Short instructions for use: From VCF File to mtDNA Variant Analysis

1. Change the path in all .py files in the scripts folder to yours path,

2. Paste your VCF file into the Input folder with the following name "input",

3. Run run.py in Python,

Workflow described Figure 1.

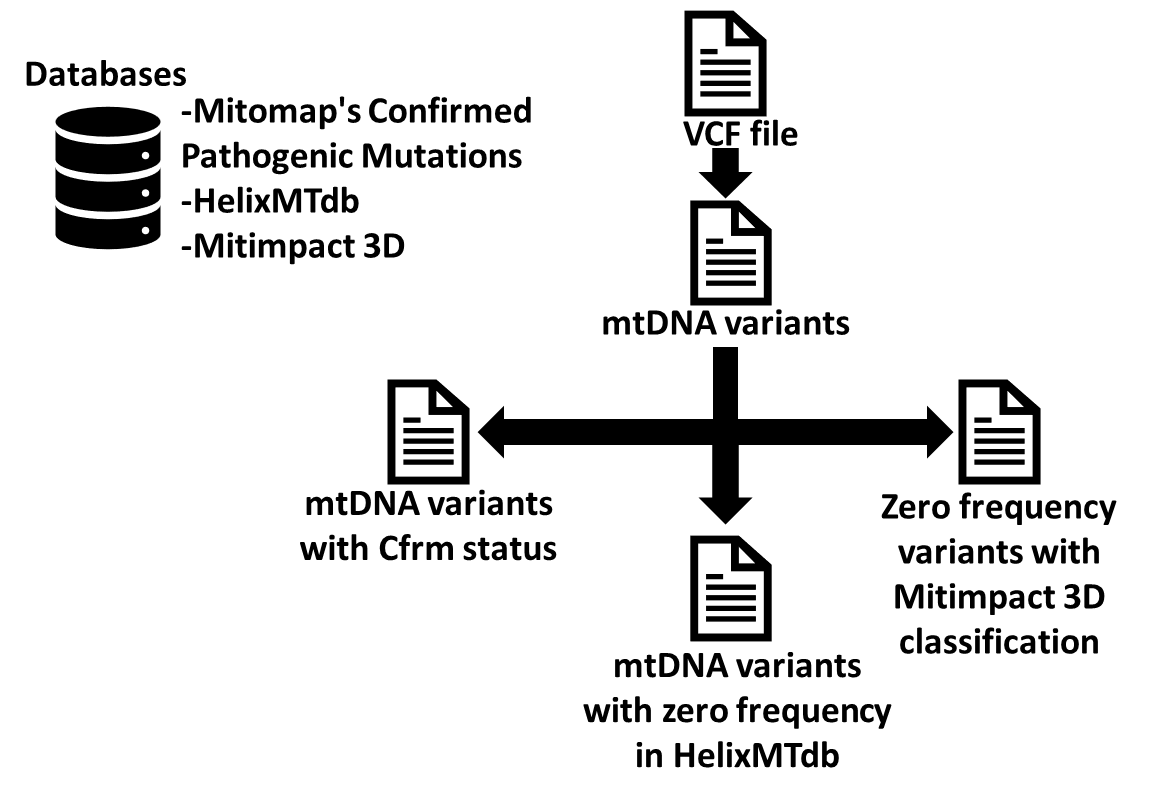


Figure 1 From VCF File to mtDNA Variant Analysis

* Data Analysis
* Step 1 - Extract from your VCF file MT position, reference allel, alternative allel, VAF and coverage into Step\_1.xslx,
* Step 2 - Check if some variants from VCF file are confirmed pathogenic variants by MITOMAP in Step\_2.xslx,
* Step 3 - Filtering variants by frequency in healthy population HelixMTdb, we got variants with zero frequency step\_3\_non\_matching.xslx and other variants matching frequency step\_3\_matching.xslx,
* Step 4 - Step\_3\_non\_matching.xslx is combined with Mitimpact 3D to specify the deleterious trait.

**Detailed instructions for use:**

1. Change the path in all .py files (5 files) in the scripts folder to your path (13 path for change), for example:

A picture containing text, font, screenshot

Description automatically generated

A picture containing text, font, screenshot

Description automatically generated

Paste your VCF file into the Input folder with the following name: input.vcf

If you are new to mtDNA variant analysis, we recommend that you use a combined VCF file from the same method to find artefacts.

**Step\_1\_VCF.py**

This script will search in VCF file following phrases:

* "MT" as mitochondrial chromosome, if you have a different name e.g. "chrM" change it in Step\_1\_VCF.py
* "VAF" as variant allele frequency
* "DP" as coverage

Output of this script is Step\_1.xslx composed of "Pos\_ref\_alt", "SampleID", "Variant-Level", "Coverage-Total" e.g.:

|  |  |  |  |
| --- | --- | --- | --- |
| **Pos\_ref\_alt** | **SampleID** | **Variant-Level** | **Coverage-Total** |
| 3243:A:G | One | 0.197368 | 152 |
| 3243:A:G | Two | 0.229412 | 170 |
| 3243:A:G | Three | 0.220126 | 159 |

**Step\_2\_VCF.py**

Using Mitomap's Confirmed Pathogenic Mutations list, this script filters for confirmed pathogenic variants in Step\_1.xslx for e.g.:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Pos\_ref\_alt** | **SampleID** | **Variant-Level** | **Coverage-Total** | **Locus Type** | **Locus** | **Associated Diseases** | **aaΔ or RNA** | **Status ♣(Mitomap [ClinGen])** | **Last StatusUpdate** |
| 1555:A:G | One | 0.940887 | 203 | tRNA | MT-RNR1 | DEAF | 12S rRNA | Cfrm [P] | 2018.04.18 |
| 3243:A:G | Two | 0.197368 | 152 | tRNA | MT-TL1 | MELAS | tRNA Leu (UUR) | Cfrm [P] | 2018.04.18 |
| 3243:A:G | Three | 0.229412 | 170 | tRNA | MT-TL1 | MELAS | tRNA Leu (UUR) | Cfrm [P] | 2018.04.18 |

**If your input VCF does not contain pathogenic variants, Step\_2.xslx will have no variants, only headers.**

**Step\_3\_VCF.py**

Filtering variants by frequency in healthy population HelixMTdb, we got one file with zero frequency mtDNA variants in step\_3\_non\_matching.xslx and second file step\_3\_matching.xslx with other variants existing in healthy population.

**Step\_4\_VCF.py**

In this step zero frequency variants from VCF file are compared to the Mitimpact 3D classification to out\_step\_4.xslx.