IVDP - Integrated Variant Discovery Pipeline

User manual and guide

IVDP was designed for calling variants - SNPs (single-nucleotide polymorphisms) and Indels (insertions and deletions) - from Whole Genome Sequence (WGS) and RNA Sequence (RNAseq) data, as well gene counts of RNAseq. IVDP combines 25 bioinformatic softwares to automatically create annotated VCF files from fastq files. The following softwares must be installed before using IVDP:

Download IVDP

```
git clone https://github.com/rodrigopsav/IVDP.git
```

Download Examples

Please, download the examples folder from this link and save them in IVDP folder.

Install IVDP dependencies

IVDP uses a set of bioinformatic tools that are combined in a way to automatically output the annotated VCF files of any number of samples. The bioinformatic programs are managed using conda environments. Before run IVDP, you must install the dependencies (install_ivdp_dependencies.sh file within install_ivdp_dependencies folder) even if you have already installed the programs in your machine. The install_ivdp_dependencies.sh will install all the programs in specific conda environments created during the installation. To install IVDP dependencies, run:

```
cd install_ivdp_dependencies
./install_ivdp_dependencies.sh -f path/to/install/dependencies/programs
```

The bash script uses conda functions to install the most recent versions of the programs. Unfortunately, errors can occur when install the most recent version of the programs. If you detect some errors after install IVDP dependencies, use the following command lines to remove the previous installation and to install the version of the programs that were used originally to develop IVDP:

```
conda env remove --name ivdp
conda env remove --name ivdp2
./install_ivdp_dependencies_versions.sh -f path/to/install/dependencies/programs
```

Please visit the official websites of the programs used on IVDP to learn more about them: BBMap BCFtools BEDtools Bowtie Bowtie2 BWA-MEM BWA-MEM2 FastQC featureCounts Freebayes Freebayes-parallel GATK GSNAP HISAT2 HTSeq HTSlib MultiQC Picard Platypus Sambamba Samtools snpEff and snpSift STAR Trimmomatic VCFtools Zlib

All these programs are installed with install_ivdp_dependencies.sh file. You can use them in any routine without IVDP. To access the ivdp conda environments, use:

```
conda activate ivdp
conda activate ivdp2
```

Only freebayes and platypus are installed in ivdp2 environment. The other programs are in ivdp environment. It is only possible to use one conda environment at time. To deactivate the environments, use:

```
conda deactivate
```

Run IVDP

To run IVDP, use the command line:

```
./ivdpRun.sh -p parFile.txt -c chromNames.txt
```

-p: parameter file -c: chromosome names file

There is an example of parameter file for single-end (parSE.txt) and paired-end (parPE.txt) reads. The parameter file is mandatory to run IVDP but the chromosome names file is not. If the -c flag is omitted, IVDP will used all the chromosomes including DNA that has not been assigned a chromosome. Both parameter and chromosome name files can be renamed as you wish.

NOTE: Do not use nohup and & to run IVDP in the background. After ivdpRun.sh starts, all the processes will be automatically sent to the backgroud.

Log in the screen

To check the main log file on your screen, run the command line:

```
./ivdpCheckLog.sh -p parFile.txt
```

To skip it anytime, press Ctrl+C.

Chromosome names file (chromNames.txt)

Each chromosome name must be written in each line as bellow:

```
$ cat chromNames.txt
1
2
15
26
Y
X
MT
```

Parameter file (parFile.txt)

The parameter file contains variables that you have to assign options to them:

########## Input data section ##########

- **INPUT_DIR**: Directory path for the original fastq files.
- **OUTPUT_DIR**: Directory path for the outputs of the analysis (Default is the home directory if left blank: OUTPUT_DIR=).
- **OUTPUT_NAME**: Name for analysis. Choose a single name without spaces (Default is user name if left blank: OUTPUT_NAME=).
- **LIST_SAMPLES**: File directory for a list with samples id and individual id, separated by space. Use LIST_SAMPLES=none if no list is available.
- **ANALYSIS**: Choose between Whole Genome Sequence (**ANALYSIS=1**) or RNAseq data (**ANALYSIS=2**).
- **EXTENSION_SE**: For single-end fastq files, write the extension of original fastq files. If paired-end files, left blank (EXTENSION_SE=).
- **EXTENSION_PE1**: For paired-end fastq files, write the extension of PE1 fastq files. If single-end files, left blank (EXTENSION_PE1=).
- **EXTENSION_PE2**: For paired-end fastq files, write the extension of PE2 fastq files. If single-end files, left blank (EXTENSION_PE2=).

