A Definitions of downsampled and windowed syncmer schemes

Downsampled syncmers: Given a uniformly random hash function \( h : \Sigma^k \rightarrow [0, H] \), for a given string \( S \), downsampling selects syncmers only from the set of \( |\Sigma|^k / \delta \) \( k \)-mers with the lowest hash values.

\[
DS_{k,s,o,x_1,...,x_n}(S, h, \delta) = \{ i \mid i \in S_{k,s,o}(x_1, ..., x_n) \wedge h(S[i, i + k - 1]) < H / \delta \}
\]

We call \( \delta \) the downsampling rate.

Windowed syncmers: Windowed syncmers fill in gaps using a minimizer scheme, thus providing a window guarantee. For clarity in the definition below let \( S(S) \) represent \( S_{k,s,o}(x_1, ..., x_n) \).

\[
f_{k,s,w,o,x_1,...,x_n}(S) = \{ i \mid i \in S(S) \bigcup i \in M_{k,w,o}(S[j, j + w - 1]) \forall j \text{ s.t. } S(S[j, j + w - 1]) = \emptyset \}
\]

Letting \( S \) represent all \( k \)-mers that can be syncmers in \( S_{k,s,o}(x_1, ..., x_n) \), an equivalent definition would be:

\[
M'_{k,w,o}(S) \text{ where } o' \text{ is defined such that } x \in S, x' \in \Sigma^k \setminus S \implies x < x'.
\]

B Analysis of parameterized syncmer schemes

In this section we provide our full theoretical analysis of PSS.

B.1 Recursive expressions for conservation of PSS

Consider a window of \( \alpha \) consecutive \( k \)-mers. We assume random sequence (i.e., made up of iid bases) throughout. Let \( s_\beta \) be the \( s \)-minimizer in the \( \alpha \)-window, at position \( \beta \). Then if \( t \) is a parameter of the syncmer scheme, \( s_\beta \) generates a syncmer if it is not in the first \( t - 1 \) or last \( k - t \) positions in the \( \alpha \)-window. If \( \beta \) is not in a position where it generates a syncmer, we recursively check to the left or right of \( \beta \) to see if a syncmer is generated by the \( s \)-minimizer of that region. See Fig ?? for an example.

For a 1-parameter scheme \( f \) with \( k \)-mer length \( k \), \( s \)-minimizer length \( s \), and parameter \( t \) let \( P(\alpha) \) be the probability of selecting at least one syncmer in a window of \( \alpha \) adjacent \( k \)-mers. Then, assuming a uniformly random hash over the \( s \)-mers, and conditioning on the position of the \( s \)-minimizer of the \( \alpha \)-window, \( \beta \):

\[
P(\alpha) = p_\beta \sum_\beta P(\alpha|\beta) \approx \frac{1}{k + \alpha - s} \left[ \sum_{\beta=1}^{t-1} P(\alpha - \beta) + \sum_{\beta=1}^{t+a-1} P(\alpha - \beta) + \sum_{\beta=1}^{t+a-s} P(\beta - k + s - 1) \right].
\]

The probability of any of the \( k + \alpha - s \) starting positions being the \( s \)-minimizer is denoted as \( p_\beta \) and assumed to be uniform. This assumption starts to break down when the \( s \)-minimizer is not unique, thus we note that the probability is approximate. If \( \beta \) is in the first \( t - 1 \) or last \( k - t \) starting positions (red regions in Fig ??), then a syncmer may be generated by the remaining \( \alpha - \beta \) positions to the right or \( \beta - k + s - 1 \) positions to the left, respectively. Note we define \( \sum_{x=i}^{j} = 0 \) when \( i > j \) and \( P(x) = 0 \) when \( x \leq 0 \).
When downsampling is used then the 1 in the second and fifth sums is replaced by \( \frac{1}{\delta} \).

When downsampling syncmers, there is a probability of \( \frac{1}{\delta} \) that an \( s \)-minimizer in the syncmer generating region (i.e. the green region in Fig A) really generates a syncmer. If it does not, then the left and right regions are considered recursively, yielding the following expression, where we simplify notation by letting \( P(\alpha - \beta) = P_R \) and \( P(\beta - k + s - 1) = P_L \):

\[
P(\alpha) \approx \frac{1}{k + \alpha - s} \left[ \sum_{\beta = 1}^{t-1} P_R + \sum_{\beta = t}^{t+\alpha-1} \left( \frac{1}{\delta} + (1 - \frac{1}{\delta}) (P_R + P_L - P_R \cdot P_L) \right) + \sum_{t+\alpha}^{k+\alpha-s} P_L \right].
\]

In the case of 2-parameter schemes, two syncmers are generated by \( s_\beta \) in regions that will overlap if the parameters \( t_1 \) and \( t_2 \) are within \( s \) of each other, and will be disjoint otherwise (see Fig A,B,C). Combining these two cases into a single recursive expression yields:

\[
P(\alpha) \approx \frac{1}{k + \alpha - s} \left[ \sum_{\beta = 1}^{t_1-1} P_R + \sum_{\beta = t_1}^{\min(t_2,t_1+\alpha)} 1 + \sum_{\beta = \min(t_2,t_1+\alpha)}^{t_1+\alpha-1} 1 + \sum_{\beta = \min(t_2,t_1+\alpha)}^{t_2-1} (P_R + P_L) + \sum_{\beta = \max(t_1+\alpha,t_2)}^{\min(t_2,t_1+\alpha)+1} \sum_{t+\alpha}^{k+\alpha-s} P_L \right].
\]

When downsampling is used then the 1 in the second and fifth sums is replaced by \( \frac{1}{\delta} + (1 - \frac{1}{\delta}) (P_R + P_L - P_R \cdot P_L) \) as in the one parameter case. The third sum expresses the overlapped region where either parameter creates a syncmer, when it exists. When both generated syncmers are downsampled then the left and right sides are recursively checked, thus the 1 is replaced by \( (1 - (1 - \frac{1}{3})^2 + (1 - \frac{1}{3})^2) (P_R + P_L - P_R \cdot P_L) \).

