Primer 3 release 0.3

Contents

1 Terms of Usage

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Redistributions of source code must also reproduce this information in the source code itself.

2. If the program is modified, redistributions must include a notice (in the same places as above) indicating that the redistributed program is not identical to the version distributed by Whitehead Institute.

3. All advertising materials mentioning features or use of this software must display the following acknowledgment: This product includes software developed by the Whitehead Institute for Biomedical Research.

4. The name of the Whitehead Institute may not be used to endorse or promote products derived from this software without specific prior written permission.

We also request that use of this software be cited in publications as

> Steve Rozen, Helen J. Skaletsky (1996)
> Primer 3. Code available at
> `http://www-genome.wi.mit.edu/genome_software/other/Primer3.html`

This software is provided by the Whitehead Institute "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the Whitehead Institute be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

2 Introduction

Primer 3 is a complete rewrite of the original PRIMER program (Primer 0.5), written by Steve Lincoln, Mark Daly, and Eric Lander. See section 10, p. 14, for a discussion of how Primer 3 differs from its predecessors, Primer 0.5 and Primer v2.

Primer 3 picks primers for PCR reactions, considering as criteria:

- oligonucleotide melting temperature, size, GC content, and primer-dimer possibilities,
- PCR product size,
- positional constraints within the source sequence, and
- miscellaneous other constraints.

All of these criteria are user-specifiable as constraints, and some are specifiable as terms in an objective function that characterizes an optimal primer pair.

The Whitehead/MIT Center for Genome Research offers web-based front end to Primer3 at

`http://www.genome.wi.mit.edu/cgi-bin/primer/primer3.cgi.`

3 Installation Instructions

1. Go to the src directory of the distribution.
2. If you do not use gcc, modify the makefile to use your (ANSI) C compiler and appropriate compile and link flags.
3. `$ make primer3`

   Warnings about `I`, `J`, and `score` being possibly used unset in `dpal.c` are normal and harmless.
4. `$ cd ../test`

   If you do not have perl 5 in `/usr/local/bin/perl5` modify the first line of `primer_test.pl` to point to your path for perl 5.
5. `$ primer_test.pl`

   You should not see 'FAILED' during the tests.

4 System Requirements

Primer 3 has been successfully installed and tested on the following systems

- Sparc running SunOS 4.1 (gcc 2.7.0)
- Alpha running DEC Unix 3.2 (gcc 2.7.0 and DEC cc)
- Pentium running Linux 1.2 (Red Hat) (gcc 2.7.0)
- Pentium running Linux 2.0 (gcc 2.7.2)

Primer 3 may well compile and run on other POSIX architectures with ANSI C compilers. Please contact the authors with portability suggestions.

5 Input and Output Conventions

By default, Primer 3 accepts input and produces output in Boulder-io format, a text-based input/output format used as a program-to-program data interchange format in many information systems at the Whitehead/MIT Center for Genome Research. See `http://www.genome.wi.mit.edu` and follow the links to "Genome Center Software" for more information. When run with the `-format_output` command-line flag, Primer 3 prints a more user-oriented report for each sequence. Additional command-line flags include `-2x_compat` (which causes Primer 3 to print its output using Primer v2 compatible tag names), and `-strict_tags` (both discussed below). Primer 3 exits with 0 status if it operates correctly. See section 11, p. 15, below for additional information.

The syntax of the version of Boulder-io recognized by Primer 3 is as follows:

- Input consists of a sequence of `RECORD`s.
- A `RECORD` consists of a sequence of (`TAG,VALUE`) pairs, each terminated by a newline character (`\n`). A `RECORD` is terminated by '`=`' appearing by itself on a line.
- A (`TAG,VALUE`) pair has the following requirements:
  - the `TAG` must be immediately (without spaces) followed by '`=`'.
  - the pair must be terminated by a newline character.

