ProDerAl – Use as directed

1| Quick Start

Windows Users: Download the binaries from

https://github.com/Benjamin-Crysup/proderal/releases/download/v1.1/proderal.zip

If you are dead set on building from source, you can run the relevant batch file, but you will need MinGW64 installed (and the binaries on the path environment variable).

*nix Users (and MacOS): Download the source from

https://github.com/Benjamin-Crysup/proderal.git

To build, run the relevant shell script (Buildit_***processor***_nix.sh) for your system. The binary will be in the builds folder (for most of you, builds/bin_x64_linux). You will need the zlib-devel package installed.

Once you have the software built, you will need a genetic reference, a description of problematic regions, a cost specification in those regions, and a SAM file you want to re-evaluate (if you have a BAM, use samtools view: obviously, you will need samtools installed to do this).

A minimal example is provided (with both source and binary distributions) and can serve as a template. A GUI is also provided (requires Python). For the most common use of this program, open up the command line / terminal and run (as one line):

```
samtools view /path/to/start.bam | proderal --ref
     /path/to/reference.fa --prob /path/to/prob.bed --cost
     /path/to/cost.pdc | samtools view -b > /path/to/result.bam
```

2| Command Line Arguments

```
proderal
   --ref ref.fa
   [--prob prob.bed] | [--promiscuous]
   --cost cost.pdc
   [--overrun num]
   [--maxexp num]
   [--atlocal] | [--atsemi] | [--atglobal]
   [--rank maxr]
   [--count maxc]
   [--reclaimsoft numper]
   [--qualm change.qca]
   [--hfuzz maxf]
   [--thread num]
   [--order]
   [--out dump.sam]
   FILE*
```

Proderal takes a set of reference sequences (--ref ref.fa), position dependent cost information (--cost cost.pdc) and, optionally, a specification of the regions of interest (--prob prob.bed). Then, for each sam file specified (stdin if none), every entry is examined. All supplemental alignments are discarded.

If the entry does not overlap a region of interest in the reference (and promiscuous mode has not been specified), it is passed through as is. Otherwise, the region mapped to in the reference is extended to the ends of any partially overlapped interesting regions (if any), and then further extended by the overrun (which defaults to 20). The total amount of expansion can optionally be limited with --maxexp. If reclaimsoft is set to anything other than zero, soft clipped bases in the read will further extend the relevant edge by numper for each base. A new alignment is then performed against the read sequence (if reclaimsoft is zero, excluding the soft clipped bases) and the expanded reference region. Only the top (maxr+1) alignment scores are considered, and the top maxc alignments are reported (this defaults to 0 for the maximum rank and 1 for the maximum count). Additionally, for highly degenerate alignments, the initial search for the optimal scores can be terminated after paths with the same score have been encountered maxf number of times (--hfuzz).

Realignment can be done using either local alignment or semiglobal alignment. Use local alignment to find related sequences on a background of unrelated sequences, and use semiglobal alignment for matching partial reads of a sequence (this is the default). The global option is deprecated since version 0.0 (it's only there for completeness).

It's also possible to specify how to change alignment parameters given the quality scores of the read bases (--qualm change.qca).

By default, ProDerAl outputs the results to stdout: a file can be specified using --out. Additionally, ProDerAl can be made to run with multiple threads (--thread). Again, by default, ProDerAl makes no attempt to preserve the order of the sam entries: if this is needed, --order can be used.

3| Cost Function

Position dependent cost functions for proderal are specified in chunks; each reference gets its own chunk. Each chunk starts with the reference name and ends with a pipe character (|). After the reference name is a default alignment parameter set (detailed later), followed by the number of specific parameter sets, followed by the specific parameter sets. All entries are separated by any amount of whitespace (ASCII 9, 10, 13, 32).

