SPAdes 3.5.0 Manual

1. About SPAdes
   1.1. Supported data types
   1.2. SPAdes pipeline
   1.3. SPAdes performance

2. Installation
   2.1. Downloading SPAdes Linux binaries
   2.2. Downloading SPAdes binaries for Mac
   2.3. Downloading and compiling SPAdes source code
   2.4. Verifying your installation

3. Running SPAdes
   3.1. SPAdes input
   3.2. SPAdes command line options
   3.3. Assembling Ion Torrent reads
   3.4. Assembling long Illumina paired reads (2x150 and 2x250)
   3.5. SPAdes output
   3.6. Assembly evaluation

4. Citation

5. Feedback and bug reports

1. About SPAdes

SPAdes – St. Petersburg genome assembler – is intended for both standard isolates and single-cell MDA bacteria assemblies. This manual will help you to install and run SPAdes. SPAdes version 3.5.0 was released under GPLv2 on December 7, 2014 and can be downloaded from http://bioinf.spbau.ru/en/spades.

1.1 Supported data types

The current version of SPAdes works with Illumina or IonTorrent reads and is capable of providing hybrid assemblies using PacBio, Oxford Nanopore and Sanger reads. You can also provide additional contigs that will be used as long reads.

Version 3.5.0 of SPAdes supports paired-end reads, mate-pairs and unpaired reads. SPAdes can take as input several paired-end and mate-pair libraries simultaneously.

Note, that SPAdes was initially designed for small genomes. It was tested on single-cell and standard bacterial and fungal data sets. SPAdes is not intended for larger genomes (e.g. mammalian size genomes) and metagenomic projects. For such purposes you can use it at your own risk.

SPAdes has also a separate module for assembling highly polymorphic diploid genomes. For more information see dipSPAdes manual.

1.2 SPAdes pipeline

SPAdes comes in several separate modules:

- BayesHammer – read error correction tool for Illumina reads, which works well on both single-cell and standard data sets.
- IonHammer – read error correction tool for IonTorrent data, which also works on both types of data.
- SPAdes – iterative short-read genome assembly module; values of K are selected automatically based on the read length and data set type.
- MismatchCorrector – a tool which improves mismatch and short indel rates in resulting contigs and scaffolds; this module uses the BWA tool [Li H. and Durbin R., 2009]; MismatchCorrector is turned off by default, but we recommend to turn it on (see SPAdes options section).
- dipSPAdes – module for assembling highly polymorphic diploid genomes; for more information see dipSPAdes manual.

We recommend to run SPAdes with BayesHammer/IonHammer to obtain high-quality assemblies. However, if you use your own read correction tool, it is possible to turn error correction module off. It is also possible to use only the read error correction stage, if you wish to use another assembler. See the SPAdes options section.

1.3 SPAdes' performance

In this section we give approximate data about SPAdes' performance on two data sets:

- Standard isolate E. coli; 6.2Gb, 28M reads, 2x100bp, insert size ~ 215bp
- MDA single-cell E. coli; 6.3 Gb, 29M reads, 2x100bp, insert size ~ 270bp
We ran SPAdes with default parameters using 16 threads on a server with Intel Xeon 2.27GHz processors. BayesHammer runs in approximately 40 minutes and takes up to 10Gb of RAM to perform read error correction on each data set. Assembly takes about half an hour for the E. coli isolate data set and a bit less than an hour for the E. coli single-cell data set. Both data sets require 9Gb of RAM (see notes below). MismatchCorrector runs for about an hour on standard E. coli, 1.5 hours on the single-cell data set, and requires 1Gb of RAM. All modules also require additional disk space for storing results (corrected reads, contigs, etc) and temporary files. See the table below for more precise values.

<table>
<thead>
<tr>
<th>Data set</th>
<th>E. coli isolate</th>
<th>E. coli single-cell</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage</td>
<td>Time</td>
<td>Peak RAM usage (Gb)</td>
</tr>
<tr>
<td>BayesHammer</td>
<td>40m</td>
<td>9</td>
</tr>
<tr>
<td>SPAdes</td>
<td>27m</td>
<td>9</td>
</tr>
<tr>
<td>MismatchCorrector</td>
<td>53m</td>
<td>1</td>
</tr>
<tr>
<td>Whole pipeline</td>
<td>2h</td>
<td>9</td>
</tr>
</tbody>
</table>

Notes:

- Running SPAdes without preliminary read error correction (e.g. without BayesHammer or IonHammer) will likely require more time and memory.
- Each module removes its temporary files as soon as it finishes.
- SPAdes uses 512 Mb per thread for buffers, which results in higher memory consumption. If you set memory limit manually, SPAdes will use smaller buffers and thus less RAM.
- Performance statistics is given for SPAdes version 3.5.