This expression can be greatly simplified by introducing the notation \( \text{count}(\beta) \) that represents the number of syncmers generated by the \( s \)-minimizer \( s_\beta \). For example, \( \text{count}(\beta) = 0 \) in the red region of Fig A and \( \text{count}(\beta) = 2 \) in the overlapped region when \( \beta = 6 \) or 7 in Fig A,C. The general expression for \( P(\alpha) \) for any PSS is:

\[
P(\alpha) \approx \frac{1}{k + \alpha - s} \left[ \sum_{\beta = 1}^{k+\alpha-s} \left( 1 - (1 - \frac{1}{\delta})^{\text{count}(\beta)} \right) + (1 - \frac{1}{\delta})^{\text{count}(\beta)} (P_R + P_L - P_R \cdot P_L) \right].
\]

Note that this definition relies on the definition \( P(x) = 0, x < 0 \) to include the correct terms for the correct values of \( \beta \).

The value of \( P(\alpha) \) can thus be computed efficiently for any PSS and used to calculate the conservation using the formula from [1].
B.2 Calculating the distance distribution

For a given scheme, the distribution of distances between adjacent syncmer positions is specified by \( Pr(d = x) \), the probability that the distance \( d \) is \( x \). To calculate this probability, we define the new quantities \( F(\alpha) \) and \( L(\alpha) \) denoting the probability that only the first or only the last \( k \)-mer in a window of \( \alpha \) \( k \)-mers is a syncmer, respectively. We refer to these \( k \)-mers as \( K_1 \) and \( K_\alpha \) respectively. Note that due to symmetry \( F(\alpha) = L(\alpha) \).

Note also that \( 1 - P(\alpha) \) gives the probability that no \( k \)-mer in an \( \alpha \)-window is a syncmer.

We compute \( F(\alpha) \) by conditioning on \( \beta \) as before. For simplicity we divide the sum over \( \beta \) into cases based on the syncmers that are generated by \( s_\beta \) rather than breaking up the sum across different values of \( \beta \). With some abuse of notation, we let \( K_1 \) represent the event that the \( s_\beta \) generates \( K_1 \) as a syncmer.

\[
F(\alpha) \approx \frac{1}{k + \alpha - s} \sum_{\beta = 1}^{k+\alpha-s} \begin{cases} 
\frac{1}{3} \cdot (1 - \frac{1}{3})^{count(\beta) - 1} \cdot (1 - P(\alpha - \beta)) & \text{if } \beta \text{ generates } K_1 \\
\frac{1}{3} \cdot (1 - \frac{1}{3})^{count(\beta) - 1} \cdot F(\beta - k + s - 1) \cdot (1 - P(\alpha - \beta)) & \text{otherwise} 
\end{cases}
\]

In the first case we have the probability that \( K_1 \) is not downsampld, any other syncmer generated by \( s_\beta \) is downsampld, and there are no other syncmers generated to the right of \( \beta \). In the second case we have the probability that any syncmers generated by \( s_\beta \) are downsampld, no syncmers are generated to the right of \( \beta \), and the recursive computation of the probability that the \( s \)-minimizer of the segment to the left of \( \beta \) generates a syncmer at \( K_1 \).

Similarly, define \( D(\alpha) \) to be the probability that in a window of \( \alpha \) \( k \)-mers only the first and last \( k \)-mers are syncmers. Then

\[
D(\alpha) \approx \frac{1}{k + \alpha - s} \sum_{\beta = 1}^{k+\alpha-s} \begin{cases} 
\frac{1}{3} \cdot (1 - \frac{1}{3})^{count(\beta) - 2} \cdot (1 - \frac{1}{3})^{count(\beta)} \cdot P_r(\alpha - 1) & \text{if } \beta \text{ generates } K_1, K_\alpha \\
\frac{1}{3} \cdot (1 - \frac{1}{3})^{count(\beta) - 1} \cdot F(\alpha - \beta) & \text{if } \beta \text{ generates } K_1, \neg K_\alpha \\
\frac{1}{3} \cdot (1 - \frac{1}{3})^{count(\beta) - 1} \cdot F(\beta - k + s - 1) \cdot (1 - P(\alpha - \beta)) & \text{if } \beta \text{ generates } \neg K_1, K_\alpha \\
\frac{1}{3} \cdot (1 - \frac{1}{3})^{count(\beta) - 1} \cdot F(\beta - k + s - 1) \cdot F(\alpha - \beta) & \text{otherwise} 
\end{cases}
\]

B.3 Calculating \( \ell_{2,\text{mut}} \)

To compute the desired metric \( \ell_{2,\text{mut}} \), we must calculate the metrics from the previous section but only with conserved syncmers. We add the subscript ‘mut’ to a value to indicate that only conserved syncmers are considered. The impact of mutations is similar to that of downsampling shown in the previous section, except that when a syncmer is lost due to mutation, the surrounding \( k \)-mers are also lost. In this case we consider no downsampling to make the expressions simpler.

Let \( \Omega_\beta \) be the set of syncmers generated by \( s_\beta \), and \( \omega_\beta_i \) be the members of this set. Note that \(|\Omega_\beta| = count(\beta)|\).

Then,

\[
P_{\text{mut}}(\alpha) = \frac{1}{k + \alpha - s} \sum_{\beta = 1}^{k+\alpha-s} \left( Pr(\exists \text{ conserved } \omega_\beta_i \in \Omega_\beta) + Pr(\exists \text{ conserved } \omega_\beta_i \in \Omega_\beta, \exists \text{ syncmer to the left or right}) \right)
\]

For convenience we call the first probability \( P_{\text{conserved}} \) and the second \( P_{\text{recurse}} \).

\( P_{\text{conserved}} \) is computed using the inclusion-exclusion principle:

\[
P_{\text{conserved}} = \sum_i Pr(\omega_i \text{ conserved}) - \sum_{i<j} Pr(\omega_i, \omega_j \text{ conserved}) + \sum_{i<j<k} Pr(\omega_i, \omega_j, \omega_k \text{ conserved}) - ... 
\]

Every term in this series is calculated as \((1 - \theta)^{countBases}\) where \( \theta \) is the mutation rate and \( countBases \) counts the number of bases covered by the conserved \( k \)-mers (i.e. if two conserved syncmers overlap, the shared bases are counted only once).

