Sample invocations of ONE View

Simple command line

- Original line:

```
$./my_exe arg1 arg2 arg3
```

- Corresponding ONE View invocation:

```
$ maqao oneview -R1 -- ./my_exe arg1 arg2 arg3
```

Parallel (MPI / OMP) run

- Interactive command line
  - Original line:

```
$ OMP_NUM_THREADS=2 mpirun -n 4 ./my_exe arg1 arg2
```

  - Corresponding ONE View invocation:

```
$ OMP_NUM_THREADS=2 maqao oneview -R1 --number-processes=4\
--mpi-command="mpirun -n <number_processes>" \
--number-processes-per-node=2 -- ./my_exe arg1 arg2
```

    - OR (declaring the OMP_NUM_THREADS environment variable as a ONE View experiment parameter)

```
$ maqao oneview -R1 --envv_OMP_NUM_THREADS=2 --number-processes=4\
--mpi-command="mpirun -n <number_processes>" \
--number-processes-per-node=2 -- ./my_exe arg1 arg2
```

- Using ONE View configuration file
  - Generating empty template

```
$ maqao oneview --create-config=my_config.json
```

  - Filling up configuration file my_config.json (using interactive MPI run example above)

```
"executable": "my_exe",
"run_command": "<executable> arg1 arg2",
"number_processes":4,
"number_processes_per_node": 2,
"mpi_command": "mpirun -n <number_processes>",
"envv_OMP_NUM_THREADS": "2",
```

⚠ Environment variables must be declared as envv_<variable name> in a configuration file

  - Invocating ONE View with configuration file

```
$ maqao oneview -R1 --config=my_config.json
```

- Application launched through a script (e.g. job scheduler)
  - Original command line:

```
$ job_launcher my_script.job # job_launcher: slurm, bash, etc
```

  - Original script my_script.job: ⟹ Create new script my_script_maqao.job

```
#SBATCH -N 2  # Nb nodes
#SBATCH -n 4  # Nb tasks
…# some additional stuff
export OMP_NUM_THREADS=8
mpirun -n 4./my_exe arg1 arg2
```

```
#SBATCH -N <number_nodes>
#SBATCH -n <number_processes>
…# some additional stuff
export OMP_NUM_THREADS=<OMP_NUM_THREADS>
<mpi_command> <run_command>
```

  - Invoking ONE View using configuration file (`batch_command` and `batch_script` can also be set in the configuration file)

```
$ maqao oneview -R1 --batch-command="job_launcher <batch_script>" \
--batch-script="my_script_maqao.job" -config=my_config.json
```

Common options for ONE view

```
$ maqao oneview -R1 [options] -- ./my_exe my_args
```

ONE View general options:

- `--help`: Displays list of command line options and experiments parameters.
- `-xp=exp dir`: If specified, the results will be stored in directory `exp dir`. If omitted, directory `maqao_<timestamp>` will be created in the current directory.
- `--output-format=html(default)|xlsx|text|all`: Output format.
- `--with-scalability=[strong(default)|weak]`: Toggles scalability mode. ⚠ The `multiruns_params` array must be filled in the configuration file.

Using a configuration file for ONE View:

- `--config=config file`: Retrieves experiment parameters from `config file`.
  - Parameters in a configuration file are identical to those on the command line, but with no preceding '--' and replacing with '-' with '_'
  - They are declared as Json variables: `"option": "value"` or `"option": number`
  - Other parameters can be referenced by enclosing them in brackets (<>).
    - For instance: `mpi_command="mpirun -n <number_processes>"`
  - ⚠ Parameters names prefixed with '_' are considered **commented**
- `--create-config[=name]`: Generates empty configuration file.
  - If `name` is omitted, file will be named "**`config.json`**" and created in the current directory.
- `--create-config-template[=type]`: Generates sample configuration file.
  - If `type` is omitted, all templates will be created in the current directory.

Compare existing ONE View reports

```
$ maqao oneview --compare-reports --inputs=exp dir1,exp dir2,…
```

Viewing reports:

- Text reports are displayed directly on the console output.
- The path to the reports is displayed at the end of ONE View analysis.
  - HTML: open **`exp dir/RESULTS/<executable_name>_one_html/index.html`** in a browser to display the HTML reports.
  - XLSX reports are in file **`exp dir/RESULTS/<executable_name>_one_0_0.xlsx`**

Reading reports:

ONE View HTML reports regroup information across different tabs:

- **Global**: Global metrics for the whole application
- **Summary**: High-level recap of the main issues identified in the application and evaluation of the quality of the experiment
- **Application**: More detailed metrics at the application level
- **Functions**: Profiling results at the function and loop level
- **Loops**: Profiling and static analysis results at the loop level
- **Topology**: Information on thread execution and hardware usage