########## Input genome section ##########

- **ASSEMBLY**: Assing a name for the reference genome. Eg. ASSEMBLY=hg38.
- **REFERENCE**: File directory for reference genome **(*.fa)**. The reference genome must be **uncompressed file** and not **(*.fa.gz)**.
- **VARIANTS**: File directory for variants file **(*.vcf.gz)** for the reference genome. Use VARIANTS=none if no variants is available.The variants must be **(*.vcf.gz)** and not **(*.vcf)**.
- **ANNOTATION**: File directory for annotation file **(*.gtf)** for the reference genome. Use ANNOTATION=none if no annotation is available. The annotation must be **uncompressed file** and not **(*.gtf.gz)**.

########## Workflow steps section ##########

- **STEP_QC_TRIM** Activate quality control and adapter trimming step? STEP_QC_TRIM=yes or STEP_QC_TRIM=no (Default yes).
- **STEP_ALIGNMENT**: Activate alignment step? STEP_ALIGNMENT=yes or STEP_ALIGNMENT=no (Default yes).
- **STEP_GENECOUNT**: # Activate gene count step? STEP_GENECOUNT=yes or STEP_GENECOUNT=no (Default: yes for RNAseq and no for WGS).
- **STEP_MDUP**: Activate mark duplicated read step? STEP_MDUP=yes or STEP_MDUP=no (Default: yes). It includes splitncigar step for RNAseq.
- **STEP_BQSR**: Activate base quality score recalibration step? STEP_BQSR=yes or STEP_BQSR=no (Default: yes).
- **STEP_VCF_CALL**: Activate variant calling step? STEP_VCF_CALL=yes or STEP_VCF_CALL=no (Default: yes).
- **STEP_VCF_FILTER**: Activate vcf filtering step? STEP_VCF_FILTER=yes or STEP_VCF_FILTER=no (Default: yes).
- **STEP_VCF_COMBINE**: Activate combine filtered snps from different vcfs? STEP_VCF_COMBINE=yes or STEP_VCF_COMBINE=no (Default: yes).
- **STEP_VCF_ANNOTATE**: Activate annotation of vcf files? STEP_VCF_ANNOTATE=yes or STEP_VCF_ANNOTATE=no (Default: yes).

########## Choose the programs ##########

- **ALIGNER_PROGRAM**: Choose programs for alignment step. If more than one program are chosen, use ',' between them (Options: bbmap, bowtie, bowtie2, bwa, bwa2, gsnap, hisat2, star). If ALIGNER_PROGRAM is empty, the default is ALIGNER_PROGRAM=bwa2 if ANALYSIS=1 and ALIGNER_PROGRAM=hisat2 if ANALYSIS=2.
- **CALLER_PROGRAM**: Choose programs for variant calling step. If more than one program are chosen, use ',' between them (Options: bcftools, freebayes, freebayes-parallel, gatk4, gatk4GenomicsDBImport, gatk4CombineGVCFs, platypus, varscan). If CALLER_PROGRAM is empty, the default is CALLER_PROGRAM=gatk4GenomicsDBImport if ANALYSIS=1 and CALLER_PROGRAM=freebayes-parallel if ANALYSIS=2.

