MacVector 17

for Mac OS X

Assembling Plasmids from NGS data using MacVector with Assembler

Mailector In

Software for Scientists

Copyright statement

Copyright MacVector, Inc, 2019. All rights reserved.

This document contains proprietary information of **MacVector**, **Inc** and its licensors. It is their exclusive property. It may not be reproduced or transmitted, in whole or in part, without written agreement from **MacVector**, **Inc**.

The software described in this document is furnished under a license agreement, a copy of which is packaged with the software. The software may not be used or copied except as provided in the license agreement.

MacVector, Inc reserves the right to make changes, without notice, both to this publication and to the product it describes. Information concerning products not manufactured or distributed by **MacVector, Inc** is provided without warranty or representation of any kind, and **MacVector, Inc** will not be liable for any damages.

This version of the Assembler tutorial was published in May 2019.

Contents

OVERVIEW	4
OPTIMIZATION OF VELVET AND SPADES PARAMETERS	5
HOW TO SPLIT FASTQ FILES	5
ASSEMBLING WITH VELVET	7
ANALYZING THE CONSENSUS SEQUENCE	11
Circularizing Using a Right-Click Menu	11
Circularizing Using a Dot-plot Analysis	12
REPEATING THE ASSEMBLY	17
ASSEMBLY USING SPADES	18
CONFIRMING THE CORRECT SEQUENCE USING THREE CONTIGS	20
Determining Orientation	20
Changing Circular Origin	22
CONFIRM IDENTITY USING MULTIPLE SEQUENCE ALIGNMENT	23

Overview

The tutorial will show you how to use MacVector with Assembler to assemble plasmid sequences from NGS data. This particular example uses a sample set for an anonymous plasmid consisting of a pair of files;

plasmid-1_S5_L001_R1_001.fastq

plasmid-1_S5_L001_R2_001.fastq

These files are each 285.3 MB in size and each contain approximately 1,375,000 reads of 75nt in length for a total number of sequenced residues of approximately 20.6 million. The plasmid is 8,859bp in size, so this means every residue in the plasmid has been sequenced an average of 2,328 times. This massive oversequencing causes some considerable problems – most of the *de novo* assemblers have been tuned to assemble sequences such as bacterial chromosomes, where you might have a 1 Mb to 10 Mb sequence with an average of just 20-200 fold, or even less.

MacVector uses two different assemblers for *de novo* assembly of NGS reads.

Velvet: We chose this because it is capable of assembling 20-50million+ reads into 10 Mb+ sequences with relatively minimal memory requirements, so it can be used on personal Macintosh computers with as little as 8 GB of RAM. However, the more RAM you have the better.

SPAdes: This is a more recent assembler with an even smaller memory footprint that does a slightly better job getting past repeats, resulting in longer contigs. It also requires less tweaking of parameters to get optimal assemblies. However, it is significantly slower than *Velvet*.

As with most of these *de novo* assemblers, you can supply *Velvet* or *SPAdes* with too much data. When your coverage is only ~100x or less, *Velvet* will ignore the occasional read that has a sequencing error. However, once you get into the 200+x range, the ~5% or so of reads that have errors suddenly can be assembled into their own contigs due to random similar errors. This confuses the assembler as it thinks it is trying to resolve closely related direct repeats. Eventually it gets confused enough that it can't resolve all of the similar contigs and typically gives up and reports only those contigs that have few ambiguities. These often represent just a small portion of the sequenced molecule.

Experiments with the plasmid data set (see below) indicate that only the first 10,000 reads of each file is really required to generate a clean unambiguous sequence for the plasmid. $20,000 \times 75$ nt reads = 169x coverage of the 8.9 kb plasmid.

This tutorial will run through the optimized assembly of the plasmid, using both assemblers, and also showing how you can repeat the assembly in triplicate and confirm the results to be confident the sequence is correct.

Optimization of Velvet and SPAdes Parameters

The plasmid data set was split into a series of smaller files and used in a comprehensive set of assembly experiments using MacVector. The *Velvet* implementation in MacVector can only take a single KMER value, and it should be an odd value between 11 and 255. In general, a value of about two-thirds of the average length of the input reads is a good place to start. During assembly, reads are initially aligned by looking for perfect matches of length KMER, so there is no point to having KMER be longer than the average length of the reads. The *SPAdes* implementation lets you input multiple values and it will try them all, keeping the results for the value it believes to be optimal. In theory, you could enter every potential (odd) KMER value up to the length of the reads, though in practice that results in a very slow assembly. In the example below, *SPAdes* was run with values of "31,41,51,61,71";

pRGN7782 - 8859bp					VELVE	VELVET (KMER)					SPADES (K	MERS)
Total Reads	35	39	43	47	51	55	59	63	67	71	31,41,51,6	1,71
20,000	8,870	8,897	8,901	8,905	8,909	8,830	8,795	8,795	2,754	531	8,930	
40,000	8,870	8,897	8,901	8,905	8,909	8,913	8,917	8,921	8,806	1,469	8,930	
60,000	8,870	8,897	8,901	8,905	8,909	8,913	8,917	8,921	8,890	2,281	8,930	
100,000	7,855	7,886	7,894	8,891	8,895	8,899	8,917	8,921	8,890	6,542	8,930	
160,000	7,844	7,886	7,894	7,914	8,909	8,899	8,917	8,921	8,923	8,561	8,930	
240,000	7,847	7,855	7,884	7,902	8,348	8,899	8,917	8,921	8,925	8,686	8,930	
350,000	7,722	7,855	7,877	7,355	7,363	7,371	8,917	8,921	8,925	8,928	8,930	
500,000	7,723	7,873	7,877	7,892	7,900	7,908	8,903	8,921	8,925	8,929	8,930	
700,000		7,713	7,716	7,893	7,874	8,889	7,926	8,907	8,925	8,929	8,930	
1,000,000			4,634	7,721	7,724	7,705	7,926	8,907	8,925	8,929	8,930	
2,740,000				683	8,522	7,519	7,480	7,870	7,546	8,929	4,445	

The above table lists the longest contig resulting from each assembly. Those in black could be circularized to create the correct plasmid sequence. Those in red could not.

It is immediately obvious that there is a relationship between the number of reads in the assembly and the optimal *Velvet* KMER. Assemblies using 60,000 reads (2 x 30,000) gave the best results with the broadest range of KMER values. With the multiple KMER values used, *SPAdes* did a much better job of successfully assembling the complete sequence from a wide range of input reads, but even so, it could not assembly the complete set.