\( P_{\text{recurse}} \) is more complicated. We again sum over all values of \( \beta \). When \( \Omega_\beta \) is empty (e.g. \( \beta \) is in the red region), then the recursion is similar to the case without mutation. Otherwise, all of the syncmers are lost.
due to mutation, and we additionally sum over the possible positions of the first and last points of mutation in $\Omega_\beta$, named $f$ and $l$, respectively.

$$P_{\text{recurse}} \approx \frac{1}{k + \alpha - s} \times$$

$$\sum_{\beta=1}^{k+\alpha-s} \begin{cases} 
P_{\text{mut}}(\alpha - \beta) + P_{\text{mut}}(\beta - k + s - 1) - P_{\text{mut}}(\alpha - \beta) \cdot P_{\text{mut}}(\beta - k + s - 1) & \Omega_\beta = \emptyset \\
\sum_{f \leq l} P_{\text{recurse}}(f, l) \cdot (P_{\text{mut}}(left) + P_{\text{mut}}(right) - P_{\text{mut}}(left) \cdot P_{\text{mut}}(right)) & \text{otherwise}
\end{cases}$$

Here \( left = \max(0, \min(f - k, \beta - k + s - 1)) \) and \( right = \max(0, \min(\alpha - l, \alpha - \beta)) \). We expand the joint probability as

$$P_{\text{recurse}}(f, l) = \theta^y \cdot (1 - \theta)^x \cdot P_{\text{recurse}}(f, l)$$

where $y$ is 1 if $l = f$ and 2 otherwise, and $x$ is the number of unmutated bases that is fixed by the given values of $f$ and $l$. The conditional probability is written as

$$P_{\text{recurse}}(f, l) = 1 - P_{\exists} \text{ conserved}$$

and is computed using the inclusion-exclusion principle as above.

$F(\alpha)$ and $D(\alpha)$ are extended to the case of mutation similarly:

$$F_{\text{mut}}(\alpha) \approx \frac{1}{k + \alpha - s} \times$$

$$\sum_{\beta=1}^{k+\alpha-s} \begin{cases} 
\sum_{f \leq l} P_{\text{recurse}}(f, l) \cdot (1 - P_{\text{mut}}(right|b)) & K_1, \Omega_\beta \setminus K_1 \neq \emptyset \\
\sum_{f \leq l} P_{\text{recurse}}(f, l) \cdot (1 - P_{\text{mut}}(\alpha - \beta|b)) & K_1, \Omega_\beta \setminus K_1 = \emptyset \\
F_{\text{mut}}(\beta - k + s - 1) \cdot (1 - P_{\text{mut}}(\alpha - \beta)) & \text{otherwise}
\end{cases}$$

$$D_{\text{mut}}(\alpha) \approx \frac{1}{k + \alpha - s} \times$$

$$\sum_{\beta=1}^{k+\alpha-s} \begin{cases} 
\sum_{f \leq l} (1 - \theta)^k \cdot F_{\text{mut}}(\alpha - \beta|b) & K_1, -K_\alpha, \Omega_\beta \setminus K_1 \neq \emptyset \\
\sum_{f \leq l} (1 - \theta)^k \cdot F_{\text{mut}}(\beta - k + s - 1|b) & K_1, -K_\alpha, \Omega_\beta \setminus K_1 = \emptyset \\
\sum_{f \leq l} P_{\text{recurse}}(f, l) \cdot F_{\text{mut}}(left|b) & K_\alpha, -K_1, \Omega_\beta \setminus K_\alpha \neq \emptyset \\
P_{\text{recurse}}(\{K_1 \cup K_\alpha\}) \cdot F_{\text{mut}}(right|b) & K_1, K_\alpha, \Omega_\beta \setminus \{K_1 \cup K_\alpha\} \neq \emptyset \\
F_{\text{mut}}(\alpha - \beta) \cdot F_{\text{mut}}(\beta - k + s - 1) & K_1, K_\alpha, \Omega_\beta \setminus \{K_1 \cup K_\alpha\} = \emptyset \\
K_1, -K_\alpha, \Omega_\beta \setminus K_1 = \emptyset \\
-K_1, -K_\alpha, \Omega_\beta \neq \emptyset \\
-K_1, -K_\alpha, \Omega_\beta = \emptyset
\end{cases}$$

Here we again divide into cases depending on whether there are syncmers that can be lost. We have also used recursive expressions that are similar to the above except we are given that $b$ bases to the left or right of the defined region are conserved. These are calculated using similar techniques as above.

Finally, we can use these expressions to compute:

$$\ell_{\text{mut}} = \sum_{x=k+1}^{\infty} (x - k) \cdot D_{\text{mut}}(x + 1)$$

$$\ell_{2,\text{mut}} = \sqrt{\sum_{x=k+1}^{\infty} (x - k)^2 \cdot D_{\text{mut}}(x + 1)}$$
Table A Comparison of theoretical and simulated values of $\ell_{2,\text{mut}}$. Values of $\ell_{2,\text{mut}}$ for a selection of schemes with $k = 15$, $s = 5$, $\theta = 0.15$. Theoretical values were computed using 1000 terms. Simulated values were found on a simulated sequence of length 50,000,000.

Note that, unlike $P(\alpha)$, which can be computed efficiently, the computation of these metrics includes an infinite sum. The sum can be truncated at an appropriate distance $x$, however there are still many more terms than in the computation of the conservation. In practice, simulating a very long sequence, selecting syncmers, and simulating mutations to determine these metrics empirically is much less time consuming and yields results that are very close to the true values. We used this simulation method to compute $\ell_{2,\text{mut}}$ for $k = 11, 13, 15, 17$ and 19, mutation rates 0.05, 0.15 and 0.25, and all 2- and 3-parameter schemes Results are presented in Supplementary Data File 1, Table SD2 (note that for 1-parameter schemes the best $\ell_2$ and $\ell$ are the same, and thus already known from [1]). The computed theoretical expressions for some parameter combinations are available in Supplementary Data File 1, Table SD3. A comparison of the values computed theoretically and by simulation for the 2-parameter schemes $S(3, x)$, $x \in [4, 11]$ with $k = 15$, $s = 5$, 15% mutation rate can be seen in Table A.