2. Installation

SPAdes requires a 64-bit Linux system or Mac OS and Python (supported versions are 2.4, 2.5, 2.6, 2.7, 3.2 and 3.3) to be pre-installed on it. To obtain SPAdes you can either download binaries or download source code and compile it yourself.

2.1 Downloading SPAdes Linux binaries

To download SPAdes Linux binaries and extract them, go to the directory in which you wish SPAdes to be installed and run:

```
wget http://spades.bioinf.spbau.ru/release3.5.0/SPAdes-3.5.0-Linux.tar.gz
tar -xzf SPAdes-3.5.0-Linux.tar.gz
cd SPAdes-3.5.0-Linux/bin/
```

In this case you do not need to run any installation scripts – SPAdes is ready to use. The following files will be placed in the bin directory:

- `spades.py` (main executable script)
- `hammer` (read error correcting module for Illumina reads)
- `ionhammer` (read error correcting module for IonTorrent reads)
- `spades` (assembly module)
- `bwa-spades` (BWA alignment module which is required for mismatch correction)
- `corrector` (mismatch correction module)
- `dipspades.py` (main executable script for dipSPAdes)
- `dipspades` (assembly module for highly polymorphic diploid genomes)

We also suggest adding SPAdes installation directory to the PATH variable.

2.2 Downloading SPAdes binaries for Mac

To obtain SPAdes binaries for Mac, go to the directory in which you wish SPAdes to be installed and run:

```
curl http://spades.bioinf.spbau.ru/release3.5.0/SPAdes-3.5.0-Darwin.tar.gz -o SPAdes-3.5.0-Darwin.tar.gz
tar -xzf SPAdes-3.5.0-Darwin.tar.gz
cd SPAdes-3.5.0-Darwin/bin/
```

Just as in Linux, SPAdes is ready to use and no further installation steps are required. You will get the same files in the bin directory:

- `spades.py` (main executable script)
2.3 Downloading and compiling SPAdes source code

If you wish to compile SPAdes by yourself you will need the following libraries to be pre-installed:

- g++ (version 4.7 or higher)
- cmake (version 2.8.8 or higher)
- zlib
- libbz2

If you meet these requirements, you can download the [SPAdes source code](http://spades.bioinf.spbau.ru/release3.5.0/SPAdes-3.5.0.tar.gz):

```bash
wget http://spades.bioinf.spbau.ru/release3.5.0/SPAdes-3.5.0.tar.gz
tar -xzf SPAdes-3.5.0.tar.gz
cd SPAdes-3.5.0
```

and build it with the following script:

```bash
./spades_compile.sh
```

SPAdes will be built in the directory `.bin`. If you wish to install SPAdes into another directory, you can specify full path of destination folder by running the following command in bash or sh:

```bash
PREFIX=<destination_dir> ./spades_compile.sh
```

for example:

```bash
PREFIX=/usr/local ./spades_compile.sh
```

which will install SPAdes into `/usr/local/bin`.

After installation you will get the same files in `/bin` (or `<destination_dir>/bin` if you specified PREFIX) directory:

- spades.py (main executable script)
- hammer (read error correcting module for Illumina reads)
- ionhammer (read error correcting module for IonTorrent reads)
- spades (assembly module)
- bwa-spades (BWA alignment module which is required for mismatch correction)
- corrector (mismatch correction module)
- dippspades.py (main executable script for dipSPAdes)
- dippspades (assembly module for highly polymorphic diploid genomes)

We also suggest adding SPAdes installation directory to the PATH variable.

2.4 Verifying your installation

For testing purposes, SPAdes comes with a toy data set (reads that align to first 1000 bp of *E. coli*). To try SPAdes on this data set, run:

```bash
<spades installation dir>/spades.py --test
```

If you added SPAdes installation directory to the PATH variable, you can run:
spades.py --test

For the simplicity we further assume that SPAdes installation directory is added to the PATH variable.