How to Split Fastq Files

MacVector does not have a built-in function to split fastq files. However, over the years we have developed a few utilities to help this. One such utility is *"SplitFastqFile.app"*. You can download this from our web site on the macvector.com/Downloads page.

After downloading SplitFastqFile.app.zip, make sure it has been extracted (double-click to extract if it has not) and move *SplitFastqFile.app* to a convenient location e.g. your desktop.

Locate the fastq file(s) you want to split into smaller segments, select them (you can select more than one if you wish), then drag and drop them onto the *SplitFastqFile.app* icon.

You will first be prompted for the number of reads to be saved into each split file;

	How many reads sh file?:	nould be saved in	each split
\checkmark	30000		
		Cancel	ОК

Lets use 30,000 as we will see this is more than enough for this data set and gave the best results in the analysis described above (complete assemblies can be generated with fewer than 10,000 reads from each starting file).

Next you will be prompted for an output folder;

	Select a F	Folder where you want	to save the split files	;		
		Split30000	٢			λ Search
Favorites	Name					Date Modified
Dropbox						
All My Files						
C iCloud Drive						
Desktop						
Users						
😭 kendall						
Applications						
Documents						
Movies						
\land Adobe AIR Applicati						
🎵 Music						
Dictures						
New Folder					Cancel	Choose

I created a new folder called Split30000 to save the files into.

Next you will be asked for a "Prefix" for the saved files for the first file (plasmid-1_S5_L001_R1_001.fastq);

What prefix should b kendall/Dropbox/Seq pRGN7782-plasmid- (Do not use spaces!)	e used for '/Volu uences/DataFo 1_S5_L001_R1_C :	umes/Data/ rRegeneron/)01.fastq'?
pRGN7782-R1-		
	Cancel	ОК

The full name is rather a mouthful so I compacted this to just "plasmid-R1-". The files will be generated with names adding "aaa", "aab", "aac" etc onto the Prefix, so we should get filenames like plasmid-R1-aaa.fastq, plasmid-R1-aab.fastq, plasmid-R1-aac.fastq etc.

When you click OK, your will see a "Job Running" dialog flash in and out of existence. Behind the scenes the file is being quietly and efficiently split. After a few seconds, you will get prompted for a prefix for the second file;

*	What prefix should b kendall/Dropbox/Sec pRGN7782-plasmid- (Do not use spaces!)	be used for '/Volu quences/DataForl ·1_S5_L001_R2_0):	mes/Data/ Regeneron/ 01.fastq'?
	pRGN7782-R2-		
		Cancel	ОК

I just named this ... R2- to distinguish it from the first.

When the second file has been split you will get a message that the file extensions are being fixed, followed by a message letting you know the processing is complete;



If you look in the output directory you should see all of the split files;

Users/kendall/Desktop/Split30000													
	:• A C *•	·	Q Search										
Favorites	Name	Date Modified	Size	Kind									
Stropbox	pRGN7782-R1-aaa	a.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
	pRGN7782-R1-aab	o.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
All Wiy Flies	pRGN7782-R1-aac	c.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
iCloud Drive	pRGN7782-R1-aac	d.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
AirDrop	pRGN7782-R1-aae	e.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
Dealstern	pRGN7782-R1-aaf	fastq Today, 12:56 PM	6.2 MB	FASTQuence									
L Desktop	pRGN7782-R1-aag	g.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
Users	pRGN7782-R1-aah	n.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
kendall	pRGN7782-R1-aai	.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
A	pRGN7782-R1-aaj	.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
	pRGN7782-R1-aak	c.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
🖺 Documents	pRGN7782-R1-aal	.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
Movies	pRGN7782-R1-aar	n.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
	pRGN7782-R1-aar	n.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
Adobe AIR Application	pRGN7782-R1-aac	o.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
🞵 Music	pRGN7782-R1-aap	o.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
- Pictures	pRGN7782-R1-aac	q.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
Fictures	pRGN7782-R1-aar	.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
Public	pRGN7782-R1-aas	s.fastq Today, 12:56 PM	6.2 MB	FASTQuence									
Preferences	pRGN7782-R1-aat	.fastq Today, 12:56 PM	6.2 MB	FASTQuence									

Assembling with Velvet

The next phase is to assemble the reads into contigs. We will do this in triplicate – we have lots of split files, so let's repeat the assembly with three sets of independent data and see if we get the same answer.

Choose File | New | Assembly Project. An empty Assembly Project window will open. Click on the Add Seqs toolbar button and navigate to the folder where you saved the split files;

AVORITES	Name	Date Modified	Size	Kind
Dronbox	pRGN7782-R1-aaa.fasi	tq Today 8:47 PM	2.1 MB	FASTnce File
	pRGN7782-R1-aab.fas	tq Today 8:47 PM	2.1 MB	FASTnce File
MV14	pRGN7782-R1-aac.fast	q Today 8:47 PM	2.1 MB	FASTnce File
📃 All My Files	pRGN7782-R1-aad.fas	tq Today 8:47 PM	2.1 MB	FASTnce File
Macintosh HD	pRGN7782-R1-aae.fast	tq Today 8:47 PM	2.1 MB	FASTnce File
Applications	pRGN7782-R1-aaf.fast	q Today 8:47 PM	2.1 MB	FASTnce File
Applications	pRGN7782-R1-aag.fas	tq Today 8:47 PM	2.1 MB	FASTnce File
Desktop	pRGN7782-R1-aah.fas	tq Today 8:47 PM	2.1 MB	FASTnce File
🚹 Documents	pRGN7782-R1-aai.fast	q Today 8:47 PM	2.1 MB	FASTnce File
EastOTests	pRGN7782-R1-aaj.fast	q Today 8:47 PM	2.1 MB	FASTnce File
Distribution	pRGN7782-R1-aak.fas	tq Today 8:47 PM	2.1 MB	FASTnce File
Distribution	pRGN7782-R1-aal.fast	q Today 8:47 PM	2.1 MB	FASTnce File
Songbooks	pRGN7782-R1-aam.fas	tq Today 8:47 PM	2.1 MB	FASTnce File
Ownloads	pRGN7782-R1-aan.fas	tq Today 8:47 PM	2.1 MB	FASTnce File
Movies	pRGN7782-R1-aao.fas	tq Today 8:47 PM	2.1 MB	FASTnce File
		T 1 0 13 001	2 1 1 12	F107 F1

Select the first three "R1" files - \dots R1-aaa, \dots R1-aab and \dots R1-aac and click on the Open button.

