Supplementary material

Claude Opus 5.5 for bioinformatics: SwiftSNP, a byte-identical snp-dists that is 336-fold faster than the fastest equivalent tool

Joe Christensen (ORCID 0009-0007-0908-1299). Translational Neuropsychiatry Unit (TNU), Department of Clinical Medicine, Aarhus University, Denmark.

S1 Datasets

The synthetic sets are generated by gen_data.py from fixed seeds; the real alignments are described in main-text Section 4.3 and in the data availability statement.

Table S1. The real alignments span 100 to 8,596 genomes and 29,903 to 4,047,835 columns, of which 0.04% to 100% are informative. N, number of sequences; L, alignment columns; informative columns, columns with at least two different letters of A/C/G/T after upper-casing (snp-dists' default semantics), counted independently of SwiftSNP; non-ACGT cells, share of cells that snp-dists ignores by default.

dataset	set	organism / source	N	L	informative columns	non-ACGT cells
vcholerae_n100	real	*V. cholerae* O1, Snippy core genome (Zenodo 16909942)	100	4,047,835	1,720	3.7%
vcholerae_n100_gz	real	same alignment, gzip -6 -n	100	4,047,835	1,720	3.7%
mtb_n1788_snps	real	*M. tuberculosis*, Snippy SNP alignment (Zenodo 7772652)	1,788	88,562	88,562	0.3%
sarscov2_europe_n2012	real	SARS-CoV-2, Nextstrain open build europe	2,012	29,903	6,201	3.4%
sarscov2_global_n4637	real	SARS-CoV-2, Nextstrain open build global	4,637	29,903	8,889	3.5%
sarscov2_n8596	real	SARS-CoV-2, union of 7 Nextstrain open regional builds	8,596	29,903	13,360	3.2%
outbreak_full	dev	synthetic outbreak, full length, wrapped at 60	200	1,000,000	1,627	2.5%
core_snps	dev	synthetic SNP-only core	1,500	8,000	7,875	7.7%
large_n_snps	dev	synthetic SNP-only, large N	4,000	1,500	1,438	32.4%
diverse_full	dev	synthetic diverse, IUPAC codes, wrapped at 80	60	3,000,000	103,028	4.0%
gz_wrapped	dev	synthetic, gzip -1, wrapped at 60	300	300,000	3,830	2.6%
h_mid_full	holdout	mid-size, wrapped	700	200,000	2,599	4.2%
h_snps_2500	holdout	SNP-only with IUPAC codes	2,500	3,000	2,861	21.5%
h_heavy_n	holdout	gzip, N-rich, lowercase	100	2,000,000	24,981	22.2%
h_identical	holdout	near-identical sequences	500	100,000	47	0.0%

S2 Real data with other options

Table S2. Real-data runs with option combinations on the AVX-512 VM: snp-dists 1.2.0 `-q -j 1 <options>` once, SwiftSNP best of 3 at `-j 1` and `-j 2`. All 28 SwiftSNP outputs were byte-identical to snp-dists, and every snp-dists output matched its sha256 recorded on a second machine. `-a` is served by a generic path that was not optimised.

dataset	options	snp-dists `-j 1` (s)	SwiftSNP `-j 1` (s)	speed-up (fold)	SwiftSNP `-j 2` (s)	speed-up (fold)	identical (`-j 1`/`-j 2`)
vcholerae_n100	`-a`	28.9	0.935	31	0.543	53	yes/yes
vcholerae_n100	`-k`	29.7	0.058	510	0.039	753	yes/yes
vcholerae_n100	`-x 20`	8.5	0.081	105	0.040	210	yes/yes
vcholerae_n100	`-L -b`	15.5	0.072	216	0.038	407	yes/yes
vcholerae_n100	`-c -b`	30.2	0.075	404	0.041	734	yes/yes
vcholerae_n100	`-m`	30.3	0.058	527	0.027	1,107	yes/yes
vcholerae_n100	`-a -k -c`	29.6	0.959	31	0.594	50	yes/yes
sarscov2_europe_n2012	`-a`	93.9	1.618	58	0.901	104	yes/yes
sarscov2_europe_n2012	`-k`	92.4	0.037	2,473	0.036	2,555	yes/yes
sarscov2_europe_n2012	`-x 20`	37.8	0.037	1,035	0.024	1,566	yes/yes
sarscov2_europe_n2012	`-L -b`	46.3	0.043	1,091	0.032	1,456	yes/yes
sarscov2_europe_n2012	`-c -b`	92.5	0.038	2,418	0.023	4,009	yes/yes
sarscov2_europe_n2012	`-m`	94.2	0.140	673	0.085	1,114	yes/yes
sarscov2_europe_n2012	`-a -k -c`	97.6	1.669	59	0.854	114	yes/yes

The comparison tools in main-text Figure 3 were timed by a script whose process timer adds up to about 50 ms to each run. This changes the baseline times (seconds to minutes) by less than 1% but inflates SwiftSNP's sub-second times, so the main comparison uses SwiftSNP's times from the real-data runs; with SwiftSNP's times from the comparison script the geometric-mean margin is 263-fold instead of 336-fold.

