Getting Started with VM2

VM2 Version 3.2

VeraChem LLC

Copyright (c) 2015-2026, VeraChem LLC, Germantown, MD, USA. All rights reserved.

VeraChem has been issued a patent **(USPTO Patent No. 8,140,268)** for the VM2 method.

Contact:

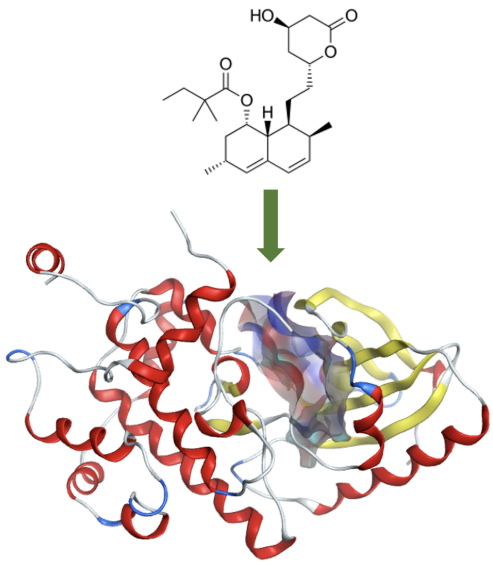
For information regarding VM2 software package licensing contact VeraChem LLC at sales@verachem.com

For technical support contact VeraChem LLC at support@verachem.com

For general enquiries contact VeraChem LLC at info@verachem.com

1 Welcome to VM2

Thank you for selecting VeraChem's VM2 software. VM2 is a comprehensive free energy package capable of generating fast and accurate predictions of binding affinity for protein-ligand and host-guest systems. A basic need in the small molecule drug discovery process is to determine how strongly a particular ligand or series of ligands bind to a protein active site and why.



VM2 has been designed and developed to provide computational and medicinal chemists with predicted binding affinity data across ligand series, for purposes of rank ordering, as well as for individual ligands for feedback on ligand design ideas. Binding free energies are computed directly by VM2 as the difference between the chemical potential of the complex and the sum of the chemical potentials of the ligand and the receptor alone.

$$\Delta G^{o}_{\mathrm{bind}} \;=\; \mu^{o}_{\mathrm{complex}} - (\mu^{o}_{\mathrm{rec}} + \mu^{o}_{\mathrm{lig}}) \tag{1}$$

The chemical potentials of the complex, receptor, and ligand are obtained by separate mining minima calculations, which search for low energy conformers and evaluate the contribution of each corresponding energy well to the system's total configuration integral Z, using a harmonic approximation plus numerical mode scanning to account for anharmonic effects:

$$Z_{\mathrm{i}} = \int_{\text{well i}} e^{-(U(\mathbf{r})+W(\mathbf{r}))/RT} d\mathbf{r} \tag{2}$$

$$\mu^{o} = -RTln((8\pi^{2})/C^{o}) \sum_{\mathrm{i}} Z_{\mathrm{i}}) \tag{3}$$

where R is the gas constant, T is the temperature, C^{o} is standard concentration, U is the potential energy and W is the solvation energy. More information on the theoretical basis of VM2 and the software package, as well as results of benchmarking of VM2 against experiment, can be seen in this presentation and on VeraChem's website.

2 Requirements

2.1 Installed VM2 Package

To install VM2 follow the instructions in the file `VM2_3.2_package_installation.txt` provided with the software package; alternatively, the same instructions can be found in the VM2 "Quick Start: Installation" documentation. Be certain to check that the automated installation tests were successful before proceeding. Note that a common source of failure for the installation/installation tests is unset environment variables, so in the case of failed tests make sure the following are set in your `.bashrc` file:

```
export $VCHOME=/home/yourname/installdir/vcCompChem_3.2
export VM2HOME=$VCHOME

export VCPYTHON=$VCHOME/exe/vc_python
export VM2PYTHON=$VCPYTHON
```

If you are still unable to sucessfully install VM2, contact VeraChem at support@verachem.com

2.2 Other Software

VM2 can optionally use AmberTools (version 18+) to prepare and parameterize protein receptors and ligands. This requires an AmberTools installation to be available on the user's default path so that it can be found by VM2 at runtime. We recommend installing AmberTools via conda (use of the miniforge distribution is recommended) and activating the resulting environment prior to running VM2. If access to OpenFF (including the AM1BCC-GNN partial charge method) is also desired, it is possible to install a single environment containing both AmberTools and OpenFF (see below). Instructions for obtaining and installing AmberTools alone can be found here.

While versions up to 23 are available via conda-forge, versions 24 and above are only available on the special `ambertools-dac` channel (as of the date of this release). AmberTools installations from `ambertools-dac` include scripts that run when the environment is activated. Among other things, these scripts set the environment variable `PYTHONPATH`, which interferes with the VM2 workflow by overriding normal Python path resolution. If using AmberTools version 24 or above, you must unset this variable before running VM2:

```
unset PYTHONPATH
```

This does not appear to affect any of the AmberTools functions used by VM2, because the `PYTHONPATH` set by the AmberTools activation script is already present in the Python system path. The explicit setting of `PYTHONPATH` is therefore unnecessary.

VM2 can also use the OpenFF toolkit to assign OpenFF parameters to ligands. This requires the `openff-toolkit` package and its associated dependencies to be installed and available. It is recommended that conda be used and the resulting environment activated when running VM2. By default, installation of `openff-toolkit` will also install AmberTools, so a single environment can cover both the AmberTools and OpenFF requirements. Installation instructions for OpenFF can be found here.

Note: An installation of AmberTools and/or OpenFF is currently recommended; however, the user also has the option to provide VM2 with receptor and ligand force field parameter and topology files directly in Amber prmtop format, in which case access to AmberTools or OpenFF is not required.

2.3 Computational Resources

The minimum recommended resources are a Red Hat-based linux system (Version 7+) with 8 to 12 CPU cores (Intel or AMD) and approximately 24 GB of RAM (2 GB per CPU core). VM2 may request up to ~8GB of virtual memory per cpu assigned, but resident usage is generally ~2GB / cpu for normal sized systems (scaling to ~4GB for those with very large mobile atom sets).

To compute the binding affinity for a protein receptor and a series of ligands using the VM2 the total number of free energy (chemical potential) calculations required is equal to the number-of-ligands x 2 + 1. On typical Intel or AMD processors the calculations for the receptor alone and complexes will each take approximately 20-30 minutes and the ligand-only calculations will take approximately 5 minutes each. Because VM2 based binding affinities are generated from a set of entirely independent free energy calculations, access to a computer cluster, where multiple 12-core calculations can be run concurrently will greatly reduce the wall-clock time taken to calculate binding affinities for a receptor and a series of ligands. Use of a resource management/queue system such as SLURM or PBS is highly recommended.

3 VM2 Package Overview

The VM2 package is composed of a highly efficient, compiled language, free energy calculation engine that is parallelized on CPUs and GPUs, and a python based workflow capability, with associated molecular manipulation tools, which facilitates application of the calculation engine to receptor and ligand series binding affinity calculations. The VM2 workflow, given raw receptor and ligand series structure files (e.g. PDB for protein, SD files for ligands) automates the required setup (molecule preparation, ligand 2D to 3D, forcefield parameter assignment, partial atomic charge assignment), generates run directories, allows for submission of free energy calculations via a resource manager or on the user's local machine, and, on completion of calculations, collects and organizes receptor-ligand series calculated binding affinities and other energy data for analysis, as well molecular conformers in formatted structure files e.g. PDB, MOL2, SDF, XYZ.

4 VM2 Workflow

4.1 Tutorials and Example VM2 Workflows

Two methods are provided to assist with setting up systems for VM2 calculations. The first uses pre-configured run templates (found in `$VCHOME/run_templates`) that match common use cases, such as computing binding energies for already-docked ligands or for congeneric ligand series that are not yet docked. Using a run template involves copying the appropriate template and supplying a few files describing your receptor and ligands, typically a PDB file for the receptor and MOL/SDF files for the ligands. In most cases, run templates are the quickest and simplest way to set up a new calculation. For cases that do not match any of the provided templates, the VM2 workflow script generator may be used. This text-based tool guides you through a series of questions to help configure your system, and offers greater flexibility, but may require more familiarity with VM2 than the run template approach.

Detailed step-by-step tutorials for both the run template and script generator approaches are provided in the `$VCHOME/tutorials` directory, covering the full setup of protein–ligand binding free energy calculations:

```
$VCHOME/tutorials/protein_ligand/run_templates/
    pl_run_template_1_predocked_ligands_tutorial/
    pl_run_template_2_congeneric_ligands_not_predocked_tutorial/

$VCHOME/tutorials/protein_ligand/script_generator/
    pl_tutorial_1/
    pl_tutorial_2/
```

`pl_tutorial_1` and `pl_run_template_2` are examples of setting up a complete end-to-end automated workflow for HIV-1 Protease and 38 ligands, where the ligands are supplied as 2D structures. In this tutorial the protein and ligand force field parameterization steps are automated, as well as the ligand initial placements in the binding site.

`pl_tutorial_2` and `pl_run_template_1` are examples of a workflow situation where the user has previously carried ligand placement within the binding site (via docking or other methods using third party or in-house tools). The system is c-Met Kinase and 24 ligands.

It is highly recommended that you work through at least one of them before working on your own system.

The installed VM2 package also includes VM2 workflow examples for various types of protein-ligand series and host-guest series binding affinity calculations. See the following directories:

```
$VCHOME/examples/vm2/workflow/protein_ligand

$VCHOME/examples/vm2/workflow/host_guest
```

The example workflows contain detailed README files. Running at least one of these example workflows is recommended as a useful test of your installation. Again, there are examples given for both set up methods. Run template examples are found in the `applied_run_templates/` directory.

When running any of the examples, be sure to update the resource manager related variables in the in the `set_vc_workflow_control_vars.sh` file, located in the run directory of each example, to match your own computing environment:

```
export QUEUETYPE='slurm'
export QUEUENAME='default'
```

As shipped, the examples are configured to use the SLURM resource manager and submit to whatever the default queue is for your environment. PBS is also supported (change QUEUETYPE to 'pbs') and if you do not have access to either SLURM or PBS you can select to run on the local machine, one calculation at a time, by changing QUEUETYPE to 'bsh'.

The examples are all set to use 12 cores for each calculation. You can change this by changing the following variables in the `set_vc_workflow_control_vars.sh`

```
export NUMMPIPROCS='12'
...
export NUMLIGMPIPROCS='12'
```

The first controls the number of cores used for calculations involving the protein (protein alone and complexes) and the second controls the number used for ligand alone calculations. Generally it is recommended that a minimum of 8 be used for good performance. See Section 4.2 below for more information regarding the `set_vc_workflow_control_vars.sh` file and the other run scripts.

Protein-ligand Series Example Workflow

For an example of a protein-ligand series (HIV-1 protease + 5 ligands) workflow where there is a co-crystalized ligand available with the same scaffold as the ligand series, see the directories:

```
$VCHOME/examples/vm2/workflow/protein_ligand/applied_run_templates/applied_run_template_2_congeneric_ligands_not_predocked

$VCHOME/examples/vm2/workflow/protein_ligand/vm2pkg_pl_hivp_umass5_workflows/vm2_coxtal_example

$VCHOME/examples/vm2/workflow/protein_ligand/vm2pkg_pl_hivp_umass5_workflows/rawdata_hivp_umass5_ad81template
```

Please see the readme text files found in the example directories for further details on how to run these protein-ligand VM2 workflows. If you would like to try a relatively short calculation the script generator umass5 example is set up with a five ligand subset of the full 38 ligand series. If umass5 is run on a single 12-core node, this protein-ligand series example will take approximately two hours to complete; on two 12-core nodes via SLURM/PBS it will take approximately one hour to complete.

