NucDiff: in-depth characterization and annotation of differences between two sets of DNA sequences:

**Supplementary materials**

Ksenia Khelik, Karin Lagesen, Geir Kjetil Sandve, Torbjørn Rognes, Alexander Johan Nederbragt
Supplementary results

Effect of different MUMmer parameters: result comparison approach

In general, the simulated difference is assumed to be correctly detected if its location intersects with the location of a NucDiff-detected difference of the same type. However, some exceptions had to be made in order to allow a fair comparison in cases where there are identical bases nearby just by incident. First, in the cases with all types of deletions and simple relocations and translocations, the detected difference may be located not more than 3 bases before or after the simulated difference. Second, some differences are allowed to have several corresponding types, i.e. simulated simple relocations and translocations may be detected by NucDiff as simple relocations and translocations or relocations and translocations with overlap. In spite of the chosen NUCmer and delta-filter parameter values, we defined that all repeat related differences are detected as non-repeat related if they are shorter than 30 bases. If some fragment was relocated to another place in the query sequence, we defined that it is detected as a simple insertion if it is shorter than 30 bases. These limits are tool independent and were introduced to avoid detection of random duplications and fragment relocations.

Supplementary figures and tables

Figure S1 Reference fragments placement order depending on query fragment orientations during detection of local differences. a) case shows the placement order of A* and B* when A and B have the same orientation as A* and B*, b) case shows their placement order when A and B have the opposite orientation. The placement relation between A and B, A* and B* may be differ from what is shown here.
**Figure S2** Circular genome alignment alternatives

**Figure S3** Number of differences in each category obtained by NucDiff with the default parameter setting for all assemblers. The noRepeats group consists of simple insertion, simple deletion, simple substitution, unaligned end and unaligned beginning differences. The Transloc/Reloc group consists of all types of translocations and relocations. Inv/Reshuf consists of inversions and reshufflings. The unalignedSeq group consists of unaligned sequence differences. The default parameter setting description can be found in Table S4.
Figure S4 Comparison of multiple assemblies against one reference using NucDiff. The seven assemblies downloaded for the V. cholerae genome are created by the ABySS, CABOG, MaSuRCA, SGA, SOAPdenovo, SPAdes and Velvet assemblers based on a HiSeq read dataset. In a)-c) the top panel shows the focus area on the reference chromosome. The next seven panels corresponds to the reference mapped blocks in a), to the structural and long differences in b) and to the short and medium local differences in c).
Figure S5 Examples of detection of long deletions located in all assemblies at the same place in the reference sequence. The deletion differences were found by NucDiff with the default parameter settings. The first entry corresponds to the *V. cholerae* annotation. In a) the deletion location coordinates are 262291 - 263545 bases in NC_002505.1. In b) the deletion location coordinates are 310944 - 312310 bases in NC_002506.1.
Table S1: Alignment fragmentation cases caused by simple differences. K’, M’, R’, T’ are the reverse compliments of K, M, R, T, respectively. Len means length.

| Fragmentation case | Sequences | Differences | Symmetric fragmentation case |
|--------------------|-----------|-------------|------------------------------|
| 1.                 | R: MK     | no differences between fragments | K’ B’ A’ |
|                    | Q: MK     |                          | A B |
| 2.                 | R: MK     | 1) simple insertion or 2) inserted gap | K’ B’ A’ |
|                    | Q: MLK    |                          | A B |
| 3.                 | R: MDK    | simple deletion | K’ B’ A’ |
|                    | Q: MK     |                          | A B |
| 4.1                | R: MDK    | 1) simple substitution and simple insertion or 2) gap and inserted gap | K’ B’ A’ |
|                    | Q: MLK    |                          | A B |
|                    | Len(D)≤Len(L) |                        | A B |
4.2

R: MDK
Q: MLK

Len(D) > Len(L)

1) simple substitution and simple deletion
or
2) gap and simple deletion

5.

R: MRRK
Q: MRK

collapsed tandem repeat

6.

R: MRK
Q: MRRK
	
tandem duplication

7.