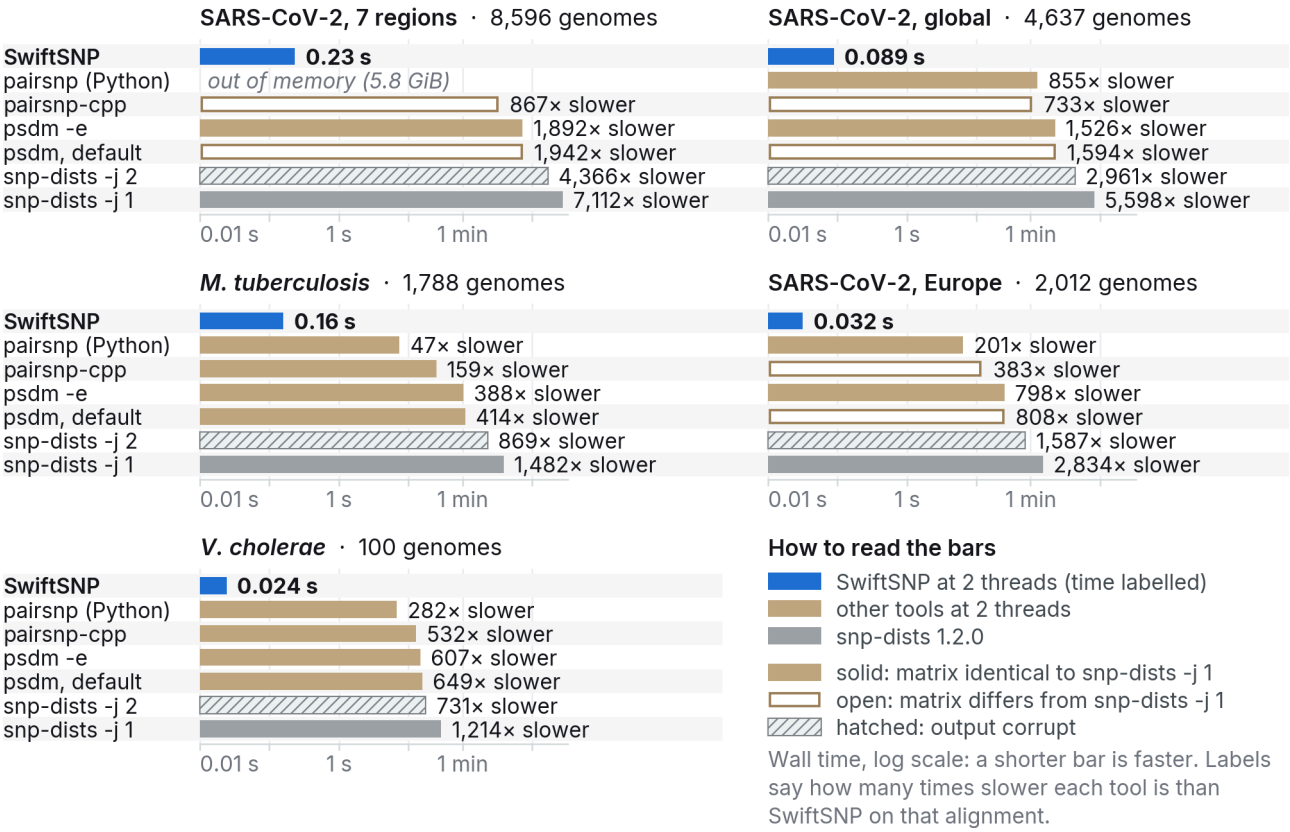

Figure S1. SwiftSNP is the fastest tool on every alignment. Wall time of every tool on each real alignment, log scale from 0.01 s; labels say how many times slower than SwiftSNP at two threads each tool was. Solid bars, matrix identical to `snp-dists -j 1`; open bars, matrix differs; hatched bars, output corrupt. Two threads on the AVX-512 VM except `snp-dists -j 1`; baselines best of three runs on *V. cholerae* and SARS-CoV-2 Europe and one run on the three largest alignments.