> **NOTE**: freebayes and freebayes-parallel are the same program, but the second one is much more faster. The gatk4 option calls the variants using GATK HaplotypeCaller from a multi-sample the joint calling approach. The gatk4GenomicsDBImport and gatk4CombineGVCFs options use GATK HaplotypeCaller with gVCF MODE, i.e. a gVCF file will be created for each individual and after that merged into a single one using GATK GenomicsDBImport or GATK CombineGVCFs. Using any of these two options, the VCF file will be created from the gVCFs. Both options do the same job.

> So, if you want to use the gVCF approach fo your analysis, choose either gatk4GenomicsDBImport or gatk4CombineGVCFs. Do not use both at the same time. Generally, gatk4GenomicsDBImport is faster than gatk4CombineGVCFs and the last one is just used when the first one does not work.

- **GENECOUNT_PROGRAM**: Choose programs for gene count step. If more than one program are chosen, use ',' between them (Options: htseq, featurecounts). Use GENECOUNT_PROGRAM=none if you do not want gene counts.
- **FEATURE_TYPE**: Any feature of the third column of gtf file. (Default FEATURE_TYPE=exon). Use FEATURE_TYPE=none if you do not want gene counts. To check the possible features on third column of the gtf file, type on linux terminal `$ grep -v "#" name_of_file.gtf | awk '{print $3}' | sort | uniq`

########## General parameters section ##########

- **CALL_BY_CHROM**: Should variant calling be done by chromosome? CALL_BY_CHROM=no or CALL_BY_CHROM=yes (Default no).
- **MIN_READ_LENGTH**: Minimum read length to be kept after adapter trimming (Default 36).
- **MAX_READ_LENGTH**: Maximum read length to be kept after adapter trimming (Default 150).
- **DOWN_SAMPLING_FASTQ**: Down sampling the fastq files. (Default 0). DOWN_SAMPLING_FASTQ=0 means no reads sampling from fastq files. Any number between 0.01 and 0.99 means to sampling the fraction of reads from fastq files. Any number equal or greater than 1 means to sampling the number of reads from fastq file.

  > **NOTE:** Down-sampling reads can save computational resources and analysis time when the fastq files are extremely large. For example, it is possible to down-sampling the reads to decrease the coverage from 10x to 5x to call the variants. The relation between the number of reads, average read length and coverage can be calculated using:

  C = (N*L)/G, which C=coverage, N=number of reads, L=average read length (in base pairs), and G=size of reference genome (in base pairs). **Multiply this formula by 2 if fastq files are paired-end**.

  > For example, considering a paired-end fastq file (so, multiply the formula by 2) with 150 million of reads in each file, average read length of 100bp, and the human reference genome (which contains approximately 3 billion of base pairs), the coverage will be approximately:

  C = (150,000,000 * 100 * 2) / 3,000,000,000 = 10x

  > To down-sampling the fastq file from 10x to 5x of coverage:

  5 = (N * 100 * 2) /3,000,000,000 --> N = 75 million of reads.

  > So, in this case, **DOWN_SAMPLING_FASTQ=75000000** in the parameter file will down-sampling the fastq files from 10x to 5x of coverage. Before start the analysis, you can get the number of reads and average read length of your fastq files. Run this function on linux terminal:

```
fix_base_count() {
  local counts=($(cat))
  echo "${counts[0]} $((${counts[1]} - ${counts[0]}))"
}
```

  > After that, use the command line to count the reads and get the average read length of you fastq file. Just change *NAME_OF_FASTQ_FILE* to the actual name of your fastq file:

```
pigz -fdc *NAME_OF_FASTQ_FILE* \
| awk 'BEGIN { t=0.0;sq=0.0; n=0;} ;NR%4==2 {n++;L=length($0);t+=L;sq+=L\*L;} \
END{m=t/n;printf("total=%d avg=%f stddev=%f\n",n,m,sq/n-m*m);}'
```

- **MIN_DEPTH**: Minimum locus coverage for the filtering step (Default 3).
- **COMBINE_VCF**: Choose COMBINE_VCF=none, COMBINE_VCF=partial or COMBINE_VCF=full.

