model. In the constant-voltage model, each k-mer state was characterized by its mean conductance value (G), its typical conductance noise (dG), and the variance of both of these quantities. In contrast, during variable-voltage sequencing, the k-mer state occupied at each enzyme step is not a constant conductance value characterized by a mean and noise, but instead a conductance vs. voltage ($G - V$) curve. We found that each variable-voltage k-mer state is well characterized by its 3 principle component amplitudes $\boldsymbol{p}$ and their covariance $\boldsymbol{\Sigma^p}$ (Supplemental Note 3).

Previous work by our lab used the Φ29 DNA polymerase (DNAP) as the motor protein controlling the DNA, which steps in full nucleotide increments. This work instead uses the Hel308 DNA helicase as the motor protein, which takes two distinct steps per nucleotide, an ATP-dependent step and an ATP-independent step (Derrington et al. Nat. Biotechnol. 2015). As the signal now contains two distinct enzyme states per nucleotide, each single k-mer is now associated with two distinct states, and the 4096 6-mers in the model represent 8192 total states, two for each k-mer.

7.2 Initial Model

To construct the variable-voltage 6-mer model for the two-step-per-nucleotide Hel308 helicase motor protein, we refined the existing constant-voltage 4-mer model. We note that the 6-mer denoted $N_1N_2N_3N_4N_5N_6$ (where N_i denotes a nucleotide A, C, G, or T) is made up of 3 distinct 4-mers: $N_1N_2N_3N_4$, $N_2N_3N_4N_5$, and $N_3N_4N_5N_6$. Additionally, we recall that the variable-voltage signal samples the conductance of the translocating DNA as a function of position. This function should interpolate smoothly between the discrete samples taken at constant voltage. We approximate the smooth conductance vs. position curve interpolating the DNA positions between the three constituent 4-mers by a quadratic fit to the three conductances known from the phi29 DNAP 4-mer model (Fig S9).

Sampling this curve at the appropriate DNA positions gives an estimate of the variable-voltage 6-mer states. The first of the two Hel308 helicase states is known to have the same DNA registration within the pore as Φ29 DNAP (Derrington et al. Nat. Biotechnol. 2015). The constant-voltage model was derived from experiments run at a bias of $180mV$, and we know from the DNA force-extension curve (Supplemental Note 1.1) that the variable-voltage sweep stretches the DNA $\sim +0.1nt$ from the $180mV$ position at its highest voltage and $\sim -0.9nt$ from the $180mV$ position at its lowest voltage. So, to predict the state 1 conductance vs. position curve, we sample the interpolating curve at equally-spaced points from $-0.9nt$ to $+0.10nt$ (Fig S9a). The second of Hel308's two states is $0.55nt$ 3' from state 1. So, state 2 is predicted by sampling the interpolating curve between $-0.35nt$ and $0.65nt$ (Fig S9b).

The amplitudes of the 3 principle components $\boldsymbol{p}$ for each 6-mer state can now be calculated from the predicted $G - V$ curve (Supplemental Note 3). All 8192 states in the initial guess map were assigned the same default covariance for their 3 principle components. The 3 principle components are sufficient for this

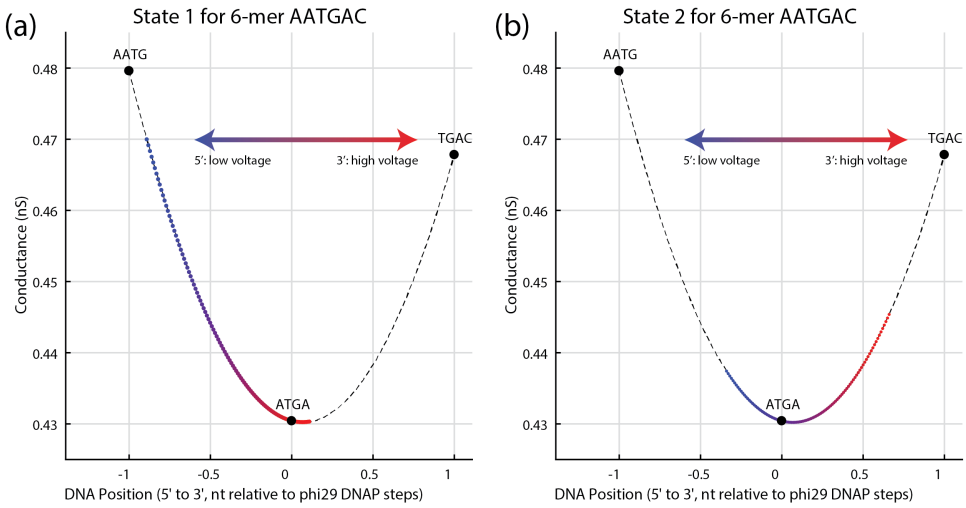


Figure S 9: Method of generating variable-voltage Hel308 helicase 6-mer predictions from the constant-voltage Φ29 DNAP 4-mer model. Black points show the constant-voltage 4-mer model predictions for the 3 4-mers comprising the 6-mer of interest. Black dashed line shows the quadratic fit to the 4-mer model predictions, which acts as an estimate of the smooth conductance vs. position profile explored by variable-voltage. The blue to red points show the predicted conductance as a function of DNA position for the given 6-mer. Blue-er points correspond to lower voltages, red-er points to higher voltages. **(a)** Prediction for the first of the two Hel308 states. The first of the two Hel308 states has the same DNA registration within the pore as the Φ29 DNAP full-nucleotide steps. The conductance value of the Hel308 state 1 prediction coincides with the Φ29 DNAP conductance prediction for the central 4-mer in the 6-mer of interest. **(b)** Prediction for the second of the two Hel308 states. The second state is shifted $0.55nt$ to the 3' of the first state.

initial model to provide a framework on which to build an empirical 6-mer model based on measurements of DNA under variable-voltage conditions.

7.3 Measuring Genomic DNA of Known Sequence

To build the 6-mer model we will ultimately use for DNA sequencing, we measure the signal produced by all 4096 of the 6-mers during variable-voltage experiments. We read DNA of known sequence under the variable-voltage sequencing conditions, then use the measured signals of this known DNA to update the initial 6-mer model.

7.3.1 ΦX174

We first measured the 5386 bp ΦX174 genome (New England Biolabs). We prepared the circular genome for variable-voltage nanopore sequencing experiments as follows:

1. The circular genome was linearized via a double digest using the restriction enzymes PstI and AvaII (New England Biolabs). ΦX174 was prepared in 20 μg batches. For each batch, a mixture of
   (a) 40 μL of ΦX174 DNA at 500 $\frac{ng}{\mu L}$
   (b) 5 μL of 10x CutSmart (New England Biolabs) Buffer
   (c) 1 μL of PstI-HF restriction enzyme at 100 $\frac{Units}{\mu L}$
   (d) 1 μL of AvaII restriction enzyme at 10 $\frac{Units}{\mu L}$

(e) 3 μL of molecular biology grade water

was incubated at 37°C for 60 minutes, then heat inactivated via heating to 80°C for 20 minutes. Each of PstI and AvaII have a single cut site in $\Phi X174$, so the double digest yields two linear fragments, one of 5042 bp, the other of 344 bp (Fig S10a, b).

2. The linearized fragments were purified from the heat-inactivated restriction enzymes on a DNA Clean and Concentrator column (Zymo Research), and eluted into 50 μL of molecular biology grade water.

3. Two DNA adapters are attached to each of the fragments enabling reading by the nanopore (Fig S10c, d). At one end, we ligate a threading adapter, which promotes capture a single strand into the pore, entering with the 5' end of the DNA threading into the pore. This threading adapter also features a cholesterol tagged 3' end. The cholesterol tagged 3' end inserts into the lipid bilayer, localizing the DNA strands near the pore and increasing the rate of 5' end capture. The threading adapter is made up of two partially complementary strands: the $\Phi X174$ threading strand and the $\Phi X174$ cholesterol blocker (Table S1). The threading adapter is formed at high concentration by mixing equal volumes of 12.5 μM threading strand and cholesterol blocker, then annealing to yield a 12.5 μM solution of the fully formed threading adapters.

4. At the other end, we ligate a loading adapter which promotes loading of the Hel308 helicase onto the DNA construct. This adapter consists of two partially complementary strands: the $\Phi X174$ loading strand and the $\Phi X174$ loading blocker (Table S1). The loading adapter is formed at high concentration by mixing equal volumes 12.5 μM loading strand and loading blocker, then annealing to yield a 12.5 μM solution of the fully formed loading adapters.

5. The threading and loading adapters are ligated to the sticky ends of the linearized $\Phi X174$ DNA fragments. A mixture of

   (a) 48 μL of 100 nM $\Phi X174$ DNA
   (b) 6 μL of 10x T4 ligase buffer (New England Biolabs)
   (c) 2 μL of 12.5 μM threading adapters (to give 5:1 ratio of adapters to target sticky ends)
   (d) 2 μL of 12.5 μM loading adpaters (to give 5:1 ratio of adapters to target sticky ends)
   (e) 1 μL of molecular biology grade water
   (f) 1 μL of T4 DNA ligase at 400 $\frac{Units}{\mu L}$ (New England Biolabs)

   was incubated at 16°C for 60 minutes, then heat inactivated by heating to 65°C for 10 minutes.

6. The fully formed DNA constructs (Fig S10e) were purified from the remaining un-ligated adapters and the heat-inactivated ligase on a DNA Clean and Concentrator column, and eluted into 50 μL of molecular biology grade water.

These two fragments, now with the necessary adapters attached, were run using the standard variable-voltage nanopore sequencing conditions. In total, we observed 155 individual reads comprising 188543 enzyme steps, or 94272 nucleotides.