	📰 🔻 📄 Split10000	÷ Q	
FAVORITES	Name	Date Modified	Size Kind
📄 Dropbox	pRGN7782-R1-aff.fastq	Today 8:47 PM	2.1 MB FASTnce File
MV14	pRGN7782-R1-afg.fastq	Today 8:47 PM	2.1 MB FASTnce File
	pRGN7782-R1-afh.fastq	Today 8:47 PM	632 KB FASTnce File
All My Files	📄 pRGN7782-R2-aaa.fastq	Today 8:48 PM	2.1 MB FASTnce File
Macintosh HD	pRGN7782-R2-aab.fastq	Today 8:48 PM	2.1 MB FASTnce File
Applications	pRGN7782-R2-aac.fastq	Today 8:48 PM	2.1 MB FASTnce File
Desktop	🍺 pRGN7782-R2-aad.fastq	Today 8:48 PM	2.1 MB FASTnce File
	pRGN7782-R2-aae.fastq	Today 8:48 PM	2.1 MB FASTnce File
Documents	pRGN7782-R2-aaf.fastq	Today 8:48 PM	2.1 MB FASTnce File
FastQTests	pRGN7782-R2-aag.fastq	Today 8:48 PM	2.1 MB FASTnce File
Distribution	pRGN7782-R2-aah.fastq	Today 8:48 PM	2.1 MB FASTnce File
Songbooks	pRGN7782-R2-aai.fastq	Today 8:48 PM	2.1 MB FASTnce File
	pRGN7782-R2-aaj.fastq	Today 8:48 PM	2.1 MB FASTnce File
U Downloads	pRGN7782-R2-aak.fastq	Today 8:48 PM	2.1 MB FASTnce File
Movies	pRGN7782-R2-aal.fastq	Today 8:48 PM	2.1 MB FASTnce File
			Cancel Open

Repeat, but this time, choose the first three "R2" files;

Now select the pair of ...aaa.fastq files in the *Assembly Project* window (use the command key to toggle selections on and off) and click on the **Velvet** icon;

•						Untitle	d —	Project						
ې ن Add I	Reads Add Seqs	Add Ref	Add Contig	Remove	Ø Reset	Prefs Ph) red	CrossMatch	🛐 Phrap	Bowtie	SPAdes	O Velvet	Q Name Filter	
	Project	Proper	ties											
N	lame		Status			Length	~	#	ClipL		ClipR	Start	Stop	D
	pRGN7782-R1	l-aaa.fast	q Ilumina	Paired-	end		75	30000						/
	pRGN7782-R2	2-aaa.fast	q Ilumina	Paired-	end		75	30000)					/
	pRGN7782-R1	I-aab.fast	q Ilumina	Paired-	end		75	20000)					/
	pRGN7782-R1	l-aac.fast	q Ilumina	Paired-	end		75	20000	E.					/
	pRGN7782-R2	2-aab.fast	q Ilumina	Paired-	end		75	20000)					/
	pRGN7782-R2	2-aac.fast	q Ilumina	Paired-	end		75	20000)					/

A Velvet setup dialog appears;

Read pre-processing	Initial velveth Processing									
"Long" reads are at least 500 nt	Hash ("K-MER")									
Discard reads less than 33 nt	E1 (5 200)									
Trim ends with quality less than 20										
Trim N's from ends	\smile									
Discard short reads that contain any N's										
Source files contain paired reads										
Auto Short Read insert length: 400 Long Read in	sert length: 10,000									
Override automatic coverage defaults										
Coverage cutoff: 5 Auto Min. con	tig length: 500									
Expected coverage: 50 Auto Maximum	coverage: 500									
Advanced parameters										
Disable scaffolding Long Read merge cuto	off: 2 (0-20)									
Min. pair count: 5 (1-20) Max. branch leng	th: 100 nt									
Max. branch gaps: 3 (0-10) Max. branch divergen	ce: 0.2 (0.0-1.0)c									
Defaults	Cancel OK									

The most critical parameters in this are the **Hash** (K-MER") setting and the checkbox to let *Velvet* know you are using paired-end read files.

The K-MER value is generally the first parameter you should consider changing. It should have an odd value between 11 and 255. For *Illumina* reads like these, a good starting point is to aim at two-thirds of the read length. So, we will try 51.

The remainder of the parameters we will leave as the default. Click **OK**.

A progress dialog will appear. The entire assembly should take just a few seconds, then a new "*Contig 1*" entry appears in the *Assembly Project* window;

•	O Untitled — Project																	
Ad	d Reads	Add Seqs	Add Ref	Add Contig	Remove	Ø Reset	S= Prefs	Dhred	Cro	S Match	Phrap	Bowtie	SPAdes	() Velvet		Q	Name Filter	
	Projec	t 🦵	Propert	ies														
	Name			 Status 				Length		#		ClipL	ClipR		Start		Stop	
	Unus	edReads.	fa	Other					64		1000							/
	pRGN	7782-R1	-aaa.fasto	llumina	Paired-e	end			75	3	0000							/
	pRGN	7782-R1	-aab.fastc	llumina	Paired-e	end			75	2	0000							/
	pRGN	7782-R1	-aac.fasto	llumina	Paired-e	end			75	2	0000							/
	pRGN	7782-R2	-aaa.fasto	q Ilumina	Paired-e	nd			75	3	0000							/
	pRGN	7782-R2	-aab.fasto	q Ilumina	Paired-e	end			75	2	0000							/
	pRGN	7782-R2	-aac.fasto	q Ilumina	Paired-e	nd			75	2	0000							/
	Conti	g 1	>					8	909	5	8789		1	8909				

The fact that a single contig appears is a good thing. It is also in the size range we would expect for our plasmid. Double-click on *Contig 1* to open up the *Contig Editor* window;