A default set of alignment parameters starts with the cost of opening a gap (should be negative), the cost of extending a gap, and the cost of closing a gap. After that is the number of relevant types of characters that can show up in the file (if you want to align amino acid sequences, set this to 21, for nucleic acids it should be 4… ish). The ASCII code points of those characters follows: for the most common cases,

ACGT -> 4 65 67 71 84

ACDEFGHIKLMNPQRSTUVWY -> 21 65 67 68 69 70 71 72 73 75 76 77 78 80 81 82 83 84 85 86 87 89

Following the list of bases is a 2d array of costs for character matching and mismatching: for nucleic acids, this will have 16 entries, and for amino acids, there will be 441. As for the order, an example is in order: for the nucleic acids, the first entry is for both the reference and the read being an A, the second entry is for the reference being an A and the read being a C, etc… (row major order).

The specific parameter sets have the same information as the default, but are preceded by the base indices that are covered. The first two are the start (zero based) and end (exclusive) in the reference, and the last two are the start and end in the read. Any position that is negative does not act as a constraint: -1 15 5 -1 would be any base in the reference up to (but not including) base 15, and any base in the read at or after base 5.

As an example:

```
chr1
-3 -1 -3
  4 65 67 71 84
  1 -4 -4 -4 -4 1 -4 -4 -4 -4 1 -4 -4 -4 -4 1
1
10 21 -1 -1
```

```
  -1 0 -1
  4 65 67 71 84
  1 -4 -4 -4 -4 1 -4 -4 -4 -4 1 -4 -4 -4 -4 1
|
```

This will use the default bwa mem parameters for chromosome 1, except between bases [10,20], where the gap costs are lower.

4| Quality Changes

Specifying changes to alignment parameters in response to read quality is done through commands on individual lines in a text file. The commands are:

Command	Description
r name r chr1	Set the reference that the following alterations will apply to.
q from to q 35 40	Set the quality characters in the read the following alterations will apply to. For instance, q 35 40 will apply to the ascii characters #$%&'(
~ ref read [ops]* ~ 65 84 * 5 - 1	Alter the match/mismatch penalties. When the reference and read have given bases (A and T in this case), apply the following operations to the current parameter.
Go [ops]* Go / 2	Alter the gap open penalty.
Gx [ops]* Gx + 3	Alter the gap extend penalty.
Gc [ops]* Gc = 0	Alter the gap close penalty.
Q	Marks the end of the current quality score.
R	Marks the end of a reference.

The operations are performed left to right. The set of possible operations is:

* 5	Multiply the current value by an integer (see the documentation of the c function atoi for the definition of an integer).
+ 5	Add an integer to the current value.
- 5	Subtract an integer from the current value.
/ 5	Divide the current value by an integer.
= 5	Set the current value to an integer.

5| Other Files

The reference file is a fasta file: each entry represents a single chromosome with the name given on the name line. On the name line, everything after a whitespace does not matter, and newlines can show up in the sequence itself. As a simple example

```
>chr1
ACGCTGACTACCAGGAA
AATGC
>chr2 extra stuft that doesn't matter
AGTTTACAAACAGT
```

The problematic region file is a bed file. A bed file is a tab separated value file. Proderal only uses the first three entries: the chromosome name, the start position (zero based, inclusive) and the end position (zero based, exclusive). As an example, the following

```
chr1 3    8
chr1 10   13
chr2 5    9
```

specifies the following bases

```
chr1
ACG<u>**CTGAC**</u>TA<u>**CCA**</u>GGAAAATGC
chr2
AGTTT<u>**ACAA**</u>ACAGT
```

The starting SAM file is a collection of alignments that need to be re-evaluated. The SAM format, while essentially a tab separated value file, is too complex to describe here: see https://samtools.github.io/hts-specs/SAMv1.pdf for a complete description.

6| Citing and License

This software is distributed under the GNU LGPL: redistribute the source and/or binary to your heart’s content, and you may alter and/or use this software (including for paid purposes) so long as the license on this component remains unadulterated and so long as proper attribution is given.

Speaking of proper attribution, should you use this software in your own papers, please cite

TBD