7.3.2 λ Phage

In order to get better coverage of numerous 6-mers not present in the $\Phi X174$ genome, and to increase the context diversity of all of our measurements, we next decided to measure a larger genome. For this second round of measurements, we chose the 48502 bp λ bacteriophage genome. We chose a new approach to fragmentation for this experiment in order to provide uniform read coverage over the entire genome. Due to the limited processivity of our Hel308 helicase (∼1000 nt, Supplemental Note 10), restriction enzyme fragmentation results in most reads starting at the restriction site, but terminating prior to reading the entire fragment. Consequently, such a fragmentation gives excellent read coverage near the restriction sites, but poor coverage further away from them.

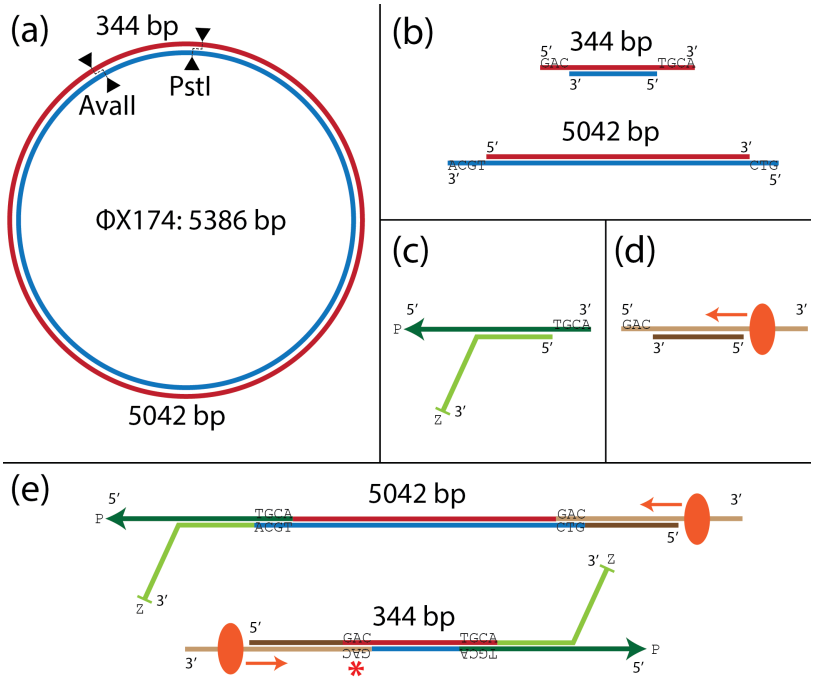


Figure S 10: Preparation of $\Phi X174$ for variable-voltage sequencing. **(a)** The circular 5386 bp genome is cut twice using the AvaII and PstI restriction enzymes. **(b)** Restriction results in two fragments of 344 and 5042 bp, with sticky ends of size 3 and 4 nt. **(c)** The threading adapter consists of a threading strand (dark green) featuring a 5' phosphate (P, arrowhead) which promotes capture of this end by the pore, and a cholesterol blocker strand (light green), featuring a 3' cholesterol tag (Z, crossbar) which associates with the lipid bilayer to concentrate the DNA constructs near the pore and increase the capture rate. **(d)** The loading adapter consists of a loading strand (tan) which overhangs the blocking strand (brown) at the 3' end to provide a loading site for the Hel308 helicase (orange ellipse). The helicase loads at the ss-dsDNA junction and proceeds to walk in a 3' to 5' direction along the loading strand (orange arrow). **(e)** After adapter ligation, the DNA constructs are now ready to be run in the variable-voltage sequencing experiments. Our adapters are designed such that we will read the sense (red) strand of the long fragment, and the antisense (blue) strand fo the short fragment. The red asterisk marks a sticky end mismatch at the loading end of the short fragment, a byproduct of the non-palindromic AvaII cut site. Despite this mismatch, we still observe a population of reads of this smaller fragment, indicating that the loading adapter did still attach with some efficiency.

For uniform coverage, we instead use two separate Covaris products giving random shearing over the entire genome into fragments of a well-defined size range. In one λ library preparation, we used the Covaris Blue DNA miniTUBE, which yielded random fragments of on average 3 kbp in length. For our second library preparation, we used Covaris gTUBEs to get random fragments of on average 6 kbp in length. We switched from miniTUBEs to gTUBEs simply for easy of use, as these required only a centrifuge, and not the Covaris sonicator instrument. For both shearing methods, the library preparation proceeded as follows:

1. The full length λ DNA (Promega) was fragmented using either Blue miniTUBEs (in 20 μg batches) or gTUBEs (in 30 μg batches) (Fig S11a, b).

   For miniTUBE fragmentation, 20 μg of λ DNA was suspended in Tris EDTA buffered at pH 8.0 to a total volume of 200 μL. DNA was then fragmented using the Covaris M220 Focused-ultrasonicator, using the recommended settings for product fragments of ∼3000 bp in length.

   For gTUBE fragmentation, 30 μg of λ DNA was suspended in molecular biology grade water to a total volume of 150 μL. The gTUBE was then centrifuged on an Eppendorf 5417R centrifuge for 30 seconds at 12400 rpm (corresponding to 16200 g), resulting in fragments of ∼6000 bp in length.

2. Following fragmentation, the DNA fragments have random 3' and 5' overhangs. Before proceeding with adapter ligation, we ensure that all DNA fragments are blunt-ended by running an end repair protocol (Fig S11c). Using the NEBNext end repair module (New England Biolabs), a mixture of

   (a) 5 μg fragmented DNA
   (b) 10 μL of NEBNext 10x End Repair Reaction Buffer
   (c) 5 μL of NEBNext End Repair Enzyme Mix
   (d) Molecular biology grade water to total volume of 100 μL

   was incubated at 20°C for 30 minutes. The end-repaired fragments (now blunt-ended) were purified on a DNA Clean and Concentrate column.

3. After end repair, we used the NEBNext dA-tailing module (New England Biolabs) to attach a dA monomer at the 3' end of each strand as a target for adapter ligation (Fig S11d). A mixture of

   (a) 5 μg of λ DNA
   (b) 5 μL of 10x NEBNext dA-Tailing Reaction Buffer
   (c) 3 μL of Klenow Fragment (3' → 5' exo$^-$)
   (d) Molecular biology grade water to a total reaction volume of 50 μL

   was incubated at 37°C for 30 minutes, then purified on a DNA Clean and Concentrate column.

4. Similar threading and loading adapters are used for variable-voltage sequencing experiments on the λ DNA as were used for $\Phi X174$, differing only in the sequence at the sticky ends to be ligated onto the genomic DNA fragments. For the threading adapter (Fig S11e), equimolar parts of the λ threading strand and the λ cholesterol blocker (Table S1) were mixed and annealed to a final concentration of 10 μM. Similarly, for the loading adapter (Fig S11f), equal molar parts of the λ loading strand and the λ loading blocker were mixed and annealed to a final concentration of 10 μM.

5. Adapters were ligated to the dA-tailed λ DNA fragments using T4 DNA ligase (New England Biolabs). A mixture of

   (a) 10 μg of λ DNA fragments
   (b) 3 μL of 10 μM threading adapters (for a ∼ 10:1 adapter to dA-tail end ratio)
   (c) 3 μL of 10 μM loading adapters (for a ∼ 10:1 adapter to dA-tail end ratio)
   (d) 15 μL of 10X Ligation Buffer
   (e) 7.5 μL T4 DNA Ligase

(f) Molecular biology grade water up to a total reaction volume of 150 μL

was incubated at $22°C$ for 125 minutes, then heat inactivated at $65°C$ for 10 minutes. The ligation products (Fig S11g) were purified on DNA Clean and Concentrator columns to remove the inactive ligase and residual un-ligated adapters, and eluted into molecular biology grade water.

As all the 3' ends of the λ fragments have the same single dA overhang, not all ligation products will have the correct conformation of one threading adapter and one loading adapter. 25% of the population will have loading adapters at each end, and 25% will have threading adapters at each end. This reduces the overall effective yield of this library preparation by half, but a sufficient number of constructs were well formed to allow us to generate 128 individual reads comprising 120867 enzyme steps, or 60434 nucleotides.

7.4 Building the Empirical 6-mer Model from Genomic Reads

Having measured a total of 309410 enzyme steps along genomic DNA tracks (120867 in λ, 188543 in $\Phi X174$) representing 154705 measured nucleotides, we now organize these measurements to empirically update the initial model of the predicted nanopore signal for each of the 8192 model states (2 enzyme states for each of the 4096 6-mers). Each observed enzyme step is a measurement of one of the two Hel38 helicase states at one of the 4096 possible 6-mers.

To update the model, we must associate the signal at each enzyme step with the sequence that generated it. We get this association by aligning the measured signal to the predicted signal for the known DNA sequence being measured ($\Phi X174$ or λ). For the first construction of the empirical model, the predicted signal is given by the initial model described in section 7.2.

Each read of genomic DNA is aligned to the predicted signal for its reference sequence using the BCJR alignment algorithm (Bahl et al. IEEE Trans. Inf. Theory, 1974). The alignment maps the $G - V$ curve at each measured conductance state to a state in the predicted signal, which represents a known location in the reference sequence. In addition to an alignment location, the BCJR algorithm also returns a likelihood that each alignment location is the true alignment location for the measured state. We update the mean values in the 6-mer model by filling each state in the model with the weighted average (weighted by the likelihood score of alignment) of all measured states aligning to locations in the reference corresponding to that enzyme and 6-mer state. Additionally, the covariance of each state in the model is updated with the covariance of all measured states aligning to reference locations corresponding to that state.

The above procedure of generating predictions, aligning reads, and updating the predictions can be iterated (Fig S12). For the work presented here, we ran two iterations: one starting from the interpolated initial model and second aligning to the first version of the empirical model. Though we found that two iterations yielded a good quality model, it is possible that a larger data set of genomic DNA reads combined with further iterations of the model generation could result in an improved model.