00			— Editor			
Lacked Tout View				ACGT ACGT	⊕ ⇒	
Editor	Man		Annotations	Qualities Dots	Create	
Eultor	мар	reatures	Annotations	Summary		
Conse		9 20 AAGGGAAAAAGCAAGA	30 40	9 50 AGGAACTGCTGGG	60 70 GATCACAATTATGGAGGAGGAG	80 CAGCTTCGAAAA
16393	AAGGTGGAG	AGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAG	CA
17328	► AGGTGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAG	CAG
1452	► GGTGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAG	CAG
3769	► GGTGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAG	CAG
12060	► GGTGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAG	CAG
14474	► GGTGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAG	CAG
4073	► GTGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAG	CAG
4253	► GTGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAG	CAG
5114	► GTGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAG	CAG
15329	► GTGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAG
2405	GTGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAG
2939	▲ GTGGAG/	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAGC
6567	▲ GTGGAG/	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAGC
3453	TGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAGCTT
4913	TGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAGCTT
5334	TGGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAGCTT
10531	► GGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAG	CAGCTT
11389	GGAG	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCAAAATTATGGAGAGGAG	CATCTT
4207	AG/	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAGCTTCGA
6523	•	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAGCTTCGAAA
13096	•	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAGCTTCGAAA
19668	•	AAGGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAGCTTCGAAA
17476	•	GGGAAAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAGCTTCGAAA
2131	•	AAAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	ATCACAATTATGGAGAGGAG	CAGCTTCGAAA
9838	•	AAAGCAAGA	AACTGAAATCCGTC	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAG	CAGCTTCGAAA
505			ACTGAAATCCGTC	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAG	CAGCTTCGAAA

The **Editor** tab shows the aligned reads with the consensus across the top. Note that *Velvet* renames the reads, so unfortunately the names in the left hand margin do not correspond to names in the original fastq file. Unlike ABI contig assemblies, this assembly is NOT editable. The displayed alignment is for informational purposes only. You can click on the **Dots** toolbar item to get a feel for the quality of the alignment;

00			— Editor		
		0		ACGT	ф_
Locked Text View Pr	efs Replica	Translations W	/idth Basecalls	Qualities Dots	Create
Editor	Мар	Features	Annotations	Summary	
	10	20	30 46	50	<u> </u>
Consens	AAGGTGGAGA	AGGGAAAAAGCAAGA	AACTGAAATCCGTCA	AGGAACTGCTGGG	GATCACAATTATGGAGAGGAGCAGCTTCGAAA/
16393	▶				
17328	• • • • • • • • • • • • • • • • • • • •				
1452	• • • • • • • • • • • • • • • • • • • •				
3769	▶ ·····				
12060	▶ ·····				
14474	▶				
4073	▶ ·····				
4253	▶				
5114	• • • • • • • • • • • • • • • • • • • •				
15329	• • • • • • • • • • • • • • • • • • • •				
2405	• • • • • • • • • • • • • • • • • • • •			• • • • • • • • • • • • • • • •	
2939	• • • • • • • • • • • • • • • • • • • •				
6567	• • • • • • • • •				
3453	• • • • • • • • • • • • • • • • • • • •				
4913	• • • • • • • • • • • • • • • • • • • •				
5334	• • • • • • • • • • • • • • • • • • • •				
10531	▶ ·····				
11389	• · · · · · · · · · · · · · · · · · · ·				A
4207	• • • • •	••••••	••••••		
6523	•	••••••	••••••	•••••	
13096	-				
19008					
1/4/6					••••••••••••••••••••••••••••••••••••
2131	P				• • • • • • • • • • • • • • • • • • • •
9030					•••••••••••••••••••••••••••••••••••••••
505	•				••••••••••••••••

In this case, the alignment is pretty good, with just a couple of mismatched residues in one of the reads.

Perhaps more informative is the **Map** tab;



Here we can see a coverage map of the reads across the assembly. You can see that while the average is around 400x, there are some regions with over 1000x coverage and one region, centered around 2800, that has only about 30x coverage. The **Summary** tab has more information on the coverage;



One thing to note about the **Summary** output is that *Velvet* does not assign quality values to the consensus sequence (in contrast to *phrap*, which generates quality values for every base in the consensus). Thus the **Summary** will always report that all of the residues are of poor quality – this can be ignored.

Analyzing the Consensus Sequence

Circularizing Using a Right-Click Menu

In common with other *de novo* assemblers, *Velvet* will not automatically circularize sequences, so we have to do that manually. Luckily, MacVector

provides several functions to simplify this. By far the easiest to use is a new feature added in MacVector 16.0.1. In the **Editor** tab of the *Contig* window, right-click (<ctrl>-click if you don't have a right mouse button) and a popup context-sensitive menu will appear;



The menu item we are interested in here is the one at the bottom. If the contig has direct repeats at the ends, the item will be active and will have text similar to "**Circularize Consensus (50 nt overlap)**", where it will report the actual length of the overlap. This needs to be at least 15 nt. In the absence of direct repeats, the menu item will be disabled and have the text "**Cannot Circularize Consensus**". If you select the menu item, a new circular sequence document window appears.



Save this sequence as *plasmid-1.nucl*.

Circularizing Using a Dot-plot Analysis

The right-click menu approach works fine most of the time. But, if you do not have a perfect repeat at the ends of the contig, you may need to manually circularize the molecule. Here's how to do that so you can explore any similarities at the ends and potentially spot and fix sequencing errors that might prevent circularization.

First, to simplify the analysis, lets retrieve the consensus sequence from *Contig 1*. There are several ways to do this (e.g. you can **File | Export** as a single MacVector Nucleic Acid sequence), but perhaps the simplest is to use the right-click menu again and select the **Export Consensus Without Gaps**... item;



The next step is to circularize the molecule. But first we have to work out if there is an overlap between the beginning and the end of the sequence. One important thing to know about *Velvet* is that when it assembles a circular molecule, it tends to produce duplications at the beginning and end of approximately the value of K-MER. You may remember we used a value of 51 for this, so we are going to expect a duplication of approximately that value. This is just a rough guide – it is never longer than this, but frequently it is shorter, particularly with low coverage assemblies.

One quick way to look for direct repeats in a DNA molecule in MacVector is to use the **Dot Plot** functionality. With the new sequence window active, choose **Analyze | Create Dot Plot | Pustell DNA Matrix**;

Contig 1	Contig 1
Untitled 2	Untitled 2
X Region	Y Region
1 to 8909	1 to 8909
Options	
Set Scoring Matrix 隆 DNA	identity with penalties matrix.nmat 🛟
Window Size: 20	Hash Value: 5 💠
Minimum % Score: 95	Strand: Both 🗘
\smile	Jump: 1 🛟

Set the **Window Size** to 20 (or a value that is significantly less than the expected overlap between the ends) and **Minimum % Score** to 95%, as we are expecting essentially identical overlaps at the ends. Make sure you are comparing the new sequence (*Untitled 2* in this case) against itself. Click OK.