R: MRDRK
Q: MRK
		simple deletion and collapsed repeat
In Table S1 case 1 may appear only after merging the fragments in the nested fragment cases. It will never be met in the NUCmer output. In all cases with overlaps between query or reference fragments (cases 5, 6, 7, and 8), the lengths of corresponding differences ($L_{\text{diff}}$) are calculated based on the length of the overlap and information about the differences inside fragment by the following formulas:

a) If there is an overlap between reference fragments (cases 6 and 8):

$$L_{\text{diff}} = L_{\text{ref}} + L_{\text{ins}} - L_{\text{del}},$$

where

$L_{\text{ref}}$ - the length of the reference overlapping region

$L_{\text{ins}}$ - the length of an insertion that have a corresponding reference coordinate within the overlapping region
\( L_{\text{del}} \) - the length of a deletion that have a corresponding reference coordinate within the overlapping region

b) If there is an overlap between query fragments (cases 5 and 7):

\[
L_{\text{diff}} = L_q - L_{\text{ins}} + L_{\text{del}}, \text{ where}
\]

\( L_q \) - the length of the query overlapping region

\( L_{\text{ins}} \) - the length of an insertion that locates within the overlapping region

\( L_{\text{del}} \) - the length of a deletion that locates within the overlapping region

In cases where there are overlaps between reference and query fragments (cases 9.1 and 9.2), the type and length of difference are determined based on the lengths of both overlaps and information about the differences inside fragments. First, NucDiff calculates the difference length \( L_{\text{diff}} \) by the formula:

\[
L_{\text{diff}} = L_q - L_{\text{ref}} + L_{\text{ins}} + L_{\text{del}}, \text{ where}
\]

\( L_{\text{diff}} \) - the length of overlap between reference fragments

\( L_{\text{ref}} \) - the length of overlap between query fragments

\( L_{\text{ins}} \) - the length of all simple insertions and inserted gaps within the query overlapping region

\( L_{\text{del}} \) - the length of all deletions within the query overlapping region

Then it determines the placement case and corresponding difference type:

If \( L_{\text{diff}} > 0 \), then there is a case 9.2

If \( L_{\text{diff}} < 0 \), then there is a case 9.1

If \( L_{\text{diff}} = 0 \), then there is no differences between fragments
Table S2 Genome modifications implemented during the simulation process. R and Q denote reference and query genomes respectively. Letters A, B, C,K, x, y denote reference and query sequence regions. Diff means difference. A', B' and F' are the reverse complements of A, B and F, respectively.

| Deletions | Insertions |
|-----------|------------|
| 1. R: ABC | 1. R: AC  |
| Q: AC    | Q: ABC    |
| Diff: deletion | Diff: deletion |
| 2. R: ABCxBy | 2. R: ACxBy |
| Q: AC    | Q: ABCxBy |
| Diff: deletion | Diff: duplication |
| 3. R: ABCxBy | 3. R: ABC |
| Q: ACxBy | Q: ABKBC |
| Diff: collapsed repeat | Diff: insertion + duplication |
| 4. R: ABKBC | 4. R: ABCxKy |
| Q: ABC    | Q: ABKBCxKy |
| Diff: deletion + collapsed repeat | Diff: duplication + duplication |
| 5. R: ABBC | 5. R: ABC |
| Q: ABC    | Q: ABKBC |
| Diff: tandem collapsed repeat | Diff: tandem duplication |
| 6. R: ABKKBKBC | 6. R: ABKBC |
| Q: ABKBC | Q: ABKKBKBC |
| Diff: tandem collapsed repeat | Diff: tandem duplication |

| Substitutions | Gaps |
|---------------|------|
| 1. R: ABC | 1. R: ABC |
| Q: AKC | Q: ANC |
| Diff: substitutions (with and without insertions and deletions) | Diff: gap |
| 2. R: ABCxKy | 2. R: ABCxKy |
| Q: AKCxKy | Q: ABC |
| Diff: substitution | Diff: relocation |