C Windowed PSS implementation

Algorithm A describes the implementation of windowed PSS.

**Algorithm A** Windowed syncmer selection

**Input:** Sequence $S$, syncmer parameters $x_1, x_2, \ldots, x_n$, $k$-mer length $k$, $s$-mer length $s$, window $w$, downsampling rate $\delta$ (default: 1)

**Output:** $P$, a list of selected positions.

1. $P \leftarrow \emptyset$
2. for $j \in 1$ to $|S| - w + 1$ do
3.  \hspace{1em} hasSync $\leftarrow$ FALSE
4.  \hspace{1em} for $l \in 0$ to $w - k$ do
5.  \hspace{2em} kmer = Canonical$_{h_1}(S[j + l, j + l + k - 1])$
6.  \hspace{2em} $m = \min_{t \in [0, k-s]} h_2($ Canonical$_{h_1}(S[j + l, j + l + k - 1]))$
7.  \hspace{2em} if $m \in \{x_1, x_2, \ldots, x_n\}$ and $h_2(kmer) < 1/\delta$ then
8.  \hspace{3em} $P \leftarrow P \cup \{j + l\}$
9.  \hspace{3em} hasSync $\leftarrow$ TRUE
10. \hspace{1em} if hasSync is FALSE then
11. \hspace{2em} $m = \min_{t \in [0, w-k]} h_2($ Canonical$_{h_1}(S[j + l, j + l + k - 1]))$
12. \hspace{2em} $P \leftarrow P \cup \{j + m\}$
13. return $P$
D Syncmer based mapping implementations

Modifications to the mappers were minimal. Only the code that selects the k-mers to use as seeds to index the reference sequence and in the query reads was modified. Here we describe implementation details and optimizations in the code that differ from the high-level descriptions in Algorithm ?? and [A].

In minimap2 the most frequent minimizers (0.02% by default) are dropped to reduce spurious matches and lower the runtime and memory usage. We also drop the most frequent selected k-mers as the last stage of all minimap syncmer variants for consistency. In Winnowmap, the most frequent k-mers (also 0.02% by default) are re-weighted in the minimizer order so they are less likely to be selected as minimizers. In the syncmer-winnowmap variant, we do not consider k-mer weighting, and thus we simply drop these k-mers if they are selected. However, in the windowed syncmer-winnowmap variant we do re-weight the frequent k-mers before selecting minimizers in empty windows.

We use several different hashes in our syncmer variants of the mappers: $h_{can}$ to select canonical k-mers, $h_s$ to select s-minimizers, $h_{min}$ to select minimizers for windowed variants and $h_{down}$ for downsampling. We require that $h_{can}$ ≠ $h_{down}$ to maintain random downsampling. In minimap syncmer variants we use hash64 from minimap2 for $h_{min}$ and a variant of MurmurHash2 that ensures murmur2(0) ≠ 0 to ensure randomness for the other hashes. Thus $h_{min} = -\text{hash64/UINT64\_MAX}$, $h_s(x) = \text{murmur2}(x)$, $h_{down}(x) = \text{murmur2}(x)$, and $h_{can}(x) = \text{murmur2}(x << 1 + 5)$ to ensure that it has a different value than $h_{down}$. For winnowmap variants we use $h_{can}(x) = \text{lexicographic}(x)$ as this is what is used by the k-mer counter Meryl, $h_{min}(x) = -\left(\text{murmur2}(x)/\text{UINT64\_MAX}\right)^8$ in the case that the minimizer is one of the most frequent and $-\text{murmur2}(x)/\text{UINT64\_MAX}$ otherwise. The other hashes are as in the minimap variants.

In all windowed variants, downsampling occurs before filling in empty windows with minimizers.

ONT reads were mapped using the map-ont option in all mappers, while PacBio reads were mapped using the map-pb option (map-pb-clr in Winnowmap and variants). The latter uses homopolymer compression (HPC) and thus has a real compression (on the non-HPC sequence) that is above the theoretical one. In Winnowmap2 the map-ont option uses SV mode that attempts to resolve structural variation and may affect performance.

E Mapper command line arguments

The command line arguments of syncmer-minimap and syncmer-winnowmap extend the definitions of the original tools. minimap2 and Winnowmap2 were called as follows:

```
./minimap2 -x [map-pb|map-ont] -w <w> -k <k> -t <# threads> -a <ref> <reads>
```

```
./winnowmap -W <repeats file> -x [map-pb-clr|map-ont] [-H] -w <w> -k <k> -t <# threads> -a <ref> <reads>
```

The options map-pb, map-pb-clr and -H are used for PacBio reads and map-ont for ONT reads. For the syncmer versions, command line options are added for the PSS parameters (positions of the s-minimizer), s-minimizer length, and the downsampling rate:

```
./minimap2 -x [map-pb|map-ont] --downsample <d> --s-mer <s> --pos1 <x1> [--pos2 <x2> --pos3 <x3> [--pos4 <x4>]] [-w <w>] -k <k> -t <# threads> -a <ref> <reads>
```

```
./winnowmap -W <repeats file> -x [map-pb-clr|map-ont] [-H] --downsample <d> --s-mer <s> --pos1 <x1> [--pos2 <x2> --pos3 <x3> [--pos4 <x4>]] [-w <w>] -k <k> -t <# threads> -a <ref> <reads>
```

Note that in the syncmer mappers the option -w is used to define windowed syncmer schemes and is set to 0 for no windowing by default.
F Simulation details

PacBio reads were simulated using PBSim [2] with the CLR model and the following parameter settings:
depth 10, mean length 9000, length std 6000, minimum length 100, maximum length 40000, mean accuracy
0.87, accuracy std 0.02, minimum accuracy 0.85, maximum accuracy 1, and difference ratio 10:48:19. Error
rates and read lengths roughly matched to the statistics observed in a recent benchmark of long read correction

ONT reads were simulated using NanoSim [4] with default parameters and the human pre-trained model for
Guppy base calls.