7.5 Filling in Unmeasured 6-mers

After constructing the empirical 6-mer model from long reads of $\Phi X174$ and λ DNA, we found that for a small fraction (168 out of 4096) of the 6-mers, one or both of the enzyme states had not been well measured. In order to efficiently measure the remaining states, we used a de Bruijn graph approach (de Bruijn, Koninklijke Netherlandse Akademie v. Wetenschappen, 1946) to construct a minimal sequence of length 337 nt containing all 168 of the poorly measured 6-mers. We then split up this minimal sequence over a total of 6 short synthetic DNA oligos (Fill-in strands 1-6 in Table S1, Fig S13). We ran these 6 strands using standard variable-voltage sequencing conditions, collecting a total of 172 reads comprising 16675 enzyme steps (8338 nucleotides) across the 6 strands. Using these reads, we filled in the remaining gaps in the empirical 6-mer model using the same approach detailed in section S7.4: we predicted the signal for the known sequence based on the existing 6-mer model, aligned the reads to this prediction, then updated the model based on the alignments and iterated the process. We iterated the predict/align/update cycle 10 times for the short strands in order to generate the final version of the 6-mer model which we ultimately use for sequencing.

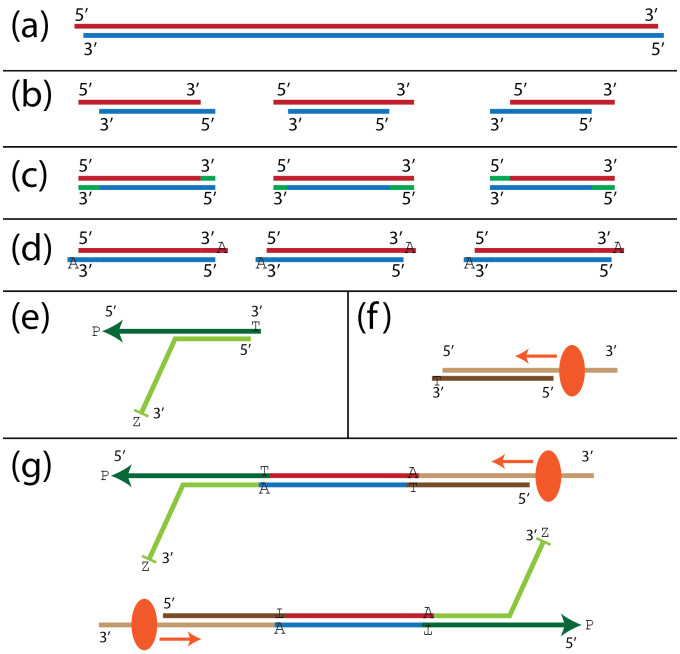


Figure S 11: λ DNA Fragmentation. **(a)** Full length double stranded λ DNA, 48502 bp. **(b)** The genomic DNA is sheared into random fragments of average length 3 kb (miniTUBE) or 6 kb (gTUBE). **(c)** The random 3' and 5' overhangs generated through the shearing are repaired (green segments) using the NEBNext end repair module. **(d)** A single base dA overhang is added to each 3' end using the NEBNext dA-tailing module. **(e)** The threading adapter is composed of two strands. The threading strand (dark green) has a 5' phosphate (P, arrowhead) to facilitate capture by the pore and a single base dT 3' overhang for ligation onto the λ fragment. The cholesterol blocker (light green) is partially complementary to the threading strand, with a non-complementary 3' end, and a terminal 3' cholesterol (Z, crossbar) which inserts into the lipid bilayer to up-concentrate DNA near the pore. **(f)** The loading adapter is also composed of two strands. The loading strand (tan) has an overhanging 3' end where the Hel308 helicase (orange ellipse) can load onto a ss-dsDNA junction, and proceed in a 3' to 5' direction along the strand. The loading blocker (brown) is complementary to the non-overhanging region of the loading strand, with a single base dT 3' overhang for ligation onto the λ fragment. **(g)** Our fully formed DNA constructs are now ready to be run in variable-voltage nanopore sequencing experiments. This library preparation can obtain reads of both the sense (red, top construct) and antisense (blue, bottom construct) strand.

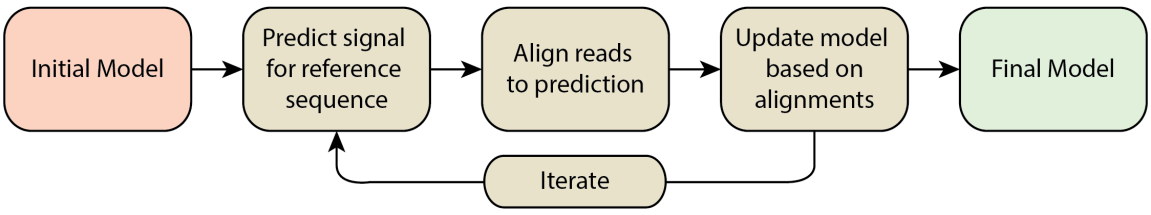


Figure S 12: Iterative map construction flow chart

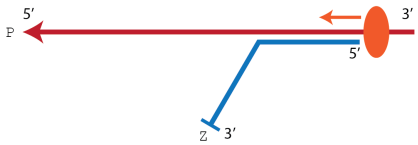


Figure S 13: DNA constructs for the fill-in data set are composed of two partially-complementary short oligos. The fill-in template (one of a possible 6) in red has a 5' terminal phosphate (P, arrowhead) facilitating threading of this end into the pore. The 5' phosphate is followed by a section of the minimal sequence containing the missing 6-mers, then by a target sequence for duplexing the complementary strand, and finally an overhanging non-complementary 8 nucleotide section at the 3' end for loading the Hel308 helicase (orange oval). The fill-in cholesterol blocker (blue) contains at its 5' end the complementary sequence to the target duplex sequence in the template strand. The complementary sequence is followed by a series of 4 18-Carbon spacers and a 3' terminal cholesterol tag (Z, crossbar). The 18-Carbon spacers give the construct a flexible fan-tail, and the terminal cholesterol tag associates with the lipid bilayer, up-concentrating the constructs near the pore.

8 Constant-Voltage Model Extraction

The variable-voltage 6-mer map contains as a subset all the information required for a constant-voltage 6-mer model. In evaluating the performance of the two sequencing methods, we used the constant-voltage 6-mer model extracted from the variable-voltage model to provide a fair test. By doing this, any errors present in one model detracting from the sequencing accuracy will be present in both models, and will affect the accuracy of both methods equally.

The constant-voltage model is extracted from the variable-voltage model as shown in Fig S14. The constant-voltage mean conductance for each 6-mer is extracted from the corresponding variable-voltage conductance curve by taking the value of the curve at the point corresponding to 180 mV (the operating voltage for our constant-voltage sequencing experiments). The variance of the mean constant-voltage conductance is taken as the variance in the value of the variable-voltage conductance curve at that same point.

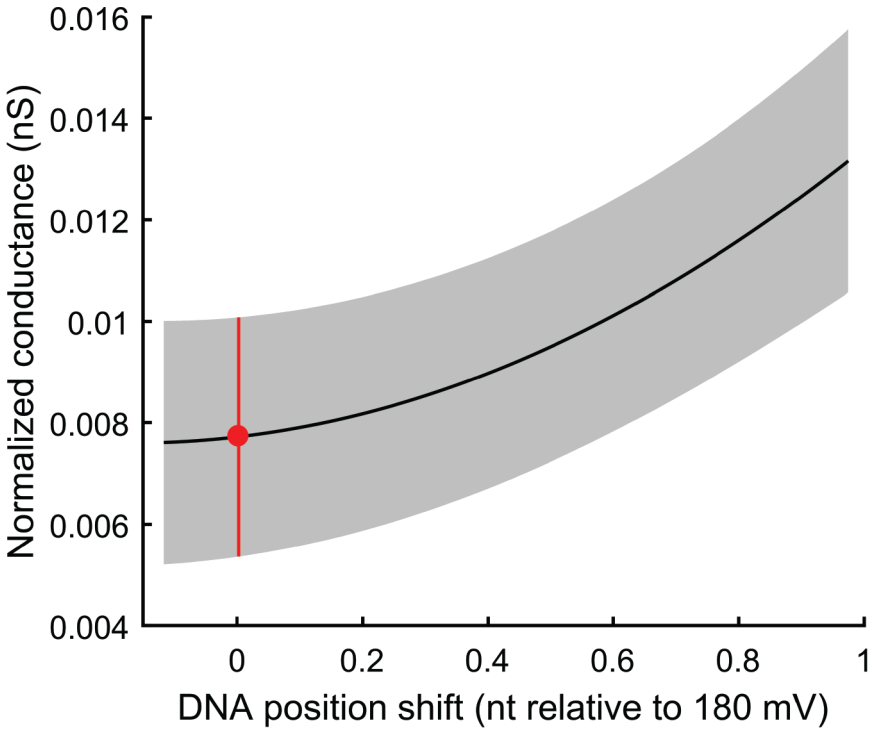


Figure S 14: Constant voltage model extraction. The constant-voltage model value for a given 6-mer (e.g. *ATGAGA*) is taken as the point (red) in the variable-voltage conductance curve for that 6-mer in the variable-voltage model (black) corresponding to 180 mV (0 nt shifted relative to 180 mV). The uncertainty (red line) is taken as the variation in the variable-voltage model prediction (gray shading) at the 180 mV point.

9 Sequencing Verification Experiment

We tested the relative performances of our constant-voltage and variable-voltage sequencing methods by using both methods to sequence DNA from the pET-28a vector. The pET-28a vector was chosen as it represented a readily available genomic DNA sequence and was not involved in our 6-mer model construction, thus avoiding the risk of over-training artificially boosting our sequencing numbers. Given Hel308's limited processivity (Supplemental Note 10), an experiment in which all reads began from the same start point in pET-28a would be unlikely to generate good coverage throughout the sequence and instead concentrate most reads on the same ∼1000 base pairs nearest the start point. To get broad coverage throughout the sequence, and to get reads of both the sense and antisense strands, we fragmented the pET-28a sequence using a double restriction digest. Digestion gave us a variety of 100-1000 base fragments for our sequencing experiments. These fragments were generated, then prepared for Hel308 sequencing experiments as follows.

