

Note S1 Methods

Schema of the proposed method is illustrated in Fig. S1. There are five major stages in this method, including

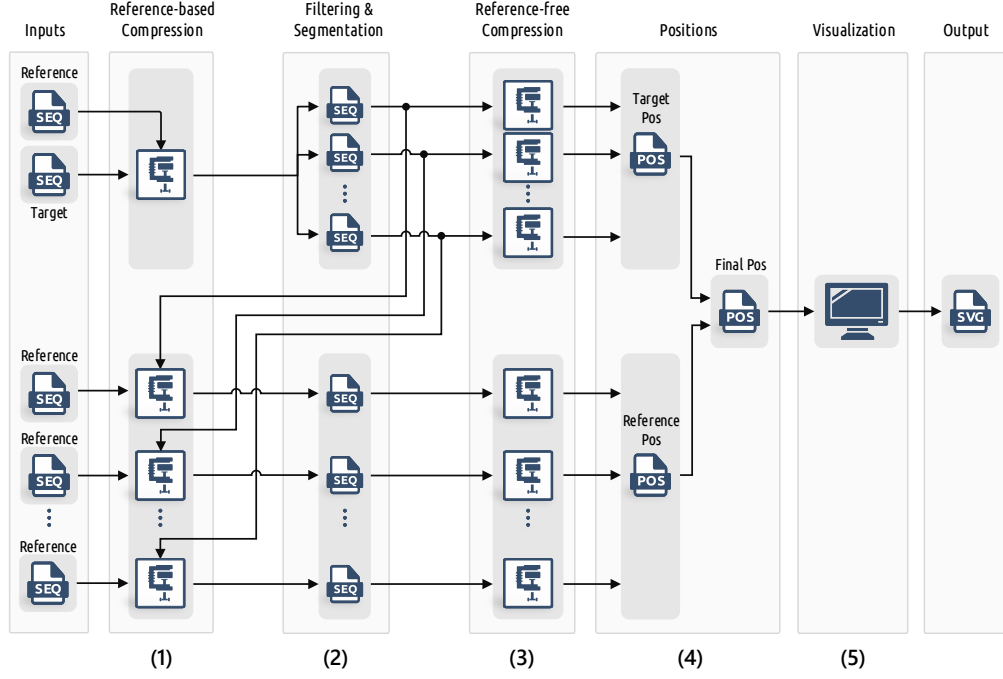


Fig. S1. The schema of Smash++.

S1.1 Building models of the data

S1.2 finding similar regions

In order to smooth the profile information, we use Hann window, which is a discrete window function given by

$$w[n] = 0.5 - 0.5 \cos\left(\frac{2\pi n}{N}\right) = \sin^2\left(\frac{\pi n}{N}\right), \quad (\text{Eq. S1})$$

in which, $0 \leq n \leq N$ and length of the window is $N + 1$ (Fig. S2).

S1.3 Computing complexities

S1.4 The software

Besides Hann window, that is used as default to filter the profile information obtained by the reference-based compression, we have implemented several other window functions (Fig. S3), including Blackman, Hamming, Nuttall, rectangular, sine, triangular and Welch windows. These functions

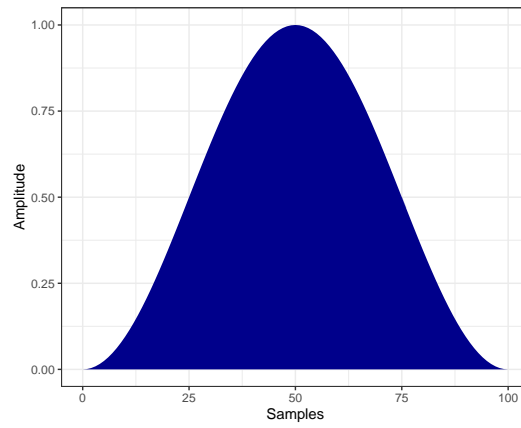


Fig. S2. Hann window for 101 samples.

are given by

$$\begin{aligned}
 w[n] &= 1, & (\text{rectangular}) \\
 w[n] &= 1 - \left| \frac{n-N/2}{L/2} \right|, \quad L = N, & (\text{triangular/Bartlett}) \\
 w[n] &= 1 - \left(\frac{n-N/2}{N/2} \right)^2, & (\text{Welch}) \\
 w[n] &= \sin \left(\frac{\pi n}{N} \right), & (\text{sine}) \\
 w[n] &= 0.54348 - 0.45652 \cos \left(\frac{2\pi n}{N} \right), & (\text{Hamming}) \\
 w[n] &= 0.42659 - 0.49656 \cos \left(\frac{2\pi n}{N} \right) + 0.07685 \cos \left(\frac{4\pi n}{N} \right), & (\text{Blackman}) \\
 w[n] &= 0.35577 - 0.48740 \cos \left(\frac{2\pi n}{N} \right) + 0.14423 \cos \left(\frac{4\pi n}{N} \right) - 0.01260 \cos \left(\frac{6\pi n}{N} \right). & (\text{Nuttall})
 \end{aligned}$$

