SuSi

StrUcture SImulation

Users Manual

SuSi Version 2.0.0
March, 2025

User Manual Version 1.0.0

I. **Disclaimer**

SuSi is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

SuSi is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with SuSi. If not, see <https://www.gnu.org/licenses/>.

Neither the names of the EEBE, the Universitat Politecnica de Catalunya, nor the names of any of the copyright holders of SuSi may be used to endorse or promote any products derived from this Software without specific, prior, written permission.

II. Acknowledgments

The version 2.0.0 has received support from the Grant PID2021-125257OB-I00, by MCIN/AEI/10.13039/501100011033 and by ERDF "A way of making Europe".

III. Contributions

Original Design:

- David Naranjo (Universitat Politècnica de Catalunya)
- Carlos Alemán (Universitat Politècnica de Catalunya)
- Juan Torras (Universitat Politècnica de Catalunya)

List of contributors (alphabetical order):

- Alemán, Carlos (Universitat Politècnica de Catalunya)
- Català, Montse (Universitat Politècnica Catalunya)
- Curcó, David (Universitat de Barcelona)
- Martí, Didac (Universitat Politècnica de Catalunya)
- Naranjo, David (Universitat Politècnica de Catalunya)
- Torras, Juan (Universitat Politècnica de Catalunya)

V. Standard Citations

Scientific papers and presentations incorporating results obtained using SuSi must

reference the code as follows:

"SuSi, Structure Simulation version 2.0, (2025) a software product of the Universitat Politècnia de Catalunya, D. Naranjo, C. Alemán, J. Torras, D. Martí, D. Curcó, and M. Català."

Contents

1. Introduction

This document is a complete manual for the use of **SuSi (StrUcture SImulation)**, a computer application programmed in Python to construct high-density amorphous polymer matrices.

The program aims to simplify the preparation of molecular dynamics simulations of polymer systems. Even though polymers are always present in different areas of industry, there is still much to learn about their behavior at the atomistic level. For this reason, programs that allow simulations of their molecular behavior can provide much information about their properties at both the molecular and atomic levels.

Polymer molecules can be composed of long chains made up of either single type of monomer or multiple types. They may also form mixtures containing chains of various lengths, branched, crosslinked, and other monomeric units. These characteristics make their simulation very complex, even with the aid of advanced computer programs.

One of the major challenges in atomistic simulations of complex polymeric systems is generating the initial structure. Unlike crystalline materials with well-defined, repetitive arrangements, many polymers exhibit amorphous, heterogeneous, and highly entangled structures. This lack of uniformity makes it difficult to construct realistic initial configurations that accurately capture the material's physical and chemical properties. Additionally, achieving proper chain packing, entanglement, and realistic density requires sophisticated additional work.

SuSi aims to facilitate the simulation of polymer matrices. Starting from an input file with the system's specifications, it allows the creation of a system formed by different polymer molecules within a 3D box. Starting from the definition of the box's size and the number of monomers in the system, an amorphous system with the required density can be obtained. The structure of each monomers is defined using a file in PREPI format, which is commonly used for specifying protein residues in the AMBER[1]). The program will create a file in PDB format containing the cartesian coordinates of all the atoms of the polymer structure.

Users must define the characteristics of the system (polymer matrix) in the input file that they want to build, following the guidelines outlined in the subsequent sections of the manual. In such file, it will be necessary to specify:

[1] **AMBER**: Information about the program and residues definition can be found at: http://ambermd.org

- System name (resulting PDB file) and size of the 3D periodic box to pack the system.
- Parameters for collision detection: minimum collision distance, retries to place a residue, and maximum percentage of deformation of a residue when a collision is detected.
- Definition of the system to be generated: Number of molecules, sequence of each molecule, and location of the files in PREPI format (monomers).
- In the case of crosslinked systems, it is also necessary to define the total number of crosslink molecules in the system, the distances between the first atom of the backbone of the crosslinking molecule and the starting atom found in a linear chain of the system; the distance between the last atom of the backbone of the crosslinking molecule and the closing atom found in a linear chain of the system. Also, the maximum angle in degrees a crosslink residue can be rotated to find a new closing atom.

1.1. An overview of how SuSi Works

The construction of linear chains is based on the successive placement of monomeric units for which the information on the position of their atoms is known by means of internal coordinates. The linear chains and their constituting monomers are processed individually, placing them within a space defined by a box in an iterative manner. Each time a monomer is placed, the collisions between other atoms previously placed in the system are analyzed. If no collisions are detected, the next monomer is processed.

When collisions are detected, the monomer's internal coordinates of the main chain are deformed, and the algorithm attempts to place the monomer again in the system. The number of attempts in this process depends on a user-defined retries parameter, when the attempts exceed the maximum number of retries, the preceding monomer is reprocessed. This reprocessing can be repeated even if the first residue of the chain has to be reprocessed. Nevertheless, when this occurs and if the attempts to place the first monomer exceed the maximum number of retries, the algorithm relocates it to another region within the box.

The iterative placing process works in an analogous way for the chains; however, if the number of attempts to place a chain exceeds the maximum number of retries, the previous chain is not reprocessed. Still, a system containing only the successfully placed chain is generated.

When crosslinking chains are specified, the program searches for atoms within a distance defined by the crosslink chain size. To maintain the consistency of the chemical bonds when growing a crosslink chain, the program also allows the removal of atoms defined by the user.

As a result, the program provides a PDB file with all the atoms in the system. In the case of systems with crosslinks, an additional file is obtained where the links between the atoms of the linear and the crosslink chains are specified in LEaP format (a component of the AMBER suite).

2. Installation and Requirements

2.1. Downloading the Program

To use SuSi, download the latest version from GitHub. The downloaded package contains:

- Python classes that define polymer structures and their assembly rules.
- Example files to help you get started.

Table 1 summarizes the content the users will download.

Table 1. Contents of the SuSi package.