If the installation is successful, you will find the following information at the end of the log:

===== Assembling finished.
* Corrected reads are in spades_test/corrected/
* Assembled contigs are in spades_test/contigs.fasta (contigs.fastg)
* Assembled scaffolds are in spades_test/scaffolds.fasta (scaffolds.fastg)
====== SPAdes pipeline finished.

SPAdes log can be found here: /home/andrey/ablab/algorithmic-biology/assembler/spades_test/spades.log

Thank you for using SPAdes!

3. Running SPAdes

3.1 SPAdes input

SPAdes takes as input paired-end reads, mate-pairs and single (unpaired) reads in FASTA and FASTQ. For IonTorrent data SPAdes also supports unpaired reads in unmapped BAM format (like the one produced by Torrent Server). However, in order to run read error correction, reads should be in FASTQ or BAM format. Sanger, Oxford Nanopore and PacBio CLR reads can be provided in both formats since SPAdes does not run error correction for these types of data.

To run SPAdes 3.5.0 you need at least one library of the following types:

- Illumina paired-end/high-quality mate-pairs/unpaired reads
- Ion Torrent paired-end/high-quality mate-pairs/unpaired reads
- PacBio CCS reads

Illumina and Ion Torrent libraries should not be assembled together. All other types of input data are compatible. SPAdes should not be used if only PacBio CLR, Oxford Nanopore, Sanger reads or additional contigs are available.

SPAdes supports mate-pair only assembly. However, we recommend to use only high-quality mate-pair libraries in this case (e.g. that do not have a paired-end part). We tested mate-pair only pipeline using Illumina Nextera mate-pairs. See more here.

Current version SPAdes also supports Lucigen NxSeq® Long Mate Pair libraries, which always have forward-reverse orientation. If you wish to use Lucigen NxSeq® Long Mate Pair reads, you will need Python `regex library` to be pre-installed on your machine. You can install it with Python `pip-installer`:

```
pip install regex
```

or with the `Easy Install` Python module:

```
easy_install regex
```

Notes:

- It is not recommended to run SPAdes on PacBio reads with low coverage (less than 5).
- We suggest not to run SPAdes on PacBio reads for large genomes.
- SPAdes accepts gzip-compressed files.

Read-pair libraries

By using command line interface, you can specify up to five different paired-end libraries, up to five mate-pair libraries and also up to five high-quality mate-pair ones. If you wish to use more, you can use `YAML data set file`. We further refer to paired-end and mate-pair libraries simply as to read-pair libraries.

By default, SPAdes assumes that paired-end and high-quality mate-pair reads have forward-reverse (fr) orientation and usual mate-pairs have reverse-forward (rf) orientation. However, different orientations can be set for any library by using SPAdes options.
To distinguish reads in pairs we refer to them as left and right reads. For forward-reverse orientation, the forward reads correspond to the left reads and the reverse reads, to the right. Similarly, in reverse-forward orientation left and right reads correspond to reverse and forward reads, respectively, etc.

Each read-pair library can be stored in several files or several pairs of files. Paired reads can be organized in two different ways:

- In file pairs. In this case left and right reads are placed in different files and go in the same order in respective files.
- In merged files. In this case, the reads are interlaced, so that each right read goes after the corresponding paired left read.

For example, Illumina produces paired-end reads in two files: s_1_1_sequence.txt and s_1_2_sequence.txt. If you choose to store reads in file pairs make sure that for every read from s_1_1_sequence.txt the corresponding paired read from s_1_2_sequence.txt is placed in the respective paired file on the same line number. If you choose to use merged files, every read from s_1_1_sequence.txt should be followed by the corresponding paired read from s_1_2_sequence.txt.

Unpaired (single-read) libraries

By using command line interface, you can specify up to five different single-read libraries. To input more libraries, you can use YAML data set file.

Single libraries are assumed to have high quality and a reasonable coverage. For example, you can provide PacBio CSS reads as a single-read library. Additionally, if you have merged a paired-end library with overlapping read-pairs (for example, using FLASH), you can provide the resulting reads as a single-read library.

Note, that you should not specify PacBio CLR, Sanger reads or additional contigs as single-read libraries, each of them has a separate option.

PacBio and Oxford Nanopore reads

SPAdes can take as an input an unlimited number of PacBio and Oxford Nanopore libraries.

