The software suite consists of 3 components:

1) Preprocessor
2) Simulator
3) Postprocessor

**1) Preprocessor:**

The preprocessor currently contains a code to determine the initiation rate constant from the spectral irradiance of the curing light source and the absorption spectrum of the initiator molecule. Both of the information is expected to be available in two .csv files, with wavelength in the first column and spectral (absorption) intensity in the second. The initiation rate can be computed as:

$$ \text{python getInitiationRate.py -I <light_source_file> -a <absorption_spectrum_file>} $$

**2) Simulator:**

The simulator contains the codes for reaction diffusion simulation and computing the radial distribution functions and effective Green’s functions of disordered networks. The arguments for the script `virtual_curing.py` are:

Options:

- `-h`, `--help` show this help message and exit
- `-i INPUTFILE` Input filename for a photopolymerization simulation.
- `-n SAMPLE` Number of stochastic trajectories to be simulated.
- `-v VERBOSE` Verbosity of display outputs.

The command for starting a new simulation is:

$$ \text{python virtual_curing.py -i <input_filename>.txt -n 1 -v 0} $$

Without the option `-v 0` the code will print the degree of conversion and other outputs to stdout after every transformation.

Below we list the contents of the hierarchy of input files.

**<input_filename>.txt:**

Case folder = <Folder containing the following 4 text files>
Monomers inputs = monomers_inputs.txt
Initiator inputs = initiator_inputs.txt
Curing inputs = curing_inputs.txt
Simulation inputs = simulation_inputs.txt
monomers_inputs.txt:

# Polymerization parameters
Cyclization (or cross-linking) factor = 1.0
Mean neighboring functional groups = 10
Interaction range = 2
Site occupancy fraction = 0.7
Monomer types = 1
Monomer 1: Mole fraction = 0.5
Monomer 1: Functionality = 1
Monomer 2: Mole fraction = 0.5
Monomer 2: Functionality = 2
Rate constants file = propagation_rate_constant_matrix.txt
Termination constants file = termination_rate_constant_matrix.txt

# Monomer properties
Monomer 1: Molecular volume (in Angstroms^3) = 500
Monomer 1: Molecular weight (in Da) = 500
Monomer 2: Molecular volume (in Angstroms^3) = 400
Monomer 2: Molecular weight (in Da) = 300
Viscosity (in Pa.s) = 1.0
Density (in g/cm^3) = 1.25

# Bond and interaction energies
Bond length stiffness (in eV) = 0.1295
Monomer 1: Self interaction energy (in eV) = 0.259
Monomer 2: Self interaction energy (in eV) = 0.0259
Mean interaction energy with initiators (in eV) = 0.1293
Mean interaction energy with coinitiators (in eV) = 0.1293

propagation_rate_constant_matrix.txt:

100.0 100.0
100.0 100.0

termination_rate_constant_matrix.txt:

10000.0 10000.0
10000.0 10000.0
**initiator_inputs.txt:**

Initiation type (unimolecular or bimolecular) = unimolecular
Percentage of initiators (mole fraction) = 1.0

# The following inputs are only for a bimolecular initiation system
Percentage of co-initiators (mole fraction) = 1.0
Initiator volume (in A^3) = 100
Co-initiator volume (in A^3) = 100
Initiator surface area (in A^2) = 100
Co-initiator surface area (in A^2) = 100
Initiator molecular weight = 100
Co-initiator molecular weight = 100
Excitation lifetime of the initiator = 1E-6

**curing_inputs.txt:**

Lighting mode (Linear ramp or Pulse) = Linear ramp
Excitation rate constant = 0.1
Curing time (in seconds) = 10
Temperature (in K) = 300

# Additional input for linear ramp
Ramp time (in seconds) = 0

# Additional inputs for pulse
Pulse width (in seconds) = 0.0
Pulse gap (in seconds) = 0.0

**simulation_inputs.txt:**

Lattice half size = 50
Simulation time (in s) = 1000
Output step interval = 1000
Output casename = testcase
Write video output = No
3) **Postprocessor:**

The postprocessor folder contains scripts to compute the average of the stochastic reaction-diffusion trajectories.

```
$ python compute_all_stats.py -i <PATHNAME> -n <SAMPLES> -t <TIME>
```

- `<PATHNAME>` - The folder containing the stochastic trajectories.
- `<SAMPLES>` - Number of sample stochastic trajectories.
- `<TIME>` - Cutoff time in the trajectory.

The above script computes the average degree of conversion, network fluctuation rate curves, and the radial distribution functions.

For each stochastic trajectory, the coordinates and the connectivity of the polymer network nodes are saved. Separate lattice files also contain the type of monomer at each network site, and the fluctuation propensity of each network node. These files are equivalent to a virtual sample of the photocured network, which can be used for further analysis. Currently, we have scripts to determine the probability distribution of force constants from the network structure, which is then used to compute the effective phonon Green’s function of the network.

```
$ python compute_K_distribution.py -i <PATHNAME> -n <SAMPLES> -k <K_RATIO>
```

- `<PATHNAME>` - The folder containing the lattice polymer networks.
- `<SAMPLES>` - Number of sample networks.
- `<K_RATIO>` - Ratio between the force constants of bonded and non-bonded interaction.
Example Results

The degree of conversion (DC) and crosslink density ($\rho_{cl}$) are obtained during the simulation, as mole fractions of converted double-bonds, and doubly-polymerized monomers, respectively. Radial Distribution Functions ($g_2(r)$) at the end of simulation are useful to identify how spatial correlations are sensitive to polymerization protocols.

The network structure is used for determining the variability in effective force constants ($\rho(K)$) due to spatial heterogeneity. Force constants for bonded ($K_b$) and non-bonded force ($K_{nb}$) interactions are assumed. The distribution in force constant is subsequently used to compute phonon density of states ($g(\omega)$) and mean free path.