Directory or file	Description
/test	A directory that contains input files with examples to build polymer matrices.
/prepis	A directory that contains monomers in PREPI format used in the provided examples.
/susi	A directory that contains Python the modules necessary to run SuSi: • __init__.py • atom.py • builder.py • collision.py • filePDB.py • filePREPI.py • files.py • int2car_M.py • messages.py • molecule.py • parserSusi.py • progressBar.py • residue.py • system.py
Susi.py	Python script to run the SuSi.
Readme.txt	File with brief instructions to use the program.
SuSiManual.pdf	Program usage manual.
COPYING	GNU General Public License 3 (GPL)

2.2. System Requirements

SuSi runs directly in the terminal of any Linux-based system and does not require installation. However, you must have Python installed (minimum version of 3.11.4) and verify it by running the following command in the terminal:

```
python --version
```

SuSi requires the NumPy library. If it is not installed, you can install it with:

```
pip install numpy
```

Once these requirements are met, users can start using SuSi to build polymeric Systems structures.

3. Data processing

SuSi reads a text file with a specific format where the characteristics of the system to be created are specified (input file), as well as the locations of the files to be read with structural information of all the monomers that make up the system (prepi files), and the output files.

Once the data has been interpreted and the input structural files have been read, the system construction process begins with its validations, and finally, if everything is correct, the desired system is produced in PDB format (Fig. 1).

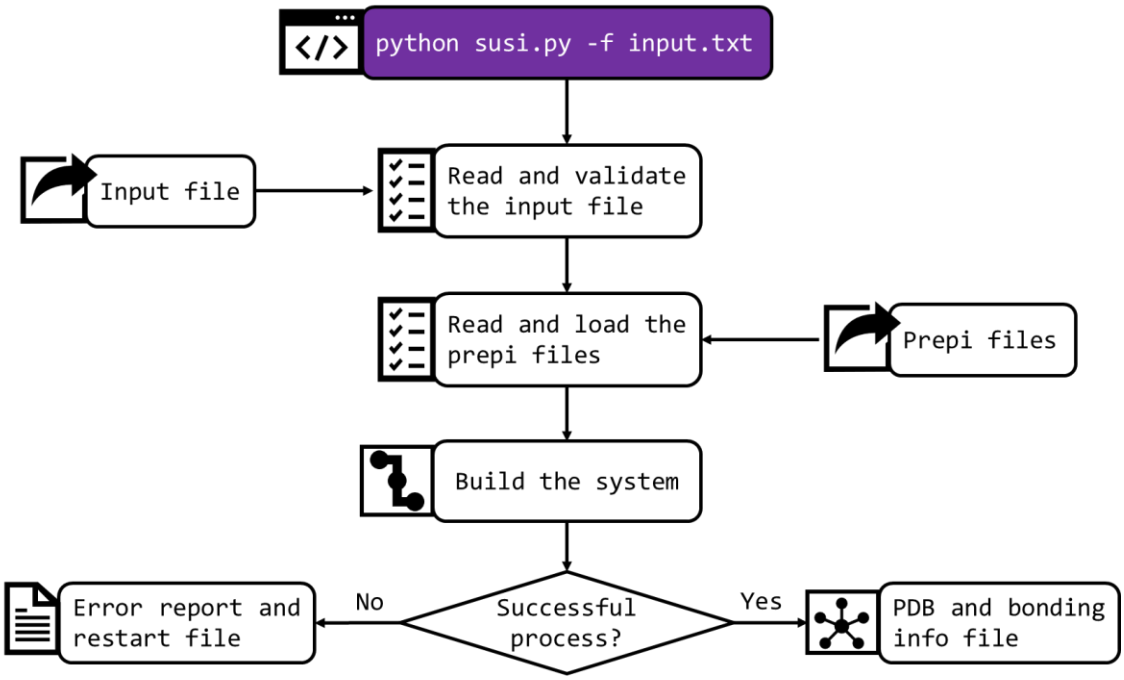

Fig. 1. Scheme of SuSi workflow.

3.1. Input data

The basic input data to build a system are:

- All the monomers in PREPI format.
- An input file with the information of the system.

3.1.1. Structural file, PREPI

Polymeric chains are constructed using the monomer coordinates provided in an AMBER prep file format in internal coordinates or PREPI[2]. The PREPI file (Fig. 2) contains essential

[2] **Prep format:** https://ambermd.org/doc/prep.html

information about the monomer's internal coordinates, including atom types, charges, and connectivity parameters, which are crucial for defining the molecular geometry.

```
    0    0    1

   Polyethylene body monomer
PEB
PEB    INT  0
CORRECT     OMIT DU   BEG
  0.0000
   1  DUMM  DU    M    0  -1  -2     0.000      .0        .0      .00000
   2  DUMM  DU    M    1   0  -1     1.529      .0        .0      .00000
   3  DUMM  DU    M    2   1   0     1.529   109.461      .0      .00000
   4  C1    c3    M    3   2   1     1.529   109.465   178.348 -0.041182
   5  H2    hc    E    4   3   2     1.088   108.887    62.420  0.007304
   6  H3    hc    E    4   3   2     1.089   108.382   -62.420  0.020252
   7  C2    c3    M    4   3   2     1.529   109.460   178.348 -0.023226
   8  H4    hc    E    7   4   3     1.088   108.887    62.420  0.027114
   9  H5    hc    E    7   4   3     1.089   108.382   -62.420  0.009739


LOOP

IMPROPER

DONE
STOP
```

Fig. 2. Example of a PREPI file of the polyethylene monomer.

The first lines contain general information used in LEaP[3] when processing the structure generated by SuSi. The key parameters used by SuSi are in the columns as follows:

- Column 1: Index of each atom of the residue.
- Column 2: Unique identifier name of each atom of the residue (except for DUMMY atoms).
- Column 3: Atom-type symbol defining the atomic interactions within the force field.
- Column 4: Topological type for each atom. A complete description of each type is given on the Amber official site. For SuSi, the most important is the M-type atoms, which define the backbone of the polymeric chain.
- Column 5: Atom index to which the current atom is bonded.
- Column 6: Atom index to which the current atom forms a plane angle along with the atom specified in column 5.
- Column 7: Atom index to which the current atom forms a dihedral angle along with the atoms specified in columns 5 and 6.
- Column 8: Value of the bond distance between two atoms
- Column 9: Value of the planar angle between three atoms.
- Column 10: Value of the dihedral angle between four atoms.
- Column 11: Value of the partial atomic charge of the current atom (not used in SuSi).