Host-guest Series Example Workflow

For an example of a host-guest series (cucurbit[7]uril + 29 guests) workflow, where starting host-guest conformations are automatically generated by randomly rotating guest-only conformations and placing them into the host, see the directories:

```
$VCHOME/examples/vm2/workflow/host_guest/cb7_gilson_set/cb7_gilson_vm2

$VCHOME/examples/vm2/workflow/host_guest/cb7_gilson_set/rawdata_hg_cb7_gilson_set
```

The first directory 'cb7_gilson_vm2' contains the scripts for running the workflow steps, and the second directory 'rawdata_hg_cb7_gilson_set' contains the raw inputa data i.e. host and ligand SD/mol files. In the 'cb7_gilson_vm2' directory see the text file README.hgvm2 for more detail on how to run this VM2 host-guest workflow. **Note:** If run on a single 12-core node, this host-guest series example will take approximately four hours to complete; on two 12-core nodes via SLURM/PBS it will take approximately two hours to complete.

Schindler and Gapsys Benchmark Examples

The supporting information for (Gilson *et al.*, Rapid, Accurate, Ranking of Protein–Ligand Binding Affinities with VM2, the Second-Generation Mining Minima Method. *J. Chem. Theory Comput.* **2024**, *20* (14), 6328–6340. https://doi.org/10.1021/acs.jctc.4c00407) Contains complete ready-to-run setups, with several run setting variations, for all of the protein-ligand systems in the Schindler (Schindler *et al.* . *J. Chem. Inf. Model.* **2020**, *60* (11), 5457–5474) and Gapsys (Gapsys *et al.* Chem. Sci. **2020**, *11* (4), 1140–1152) free energy benchmarks. This includes 20 systems and a total of 536 ligands. It can be downloaded at: https://zenodo.org/records/11660114

4.2 VM2 Workflow Components

Whether you use run templates or the workflow script builder tool to set up your runs it is important to be aware of the structure of the VM2 workflow and the role of each of its components.

Given the required raw input data, e.g. protein PDB file, ligands SD file etc., the basic form of the workflow is 1) setup of molecular system, including forcefield parameter assignment, 2) pre-generation of ligand conformers, 3) generation of VM2 free energy calculation run directories, 4) submission of all VM2 free energy calculations, possibly through a resource manager, and 5) binding free energy results extraction into comma-separated-value (CSV) files and organization of generated molecular conformers in standard formatted files for visualization etc.

As seen in the example workflows, to carry out the full workflow just described above, the user can invoke the following five scripts, each corresponding to a workflow step:

```
./vc_workflow_1_setup.sh*
./vc_workflow_2_ligconfs.sh*
./vc_workflow_3_genrundirs.sh*
./vc_workflow_4_run.sh*
./vc_workflow_5_extract_results.sh*
```

The user can control aspects of each of these workflow steps by editing the workflow control parameter values in the file:

```
set_vc_workflow_control_vars.sh
```

Further details of required raw input data, workflow control parameters, workflow steps, and the workflow script builder tool now follow.

Raw Input Data

To use VM2's workflow framework, the first requirement is the placement of formatted structure files for the molecular system in a simple directory structure. Below is an example for a protein-ligand system:

```
rawinputdatadir/protein
rawinputdatadir/ligands
rawinputdatadir/experiment
rawinputdatadir/template_ligand
```

Typically, a PDB file for the protein receptor is placed in the `/protein` directory and a set of MOL files or an SD file are/is placed in `/ligands` . If experimental binding affinities are available to compare with the VM2 predicted values, they can be placed in a simply formatted text file in `/experiment` . Finally, to generate good protein-ligand complex starting structures for the ligand series, a SDF or PDB file of a ligand in a reasonable bound conformation in the protein active site should be supplied in `/template_ligand`. Often the simplest route to obtaining a template ligand is to extract a ligand bound to your target protein from the crystal structure used to provide the protein PDB file, i.e., the a co-crystalized ligand.

Control Parameters

Run parameters may be set by editing the file `set_vc_workflow_control_vars.sh` This allows control of many aspects of the VM2 binding free energy calculations, such as the path/name of the project directory, how many cpu cores to allocate to each calculation, how to assign partial atomic charges, how much of the protein should be mobile, and so on. The following are example parameters that set the resource manager as SLURM, targeting the queue name 'all' for both receptor involved and ligand-only calculations, requesting a single node and 12 MPI processes for each calculation.

```
export VCPROJECTDIR=$PWD/yourvm2project
...
export QUEUETYPE='slurm'
export QUEUENAME='all'
export QUEUENAMELIG='all'
...
export NUMNODES='1'
export NUMMPIPROCS='12'
export NUMOPENMPTHREADS='0'
export NUMGPUS='0'
...
```

These settings apply to each of the calculations required to compute binding energies for the ligand series. For N ligands, there are 2N+1 free energy calculations required: N ligands alone, N complexes, and one for the protein alone. Each of these calculations will be assigned one node and 12 cores. If SLURM or another resource manger is used, each of these calculations will be submitted as a separate job. Without a resource manager, each will be run in series, one at a time.

Keyword definitions are documented in comment blocks within the `set_vc_workflow_control_vars.sh` file itself. Further examples are given in the sections directly below. A full list of available keywords (control variables) are listed at the end of this document here along with a brief description of each one's purpose.

Molecular System Setup

The workflow setup invoked by `vc_workflow_1_setup.sh` carries out basic molecule preparation e.g. H-atom addition, ligand 2D to 3D conversion, forcefield atom typing/parameter assignment, assignment of partial atomic charges if necessary, as well as any requested optimization/relaxation of provided protein receptor structures. The setup step will be run entirely on the users local machine and will take several minutes to complete with the default settings. If a Conda installed AmberTools is being used, care must be taken that the appropriate Conda environment is set before proceeding.

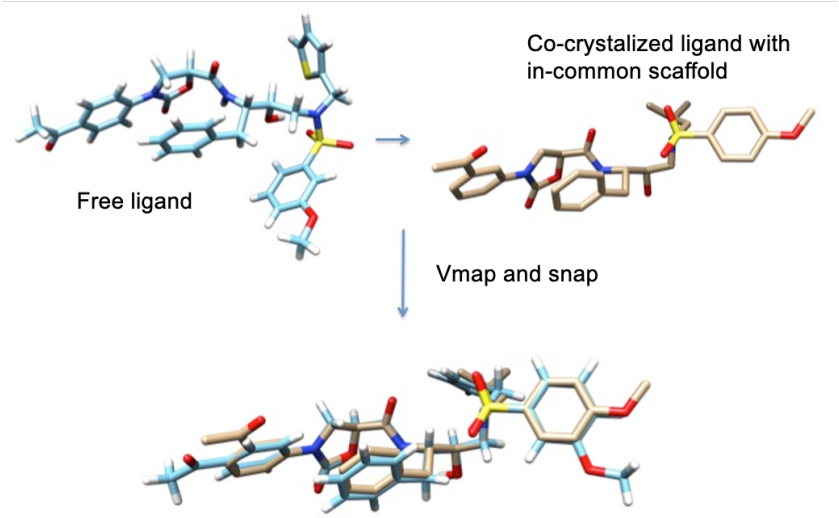
The following are examples of control parameters for the setup step, which set AmberTools as the protein and ligand preparation/typing/parameter assignment tool, the protein forcefield as 'ff14sb', the ligand forcefield as 'gaff2', VeraChem's Vconf tool for ligand 2D to 3D conversion, VeraChem's Vcharge tool for partial charge assignment, and so on:

```
export PROTEINFORCEFIELD='amber'
export PROTEINPREP='ambertools'
export PROTEINTYPER='ambertools'
export AMBERFORCEFIELD='ff14sb'
...
export REALATOMCUTOFF=6.0
export LIVEATOMCUTOFF=4.0
...
export LIGANDFORCEFIELD='gaff2'
export LIGANDPREP='ambertools'
export LIGANDTYPER='ambertools'
export LIGAND2D3D='vconf'
export LIGANDCHARGER='vcharge'
...
```

Ligand Conformer Pre-generation

The script `vc_workflow_2_ligconfs.sh` invokes the pre-generation of starting conformers for the ligand series. If being used, one option for the ligand conformer pre-generation step is a 'map and snap' setting,

```
export MAPLIGTOTEMPLATE='vmap'
export PREGENLIGCONFS='snap'
```

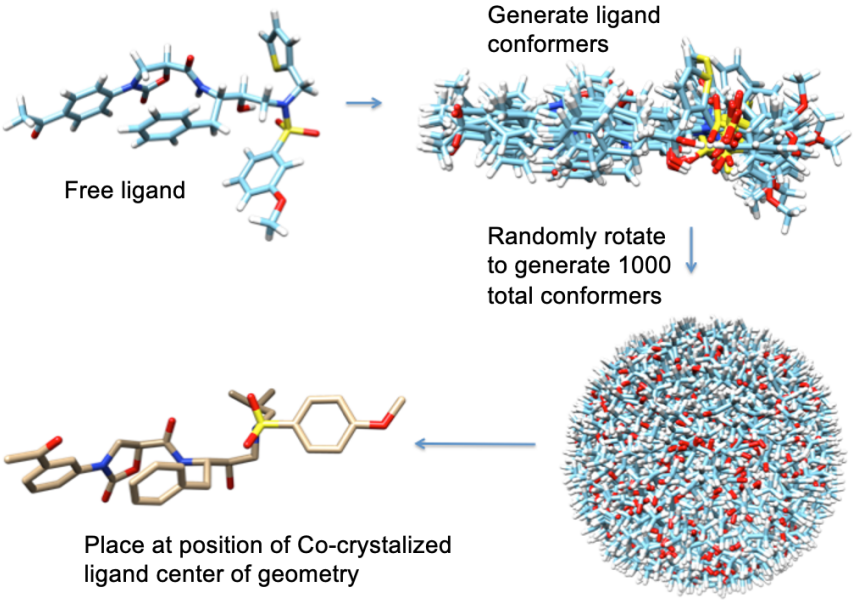


which uses VeraChem's Vmap tool for automatic mapping of common scaffold atoms on to a template scaffold e.g. co-crystalized ligand scaffold, followed by superposition of the common atoms with a guiding potential that 'snaps' the common scaffold into place. The results of a requested 'map and snap' step will be automatically extracted for use in later steps. They can be found in the directory:

```
/yourvm2project/run1/calculations/snap
```

The second option is the generation of a diverse set of conformers with which to seed the full VM2 conformational search when only the location of the active site is known through a co-crystalized ligand i.e. it does not share the same scaffold/general pose as the ligand series.

```
export MAPLIGTOTEMPLATE='none'
export PREGENLIGCONFS='random'
```



This is achieved through generation of ligand conformers, which are then randomly rotated around their center of geometry and placed at the center of geometry of the template ligand e.g. co-crystalized ligand. The results of a requested 'random' pre-generation of ligand conformers will, as in the 'map and snap' case, be automatically extracted for use in later steps. They can be found in the directory:

```
/yourvm2project/run1/calculations/random
```

The conformer pre-generation step takes approximately 30 seconds per ligand. It can be run locally or with faster turnaround on a cluster through a resource manager (i.e., SLURM or PBS).