S3 Bugs in snp-dists 1.2.0: reproductions and patches

Multithreaded output is corrupt (main.c lines 181–186 and 203–212). On 120 seeded sequences × 2,000 columns, `snp-dists -q -j 1` gives md5 c5f6aa84058d, and five runs of `-j 2` gave five different md5 values, with 91 to 114 of 120 rows malformed and fused labels such as `s0s60`. All 9 `-j 2` matrices in the comparison runs were corrupt. With the patch below, 110 of 110 runs (11 option sets × `-j` 1, 2, 3, 4, 7 × 2 repeats) matched the unpatched `-j 1` output.

```
--- a/main.c
+++ b/main.c
@@ -178,12 +178,15 @@
     if (moltenheader) {
       printf("sequence_1%csequence_2%cdistance\n", sep, sep);
     }
-#pragma omp parallel for
     for (int j = 0; j < N; j++) {
       int start = lower ? j : 0;
+      // compute row j in parallel, then print it in order
+#pragma omp parallel for
       for (int i=start; i < N; i++) {
-        size_t d = distance(seq[j], seq[i], L, maxdiff);
-        printf("%s%c%s%c%zu\n", name[j], sep, name[i], sep, d);
+        d[i] = distance(seq[j], seq[i], L, maxdiff);
+      }
+      for (int i=start; i < N; i++) {
+        printf("%s%c%s%c%zu\n", name[j], sep, name[i], sep, d[i]);
       }
     }
   }
@@ -200,14 +203,16 @@

     // Output the distance matrix to stdout
     // (does full matrix, wasted computation i know)
-#pragma omp parallel for
     for (int j = 0; j < N; j++) {
-      printf("%s", name[j]);
       int end = lower ? j+1 : N;
+      // compute row j in parallel, then print it in order
+#pragma omp parallel for
+      for (int i=0; i < end; i++) {
+        d[i] = distance(seq[j], seq[i], L, maxdiff);
+      }
+      printf("%s", name[j]);
       for (int i=0; i < end; i++) {
-        // d[i] = distance(seq[j], seq[i], L, maxdiff);
-        size_t d = distance(seq[j], seq[i], L, maxdiff);
-        printf("%c%zu", sep, d);
+        printf("%c%zu", sep, d[i]);
       }
       printf("\n");
     }
```

A malformed FASTQ record silently ends the input. On a three-record FASTQ file whose second quality string is one character short, snp-dists prints a 1 × 1 matrix of the first record and exits 0, because its read loop `while ((l = kseq_read(kseq)) >= 0)` treats kseq's error codes −2 and −3 like the end of the file. With the patch below it reports the record and exits 1. SwiftSNP reproduces the 1.2.0 behaviour, as byte identity requires.

```
--- a/main.c
+++ b/main.c
@@ -158,6 +158,11 @@
     // keep track of how many we have
     N++;
   }
+  // kseq_read returns -2 (malformed FASTQ quality) or -3 (stream error): do not drop records
       silently
+  if (l < -1) {
+    fprintf(stderr, "ERROR: could not read sequence #%zu (malformed FASTQ record or read
       error)\n", N + 1);
+    exit(EXIT_FAILURE);
+  }
   kseq_destroy(kseq);
   gzclose(fp);
```

The help gives the `-x` default as 99999; the code uses 9,999,999. Two sequences of 200,000 differing bases, without `-x`, print a distance of 200000. The patch changes only the help text and fixes two typos in the `-L` line.

```
--- a/main.c
+++ b/main.c
@@ -47,11 +47,11 @@
       "  -a       Count all differences not just [AGTC]\n"
       "  -k       Keep case, don't uppercase all letters\n"
       "  -m       Output MOLTEN instead of TSV\n"
-      "  -L       Ootput lower-trangle only (unique pairs)\n"
+      "  -L       Output lower-triangle only (including the diagonal)\n"
       "  -t       Add header to MOLTEN output\n"
       "  -c       Use comma instead of tab in output\n"
       "  -b       Blank top left corner cell\n"
-      "  -x INT   Stop counting distance beyond this [99999]\n"
+      "  -x INT   Stop counting distance beyond this [9999999]\n"
       "URL\n  %s\n"
   };
   fprintf(out, str, EXENAME, cpus, GITHUB_URL);
```

pairsnp 0.0.7 from PyPI fails under numpy 2.5.3, because binary-mode `np.fromstring` is no longer supported. It was run with numpy 1.26.