[3] **Fundamentals of LEaP:** https://ambermd.org/tutorials/pengfei/index.php

Additionally, the PREPI file includes atoms of type DUMM (dummy atoms). These are virtual atoms used to maintain internal coordinate consistency during modeling. DUMM atoms help define geometry but are not part of the physical structure and have no mass or charge.

In constructing a linear polymer chain with repeating monomer units, it is essential to define three specific prepolymer units: a head, body, and tail. Each of these prepolymer units serves distinct roles in the chain's initiation, propagation, and termination, ensuring consistent molecular architecture throughout the polymer.

For instance, in the case of polyethylene (Fig. 3), the head unit consists of two carbon atoms (C1, C2) and five hydrogen atoms (H1-H5). The body unit, responsible for the chain propagation, contains two carbon atoms (C1, C2) and four hydrogen atoms (H2-H5). Finally, the tail unit terminates the chain and includes two carbon atoms (C1, C2) and five hydrogen atoms (H2-H6). This specific configuration of head, body, and tail monomer units ensures that the carbon atoms maintain proper saturation, thereby preserving the structural integrity of the polyethylene chain.

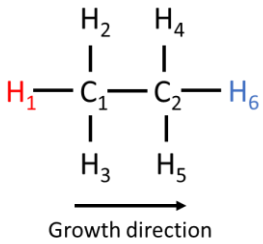

Fig. 3. Scheme of the polyethylene residue with different hydrogen atoms depending on the type. (Head H1 to H5, Body H2 to H5, Tail H2 to H6) Scheme of the polyethylene residue with different hydrogen atoms depending on the type. (Head H1 to H5, Body H2 to H5, Tail H2 to H6).

3.1.2. Input file

The user sets the input parameters in a text file. These parameters are defined line by line. A system with linear chains only requires parameters that define the dimensions of the systems, the parameters for collision detection, and the retries and deformation threshold. On the other hand, systems with crosslinks require additional parameters, including the starting and closing atoms and the bonding distances between the crosslink and a molecule. Each of the parameters required in the input file is as explained below.

- **System name**: This is a short text with the name of the system and the resulting file.

Example. Build a system where the resulting file will be PolyethyleneBox.pdb.

```
system_name,PolyethyleneBox
```

Warning: If the user sets the system name to be the same as a file that already exists, SuSi will process the system, but if it finishes successfully, it will not overwrite the existing file, and the information will be lost.

- **Box:** The dimensions of the box are specified as Lx n A, Ly n A, Lz n A, where n is the value of the dimension in each of the three dimensions. Currently, the program only allows the introduction of values in angstroms.

  Example. Build a system within an orthogonal prism box of 35x50x75.5 A^3

  ```
  box,Lx 35 A,Ly 50 A,Lz 75.5 A
  ```

- **Periodical coordinates:** The user can generate the systems inside a periodical simulation box by specifying Y in this parameter or N otherwise.

  When Y is specified, the entire system is created within the periodic box with a discontinuous representation of the molecules while keeping the imposed periodicity conditions. On the other hand, when N is specified, despite considering the periodic boundary conditions, the system is created without discontinuities in the molecules representing the periodic images outside the unit cell.

  Example. Build a system in a non-periodical box.

  ```
  periodical_coordinates,N
  ```

- **Retries:** Here, the maximum number of retries to attempt placing a residue and a molecule inside the box is specified. Since the attempts to place the residues and molecules follow a cascading pattern, the higher the retries parameter, the higher the number of potential attempts (loop execution) will be carried out. Thus, for a system with M molecules, with r_i residues and with R maximum retries, the theoretical maximum number of attempts, N_{max}, will be:

$$N_{max} = R^2 \sum_{i=1}^{M} r_i$$

  In this regard, the user must choose a retries parameter that balances processing time and effectiveness in placing residues. Normally, denser systems require a higher number of retries than less dense systems.

Example. Set the program to make a maximum of 10 attempts to place each residue and each molecule recursively.

```
retries, 10
```

- **Minimum distance:** This parameter specifies the minimum distance in angstroms between two atoms in the system before a collision is detected. All atoms that are closer than this value will be detected as colliders

  Example. Set that any atom located at a distance less than 1.5 Å of another will be detected as a collision in the system.

  ```
  minimum_distance, 1.5 A
  ```

- **Maximum deviation:** This parameter represents the maximum percentage deviation that can be made to the internal coordinates of a residue when having to deform it due to a collision. Then, when deforming a residue, the values of the bond length, planar angle, and dihedral angle will be multiplied by a random factor r ~ Uniform(- Dev_{max}, +Dev_{max}), where Dev_{max} represents the maximum deviation as a unitary factor.

  Example. Set that when a collision is detected, the program will introduce a random deformation in the structural parameters of the residue within ± 10% of their original values.

  ```
  maximum_deviation, 10
  ```

- **PREPIs path:** Directory where all the PREPI files used in the system are located.

  Example. Set the program to look for all the PREPI files in the directory prepis/system1. The user should be aware that the location of this directory is relative to the location of SuSi. For example, if SuSi is located in the home directory, the program will look for the PREPIs in the directory ~/susi/prepis/system1.

  ```
  path_prepi,prepis/system1
  ```

- **Number of molecules**: A number representing the total number of linear chains in the system.

  Example. Create a system with four linear chains.

  ```
  molecules_number,4
  ```

- **Molecule lines definition:** Each linear chain must be defined on a single line. The total number of lines must match the value of the parameter for the number of molecules and the name of the residues in each chain should match the name of existing PREPI files in the PREPIs' directory.

  To define a linear chain, it is necessary to specify the sequence of residues that it forms. For this, the residue name defined in the PREPI file will be used. The chain's growth direction will always be from the leftmost residue in the sequence to the rightmost one. Usually, the terminal residues are of the Head and Tail type, respectively, while the residues in the middle are of the Body type. In addition, a sequence of repeated residues in a chain can be expressed consensually simply by specifying the number of times the residue is repeated.

