# CReM: practical structure generation and optimization

The tutorial will cover the following tools: CReM, CReM-dock and CReM-pharm. It will also introduce EasyDock, which automates docking and is also a backbone of CReM-dock.

Links to individual projects:

CReM - <https://github.com/DrrDom/crem>

EasyDock - <https://github.com/ci-lab-cz/easydock>

CReM-dock - <https://github.com/ci-lab-cz/crem-dock>

CReM-pharm - <https://github.com/ci-lab-cz/crem-pharm>

Contents

[CReM: practical structure generation and optimization 1](#_Toc207357545)

[Installation and set up 1](#_Toc207357546)

[Tutorial 6](#_Toc207357547)

[Low level CReM operations 6](#_Toc207357548)

[CReM-pharm 6](#_Toc207357549)

[CReM-dock 7](#_Toc207357550)

[De novo generation guided by molecular docking (no restrictions) 7](#_Toc207357551)

[Fragment expansion (PLIF restriction) 8](#_Toc207357552)

# Installation and set up

**Installation of prerequisites**

For Windows:

Install git - <https://git-scm.com/downloads/win>

**Tutorial directory preparation**

1. Clone or download the repository from <https://github.com/rdkit/UGM_2025/tree/main/workshops/crem_tutorial>.

git clone https://github.com/rdkit/UGM\_2025.git

2. Download CReM fragment database compiled from molecules from ChEMBL33 filtered by SA score <= 2 (<https://zenodo.org/records/16909329/files/chembl33_sa2_f5.db.gz?download=1>) and place it to the directory crem\_tutorial/cremdb.

More precompiled CReM databases can be found here - <https://qsar4u.com/pages/crem.php>. There are not necessary to the tutorial.

3. The structure of the tutorial directory should look like this:

crem\_tutorial

├── 3ral

│   ├── 3ral\_ligand\_core.sdf

│   ├── 3ral\_ligand.pdb

│   ├── 3ral\_ligand.sdf

│   ├── 3ral.pdb

│   ├── 3ral.pdbqt

│   ├── config\_gnina.yml

│   ├── config\_linux.yml

│   ├── config\_win.yml

│   ├── config.yml

│   ├── grid.txt

│   └── pharm

│   ├── 3ral\_full.xyz

│   └── 3ral.xyz

├── cremdb

│   └── chembl33\_sa2\_f5.db

├── frags

│   ├── all\_frags.smi

│   ├── large\_set.dat

│   ├── large\_set.dir

│   ├── large\_set.smi

│   └── small\_set.smi

├── pymol

│   └── visualize\_pharms.py

├── CReM\_Polishchuk\_2024.pdf

├── CReM\_Polishchuk\_2024.pptx

├── CReM\_tutorial\_instructons.docx

├── env\_linux.yml

└── env\_win.yml

**Installation of a conda environment**

Install conda/miniconda if not available - <https://www.anaconda.com/docs/getting-started/miniconda/install>.

There are two options to install the environment: from an yml-file or step by step installation.

1. From an yml-file (recommended).

On Linux/MascOS:

conda env create -n crem -f env\_linux.yml

On Windows:

conda env create -n crem -f env\_win.yml

2. Step by step installation

# create and active env

conda create -n easydock

conda activate easydock

# install easydock

conda install -c conda-forge python=3.11 rdkit numpy==1.26

conda install -c conda-forge scipy dask distributed scikit-learn notebook sqlite

pip install paramiko prolif

pip install git+https://github.com/forlilab/Meeko.git@develop

pip install gemmi

pip install git+https://github.com/ci-lab-cz/easydock.git

# for Linux and MacOS

pip install vina

# for MacOS only

conda install -c conda-forge boost swig

# install MolGpKa

pip install git+https://github.com/ci-lab-cz/MolGpKa.git

pip install torch\_geometric

pip install torch==2.4.1 torchvision torchaudio --index-url https://download.pytorch.org/whl/cpu

pip install pyg\_lib torch\_scatter torch\_sparse torch\_cluster torch\_spline\_conv -f https://data.pyg.org/whl/torch-2.4.1+cpu.html

# install CReM packages

pip install crem

pip install git+https://github.com/ci-lab-cz/crem-dock.git

# install crem-pharm and its dependencies

conda install -c conda-forge openbabel networkx=3.3

pip install pmapper

pip install git+https://github.com/meddwl/psearch.git@crempharm

pip install git+https://github.com/ci-lab-cz/crem-pharm.git

**Installation of other dependencies**

For Windows only, there is a need to install a binary of AutoDock Vina - <https://vina.scripps.edu/downloads/>.