PacBio CLR and Oxford Nanopore reads are used for hybrid assemblies (e.g. with Illumina or IonTorrent). There is no need to pre-correct this kind of data. SPAdes will use PacBio CLR and Oxford Nanopore reads for gap closure and repeat resolution.

For PacBio you just need to have filtered subreads in FASTQ/FASTA format. Provide these filtered subreads using --pacbio option. Oxford Nanopore reads are provided with --nanopore option.

PacBio CCS reads or pre-corrected (using third-party software) PacBio CLR / Oxford Nanopore reads can be simply provided as single reads to SPAdes.

Additional contigs

In case you have contigs of the same genome generated by other assembler(s) and you wish to merge them into SPAdes assembly, you can specify additional contigs using --trusted-contigs or --untrusted-contigs. First option is used when high quality contigs are available. These contigs will be used for graph construction, gap closure and repeat resolution. Second option is used for less reliable contigs that may have more errors or contigs of unknown quality. These contigs will be used only for gap closure and repeat resolution. The number of additional contigs is unlimited.

Note, that SPAdes does not perform assembly using genomes of closely-related species. Only contigs of the same genome should be specified.

3.2 SPAdes command line options

To run SPAdes from the command line, type

```
spades.py [options] -o <output_dir>
```

Note that we assume that SPAdes installation directory is added to the PATH variable (provide full path to SPAdes executable otherwise: `<spades installation dir>/spades.py`).

Basic options

- `-o <output_dir>`
  Specify the output directory. Required option.

- `--sc`
  This flag is required for MDA (single-cell) data.
This flag is required when assembling IonTorrent data. Allows BAM files as input. Carefully read section 3.3 before using this option.

--test
Runs SPAdes on the toy data set; see section 2.3.

-h (or --help)
Prints help.

Pipeline options

--only-error-correction
Performs read error correction only.

--only-assembler
Runs assembly module only.

--careful
Tries to reduce the number of mismatches and short indels. Also runs MismatchCorrector – a post processing tool, which uses BWA tool (comes with SPAdes). This option is recommended.

--continue
Continues SPAdes run from the specified output folder starting from the last available check-point. Check-points are made after:

- error correction module is finished
- iteration for each specified K value of assembly module is finished
- mismatch correction is finished for contigs or scaffolds

For example, if specified K values are 21, 33 and 55 and SPAdes was stopped or crashed during assembly stage with K = 55, you can run SPAdes with the --continue option specifying the same output directory. SPAdes will continue the run starting from the assembly stage with K = 55. Error correction module and iterations for K equal to 21 and 33 will not be run again. Note that all options except -o <output_dir> are ignored if --continue is set.

--restart-from <check_point>
Restart SPAdes run from the specified output folder starting from the specified check-point. Check-points are:

- ec – start from error correction
- as – restart assembly module from the first iteration
- k<int> – restart from the iteration with specified k values, e.g. k55
- mc – restart mismatch correction

In comparison to the --continue option, you can change some of the options when using --restart-from. You can change any option except: all basic options, all options for specifying input data (including --dataset), --only-error-correction option and --only-assembler option. For example, if you ran assembler with k values 21,33,55 without mismatch correction, you can add one more iteration with k=77 and run mismatch correction step by running SPAdes with following options:

--restart-from k55 -k 21,33,55,77 --mismatch-correction -o <previous_output_dir>

Since all files will be overwritten, do not forget to copy your assembly from the previous run if you need it.

--disable-gzip-output
Forces read error correction module not to compress the corrected reads. If this options is not set, corrected reads will be in *.fastq.gz format.

Input data

Specifying one library (previously used format)

--12 <file_name>
File with interlaced forward and reverse paired-end reads.

-1 <file_name>
File with forward reads.

-2 <file_name>
File with reverse reads.

-s <file_name>
File with unpaired reads.

Specifying multiple libraries (new format)
- **Single-read libraries**

  ```
  -s <#> <file_name>
  ```

  File for single-read library number `<#>` (=1,2,3,4,5). For example, for the first paired-end library the option is:

  ```
  -s 1 <file_name>
  ```

  Do not use `-s` options for single-read libraries, since it specifies unpaired reads for the first paired-end library.