  Example. Different ways to define four linear chains of polyethylene containing 12 repeating units. The first and the last residues are the Head and Tail types, respectively.

  ```
  PEH, PEB, PEB, PEB, PEB, PEB, PEB, PEB, PEB, PEB, PEB, PET
  PEH, 10 PEB, PET
  PEH, 5 PEB, 5 PEB, PET
  PEH, 3 PEB, PEB, PEB, PEB, PEB, 3 PEB, PET
  ```

  A system with linear chains only requires the parameters enlisted up to this point. Systems with crosslinks require the following additional parameters.

- **Crosslinks:** The total number of crosslink molecules in the system. Once this parameter is specified, it is necessary to define the additional parameters to build a system with crosslinks.

  Example. Create a system with five crosslinks.

  ```
  crosslinks,5
  ```

- **Starting atom bond length:** This parameter defines the distance in angstroms between the first atom of the backbone of the crosslink molecule and the starting atom found in a linear chain of the system.

  Example. Set the starting atom bond length to 1.5 Å.

  ```
  bond_length_CL_start, 1.5 A
  ```

- **Closing atom bond length:** This parameter defines the distance in angstroms between the last atom of the backbone of the crosslink molecule and the closing atom found in a linear chain of the system.

Example. Set the closing atom bond length to 1.5 Å.

```
bond_length_CL_close, 1.5 A
```

- **Distortion angle:** This parameter defines the maximum rotation angle (in degrees) allowed for a crosslink residue when attempting to find a new closing atom. If the crosslink cannot be completed at the intended target atom, the residue can be rotated within this limit to facilitate the connection. All those potential closing atoms that require a rotation angle higher than the value set in this parameter are discarded.

  Example. Set the distortion angle to 30º.

  ```
  distortion_angle, 30
  ```

- **Crosslink lines definition:** Each crosslink molecule must be defined on a single line. The total number of lines must match the value of the Crosslinks parameter.

  Just as when defining a linear chain, it is necessary to determine the sequence of residues that make up a crosslinking molecule and also the name of the starting and closing atoms, and, if necessary, the type of atoms that must be removed to maintain the consistency of the chemical bond.

  The way of defining a crosslink line is:

  [starting atom name, removable atoms: closing atom name, removable atoms] sequence.

  The names of the starting and closing atoms are identified using the same nomenclature as the corresponding entries in the PREPI file, (column 2), which defines the atoms names for the associated residue in the linear chains.

  The removable atoms to maintain the consistency of the chemical bond can be specified in three different ways:

  - a) specifying the name of the atom as for starting and closing atoms;
  - b) setting that all non-M type atoms bonded to the starting or closing atom are removed by writing the keyword `branch`;
  - c) setting that removing any atom is unnecessary by writing the keyword `null`.

  The residue sequence is then defined as for linear chains.

Additionally, SuSi allows the creation of grafted polymer and dendrimer systems. To do this, once the linear chains have been specified, the user must use the keywords "graft" and "dendrimer" as follows:

- a) define a graft chain connected to an atom in a linear chain.
- b) define a dendrimer fraction that can be connected to a graft chain already placed in the system.

Example. Create a crosslink made of 20 PEB units. The first residue of the crosslink is bonded to any C1 atom of the starting linear chains. The last residue of the crosslink is bonded to any C2 atom of the ending linear chains. To keep the chemical consistency, remove H1 atom bonded to C1 atom and H2 atom bonded to C2 atoms.

```
[C1,H1:C2,H2] 20 PEB
```

Example. Create a crosslink made of 10 PEB units. The first residue of the crosslink is bonded to any C1 atom of the linear chains. The last residue of the crosslink is also bonded to any C2 atom of the linear chains. Remove the H3 atom bonded to the C1 atom in the starting position and all the heavy non-M type atoms bonded to the C2 atoms in the closing position.

```
[C1,H3:C2,branch] 10 PEB
```

Example. Create a crosslink of 5 PEB units, one PSB, and 5 PEB units. The first residue of the crosslink is bonded to any C1 atom of the linear chains. The last residue of the crosslink is bonded to any C2 atom of the linear chains. It is not necessary to remove any atom bonded to C1. Remove H4 bonded to C2 atoms.

```
[C1,null:C2,H4] 5 PEB, PSB, 5 PEB
```

Example. Create a graft chain of 10 PE units. The first residue of the graft chain is bonded to any C1 atom of the linear chains. Remove the H3 atom bonded to the C1 atom in the starting position.

[C1,H3:graft,null] 10 PE

Example. Create a dendrimer fraction of a single PE unit bonded to any C1 atom of the system. Remove the H3 atom bonded to the C1 atom in the starting position

[C1,H3:graft,dendron] PE

4. Running SuSi

To execute SuSi, ensure the following files are properly placed in the SuSi directory:

1. The **input file** containing the system's specifications.
2. The **PREPI files** required for the simulation. These files can either be located in the same directory as SuSi or organized within their subdirectory (according to what has been defined in the input file).

Once the files are correctly positioned, run the program using the following command:

```
python3 susy.py -f <input>
```

Replace <input> with the name of your input file.

Execution Modes and Verbosity Levels

SuSi can be executed in two distinct modes and two levels of verbosity. These options provide flexibility in how the program runs and how much information it outputs during execution. To view detailed information about these modes and verbosity levels, use the help command:

```
python3 susy.py -h
```

This command will display all available options and their descriptions.

```
usage: susi.py [-h] [-t] [-v] -f <input>
options:
      -h,   --help                  Shows this help message
      -t,   --testmode              File pdb won't be created
      -v,   --verbosity             Increase verbosity of log file
      -f <input>, --file <input>    Input file with system properties
```

When the program is executed in test mode, no pdb file will be generated at the end of the process. All the information about the system will be shown on the terminal.

4.1. Examples

All necessary files are included with the SuSi package to ensure seamless operation.