  > ***COMBINE_VCF=none:*** do not combine different vcfs; ***COMBINE_VCF=partial:*** If it is used more than one program for variant calling, combine common SNPs that appeared AT LEAST in two different vcf files. ***COMBINE_VCF=full:*** If it is used more than one program for variant calling, combine common SNPs that appeared IN ALL the vcf files.
  >
  > ***NOTE:*** IVDP allows to combine SNPs from different VCF files in two ways. Let's suppose you chose three variant callers (i.e. bcftools, freebayes and gatk4). In the end, IVDP will output one VCF file from each variant caller. If you choose **COMBINE_VCF=partial**, IVDP will combine these three VCFs in a single one selecting the SNPs that appeared at least on two different VCF files. If you choose **COMBINE_VCF=full**, IVDP will combine these three VCFs in a single one selecting only the SNPs that appeared in all the VCF files. Now let's suppose you used two aligners (let's say bowtie2 and bwa) and three variant callers (i.e. bcftools, freebayes and gatk4). In the end, IVDP will output six VCF files (bowtie2_bcftools, bowtie2_freebayes, bowtie2_gatk4, bwa_bcftools, bwa_freebayes, and bwa_gatk4). No matter if you chose **COMBINE_VCF=partial** or **COMBINE_VCF=full**, IVDP will combine the VCFs within aliners. In this case, IVDP will output two VCF files: one VCF file from the combination among bcftools, freebayes and gatk4 from **bowtie2** and another VCF file from the combination among bcftools, freebayes and gatk4 from **bwa**. If left blank, the default is **COMBINE_VCF=partial**.

- **THREADS**: Number of threads for analysis.
- **BATCH**: Number of samples to be processed at same time.
- **KEEP_LOG**: Keep the log files in the output folder? KEEP_LOG=no or KEEP_LOG=yes (Default no)
- **KEEP_INTERMEDIATE_DATA**: Keep intermediate fastq and bam files in the output folder? KEEP_INTERMEDIATE_DATA=no or KEEP_INTERMEDIATE_DATA=yes (Default no)

IVDP Output

IVDP outputs a series of files, statistics and reports from the bioinformatic programs used on it. You will find five subfolders in the output folder: chromosomes, data, log, report, and stats. To access these folders and files in a more organized way, a html file is created in the output folder. The html output file looks like that.

The first lines in orange show a brief description of the analysis. The information about the chromosomes and the list with samples used for the analysis can be accessed by clicking on the respective links. The main table contain three columns:

- **FIRST COLUMN: DATA FILES**
    - QUALITY CONTROL OF FASTQ FILES
    - *TRIMMED FASTQ FILES*:* contain the fastq.gz files after adapter removal and quality control by trimmomatic.
    - ALIGNMENT AND GENE COUNTS
    - *ALIGNED BAM FILES*:* contain the bam files with defined read groups (RG).
    - *MDUP AND BQSR BAM FILES*:* contain the bam files with mark duplicated reads and base quality score recalibration.
    - *GENE COUNTS**:* contain the files for gene counts.
    - VARIANTS
    - *ORIGINAL VCF FILES:* contain original unfiltered vcf.gz files with all the samples.
    - *FILTERED SNP VCF FILES**:* contain only filtered SNPs vcf.gz files with all the samples.
    - *FILTERED SNP INDEL FILES**:* contain only filtered indels vcf.gz files with all the samples.
    - *COMBINED SNP VCF FILES**:* contain the combined SNPs from different filtered vcf files (if required in the parameter file - **COMBINE_VCF=partial** or **COMBINE_VCF=full** - and if it was used more than one variant calling program).
    - *ANNOTATED SNP VCF FILES**:* contain the annotated VCF files with snpEff and snpSift.

  NOTE: *These paths are empty if the intermediate files were removed (***KEEP_INTERMEDIATE_DATA***=no in parameter file). **These paths are empty if their steps were deactivated in the parameter file (STEP_GENECOUNT=no / STEP_VCF_FILTER=no / STEP_VCF_COMBINE=no / STEP_VCF_ANNOTATE=no). If it was chosen ANALYSIS=1 the gene counts will be skipped and if it was chosen only one variant calling program, the combined vcf files step will be skipped too.