An example of a legal (TAG,VALUE) pair is

```
PRIMER_SEQUENCE_ID=my_marker
```

and an example of a BOULDER-IO record is

```
PRIMER_SEQUENCE_ID=test1
SEQUENCE=GACTGATCGATGCTAGCTACGATCGATCGATGCATGCTAGCTAGCTAGCTGCTAGC
=
```

Many records can be sent, one after another. Below is an example of three different records which might be passed through a boulder-io stream:

```
PRIMER_SEQUENCE_ID=test1
SEQUENCE=GACTGATCGATGCTAGCTACGATCGATCGATGCATGCTAGCTAGCTAGCTGCTAGC
=
PRIMER_SEQUENCE_ID=test2
SEQUENCE=CATCATCATCATCGATGCTAGCATCNNACGTACGANCANATGCATCGATCGT
=
PRIMER_SEQUENCE_ID=test3
SEQUENCE=NACGTAGCTAGCATGCACNACTCGACNACGATGCACNACAGCTGCATCGATGC
=
```

Primer 3 reads boulder-io on `stdin` and echos its input and returns results in boulder-io format on `stdout`. Primer 3 indicates many user-correctable errors by a value in the `PRIMER_ERROR` tag (see below) and indicates other errors, including system configuration errors, resource errors (such as out-of-memory errors), and detected programming errors by a message on `stderr` and a non-zero exit status.

Below is the list of input tags that Primer 3 recognizes. Primer 3 echos and ignores any tags it does not recognize, unless the `-strict_tags` flag is set on the command line, in which case Primer 3 prints an error in the `PRIMER_ERROR` output tag (see below), and prints additional information on `stdout`; this option can be useful for debugging systems that incorporate primer.

Except for tags with the type "interval list" each tag is allowed only **once** in any given input record. This restriction is not systematically checked in this beta release: use care.

There are 2 major classes of input tags. **"Sequence"** input tags describe a particular input sequence to Primer 3, and are reset after every boulder record. **"Global"** input tags describe the general parameters that Primer 3 should use in its searches, and the values of these tags persist between input boulder records until or unless they are explicitly reset. Errors in "Sequence" input tags invalidate the current record, but Primer 3 will continue to process additional records. Errors in "Global" input tags are fatal because they invalidate the basic conditions under which primers are being picked.

5.1 "Sequence" Input Tags

`PRIMER_SEQUENCE_ID` (string, optional)

`MARKER_NAME` is a deprecated synonym maintained for v2 compatibility. An identifier that is reproduced in the output to enable users to identify the source of the chosen primers. This tag must be present if `PRIMER_FILE_FLAG` is non-zero.

`SEQUENCE` (nucleotide sequence, **required**)

The sequence from which to choose primers. The sequence must be presented $5' \longrightarrow 3'$ (see the discussion of the `PRIMER_SELF_END` argument, section 5.2, p. 8). The bases may be upper or lower case. No newlines should be inserted into the sequence, because the Boulder-IO parser will assume that a line ends at a newline.

INCLUDED_REGION (interval, optional)

A sub-region of the given sequence in which to pick primers. For example, often the first dozen or so bases of a sequence are vector, and should be excluded from consideration. The value for this parameter has the form

```
<start>,<length>
```

where <start> is the 0-based index of the first base to consider, and <length> is the number of subsequent bases in the primer-picking region.

TARGET (interval list, default empty)

If one or more Targets is specified then a legal primer pair must flank at least one of them. A Target might be a simple sequence repeat site (for example a CA repeat) or a single-base-pair polymorphism. The value should be a space-separated list of

```
<start>,<length>
```

pairs where <start> is the (0-based) index of the first base of a Target, and <length> is its length.

For backward compatibility Primer 3 accepts (but ignores) a trailing

```
,<description>
```

for each element of this argument.

EXCLUDED_REGION (interval list, default empty)

Primer oligos may not overlap any region specified in this tag. The associated value must be a space-separated list of

```
<start>,<length>
```

pairs where <start> is the (0-based) index of the first base of the excluded region, and <length> is its length. This tag is useful for tasks such as excluding regions of low sequence quality or for excluding regions containing repetitive elements such as ALUs or LINEs.

5.2 "Global" Input Tags

PRIMER_EXPLAIN_FLAG (boolean, default 0)

If this flag is non-0, produce

PRIMER_LEFT_EXPLAIN,

PRIMER_RIGHT_EXPLAIN, and

PRIMER_INTERNAL_OLIGO_EXPLAIN

output tags, which are intended to provide information on the number of oligos and primer pairs that Primer 3 examined, and statistics on the number discarded for various reasons. If -format_output is set similar information is produced in the user-oriented output.