1. The pET-28a vector was digested using the NspI and Sau3aI restriction enzymes (New England Biolabs). We used 3.5 μL of 10000 $\frac{U}{mL}$ NspI and 7 μL of 5000 $\frac{U}{mL}$ Sau3aI per 17.5 μg of vector DNA. Following digestion, fragments were cleaned on DNA Clean and Concentrator column (Zymo Research).

2. Following digestion, we prepared 4 distinct adapter constructs for ligation. The four constructs were

   (a) Sau3aI threading adapter, composed of the Sau3aI threading strand and the Sau3aI cholesterol blocker (Table S2)

   (b) Sau3aI loading adapter, composed of the Sau3aI loading strand and the Sau3aI loading blocker

   (c) NspI threading adapter, composed of the NspI threading strand and the NspI cholesterol blocker

   (d) NspI loading adapter, composed of the NspI loading strand and the NspI loading blocker

   We require four adapter constructs as each pET-28a fragment needs an threading adapter to facilitate capture into the pore and a loading adapter to facilitate Hel308 loading onto the DNA. Each of the two cutsites needs its own set of loading and threading adapters as the two restriction enzymes leave different sticky-end overhangs. Adapters were prepared individually by mixing equimolar portions of the two constituent oligos and annealing using standard annealing protocols (Supplemental Note 7).

3. We ligated the several adapters to the pET-28a fragments by mixing the fragmented DNA with the annealed adapter constructs in approximately equimolar ratios, then incubating with T4 DNA ligase. Following ligation, the final products were purified using another DNA Clean and Concentrator column.

There are a few important drawbacks to the above-described preparation procedure that must be considered in estimating the overall yield and in conducting downstream analysis. First, due to the palindromic nature of both the NspI and Sau3aI cutsites, we are not guaranteed to correctly get one each of the loading and threading adapters ligated to each fragment. Indeed, 25% of the total fragments will have the correct adapters for a sense strand read, 25% will have the correct adapters for an antisense strand read, and 50% will have either 2 loading adapters or 2 threading adapters and will be unlikely to produce reads. Even with this 50% drop off in the effective yield of this preparation procedure, we were still able to generate plenty of DNA to collect the data needed.

On this same note, the loading and threading adapters for each cutsite are themselves self-complementary at their overhanging sticky ends. This can lead to the formation of so-called adapter dimers, where two adapters ligate to each other. When a loading adapter ligates to a threading adapter, we create a DNA construct that can both load Hel308 and thread into the pore, and so is likely to be read. We see a population of these dimers in our experiments, and discard them from later analysis based on their characteristically short length and recognizable pattern of states.

The final drawback also stems from the palindromic nature of the restriction cutsites. The sticky-end overhangs left on the pET-28a fragments after digestion are self-complementary, which can lead to chimera formation. Chimeras occur when different fragments from disparate parts of the pET-28a reference sequence ligate together. We see a population of these chimeras in our reads. There is nothing intrinsically wrong with the chimera reads, but determining the base calling accuracy for these reads is more difficult. In these cases, we must piece together the ground truth reference sequence by separately aligning the smaller

fragments composing the chimera to find which parts of the reference sequence have been stitched together. The called sequence is then compared against this stitched-together ground truth sequence to evaluate the read accuracy.

10 Hel308 Processivity

The read length in both our constant-voltage and variable-voltage sequencing experiments is limited by the processivity of the Hel308 helicase enzyme we use to control DNA motion through the pore. The enzyme's processivity is the typical number of nucleotides it translocates through the pore before it dissociates from the DNA, ending the event. Processivity can in principle be a function of various experimental conditions, including temperature, substrate and salt concentration, pH, and applied force (i.e. applied voltage). Hel308's activity is insensitive to force over the range of forces (voltages) we apply in our experiments, so its stepping rate and processivity should not change with the variable applied voltage (Craig et al. Proc. Natl. Acad. Sci., 2017).

We observed Hel308's processivity in both the constant- and variable-voltage conditions by looking at the read lengths obtained on our ΦX-174 construct. Specifically, we looked at read lengths on the larger, 5042 bp fragment, as this fragment is long enough that nearly all reads terminated due to the helicase unbinding from the DNA prior to reaching the end of the strand. Based on alignments of the reads to the ΦX-174 reference, we investigated 49 constant-voltage reads and 50 variable-voltage reads of the long fragment, all starting at the same location in the genome (at the AvaII cut site, Supplemental Note 7.3.1). Hel308 shows little ability to unwind dsDNA in our experimental conditions, so all reads began at the loading site and only progressed once the duplex strand had been sheared away by the pore. The read survival fraction as a function of read length was calculated as the number of reads reaching a given position in ΦX-174 over the total number of reads. We found that the survival fraction f as a function of read length l was well modeled by a single exponential function of the form $f(l) = e^{\frac{l}{l_p}}$, where l_p is the characteristic processivity of the enzyme (Fig S15). The single-exponential form of the survival fraction indicates that Hel308 dissociation from the ssDNA track in our experiments is dominated by a single rate-limiting step. From our data, we found a best fit processivity of $l_p = 999 \pm 174$ nt for the constant-voltatge data and $l_p = 933 \pm 132$ nt for the variable-voltage data. The processivity in the two conditions is identical within statistical uncertainty, meaning the change from constant to variable voltage has no effect on read length.

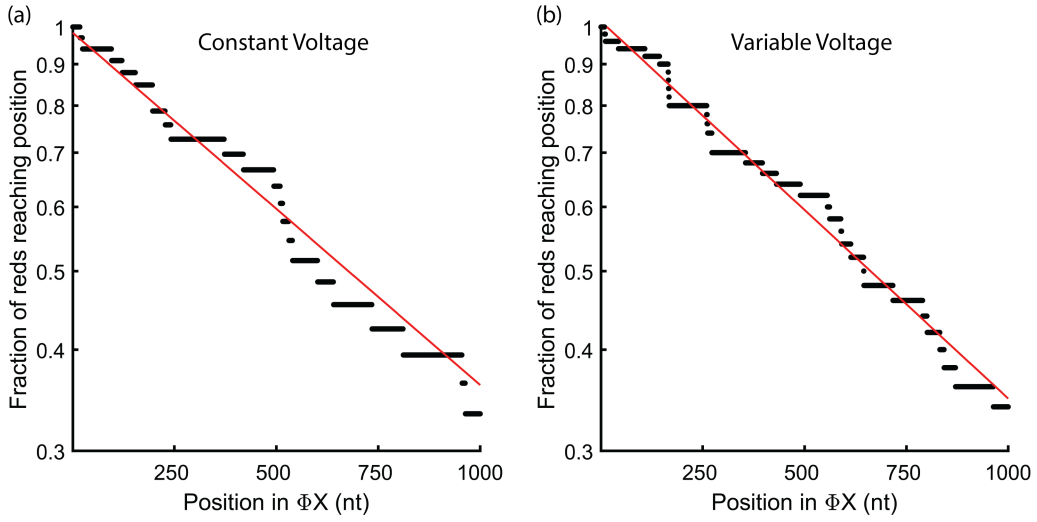


Figure S 15: Hel308 processivity in constant voltage (a) and variable voltage (b) sequencing conditions. The fraction of reads (all starting from the same cut site in ΦX-174) reaching a given position in the genome is plotted as black dots on a logarithmic y-scale. We see an exponential fall-off in read survival, indicating a read-termination process dominated by a single off-rate. The red line shows the best-fit single exponential model of the form $f(l) = e^{\frac{l}{l_p}}$. The best fit for constant voltage is given by $l_p = 999$ nt, the best fit for variable voltage is given by $l_p = 933$ nt. The processivity in the two conditions is identical within statistical uncertainty.

11 Change Point Detection Algorithm

11.1 Basic Description

In both constant-voltage and variable-voltage sequencing, our first step is to partition the raw time-series ionic current data into segments corresponding to enzyme steps. Partitioning simplifies the data stream passed to the hidden Markov model by turning the many noisy measurements making up an enzyme step observation into a series of a few low-noise parameters describing each step (Fig S16). In the case of constant-voltage sequencing, each enzyme step is described by a mean ionic current and an associated variance. For variable-voltage sequencing, we use the coefficients of the top three principal components (Supplemental Note 3), along with their associated covariance.

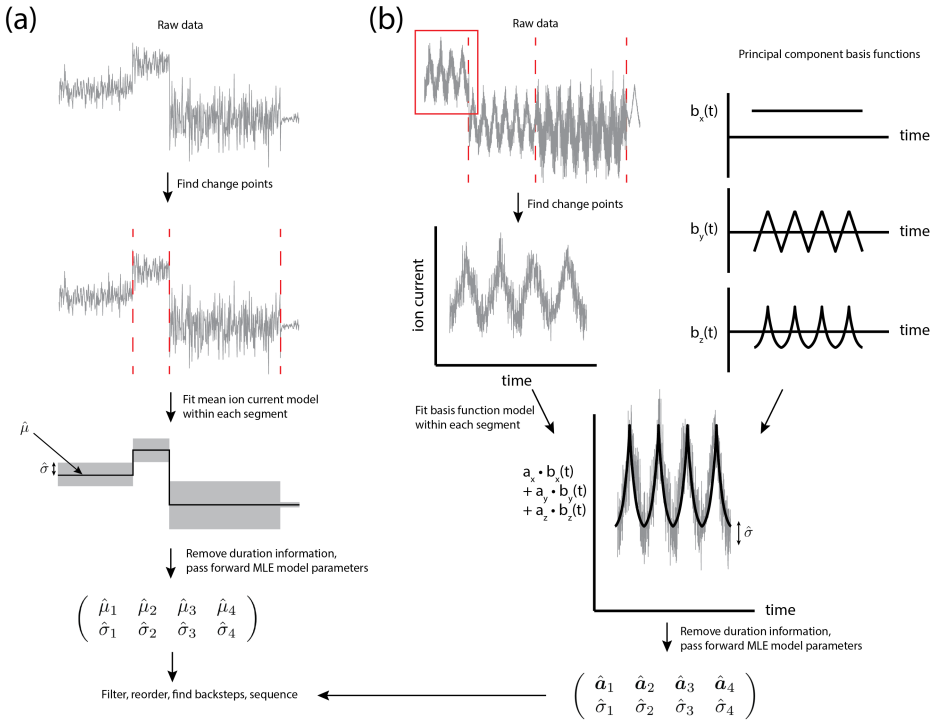


Figure S 16: **(a)** For constant-voltage experiments, change point detection involves finding transitions between states of different ionic current means and variances. **(b)** For variable-voltage experiments, we are looking for changes in the parameters of a time-dependent model describing the data. The model we use is a sum of basis functions which are the principal components of the nanopore signal, parameterized by the amplitude of each basis function.