- **Paired-end libraries**

  ```
  -pe<#>-12 <file_name>
  ```

  File with interlaced reads for paired-end library number `<#>` (=1,2,3,4,5). For example, for the first single-read library the option is:

  ```
  -pe1-12 <file_name>
  ```

  ```
  -pe<#>-1 <file_name>
  ```

  File with left reads for paired-end library number `<#>` (=1,2,3,4,5).

  ```
  -pe<#>-2 <file_name>
  ```

  File with right reads for paired-end library number `<#>` (=1,2,3,4,5).

  ```
  -pe<#>-s <file_name>
  ```

  File with unpaired reads from paired-end library number `<#>` (=1,2,3,4,5)

  For example, paired reads can become unpaired during the error correction procedure.

  ```
  -pe<#>-<or> <file_name>
  ```

  Orientation of reads for paired-end library number `<#>` (=1,2,3,4,5; `<or>` = "fr","rf","ff").

  The default orientation for paired-end libraries is forward-reverse. For example, to specify reverse-forward orientation for the second paired-end library, you should use the flag:

  ```
  -pe2-rf
  ```

- **Mate-pair libraries**

  ```
  -mp<#>-12 <file_name>
  ```

  File with interlaced reads for mate-pair library number `<#>` (=1,2,3,4,5).

  ```
  -mp<#>-1 <file_name>
  ```

  File with left reads for mate-pair library number `<#>` (=1,2,3,4,5).

  ```
  -mp<#>-2 <file_name>
  ```

  File with right reads for mate-pair library number `<#>` (=1,2,3,4,5).

  ```
  -mp<#>-<or> <file_name>
  ```

  Orientation of reads for mate-pair library number `<#>` (=1,2,3,4,5; `<or>` = "fr","rf","ff").

  The default orientation for mate-pair libraries is reverse-forward. For example, to specify forward-forward orientation for the first mate-pair library, you should use the flag:

  ```
  -mp1-ff
  ```

- **High-quality mate-pair libraries** (can be used for mate-pair only assembly)

  ```
  -hqmp<#>-12 <file_name>
  ```

  File with interlaced reads for high-quality mate-pair library number `<#>` (=1,2,3,4,5).

  ```
  -hqmp<#>-1 <file_name>
  ```

  File with left reads for high-quality mate-pair library number `<#>` (=1,2,3,4,5).

  ```
  -hqmp<#>-2 <file_name>
  ```

  File with right reads for high-quality mate-pair library number `<#>` (=1,2,3,4,5).

  ```
  -hqmp<#>-s <file_name>
  ```

  File with unpaired reads from high-quality mate-pair library number `<#>` (=1,2,3,4,5)

  ```
  -hqmp<#>-<or> <file_name>
  ```

  Orientation of reads for high-quality mate-pair library number `<#>` (=1,2,3,4,5; `<or>` = "fr","rf","ff").

  The default orientation for high-quality mate-pair libraries is forward-reverse. For example, to specify reverse-forward orientation for the first high-quality mate-pair library, you should use the flag:

  ```
  -hqmp1-rf
  ```

- **Lucigen NxSeq® Long Mate Pair libraries** (see section 3.1 for details)

  ```
  -nxmate<#>-1 <file_name>
  ```

  File with left reads for Lucigen NxSeq® Long Mate Pair library number `<#>` (=1,2,3,4,5).

  ```
  -nxmate<#>-2 <file_name>
  ```

  File with right reads for Lucigen NxSeq® Long Mate Pair library number `<#>` (=1,2,3,4,5).

  **Specifying data for hybrid assembly**

  ```
  --paciobio <file_name>
  ```
File with PacBio CLR reads. For PacBio CCS reads use `-s` option. More information on PacBio reads is provided in section 3.1.

```
--nanopore <file_name>
  File with Oxford Nanopore reads.

--sanger <file_name>
  File with Sanger reads

--trusted-contigs <file_name>
  Reliable contigs of the same genome, which are likely to have no misassemblies and small rate of other errors (e.g. mismatches and indels). This option is not intended for contigs of the related species.

--untrusted-contigs <file_name>
  Contigs of the same genome, quality of which is average or unknown. Contigs of poor quality can be used but may introduce errors in the assembly. This option is also not intended for contigs of the related species.
```