PRIMER_PRODUCT_SIZE_RANGE (size range list, default 100-300)

The associated values specify the lengths of the product that the user wants the primers to create, and is a space separated list of elements of the form

`<x>-<y>`

where an `<x>-<y>` pair is a legal range of lengths for the product. For example, if one wants PCR products to be between 100 to 150 bases (inclusive) then one would set this parameter to `100-150`. If one desires PCR products in either the range from 100 to 150 bases or in the range from 200 to 250 bases then one would set this parameter to `100-150 200-250`.

Primer 3 favors ranges to the left side of the parameter string. Primer 3 will return legal primers pairs in the first range regardless the value of the objective function for these pairs. Only if there are an insufficient number of primers in the first range will Primer 3 return primers in a subsequent range.

`PRIMER_PICK_INTERNAL_OLIGO` (boolean, default 0)

If the associated value is non-0, then Primer 3 will attempt to pick an internal oligo. Briefly, an "internal oligo" is intended to be used as a hybridization probe to detect the PCR product after amplification.

`PRIMER_GC_CLAMP` (int, default 0)

Require the specified number of consecutive `G`s and `C`s at the $3'$ end of both the left and right primer. (This parameter has no effect on the internal oligo if one is requested.)

`PRIMER_OPT_SIZE` (int, default 20)

Optimum length (in bases) of a primer oligo. Primer 3 will attempt to pick primers close to this length.

`PRIMER_DEFAULT_SIZE` (int, default 20)

A deprecated synonym for `PRIMER_OPT_SIZE`, maintained for v2 compatibility.

`PRIMER_MIN_SIZE` (int, default 18)

Minimum acceptable length of a primer.

`PRIMER_MAX_SIZE` (int, default 27)

Maximum acceptable length (in bases) of a primer. Currently this parameter cannot be larger than 35. This limit is governed by maximum oligo size for which Primer 3's melting-temperature is valid.

`PRIMER_OPT_TM` (float, default 60.0 °C)

Optimum melting temperature (°C) for a primer oligo. Primer 3 will try to pick primers with melting temperatures are close to this temperature. The oligo melting temperature formula in Primer 3 is that given in Rychlik, Spencer and Rhoads, Nucleic Acids Research, vol 18, num 12, pp 6409-6412 and Breslauer, Frank, Bloeker and Marky, Proc. Natl. Acad. Sci. USA, vol 83, pp 3746-3750. Please refer to the former paper for background discussion.

`PRIMER_MIN_TM` (float, default 57.0°C)

Minimum acceptable melting temperature (°C) for a primer oligo.

`PRIMER_MAX_TM` (float, default 63.0 °C)

Maximum acceptable melting temperature(°C) for a primer oligo.

`PRIMER_MAX_DIFF_TM` (float, default 100.0 °C)

Maximum acceptable (unsigned) difference between the melting temperatures of the left and right primers.

`PRIMER_MIN_GC` (float, default 20.0%)

Minimum allowable percentage of `G`s and `C`s in any primer.

`PRIMER_MAX_GC` (float, default 80.0%)

Maximum allowable percentage of `G`s and `C`s in any primer generated by Primer.

`PRIMER_SALT_CONC` (float, default 50.0 mM)

The concentration of salt (usually KCl) in the PCR. Primer 3 uses this argument to calculate oligo melting temperatures.

`PRIMER_DNA_CONC` (float, default 50.0 nM)

The concentration of annealing oligos in the PCR. Primer 3 uses this argument to calculate oligo melting temperatures. The default works well with the standard protocol used at the Whitehead/MIT Center for Genome Research – 0.5 µl of 20 µmol concentration for each primer oligo in a 20 µl reaction with 10 ng template, 0.025 U/µl Taq polymerase in 0.1 mM each dNTP, 1.5 mM $MgCl_2$, 50 mM KCl, 10 mM Tris·HCL (pH 9.3) using 35 cycles with an annealing temperature of 56 °C. This parameter corresponds to c in Rychlik, Spencer and Rhoads' equation (ii) (Nucleic Acids Research, vol 18, num 12) where a suitable value (for a lower concentration of template) is "empirically determined". The value of this parameter is less than the actual concentration of oligos in the reaction because it is the concentration of annealing oligos, which in turn depends on the amount of template (including PCR product) in a given cycle. This concentration increases a great deal during a PCR; fortunately PCR seems quite robust for a variety of oligo melting temperatures.