- **SECOND COLUMN: REPORTS** All the reports are generated with MultiQC.
    - QUALITY CONTROL OF FASTQ FILES
    - *QC AND ADAPTER TRIMMING:* reports for original fastqc files, trimmed fastqc files and trimmomatics report.
    - ALIGNMENT AND GENE COUNTS
    - *ALIGNMENT:* reports for the alignment steps.
    - *GENE COUNTS:* reports for the gene counts.
    - VARIANTS
    - *ORIGINAL VCF FILES:* reports for the original unfiltered vcf files.
    - *FILTERED SNP VCF FILES:* reports for the filtered SNPs vcf files.
    - *FILTERED SNP INDEL FILES:* reports for the filtered indels vcf files.
    - *COMBINED SNP VCF FILES:* reports for the combined SNPs vcf files.
    - *ANNOTATED SNP VCF FILES:* report for the annotated vcf files.
- **THIRD COLUMN: STATISTICS**
    - QUALITY CONTROL OF FASTQ FILES
    - *FASTQC OF ORIGINAL FILES:* fastqc statistics for each original sample.
    - *FASTQC OF TRIMMED FILES:* fastqc statistics for each trimmed sample.
    - *TRIMMED FILES:* statistics for each sample of trimmomatic.
    - ALIGNMENT
    - *ALIGNMENT:* statistics of aligned reads.
    - VARIANTS
    - *ORIGINAL VCF FILES:* statistics for the original unfiltered vcf files.
    - *FILTERED SNP VCF FILES:* statistics for the filtered SNPs vcf files.
    - *FILTERED SNP INDEL FILES:* statistics for the filtered indels vcf files.
    - *COMBINED SNP VCF FILES:* statistics for the combined SNPs vcf files.
    - *ANNOTATED SNP VCF FILES:* statistics for the annotated vcf files.

The alignment statistics output contain the summary statistics of the alignment (align_stat.txt), the breadth (breadth_by_chr.txt) and the depth of coverage (depth_by_chr.txt) of each sample by chromosome. Besides that, there are files with the locus coverage (*.coverage) and the proportion of loci in function of the coverage (*.propcoverage) for each sample. A brief explanation about breadth and depth of coverage can be found here. The *.coverage *.propcoverage files are calculated with BEDtools. The columns of *.coverage files are:

```
CHROM    START_POSITION    END_POSITION    COVERAGE
```

The columns of *.propcoverage files are:

```
CHROM    COVERAGE    NUMBER_OF_LOCUS_WITH_THAT_COVERAGE    CHROM_LENGTH    PROPORTION.
```

The statistics of vcf files (original, filtered, combined) contain the statistics (*.stat) for each vcf file calculated with BCFtools stats. There are other statistics for each sample calculated with VCFtools: The *.frq files are the allele frequency. The *.ldepth.mean are the mean depth of coverage for each locus according to the samples used in the analysis. The *.imiss are the frequency of missing genotypes by individual.

The statistics for the annotated vcf files are created by snpEff and SnpSift programs.

IVDP Examples

IVDP comes with some examples to illustrate its use in different situations. Download the examples folder from here and save it in the IVDP folder. The samples and the reference genomes are from cattle (*Bos taurus*). For the paired-end samples, we used samples from SRA runs ERR035727, ERR035728, SRR4001708 and SRR4003107. For single-end samples, we used the same SRA runs, but only the _1.fastq file. We downloaded these sra samples and converted them to fastq files using prefetch and fasterq-dump functions of SRAtoolkit. A brief tutorial how to use if can be accessed here, and here. There are two cattle reference genomes: ARS1.2 and UMD3.1. We used only chromosome 29. We downloaded the reference genome, variants and annotation file from FTP Download - Ensembl. In the filter box on the right bottom corner, search for the desired specie (in our case cattle), and download the FASTA, GTF and VCF files. Please, see this picture.