The **Matrix Plot** usually gives a good overview of the similarities between two sequences, or of direct repeats within a sequence. The long black diagonal line indicates the expected end-to-end identity when you compare a sequence to itself. In this case we are hoping to see direct repeats at the ends of the sequence – unfortunately, these are hard to see at the fully zoomed out resolution, though if

you look carefully you can see small "blips" at the circled corners. You can click and drag in the corners to "zoom" into those regions;



Once zoomed in. the diagonal representing the repeat is much easier to see. Even more clear, is to simply click on the **Aligned Sequence** tab;

● ● ●	Untitled 2 — Results
🛛 Matrix Plot 🛇 Alig	ned Sequence
Sequence: Untitled Ra	nge: 1 to 8909
Untitled	10 20 30 40 50 60 70 80 AAGGTGGAGA AGGGAAAAAG CAAGAAACTG AAACCGTCA AGGAACTGCT GGGGATCACA ATTATGGAGA GGAGCAGCTT TTCCAACCTCT TCCCTTTATCC ATTLAGGCAGT TTCCTTGACGA CCCTAGTGT TAATACCTCT CCTCGTGGAA
Untitled + Untitled	860 8870 8880 8890 8900 AAGGTGGAGA AGGGAAAAAG CAAGAAACTG AAATCCGTCA AGGAACTGCT> IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
Untitled + Untitled	10 20 30 40 50 60 70 80 AAGGTGGAGA AGGGAAAAAG CAAGAAACTG AAATCCGTCA AGGAACTGCT GGGGATCACA ATTATGGAGA GGAGCAGCTT> 11111111111111111111111111111111111
Untitled	90 100 110 120 130 140 150 160 CGAAAAGAAT CCTATCGATT TICTGGAGGC CAAAGGTAT AAGGAAGTGA AGAAAGACT GATCATCAAG CTGCCAAAGT GCTTTTCTTA GGATAGCTAA AGAACTCCCG GTTTCCCTAT TICCTTCACTG CTGATGTTC GACGGTTCA
Untitled + Untitled	90 100 110 120 130 140 150 160 CGAAAAGAAT CCTATCGATT TTCTGGAGGC CAAAGGGGTAT AAGGAAGGTGA AGAAAGACCT GATCATCAG CTGCCAAAGT> IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
Untitled	170 180 190 200 210 220 230 240 ACTCTCTGTT TGAGCTGGAA AACGGCAGAA AGCGGATGCT GGCAAGTGCC GGCGAGCTGC AAAAGGAAA TGAACTGGCC TGAGAGAGACAA ACTCGACCT TTGCCGTCAGC CCGTTCACGG CCGCTCGACG TTTTCCCTTT ACTTGGACCGG a/DATE="25-APR-2016"; /NOTE="1 TO 8909 OF CONTIG 1"a>
Untitled + Untitled	170 180 190 200 210 220 220 240 ACTCTCTGTT TGAGCTGGAA AACGGCAGAA AGCGGATGCT GGCAAGTGCC GGCGAGCTGC AAAAAGGAA TGAACTGGCC> IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
. lintitlad	250 260 270 280 290 300 310 320

The repeat from the end of the sequence is now clearly shown aligned to the beginning. We can see that it does look to be right about 50nt in length.

So, now we know that we do have the expected repeat, indicating that we have truly sequenced a circular sequence that is "wrapping around, we need to truncate the sequence to remove the duplicated segment. By far the easiest way to do this is to use MacVector's **Find** functionality. Close the *Dot Plot* result window and select the first ~20 nt in the sequence. If there was a mismatch in the alignment, you should make sure that residue is not included in the selection.

				I	Untitled	2 — Ed	tor				
DNA ÷	Unlocked	Text	View Pref	s Rep	lica To	pology	Blocking	Voice	1:20 Verify	(20 bp)	>>
Edite	or 🥤	1	Мар	Fe	eatures	Anr	otations				
+ AAGGTGGAG	A AGGGA	AAAAG	CAAGAAA	CTG AAA	тссбтса	AGGAACT	GCT GGG	GATCACA	ATTATGGAGA	80 GGAGCAGCTT	
CGAAAAGAA		CGATT	TTCTGGA	GGC CAA	AGGGTAT	AAGGAAG	TGA AGA	AAGACCT	GATCATCAAG	CTGCCAAAGT	
ACTCTCTGT	T TGAGC	TGGAA	AACGGCA	GAA AGC	GGATGCT	GGCAAGT	GCC GGC	GAGCTGC	AAAAAGGAAA	240 TGAACTGGCC	
CTGCCCTCA	A AGTAC	GTGAA	сттсстб	ГАТ СТБ	GCTAGCC	ACTACGA	GAA GCT	GAAAGGC	TCCCCTGAGG	320 ATAACGAACA	
GAAACAGCT	G TTTGT	GGAGC	AGCACAA	GCA TTA	TCTGGAC	GAGATCA	TTG AAC	AGATTAG	CGAGTTCTCC	400 AAACGCGTGA	
тсстббсто	A CGCAA/	ATCTG	GATAAGG	гсс тбт	СТБСАТА	CAACAAA	CAC AGG	GACAAGC	CAATCAGAGA	480 GCAGGCCGAA	
AATATCATT	C ATCTG	ТТСАС	TCTGACC	ΑΑС СТG	GGAGCCC	CCGCAGC	CTT CAA	GTATTTT	GACACTACCA	560 TCGATCGCAA	
ACGATACAC	A AGCAC	TAAGG	AGGTGCT	GGA TGC	ТАСССТС	ATCCACC	AGA GCA	TTACTGG	GCTGTACGAG	640 ACAAGGATCG	
ACCTGTCCC	A GCTGG	GGGGA	GACAAAC	GCC CAG	CCGCCAC	CAAGAAA	GCA GGA	CAGGCAA	AGAAGAAGAA	720 GTGAGTCGAC	
AATCAACCT	C TGGAT	ТАСАА	AATTTGT	GAA AGA	ттбастб	GTATTCT	таа ста	таттаст	CCTTTTACGC	800 TATGTGGATA	
CCCTCCTT	A ATCCC	TTTCT	ATCATCO		ттессет	ATCCCTT		тетесте	CTTCTATAAA	880	
COCTOCITI	A AIGUL	11101	ATCATGC	IAI IGC	TICCUGI	AIGGUII	ICA III	TETEETE	CITGIATAAA	960	

Choose Edit | Copy to copy the sequence, then choose Edit | Find | Find... Paste the copied sequence into the *Find* edit box;

Θ O Θ		Find	
		Feature Sequence	
Find:	AAGGTGGAGAAGC	GGAAAAAG	•
Replace:			•
	Replace All Replace	Strand: Both Wrap Around Frame:	¢
	Replace & Find	Find Previous Find Next Find	

Click on the **Find** button – it will simply re-select the first \sim 20nt in the sequence window. Then click **Find Next**. You should see that it finds a match close to the end of the sequence;



This is the start of the duplicated region at the end. To extend the selection to the end of the sequence, hold down the <shift> key and click just beyond the last character;



Now press the <delete> key to remove the duplication. Finally, click on the **Topology** button to circularize the sequence. then **File | Save** it under a suitable name (e.g. pRGN7782-1). We have the sequence of our plasmid!