4.1.1. Building linear chains of polyethylene

To construct linear chains of polyethylene (Fig. 4), three PREPI files are necessary (Fig. 2): **head (PEH)**, **tail (PET)**, and **body (PE)**. These files define the connectivity and structural elements of the polymer.

Fig. 4. Scheme of a Polyethylene chain

Input File

```
system_name,PE
box,Lx 48 A,Ly 48 A,Lz 48 A
periodical_coordinates,N
retries,10
minimum_distance, 1.3 A
maximum_deviation, 10
path_prepi,prepis/Monomers
molecules_number,20
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
PEH, 20 PE, PET
```

The input file for this system specifies the following:

- A cubic box with a side length of 48 Å.
- No periodic rearrangement.
- Any atom located less than 1.3 Å from another will be considered a collision.
- During construction, the program handles collisions by deforming the internal coordinates of the affected residue within ± 10% of their original value. This deformation process is recursive, and SuSi will make up to 10 attempts to successfully place a residue or molecule in the system.
- When SuSi processes the input file successfully, it will generate a file named PE.pdb containing the resulting polymer structure. The system will consist of 20 linear molecules composed of 22 monomers.

Results

After successful processing, the program will display the total number of processed residues, the time it took to build the system, the number of molecules created, and the name of the created pdb file.

```
/susi$ python3 susi.py -f PE.txt
Computing[##########################] 440/  440 Residues [delta 3.0 s]
### Process completed ###
NUMMOLS to print 20
file PE.pdb created
```

The resulting pdb file can be visualized in any molecular visualization program (Fig. 5) and used as a starting configuration within LEaP for molecular dynamics simulations. From this interlaced structure of linear chains at low density, the parameter file should be generated with AmberTools to find the equilibrated value of the systems density.

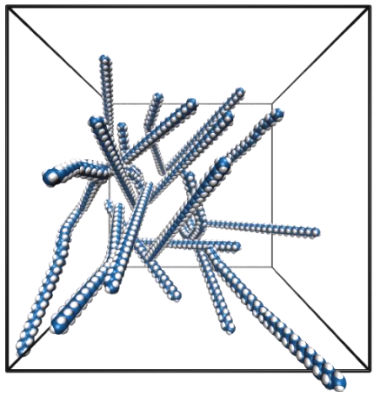

Fig. 5. Simulation box generated with SuSi. Carbon atoms are represented in blue and hydrogen atoms in white.

4.1.2. Crosslinked system

To construct the poly(N-isopropyl acrylamide) (PNIPAAm) crosslinked with N,N′-methylenebisacrylamide (MBA) (Fig. 6), four PREPI files are necessary (Fig. 7): **head (PEH)**, **tail (PET)**, and **body (PE)**. These files define the connectivity and structural elements of the polymer.

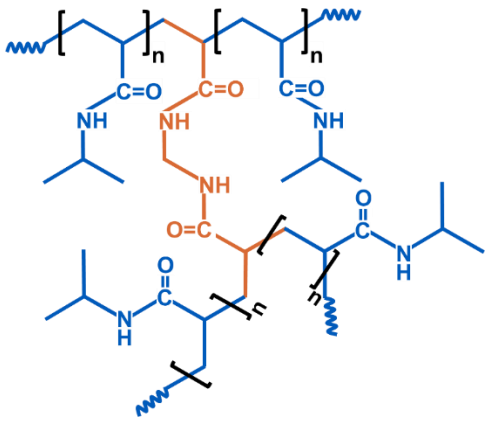

Fig. 6. Scheme of the crosslinked system p(NIPAAm-co-MBA).

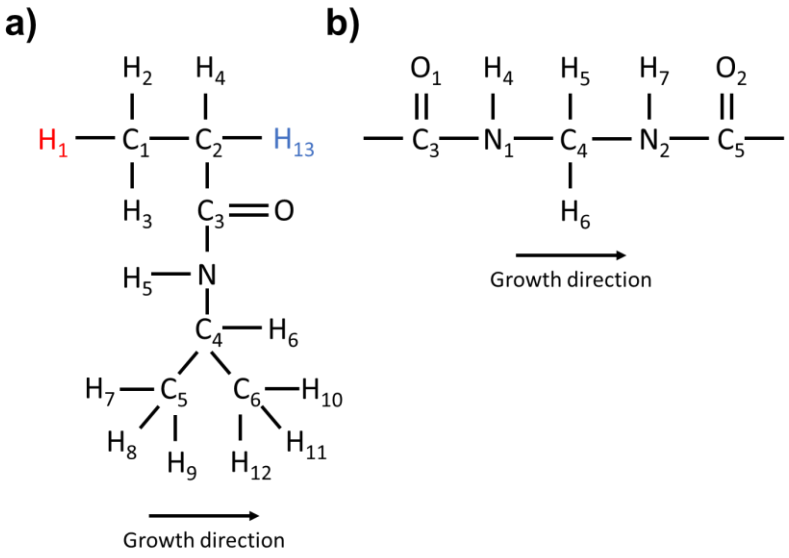

Fig. 7. Scheme of the a) NIPAAm residue with different hydrogen atoms depending on the type. (Head H1 to H12, Body H2 to H12, Tail H2 to H13), b) Cross linker residue of MBA.

Input File

```
system_name,NIPAM
box,Lx 48 A,Ly 48 A,Lz 48 A
periodical_coordinates,N
retries,10
minimum_distance, 1.3 A
maximum_deviation, 20
path_prepi,prepis/PNiPA
molecules_number,16
IAH, 12 IAB, IAT
IAH, 12 IAB, IAT
IAH, 12 IAB, IAT
IAH, 12 IAB, IAT
IAH, 12 IAB, IAT
IAH, 12 IAB, IAT
IAH, 12 IAB, IAT
IAH, 12 IAB, IAT
IAH, 12 IAB, IAT
IAH, 12 IAB, IAT
IAH, 26 IAB, IAT
IAH, 26 IAB, IAT
IAH, 26 IAB, IAT
IAH, 26 IAB, IAT
IAH, 26 IAB, IAT
IAH, 26 IAB, IAT
crosslinks,4
bond_length_CL_start, 1.45 A
bond_length_CL_close, 1.80 A
distortion_angle, 30
[C2,branch:C2,branch] MBA
[C2,branch:C2,branch] MBA
[C2,branch:C2,branch] MBA
[C2,branch:C2,branch] MBA
```

The input file for this system specifies the following:

- A cubic box with a side length of 48 Å.
- No periodic rearrangement.
- Any atom located less than 1.3 Å from another will be considered a collision.
- During construction, the program handles collisions by deforming the internal coordinates of the affected residue within ± 20% of their original value. This deformation process is

recursive, and SuSi will make up to 10 attempts to successfully place a residue or molecule in the system.