**Specifying input data with YAML data set file (advanced)**

An alternative way to specify an input data set for SPAdes is to create a YAML data set file. By using a YAML file you can provide an unlimited number of paired-end, mate-pair and unpaired libraries. Basically, YAML data set file is a text file, in which input libraries are provided as a comma-separated list in square brackets. Each library is provided in braces as a comma-separated list of attributes. The following attributes are available:

- orientation ("fr", "rf", "ff")
- type ("paired-end", "mate-pairs", "hq-mate-pairs", "single", "pacific", "nanopore", "sanger", "trusted-contigs", "untrusted-contigs")
- interlaced reads (comma-separated list of files with interlaced reads)
- left reads (comma-separated list of files with left reads)
- right reads (comma-separated list of files with right reads)
- single reads (comma-separated list of files with single reads)

To properly specify a library you should provide its type and at least one file with reads. Orientation is an optional attribute. Its default value is "fr" (forward-reverse) for paired-end libraries and "rf" (reverse-forward) for mate-pair libraries.

The value for each attribute is given after a colon. Comma-separated lists of files should be given in square brackets. For each file you should provide its full path in double quotes. Make sure that files with right reads are given in the same order as corresponding files with left reads.

For example, if you have one paired-end library split into two pairs of files:

```
lib_pe1_left_1.fastq
lib_pe1_right_1.fastq
lib_pe1_left_2.fastq
lib_pe1_right_2.fastq
```

one mate-pair library:

```
lib_mp1_left.fastq
lib_mp1_right.fastq
```

and PacBio CCS and CLR reads:

```
pacbio_css.fastq
pacbio_clr.fastq
```

YAML file should look like this:

```
[  
    
    {      
      orientation: "fr",  
      type: "paired-end",   
      right reads: [    
          
            
              
                /FULL_PATH_TO_DATASET/lib_pe1_right_1.fastq",  
            
                
                /FULL_PATH_TO_DATASET/lib_pe1_right_2.fastq"    
          
        ],  
      left reads: [    
            
                /FULL_PATH_TO_DATASET/lib_pe1_left_1.fastq",  
            
                /FULL_PATH_TO_DATASET/lib_pe1_left_2.fastq"    
          
        ]  
    
  ]
```

Once you have created a YAML file save it with .yaml extension (e.g. as my_data_set.yaml) and run SPAdes using the --dataset option:

```
--dataset <your YAML file>
```

Notes:

- The --dataset option cannot be used with any other options for specifying input data.
- We recommend to nest all files with long reads of the same data type in a single library block.

Note that the --dataset option cannot be used with any other options for specifying input data.

**Advanced options**

- **-t** *(or --threads)*
  Number of threads. The default value is 16.

- **-m** *(or --memory)*
  Set memory limit in Gb. SPAdes terminates if it reaches this limit. The default value is 250 Gb. Actual amount of consumed RAM will be below this limit. Make sure this value is correct for the given machine. SPAdes uses the limit value to automatically determine the sizes of various buffers, etc.

- **--tmp-dir** *(dir_name)*
  Set directory for temporary files from read error correction. The default value is <output_dir>/corrected/tmp

- **-k** *(<int>,...)*
  Comma-separated list of k-mer sizes to be used (all values must be odd, less than 128 and listed in ascending order). If --sc is set the default value are 21,33,55. For multicell data sets K values are automatically selected using maximum read length *(see note for assembling long Illumina paired reads for details)*. To properly select K values for IonTorrent data read section 3.3.

- **--cov-cutoff** *(float)*
  Read coverage cutoff value. Must be a positive float value, or 'auto', or 'off'. Default value is 'off'. When set to 'auto' SPAdes automatically computes coverage threshold using conservative strategy.

- **--phred-offset** *(33 or 64)*
  PHRED quality offset for the input reads, can be either 33 or 64. It will be auto-detected if it is not specified.