Of course, if you had a mismatch in the repeat region, you would now need to confirm the sequence at the ends. We will look at some ideas on how to do that later on.

Repeating the Assembly

Lets double-check that the sequence is correct. We imported 3 pairs of files into our original *Assembly Project*. So we can repeat the assembly using the aab and aac pairs.

0		0					11-1-1	al a d	Duritant							
•		0					Unti	tied –	Project							
	÷Z	Jour	3	¥	Q.	Ø.	¥- ¥-	0	1	3		3		lame		
A	dd Seo	qs Ado	d Ref	Add Contig	Remove Seqs	Reset	Prefs	Phred	CrossMato	h Phrap	Bowtie	Velvet		Filter		
	Р	roject		Propertie	es											
	Nar	Name 🔺			Status		Lengt	h	#	0	ClipL	ClipR		Start	Stop	ĺ
	ι	Jnused	Read	ls.fa	Unknown for	mat		7	5	306						
	I	RGN7	782-	R1-aaa.fastq	FASTQ Seque	ence		7	5	9923						
	I	oRGN7	782-1	R1–aab.fastq	FASTQ Seque	ence		7	4	10000						
	F	RGN7	782-	R1-aac.fastq	FASTQ Seque	ence		7	5	9000						
	F	RGN7	782-	R2-aaa.fastq	FASTQ Seque	ence		7	5	9923						
H	F	oRGN7	782-1	R2-aab.fastq	FASTQ Seque	ence		7	5	10000						
	F	RGN7	782-	R2-aac.fastq	FASTQ Seque	ence		7	5	10000						
Æ	. (Contig	1					890	9	19459		1	8909			

Select the aab pair of files and click the Velvet button;

You will get a warning dialog about contigs already having been created;

Z	Delete existing contigs?.
6	The Project contains contigs from a previous assembly. Should these be deleted before running Velvet? It is recommended that you do delete them to avoid confusion. If you are trying compare different assembly parameters, choose Cancel and create a conv of the project first.
(No Cancel Yes

In this case, we know we are assembling a different pair of files, so we can safely click the **No** button as we don't want to delete *Contig 1*.

After assembly, we see a new *Contig 2* item.

•	0	0				Unti	itled –	- Project						
Ad	d Sec	s Add R	ef Add Contig	Remove Seqs	Reset	Ğ= ₽refs	کی Phred	CrossMatc	h Phrap	Bowtie	Velvet	Q* 1	Name Filter	
	Pr	oject	Properti	es										
	Nan	ne		Status	_	Lengt	:h	#	1	ClipL	ClipR		Start	Stop
	ι	InusedRe	ads.fa	Unknown for	mat		7	5	306					
	ι	InusedRe	ads.fa	Unknown for	mat		7	5	253					
	p	RGN778	2-R1-aaa.fastq	FASTQ Seque	nce		7	5	9923					
	, p	RGN778	2-R1-aab.fastq	FASTQ Seque	nce		7	5	9894					
	p	RGN778	2-R1-aac.fastq	FASTQ Seque	nce		7	5	9000					
	p	RGN778	2-R2-aaa.fastq	FASTQ Seque	nce		7	5	9923					
	, p	RGN778	2-R2-aab.fastq	FASTQ Seque	nce		7	5	9894					
	p	RGN778	2-R2-aac.fastq	FASTQ Seque	nce		7	5	10000					
-TE	C	ontig 1					890	9	19459		1	8909		
GE	0	Contig 2	>				890	9	19502		1	8909		

Repeat the previous steps to circularize the sequence and save it as pRGN7782-2.

Assembly using SPAdes

For the third set of files we will try using the *SPAdes* assembler. Select the pair of *...aac.fastq* files and click on the **SPAdes** toolbar button.

Note that we are going to use a custom K-MER – type the values 31,41,51,61,71 into the box. The values must be odd and must be separated by commas. You must also provide at least 3 values. You can add odd values up to 127.

Unlike *Velvet*, *SPAdes* does not create alignments, it merely generates a list of consensus sequences. If you are interested in looking at the actual alignments, then MacVector offers the option of running a *Bowtie* alignment on each of the consensus sequences. This is very useful for visualization, so we will select that option too.

Read pre-processing												
Discard reads less than	33 nt											
Trim ends with quality less than 20												
Trim N's from ends												
Discard short reads that contain any N's												
SPAdes Options												
Override coverage cutoff:	5											
Use custom K-MER:	31,41,51,61,71											
Threads: 7	Enter odd values less than 128 in ascending order, separated by commas (e.g. 21,33,55).											
Generate Alignments Using Bow	tie											
Threads: 7												
Defaults	Cancel OK											

With such a small data set, *SPAdes* will still complete in just a few seconds. However, with larger genomic data sets, this can take several hours.

•	•					Unti	tled — F	Project					
1		23	ŵ.	Ø.	Ø.	¥- *-	() -	ġ.	3		্ৰ 🍠	Q Na	me
Ad	d Reads Add Seqs	Add Ref	Add Contig	Remove	Reset	Prefs	Phred C	rossMatch	Phrap	Bowtie SP	Ades Velvet	Fil	er
	Project	Proper	ties										
	Name				^	Status		Lengt	h	#	ClipL	ClipR	Start
	Unaligned_Rea	ads_3.fq.g	JZ			llumina	Unpaired	ł	48	240			
	UnusedReads	.fa				Other			64	1000			
	UnusedReads	.fa				Other			64	1000			
	pRGN7782-R1	l-aaa.fast	q			Ilumina	Paired-e		75	30000			
	pRGN7782-R1	l-aab.fast	q			Ilumina	Paired-e		75	20000			
	pRGN7782-R1	l-aac.fast	q			Ilumina	Paired-e		75	20000			
	pRGN7782-R2	2-aaa.fast	q			Ilumina	Paired-e		75	30000			
	pRGN7782-R2	2-aab.fast	q			Ilumina	Paired-e		75	20000			
	pRGN7782-R2	2-aac.fast	:q			Ilumina	Paired-e		75	20000			
-	Contig 1								8909	58789		1 8909)
-	Contig 2								8909	58635		1 8909)
<	NODE_1_lengt	h_8930_c	ov_29.91692	21 Contig	3				8930	59758		1 8930	
-	NODE_2_lengt	th_72_cov	4.000000	Contig 3					/2	2		1 72	2

Note that *SPAdes* uses a different naming convention for the contigs, calling them "*NODE_x_length_<len>_cov_<coverage>*" which MacVector honors in the display. Again, you can double-click on the longest contig to open up the *Contig Editor* window. Note that because *Bowtie* retains the read names, you can see the original read names in the assembly.