See advice for picking primers, section 9, p. 14.

`PRIMER_NUM_NS_ACCEPTED` (int, default 0)

Maximum number of unknown bases (`N`) allowable in any primer.

`PRIMER_SELF_ANY` (decimal,9999.99, default 8.00)

The maximum allowable local alignment score when testing a single primer for (local) self-complementarity and the maximum allowable local alignment score when testing for complementarity between left and right primers. Local self-complementarity is taken to predict the tendency of primers to anneal to each other without necessarily causing self-priming in the PCR. The scoring system gives 1.00 for complementary bases, −0.25 for a match of any base (or `N`) with an `N`, −1.00 for a mismatch, and −2.00 for a gap. Only single-base-pair gaps are allowed. For example, the alignment

```
5' ATCGNA 3'
   || | |
3' TA-CGT 5'
```

is allowed (and yields a score of 1.75), but the alignment

```
5' ATCCGNA 3'
   ||  | |
3' TA--CGT 5'
```

is not considered. Scores are non-negative, and a score of 0.00 indicates that there is no reasonable local alignment between two oligos.

PRIMER_SELF_END (decimal 9999.99, default 3.00)

The maximum allowable 3′-anchored global alignment score when testing a single primer for self-complementarity, and the maximum allowable 3′-anchored global alignment score when testing for complementarity between left and right primers. The 3′-anchored global alignment score is taken to predict the likelihood of PCR-priming primer-dimers, for example

```
5' ATGCCCTAGCTTCCGGATG 3'
             ||| |||||
        3'  AAGTCCTACATTTAGCCTAGT 5'
```

or

```
5' AGGCTATGGGCCTCGCGA 3'
               ||||||
            3'  AGCGCTCCGGGTATCGGA 5'
```

The scoring system is as for the Maximum Complementarity argument. In the examples above the scores are 7.00 and 6.00 respectively. Scores are non-negative, and a score of 0.00 indicates that there is no reasonable 3′-anchored global alignment between two oligos. In order to estimate 3′-anchored global alignments for candidate primers and primer pairs, Primer 3 assumes that the sequence from which to choose primers is presented $5' \longrightarrow 3'$. It is nonsensical to provide a larger value for this parameter than for the Maximum (local) Complementarity parameter because the score of a local alignment will always be at least as great as the score of a global alignment.

PRIMER_DEFAULT_PRODUCT (size range list, default 100-300)

A deprecated synonym for PRIMER_PRODUCT_SIZE_RANGE, maintained for v2 compatibility.

PRIMER_FILE_FLAG (boolean, default 0)

If the associated value is non-0, then primer creates two output files for each input SEQUENCE. File <sequence_id>.for lists all acceptable left primers for <sequence_id>, and <sequence_id>.rev lists all acceptable right primers for <sequence_id>, where

<sequence_id> is the value of the PRIMER_SEQUENCE_ID

tag (which must be supplied). In addition, if the input tag

PRIMER_PICK_INTERNAL_OLIGO

is non-0, Primer 3 produces a file

<sequence_id>.int,

which lists all acceptable internal oligos.

PRIMER_MAX_POLY_X (int, default 5)

The maximum allowable length of a mononucleotide repeat, for example AAAAAA.

PRIMER_LIBERAL_BASE (boolean, default 0)

This parameter provides a quick-and-dirty way to get Primer 3 to accept IUB / IUPAC codes for ambiguous bases (i.e. by changing all unrecognized bases to N). If you wish to include an ambiguous base in an oligo, you must set `PRIMER_NUM_NS_ACCEPTED` to a non-0 value.

Perhaps '`-`' and '`*`' should be squeezed out rather than changed to '`N`', but currently they simply converted to `N`'s. The authors invite user comments.

`PRIMER_NUM_RETURN` (int, default 5)

The maximum number of primer pairs to return. Primer pairs returned are sorted by their "quality", in other words by the value of the objective function (where a lower number indicates a better primer pair).