Links to download the original files (with all the chromosomes)
ARS1.2

```
Reference: ftp://ftp.ensembl.org/pub/release-100/fasta/bos_taurus/dna/Bos_taurus.ARS-UCD1.2.dna.toplevel.fa.gz

Annotation: ftp://ftp.ensembl.org/pub/release-100/gtf/bos_taurus/Bos_taurus.ARS-UCD1.2.100.gtf.gz

Variants: ftp://ftp.ensembl.org/pub/release-100/variation/vcf/bos_taurus/bos_taurus.vcf.gz
```

UMD3.1

```
Reference: ftp://ftp.ensembl.org/pub/release-94/fasta/bos_taurus/dna/Bos_taurus.UMD3.1.dna.toplevel.fa.gz

Annotation: ftp://ftp.ensembl.org/pub/release-94/gtf/bos_taurus/Bos_taurus.UMD3.1.94.gtf.gz

Variants: ftp://ftp.ensembl.org/pub/release-94/variation/vcf/bos_taurus/bos_taurus.vcf.gz
```

NOTE: If you want to split the reference genome, variant and annotation files by chromosom, here are the codes:

```
### SPLIT REFERENCE GENOME BY CHROMOSOME
CHR=$(echo "$(seq 1 1 29) X Y MT")
bgzip -d -@16 Bos_taurus.ARS-UCD1.2.dna.toplevel.fa.gz
for chr in $CHR; do
   samtools faidx Bos_taurus.ARS-UCD1.2.dna.toplevel.fa $chr > ARS1.2_"$chr".fa
   samtools faidx ARS1.2_"$chr".fa
done

### SPLIT ANNOTATION FILE
bgzip -d -@16 Bos_taurus.ARS-UCD1.2.100.gtf.gz
for chr in $CHR; do
   awk -v chr="$chr" '$1 ~ /^#/ {print $0;next} {if ($1 == chr) print}' Bos_taurus.ARS-UCD1.2.100.gtf > ARS1.2_"$chr".gtf
done

### SPLIT VARIANT FILE
for chr in $CHR; do
   bcftools view bos_taurus.vcf.gz --regions $chr > bos_taurus_"$chr".vcf
   wait
   bgzip -f -@ 16 bos_taurus_"$chr".vcf
   bcftools index -f -t --threads 16 bos_taurus_"$chr".vcf.gz
done
```

Example 1

RNAseq analysis (ANALYSIS=2) with **Single-End** fastq files. In this example, it was used two aligners (bowtie2 and hisat2), 2 variant callers (bcftools and freebayes-parallel), the gene counts programs (htseq and featurecounts), and the VCF files will be combined using the option *partial* (COMBINE_VCF=partial). IVDP will process 2 samples at time (BATCH=2). The vcf files will be filtered using MIN_DEPTH=3, i.e. keep locus above 3x of coverage. To run:

```
./ivdpRun.sh -p examples/parEx1.txt -c examples/chromNames.txt
```

Example 2

Same as example 1 but considering repeated samples for the same animal. It is important specify repeated samples to the variant calling step. If you do not specify it, the variant calling programs will treat each sample as an independent animal. The list with sample names and the id of the animals should be created manually as this example:

```
cat examples/single_end/example_sample_id.txt
ERR035727 ANIMAL1
ERR035728 ANIMAL2
SRR4001708 ANIMAL3
SRR4003107 ANIMAL1
```

The first column is the prefix of sample files (ERR035727_R1.fastq.gz became ERR035727) and the second column is the animal ID. The only thing that changes in parEx2.txt compared with parEx1.txt is the variable **LIST_SAMPLES**

```
LIST_SAMPLES=/home/work/rps/IVDP/examples/single_end/example_sample_id.txt
```

To run:

```
./ivdpRun.sh -p examples/parEx2.txt -c examples/chromNames.txt
```

Example 3

Same as example 1, but only for gene counts and not for variant calling. The difference between parEx1.txt and parEx3.txt is that in parEx3.txt some options were deactivated to avoid run marduplicated reads, SplitnCigar, base quality score recalibration and the variant calling steps:

```
STEP_MDUP=no
STEP_BQSR=no
STEP_VCF_CALL=no
STEP_VCF_FILTER=no
STEP_VCF_COMBINE=no
STEP_VCF_ANNOTATE=no
```

Deactivating these steps, it **does not** matter the options chosen for **CALLER_PROGRAM**, **MIN_DEPTH**, and **COMBINE_VCF** variables. These variables will be deactivated automatically.