			1 🗋	NOC	E_1_le	ngth_8	8930_0	cov_2	29.916	6921	Conti	g 3 —	Edito	r					
	3-	- -	0 ~				TGCA	i hu		ACGT	÷.								
Locked Text View	Profe	Paplica	Conception Translation		Width	Ba		Quali	ities	ALGI	Creat	0							
LOCKED TEXT VIEW	Field	(epilea	(15	width	Da	isecalis	Quan	lies	DOUS	Great								
Editor	N	lap	Fea	tures	s	Anno	otations	5	Su	mmar	у								
						10		20		10	4	IA	50		60		70	80	96
			Consen	sus	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	GCCCA	GTAATO	SCTCTG	GTGGAT	CAGGGT	AGCATO	CAGCACC	TCCTTAGT	GCTTGTG
M03818:33:0000000	00-AL4P6:1	:1103:1555	5:10943		CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	GCCCA	GTAAT	SCTCTG	GTGGAT	CAGGGT	AGCATO	CAGCAC		
M03818:33:0000000	00-AL4P6:1	:1102:8384	:20551		CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	\GCCCA	GTAATO	SCTCTG	GTGGAT	CAGGGT	AGCATO	CAGCAC		-
M03818:33:0000000	00-AL4P6:1	:1103:4888	3:4334	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	AGCCCA	GTAATO	SCTCTG	GTGGAT	CAGGGT	AGCATO	CAGCAC		
M03818:33:0000000	00-AL4P6:1	:1103:1722	7:12514		CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	AGCCCA	GTAATO	SCTCTG	GTGGAT	CAGGGT	AGCATO	CAGCAC		
M03818:33:0000000	00-AL4P6:1	1:1102:7940	:18889	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	AGCCCA	GTAAT	SCTCTG	GTGGAT	CAGGGT	AGCATO	CAGCACC		
M03818:33:0000000	00-AL4P6:1	:1102:2223	3:19306	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	GCCCA	GTAATO	SCTCTG	GTGGAT	CAGGGT	AGCATO	CAGCAC		
M03818:33:0000000	00-AL4P6:1	:1102:2704	0:19809	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	GCCCA	GTAATO	SCTCTG	GTGGAT	CAGGGT	AGCATO	CAGCAC		
M03818:33:0000000	00-AL4P6:1	1:1102:7473	3:21945	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	AGCCCA	GTAATO	SCTCTG	GTGGAT	CAGGGT	AGCATO	CAGCAC		
M03818:33:0000000	00-AL4P6:1	1:1102:1904	6:22641	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	AGCCCA	GTAAT	SCTCTG	GTGGAT	CAGGGT	AGCATO	CAGCAC		
M03818:33:0000000	00-AL4P6:1	1:1102:1506	3:23207	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	GCCCA	GTAATO	SCTCTG	GTGGAT	CAGGGT	AGCATO	CAGCAC		
M03818:33:0000000	00-AL4P6:1	1:1102:1120	5:23300	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	GCCCA	GTAA							
M03818:33:0000000	00-AL4P6:1	:1102:1714	8:24222	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	AGCCCA	GTAATO	стсто	GTGGAT	CAGGGT	AGCATO	CAGCAC		
M03818:33:0000000	00-AL4P6:1	1:1103:1124	3:4724	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	AGCCCA	GTAAT	бстстб	GTGGAT	CAGGGT	AGCATO	C		
M03818:33:0000000	00-AL4P6:1	1:1103:1642	2:6007	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	GCCCA	GTAA							
M03818:33:0000000	00-AL4P6:1	:1103:6820	:6554	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	GCCCA	GTAATO	SCTCTG	STGGAT	CAGGGT	AGCATO	CAGCAC		
M03818:33:0000000	00-AL4P6:1	1:1103:2490	3:10753	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTACA	GCCCA	GTAATO	бстстб	GTGGAT	CAGGGT	AGCATO	CAGCAC		
M03818:33:0000000	00-AL4P6:1	1:1102:1076	0:17174	•	CCCAGC	TGGGAC	AGGTCG	АТССТІ	төтсто	GTACA	GCCCA	GTAATO	бстстб	GTGGAT	CAGGGT	AGCATO	CAGCAC		
M03818:33:0000000	00-AL4P6:1	1:1102:1284	9:18656	•	CCCAGC	TGGGAC	AGGTCG	АТССТІ	төтсто	GTACA	GCCCA	GTAAT	бстстб	GTGGAT	CAGGGT	AGCATO	CAGCACC		
M03818:33:0000000	00-AL4P6:1	:1102:1280	5:20000	•	CCCAGC	TGGGAC	AGGTCG	АТССТІ	төтсто	GTACA	GCCCA	GTAAT	бстстб	GTGGAT	CAGGGT	AGCATO	CAGC		
M03818:33:0000000	00-AL4P6:1	:1102:1459	6:22106	•	CCCAGC	TGGGAC	AGGTCG	ATCCT	төтсто	GTAC	GCCCA	GTAATO	стсто	STGGAT	CAGGGT	AGCATO	CAGCACC		
M03818:33:0000000	00-AL4P6:1	:1102:1350	2:24800	•	CCCAGC.	TGGGAC	ACCTCC	АТССТ	татете	GTACA		GTAATO	стста	TAGAT	CAGGGT	AGCATO	- CAGCACC		
1100010 00 0000000			0.400	4															

Again, you can right-click in the **Editor** tab to see if the contig can be circularized. In this case we have a 71 nt overlap. Create the new circular sequence and save it as pRGN7782-3.

Confirming the Correct Sequence using Three Contigs

So, now we have our three circular contig sequences, each generated from a different set of input reads. However, before we can compare them to make sure they are identical, there are two additional factors to consider;

- (a) **Orientation** contigs can be created in one of two orientations. We need to make sure that all three are in the same orientation before we can do the comparison.
- (b) **Circular Origin**. the assemblers essentially choose a random location for the start of the assembly. Thus, when we circularize, the contigs are likely to be "split" in a different location. We need a way to ensure they are all split at the same point.