**Examples**

To test the toy data set, you can also run the following command from the SPAdes bin directory:

```
spades.py --pe1-1 ../share/spades/test_dataset/ecoli_1K_1.fq.gz
--pe1-2 ../share/spades/test_dataset/ecoli_1K_2.fq.gz -o spades_test
```

If you have your library separated into several pairs of files, for example:

```
libl_forward_1.fastq
```
make sure that corresponding files are given in the same order:

```
spades.py --pe1-1 lib1_forward_1.fastq --pe1-2 lib1_reverse_1.fastq \
--pe1-1 lib1_forward_2.fastq --pe1-2 lib1_reverse_2.fastq \
-o spades_output
```

Files with interlacing paired-end reads or files with unpaired reads can be specified in any order with one file per option, for example:

```
spades.py --pe1-12 lib1_1.fastq --pe1-12 lib1_2.fastq \
--pe1-s lib1_unpaired_1.fastq --pe1-s lib1_unpaired_2.fastq \
-o spades_output
```

If you have several paired-end and mate-pair reads, for example:

- paired-end library 1
  ```
  lib_pe1_left.fastq
  lib_pe1_right.fastq
  ```

- mate-pair library 1
  ```
  lib_mp1_left.fastq
  lib_mp1_right.fastq
  ```

- mate-pair library 2
  ```
  lib_mp2_left.fastq
  lib_mp2_right.fastq
  ```

make sure that files corresponding to each library are grouped together:

```
spades.py --pe1-1 lib_pe1_left.fastq --pe1-2 lib_pe1_right.fastq \
--mp1-1 lib_mp1_left.fastq --mp1-2 lib_mp1_right.fastq \
--mp2-1 lib_mp2_left.fastq --mp2-2 lib_mp2_right.fastq \
-o spades_output
```

If you have IonTorrent unpaired reads, PacBio CLR and additional reliable contigs:

```
it_reads.fastq
pacbio克拉.fastq
contigs.fasta
```

run SPAdes with the following command:

```
spades.py --iontorrent -s it_reads.fastq \
--pacbio pacbio克拉.fastq --trusted-contigs contigs.fasta \
-o spades_output
```

If a single-read library is splitted into several files:

```
unpaired1_1.fastq
unpaired1_2.fastq
unpaired1_3.fasta
```

specify them as one library:
spades.py --s1 unpaired1_1.fastq \
--s1 unpaired1_2.fastq --s1 unpaired1_3.fastq \
-o spades_output

All options for specifying input data can be mixed if needed, but make sure that files for each library are grouped and files with left and right paired reads are listed in the same order.

### 3.3 Assembling IonTorrent reads

Only FASTQ or BAM files are supported as input.

The selection of k-mer length is non-trivial for IonTorrent. If the dataset is more or less conventional (good coverage, not high GC, etc), then use our [recommendation for long reads](http://spades.bioinf.spbau.ru/release3.5.0/manual.html) (e.g. assemble using k-mer lengths 21,33,55,77,99,127). However, due to increased error rate some changes of k-mer lengths (e.g. selection of shorter ones) may be required. For example, if you ran SPAdes with k-mer lengths 21,33,55,77 and then decided to assemble the same data set using more iterations and larger values of K, you can run SPAdes once again specifying the same output folder and the following options: **--restart-from k77 --k 21,33,55,77,99,127 --mismatch-correction -o <previous_output_dir>**. Do not forget to copy contigs and scaffolds from the previous run. We're planning to tackle issue of selecting k-mer lengths for IonTorrent reads in next versions.

You may need no error correction for Hi-Q enzyme at all. However, we suggest trying to assemble your data with and without error correction and select the best variant.

For non-trivial datasets (e.g. with high GC, low or uneven coverage) we suggest to enable single-cell mode (setting **--sc** option) and use k-mer lengths of 21,33,55.

### 3.4 Assembling long Illumina paired reads (2x150 and 2x250)

Recent advances in DNA sequencing technology have led to a rapid increase in read length. Nowadays, it is a common situation to have a data set consisting of 2x150 or 2x250 paired-end reads produced by Illumina MiSeq or HiSeq2500. However, the use of longer reads alone will not automatically improve assembly quality. An assembler that can properly take advantage of them is needed.

SPAdes' use of iterative k-mer lengths allows benefiting from the full potential of the long paired-end reads. Currently one has to set the assembler options up manually, but we plan to incorporate automatic calculation of necessary options soon.

Please note that in addition to the read length, the insert length also matters a lot. It is not recommended to sequence a 300bp fragment with a pair of 250bp reads. We suggest using 350-500 bp fragments with 2x150 reads and 550-700 bp fragments with 2x250 reads.

**Multi-cell data set with read length 2x150**

Make sure your reads are corrected prior to assembly with Quake (recommended), or BayesHammer (integrated into SPAdes pipeline).