Linux/MacOS users may also install it and run, but on these platforms we will use Vina python package, which was installed within the conda environment.

CDPKit

Install CDPKit by downloading the latest release - <https://github.com/molinfo-vienna/CDPKit/releases>.

Linux (example)

sh CDPKit-1.2.3-Linux-x86\_64.sh --prefix=/home/XXX/opt

Set PATH environment variable to include CDPKit binaries

On Linux:

add CDPKit/Bin directory to your $PATH variable in .bash\_profile

Example, substitute with a real path:

echo ‘PATH=/path/to/CDPKit/Bin:$PATH’ >> .bash\_profile

To activate environment variables, restart a session or invoke

source .bash\_profile

On MacOS (example, substitute with a real path):

echo 'PATH=/path/to/CDPKit/Bin:$PATH' >> .zshrc

On Windows:

1. Press **Win + R**, type sysdm.cpl, and press Enter.
2. Go to Advanced → Environment Variables.
3. Under **User variables** (or **System variables** if you want it for all users), select **Path** and click **Edit**.
4. Click **New**, then paste in your custom path (e.g. C:\my\custom\path).
5. Click **OK** on all dialogs.
6. Restart your terminal (or log out/in).

**Checking the installations**

1. To check that main programs were installed correctly and PATH variable was properly configured you may run the following commands (in the case of success, they will print help messages):

conda activate crem

cremdock -h

crempharm -h

# Linux

confgen –h

# Windows

confgen.exe -h

2. Check installation of torch

conda activate crem

python -c "import torch"

In the case of success, it will print nothing, otherwise an error message will be printed

**Optional installations**

There are needed for visualization and analysis of the outputs. All these steps we will do together, therefore their installation is not strictly required. However, you may find these tools useful for your work.

PyMOL (<https://www.pymol.org/>)

<https://pymolwiki.org/index.php/Linux_Install>

<https://pymolwiki.org/index.php/MAC_Install>

<https://pymolwiki.org/index.php/Windows_Install>

SQLite browser - <https://sqlitebrowser.org/>

DataWarrior - <https://openmolecules.org/datawarrior/download.html>

# Tutorial

The major target for the whole tutorial will be the complex of CDK2 with its inhibitor (PDB: 3RAL).

**Note**: all commands were written to be from the directory crem\_tutorial. If you change the directory, you have to adjust all paths, if they are not full absolute ones.

Activate the installed conda environment before executing further commands in the tutorial

conda activate crem

## Low level CReM operations

This tutorial will show low-level operations which makes CReM highly flexible and able to address different tasks.

On Windows run (mini)conda console.

cd path/to/crem\_tutorial

conda activate crem

jupytrer-notebook

## CReM-pharm

**Preparation**

1. Augment CReM DB with pharmacophore features (do not run, just an example):

crempharm\_add\_pmapper -i crem.db -c 10 -v

2. Create a database of starting fragments (do not run, because the database was already prepared, however this is fast)

gen\_db -i frags/large\_set.smi -o frags/large\_set.dat -n 10 -r 0.1 -c 10 -v

**De novo compounds generation** using a 3D pharmacophore model

crempharm --query 3ral/pharm/3ral\_full.xyz --fragments frags/large\_set.dat -o crem-pharm --ids 2 5 6 --clustering\_threshold 3 --nconf 5 --conf\_gen cdpkit --dist 1.5 --exclusion\_volume 2.4 --db cremdb/chembl33\_sa2\_f5.db --radius 5 --mw 450 --tpsa 120 --rtb 7 --logp 4 -w 3 -c 2 --log crem-pharm/out.log

Analysis

1. Switch to the directory

cd crem-pharm

2. Get SDF of de novo designed structures which matched at least 6 pharmacophore centers

get\_sdf\_from\_dock\_db -i res.db -o res.sdf -f --add\_sql "matched\_ids\_count >= 6" --fields matched\_ids\_count

3. Get SMILES of de novo designed structures which matched at least 6 pharmacophore centers

get\_sdf\_from\_dock\_db -i res.db -o res.smi -f --add\_sql "matched\_ids\_count >= 6" --fields id matched\_ids\_count

4. Docking of generated structures:

Before run docking you have to adjust paths in corresponding config.yml files which are stored in the directory crem\_tutorial/3ral. These file store settings for docking using EasyDock. Insert absolute paths to protein PDBQT file, grid box and a binary of a docking program (on Windows).