| Relocations | Translocations |
|-------------|----------------|
| 1. R: ACxBy | 1. R1: ABC   |
| Q: ABC    | Q: ABKBC   |
| Diff: relocation | Diff: translocation + duplication |
| 2. R: ABCxKy | 2. R1: A  |
| Q: ABKBC | Q: AC  |
| Diff: relocation + duplication | Diff: translocation |
| 3. R: ABCxKy | 3a. R1: A  |
| Q: AKC | Q: ABC (B consists of ATGC’s) |
| Diff: relocation | Diff: translocation with insertion |
| 4. R: AxC | 3b. R1: A  |
| Q: AC | Q: ABC (B consists of N’s) |
| Diff: relocation | Diff: translocation with inserted gap |
| 5a. R: AxC | 3c. R1: A  |
| Q: ABC (B consists of ATGC’s) | Q: ABC (B consists of ATGNC’s) |
| Diff: relocation with insertion | Diff: translocation with insertion and inserted gap |
### Inversions

| R: | Q: | Diff: |
|---|---|---|
| ABC | AB'C | inversion |
| AKBLC | ATB'MC | inversions with and without insertions and deletions |

### Reshufflings

| R: | Q: | Diff: |
|---|---|---|
| ABLCKMD | CTADWBK | reshufflings |

### Additional tests

| R1: | R2: | Q: | Diff: |
|-----|-----|---|---|
| ABCDFxKy | L | CA/DBF/KL | translocation, relocation, reshuffling and inversion events |

### Unaligned sequences

| R: | Q: | Diff: |
|---|---|---|
| Correct small query sequences |
| Completely wrong query sequences |

---

In the simulated modifications the following lengths of regions were used:

1. Len(A/C) = 500 bases in all described cases, except reshuffling case
2. Len(x/y) = 10500 bases in all described cases
3. Distance between each manipulation case = 10500
4. Len(B) = \{5,20,50,65,85,88,100,150,200,250,300,350,400\} bases in deletion (1,2,3,5), insertion (1,2,5), relocation (1,5,6), translocation (2,3) and inversion (1) cases
5. Len(B/K) = \{5,30,80,150,250,350,400\} bases in deletion (4,6), insertion (3,4,6), all substitution and relocation (2,3) cases
6. Len(B/N) = \{5,30,80,150,250,350,400\} bases in gap related cases
7. Len(B) = 250 and Len(K/L/T/M) = \{0,5,250,400\} bases in inversion (2) case
8. Len(A/B/C/D/K) = 250 and Len(L/M/T/W) = \{0,5,250,400\} bases in reshuffling case
9. Len(D/B/F/K/L) = 500 bases in additional test (1) case
10. Len(B1/B2/B3) = 150 and Len(D) = 45 bases in additional test (2) case. B3 contains one nucleotide difference with each of B1 and B2. The differences are located in the reference sequence at the same positions where B1 and B2 have two differences of one nucleotide lengths.
11. Len(correct/completely wrong seq) = \{5,20,50,65,85,88,100,150,200,250,300,350,400\} bases in all unaligned sequence cases.
Table S3 List of *E. coli* genomes used in the Comparison of genomes from different strains of the same species section.

| Strain          | Accession number |
|-----------------|------------------|
| AG100           | NZ_LN832404.1    |
| BW25113         |                  |
| BW2952          | NC_012759.1      |
| DH10B           | NC_010473.1      |
| EcHMS174Chr     | NZ_LM993812.1    |
| EcRV308Chr      |                  |
| ER3413          | NZ_CP009789.1    |
| ER3435          | NZ_CP010445.1    |
| ER3445          | NZ_CP010439.1    |
| ER3446          | NZ_CP010443.1    |

| Strain          | Accession number |
|-----------------|------------------|
| ER3454          | NZ_CP010438.1    |
| ER3466          | NZ_CP010442.1    |
| ER3475          | NZ_CP010444.1    |
| ER3476          | NZ_CP010440.1    |
| GM4792 Lac+     | NZ_CP011342.2    |
| MDS42           | NC_020518.1      |
| MG1655          | U00096.3         |
| MG1655_TMP32XR1 | NZ_CP011343.2    |
| MG1655_TMP32XR2 |                  |
| W3110           | NC_007779.1      |