If you have enough coverage (50x+), then you may want to try to set k-mer lengths of 21, 33, 55, 77 (selected by default for reads with length 150bp).

Make sure you run assembler with the **--careful** option to minimize number of mismatches in the final contigs.

We recommend that you check the SPAdes log file at the end of the each iteration to control the average coverage of the contigs.

For reads corrected prior to running the assembler:

```bash
spades.py -k 21,33,55,77 --careful --only-assembler <your reads> -o spades_output
```

To correct and assemble the reads:

```bash
spades.py -k 21,33,55,77 --careful <your reads> -o spades_output
```

**Multi-cell data set with read lengths 2 x 250**

Make sure your reads are corrected prior to assembly with Quake (recommended), or BayesHammer (integrated into SPAdes pipeline).

By default we suggest to increase k-mer lengths in increments of 22 until the k-mer length reaches 127. The exact length of the k-mer length:

```bash
spades.py -k 21,33,55,77 --careful --only-assembler <your reads> -o spades_output
```

To correct and assemble the reads:

```bash
spades.py -k 21,33,55,77 --careful <your reads> -o spades_output
```
mer depends on the coverage: k-mer length of 127 corresponds to 50x k-mer coverage and higher. For read length 250bp SPAdes automatically chooses K values equal to 21, 33, 55, 77, 99, 127.

Make sure you run assembler with --careful option to minimize number of mismatches in the final contigs.

We recommend you to check the SPAdes log file at the end of the each iteration to control the average coverage of the contigs.

For reads corrected prior to running the assembler:

```
spades.py -k 21,33,55,77,99,127 --careful --only-assembler <your reads> -o spades_output
```

To correct and assemble the reads:

```
spades.py -k 21,33,55,77,99,127 --careful <your reads> -o spades_output
```

**Single-cell data set with read lengths 2 x 150 or 2 x 250**

The default k-mer lengths are recommended. For single-cell data sets SPAdes selects k-mer sizes 21, 33 and 55.

However, it might be tricky to fully utilize the advantages of long reads you have. Consider contacting us for more information and to discuss assembly strategy.

### 3.5 SPAdes output

SPAdes stores all output files in `<output_dir>` , which is set by the user.

- `<output_dir>/corrected/` directory contains reads corrected by BayesHammer in `*.fastq.gz` files; if compression is disabled, reads are stored in uncompressed `*.fastq` files
- `<output_dir>/contigs.fasta` contains resulting contigs
- `<output_dir>/contigs.fasta` containing contigs in **FASTG format**
- `<output_dir>/scaffolds.fasta` contains resulting scaffolds
- `<output_dir>/scaffolds.fasta` containing scaffolds in **FASTG format**

The full list of `<output_dir>` content is presented below:

- `contigs.fasta`, `contigs.fasta` – resulting contigs
- `scaffolds.fasta`, `scaffolds.fasta` – resulting scaffolds
- `corrected/` – files from read error correction
- `config/` – configuration files for read error correction
- `corrected.yaml` – internal configuration file
- `Output files with corrected reads`
- `params.txt` – information about SPAdes parameters in this run
- `spades.log` – SPAdes log
- `dataset.info` – internal configuration file
- `input dataset.yaml` – internal YAML data set file
- `K=<##>/` – directory containing files from the run with K=<##>

SPAdes will overwrite these files and directories if they exist in the specified `<output_dir>`.

### 3.6 Assembly evaluation

**QUAST** may be used to generate summary statistics (N50, maximum contig length, GC %, # genes found in a reference list or with built-in gene finding tools, etc.) for a single assembly. It may also be used to compare statistics for multiple assemblies of the same data set (e.g., SPAdes run with different parameters, or several different assemblers).

### 4. Citation

If you use SPAdes in your research, please include Nurk, Bankevich et al., 2013 in your reference list. You can also add Bankevich, Nurk et al., 2012 instead.

In addition, we would like to list your publications that use our software on our website. Please email the reference, the name of your lab, department and institution to spades.support@bioinf.spbau.ru.

### 5. Feedback and bug reports
Your comments, bug reports, and suggestions are very welcomed. They will help us to further improve SPAdes.

If you have any troubles running SPAdes, please send us params.txt and spades.log from the directory <output_dir>.

Address for communications: spades.support@bioinf.spbau.ru.