`PRIMER_FIRST_BASE_INDEX` (int, default 0)

This parameter is the index of the first base in the input sequence. For input and output using 1-based indexing (such as that used in GenBank and to which many users are accustomed) set this parameter to 1. For input and output using 0-based indexing set this parameter to 0. (This parameter also affects the indexes in the contents of the files produced when the primer file flag is set.) In the WWW interface this parameter defaults to 1.

Like the arguments governing PCR primer selection, the input tags governing internal oligo selection are divided into sequence input tags and global input tags, with for former being automatically reset after each input record, and the latter persisting until explicitly reset.

Because the laboratory detection step using internal oligos is independent of the PCR amplification procedure, internal oligo tags have defaults that are independent of the parameters that govern the selection of PCR primers. For example, the melting temperature of an oligo used for hybridization might be considerably lower than that used as a PCR primer.

5.3 Internal Oligo "Sequence" Input Tags

`PRIMER_INTERNAL_OLIGO_EXCLUDED_REGION` (interval list, default empty)

Middle oligos may not overlap any region specified by this tag. The associated value must be a space-separated list of

`<start>,<length>`

pairs, where `<start>` is the (0-based) index of the first base of an excluded region, and <length> is its length. Often one would make Target regions excluded regions for internal oligos.

5.4 Internal Oligo "Global" Input Tags

These tags are analogous to the global input tags (those governing primer oligos) discussed above. The exception is

`PRIMER_INTERNAL_OLIGO_SELF_END`

which is meaningless when applied to internal oligos used for hybridization-based detection, since primer-dimer will not occur. We recommend that

`PRIMER_INTERNAL_OLIGO_SELF_END`

be set at least as high as

`PRIMER_INTERNAL_OLIGO_SELF_ANY`.

```
PRIMER_INTERNAL_OLIGO_OPT_SIZE (int, default 20)
PRIMER_INTERNAL_OLIGO_MIN_SIZE (int, default 18)
PRIMER_INTERNAL_OLIGO_MAX_SIZE (int, default 27)
PRIMER_INTERNAL_OLIGO_OPT_TM (float, default 60.0 degrees C)
PRIMER_INTERNAL_OLIGO_MIN_TM (float, default 57.0 degrees C)
PRIMER_INTERNAL_OLIGO_MAX_TM (float, default 63.0 degrees C)
PRIMER_INTERNAL_OLIGO_MIN_GC (float, default 20.0%)
PRIMER_INTERNAL_OLIGO_MAX_GC (float, default 80.0%)
PRIMER_INTERNAL_OLIGO_SALT_CONC (float, default 50.0 mM)
PRIMER_INTERNAL_OLIGO_DNA_CONC (float, default 50.0 nM)
PRIMER_INTERNAL_OLIGO_SELF_ANY (decimal 9999.99, default 12.00)
PRIMER_INTERNAL_OLIGO_MAX_POLY_X (int, default 5)
PRIMER_INTERNAL_OLIGO_SELF_END (decimal 9999.99, default 12.00)
```

6 An Example

One might be interested in performing PCR on an STS with a `CA` repeat in the middle of it. Primers need to be chosen based on the criteria of the experiment.

We need to come up with a boulder-io record to send to Primer 3 via `stdin`. There are lots of ways to accomplish this. We could save the record into a text file called '`input`', and then type the UNIX command '`primer3 < input`'. Or, we could simply type the command '`primer3`', which would start primer with no input. Then, we could type in the record line by line, terminating it with the '`=`' character, and then inserting an `EOF` by holding down the '`ctrl`' button while pressing '`d`'. In any case, we assume that you will be able to get this record into Primer 3's `stdin`. See the man page for your shell if you are having problems with this.

Let's look at the input record itself[1]:

```
PRIMER_SEQUENCE_ID=example
SEQUENCE=GTAGTCAGTAGACNATGACNACTGACGATGCAGACNACACACACACACACAGCACACAGGTATTAGTGGGCCATTCGATCCCGACCCAAATCGATAGCTACGATGACG

TARGET=37,21
PRIMER_OPT_SIZE=18
PRIMER_MIN_SIZE=15
PRIMER_MAX_SIZE=21
PRIMER_NUM_NS_ACCEPTED=1
PRIMER_PRODUCT_SIZE_RANGE=75-100
PRIMER_FILE_FLAG=1
PRIMER_PICK_INTERNAL_OLIGO=1
PRIMER_INTERNAL_OLIGO_EXCLUDED_REGION=37,21 PRIMER_EXPLAIN_FLAG=1
=
```

A breakdown of the reasoning behind each of the `TAG=VALUE` pairs is below:

```
PRIMER_SEQUENCE_ID=example
```

The main intent of this tag is to provide an identifier for the sequence that is meaningful to the user, for example when Primer 3 processes multiple records, and by default this tag is optional. However, this tag is **required** when `PRIMER_FILE_FLAG` is non-0 Because it provides the names of the files that contain lists of oligos that Primer 3 considered.