- The C3 atom of MBA will bond to any C2 atom of NIPAAm within a distance of 1.45 Å to start the crosslinking.
- The C5 atom of MBA will bond to any C2 atom of NIPAAm within a distance of 1.80 Å to start the crosslinking. This larger distance will increase the threshold for the program to create crosslinking bonds among the linear chains.
- During the construction of crosslinks, the program will rotate the crosslink residue up to 30º if it is necessary to find a new closing atom due to collisions in the system.
- All the atoms bonded to C2, but H2 will be removed to maintain the consistency of the chemical bond.
- When SuSi processes the input file successfully, it will generate a file named PNIPAAm.pdb containing the resulting polymer structure. The system comprises ten linear chains of p(NIPAAm), each consisting of 14 monomer units, and six additional p(NIPAAm) chains, each containing 28 monomers. The crosslinking is made via 4 MBA residues.

Results

After successful processing, the program will display the total number of processed residues, the time it took to build the system, the number of molecules created, and the name of the created pdb file. In the case of systems containing crosslinks, an associated bonding file is generated. This file lists the bonds between the crosslinking atoms and the starting and closing atoms in a LEaP-compatible format, specifically as:

```
bond mol.##.atom mol.##.atom
```

Additionally, if a residue has been modified by removing atoms, the file details which atoms were removed and how the residue was renamed.

```
/susi$ python3 susi.py -f PNIPAAm.txt
Computing[#########################] 312/  312 Residues [delta 163.5 s]
### Process completed ###
NUMMOLS to print 20
file PNIPAAm.pdb created
```

Because atoms have been removed, the program has renamed the residue, as seen in the bonding information file. The user should now generate a new PREPI file for this new residue Fig. 8

```
#####  Bond info CL: 1####
#Removed Branch Atoms. New residue name: IAb
bond    mol.298.C2    mol.309.C3
#Removed Branch Atoms. New residue name: IAb
bond    mol.118.C2    mol.309.C5

#####  Bond info CL: 2####
#Removed Branch Atoms. New residue name: IAb
bond    mol.150.C2    mol.310.C3
#Removed Branch Atoms. New residue name: IAb
bond    mol.116.C2    mol.310.C5

#####  Bond info CL: 3####
#Removed Branch Atoms. New residue name: IAb
bond    mol.107.C2    mol.311.C3
#Removed Branch Atoms. New residue name: IAb
bond    mol.76.C2    mol.311.C5

#####  Bond info CL: 4####
#Removed Branch Atoms. New residue name: IAb
bond    mol.299.C2    mol.312.C3
#Removed Branch Atoms. New residue name: IAb
bond    mol.114.C2    mol.312.C5
```

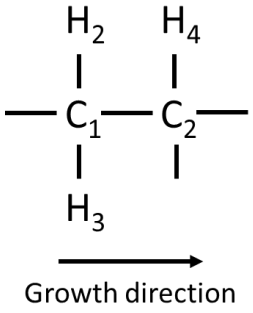

Fig. 8. Scheme of the IAb residue.

The resulting pdb file can be visualized in any molecular visualization program (Fig. 9) and used as a starting configuration within LEaP for molecular dynamics simulations.

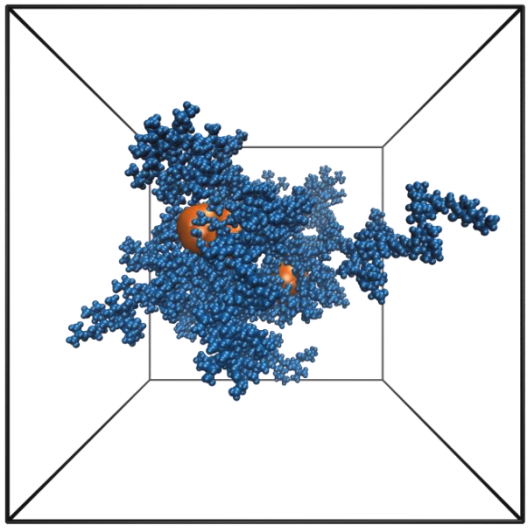

Fig. 9. Simulation box generated with SuSi. NIPAAm residues are represented in blue, while MBA residues are represented in orange.

4.1.3. Grafted system

To construct the poly(ethylente) grafted poly(styrene) system (Fig. 10), three PREPI files are necessary (Fig. 2): **head (PEH)**, **tail (PET)**, and **body (PE)** of the linear chain and the **body (PE)** and **tail (PST)** of the graft chains (Fig. 11).

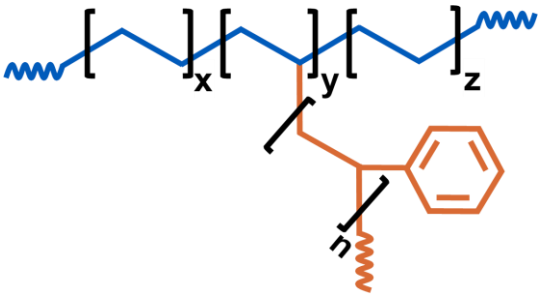

Fig. 10. Scheme of the poly(ethylene) grafted poly(styrene).