The data is partitioned into enzyme steps using a change point detection algorithm (algorithm S4). The same fundamental algorithm works for both constant-voltage and variable-voltage sequencing data. Simply, the change point algorithm chooses between two competing hypotheses. Given a segment of data $\{x_i\}$, is the data best modeled by a single model (parameterized as θ_T) or by two models (θ_L, θ_R) each separately describing the data to the left and right of some transition point t? If the single-model hypothesis proves better, no change point is present in the segment. If the two-model hypothesis is better, a change point is called at the best transition point.

The basic considerations in this type of algorithm are how to model the data, and how to prevent over-calling transitions. In the case of constant-voltage data, we use a mean ionic current and a variance to describe the individual states. We modeled the variable-voltage data by the five largest principal components of the periodic functions that represent the raw data of each enzyme state (Fig S17). These principal components were determined by choosing change points by-eye, then averaging each enzyme state into a single 250-

sample period of the waveform We then treated each state as a separate measurement for the purposes of principal component analysis. The principal components provide a descriptive, concise basis with which we can describe the variable-voltage time series data.

The over-calling issue is a consequence of the fact that a model with more parameters can always describe a data set better than a model with fewer parameters, even if it is not actually more predictive. Consequently, the two-model hypothesis will always fit the data better–the question is rather is it sufficiently better to justify the addition of more parameters into our description of the data? We correct this bias by penalizing the addition of extra parameters, using the results of Lamont and Wiggins (Lamont, Wiggins, Neural Comput. 2016) to determine the appropriate penalty.

Algorithm S 4 Change point detection

Input:
d-dimensional data $\{x_i\}$, $i \in \{1 : N\}$
Transition threshold $\mathcal{T}$
Initialize: $\{t\} \leftarrow []$ ▷ Initialize an empty list of transition points
function PARTITION($\{x_i\}, \mathcal{T}$, $\{t\}$)
    Score: assign a score $\mathcal{S}_i$ for the placement of a transition at each point $i \in \{1 : N\}$
    $\mathcal{S}_{best} \leftarrow max(\{\mathcal{S}_i\})$
    $t_{best} \leftarrow i$ such that $\mathcal{S}_i = \mathcal{S}_{best}$
    if $\mathcal{S}_{best} > \mathcal{T}$) **then** ▷ The best transition point is good enough to call a transition
        $\{t\}[end] \leftarrow t_{best}$ ▷ Add the found transition to the growing list
        $\{t\} \leftarrow$ PARTITION($\{x_i\}$ $i \in \{1 : t_{best}\}$, $\mathcal{T}$, $\{t\}$) ▷ Recursively find partitions in the data to the left of the found transition point
        $\{t\} \leftarrow$ PARTITION($\{x_i\}$ $i \in \{t_{best} : N\}$, $\mathcal{T}$, $\{t\}$) ▷ Recursively find partitions in the data to the right of the found transition point
    else
        Output: $\{t\}$
    end if
end function

11.2 Mathematical Description

The following is a full mathematical description of the change point detection procedure.

The change point problem is formulated mathematically as follows: given a time series of d-dimensional data $\{x_1, x_2, \ldots, x_N\}$, $x \in \mathbb{R}^k$, choose a model consisting of some number of change points $\{t_a, t_b, ...\}$, and a different set of parameters $\{\theta_a, \theta_b, ...\}$ describing the data between each change point. Our change point detection algorithm (algorithm S4) finds a close-to-optimal partitioning of time series data using this model.

We assume each state is a function f of time t and parameters θ with normally distributed noise σ. Under these assumptions, the probability density of obtaining a measurement $x(t)$ at a time t given a choice of parameters θ for that time is

$$p\left(x(t), t \,|\, \theta\right) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(f(t;\theta)-x(t))^2}{2\sigma^2}}$$

For a number of measurements indexed by time $t = 1, 2, 3, ...$, the probability density is the product of the probabilities of each measurement:

$$p\left(x \,|\, \theta\right) = \prod_{t=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(f(\theta)_t - x_t)^2}{2\sigma^2}}$$

We convert this probability into a log-likelihood $L(\theta \,|\, x) = p(x \,|\, \theta)$ to simplify calculations, giving

$$\log \mathcal{L} = -\frac{1}{2} \sum_{t=1}^{N} \log 2\pi\sigma^2 + \frac{(f(\theta)_t - x_t)^2}{\sigma^2}$$

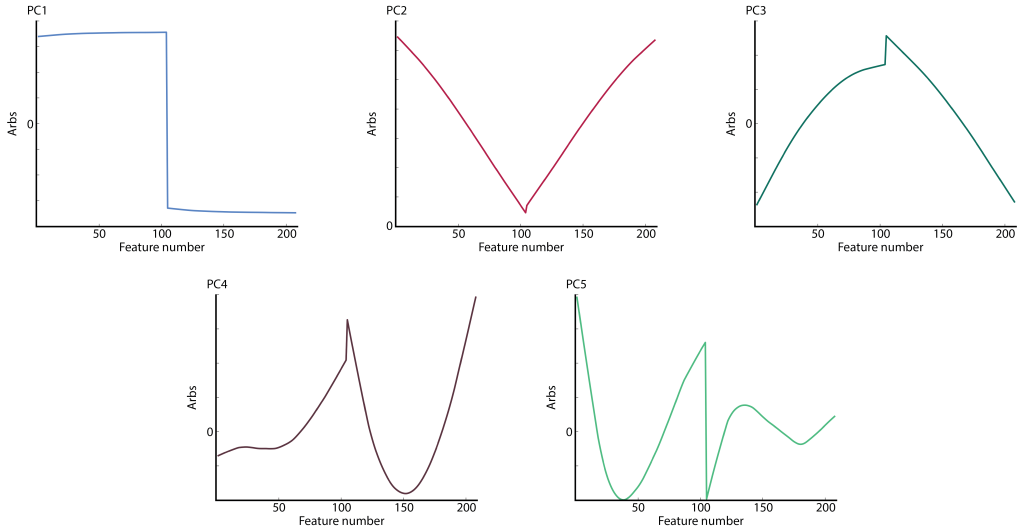


Figure S 17: Principal components for change point detection. The five principal component vectors used to model the variable-voltage time series date are shown. Linear combinations of these five vectors can describe the observed data.

For change point detection, we are interested in the relative likelihood between using two different sets of parameters θ_L and θ_R to model the data to the left and right of a possible change point, versus using one set of parameters θ_T to describe the total region in question. Defining the first time index of the region as L, the final index as R, and the number of points to the left, the right and in the whole region as N_L, N_R, and $N_T = R - L + 1$ respectively, the relative log-likelihood is

$$\log \mathcal{L} = -\frac{1}{2}\left[\sum_{t=L}^{L+N_L-1} \log 2\pi\sigma_L^2 + \frac{(f(\theta_L)_t - x_t)^2}{\sigma_L^2} + \sum_{t=L+N_L}^{N_T} \log 2\pi\sigma_R^2 + \frac{(f(\theta_R)_t - x_t)^2}{\sigma_R^2} - \sum_{t=L}^{L+N_T-1} \log 2\pi\sigma_T^2 + \frac{(f(\theta_T)_t - x_t)^2}{\sigma_T^2} \right]$$

If our maximum likelihood estimate for θ given the data is $\hat{\theta}$, the residual variance is $\hat{\sigma}^2 = \frac{1}{N}\sum_t (f(\hat{\theta})_t - x_t)^2$. We find the maximum log-likelihood $\log \hat{\mathcal{L}}$ by plugging in these estimators:

$$\log \hat{\mathcal{L}} = -\frac{1}{2}\left[N_L \log 2\pi\hat{\sigma}_L^2 + N_L \frac{\hat{\sigma_L}^2}{\hat{\sigma}_L^2} + N_R \log 2\pi\hat{\sigma}_R^2 + N_R \frac{\hat{\sigma}_R^2}{\hat{\sigma}_R^2} - N_T \log 2\pi\hat{\sigma}_T^2 - N_T \frac{\hat{\sigma}_T^2}{\hat{\sigma}_T^2} \right]$$

$$= -\frac{1}{2}\left[N_R \log \hat{\sigma}_R^2 + N_L \log \hat{\sigma}_L^2 - N_T \log \hat{\sigma}_T^2 \right]$$