Note that the ligand conformer pre-generation step can be skipped altogether, and the following parameter settings given in the control variable file, if the user is supplying ligand series coordinates that are *already* good starting poses in the binding site.

```
export MAPLIGTOTEMPLATE='none'
export PREGENLIGCONFS='none'
```

Generation of Run Directories

Running the script `vc_workflow_3_genrundirs.sh` builds and populates the directories in which the individual free energy (or chemical potential) calculations are cariied out. For a protein-ligand series workflow the directory hierarchy has the form:

```
/yourvm2project/run1/calculations/vm2/complexes
/yourvm2project/run1/calculations/vm2/complexes/proteinname/protein_name--ligand1
/yourvm2project/run1/calculations/vm2/complexes/proteinname/protein_name--ligand2
/yourvm2project/run1/calculations/vm2/complexes/proteinname/protein_name--ligand3
...
/yourvm2project/run1/calculations/vm2/ligands
/yourvm2project/run1/calculations/vm2/ligands/ligand1
/yourvm2project/run1/calculations/vm2/ligands/ligand2
/yourvm2project/run1/calculations/vm2/ligands/ligand3
...
/yourvm2project/run1/calculations/vm2/protein
/yourvm2project/run1/calculations/vm2/protein/protein_name
```

with each of these directories containing the required input files and a run script for either submission to a resource manager (`runvm2.slurm` or `runvm2.pbs`) or the local machine (`runvm2.bsh`). For example, the following is a typical set of required files for running an individual protein-ligand VM2 free energy (chemical potential) calculation:

```
protein_name.crd
protein_name.mol
protein_name.top                  <---- forcefield topology/parameters
ligand1.crd
ligand1.mol
ligand1.top                       <---- forcefield topology/parameters
ligand1_snap.xyz                  <---- pre-generated ligand conformers
protein_name_ligand1_vm2.inp      <---- calculation settings
protein_name_tethered_atoms.txt   <---- list of protein atoms to be constrained
runvm2.slurm*
```

Calculation Submission

The script `vc_workflow_4_run.sh` initiates the submission of the individual free energy calculations, stepping through each run directory generated by `vc_workflow_3_genrundirs.sh` and executing, depending on the users choice, the `runvm2.slurm`, `runvm2.pbs` or `runvm2.bsh` script in each. This step comprises the vast majority of the computational effort in the VM2 workflow. For the default settings, the time for an individual protein-ligand or protein-only free energy calculation, using 12 CPU cores, for example, is roughly 20-30 minutes, and is approximately 5 minutes for a ligand-only run.

Note that every VM2 free energy calculation in this step of the workflow is entirely independent; therefore, very fast throughput of protein-ligand or host-guest series binding free energy calculations can be achieved by running even on a modestly sized cluster. For example, on a cluster with 72 available CPU cores (e.g. 6x12-core nodes), a throughput of **12 to 18 protein-ligand free energies per wall clock hour** can be achieved.

Calculations submitted through a resource manager should be monitored by the usual commands e.g. `squeue` for the SLURM resource manager and `qstat` for the PBS resource manager. It is also advisable that some of the run directories be examined and output files checked soon after submission to confirm functioning calculations.

SLURM Configuration

VM2 uses a default set of SLURM control variables when submitting calculations. For example, for a run assigned 12 cores and no GPUs:

```
#SBATCH --job-name=vchemrun
#SBATCH --time=72:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=12
#SBATCH --ntasks-per-node=12
#SBATCH --cpus-per-task=1
#SBATCH --gres=gpu:0
#SBATCH --output=vchemrun.out
#SBATCH --error=vchemrun.err
```

The partition/queue used is controlled by the QUEUENAME setting in the `set_vc_workflow_control_vars.sh` file. By default MPI process pinning is turned off (I_MPI_PIN=0). This is the best and most stable setting in the majority of cases. However, if you observe poor performance or MPI crashes during runs, you may wish to try turning it on. This can be controlled via the MPIPROCPIN variable found in `set_vc_workflow_control_vars.sh` To turn pinning on, change this variable to:

```
export MPIPROCPIN='on'
```

If it is necessary to alter other SLURM settings, or to add additional ones (e.g. to request a specific amount of RAM such as --mem-per-cpu=8G) environment variables (e.g. SBATCH_MEM_PER_CPU = 8G) may be set in the shell prior to running this stage of the workflow. Environment variables set in this way override or add to the #SBATCH directives the VM2 places in the run `runvm2.slurm` script. You can find a list of environment variables and their corresponding options here.

Results Extraction

After all the submitted individual free energy calculations, i.e., complexes, ligands, and receptor, are completed, on running the script `vc_workflow_5_extract_results.sh` the results will be collected, summarized, and placed in the directory:

```
/yourvm2project/run1/results
```

The receptor-ligand series binding free energies and their energy components can be found in a calculation summary CSV file, along with the corresponding experimental binding affinities, if they were provided by the user in the raw input data directory. In addition, the free energies and component energies of the separate complexes, ligands, and receptor are available in CSV files, e.g., for protein-ligand:

```
/yourvm2project/run1/results/protein_name_vm2_SUMMARY.csv   <---- Binding free energies

/yourvm2project/run1/results/protein_name_complex_vm2.csv   <---- Free energies of complexes
/yourvm2project/run1/results/protein_name_protein_vm2.csv   <---- Free energy of protein
/yourvm2project/run1/results/vm2_ligand.csv                 <---- Free energies of ligands
```

Conformers generated for the complexes, ligands, and receptor during the VM2 calculation are also presented:in a number of file formats, allowing for easy viewing in a third party tool of your choice (e.g. MOE, VMD, Chimera, etc.) For protein-ligand series calculations, they can be found in the following directories:

```
/yourvm2project/run1/results/conformers/complexes
/yourvm2project/run1/results/conformers/protein
/yourvm2project/run1/results/conformers/ligands
```

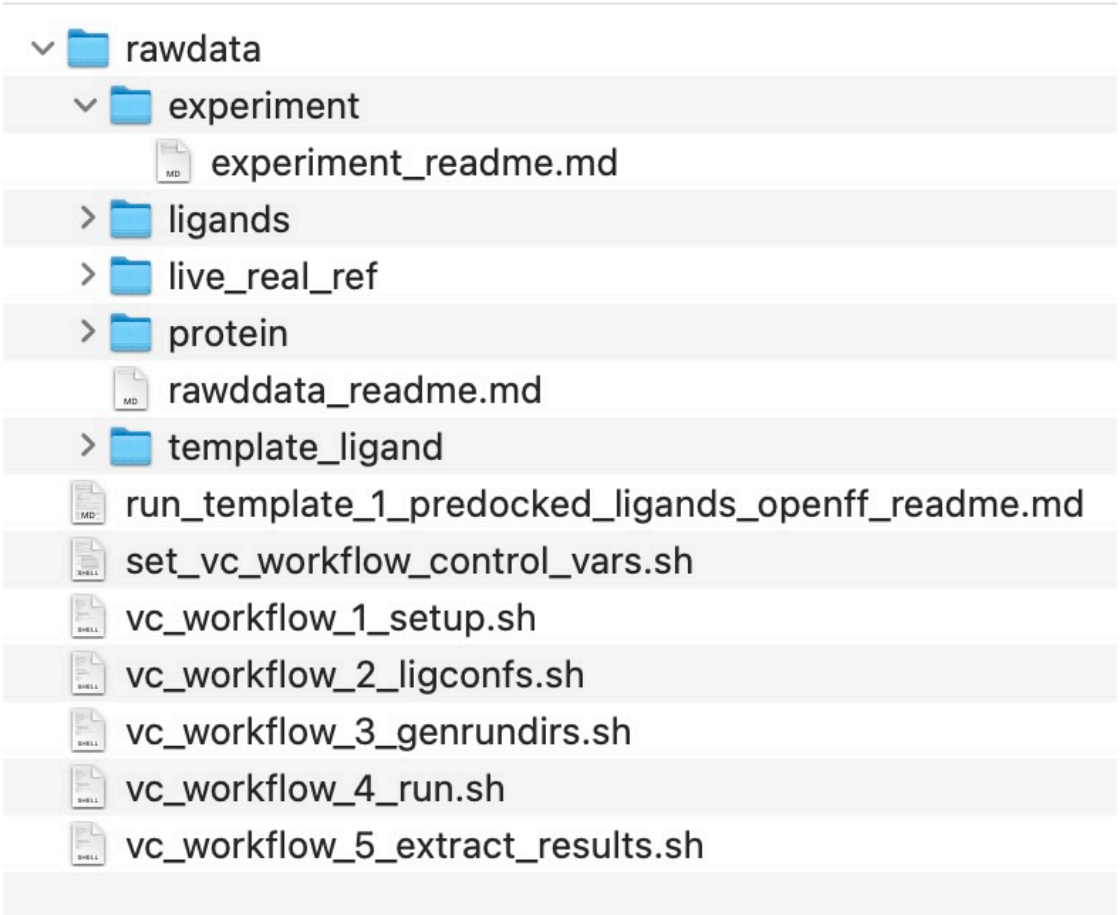
4.3 VM2 Run Templates

VM2 run templates (found in `$VCHOME/run_templates`) allow you to quickly set up a new VM2 calculation for a variety of common use cases. Each template is named to reflect the use case it implements:

```
run_template_1_predocked_ligands_gaff2
run_template_1_predocked_ligands_openff
run_template_2_congeneric_ligands_not_predocked
run_template_3_diverse_ligands_not_predocked
run_template_4_user_supplied_parameters
```

For example, VM2 Run Template 1 – Pre-docked Ligands (OpenFF) applies to cases where you have ligands already in pre-docked poses and wish to use OpenFF for ligand force field parameters. In most cases, using a VM2 run template only requires copying a few files describing your system into the corresponding input `rawdata/` directories and optionally adjusting a few run settings to match your computational environment. At that point the calculation is ready to run, and you can execute the provided shell scripts in order to carry out calculation stages 1–5.

Each run template provides a top-level readme file with instructions, a `rawdata/` directory tree in which you place the files defining your system, pre-configured run settings in `set_vc_workflow_control_vars.sh`, and a series of shell scripts for carrying out the calculation, divided into five stages. Each subdirectory of the run template contains its own readme file with further details. For example, here is an overview of the directories and files in VM2 Run Template 1:



This overview shows the readme from only one of the `rawdata/` subdirectories to avoid clutter; each subdirectory contains its own readme file. The top-level readme provides detailed information on the use case and requirements, along with step-by-step instructions for applying the template to a new system. For example, here is the beginning of the readme for VM2 Run Template 1:

VM2 Run Template 1 - Pre-docked Ligands (OpenFF)

Use this when:

You are providing ligands that are already in reasonable bound conformations, for example, if the ligands you are supplying to VM2 have already been docked to the receptor using another tool such as AutoDock Vina, DOCK, or Boltz-2. An example application of this template can be found in `/examples/vm2/workflow/protein_ligand/applied_run_templates/`

Requirements:

1. Protein receptor structure in PDB format.
2. Docked ligands in mol file format, one file per ligand, or a single sdf file.
3. (Optional) Experimental data for comparison to predicted values.

To use this template:

1. Copy the template directory to a new location and and rename to match your system.
2. Go through each of the directories in `rawdata/` and follow the instructions in the readme.md files found there.
3. Edit `set_vc_workflow_control_vars.sh` and review the following settings:

   If Slurm is not available change
   `export QUEUETYPE='slurm'`
   to
   `export QUEUETYPE='bsh`

   If using Slurm and you would like to use a non-default queue change
   `export QUEUENAME='default'`
   to designate the queue to use in your slurm configuration, e.g.
   `export QUEUENAME='fast-medium'`

   Update
   `export PROTEINFILENAME='<Protein PDB Filename>'`
   to match the filename of the protein you placed in rawdata/protein/, e.g.
   `export PROTEINFILENAME='4r1y.pdb'`

   If you have provided experimental data in `rawdata/experimental_data/` change
   `export HAVEEXPT='no'`
   to
   `export HAVEEXPT='yes'`

Other Information:

Parameters will be assigned to the protein from Amber 14SB and to the ligand from OpenFF, assigning partial charges using OpenFF's am1bcc-gnn neural network method.

Following this are sections describing important variations of the template that you may wish to explore. For example:

Variations:

Changing the number of cores used

For N ligands, there are 2N+1 free energy calculations required to compute binding energies for each ligand: N ligands alone, N complexes, and one for the protein alone. This template is configured to use 12 cores for each of these calculations, each of which is submitted to Slurm or PBS as a separate 12-core job. If you do not have a resource manager, each 12-core job will be run in series. You can change the number of cores used for each calculation; separate settings are available for protein-related calculations (i.e. complexes and the protein alone) and ligand-alone calculations (which typically require fewer resources). To change from the default settings:

For protein-related calculations, change
`export NUMMPIPROCS='12'`
to set the number of cores to use.
For ligand alone calculations, change
`export NUMLIGMPIPROCS='12'`
to set the number of cores to use.