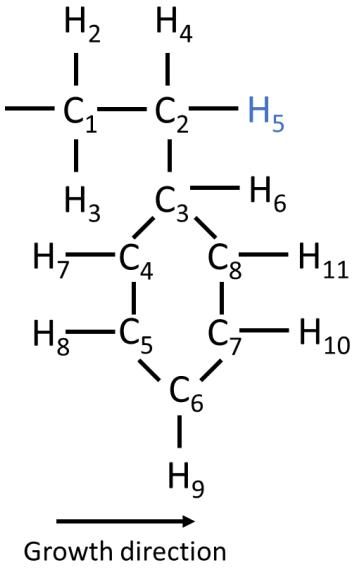

Fig. 11. Scheme of the styrene residue (PS) with different hydrogen atoms depending on the type. (The BODY type (PS) does not have H5, but the TAIL (PST) does).

Input File

```
system_name,PEgPS
box,Lx 90 A,Ly 90 A,Lz 90 A
periodical_coordinates,N
retries,10
minimum_distance, 1. A
maximum_deviation, 20
path_prepi,prepis/Monomers
molecules_number,1
44 PE
crosslinks,12
bond_length_CL_start, 1.5 A
bond_length_CL_close, 1.5 A
distortion_angle, 45
[C1,H3:graft,null] 9 PS, PST
[C1,H2:graft,null] 9 PS, PST
[C1,H3:graft,null] 9 PS, PST
[C1,H2:graft,null] 9 PS, PST
[C1,H3:graft,null] 9 PS, PST
[C1,H2:graft,null] 9 PS, PST
[C1,H3:graft,null] 9 PS, PST
[C1,H2:graft,null] 9 PS, PST
[C1,H3:graft,null] 9 PS, PST
[C1,H2:graft,null] 9 PS, PST
[C1,H3:graft,null] 9 PS, PST
[C1,H2:graft,null] 9 PS, PST
```

The input file for this system specifies the following:

- A cubic box with a side length of 90 Å.
- No periodic rearrangement.
- Any atom located less than 1. Å from another will be considered a collision.
- During construction, the program handles collisions by deforming the internal coordinates of the affected residue within ± 20% of their original value. This deformation process is recursive, and SuSi will make up to 10 attempts to successfully place a residue or molecule in the system.
- The C1atom of PS will bond to any C1 atom of PE within a distance of 1.5 Å to start the graft chain.
- All the H2 and H3 atoms bonded to a C1 with a graft will be removed to maintain the consistency of the chemical bond.

- When SuSi processes the input file successfully, it will generate a file named PEgPS.pdb containing the resulting polymer structure. The system comprises one linear chains of PE, each consisting of 44 units, and twelve PS grafts chains consisting of 10 units each.

Results

After successful processing, the program will display the total number of processed residues, the time it took to build the system, the number of molecules created, and the name of the created pdb file. In the case of systems containing grafts, an associated bonding file is generated. This file lists the bonds between the graft atoms and the starting atoms in a LEaP-compatible format, specifically as:

```
bond mol.##.atom mol.##.atom
```

Additionally, if a residue has been modified by removing atoms, the file details which atoms were removed and how the residue was renamed.

```
/susi$ python3 susi.py -f PEgPS.txt
Computing[#########################] 164/  164 Residues [delta 136.5 s]
### Process completed ###
NUMMOLS to print 13
file PEgPS.pdb created
```

Because atoms have been removed, the program has renamed the residue, as seen in the bonding information file. The user should now generate a new PREPI file for this new residues PE3 and PE2 which are basically PE body types with no H3 and H2 atoms respectively.

```
#####  Bond info GRAFT: 1####
#Removed Atom: H3. Renamed residue: PE.    New residue name: PE3
bond    mol.5.C1    mol.45.C1

#####  Bond info GRAFT: 2####
#Removed Atom: H2. Renamed residue: PE.    New residue name: PE2
bond    mol.37.C1    mol.55.C1

#####  Bond info GRAFT: 3####
#Removed Atom: H3. Renamed residue: PE.    New residue name: PE3
```

```
bond    mol.38.C1    mol.65.C1

#####  Bond info GRAFT: 4####
#Removed Atom: H2. Renamed residue: PE.    New residue name: PE2
bond    mol.23.C1    mol.75.C1

#####  Bond info GRAFT: 5####
#Removed Atom: H3. Renamed residue: PE.    New residue name: PE3
bond    mol.20.C1    mol.85.C1

#####  Bond info GRAFT: 6####
#Removed Atom: H2. Renamed residue: PE.    New residue name: PE2
bond    mol.6.C1    mol.95.C1

#####  Bond info GRAFT: 7####
#Removed Atom: H3. Renamed residue: PE.    New residue name: PE3
bond    mol.34.C1    mol.105.C1

#####  Bond info GRAFT: 8####
#Removed Atom: H2. Renamed residue: PE.    New residue name: PE2
bond    mol.22.C1    mol.115.C1

#####  Bond info GRAFT: 9####
#Removed Atom: H3. Renamed residue: PE.    New residue name: PE3
bond    mol.11.C1    mol.125.C1

#####  Bond info GRAFT: 10####
#Removed Atom: H2. Renamed residue: PE.    New residue name: PE2
bond    mol.15.C1    mol.135.C1

#####  Bond info GRAFT: 11####
#Removed Atom: H3. Renamed residue: PE.    New residue name: PE3
bond    mol.39.C1    mol.145.C1

#####  Bond info GRAFT: 12####
#Removed Atom: H2. Renamed residue: PE.    New residue name: PE2
bond    mol.26.C1    mol.155.C1
```

The resulting pdb file can be visualized in any molecular visualization program (Fig. 12) and used as a starting configuration within LEaP for molecular dynamics simulations.