This is the correct expression for the log-likelihood of a fit, giving the most *descriptive* model of the data. However, we are not interested in the most *descriptive* but rather the most *predictive* model. To find this, we need to correct for the tendency to over-fit. We can always fit better by partitioning data and supplying more parameters, but we lose information by doing so. This leads to an over-fitting bias. To correct for this, we use the results of LaMont and Wiggins (Lamont, Wiggins, Neural Comput. 2016) to subtract this bias. In general, the bias is a function of the number of points N_T and the dimensionality of the data being partitioned d, and is calculated through Monte Carlo simulations and either fitted or used as a lookup table. The test statistic is

$$\text{CPIC} = -\log \hat{\mathcal{L}} + p(N, d) = \frac{N_R}{2}\log \hat{\sigma}_R^2 + \frac{N_L}{2}\log \hat{\sigma}_L^2 - \frac{N_T}{2}\log \hat{\sigma}_T^2 + p_d(N_T)$$

where $p_d(N_T)$ is the penalty for adding parameters in modeling N_T d-dimensional data points. The natural choice is to call a level transition if $\text{CPIC}_p < 0$. A simple way to tune the sensitivity of this score is to apply a multiplier $\lambda > 0$ to p_d, which can be made higher to increase the penalty and find fewer levels. This is done to compensate for a model that does not exactly describe the data; we choose $\lambda = 4$ because it provides empirically good results. So the final score used is

$$\text{CPIC}(\lambda) = \frac{N_R}{2}\log\hat{\sigma}_R^2 + \frac{N_L}{2}\log\hat{\sigma}_L^2 - \frac{N_T}{2}\log\hat{\sigma}_T^2 + \lambda p_d(N_T)$$

To calculate the $\hat{\sigma}$'s, we need to determine the maximum likelihood estimates of the model parameters $\hat{\theta}$. Obtaining these can in general be slow and difficult, possibly even requiring nonlinear optimization not guaranteed to converge. However, in certain situations it is easy, and we can even take advantage of some tricks to avoid redundant calculation. The simplest example is the case of constant levels about a single mean. The maximum likelihood estimate of the mean in bounds $[L, R]$ is

$$\hat{\mu} = \frac{1}{R-L+1}\sum_{t=L}^{R} x_t$$

We can avoid continually re-adding the same points together by instead defining and pre-calculating the cumulate $X_t = \sum_{s=1}^{t} x_s$, in which case our expression for the mean is simply

$$\hat{\mu} = \frac{X_R - X_L}{R-L+1}.$$

This difference is much more expedient to calculate than the mean, and the calculation of its value for many possible transition points may be vectorized. We can use a similar technique to calculate the variance,

$$\hat{\sigma}^2 = \frac{1}{R-L+1}\sum_{t=L}^{R}(x_t - \hat{\mu})^2 = \left[\frac{1}{R-L+1}\sum_{t=L}^{R} x_t^2\right] - \hat{\mu}^2$$

Again defining and pre-calculating the cumulate sum

$$X_t^2 = \sum_{s=1}^{t} x_s^2,$$

we quickly calculate the MLE variance as

$$\hat{\sigma}^2 = \frac{X_R^2 - X_L^2}{R-L+1} - \hat{\mu}^2.$$

In general, the function $f(\theta)$ may depend on time. One case is if we can write $f(\theta)_t$ as a sum of p basis functions b_{it} with amplitudes θ_i,

$$f(\theta)_t = \sum_{i=1}^{p} \theta_i b_{it}.$$

Assuming again normally distributed random errors, we find maximum likelihood estimators

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \sum_{t=L}^{R}\left(x_t - \sum_{i=1}^{p}\theta_i b_{it}\right)^2$$

To this end, we set the derivative of the sum squared error to zero,

$$2\sum_{t=L}^{R}\left(x_t - \sum_{i=1}^{p}\hat{\theta}_i b_{it}\right) b_{jt} = 0$$

$$\sum_{t=L}^{R} x_t b_{jt} = \sum_{i=1}^{p}\hat{\theta}_i \sum_{t=L}^{R} b_{it} b_{jt}$$

Both of these sums over time are again precalculable from cumulates, which we define as

$$B_{ijt} = \sum_{s=1}^{t} b_{it} b_{jt} \text{ (note: } B_{ijt} \text{ is symmetric in } i \text{ and } j.)$$

$$c_{it} = \sum_{s=1}^{t} x_t b_{it}$$

Then, in vector notation, interpreting $\mathbf{c}_t$ as the vector with elements $(c_{1t},\, c_{2t},\, \ldots,\, c_{pt})$, $\boldsymbol{\theta}$ as $(\theta_1,\, \theta_2,\, \ldots,\, \theta_p)$, and B_t as the matrix with B_{ijt} as the element at row i and column j, the expression for $\hat{\theta}$ becomes

$$[\mathbf{c}_R - \mathbf{c}_L]^T = \hat{\boldsymbol{\theta}}^T [B_R - B_L]^T$$

$$[\mathbf{c}_R - \mathbf{c}_L] = [B_R - B_L]\, \hat{\boldsymbol{\theta}}$$

$$\hat{\boldsymbol{\theta}} = [B_R - B_L]^{-1} [\mathbf{c}_R - \mathbf{c}_L]$$

We can now calculate σ on that domain to be used in the CPIC calculation. The sum of squared errors is

$$\hat{\sigma}^2 = \frac{1}{R-L+1} \sum_{t=L}^{R} \left[x_t - \sum_{i=1}^{p} \hat{\theta}_i b_{it} \right]^2.$$

Expanding the squared term,

$$\hat{\sigma}^2 = \frac{1}{R-L+1} \sum_{t=L}^{R} \left[x_t^2 - 2 \sum_{i=1}^{p} \hat{\theta}_i x_t b_{it} + \sum_{i,j=1}^{p} \hat{\theta}_i b_{it} b_{jt} \hat{\theta}_j \right].$$

Plugging in our expression for $\hat{\theta}_i$,

$$\hat{\sigma}^2 = \frac{1}{R-L+1} \left[\sum_{t=L}^{R} x_t^2 - 2 \sum_{i,j=1}^{p} [B_R - B_L]^{-1}_{ij} [c_R - c_L]_j \sum_{t=L}^{R} x_t b_{it} \right.$$
$$\left. + \sum_{i,j,k,l=1}^{p} [B_R - B_L]^{-1}_{ik} [c_R - c_L]_k \left(\sum_{t=L}^{R} b_{it} b_{jt} \right) [B_R - B_L]^{-1}_{jl} [c_R - c_L]_l \right].$$

Defining one more cumulate $X_t^2 = \sum_{s=1}^{t} x_t^2$ and plugging in this as well as other cumulate expressions,

$$\hat{\sigma}^2 = \frac{1}{R-L+1} \left[X_R^2 - X_L^2 - 2 \sum_{i,j=1}^{p} [B_R - B_L]^{-1}_{ij} [c_R - c_L]_j [c_R - c_L]_i \right.$$
$$\left. + \sum_{i,j,k,l=1}^{p} [B_R - B_L]^{-1}_{ik} [c_R - c_L]_k [B_R - B_L]_{ij} [B_R - B_L]^{-1}_{jl} [c_R - c_L]_l \right].$$

$$\hat{\sigma}^2 = \frac{1}{R-L+1} \left[X_R^2 - X_L^2 - 2 \sum_{i,j=1}^{p} [B_R - B_L]^{-1}_{ij} [c_R - c_L]_j [c_R - c_L]_i \right.$$
$$\left. + \sum_{i,j=1}^{p} [B_R - B_L]^{-1}_{ik} [c_R - c_L]_j [c_R - c_L]_i \right].$$

$$\hat{\sigma}^2 = \frac{1}{R-L+1}\left[X_R^2 - X_L^2 - \sum_{i,j=1}^{p} [c_R - c_L]_i \left[B_R - B_L\right]_{ij}^{-1} [c_R - c_L]_j\right]$$

Or, in vector notation,

$$\hat{\sigma}^2 = \frac{1}{R-L+1}\left[X_R^2 - X_L^2 - [\mathbf{c}_R - \mathbf{c}_L]\left[B_R - B_L\right]^{-1}[\mathbf{c}_R - \mathbf{c}_L]\right]$$

At every possible division point we must invert a unique matrix, but these matrices are small, and with a reasonably small number of basis functions applying this algorithm is not too slow. For the variable-voltage data, we used the five largest principal components of the periodic ionic current signal, as described above.

12 Capacitance Compensation

The bilayer separating the *cis* and *trans* wells acts as a capacitor. When operating the nanopore sequencer at constant voltage, the capacitor's presence in the circuit is unimportant. However, when operating using a time-varying voltage, the capacitor introduces an additional charging and discharging ionic current I_{cap} which must be removed from the signal I_{sig} we wish to observe. Thus, rather than directly reading out the sequence-dependent ionic current signal, the observed ionic current I_{obs} takes the form $I_{obs} = I_{sig} + I_{cap}$

I_{cap} depends on both the size of the capacitor formed by the bilayer (a constant value over the course of the experiment) and the size of the resistor formed by the pore and the translocating DNA (which varies as a function of the sequence present within the pore). Because the resistance is different at each ionic current state, capacitance compensation is conducted separately for each ionic current state.

As I_{cap} is proportional to the rate of change of the voltage $\frac{dV}{dt}$, our triangle wave applied voltage causes an in-phase square wave capacitive current, plus decaying exponential contributions around the ill-defined regions of $\frac{dV}{dt}$ when the voltage transitions from up-slope to down-slope and back. The goal of our capacitance compensation procedure is to infer the I_{cap} from the asymmetry between the current values during the up-slope and down-slop voltage ramps, then subtract out this inferred signal to reveal I_{sig}.

The procedure is as follows.

1. The overall phase of the signal is calculated from the applied voltage signal for the entire read. Knowing the overall phase, along with the number of data points collected per voltage cycle (50 kHz sampling rate, with the voltage cycling at 200 Hz gives 250 points per cycle) allows us to assign an identification index between 1 and 250 to each point in the ionic current trace $I(t)$ marking its phase.