Table S4 Parameter values used for each parameter settings. The default set shows NucDiff default values for NUCmer and delta-filter options. The Quast-like set describes parameter values as used in QUAST. Values in bold are different from the default ones.

| Configuration sets | Parameter values |
|--------------------|------------------|
| default            | --nucmer_opt "-l 20 -c 65 -b 200" --filter_opt "-i 0" |
| c30                | --nucmer_opt "-l 20 -c 30 -b 200" --filter_opt "-i 0" |
| c120               | --nucmer_opt "-l 20 -c 120 -b 200" --filter_opt "-i 0" |
| l10                | --nucmer_opt "-l 10 -c 65 -b 200" --filter_opt "-i 0" |
| l65                | --nucmer_opt "-l 65 -c 65 -b 200" --filter_opt "-i 0" |
| b80                | --nucmer_opt "-l 20 -c 65 -b 80" --filter_opt "-i 0" |
| b350               | --nucmer_opt "-l 20 -c 65 -b 350" --filter_opt "-i 0" |
| quast-like         | --nucmer_opt "-l 65 -c 65 -b 200" --filter_opt "-i 95" |

- **l** - Minimum length of a maximal exact match
- **c** - Minimum cluster length
- **b** - Distance NUCmer will attempt to extend poor scoring regions before giving up
- **i** - Minimum alignment identity

All other NUCmer parameters, except --maxmatch, have the NUCmer default values and remained fixed in our tests. The --maxmatch parameter, which tells NUCmer to use all
anchor matches regardless of their uniqueness, is not used by default in NUCmer, but is required for NucDiff and thus is used in all tests.

As for the delta-filter filtering parameters, -q parameter (query alignment using length*identity weighted LIS [longest increasing subset]) is required for NucDiff to get the output results needed for the analysis and is present in all tests performed.

In the QUAST-like tests, we ran a test with the same parameter values used by QUAST, except for the -q parameter. It is not used by QUAST, but is required for NucDiff.

| Simulated difference types                                      | Qu difference types                                      |
|----------------------------------------------------------------|----------------------------------------------------------|
| insertion, deletion, substitution, duplication, tandem          | local misassembly, indel                                  |
| duplication, collapsed repeat, collapsed tandem repeat          |                                                          |
| inversion                                                      | inversion                                                |
| gap, inserted gap                                              | fake: scaffold gap size wrong estimation (denoted as gap in Table 1 and Table 3) |
| relocation, relocation with overlap, relocation with insertion, relocation with insertion and inserted gap | relocation                                               |
| relocation with inserted gap                                   | relocation, fake: scaffold gap size wrong estimation      |
| unaligned sequence                                             | unaligned                                                |
| all translocation types                                        | translocation                                            |
| reshuffling                                                    | local misassembly                                        |
| Category    | Expected/ NucDiff                                                                 | dnadiff       | QUAST                                      |
|-------------|----------------------------------------------------------------------------------|---------------|--------------------------------------------|
| nonTandem   | insertion, deletion, substitution (from query_struct.gff), wrong beginning, wrong end, inserted gap, duplication, collapsed repeat | Insertions, TotalIndels | Local misassemblies, indels               |
| Tandem      | tandem duplication, collapsed tandem repeat                                         | TandemIns     |                                            |
| Substitutions | substitution (from query_snps.gff), gap                                           | TotalSNPs     | mismatches                                 |
| Relocations | All types of relocation, reshuffling                                              | Relocations   | Relocations, scaffold gap size misassemblies |
| Translocations | All types of translocations                                                      | Translocations | translocations                            |
| Inversions  | inversion                                                                        | Inversions    | inversions                                 |
| UnalignedSeq   | Unaligned sequence                                                               | UnalignedSeqs | fully unaligned contigs                    |