To evaluate mapping correctness for the PacBio simulated data we used the mapeval utility of paftools
packaged with minimap2. In this tool, reads are considered correctly mapped if the overlap between the read
alignment and the true read location is $\geq 10\%$ of the combined length of the true read and aligned read
interval. This criterion was also used in [5].

G Bacterial species used

We chose a single representative assembly of each strain from [6] with the fewest, longest and most highly
covered contigs and concatenated all references into a single fasta file. Reads from the same samples were
downloaded. Assemblies and reads from following samples were used:

- bc1019, Bacillus cereus 971 (ATCC 14579)
- bc1059, Bacillus subtilis W23
- bc1101, Burkholderia cepacia (ATCC 25416)
- bc1102, Enterococcus faecalis OG1RF (ATCC 47077D-5)
- bc1111, Escherichia coli K12
- bc1087, Escherichia coli W (ATCC 9637)
- bc1018, Helicobacter pylori J99 (ATCC 700824)
- bc1077, Klebsiella pneumoniae (ATCC BAA-2146)
- bc1082, Listeria monocytogenes (ATCC 19117)
- bc1043, Methanocorpusculum labreanum Z (ATCC 43576)
- bc1047, Neisseria meningitidis FAM18 (ATCC 700532)
- bc1054, Rhodopseudomonas palustris
- bc1119, Staphylococcus aureus HPV (ATCC BAA-44)
- bc1079, Staphylococcus aureus subsp. aureus (ATCC 25923)
- bc1052, Treponema denticola A (ATCC 35405)

H Performance of minimizers and PSS on real genome sequences
without mutations and with low mutation rate

The metrics measured on real genomes without mutations are shown in Table B.
The metrics are shown for the low mutation rate of 4\% in Table C. The optimal PSS outperforms minimizers
even at this low mutation rate, but not at lower rates.

I Results when using forward-only (non-canonical) $k$-mers

We tested the performance metrics on real genomes when using only forward $k$-mers and not canonical ones.
Table D shows the metrics when only forward $k$-mers are used. The ranking between schemes is unchanged
and the values of the metrics remain similar.

To test the impact of using canonical $k$-mers on the distance distribution between selected positions we used
$S_{15,5}(3, 9)$ on the unmutated CHM13X. Fig B shows the distance distribution for syncmers selected only
### Table B Properties of minimizer and syncmer schemes on real sequences without mutations.
The best syncmer schemes with theoretical compression of 5.5 and 10 were chosen. The table shows the actual compression and other metrics on the real sequences, # positions is the number of positions that were selected by the scheme.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Scheme</th>
<th>Compression</th>
<th>$\ell$</th>
<th>$\ell_2$</th>
<th>$p_{90}$</th>
<th>$p_{100}$</th>
<th># positions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECK12</td>
<td>$\mathcal{M}_{15,10}$</td>
<td>5.503</td>
<td>3.23E-6</td>
<td>0.005</td>
<td>9</td>
<td>10</td>
<td>843,408</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(3,9)$</td>
<td>5.490</td>
<td>0.0191</td>
<td>0.377</td>
<td>10</td>
<td>53</td>
<td>845,419</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(1,11)$</td>
<td>5.509</td>
<td>0.0255</td>
<td>0.443</td>
<td>11</td>
<td>52</td>
<td>842,557</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(1,2)$</td>
<td>5.502</td>
<td>0.0566</td>
<td>0.715</td>
<td>13</td>
<td>66</td>
<td>843,634</td>
</tr>
<tr>
<td></td>
<td>$\mathcal{M}_{15,19}$</td>
<td>9.989</td>
<td>0.0527</td>
<td>0.398</td>
<td>18</td>
<td>19</td>
<td>464,660</td>
</tr>
<tr>
<td></td>
<td>$S_{15,6}(6)$</td>
<td>9.986</td>
<td>0.133</td>
<td>1.375</td>
<td>19</td>
<td>99</td>
<td>464,810</td>
</tr>
<tr>
<td>CHM13X</td>
<td>$\mathcal{M}_{15,10}$</td>
<td>5.489</td>
<td>5.83E-8</td>
<td>0.0005</td>
<td>9</td>
<td>10</td>
<td>28,099,399</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(3,9)$</td>
<td>5.523</td>
<td>0.0205</td>
<td>0.412</td>
<td>10</td>
<td>161</td>
<td>27,930,897</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(1,11)$</td>
<td>5.437</td>
<td>0.0259</td>
<td>0.449</td>
<td>11</td>
<td>260</td>
<td>28,371,548</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(1,2)$</td>
<td>5.509</td>
<td>0.0549</td>
<td>0.706</td>
<td>13</td>
<td>122</td>
<td>28,835,008</td>
</tr>
</tbody>
</table>

### Table C Properties of minimizer and syncmer schemes on real sequences with low mutation rate. Substitutions were introduced in the references at a rate of 4%. The values shown are for the conserved selected $k$-mers. # conserved is the number of $k$-mers selected by a scheme that were conserved under mutation.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Scheme</th>
<th>Description</th>
<th>Compression</th>
<th>$\ell$</th>
<th>$\ell_2$</th>
<th>$p_{90}$</th>
<th>$p_{100}$</th>
<th># conserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECK12</td>
<td>$\mathcal{M}_{15,10}$</td>
<td>minimap minimizer</td>
<td>75.86</td>
<td>0.86</td>
<td>13.61</td>
<td>208</td>
<td>1323</td>
<td>61,184</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(3,9)$</td>
<td>optimal PSS</td>
<td>9.795</td>
<td>0.308</td>
<td>3.012</td>
<td>25</td>
<td>205</td>
<td>1,443,355</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(1,11)$</td>
<td>closed syncmer</td>
<td>9.708</td>
<td>0.327</td>
<td>3.181</td>
<td>26</td>
<td>235</td>
<td>1,456,277</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(1,2)$</td>
<td>“bad PSS”</td>
<td>9.601</td>
<td>0.365</td>
<td>3.517</td>
<td>27</td>
<td>214</td>
<td>1,472,511</td>
</tr>
<tr>
<td>CHM13X</td>
<td>$\mathcal{M}_{15,10}$</td>
<td>minimap minimizer</td>
<td>54.27</td>
<td>0.81</td>
<td>11.70</td>
<td>152</td>
<td>1424</td>
<td>15,342,238</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(3,9)$</td>
<td>optimal PSS</td>
<td>9.499</td>
<td>0.292</td>
<td>2.874</td>
<td>24</td>
<td>220</td>
<td>16,238,925</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(1,11)$</td>
<td>closed syncmer</td>
<td>9.344</td>
<td>0.310</td>
<td>3.030</td>
<td>25</td>
<td>260</td>
<td>16,508,520</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(1,2)$</td>
<td>“bad PSS”</td>
<td>9.187</td>
<td>0.348</td>
<td>3.365</td>
<td>26</td>
<td>266</td>
<td>16,790,565</td>
</tr>
</tbody>
</table>