2. For each ionic current state, all data points in the ionic current trace $I(t)$ are grouped by their previously determined identification index, thus binning together all data points collected at the same location in the voltage sweep. For each ionic current state, the ionic current trace $I(t)$ is divided into "up-slope" and "down-slope" based on the identification index previously determined (Fig S18a,b).

3. For both the up-slope and down-slope data, we group and average all data points with the same identification index, thus finding the average ionic current value at each location in the voltage cycle. This yields the average current-voltage $(I - V)$ characteristic for both up and down: $I_{up}(V)$ and $I_{down}(V)$ (Fig S18c).

4. Taking the difference between the two $I - V$ curves, we get the asymmetry between the sweeps, $H(V) = I_{down}(V) - I_{up}(V)$ (Fig S18d).

5. To find the magnitude of the square wave component in the capacitive signal, which appears as a systematic offset m between the up and down $I - V$ curves, we fit a parabola to the residual, $H(V)$, over the second and third quartiles in the voltage (125 to 175 mV). The x-coordinate of the parabola's vertex is constrained to occur at the voltage midpoint (150 mV), and the y-coordinate is taken as the systematic offset m. Low and high voltages are omitted in order to isolate the offset, without interference from the sharp spikes appearing near the voltage turnaround points. A parabolic fit is used in lieu of a mean, as $H(V)$ exhibits some curvature even over the middle voltage quartiles due to the decaying exponential current spikes generated at the voltage turnaround points.

6. Capacitive correction functions for up and down ($Corr_{up}(V)$ and $Corr_{down}(V)$) are generated from the left and right halves of the residual function $H(V)$ (Fig S18e, f). The residual function is split around the midpoint voltage of the sweep V_{mid}, and the correction functions are given by:

   For $V < V_{mid}$,

   $$Corr_{up}(V) = H(V) - \frac{m}{2}$$

   $$corr_{down}(V) = -\frac{m}{2}$$

   And for $V > V_{mid}$

   $$Corr_{up}(V) = \frac{m}{2}$$

$$Corr_{down}(V) = \frac{m}{2} - H(V)$$

Splitting the correction in this way attributes the spike at low voltage to the up-sweep and the spike at high voltage to the down-sweep. The overall offset m is attributed equally to both sweep directions. This assignment is justified, as the capacitive effect of an instantaneous change in $V(t)$ falls off exponentially, with time constant RC. As the low voltage turnaround immediately precedes the up-slope region, the effect of this turnaround is strong in the up-slope, but negligible by the down slope. The opposite is true for the high voltage turnaround. The overall offset is the manifestation of the square wave current generated by the constant $\frac{dV}{dt}$ throughout the rest of the triangle wave, and so appears equally in both up-slope and down-slope curves.

7. Applying the up and down correction functions to their respective $I - V$ curves gives the corrected curves I^{cc}_{up} and I^{cc}_{down}:

$$I^{cc}_{up} = I_{up}(V) + Corr_{up}(V)$$

$$I^{cc}_{down} = I_{down}(V) + Corr_{down}(V)$$

The corrected curves show no residual hysteresis, and the spikes around the turnarounds have been eliminated (Fig 18g).

8. Lastly, the correction is applied to all $I(t)$, at each point according to the identification index previously determined. This yields the capacitance compensated $I(t)$ trace that will be used in all further analysis (Fig S18h).

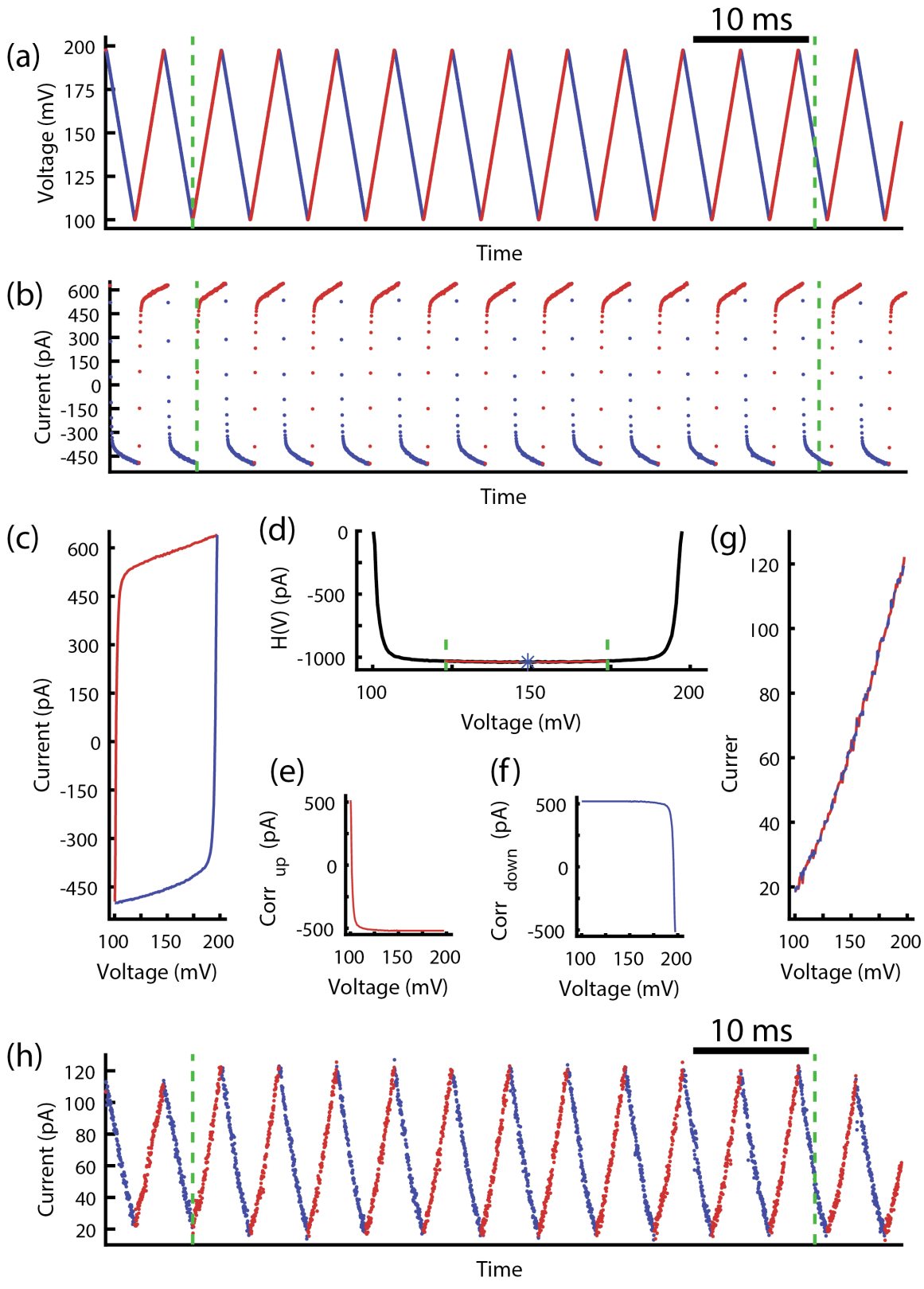

(a)
10 ms
Voltage (mV)
200
175
150
125
100
Time
(b)
Current (pA)
600
450
300
150
0
-150
-300
-450
Time
(c)
Current (pA)
600
450
300
150
0
-150
-300
-450
100 150 200
Voltage (mV)
(d)
H(V) (pA)
0
-500
-1000
100 150 200
Voltage (mV)
(e)
Corr up (pA)
500
0
-500
100 150 200
Voltage (mV)
(f)
Corr down (pA)
500
0
-500
100 150 200
Voltage (mV)
(g)
Currer
120
100
80
60
40
20
100 150 200
Voltage (mV)
(h)
10 ms
Current (pA)
120
100
80
60
40
20
Time

Figure S 18: Capacitance compensation. **(a)** Voltage time series for a single variable-voltage ion current state. Points at which the voltage is increasing are marked red, those at which the voltage is decreasing are marked blue. Dashed green lines mark the beginning and end of the ionic current state. **(b)** Raw ionic current time series for the single ionic current state in (a). Again, red points mark where the voltage is increasing, blue mark where the voltage is decreasing. Green dashed lines mark the beginning and end of the ionic current state. **(c)** Raw ionic current vs. voltage curve for the above ionic current state. Individual cycles have been averaged together. Red shows the average up-slope curve, blue the average down-slope curve. **(d)** Residual function $H(V)$ for the above ionic current state. Black shows the residual as a function of voltage. Green dashed lines mark the first and third quartiles in the voltage over which the quadratic fit is calculated. Red shows the quadratic fit to these data. The blue asterisk marks the calculated vertex. **(e, f)** Calculated correction functions to be added in to the up-slope (e) and down-slope (f) $I - V$ curves. **(g)** Corrected up-slope (red) and down-slope (blue) $I - V$ curves. Dashed lines are used as both curves lie directly on top of one another after the capacitance compensation has removed all hysteresis. **(h)** Corrected current time series for the above ionic current state. Up- and down-slopes are marked in red and blue; dashed green lines mark the beginning and end of the ionic current state.

13 DNA Sequences

Tables 1 and 2 contain a list of the short custom DNA sequences used in our sequencing and DNA stretching experiments. In addition to these short sequences, we used the λ phage (Promega) and ΦX-174 genomes (New England Biolabs) as well as the pET-28a vector (from collaborators).