[1]With the font size used in all the examples, the sequence used in this example would not fit on one line, so the `SEQUENCE`-line is printed in a much smaller font and moved to the left for no other reason than to avoid a line break, which must not occur in the sequence.

```
SEQUENCE=GTAGTCAGTAGACNATGACNACTGACGATGCAGACNACACACACACACACAGCACACAGGTATTAGTGGGCCATTCGATCCCGACCCAAATCGATAGCTACGATGACG
```

The `SEQUENCE` tag is of ultimate importance. Without it, Primer 3 has no idea what to do. This sequence is 92 bases long. Note that there is no newline until the sequence terminates completely.

```
TARGET=37,21
```

There is a simple sequence repeat in our sequence, which starts at base 37, and has a length of 21 bases. We want Primer 3 to choose primers which flank the repeat site, so we let Primer 3 know that we consider this site to be important.

```
PRIMER_OPT_SIZE=18
```

Since our sequence length is rather small (only 92 bases long), we lower the `PRIMER_OPT_SIZE` from 20 to 18. It's more likely that Primer 3 will succeed if it shoots for smaller primers with such a small sequence.

```
PRIMER_MIN_SIZE=15
```

```
PRIMER_MAX_SIZE=21
```

With the lowering of optimal primer size, it's good to lower the minimum and maximum sizes as well.

```
PRIMER_NUM_NS_ACCEPTED=1
```

Again, since we've got such a small sequence with a non-negligible amount of unknown bases (N's) in it, let's make Primer 3's job easier by allowing it to pick primers that have at most 1 unknown base.

```
PRIMER_PRODUCT_SIZE_RANGE=75-100
```

We reduce the product size range from the default of `100-300` because our source sequence is only 108 base pairs long. If we insisted on a product size of 100 base pairs Primer 3 would have few possibilities to choose from.

```
PRIMER_FILE_FLAG=1
```

Since we've got such a small sequence, Primer 3 might fail to pick primers. We want to get the list of primers it considered, then, so that we might manually pick primers ourselves if Primer 3 fails to do so. Setting this flag to 1 will force Primer 3 to output the primers it considered to a

`forward_primer` and a `reverse_primer`

output file.

```
PRIMER_PICK_INTERNAL_OLIGO=1
```

We want to see if Primer 3 can pick an internal oligo for the sequence, so we set this flag to `1` (true).

```
PRIMER_INTERNAL_OLIGO_EXCLUDED_REGION=37,21
```

Normally `CA`-repeats make poor hybridization probes (because they not specific enough). Therefore we exclude the `CA` repeat (which is the `TARGET`) from consideration for the middle oligo.

```
PRIMER_EXPLAIN_FLAG=1
```

We want to see statistics about the oligos and oligo triples (left primer, internal oligo, right primer) that Primer 3 examined.

```
=
```

The '=' character terminates the record.

Tere are some boulderio tags that we never even specified. (INCLUDED_REGION, EXCLUDED_REGION, et al.), which is perfectly legal. For the tags with default values, those defaults will be used in the analysis. For the tags with **no** default values (like TARGET, for instance), the functionality requested by the those tags will simply be absent. It's not the case that we need to surround a simple sequence repeat every time we want to pick primers!

7 Output Tags

For each boulderio record passed into Primer 3 via stdin, exactly one boulder-io record comes out of Primer 3 on stdout. These output records contain everything that the input record contains, plus a subset of the following tag/value pairs. Unless noted by (*), each tag appears for each primer pair returned. The first version is

PRIMER_{LEFT,RIGHT,INTERNAL_OLIGO,PAIR}_<tag_name>.