A minimum of 8 cores is recommended for protein-related calculations and 4 for ligand-alone calculations. Computation time scales roughly linearly in the range of 4–16 cores, so calculations assigned 8 cores will take approximately 1.5× longer than those assigned 12.

Designating a different resource manager queue for ligand alone calculations

If you are using a resource manager like Slurm, you can also designate a different queue to be used for the ligand calculations. This might be useful as ligand calculations are less computational intensive and will often run acceptably using fewer cores.

To set a different queue to use for ligands alone replace 'none' in the following with the desired queue name
`export QUEUENAMELIG='none'`
`export QUEUENAMELIG='none'`

Different forcefield

You can select different forcefields for the protein and the ligand, and a different partial charge model for the ligand by changing the following variables:

`export AMBERFORCEFIELD='ff14sb'`
...
`export LIGANDFORCEFIELD='openff'`
...
`export LIGANDCHARGER='am1bcc-gnn'`

The available options for each are documented in the comments in `set_vc_workflow_control_vars.sh`, immediately above each variable.

Whenever you use a template, it is worth quickly reviewing these variations to see if any apply to your situation. This is also an excellent way to learn more about VM2's capabilities.

As noted above, the main task required to apply a run template is to place the necessary files describing your system in the `rawdata/` input directories. Each of these directories contains its own readme with detailed guidance on the files it expects. For example, here is the readme for the `rawdata/ligands/` directory:

VM2 Run Template 1 - Pre-docked Ligands

rawdata/ligands/

Setup

Place your ligands in this directory in mol file format, one file per ligand. It is also possible to supply the ligands in a single, multi-molecule, sdf file.

Requirements

Please ensure that the ligands have explicit bond orders, that is, bond orders must be specified as single, double, or triple, and not as 'aromatic'. The RDKit package can be used to convert from 'aromatic' to explicit bond orders via the Kekulize method. Stereochemistry should also be defined, and the ligands should be in the correct protonation states.

If a single SDF file is provided instead of separate mol files, each molecule must have a unique name assigned on the first line of its header within the SDF file. In addition, the `rawdata/live_real_ref/` directory currently only supports ligands in separate mol files. So if an SDF file is provided in `rawdata/ligands/`, the molecules must be separated into individual mol files if they are to be used in `rawdata/live_real_ref/`.

Since this run template is for pre-docked ligands, the run settings assume that the ligands have 3D coordinates representing reasonable bound conformations.

To apply a run template to your own system, make a copy of the appropriate template, as determined by the template names and the information in the "Use this when" section at the top of each template's main readme file. Then follow the instructions in the readme, which will guide you through the steps required to complete the setup.

4.4 VM2 Workflow Script Generator

The installed VM2 package also includes a command line question and answer 'wizard' application that provides a straightforward route for generation of the VM2 workflow scripts for a given use case; i.e., chosen molecular system, type of calculation, type of resource manager, number of compute cores to use, forcefields to use, system setup choices, etc. This VM2 workflow script generator 'wizard' application is called `generate_vm2_workflow_scripts.sh` and can be found in the `$VCHOME/exe` directory of the VM2 installation. Prior to running the script generator it is recommended that the raw input data directory structure be set up and populated.

On running the script generator via the command `$VCHOME/exe/generate_vm2_workflow_scripts.sh` the following message and initial question will appear:

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Welcome! VeraChem VM2 Workflow Scripting Wizarding shall now begin ...

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Answer a few questions about your molecular system and desired VM2 workflow.
A set of shell scripts for running your workflow steps will then be generated.

**Note**  This helper tool is most effective if a raw input data directory
location is supplied when asked for, as this enables additional validation of
the proposed workflow settings, and it also reduces/removes the need to edit
the generated scripts. (Though, this tool can certainly be used to generate
a set of template scripts for later editing if desired by the user.)

Before proceeding, do you want to see the general form of the raw input data
directory structure? Y/N
```

Simply answer each new question to move onto the next section. Once all the wizard question/answer sections are completed, it will automatically generate the workflow control file and five workflow scripts discussed above:

```
set_vc_workflow_control_vars.sh
vc_workflow_1_setup.sh*
vc_workflow_2_ligconfs.sh*
vc_workflow_3_genrundirs.sh*
vc_workflow_4_run.sh*
vc_workflow_5_extract_results.sh*
```

To make changes later or adjust settings not covered by the wizard question/answers you can directly edit the parameter control file `set_vc_workflow_control_vars.sh`. See the control parameter reference section for details of available parameter settings. Also, the generated workflow control file and scripts can be copied and edited to repurpose them for use with other systems. It is not necessary to run the generator for every change or to start setup for a new system if you have existing scripts generated with settings similar or identical to those desired.

5 Best Practices

Not surprisingly, much like other free energy software/methodologies, the achievement of accurate binding free energy predictions using VM2 has a significant dependence on the quality of the setup of the receptor and ligand molecular models, as well as selection of calculation settings that are suited to the molecular system. The following sections suggest basic best practices regarding molecular system input data, setup for calculations, common issues that may affect accuracy, and possible tuning of calculation settings.

5.1 Input Data

Protein receptor

Construction of molecular models for VM2 requires good receptor and ligand structures. For protein receptors this means a good quality (high resolution) X-ray crystal structure in the standard PDB format. If structures from the official Protein Data Bank are being used i.e. not in-house generated structures, the various repositories e.g. RCSB PDB, PDBe, and PDBj provide validation scores and analysis tools. Alternatively, re-refined structures may be considered e.g. PDB-REDO.

It is currently ***required*** that some basic editing of the PDB file be carried out before supplying it to the VM2 workflow raw input data directory. For example, any water molecules should be removed, unless they play a role in ligand binding, in which case they can be left in the PDB file and included explicitly in VM2 setup and subsequent calculations. Co-crystalized ligands should also be removed from the protein PDB file. Any cofactors or non-metal ions will, if not removed by the user, be identified by the workflow setup algorithms and removed automatically. In general, the AmberTools manual section "Preparing PDB Files" describes well the necessary hand-edits to PDB files that will allow subsequent workflow automated protein model setup to succeed.

Template ligand

If the protein PDB file contains a co-crystalized ligand that is similar in structure (i.e., shares a common scaffold or substructure) to the ligand or ligand series being considered (i.e. the target ligand(s)), it is highly recommended that it be extracted into its own PDB file and supplied to the VM2 workflow raw input data directory as a template ligand on which the target ligand(s) can be superposed, unless the user already has a suitable template from another source. Even if the co-crystalized ligand does not share a common substructure with the target ligand(s) it can still serve as template ligand that simply identifies the location of the binding site, if an alternative template is not available.

Host receptor molecules/ligand series molecules

For host molecules or target ligand(s), the input data requirement is 2D or 3D structures in MOL or SD file format, with explicit bond orders, formal charges, and stereochemistry defined. Example 3rd-party tools to generate SD files from 2D drawings, SMILE strings, etc., are MOE, RDkit, DataWarrier, and there are many others. VeraChem offers a 2D drawing program, Vdraw , which is valence/stereochemistry aware and can seamlessly send 2D drawings for 2D to 3D conversion and output to SD files via VeraChem's Vconf application. An additional vital aspect of the host or ligand molecules supplied to the VM2 workflow is their protonation states. Currently the VM2 workflow itself cannot assign protonation states, so they must be assigned (possibly with the help of 3rd-party software) and set beforehand e.g. by formal charge placement and bond order assignment and/or hydrogen addition in the SD file supplied to the workflow.

Experimental binding affinities

The VM2 workflow, if provided experimental binding affinity data, will, at the results extraction step, output these experimental affinities directly beside the associated calculated binding free energies, facilitating performance assessment via statistical analysis. When initiating a project applying VM2 to a protein target, if there is access to experimental binding affinity data for at least 10 ligands (preferably more) related to the eventual ligand target set (e.g., common substructure or scaffold), some initial validation/tuning runs are recommended to allow for possible optimization of the model and calculation settings for the best accuracy and efficiency.

5.2 System Setups

Given the required basic raw input data files just described, the VM2 workflow proceeds with further setup tasks to provide the final molecular models used in the VM2 free energy calculations. These tasks can be grouped into two stages. **Stage One** includes clean-up of problematic structures, normal structure preparation e.g. hydrogen atom addition, and forcefield atom typing/parameter assignment, and partial atomic charge assignment. **Stage Two** tasks are VM2 calculation related setup tasks, including generation of real and live atom sets, relaxation of protein structure around the binding site, possible ligand 2D to 3D conversion, and organization of the resulting molecular models in a directory structure for later use. The workflow can be set to carry out these tasks automatically by use of a combination of AmberTools (and openff-toolkit if OpenFF is selected) and VeraChem software tools.

AmberTools modules utilized

AmberTools modules are utilized only in Stage One setup tasks:

- *reduce* : optionally, the reduce program can be used to add H atoms to protein structures
- *tleap* : carry out Amber forcefield typing and parameter assignment
- *antechamber* : ligand atom typing and GAFF/GAFF2 parameter assignment; optional am1bcc partial atomic charge assignment

Note: The AmberTools program pdb4amber is not used, instead the VM2 Stage One workflow algorithms themselves automate basic preparation tasks (e.g. removal of alt atom locations, etc.) that are required before sending PDB files to reduce and tleap. In addition, the *reduce* module was removed in AmberTools version 26, so it is only available with AmberTools version 25 or earlier.

OpenFF modules utilized

OpenFF modules are utilized only in Stage One setup tasks:

- *OpenFF* : ligand series atom force field parameter assignment
- *AM1BCC-GNN* : generation of ligand series partial atomic charges

VeraChem python-based setup tools

Various VeraChem software modules/tools are utilized in both Stage One and Stage Two setup tasks. The following are python based tools:

- *prm2top* : convert prtmtop files to VeraChem formatted files
- *vconf* : ligand 2D to 3D conversion
- *vcharge* : module for fast generation of ligand series partial atomic charges
- *vmap* : perform substructure matching

VeraChem compiled language modules

In addition, the VM2 python workflow code interfaces with the VM2 free energy calculation engine, which has various submodules used for the following setup tasks:

- *snap* : snap target ligand scaffold atoms to template ligand scaffold atoms
- *geomoptHatoms* : relax protein hydrogen atoms
- *geomopt* : relax protein atoms in and around binding site
- *constructLiveReal* : carve out distance based real and live atom sets

Setup outside the workflow

If the user prefers they can carry some or all of the Stage One tasks *outside* the workflow, read in the resulting pre-prepared files (Amber prmtop, incpcrd, and other standard formats) and proceed directly with Stage two. Protein-ligand tutorial 2 (see Section 4.1 above) deals with this case and can be found at:

```
$VCHOME/tutorials/vm2/protein_ligand/script_generator/pl_tutorial_2
```

There is also an example workflow where all Stage One protein and ligand preparation, including forcefield typing and parameter assignment, occurs outside the VM2 workflow:

```
$VCHOME/examples/vm2/workflow/protein_ligand/vm2pkg_pl_cmet_workflows
```

This option provides tremendous flexibility to the user, who can always use the available fully automated workflow, but can choose to use any 3rd-party tools they wish to trouble shoot/prepare their structures, etc., if they prefer.

5.3 Trouble shooting

There are a number of issues that are quite commonly found in PDB files as well as small molecule data files (e.g. MOL/SD files) that may lead either to outright failure of the VM2 setup stages to complete or to suboptimal performance of VM2 binding free energy calculations when compared to experiment. The following are issues to watch for with suggested approaches to address them.

Protein: PDB file clean up/structure preparation

Protein PDB files, regardless of source, usually need some clean up before they can be used as a basis for any type of molecular modeling. If the fully automated VM2 workflow is being run with AmberTools for structure preparation etc. the workflow algorithms will attempt to carry out necessary clean up, it is good practice, though, even after successful completion of setup stages in the workflow, to still check log files and prepared structures for problems, for example the following files will contain useful information:

```
/yourvm2project/setup/PrepareProteinStage1/reduce_info.log
/yourvm2project/setup/PrepareProteinStage1/run_tleap.out
/yourvm2project/setup/PrepareProteinStage1/leap.log
```

Specific problems to look for and issues that may effect the quality of final binding free energy results are:

Residue naming

See the section "Residue naming conventions" in the AmberTools manual for details of expected naming when PDB files are being processed by *tleap* for atom typing and parameter assignment. For example, histidine naming should reflect protonation state (**HIS**, **HID**, **HIP**); bridging cystines should be named **CYX**; neutral versions of the normally charged **ASP, GLU**, and **LYS** should be named **ASH**, **GLH**, and **LYN**, respectively; and neutralizing N- and C- terminal capping groups are named **ACE** (COCH3), **NHE** (CONH2) and **NME** (CONHCH3). The VM2 workflow algorithms will

usually successfully convert from regular PDB residue naming to *tleap* ready naming for standard residues using appropriate heuristics e.g. for two **CYS** residues with sulfur SG-SG atoms < 2.5 Angstrom apart found in the input PDB it will rename them to **CYX**, and for histidine groups it will assign the correct name ((**HIS**, **HID**, or **HIP**)) according to explicit hydrogens if they are present, but if they are not it will rename as **HIE** (other charge state dependent naming will be searched for), and it will examine any terminal groups and assign Amber naming to them etc. While for standard residues the workflow will almost always provide correct naming for *tleap* to succeed if it does not the resulting error messages are collected and output for convenient troubleshooting.

Missing residue atoms

If residue heavy atoms, partial side chains, or complete side chains are found to be missing they can either be added in their ideal geometry, or through rotamer libraries, to the system before typing and parameter assignment. If, however, they are very far away from the binding site or potentially will not even be present in the real/live set, they can be capped as GLY. The idealized geometry of any added side chain close to the active site should be relaxed sometime during the setup process. The *tleap* program does have some capacity to add missing atoms/side chains. If on inspection of *tleap* output and the newly modified protein structure, missing atoms problems persist, it is advisable to address this outside the workflow either with AmberTools directly (which exposes more options to the user) or with another 3rd-party software that can add missing heavy atoms/ side chains, e.g., MOE, Chimera, etc.

Non-standard (modified) amino acids

Any modified amino acid residues should be typed and parameterized if 'official' parameters exist. If 'official' parameters do not exist and the residue is far from the active site it may be simply swapped out for a standard residue; however, if the residue is close to the active site it should be typed and charged using a generalized scheme. Again, the *tleap* output should be examined and if necessary AmberTools should be used externally to the workflow to better utilize existent parameters (see the section 'Modified amino acids and nucleotides' in the AmberTools manual), or carry out non-standard parameter assignment etc.

Metals ions and other non-standard residues

If they are not already removed by the user, non-metal ions and non-standard residues, such as ligands and co-factors, will be removed from PDB files by the VM2 workflow setup algorithms. If a metal ion exists in the active site of the protein and is involved directly in ligand binding, it is advised that, currently, VM2 should not be used to compute binding free energies (though development work in this direction is ongoing). For systems with metal ions far away from the active site or even in the region of the active site, but not involved in ligand binding, it may be adequate to allow the default *tleap* typing and parameter assignment, but more sophisticated treatments such as generation of quantum mechanics based parameters for the metal ion and surrounding residues would be preferable. (Automation of such schemes is ongoing.)

Chain breaks

If a chain break (gap) occurs near the binding site or in a loop that plays an important role then the missing residues should be added. Addition of missing loops will probably be best approached outside the workflow with 3rd-party tools. If the chain break is very far away from the binding site and likely to be fixed in space or even not present in the real/live set, then the termini of the break can be capped. The VM2 workflow setup algorithms will identify chain breaks and carry out addition of TER records to the PDB and this will result in the introduction of charged capping groups, N (NH3+) or C (COO-), by *tleap* . This will suffice for regions dropped out of the real set for VM2 calculations,

and may suffice for termini inside the real set but far away from the active site, though addition of neutral caps (ACE or NME) would be preferable.

Protein stereochemistry

Stereochemistry errors that can occasionally occur in PDB files are *cis* peptide bond arrangements and incorrect alpha-carbon parity. It is preferable these issues are detected and fixed during the preparation stage. This requires manipulation of the PDB geometry coordinates. Structure visualization/analysis/manipulation tools (MOE, Chimera) should be used to correct stereochemistry errors, especially those in or close to the binding site.

Specific protein residue protonation states

There are five ionizable protein residue types: ASP, GLU, HIS, TYR, LYS. Their states are known at standard physiological pH 7.4, but the local pH in a binding pocket can vary according to the local structure. Also, it is possible that a ligand on binding will induce a change in the protonation state of binding pocket residues. Users may want to employ 3rd-party software that attempts to predict protein system protonation states by specific residue e.g. Propka. Currently, this should be done outside the VM2 workflow. Care should be taken to confirm to residue and atom naming conventions if submitting resulting adjusted PDB files to the workflow to complete the setup.

Host and ligand molecules

Host molecules and especially ligand molecules provide further challenges with respect to preparation, typing, and parameter assignment. In contrast to protein systems, where constituent amino acids are specifically characterized and preparation tools can take advantage of this prior knowledge, generalized approaches must be used. Generalized preparation steps include hydrogen addition, protonation state assignment, bond order recognition, assignment of stereochemistry, and generalized typing and parameter assignment. As described above, the VM2 workflow expects host and ligand input data files in MOL/SD file format, with protonation states, bond orders, and stereochemistry already defined. Given this, the workflow setup invokes the *antechamber* tool to carry out further preparation tasks e.g. hydrogen atom addition if hydrogen atom not already present, before carrying out atom typing and GAFF/GAFF2 parameter assignment, and, if requested, *am1bcc* partial atomic charge assignment. The faster VeraChem *vcharge* tool is used by default for partial atomic charge assignment. Alternatively, the OpenFF toolkit can be used to assign ligand force field parameters, with the option of using the *amb1bcc-gnn* charge model. Successful completion of *antechamber* and *vcharge* or *OpenFF* based setup steps relies heavily on correctly formatted definitions of bond orders, stereochemistry, and protonation states in supplied MOL/Sd files; furthermore, ultimately, chemically correct (or at least chemical sensible) assignments of these properties are vital to achieving accurate VM2 binding free energy predictions:

Bond orders

Bond orders must be supplied in host/ligand MOL/SD files as single, double, or triple bonds, i.e., Bond Block values of 1, 2 or 3. Setting bond orders as aromatic (Bond Block value 4) will result in failed ligand setup. Furthermore, it is important to assign chemically 'correct' bond orders as they can effect force field parameter assignment, and in addition the VM2 algorithms make use of bond order information when performing conformational searches and other procedures.

Stereochemistry

The stereochemistry of the host/ligand molecules supplied through the MOL/SD file is propagated through to the VM2 calculations, which unless specifically requested not to, will maintain that stereochemistry throughout conformational searches etc. rejecting conformers that violate it. As protein-ligand binding affinities can have a strong dependence on ligand stereochemistry, it is important that correct stereochemistries are supplied from the start at the setup stage.

Protonation states

Protonation states can have a very large effect on predicted protein-ligand and host-guest binding free energies and their accuracies compared to experiment. If pKa values are known for the ligand under consideration (or for a similar chemical motif) then this may be adequate for good choices. If no pKa values are available, the use of third-party software for prediction of protonation states should be considered.

5.4 Molecular model and calculation considerations

Protein receptor model

Explicit solvent

Protein PDB files usually include water molecules, with the water oxygen atoms indicating their positions. As already stated above, all water molecules that are not identified as playing a role in the binding of the co-crystalized ligand and potential target ligand series, should be removed from the PDB file before it is supplied to the VM2 workflow setup. Water molecules that are identified as important can be included in the VM2 calculation by simply leaving them in the PDB file supplied for setup, though a constraint should be applied to the water oxygen atom - see below for details.

Although the inclusion of explicit water may turn out to be required in order to achieve accurate calculated binding free energies for particular protein targets and ligand series, it is often far from obvious by initial visual inspection of structures. Users may want to utilize 3rd-party software that may algorithmically identify important water molecules in the binding site, for example Grid Inhomogeneous Solvation Theory (GIST), available in *cpptraj*, which is part of AmberTools. A more heuristic approach is to examine multiple X-ray crystal structures, with similar co-crystalized ligands (i.e. common scaffold) to the ligand target series, and look for conserved water molecules in the active site.

Choice of protein real/live set

The choice of protein atoms to include in the VM2 calculations (real atoms) and which of these should be mobile (live atoms) can have a large effect on the accuracy of VM2 predicted binding free energies. Enough protein atoms should be live so that the protein environment can adjust appropriately when the binding ligand is present. Furthermore, if the system under consideration contains a flexible loop that can affect the active site, as it the case for kinase systems, this loop should be mobile. Of course, the larger the real/live set is, the more computationally demanding each VM2 search step will be. In addition, the search space rapidly increases too, which can mean more conformational searching is required to achieve free energy convergence. So choosing a real/live set that is larger than needed should also be avoided.

The real/live atom set is defined by distance to a reference set of atoms in the binding site. The reference can be a bound molecule (e.g. co-crystalized ligand PDB coordinates), a group of molecules (e.g. a whole ligand series, already well placed and with good poses, possibly from docking), or a ligand superimposed on another smaller template with a common substructure/scaffold. All these exists as options in the VM2 workflow. A key consideration is that if the ligand series various in size, the defined real/live set must accommodate the whole range of sizes up to the largest ligand in the series.

Currently, the recommended starting real/live atom distance cutoffs are 6 Angstroms (real) and 4 Angstroms (live). On completion of the workflow step one, these atoms sets are available in PDB and mol2 formats, and so can be visualized and assessed. One possible issue to look for when there are large flexible ligands, is potential for part of the ligand to move behind the defined real set. In this case the real set can be increased, while initially retaining the same sized live set e.g. 8 Angstroms (real)/6 Angstroms (live).

Live and real set consistency between runs

If you plan to run multiple sets of ligands across multiple VM2 calculations and wish to compare the resulting binding free energies across runs, it is critical that the real and live sets be identical across all calculations. Different real and live sets will produce different energies, because differing numbers of atoms will be included in the energy calculation and, further, different atoms will be included in conformational optimization.

To ensure the same real and live sets are used across multiple VM2 runs, you must choose a definition method that does not vary between runs. This can be accomplished by defining the sets relative to a common template ligand, a common reference ligand, or a set of bound ligands that remains the same across all calculations. For example, rather than defining the sets via distance cutoffs applied to all ligands in the current calculation, you could select a representative sample of ligands that spans the sizes, shapes, and binding modes of all ligands you expect to run. This same set of ligands would then be used to define the sets in every calculation. A convenient approach is to use the ligands from the initial validation of VM2 on the receptor in question to define the live and real sets when running subsequent ligands. These options, and details on how to set them up, are discussed further in the readme files supplied with the run templates located in `$VCHOME/run_templates`.

It is also possible to directly specify the receptor atom numbers to be used for the real and live sets. In addition to keeping the sets consistent across runs, direct specification offers greater flexibility for fine-tuning the sets compared with relying on distance cutoffs. When using this method, you need to be careful not to use other settings that might change the number of receptor atoms, for example, using the VM2 workflow to add hydrogens to your protein.

If you are interested in using this method, please contact VeraChem support at support@verachem.com and we would be happy to provide an example and assistance.

Relaxation of protein atoms in and around the binding site

The relaxation of atoms in and around the binding site from their positions in the original PDB X-ray crystal structure can be important as it can relieve strain from near contacts and/or strain present because for the applied forcefield/energy model the PDB atoms are not at their minimum energy positions. In the workflow setup, if not turned off by the user, relaxation of hydrogen atom positions only is carried out by default in a region 12 Angstroms from the reference. Next, optionally all protein atoms are relaxed in a region, by default, 6 Angstroms from the reference. The workflow allows for a bound ligand to be present during these relaxations, with the motivation for this

being the avoidance of unwanted movement of protein atoms into parts of the binding site that may hinder ligand binding.

Host and ligand models

Partial atomic charge assignment

The workflow offers a choice of VeraChem's electronegativity equalization method Vcharge for host and ligand partial atomic charge assignment, or the AM1-BCC method as implemented in the *antechamber* tool in AmberTools. When using AmberTools version 24+, the ABCG2 method, for use in combination with GAFF 2+, is an additional option. In extensive benchmarking of protein-ligand and host-guest systems, the use of partial atomic charges from VeraChem's Vcharge tool did not affect the correlation of predicted binding free energies with experiment, but it is much faster. Therefore, the recommended default charging method in the VM2 workflow is Vcharge.

VM2 calculation settings

The underlying VM2 calculation engine allows for fine control over many aspects of the VM2 algorithms. Currently the VM2 workflow exposes only a few of these: the level of conformational searching and the dielectric used for the continuum solvation models.

Conformational searching

The workflow default level of conformational searching - see options and default in the **Calculation engine control** section of the workflow control parameter values reference below - has been found reliable for protein-ligand series calculations where the ligand series share a common scaffold with an available co-crystalized ligand or where good starting poses are available for the ligand series, and also where the suggested 6 Angstrom (real)/4 Angstrom (live) atom sets have been used. For larger real/live sets than this it becomes more likely that higher levels of conformational searching will be required for reliable convergence. This is also the case where a template ligand in the right confirmation/pose with a common scaffold to the target ligand series is not available, and randomly oriented ligand conformers are placed in the binding site to generate starting conformers to seed subsequent conformational searching. See the following example workflow, which greatly benefits from increased conformational searching beyond the default settings:

```
$VCHOME/examples/vm2/workflow/protein_ligand/vm2pkg_pl_hivp_umass38_workflows/vm2_ukwnpose_exa
mple

$VCHOME/examples/vm2/workflow/protein_ligand/vm2pkg_pl_hivp_umass38_workflows/rawdata_hivp_uma
ss38_xk2template
```

Continuum solvation dielectric

The Generalized Born (GB) and Poisson Boltzmann Surface Area (PBSA) continuum models are used to provide solvation energies during VM2 calculations. Testing to date has not shown that varying from the default dielectric constants (internal 1.0; external 80.0) can improve the correlation of VM2 predicted binding free energies with experiment.

5.5 Validation/Tuning

Upon the basis of VeraChem's own extensive VM2 benchmarking studies, for new systems it is highly recommended that if experimental binding affinity data is available for a set of ligands related to the target ligand series, validation/tuning studies should be carried out to establish model and calculation settings most favorable to producing good correlations with experiment. Some suggested workflow variations and steps to achieve validation/tuning are:

- Start with 6 Angstrom (real)/4 Angstrom (live) set calculation, no explicit water molecules, and default conformational search level.
- If water molecules conserved across X-ray crystal structures, include them a re-run the above workflow, if improved correlation with experiment retain the explicit waters in subsequent validation runs.
- Expand real/live set to 7 Angstroms/5 Angstroms. If improved correlation shown increase real/live to 8 Angstroms/6 Angstroms, but also increase level of conformational searching.
- Assess results across the workflow variations with respect to correlation with experiment and choose model/calculation settings to apply to target ligand series.

6 Further Workflow Details

For the most part the VM2 workflow is automated to the point where only keywords/control parameters need to be set. Currently, though, for a few capabilities additional details/formatted files need to be supplied by the user.

6.1 Explicit Water Molecules

To include explicit water molecules in protein-ligand VM2 calculations they should be included in the protein PDB file that is supplied to the workflow. For example, in the protein-ligand workflow example referred to above in Section 4.1, the PDB file supplied in the raw input data protein directory

```
rawdata_hivp_umass5_ad81template/protein/2i0d_1580.pdb
```

a single water molecule oxygen atom is present at the end:

```
...
ATOM   1501  CE2 PHE B  99      30.167  22.169  29.272  1.00 46.75           C
ATOM   1502  CZ  PHE B  99      29.238  23.190  29.224  1.00 46.64           C
ATOM   1503  OXT PHE B  99      28.098  19.116  32.273  1.00 45.39           O
TER    1504      PHE B  99
HETATM 1580  O   HOH A 511      19.943  32.396  11.709  1.00 20.12           O  <-- water
END
```

and will be processed in workflow setup steps and included in the VM2 calculations. It is important that included explicit water molecules remain about in the same region across protein-only and protein-ligand complex conformers, so a constraint on the water oxygen atom is recommended - see next section.

6.2 Adding Geometric Constraints

Harmonic or 'flat bottom' polynomial constraints can be applied automatically to the oxygen atoms of all explicit waters as well as to atoms specified in a formatted text file, which must contain `tethered_atoms.txt` in its name, is included in the raw input data protein/ directory e.g.

```
rawdata_hivp_umass5_ad81template/protein/2i0d_tethered_atoms.txt
```

Application of constraints to water and to any specified atoms is controlled by the **TETHERWATERMTD** variable in the set_vc_workflow_control_vars.sh script. (See below)

This is an example of the format of the file used to specify atoms:

```
#Tethered Atoms information
#numProtein
1
#numLigand
1

#proteinid
1

#setid
1
# water oxygen         <---- Comment only, can edit
#numTetheredAtoms
1
#atomList
3135                   <---- Atom index in the prepared protein

#ligandid
1
#numTetheredAtoms
0
#atomList

#end
end
```

For an example of host molecule constraint file see, for example, the raw input data hosts directory for the host-guest series example mentioned in Section 4.1:

```
rawdata_hg_cb7_gilson_set/hosts/cb7_tethered_atoms.txt
```

6.3 Experimental Data File Format

As described above, experimental binding affinity data can, if the user wishes, be supplied to the VM2 workflow, and then on completion of the full binding free energy calculation workflow, it is included in the same CSV file that summarizes the calculated binding free energies, and component energies, for the receptor-ligand series. This facilitates assessment of VM2 accuracy for the given molecular system, setup, and run parameters. The experimental data is supplied to the workflow through a formatted text file in the raw data directory:

```
rawinputdatadir/experiment/experimental_data.txt
```

(**Note**: the file name must contain `experimental_data.txt`, but the name can be extended as desired by the user e.g. `system_name_experimental_data.txt`)

The formatting of the file for a receptor and a series of ligands is a row for each complex containing the receptor name (or substring of the receptor name) linked by a double dash to the ligand name (or unique substring of the ligand name), comma separated with the experimental binding affinity in kcal/mol. For a protein-ligand series the basic form, then, is as follows:

```
protein_name--ligand1, affinity1
protein_name--ligand2, affinity2
protein_name--ligand3, affinity3
...
```

The workflow examples provide actual examples. For example, see the file `experimantal_data.txt` in:

```
$VCHOME/examples/vm2/workflow/protein_ligand/vm2pkg_pl_hivp_umass5_workflows/rawdata_hivp_umas
s5_ad81template/experiment
```

For two different host molecules and a common set of three ligands (guests):

```
host1_name--ligand1, affinity1
host1_name--ligand2, affinity2
host1_name--ligand3, affinity3
host2_name--ligand1, affinity4
host2_name--ligand2, affinity5
host2_name--ligand3, affinity6
```

Again an actual example can be seen in the workflow examples. See the file `sampl5_oa_experimental_data.txt` in:

```
$VCHOME/examples/vm2/workflow/host_guest/Sampl5/rawdata_sampl5_oa/experiment
```

6.4 Updating Existing Setups to Version 3.2

Version 3.0 VM2 setups should run fine with Version 3.2. However, if you would like access to the expanded options provided by Version 3.2, it is necessary to update existing settings (found in `set_vc_workflow_control_vars.sh`) and run scripts. A script is provided to do this, preserving current settings and adding new options with their default values. This script, `update_run_scripts.py`, is located in `$VCHOME/utilities`.

The script will find all existing setups under a specified directory and update them using a specified set of template scripts. Version 3.2 template scripts for both protein-ligand and host-guest systems are provided in the same directory.

For example, to update all setups found beneath a directory called `version_3.0_setups` to version 3.2, run (using python version 3+):

```
python $VCHOME/utilities/update_run_scripts.py ./version_3.0_setups
$VCHOME/utilities/protein_ligand_scripts_3.2
```

Updating setups that have already been run, and therefore contain output, is not recommended, as it will result in settings and run scripts that do not match the output, which could be confusing. Instead, preserve the originals and apply the update script to a copy. The `clone_run.py` script in the utilities directory may be useful for this, as it copies setups without their associated output. A number of other utility scripts are also present in that directory; the readme file there describes each of them.

7 Control Parameters Reference

Usage: Set the control parameter environment variables (left hand column of the reference tables below) in the file set_vc_workflow_control_vars.sh using the 'export' command e.g.,

```
export MOLSYSTEMTYPE='protein+ligand'

export CALCNTYPE='vm2'