### Table D Performance metrics of minimizer and syncmer schemes using non-canonical $k$-mers on real sequences with simulated mutations. Substitutions were introduced in the references at a rate of 15%. The values shown are for the conserved selected $k$-mers. # conserved is the number of $k$-mers selected by a scheme that were conserved under mutation. Best performance is shown in bold.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Scheme</th>
<th>Description</th>
<th>Compression</th>
<th>$\ell$</th>
<th>$\ell_2$</th>
<th>$p_{90}$</th>
<th>$p_{100}$</th>
<th># conserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECK12</td>
<td>$\mathcal{M}_{15,10}$</td>
<td>minimap minimizer</td>
<td>54.27</td>
<td>0.81</td>
<td>11.70</td>
<td>152</td>
<td>1424</td>
<td>2,842,246</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(3,9)$</td>
<td>optimal PSS</td>
<td>45.41</td>
<td>0.77</td>
<td>10.41</td>
<td>126</td>
<td>1219</td>
<td>3,397,027</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(1,11)$</td>
<td>closed syncmer</td>
<td>44.83</td>
<td>0.80</td>
<td>11.15</td>
<td>131</td>
<td>1173</td>
<td>3,440,892</td>
</tr>
<tr>
<td></td>
<td>$S_{15,5}(1,2)$</td>
<td>“bad PSS”</td>
<td>43.04</td>
<td>0.71</td>
<td>12.98</td>
<td>129</td>
<td>1481</td>
<td>3,534,565</td>
</tr>
</tbody>
</table>
A.

**Fig B Distribution of the distances between selected positions in a syncmer scheme.** The distribution of distances between consecutive selected positions of syncmer scheme $S_{15,5}(3, 9)$ on the CHM13X reference is shown. A. The distribution of syncmers selected only in the forward orientation. B. Canonical syncmers. For visualization purposes the distribution is shown only for distances with frequency $> 10^{-5}$. The true maximum distance is 161 for canonical $k$-mers (see Supplementary Table B) and 76 for the forward $k$-mers, but the frequency of the longer distances is extremely low.

using forward strand $k$-mers. The minimum distance is 3 and there is a sharp peak at 6. In general, for a scheme with parameters $x_1 < ... < x_n$, the minimum distance is $\min(x_1, k - s - x_n + 2)$ and there are peaks at $x_{i+1} - x_i$, in agreement with the expression for $D(\alpha)$ above (??). In mapping, read orientations are unknown and canonical syncmers are used. Fig B3 shows the results using canonical $k$-mers. The distance distribution still retains the peak at 6 and a local maximum at 3, but now adjacent positions are selected, and it has a much longer tail of distances (compare Supplementary Table B). We conclude that while the theory is limited to single-stranded sequences it shows trends that hold for canonical $k$-mers.

To demonstrate that the improved mapping performance does not depend on the choice to use canonical $k$-mers, we disabled selection of seeds from the reverse strand in both indexing and read mapping. Results are shown in Figure C. Although many more reads are unmapped due to having selectable $k$-mers only in the reverse orientation, we see that the gap in mapping performance is similar to what was observed using canonical $k$-mers as seeds. In addition, the runtime of syncmer-minimap is much closer to that of minimap2, as expected since it no longer keeps track of the s-minimizers on two strands.

**J Supplemental performance results**

Fig D shows additional results for the number of unmapped reads at low, medium, and high compression rates.

Fig E compares the performance of the theoretically optimal 2-parameter PSS and closed syncmers in mapping real PacBio bacterial reads against the BAC reference.

**K Windowed syncmer scheme results**

Tables F and G present the properties of windowed syncmer schemes on real genome sequences with and without mutation, respectively.

Figures F and G present the number of unmapped reads and wrongly mapped reads for simulated datasets. These correspond to Figure ?? and include the results for windowed variants. Fig F presents the impact of percent sequence identity on the windowed variants as well, corresponding to Fig ??.

Results on the real human and bacterial reads are presented in Fig I and the runtimes and RAM usage for these runs are in Figures J and K. The runtime and memory usage on different tasks for the windowed
Fig C  % unmapped, memory and runtime vs. compression using non-canonical $k$-mers.

**Top:** Percentage of unmapped reads. **Middle:** Runtime in seconds to index the reference and map reads by each method. **Bottom:** Peak RAM usage in GB to index the reference and map reads. **Left:** PacBio bacterial reads. **Right:** ONT human cell-line reads.
Fig D Percentage of unmapped reads – additional data. The percentage of unmapped reads is plotted for one simulated and one real read dataset mapped to their corresponding references. Minimizer and PSS parameters are as described in the Results section (The fraction of unmapped reads). (A) PacBio reads simulated from the CHM13 ChrX sequence mapped against CHM13X. Window sizes of windowed syncmer-minimap were $w = 13, 80, 165$ for the low, medium, and high compression variants, respectively, and for windowed syncmer-winnowmap they were $w = 14, 75, 170$, respectively. (B) 1000 PacBio reads sampled from each of the 15 bacterial species in BAC mapped against their reference genomes. Window sizes of windowed syncmer-minimap were $w = 13, 75, 175$ and in windowed syncmer-winnowmap they were $w = 16, 75, 170$ for the low, medium, and high compression variants, respectively.