Tags of additional primers chosen are of the form

PRIMER_{LEFT,RIGHT,INTERNAL_OLIGO,PAIR}_<j>_<tag_name>,

where <j>is an integer from 1 to n, where n ist at most the value of PRIMER_NUM_BEST.

In the descriptions below, 'i,n' represents a start/length pair, 's' represents a string, 'x' represents an arbitrary integer, and 'f' represents a float.

PRIMER_ERROR=s (*)

s describes user-correctible errors detected in the input (separated by semicolons). This tag is absent if there are no errors.

PRIMER_LEFT=i,n

(FORWARD_PRIMER if -v2_compat is set)

The selected left primer (the primer to the left in the input sequence). i is the 0-based index of the start base of the primer, and n is its length.

PRIMER_RIGHT=i,n

(REVERSE_PRIMER if -v2_compat is set)

The selected right primer (the primer to the right in the input sequence). i is the 0-based index of the last base of the primer, and n is its length.

PRIMER_INTERNAL_OLIGO=i,n

(MIDDLE_OLIGO if -v2_compat is set)

The selected internal oligo. Primer 3 outputs this tag if

PRIMER_PICK_INTERNAL_OLIGO

was non-0. If Primer 3 fails to pick a middle oligo upon request, this tag will not be output. `i` is the 0-based index of start base of the internal oligo, and `n` is its length.

`PRIMER_PRODUCT_SIZE=x`

(`PRODUCT_SIZE` if `-v2_compat` is set)

`x` is the product size of the PCR product.

`PRIMER_{LEFT,RIGHT,INTERNAL_OLIGO}_EXPLAIN=s (*)`

`s` is a (more or less) self-documenting string containing statistics on the possiblities that Primer 3 considered in selecting a single oligo. For example

```
PRIMER_LEFT_EXPLAIN=considered 62, too many Ns 53, ok 9
PRIMER_RIGHT_EXPLAIN=considered 62, too many Ns 53, ok 9
```

```
PRIMER_INTERNAL_OLIGO_EXPLAIN=considered 87, too many Ns 39, overlap excluded
region 40, ok 8
```

All the categories are exclusive, except the 'considered' category.

`PRIMER_PAIR_EXPLAIN=s (*)`

`s` is a self-documenting string containing statistics on picking a primer pair (plus internal oligo if requested). For example

```
PRIMER_PAIR_EXPLAIN=considered 81, unacceptable product size 49, no internal oligo
32, ok 0
```

All the categories are exclusive, except the 'considered' category.

`PRIMER_PAIR_QUALITY=f`

The value of the objective function for this pair (lower is better).

`PRIMER_LEFT,RIGHT,INTERNAL_OLIGO_SEQUENCE=s`

The actual sequence of the oligo. The sequence of left primer and internal oligo is presented $5' \longrightarrow 3'$ on the same strand as the input `SEQUENCE` (which must be presented $5' \longrightarrow 3'$). The sequence of the right primer is presented $5' \longrightarrow 3'$ on the opposite strand from the input `SEQUENCE`.

`PRIMER_LEFT,RIGHT,INTERNAL_OLIGO_TM=f`

The melting `TM` for the selected oligo.

`PRIMER_LEFT,RIGHT,INTERNAL_OLIGO_SELF_ANY=f`

`PRIMER_LEFT,RIGHT,INTERNAL_OLIGO_SELF_END=f`

The self-complementarity measures for the selected oligo.

`PRIMER_PAIR_COMPL_ANY=f`

`PRIMER_PAIR_COMPL_END=f`

The inter-pair complementarity measures for the selected left and right primer

`PRIMER_WARNING=s (*)`

lists warnings generated by primer (separated by semicolons); this tag is absent if there are no warnings.

8 Example Output

You should run it youself. Use the file 'example' in this directory as input.

9 Advice for Picking Primers

PCR is usually quite robust with respect to the estimated oligo melting `tm`. However, a low melting `tm` tends to demand short primers with the danger of loss of specificity.

9.1 Cautions

Some of the most important issues in primer picking cannot be addressed in Primer. These are sequence quality (including making sure the sequence is not vector and not chimeric) and avoiding repetitive elements.

Techniques for avoiding problems include a thorough understanding of possible vector contaminants and cloning artifacts coupled with database searches using blast, fasta, or other similarity searching program to screen for vector contaminants and possible repeats. Repbase is a useful source of repeat sequences.