(Eq. S2)

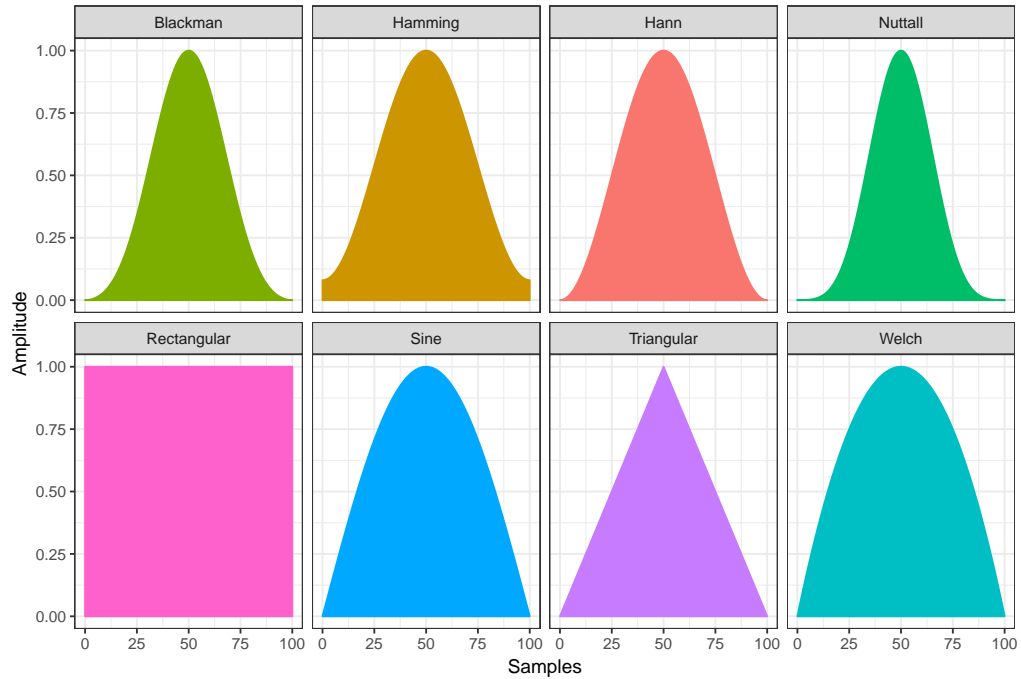


Fig. S3. Window functions.

Note S2 Experiment setup

S2.1 Datasets

Table S1. Datasets used in the experiments.

Category	Reference	Length (base)	Target	Length (base)	Description
Synthetic	RefS	1,000	TarS	1,000	
Synthetic	RefM	100,000	TarM	100,000	
Synthetic	RefL	5,000,000	TarL	5,000,000	
Synthetic	RefXL	100,000,000	TarXL	100,000,000	

Note S3 Results

Smash++ and several other methods have been carried out on a collection of synthetic and real sequences. The machine used for the tests had an 8-core 3.40 GHz Intel[®] Core[™] i7-6700 CPU with 32 GB RAM.

Smash++ is implemented in C++ language and is available at [1]. This tool is able to find rearrangements in sequences, Fasta and Fastq files; although, it is highly recommended to use sequences as input. To work with Smash++, it should be first installed. In the following sections, we describe installing and running Smash++.

In order to install Smash++, we run the following commands:

Running

provides the following guide:

[illegible]

```

42                                     0: regular (not inverted)
43                                     1: inverted, solely
44                                     2: both regular and inverted
45             (FLOAT) a: estimator
46             (FLOAT) g: forgetting factor: [0.0, 1.0)
47             (INT) t: threshold (no. substitutions)

```

The arguments “-r” and “-t” are used to specify the reference and the target files. It is highly recommended to choose short names for these files.

Here, on reference-based compression, we have replaced ‘N’ bases in the references with ‘A’s and ‘N’ bases in the targets with ‘T’s. Also, on reference-free compression, we have replaced ‘N’s in the references and the targets with ‘A’s. If a user tends to replace ‘N’ bases in a sequence with a normal distribution of ‘A’, ‘C’, ‘G’ and ‘T’s, he/she can employ GOOSE toolkit [2].

S4.3 Example

This section guides, step-by-step, employing Smash++ to find rearrangements.

Install Smash++ and provide the required files

First, we install Smash++:

```

1 git clone https://github.com/smortezah/smashpp.git
2 cd smashpp
3 cmake .
4 make

```

Then, we copy Smash++’s binary file into `example/` directory and go to that directory:

```

1 cp smashpp example/
2 cd example/

```

References

- [1] M. Hosseini, D. Pratas, and A. J. Pinho. Smash++. [Online]. Available: <https://github.com/smortezah/smashpp>
- [2] D. Pratas. Goose. [Online]. Available: <https://github.com/pratas/goose>