Fig E Comparison of 2-parameter PSS to closed syncmers. $S(3, 9)$ and closed syncmers were used in syncmer-minimap to map PacBio bacterial reads against BAC. Both schemes used $k = 15$, $s = 5$. 
<table>
<thead>
<tr>
<th>$w$</th>
<th>$\delta$</th>
<th>$c$</th>
<th>$\ell$</th>
<th>$\ell_2$</th>
<th>$p_{90}$</th>
<th>$p_{100}$</th>
<th># positions</th>
<th># syncmers</th>
<th># minimizers</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1</td>
<td>4.775</td>
<td>$2.59 \times 10^{-8}$</td>
<td>0.000255</td>
<td>9</td>
<td>10</td>
<td>32,309,357</td>
<td>27,969,919</td>
<td>4,339,438</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>5.322</td>
<td>$2.59 \times 10^{-8}$</td>
<td>0.000255</td>
<td>10</td>
<td>15</td>
<td>28,987,572</td>
<td>27,969,919</td>
<td>1,017,653</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>5.464</td>
<td>0.01047</td>
<td>0.1862</td>
<td>10</td>
<td>20</td>
<td>28,230,846</td>
<td>27,969,919</td>
<td>260,927</td>
</tr>
<tr>
<td>25</td>
<td>1</td>
<td>5.501</td>
<td>0.0167</td>
<td>0.3014</td>
<td>10</td>
<td>25</td>
<td>28,040,255</td>
<td>27,969,919</td>
<td>70,336</td>
</tr>
<tr>
<td>50</td>
<td>1</td>
<td>5.515</td>
<td>0.0197</td>
<td>0.3899</td>
<td>10</td>
<td>50</td>
<td>27,970,312</td>
<td>27,969,919</td>
<td>393</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>5.515</td>
<td>0.01973</td>
<td>0.3915</td>
<td>10</td>
<td>99</td>
<td>27,969,930</td>
<td>27,969,919</td>
<td>11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$w$</th>
<th>$\delta$</th>
<th>$c$</th>
<th>$\ell$</th>
<th>$\ell_2$</th>
<th>$p_{90}$</th>
<th>$p_{100}$</th>
<th># conserved</th>
<th>conserved syncmers</th>
<th>conserved minimizers</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2</td>
<td>5.463</td>
<td>$6.48 \times 10^{-8}$</td>
<td>0.000729</td>
<td>9</td>
<td>10</td>
<td>28,238,334</td>
<td>14,002,723</td>
<td>14,235,611</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>7.435</td>
<td>$6.48 \times 10^{-8}$</td>
<td>0.000729</td>
<td>14</td>
<td>15</td>
<td>20,748,902</td>
<td>14,002,723</td>
<td>6,746,179</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>8.812</td>
<td>0.0493</td>
<td>0.4201</td>
<td>17</td>
<td>20</td>
<td>17,505,813</td>
<td>14,002,723</td>
<td>3,503,090</td>
</tr>
<tr>
<td>25</td>
<td>2</td>
<td>9.706</td>
<td>0.10495</td>
<td>0.8324</td>
<td>20</td>
<td>25</td>
<td>15,892,867</td>
<td>14,002,723</td>
<td>1,890,144</td>
</tr>
<tr>
<td>50</td>
<td>2</td>
<td>10.93</td>
<td>0.1986</td>
<td>1.788</td>
<td>23</td>
<td>50</td>
<td>14,112,958</td>
<td>14,002,723</td>
<td>110,235</td>
</tr>
<tr>
<td>100</td>
<td>2</td>
<td>11.015</td>
<td>0.2067</td>
<td>1.9811</td>
<td>23</td>
<td>100</td>
<td>14,003,934</td>
<td>14,002,723</td>
<td>1211</td>
</tr>
</tbody>
</table>

Table E Properties of windowed syncmers on CHM13X. Properties of windowed and down-sampled variants of $S_{15,5}(3,9)$ are shown for a range of window lengths $w$ on the human chromosome X sequence of CHM13. $\delta$ is the downsampling rate and $c$ is the actual compression. The theoretical compression of the PSS (not downsampled) is 5.5. # positions is the number of positions that were selected by the scheme, # syncmers is the number of those that were selected by the PSS and # minimizers is the number of minimizers added to fill in gaps of length $\geq w$.

<table>
<thead>
<tr>
<th>$w$</th>
<th>$\delta$</th>
<th>$c$</th>
<th>$\ell$</th>
<th>$\ell_2$</th>
<th>$p_{90}$</th>
<th>$p_{100}$</th>
<th># conserved</th>
<th>conserved syncmers</th>
<th>conserved minimizers</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1</td>
<td>41.745</td>
<td>0.7826</td>
<td>10.713</td>
<td>122</td>
<td>1221</td>
<td>3,695,319</td>
<td>3,378,302</td>
<td>317,017</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>45.008</td>
<td>0.7928</td>
<td>11.041</td>
<td>131</td>
<td>1221</td>
<td>3,427,368</td>
<td>3,378,302</td>
<td>49066</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>45.558</td>
<td>0.7951</td>
<td>11.125</td>
<td>132</td>
<td>1221</td>
<td>3,386,019</td>
<td>3,378,302</td>
<td>7717</td>
</tr>
<tr>
<td>25</td>
<td>1</td>
<td>46.466</td>
<td>0.7955</td>
<td>11.126</td>
<td>132</td>
<td>1221</td>
<td>3,379,466</td>
<td>3,378,302</td>
<td>1164</td>
</tr>
<tr>
<td>50</td>
<td>1</td>
<td>46.662</td>
<td>0.7956</td>
<td>11.129</td>
<td>133</td>
<td>1274</td>
<td>3,378,304</td>
<td>3,378,302</td>
<td>2</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>46.662</td>
<td>0.7956</td>
<td>11.128</td>
<td>133</td>
<td>1274</td>
<td>3,378,302</td>
<td>3,378,302</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>52.34</td>
<td>0.8062</td>
<td>11.515</td>
<td>147</td>
<td>1221</td>
<td>2,947,249</td>
<td>1,692,160</td>
<td>1,255,089</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>71.027</td>
<td>0.8397</td>
<td>12.898</td>
<td>190</td>
<td>1691</td>
<td>2,171,847</td>
<td>1,692,160</td>
<td>479,687</td>
</tr>
<tr>
<td>20</td>
<td>2</td>
<td>81.734</td>
<td>0.8572</td>
<td>13.774</td>
<td>216</td>
<td>1703</td>
<td>1,887,340</td>
<td>1,692,160</td>
<td>195,180</td>
</tr>
<tr>
<td>25</td>
<td>2</td>
<td>86.999</td>
<td>0.8655</td>
<td>14.257</td>
<td>230</td>
<td>1736</td>
<td>1,773,121</td>
<td>1,692,160</td>
<td>80961</td>
</tr>
<tr>
<td>50</td>
<td>2</td>
<td>91.104</td>
<td>0.8722</td>
<td>14.709</td>
<td>242</td>
<td>1736</td>
<td>1,693,222</td>
<td>1,692,160</td>
<td>1062</td>
</tr>
<tr>
<td>100</td>
<td>2</td>
<td>91.161</td>
<td>0.8723</td>
<td>14.718</td>
<td>242</td>
<td>1736</td>
<td>1,692,160</td>
<td>1,692,160</td>
<td>0</td>
</tr>
</tbody>
</table>