export VCWORKFLOWEXE=$VCHOME/exe/VM2.pyc

...
```

Defaults referred to in the reference tables are the underlying default values set by the VM2 workflow script generator.

Molecular system	Options	Description of resulting action
MOLSYSTEMTYPE	**protein+ligand**	Calculate protein-ligand complex binding free energies/energies
	host+ligand	Calculate host-guest complex binding free energies/energies
	ligand	Calculate ligand-only free energies/energies

Calculation type	Options	Description of resulting action
CALCNTYPE	**vm2**	2nd generation mining minima free energy calculation, which carries out an exhaustive conformational search, calculates configuration integrals for individual conformers, and carries out a Boltzmann weighting to give a total free energy.
	feprocess	Process a set of already generated molecular conformers for total free energy.
	confsearch	Carry out a conformational search providing potential energies of the resulting conformers.
	vm2ht1	High-throughput VM2 variant: 2 VM2 iterations with 100 searches per search type. Minimal accuracy loss when good-quality previously generated ligand poses/conformations are supplied.
	vm2ht2	High-throughput VM2 variant: 1 VM2 iteration with 100 searches per search type.
	vm2ht3	High-throughput VM2 variant: 1 VM2 iteration with 50 searches per search type.
LIGANDFEPROCESSOVERRIDE	**vm2**	Carry out a full VM2 conformational search for ligand-alone calculations when **CALCNTYPE** is **feprocess**. This is the **default**, since finding the lowest energy ligand-only conformation (which a **vm2** search will guarantee) is often advantageous for accuracy.
	none	The ligand-only calculation is **feprocess** (no searching) when **CALCNTYPE** is **feprocess**.
	vm2ht1	Carry out 2 VM2 iterations with 100 searches per search type for ligand-alone calculations when **CALCNTYPE** is **feprocess**.
	vm2ht2	Carry out 1 VM2 iteration with 100 searches per search type for ligand-alone calculations when **CALCNTYPE** is **feprocess**.
	vm2ht3	Carry out 1 VM2 iteration with 50 searches per search type for ligand-alone calculations when **CALCNTYPE** is **feprocess**.

Workflow exe, data, and project dirs	Options	Description of resulting action
VCWORKFLOWEXE	*/path*	Full path and name of python workflow exe. Default is **$VCHOME**/exe/VM2.pyc
VCRAWDATADIR	*/path*	Full path of directory containing input data such as: protein PDB file, ligand mol files, experimental data file, etc.
VCPROJECTDIR	*./name*	Project output directory to be used/created e.g. **$PWD**/output

Calculation resources	Options	Description of resulting action
VCCUSTOMEXE	*/path*	VeraChem calculation engine exe custom location: full path to directory. (Optional, rarely needed)
QUEUETYPE	**slurm**	Use a SLURM resource manager. This is the **default**.
	pbs	Use a PBS resource manager.
	bsh	Generate bash run script and run jobs sequentially on local machine.
QUEUENAME	*name*	Set the queue/partition *name* to use for pbs or slurm jobs. For your system's default queue/partition set as '**default**'.
QUEUENAMELIG	*name*	Set the queue/partition *name* to use for pbs or slurm jobs specifically for any ligand-only runs carried out in the workflow. Ligand-only calculations will often be faster than other calculations, and so running on queues with shorter time limits may make sense. Default is '**none**', where ligand-only runs use the **QUEUENAME** setting.
QUEUEDELAY	*real value*	Delay in seconds (decimal fractions allowed) between submission of each job to the queue system (resource manager). A small delay can avoid overwhelming the queue system when running the workflow on a very large number of ligands. Relevant for **QUEUETYPE** '**pbs**' and '**slurm**'. Default is **1.0**

		seconds.
NUMNODES	*n*	Number of compute nodes to use for receptor (protein or host) involved calculations. Default is **1**.
NUMMPIPROCS	*n*	Total number of MPI processes to use for receptor involved calculations. Default is **12**.
NUMOPENMPTHREADS	*n*	Number of OpenMP threads to use per MPI process. If set to 0 a pure MPI run is performed. Default is **0**.
NUMGPUS	*n*	Total number of GPUs to use for receptor-included runs. Default is **0**.
NUMLIGNODES	*n*	Number of compute nodes to use for ligand calculations. Default is **1**.
NUMLIGMPIPROCS	*n*	Total number of MPI processes to use for ligand calculations. Default is **12**.
NUMLIGOPENMPTHREADS	*n*	Number of OpenMP threads to use per MPI process for ligand calculations. If set to 0 a pure MPI run is performed. Default is **0**.
NUMLIGGPUS	*n*	Total number of GPUs to use for ligand calculations. GPU use for ligand calculations not recommended. Default is **0**.
MPIPROCPIN	**off**	Turn off Intel MPI process pinning (sets I_MPI_PIN = 0 in run scripts). This is the **default**.
	on	Turn on MPI pinning (I_MPI_PIN = 1)
PASSENVARS	**on**	User's full environment is passed along by the resource manager (e.g. slurm) to all jobs being run. (For slurm, this adds the sbatch option --export=ALL to the run scripts.) This is the **default**.
	off	User's full environment is not passed. (For slurm, this adds the sbatch option --export=NONE to the run scripts.)

Prep mode	Options	Description of resulting action
PREPMODE	**-prepMode**	The run script for the requested **QUEUETYPE** is placed in each run directory, but the calculations are not submitted. (Rarely needs to be set from this script as the workflow step scripts take care of this for the most common usages.)
	blank string	Given as: export **PREPMODE**=''. Calculations are submitted/run as normal. This is the **default**.

Protein setup	Options	Description of resulting action
PROTEINFILENAME	*name*.**pdb** or *name*.**prmtop** or *name*.**top**	Protein PDB (or .prmtop, .top) file *name* present in **$VCRAWDATADIR**/protein. Provide full file name including extension. If a .pdb file name is given here **PROTEINPREP** and **PROTEINTYPER** (see below) must be set as ambertools. If a .prmtop file name is given this indicates that forcefield parameters contained in the .prmtop file will be used and -proteinPrep and -proteinTyper must be set as none, and a .pdb file associated with the generation of the .prmtop should also be supplied in the /protein directory plus an .inpcrd file if available. If a .top filename is given this indicates that previously generated VeraChem-style input files will be used directly and again -proteinPrep and -proteinTyper must be set as none. In addition associated .mol and .crd files must be supplied (starting coordinates will be taken from .crd)
PROTEINREFLIGAND	*name* or substring of *name*	Ligand *name* or substring of ligand *name* chosen from **$VCRAWDATADIR**/ligands, which will be included for protein active site geometry relaxations during setup - see **PROTEINOPTH** and **PROTEINRELAX** below. This ligand is referred to as the protein reference ligand. It can also be set to define the real/live atom cutout - see **LIVEREALREFCHOICE** = 'proteinreflig' below. If the ligands in **$VCRAWDATADIR**/ligands are not pre-positioned into a good starting poses/conformations and therefore **PREGENLIGCONFS** is set as 'snap' to superpose them on a related protein co-Xtal ligand (supplied in **$VCRAWDATADIR**/template_ligand) the reference ligand will also be superposed on the co-Xtal before any protein active site relaxation. Otherwise the ligand geometry supplied in **$VCRAWDATADIR**/ligand will be used. The ligand file name should be supplied (without file extension) or at least a substring that uniquely identifies the file name. NOTE: Setting a protein reference ligand is not mandatory, but is usually recommended if carrying out protein active site relaxation, and is required if using it to define the real/live atom set.
REFLIGPRESEARCH		Allows the protein reference ligand's conformation to be optimized in the presence of the frozen protein active site before it is used further. This can avoid cases of entanglement of the provided reference ligand starting geometry and the protein.

	none	No pre-optimization of the protein reference ligand is performed before it is used further. This is the **default**.
	geomopt	Perform a regular geometry optimization of the protein reference ligand in the presence of the frozen protein active site before it is used further.
	confsearch	Perform a short conformational search of the protein reference ligand in the presence of the frozen protein active site before it is used further.
	confsearch2	Perform a more thorough conformational search of the protein reference ligand in the presence of the frozen protein active site before it is used further. Useful when superposing the reference ligand on a template via Vmap/snap or when uncertain about the quality of the reference ligand pose.
PROTEINFORCEFIELD	**amber**	Currently only the default forcefield '**amber**' is available in the workflow.
PROTEINPREP	**ambertools**	Use AmberTools for basic protein preparation.
	none	Pre-prepared protein parameter/topology files must be supplied in **$VCRAWDATADIR**/protein. Namely, .prmtop, .inpcrd and .pdb files. An inpcrd file is optional and if one is not provided the workflow will create one based on the coordinate information in the pdb file. Also, the .prmtop file name must be set through **$PROTEINFILENAME** (see above) and the other files must share its base name.
PROTEINTYPER	**ambertools**	Use AmberTools for protein typing/parameter assignment.
	none	Pre-prepared protein parameter/topology files must be supplied in **$VCRAWDATADIR**/protein. See **PROTEINPREP** entry above.
AMBERFORCEFIELD	**ff99sb**	Choose the AMBER99SB forcefield parmeter set.
	ff14sb	Choose the AMBER14SB forcefield parameter set.
	none	No specific choice of Amber forcefeld. Used when pre-prepared Amber parameter/topology files supplied in **$VCRAWDATADIR**/protein. See **PROTEINPREP** entry above.
AMBERREDUCE	**off**	Do not use the Reduce program (part of AmberTools) during automated protein setup by the workflow. Leap will instead be used to add any missing hydrogens.
		Use the Reduce program to add hydrogens during automated protein setup. This is the

	on	**default**. Note that, especially for recent versions of AmberTools, if capping groups are present in your protein PDB, this may lead to atom and residue naming issues and inconsistent behavior between the 'reduce', 'pdb4amber', and 'tleap' programs. Reduce is only available in AmberTools <= version 25.
LIVEREALMETHOD		Choice of method to generate the live-real atom sets. The default is **reference**
	reference	Use the method specified by **LIVEREALREFCHOICE**, see below, to generate the live-real atom sets.
	feprocess	Instead of using the method specified by **LIVEREALREFCHOICE** an attempt is made to vmap/snap the whole ligand series onto **PROTIENREFLIGAND**, and then to run a rigid-protein feprocess calculation to generate starting conformations for the calculations. The lowest energy ligand conformation produced for each ligand in the series is then used to define the live-real atom sets.
	file	Instead of using the method specified by **LIVEREALREFCHOICE** directly specify using formatted file listing live-real atoms supplied either via **LIVEREALATOMSFILE** below or in the protein raw data directory with the tag '_live_real' i.e. /protein/xxx_live_real.txt
LIVEREALREFCHOICE		Choice of source of reference for distance based definition of the set of protein atoms to be included (real) in the VM2 calculations and included plus mobile (live), i.e. the real/live sets.
	proteinreflig	Use the ligand set by **PROTEINREFLIGAND**
	file	Indicates a file path to a reference molecule/structure to be used for real/live atom definition will be given by setting **LIVEREALREFFILE** - see below.
	dir	Requests that the union of all atomic coordinates of all MOL files present in **$VCRAWDATADIR**/live_real_ref (if it exists) be used as the reference. SDF files are not yet supported.
	template	Use the template provided in **$VCRAWDATADIR**/template_ligand as the reference.
LIVEREALREFFILE	*filename*	If **LIVEREALREFCHOICE**='file', the name of the file in the raw data directory **$VCRAWDATADIR**/live_real_ref to be set as the reference.
LIVEREALATOMSFILE		If **LIVEREALMETHOD** is set as '**file**' this variable defines the file to be used to specify the real and live sets.
	/path	Full path to a file in the VeraChem live/real atom listing format to be used to define the live and real sets..

	default	A file of the form xxx_live_real.txt must be supplied in the raw data **$VCRAWDATADIR**/protein directory.
REALATOMCUTOFF	*real value*	Cutoff distance to any **LIVEREALREFCHOICE** atom that defines which protein atoms are included in the VM2 calculations - residues will be completed. A suggested value to start at is **6.0** Angstroms.
LIVEATOMCUTOFF	*real value*	Cutoff distance to any **LIVEREALREFCHOICE** atom that defines which protein atoms are live i.e. which atoms are treated as mobile. A suggested value to start at is **4.0** Angstroms.
COMPLETELIVERES	**on/off**	If set as **on** residues that contain live atoms are made fully live i.e. the remaining non-live atoms in the residue are automatically added to the live atom list. Default is **off**.
MINLIVEFORCMPLT	*n*	The minimum number of live atoms in a residue for it to be completed. Default is **5**.
COMPLETESPECLIVERES		Option to choose specific residues to set as live. The residues are identified by residue count number.
	0	No specific residues are chosen to be set as live. This is the **default**.
	string of residue numbers	Choose specific residues to set as live. The residues are identified by residue count number. To set residues provide a string with the residue count numbers, e.g. **COMPLETESPECLIVERES**='45 88 145'.
PROTEINOPTH	**on/off**	If set as **on** as part of the protein setup optimize the positions of H atoms in the protein. Default is **on**.
PROTEINOPTHCUTOFF	*real value*	Cutoff distance to any **LIVEREALREFCHOICE** atom that defines which H atoms will be geometry optimized during the protein setup. The default is **12.0** Angstroms. (A size no less than **REALATOMCUTOFF** + **2.0** is recommended.)