Sequence quality can be controlled by manual trace viewing and quality clipping or automatic quality clipping programs. The beginning of a sequencing read is often problematic because of primer peaks, and the end of the read often contains many low-quality or even meaningless called bases. When picking primers from single-pass sequence it is often best to avoid the first 20 base pairs, and to prefer shorter product sizes or shortened `Included Region` lengths to avoid low-quality sequence at the end of the sequence read.

9.2 What to do if Primer 3 cannot find primers?

Try relaxing various parameters, including the self-complementarity parameters and max and min oligo melting temperatures. For example, for very `A-T`-rich regions you might have to increase maximum primer size or decrease minimum melting temperature. It is usually unwise to reduce the minimum primer size, since small primers are likely to be non-specific. Make sure that there are adequate stretches of non-`N`s in the regions in which you wish to pick primers. If necessary you can also allow an `N` in your primer.

Try setting the `PRIMER_EXPLAIN_FLAG` input tag.

10 Differences from Earlier Versions

10.1 Compared to 0.5

- Completely different input format.
- It has been reported the 0.5 deleted `N`s when they occurred in primers.
- More stringent self-complementarity defaults.
- Primer3 selects internal oligos on request (and produces `.mid` files if requested).

10.2 Compared to v2

- Tag name changes. However, Primer 3 should understand most or all Primer v2 input tags, and should produce v2-compatible output tag names when the `-v2_compat` command-line switch is used.

- The one exception is that the PRIMER_RECOMMEND tag is no longer produced. Instead Primer 3 produces the PRIMER_x_EXPLAIN output tags. The format of the data in this tags is different from the data in v2's PRIMER_RECOMMEND output tag.
- Numerous fixes.
- Uses the PRIMER_SELF_ANY and PRIMER_SELF_END parameters to govern maximum allowable complementarity between left and right primers, as well as complementarity between copies of a single oligo or within a single oligo. This behaviour is very close to that of primer 0.5; self complementarity calculations in v2 were unreliable.
- Primer 3 produces much more output information, including the TMs and self complementarity measures of selected primers.

10.3 Compared to both 0.5 and v2

- The format of the contents of .for, .rev (and .mid) files is different.
- Primer 3 offers the 5 best choices.
- Primer 3 will find a primer pair if any acceptable pair exists.

11 Exit Status Codes

- 0 on normal operation.
- −1 under the following conditions:
  - illegal command-line arguments.
  - unable to fflush stdout.
  - unable to open (for writing and creating) a .for, .rev or .mid file (probably due to a protection problem).
- −2 on out-of-memory.
- −3 empty input.
- −4 error in a "Global" input tag (message in PRIMER_ERROR).

Primer 3 calls abort() and dumps core (if possible) if a programming error is detected by an assertion violation.

SIGINT and SIGTERM are handled essentially as empty input, except the signal received is returned as the exit status and printed to stderr.

In all of the error cases above Primer 3 prints a message to stderr.

12 Acknowledgments

The development of Primer 3 was supported by Howard Hughes Medical Institute and by the National Institutes of Health, National Center for Human Genome Research under grants R01-HG00257 (to David C. Page) and P50-HG00098 (to Eric S. Lander).

We gratefully acknowledge the support of Digital Equipment Corporation, which provided the Alphas which were used for most of the development of Primer 3, and of Centerline Software, Inc., whose TestCenter memory-error, -leak, and test-coverage checker helped us discover and correct a number of otherwise latent errors in Primer 3.

12 Acknowledgments

Primer 3 was written by Helen J. Skaletsky (Howard Hughes Medical Institute, Whitehead Institute) and Steve Rozen (Whitehead Institute/MIT Center for Genome Research), based on the design of earlier versions: Primer 0.5 (Steve Lincoln, Mark Daly, and Eric S. Lander) and Primer v2 (Richard Resnick). This documentation was written by Richard Resnick and Steve Rozen. The web interface was origninally designed by Richard Resnick.

In addition, following is a partial list of people who kindly contributed to the design of Primer 3

- Ernst Molitor
- Carl Foeller

The authors of the current version would be pleased to receive error reports or requests for enhancements. Please send e-mail to

`primer3@genome.wi.mit.edu`

Index