Table F Properties of conserved windowed syncmers under mutation. Properties of windowed and down-sampled variants of $S_{15,5}(3,9)$ are shown for a range of window lengths $w$ on CHM13X after simulating substitutions at a rate of 15%. $w$ is the window size and $\delta$ is the downsampling rate. Properties of the conserved selected $k$-mers are reported.
Fig F Percentage of unmapped reads – simulated datasets. The percentage of unmapped reads is plotted for two simulated read datasets mapped to their reference sequences. Results are shown for low, medium, and high compression. (A) PacBio reads simulated from the CHM13 ChrX sequence mapped against ChrX sequences from GRCh38. Window sizes of windowed syncmer-minimap were $w = 13, 77, 175$ for the low, medium, and high compression variants, respectively. For windowed syncmer-winnowmap the window sizes were $w = 14, 75, 170$, respectively. (B) 1000 ONT reads simulated from CHM13 mapped against GRCh38. Window sizes of windowed syncmer-minimap were $w = 13, 75, 175$ for the low, medium, and high compression variants, respectively and $w = 13, 75, 170$ for the corresponding windowed syncmer-winnowmap variants.

Fig G Percentage of incorrectly mapped reads – simulated data. The percentage of incorrectly mapped reads is plotted for two simulated read datasets and their reference sequences, for mappers using low, medium, and high compression. (A) PacBio reads simulated from the CHM13 ChrX sequence mapped against CHM13X. Window sizes of windowed syncmer-minimap were $w = 13, 80, 165$ for the low, medium, and high compression variants, respectively. For windowed syncmer-winnowmap they were $w = 14, 75, 170$, respectively. (B) PacBio reads simulated from the 15 bacterial species in BAC mapped against the union of their references. Window sizes of windowed syncmer-minimap were $w = 13, 75, 175$ and in windowed syncmer-winnowmap they were $w = 16, 75, 170$ for the low, medium, and high compression variants, respectively.
Fig H Impact of percent sequence identity. We varied the mutation rate of 1000 PacBio simulated reads from CHM13X. The figures present the % unmapped and incorrectly mapped for each of the tools. (A) % unmapped reads. (B) % of the mapped reads that were incorrectly mapped.

(A) PacBio reads mapped to BAC
(B) ONT reads mapped to GRCh

Fig I Percentage of unmapped reads – real datasets. Results are shown across a broad range of compression rates. (A) Pooled PacBio bacterial reads. (B) ONT human cell-line reads.

variants is presented in Table G
Fig J Runtime vs. compression – real data. The figures show runtime in seconds to index the reference and map reads by each method. (A) PacBio bacterial reads. (B) ONT human cell-line reads.

Fig K Memory usage vs. compression – real data. Peak RAM usage in GB to index the reference and map reads for the different methods. (A) PacBio bacterial reads. (B) ONT human cell-line reads.

<table>
<thead>
<tr>
<th>Task</th>
<th>Method</th>
<th>Scheme</th>
<th>Index time</th>
<th>Index mem</th>
<th>Map time</th>
<th>Map mem</th>
</tr>
</thead>
<tbody>
<tr>
<td>pbsim bac vs BAC</td>
<td>Windowed syncmer minimap</td>
<td>$S_{15,5}(3,9)$</td>
<td>4.36</td>
<td>0.32</td>
<td>11.96</td>
<td>2.92</td>
</tr>
<tr>
<td>pbsim chm13x vs CHM13X</td>
<td>Windowed syncmer minimap</td>
<td>$S_{15,5}(3,9)$</td>
<td>11.46</td>
<td>1.01</td>
<td>104.31</td>
<td>5.45</td>
</tr>
<tr>
<td></td>
<td>Windowed syncmer winnowmap</td>
<td>$S_{15,4}(6)$</td>
<td>7.79</td>
<td>0.45</td>
<td>48.81</td>
<td>9.60</td>
</tr>
<tr>
<td>pbsim bac vs CHM13X</td>
<td>Windowed syncmer minimap</td>
<td>$S_{15,5}(3,9)$</td>
<td>As above</td>
<td>As above</td>
<td>26.10</td>
<td>9.18</td>
</tr>
</tbody>
</table>

Table G Runtime and memory. Time (in seconds) and RAM (in GB) needed to index the reference and map reads by each of the tools. The second and third dataset use the same reference and therefore have the same indexing results. For $S_{15,5}(3,9)$ window size $w = 13$ was used, with downsampling rate of 1.08 for BAC and downsampling rate of 1.05 for CHM13X. For $S_{15,4}(6)$ the standard PSS used downsampling rate of 4.13, and the windowed PSS had window size $w = 165$ and downsampling rate 4.37.
References