PROTEINRELAX	**on/off**	If set as **on** as part of the protein setup optimize the positions of cutoff defined atoms (see just below) in the protein. Default is **off**.
PROTEINRELAXCUTOFF	*real value*	Cutoff distance to any **LIVEREALREFCHOICE** atom that defines which protein atoms will be optimized (relaxed) during the protein setup. The default is **6.0** Angstroms. (A size no less than **LIVEATOMCUTOFF** + **2.0** is recommended.)

Ligand setup	Options	Description of resulting action

LIGANDFORCEFIELD	**gaff**	Use the general AMBER force field (GAFF) for ligands.
	gaff2	Use the second generation of the general AMBER force filed (GAFF2) for ligands.
	openff	Use the most recent version of OpenFF available at the time of VM2 release for ligands. A specific version can be set by appending the version string (matching the openff parameter file name). For example, for parameter file openff-2.3.0.offxml, the setting would be **openff-2.3.0**.
LIGANDPREP	**ambertools**	Use AmberTools for basic ligand preparation.
	openff	Use openff-toolkit for ligand preparation.
	none	Pre-prepared ligand parameter/topology files must be supplied in **$VCRAWDATADIR**/ligands. Namely, .prmtop, .inpcrd, and .mol files for each ligand. An inpcrd file is optional and if one is not provided the workflow will create one based on the coordinate information in the mol file.
LIGANDTYPER	**ambertools**	Use AmberTools for ligand typing/parameter assignment.
	openff	Use openff-toolkit for ligand typing/parameter assignment.
	none	Pre-prepared ligand parameter/topology files must be supplied in **$VCRAWDATADIR**/ligands. Namely, .prmtop, .inpcrd, and .mol files for each ligand. An inpcrd file is optional and if one is not provided the workflow will create one based on the coordinate information in the mol file.
LIGAND2D3D	**vconf**	Generates **3D** ligand structures using VeraChem's Vconf software when supplied **2D** structures in MOL files or SD file in **$VCRAWDATADIR**/ligands.
	none	Set to '**none**' when all ligands in **$VCRAWDATADIR**/ligands are already reasonable **3D** structures.
LIGANDCHARGER	**vcharge**	Use VeraChem's Vcharge tool to generate partial atomic charges for ligands.
	am1bcc	Use the AmberTools am1bcc charging method.
	abcg2	Use the AmberTools ABCG2 charging method. (Requires AmberTools version 24+.) Only available when **LIGANDPREP** and **LIGANDTYPER** are set as '**ambertools**'.
	am1bcc-gnn	Use the OpenFF NAGL graph neural network model for am1bcc charge prediction. Only available when **LIGANDPREP** and **LIGANDTYPER** are set as '**openff**'. The most recent NAGL model available at the time of VM2 release is used by default. A specific version can

		be set by appending the version string (matching the NAGL model file name). For example, for model file openff-gnn-am1bcc-1.0.0.pt, the setting would be **am1bcc-gnn-1.0.0** or **openff-gnn-am1bcc-1.0.0**
	none	Pre-prepared ligand parameter/topology files must be supplied in **$VCRAWDATADIR**/ligands. Namely, .prmtop, .inpcrd, and .mol files for each ligand. An inpcrd file is optional and if one is not provided the workflow will create one based on the coordinate information in the mol file. (Use '**none**' to retain charges in a supplied pregenerated prmtop, i.e. when you do not wish to replace them with e.g. vcharge charges.)
LIGANDSQMOPT		SQM geometry optimization convergence criteria when using **am1bcc** or **abcg2** as the charge assignment method and **LIGANDTYPER**='ambertools'.
	default	Standard SQM geometry optimization convergence criteria. This is the **default**.
	loose	Relaxed (faster) SQM geometry optimization convergence criteria.
	singlepoint	No SQM geometry optimization is performed (fastest).
MAPLIGTOTEMPLATE		Method to map input ligand atoms to equivalent atoms in the supplied template ligand.
	vmap	Use VeraChem's Vmap tool to map input ligand atoms to the equivalent atoms in the supplied template in **$VCRAWDATADIR**/template_ligand. This allows ligand common scaffold matching and superposition.
	none	No mapping to the template structure atoms is carried out.
LIGSDFFROMINPCRD	**-ligsdffrominpcrd**	Turns on overwriting of SDF/MOL coordinates using .inpcrd coordinates. Only used if coordinates in the provided MOL and .inpcrd files differ and you wish to use those in the .incpcrd.
	blank string	Given as: export **LIGSDFFROMINPCRD**=' '. No overwriting of input SDF/MOL coordinates.
PREGENLIGCONFS		Method to pre-generate ligand conformations to be placed in binding site and seed a free energy calculation.
	snap	Forces superposition of ligand atoms onto those in common in the template ligand in **$VCRAWDATADIR**/template_ligand. Requires that **MAPLIGTOTEMPLATE**='vmap'
	random	Randomly orientated ligand conformers generated for placement at center of geometry of template ligand.

	snap+random	Use **snap** as the default behavior, but if mapping has failed to find more than 3 common atoms with the template for a given ligand, switch to **random** for that ligand. Useful when the ligand set is mixed between congeneric and non-congeneric with the template. Requires that **MAPLIGTOTEMPLATE**='vmap'
	none	No pre-generation of ligand conformers is carried out. It is assumed that the ligands have been provided in reasonable bound conformations, e.g. from docking.
USETEMPLATESDF	**-usetemplatesdf**	If **PROTEINREFLIGAND** is specified, use its coordinates 'snapped' to the template ligand as a new template for all other maps and snaps, replacing use of the original template ligand.
	blank string	Given as: export **USETEMPLATESDF**=' '. Use the template in **$VCRAWDATADIR**/template_ligand for maps and snaps.

Calculation engine control	**Options**	**Description of resulting action**
MAXVM2ITERS	*n*	Maximum number of VM2 iterations for receptor-involved systems (e.g. protein, protein-ligand, host, host-guest). Each iteration carries out a conformational search according to the settings below, processes the conformations for free energy, and checks for convergence. The default is **60**.
CONFSEARCHSTYLE		Controls the conformational search style for receptor involved calculations, e.g. protein, protein-ligand, host, and host-guest.
	enhanced	Single mode-distort-minimze and random pair-mode-distort minimize conformational searches are carried out for receptor-involved VM2 calculations e.g. protein, protein-ligand, host, and host-guest.
	rigorous	Adds ligand and protein focused mode searching to '**enhanced**'. This is the **default**.
	vrigorous	Further adds ligand rotation/translation searching.
MAXCONFSEARCHES	*n*	Maximum number of conformational searches per search-type (e.g. single-mode, pair-mode) per VM2 iteration for receptor-involved VM2 calculations e.g. protein, protein-ligand, host, and host-guest. The default is **100**.
		Maximum number of VM2 iterations for ligand-only calculations. Each

MAXLIGANDVM2ITERS	*n*	iteration carries out a ligand conformational search according to the settings below, processes the conformations for free energy, and checks for convergence. The default is **60**.
LIGCONFSEARCHSTYLE		Controls the conformational search style for ligand only calculations.
	minimal	Single mode-distort-minimize conformational searches are carried out.
	enhanced	Single mode-distort-minimze and random pair-mode-distort minimize conformational searches are carried out. This is the **default**.
	rigorous	Adds focused mode searching to '**enhanced**'.
MAXLIGCONFSEARCHES		Maximum number of conformational searches per search-type (e.g. single-mode, pair-mode) per VM2 iteration for ligand-only VM2 calculations. The default is **100**.
PROTEINMODESEARCH		For protein and protein+ligand systems specify what mode search drivers will be included in mode-based conformational searches. Note: this setting only controls which modes are included for distortion - ALL mobile protein atoms are ALWAYS geometry optimized after distortions are carried out. For example, the backbone can relax in response to sidechain distortions.
	sidechains	Only mode search drivers without significant protein backbone character are included in mode-based conformational searches for protein and protein+ligand systems. This is the **default**.
	backbone	Only mode search drivers with significant protein backbone character and without significant sidechain character are included in mode-based conformational searches.
	sidechains+backbone	Both sidechain and backbone mode search drivers are included in mode-based conformational searches.
MPISEARCHSTYLE		Controls how the MPI-distributed conformational search behaves.
	uncoupled	All MPI processes carry out independent searches and only communicate their lowest energy conformers at the end of each search-type (i.e. single mode, random mode pairs, etc.) This is the **default**.

	coupled	Each MPI process communicates its lowest energy conformer following every search and the lowest energy coordinates are then used by all processes as a new search basis.
	mixed	Uncoupled searches are performed on even VM2 iterations and coupled searches on odd ones.
MPIMAXUCPLDITERS	*n*	Maximum number of iterations with '**uncoupled'** or '**mixed**' set. After it is reached the search style is set to '**coupled**'. For **$MPISEARCHSTYLE** = '**uncoupled**' a value of **10** is recommended, otherwise **20** is recommended for '**mixed**'. Switching to coupled following **MPIMAXUCPLDITERS** tends to drive the search to convergence more quickly.

Calculation engine spatial boundary conformer rejection	**Options**	**Description of resulting action**
LIGANDBOXSIZE		Controls rejection of ligand conformers when the center of coordinates of the ligand moves beyond a spherical boundary centered on the ligand's initial position.
	default	A reasonable spherical boundary radius is set automatically based on the molecular system and requested conformational search level. For host-guest this sets the option as '**none**'; for protein-ligand it sets **1.0** Angs. for '**enhanced**' conformational searching, **2.0** Angs. for '**rigorous**', and **4.0** Angs. for '**vrigorous**'. This is the **default**.
	real value	Radius in Angstroms of the sphere that the ligand center of coordinates must remain within. If the ligand center of coordinates moves outside this spherical 'box' the conformer is rejected.
	none	The spatial boundary conformer rejection mechanism is turned off.

Calculation engine solvation	Options	Description of resulting action
DIELECINTERNAL	*real value*	Internal dielectric used for GB and PBSA continuum solvation models. Default is **1.0**
DIELECEXTERNAL	*real value*	External dielectric used for GB and PBSA continuum solvation models. Default is **80.0**

Calculation engine tether constraints	Options	Description of resulting action
TETHERATOMSSELECT		Select atoms to apply tether constraints to.
	water	Apply tether constraints to the oxygen atom of all explicit water.
	file	Apply tether constraints to the atoms specified in a formatted file to be supplied (xxx_tethered_atoms.txt) in the **$VCRAWDATADIR**/protein directory. (See above for format information)
	file+water	Apply tether constraints to the oxygen atoms of all explicit water as well as to the atoms specified in a formatted file (xxx_tethered_atoms.txt) placed in the raw data /protein directory. (See above for format information)
	none	No tether constraints will be set regardless of the settings of **TETHERWATERMTD** and **TETHERFILEMTD** - see below.
TETHERWATERMTD		Tether constraint method applied to water oxygen atom.
	flatbottom	Use flat bottomed well (polynomial) constraints.
	harmonic	Use harmonic constraints.
	none	Do not constrain.

TETHERWATERFC	*real value*	Water polynomial/harmonic tether force constant. (Default is 2.0)
TETHERWATERDIST	*real value*	Water polynomial tether distance (Default 0.5 Angs.) Only applicable if flatbottom is selected for **TETHERWATERMTD**.
TETHERWATERORDER	*real value*	Water polynomial tether order (Default = 12.0) Only applicable if flatbottom is selected for **TETHERWATERMTD**.
TETHERFILEMTD		Tether constraint method applied to atoms specified using the 'file' option above.
	flatbottom	Use a flat bottomed well (polynomial) constraints.
	harmonic	Use harmonic constraints.
	none	Do not constrain.
TETHERFILEFC	*real value*	Polynomial/harmonic tether force constant for atoms specified using the 'file' option (Default is 2.0)
TETHERFILEDIST	*real value*	Polynomial tether distance for atoms specified using the 'file' option (Default 0.5 Angs.) Only applicable if flatbottom is selected for **TETHERFILEMTD**.
TETHERFILEORDER	*real value*	Polynomial tether order for atoms specified using the 'file' option (Default = 12.0) Only applicable if flatbottom is selected for **TETHERFILEMTD**.

Experimental data	Options	Description of resulting action
HAVEEXPT	**yes**	Requests that experimental data given in **$VCRAWDATADIR**/experiment be included in results tables for comparison. (Internally sets **EXPERIMENT**='-experiment'.)
	no	No experimental data is included in results tables. This is the **default**. (Internally sets **EXPERIMENT**='')

Results extraction	Options	Description of resulting action
EXTRACTFOR		Controls which conformer structure formatted files are generated and placed in the results directory.
	humans	In addition to generating structure files containing **MAXCONFSTOEX** conformers for each calculation (protein, ligands alone, complexes) also generate files containing only the lowest free energy conformer. This extra file generation can make manual viewing more convenient, but may not be well supported by tools expecting only single conformer files for each calculation (such as when using the VM2 plugin for MOE).
	moe	Generate structure files containing **MAXCONFSTOEX** conformers for each calculation (protein, ligands alone, complexes) . This is the **default**.
MAXCONFSTOEX	*n*	Maximum number of conformers per complex, ligand, and receptor calculation to extract from the run directories to the results directory. The default is **10**.
POSTPROCCONFLMIT	*n*	Maximum number of conformers to include in any post-processing. The default is **10**.