To run:

```
./ivdpRun.sh -p examples/parEx3.txt -c examples/chromNames.txt
```

Example 4

Whole genome sequence analysis (ANALYSIS=1) with **Paired-End** fastq files. In this example, it was used one aligner (bwa2), 1 variant caller (gatk4GenomicsDBImport), and the VCF files **will not** be combined using the option *partial* (COMBINE_VCF=none).
In this example, STEP_GENECOUNT=no, GENECOUNT_PROGRAM=none and FEATURE_TYPE=none because we do not desire gene counts from WGS.

NOTE1: Even if you forgot to change STEP_VCF_COMBINE=no and COMBINE_VCF=none, IVDP will deactivate them automatically beucase it was chosen only one variatn calling program (gatk4GenomicsDBImport). So, the output will be only one vcf file and there is nothing to be combined.

NOTE2: Even if you forgot to change those parameters, IVDP will change them automatically if you used ANALYSIS=1.

IVDP will process 2 samples at time (BATCH=2). The vcf files will be filtered using MIN_DEPTH=3, i.e. keep locus above 3x of coverage. To run:

```
./ivdpRun.sh -p examples/parEx4.txt -c examples/chromNames.txt
```

Example 5

Same as example 4 but using a differente reference genome (UMD3.1). The original data had trimmed in the previous example. So, we will skip the adapter trimming step (STEP_QC_TRIM=no) and use the output folder of the trimmed files of example 4 as the input directory of example 5. See bellow the variables that were changed from example 4 to example 5:

```
### SOME VARIABLES FROM THE PARAMETER FILE OF EXAMPLE 4 (parEx4.txt)
INPUT_DIR=/home/work/rps/IVDP/examples/paired_end
OUTPUT_NAME=cattleEx4
EXTENSION_PE1=_1.fastq.gz
EXTENSION_PE2=_2.fastq.gz

ASSEMBLY=ARS1.2
REFERENCE=/home/work/rps/IVDP/examples/ars1.2_chr29/chrom29.fa
VARIANTS=/home/work/rps/IVDP/examples/ars1.2_chr29/chrom29.vcf.gz
ANNOTATION=/home/work/rps/IVDP/examples/ars1.2_chr29/chrom29.gtf

STEP_QC_TRIM=yes
### COMMON VARIABLES BETWEEN par$x4.txt and parEx5.txt WERE OMITTED.
```

```
### SOME VARIABLES FROM THE PARAMETER FILE OF EXAMPLE 5 (parEx5.txt)
INPUT_DIR=/home/work/rps/output/ivdp_cattleEx4/data/trimmed_fastq
OUTPUT_NAME=cattleEx5
EXTENSION_PE1=.R1_paired.fastq.gz
EXTENSION_PE2=.R2_paired.fastq.gz

ASSEMBLY=UMD3.1
REFERENCE=/home/work/rps/IVDP/examples/umd3.1_chr29/chrom29.fa
VARIANTS=/home/work/rps/IVDP/examples/umd3.1_chr29/chrom29.vcf.gz
ANNOTATION=/home/work/rps/IVDP/examples/umd3.1_chr29/chrom29.gtf

STEP_QC_TRIM=no
### COMMON VARIABLES BETWEEN par$x4.txt and parEx5.txt WERE OMITTED.
```

To run this example, use:

```
./ivdpRun.sh -p examples/parEx5.txt -c examples/chromNames.txt
```