Determining Orientation

The easiest way to determine orientation is to use MacVector's dotplot analysis function. Close all open windows except for the three circular plasmids, *plasmid-1*, -2 and -3. Choose **Analyze | Create Dot Plot | Pustell DNA Matrix**, select *plasmid-1* in the left column and *plasmid-2* in the right. Set the **Minimum % Score** to 95;

pRGN7782-1.nucl pRGN7782-2.nucl pRGN7782-2.nucl pRGN7782-2.nucl pRGN7782-3.nucl PRGN7782-3.nucl X Region Y Region 1 to 8859 1 to 8859
pRGN7782-2.nucl pRGN7782-2.nucl pRGN7782-3.nucl pRGN7782-3.nucl X Region Y Region 1 to 8859 1 to 8859
pRGN7782-3.nucl pRGN7782-3.nucl X Region Y Region 1 to 8859 1 to 8859 🗊
X Region Y Region 1 to 8859 😨 1 to 8859 😨
X Region Y Region 1 to 8859 C 1 to 8859 C
X Region Y Region 1 to 8859 C 1 to 8859 C
X Region Y Region 1 to 8859 (2) 1 to 8859 (2)
X Region Y Region 1 to 8859 😨 1 to 8859 😨
X Region Y Region 1 to 8859 🔅 1 to 8859 🔅
1 to 8859 🔅 1 to 8859 🤅
1 to 8859 💭 1 to 8859
Options
Set Scoring Matrix M DNA database matrix.nmat
Window Size: 30 Hash Value: 8 🗘
Minimum % Score: 95 Strand: Both 🗘
Jump: T
Defaults Cancel OK

Run the analysis and take a look at the Matrix Plot result tab;



In this plot (your results may vary) there are two diagonal lines of identity. Because the plots are blue and travel from lower left to upper right, this indicates that this is an inverted match i.e. the two contigs are in different orientations. The fact that there are two lines, indicates that they have different circular origins.

Repeat the dot plot with the other two combinations -1 vs 3 and 2 vs 3;



In this case, 1 vs 3 (left) indicates that -1 and -3 are also in the opposite orientation, but 2 vs 3 (right) are in the same orientation as indicated by the black plots travelling from top left to bottom right.

The easiest thing to do is "flip" the orientation of *plasmid-1* so that it matches the other two. To do this, simply bring that plasmid to the front, use **Edit | Select All** to select the entire sequence, then choose **Edit | Reverse & Complement** to "flip the sequence. Don't forget to save it!

Changing Circular Origin

If we look at the **Map** tab of *pRGN7782-1*, the unique restriction enzymes are easily identifiable as they appear in red;



In this case, there is a unique *EcoRI* site at (6286). Let's make this the new circular origin. Click on the site to select it, then right-click (or <ctrl>-click) to view the context-sensitive menu;



Choose the **Set Circular Origin** item and the **Map** refreshes to show the plasmid rotated with the *EcoRI* site now at the 12 o'clock position.



Save the file and repeat with the *EcoRI* sites in *pRGN7782-2* and *pRGN7782-3*.

Confirm Identity using Multiple Sequence Alignment

With the three rotated plasmid sequences open, choose Analyze | Align Multiple Sequences Using | ClustalW;

Pairwise Alignment	Multiple Alignment
Alignment Speed: Slow 🗘 Open Gap Penalty: 15 Extend Gap Penalty: 6.66	Open Gap Penalty:15Extend Gap Penalty:6.66Delay Divergent:30%Transitions:Weighted
Sequences To Align pRGN7782-1.nucl pRGN7782-2.nucl pRGN7782-3.nucl	Defaults Cancel OK

Make sure you have all three plasmids selected, then click **OK**. After the alignment has completed you should have a window like this;



You can scroll through the alignment if you wish, looking for any mismatches, but the most useful tabs from this perspective are the **Pairwise** tab and the **Matrix** tab. The **Pairwise** tab shows each sequence aligned with each other sequence. The header for each alignment lists the gaps and identities between the pair of sequences;

```
1. pRGN7782-1 vs. pRGN7782-2
   Aligned Length = 8859
                        Gaps = 0
   Identities = 8859 (100%)
pRGN7782-1
              1 AATTCAAAAAAAGCACCGACTCGGTGCCACTTTTTCAAGTTGATAACGGACTAGCCTTAT
                                                                         60
pRGN7782-2
              1 AATTCAAAAAAGCACCGACTCGGTGCCACTTTTTCAAGTTGATAACGGACTAGCCTTAT
                                                                         60
                pRGN7782-1
             61 TTTAACTTGCTATTTCTAGCTCTAAAACCCCTGTCTGCAACTAGCTCAGTGGTCTCATAC
                                                                        120
pRGN7782-2
             61 TTTAACTTGCTATTTCTAGCTCTAAAACCCCTGTCTGCAACTAGCTCAGTGGTCTCATAC
                                                                        120
nRGN7782-1
            121 AGAACTTATAAGATTCCCAAATCCAAAGACATTTCACGTTTATGGTGATTTCCCAGAACA
                                                                       180
Here it's clear that plasmid-1 and plasmid-2 are identical with no gaps.
```

The Matrix tab simple summarizes the combinations of pairwise alignments;

ClustalW multiple sequence alignment

```
3 Sequences Aligned
                                Processing time: 9.3 seconds
Gaps Inserted = 0
                                Conserved Identities = 8859
Score = 53045
Pairwise Alignment Mode: Slow
Pairwise Alignment Parameters:
    Open Gap Penalty = 15.0
                               Extend Gap Penalty = 6.7
Multiple Alignment Parameters:
    Open Gap Penalty = 15.0 Extend Gap Penalty = 6.7
Delay Divergent = 30% Transitions: Weighted
           ** Identity Scores (%) **
    pRGN778 pRGN778 pR
                                  pRGN778
                                   2-3
                         2-2
               2 - 1
pRGN7782-1
                          100.0
                100.0
                                    100.0
pRGN7782-2
                          100.0
                100.0
                                    100.0
pRGN7782-3
                100.0
                          100.0
                                    100.0
             ** Similarity Scores (%) **
**Similarity Scores(s) are shown below the diagonal (x) with Identity Scores(I) above**
  abcde
axiiii
bsxiii
cssxii
dsssxi
essssx
```

Here we can clearly see that all three plasmids share 100% identity with each other. With this confirmation of the sequence from three different sets of reads, we can be extremely confident that we have accurately determined the sequence of our plasmid.