Run docking with EasyDock:

Linux/MacOS:

easydock -i res.smi -o res\_dock.db --program vina --config ../3ral/config\_vina.yml --protonation molgpka -c 3 -v

Windows:

easydock -i res.smi -o res\_dock.db --program qvina --config ../3ral/config\_win.yml --protonation molgpka -c 3 -v

5. Extract SDF of docked molecules

get\_sdf\_from\_dock\_db -i res\_dock.db -o res\_dock.sdf --fields docking\_score

6. Extract SMILES and docking scores

get\_sdf\_from\_dock\_db -i res\_dock.db -o res\_dock.smi --fields id docking\_score

7. Compare poses from de novo generation and after docking

## CReM-dock

### De novo generation guided by molecular docking (no restrictions)

1. Run from the tutorial root directory cd ..

Linux/MacOS:

cremdock -i frags/small\_set.smi -o crem-dock/denovo/1.db --search 2 --ntop 1 --nclust 5 -d cremdb/chembl33\_sa2\_f5.db --radius 5 --max\_replacements 3 --mw 450 --tpsa 120 --rtb 7 --logp 4 --protonation molgpka --program vina --config 3ral/config\_vina.yml -c 2

Windows:

cremdock -i frags/small\_set.smi -o crem-dock/denovo/1.db --search 2 --ntop 1 --nclust 5 -d cremdb/chembl33\_sa2\_f5.db --radius 5 --max\_replacements 3 --mw 450 --tpsa 120 --rtb 7 --logp 4 --protonation molgpka --program qvina --config 3ral/config\_win.yml -c 2

2. Change direcrtory

cd crem-dock/denovo

3. Get SDF with top docking poses and scores

get\_sdf\_from\_dock\_db -i 1.db -o 1.sdf --fields docking\_score

4. Get SMILES with molecule ids and docking scores

get\_sdf\_from\_dock\_db -i 1.db -o 1.smi --fields id docking\_score

5. Compare structures and docking poses with the poses from crem-pharm.

### Fragment expansion (PLIF restriction)

1. Get PLIF for the reference ligand

cd ../../3ral

cremdock\_plif -p 3ral.pdb -l 3ral\_ligand\_core.sdf -o 3ral\_ligand\_core.plif

2. Run fragment expansion

# return to the tutorial root directory

cd ..

On Linux/MacOS:

cremdock -i 3ral/3ral\_ligand\_core.sdf -o crem-dock/fragment\_expansion/2.db --search 2 --ntop 1 --nclust 3 -d cremdb/chembl33\_sa2\_f5.db --radius 4 --max\_replacements 50 --mw 450 --tpsa 120 --rtb 7 --logp 4 --protonation molgpka --program vina --config 3ral/config\_vina.yml -c 2 --plif glu81.a.hbdonor leu83.a.hbacceptor --plif\_cutoff 0.5 --plif\_protein 3ral/3ral.pdb

On Windows:

cremdock -i 3ral/3ral\_ligand\_core.sdf -o crem-dock/fragment\_expansion/2.db --search 2 --ntop 1 --nclust 3 -d cremdb/chembl33\_sa2\_f5.db --radius 4 --max\_replacements 50 --mw 450 --tpsa 120 --rtb 7 --logp 4 --protonation molgpka --program qvina --config 3ral/config\_win.yml -c 2 --plif glu81.a.hbdonor leu83.a.hbacceptor --plif\_cutoff 0.5 --plif\_protein 3ral/3ral.pdb

3. Change direcrtory

cd crem-dock/fragment\_expansion

4. Get SDF with top docking poses and scores

get\_sdf\_from\_dock\_db -i 2.db -o 2.sdf --fields docking\_score

5. Get SMILES with molecule ids and docking scores

get\_sdf\_from\_dock\_db -i 2.db -o 2.smi --fields id docking\_score

5. Compare structures and docking poses with the poses from crem-pharm.