DNA Construct Name	**Sequence (5'->3')**
ΦX174 Experiments	1 6 11 16 21 26 31 36 41 46 51
ΦX174 threading strand	PXAAA AAAAC CTTCC XCCTT CCCAT CATCA TCAGA TCTCA_CGCGG TGCA
ΦX174 cholesterol blocker	PCCGC GTGAG ATCTG AAAAA TTTAA ACCCA AAXZ
ΦX174 loading strand	PGACC CGCCA AGTAC AAGTA AGCCT ACGCC TACGG TTTTT TTTTT TTTTT TTTTT
ΦX174 loading blocker	CCGTA GGCGT AGGCT TACTT GTACT TGGCG G
λ-phage Experiments	1 6 11 16 21 26 31 36 41 46 51
λ threading strand	PTACT ACTAC TACTA CTACX XTTTT GAGCC TCTCA CTATC GCATT CTCAT GCAGG T
λ cholesterol blocker	PCCTG CATGA GAATG CGATA GTGAG ATCGT AGCCQ QQQZ
λ loading strand	PGGAC GTACT CTTAC GCTAT CACTC TTCGT AGCC
λ loading blocker	AGAGT GATAG CGTAA GAGTA CGTCC T
Missing 6-mer Experiments	1 6 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86 91 96 101 106 111 116 121 126 131 136 141 146 151 156
Fill-in template 1	PTACT ACTAC TACTA CTACX XTTTT TTGGC GCTTC ATACA GCCGC GCCGG CGAGA TTTTG GCGAG ACAGG CACGC GCGAG CCCAA TCTAT TTTCA ATCTA CGTAT ACTAG GGGGT TCTAG TACTT TTTCT CACTA TCGCA TTCTC ATGCA GGTCG TAGCC
Fill-in template 2	PTACT ACTAC TACTA CTACX XTTTT CTAGT ACACT AGACT AGTCC CTACT ACGAT TTTTC TACGA TTAGG GCCCT ATCTA ATCTA GAGTT TTTCT AGAGT AGGGA CCCCG GGACT CCCTT GTATT TTTCT CACTA TCGCA TTCTC ATGCA GGTCG TAGCC
Fill-in template 3	PTACT ACTAC TACTA CTACX XTTTT CCTTG TAGAT CCTAT ACGGA CGGGG TCTCT TTTTG GTCTC TAGCG CTCGA ATGTG TCGAC ACCTT TTTGA CACCT CAGAG ACCTA GCTAG GCTAG TGTTT TTTCT CACTA TCGCA TTCTC ATGCA GGTCG TAGCC
Fill-in template 4	PTACT ACTAC TACTA CTACX XTTTT CTAGT GTACA CCTCG GACCG GTGCC CTCGA TTTTC CCTCG AGAGG ACCAT GCTAG CCCCC CGCTT TTTCC CCGCT ATACA AGTAC CCGAG TTAGA ACTTT TTTCT CACTA TCGCA TTCTC ATGCA GGTCG TAGCC
Fill-in template 5	PTACT ACTAC TACTA CTACX XTTTT TAGAA CTAGG ATAGG GTGGG GCACA TACCT TTTTC ATACC TAGGT CCGAA TCGAT CTTAG CCTAT TTTTA GCCTA AGGGT AGACG TGATT GGGCC TACTT TTTCT CACTA TCGCA TTCTC ATGCA GGTCG TAGCC
Fill-in template 6	PTACT ACTAC TACTA CTACX XTTTT CATAC GTAGC ATTTT TCATA GGCCC CATTT TTCAT CCCGC GCATT TTTCA TCCTA CGCAT TTTTT CTCAC TATCG CATTC TCATG CAGGT CGTAG CC
Fill-in cholesterol blocker	CCTGC ATGAG AATGC GATAG TGAGA QQQQZ
Legend: P = phosphate, Q = 18 carbon spacer, X = abasic site, Z = cholesterol tag	

Table S 1: Table of short DNA sequences used for constructing the variable-voltage 6-mer model.

DNA Construct Name	Sequence (5′->3′)
pET28a Sequencing Experiments	1 6 11 16 21 26 31 36 41 46 51 56 61 66 71
Sau3aI threading strand	PTACT ACTAC TACTA CTACT ACTAC TACTA CXXTT TTATT GAAGT GCAGT ACTTT ACTAA TTATT GCTTT T
Sau3aI cholesterol blocker	PGATC AAAAG CAATA ATTAG TAAAG TACTG CACTT CAATQ QQQZ
Sau3aI loading strand	PGATC ATTGA AGTGC AGTAC TTTAC TAATT ATTGC TTTTT CTGAG CC
Sau3aI loading blocker	AAAAG CAATA ATTAG TAAAG TACTG CACTT CAAT
NspI threading strand	PTACT ACTAC TACTA CTACT ACTAC TACTA CXXTT TTATT GAAGT GCAGT ACTTT ACTAA TTATT GCTTT TCATG
NspI cholesterol blocker	PAAAA GCAAT AATTA GTAAA GTACT GCACT TCAAT QQQQZ
NspI loading strand	PATTG AAGTG CAGTA CTTTA CTAAT TATTG CTTTT TCTGA GCC
NspI loading blocker	AAAAG CAATA ATTAG TAAAG TACTG CACTT CAATC ATG
DNA Stretching Experiments	1 6 11 16 21 26 31 36 41 46 51 56 61 66 71 76 81 86 91 96 101 106 111 116 121 126 131 136 141 146 151 156
Stretching read strand	PTACT ACTAC TACTA CTACX XTTTT TTCTG CAGTG CAGTT TGCTT TGCGA AATGA AACTG CAGTT TGCGA AATTG CAGTT TGCGA AAGTT TGCTG CACGA AATTT TGCTG CAGGA AATTT TTTTC TCACT ATCGC ATTCT CATGC AGGTC CGAGC C
Stretching cholesterol blocker	CCTGC ATGAG AATGC GATAG TGAGA QQQQZ
Legend: P = phosphate, Q = 18 carbon spacer, X = abasic site, Z = cholesterol tag	

Table S 2: Table of short DNA sequences used for measuring DNA stretching and for validating variable-voltage sequencing performance.

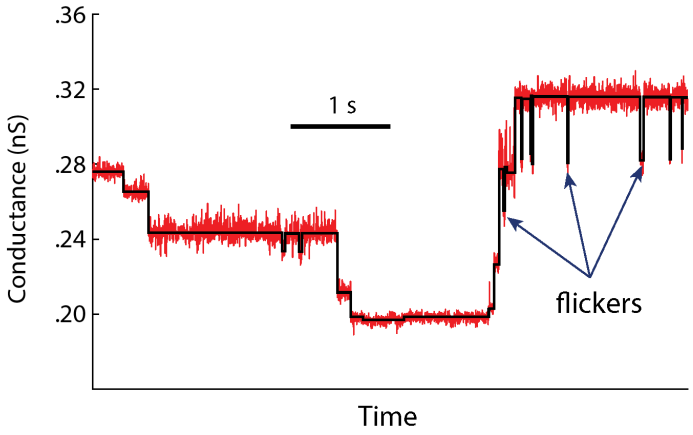


Figure S 19: Example of flicker states. The raw ionic current trace (downsampled to 5 kHz) for a Hel308-controlled DNA translocation event is shown in red, with the states found by the change point detection algorithm overlaid in black. The arrows identify several flickers– transient decreases in the ionic current that are not cannot be mapped to any sequence state.

14 Flicker Filter

In Hel308-controlled DNA translocation data, we observe short-lived states of a particular character which we refer to as "flickers". These states are milliseconds or less in duration, always have a lower conductance than the state they start from, and always return to the state they started in. These flicker states cannot be mapped to any predicted conductance state when reads are compared against the predicted signal for the known DNA sequence, and are thus not informative in decoding the DNA sequence. We remove these flickers prior to any data processing (including change point detection) as their presence decreases the performance and accuracy of downstream.

In variable-voltage data, flickers are easily identified and removed by the removal filter (Supplemental Note 5.1) as their conductance curves look starkly different from normal data. To remove these transients from the constant-voltage data, we search for outlying low states of short duration. We score every individual conductance measurement x_n in a read with a one-sample t-test against the data surrounding it:

$$t_n = \frac{x_n - \mu_{[n-k,n+k]}}{\sigma_{[n-k,n+k]}}$$

where

$$\mu_{[n-k,n+k]} = \frac{1}{2k}\left[\left(\sum_{m=n-k}^{n+k} x_m\right) - x_n\right]$$

and

$$\sigma^2_{[n-k,n+k]} = \frac{1}{2k}\left[\left(\sum_{m=n-k}^{n+k} x_m^2\right) - x_n^2\right] - \mu^2_{[n-k,n+k]}$$

are the mean and variance of the data k points to the left and right of the point being scored, not including the point itself. We discard any points $|t_n| > e$, where e is a threshold chosen to specify the desired aggressiveness of the filter. This procedure is iterated with a fixed threshold e and window k until no more points are removed, then repeated a second time with a larger window k. In the constant-voltage sequencing data in this work, we use $e = 3$ for both iterations and $k = 2$ for the first iteration and $k = 5$ for the second. Filtering is done on the 5 kHz time series data.

15 Experimental Statistics

Statistics for the variable-voltage and constant-voltage experiments conducted to generate the 6-mer model, validate the performance of variable-voltage sequencing, and measure the stretching response of DNA in MspA in response to voltage are summarized in Table S3.

Experiment Description	Enzyme control mechanism	DNA Sequence	Number of Pores	Number of Events
6-mer model building, SVM training	Hel308	ΦX174	19	155
6-mer model building	Hel308	λ phage	46	128
6-mer model building	Hel308	Fill-in template 1	3	18
6-mer model building	Hel308	Fill-in template 2	3	28
6-mer model building$_{\text{I}}$	Hel308	Fill-in template 3	7	38
6-mer model building	Hel308	Fill-in template 4	6	30
6-mer model building	Hel308	Fill-in template 5	4	25
6-mer model building	Hel308	Fill-in template 6	4	33
Variable-voltage sequencing	Hel308	pET28a	10	97
Constant-voltage sequencing	Hel308	pET28a	21	31
DNA stretching	Hel308	Stretching strand	2	18

Table S 3: Experimental statistics. The number of pores run and the total number of enzyme-controlled DNA translocation events collected are summarized for the experiments underpinning the development and demonstration of variable-voltage sequencing.