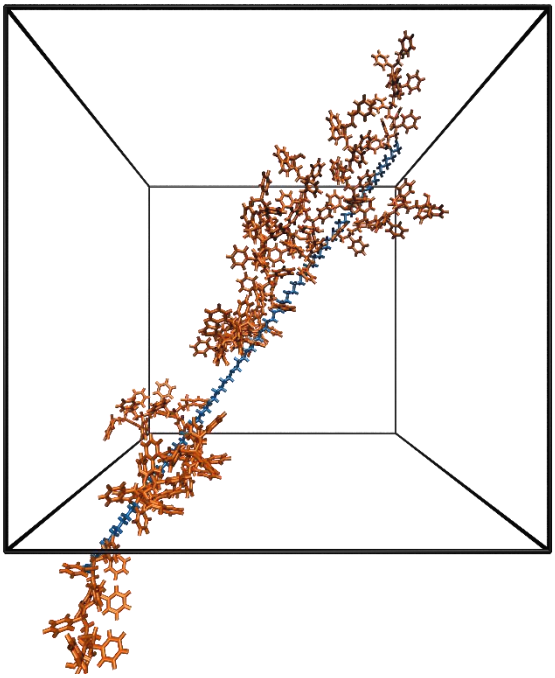

Fig. 12. Simulation box generated with SuSi. PE residues are represented in blue, while PS residues are represented in orange.

4.1.1. Dendritic system

To schematize this kind of systems we present an example of a poly(ethylene) dendrimer system (Fig. 13). In this case, three PREPI files are necessary: **body level-1 (PE)** (Fig. 2), **body level-2 (PEB)**, and **tail level-3 (PEC)** (Fig. 14).

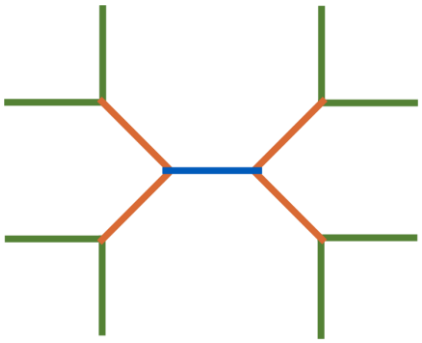

Fig. 13. Scheme of a polyethylene dendrimer with three levels.

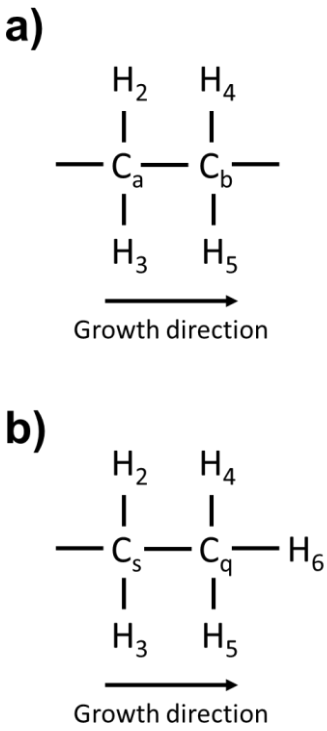

Fig. 14. Scheme of the ethylene residues for the a) second level (PEB) and b) third level (PEC) of the dendritic system.

Input File

```
system_name,PE_Dendri
box,Lx 90 A,Ly 90 A,Lz 90 A
periodical_coordinates,N
retries,20
minimum_distance, 1.4 A
maximum_deviation, 20
path_prepi,prepis/Monomers
molecules_number, 1
PE
crosslinks,12
bond_length_CL_start, 1.6 A
bond_length_CL_close, 1.8 A
distortion_angle, 45
[C1,H2:graft,dendron] 1 PEB
[C1,H3:graft,dendron] 1 PEB
[C2,H4:graft,dendron] 1 PEB
[C2,H5:graft,dendron] 1 PEB
[Cb,H4:graft,dendron] 1 PEC
[Cb,H5:graft,dendron] 1 PEC
[Cb,H4:graft,dendron] 1 PEC
[Cb,H5:graft,dendron] 1 PEC
[Cb,H4:graft,dendron] 1 PEC
[Cb,H5:graft,dendron] 1 PEC
[Cb,H4:graft,dendron] 1 PEC
[Cb,H5:graft,dendron] 1 PEC
```

The input file for this system specifies the following:

- A cubic box with a side length of 90 Å.
- No periodic rearrangement.
- Any atom located less than 1.4 Å from another will be considered a collision.

- During construction, the program handles collisions by deforming the internal coordinates of the affected residue within ± 20% of their original value. This deformation process is recursive, and SuSi will make up to 20 attempts to successfully place a residue or molecule in the system.
- The C1 atom of PEB residues will bond to C1 and C2 atoms of PE within a distance of 1.6 Å to create the second level of the dendrimer.
- The H2, H3, H4 and H5 atoms of PE will be removed to maintain the consistency of the chemical bond.
- The Cs atom of PEC residues will bond to Cb atoms of PEB within a distance of 1.6 Å to create the third level of the dendrimer.
- The H4 and H5 atoms of PE will be removed to maintain the consistency of the chemical bond.
- When SuSi processes the input file successfully, it will generate a file named PE_Dendri.pdb containing the resulting polymer structure. The system comprises thirteen residues.

Results

After successful processing, the program will display the total number of processed residues, the time it took to build the system, the number of molecules created, and the name of the created pdb file. In the case of systems containing grafts, an associated bonding file is generated. This file lists the bonds between the dendrimer atoms and the starting atoms in a LEaP-compatible format, specifically as:

```
bond mol.##.atom mol.##.atom
```

Additionally, if a residue has been modified by removing atoms, the file details which atoms were removed and how the residue was renamed.

```
/susi$ python3 susi.py -f PE_Dendri.txt
Computing[#########################] 13/   13 Residues [delta 1.5 s]
### Process completed ###
NUMMOLS to print 13
file PE_Dendri.pdb created
```

Because atoms have been removed, the program has renamed the residues, as seen in the bonding information file. The user should now generate a new PREPI file for this new residues which are basically PE residues without the corresponding removed H atoms.

```
#####  Bond info GRAFT: 1####
#Removed Atom: H2. Renamed residue: PE.    New residue name: PE2
bond    mol.1.C1    mol.2.Ca

#####  Bond info GRAFT: 2####
#Removed Atom: H3. Renamed residue: PE.    New residue name: PE3
bond    mol.1.C1    mol.3.Ca

#####  Bond info GRAFT: 3####
#Removed Atom: H4. Renamed residue: PE.    New residue name: PE4
bond    mol.1.C2    mol.4.Ca

#####  Bond info GRAFT: 4####
#Removed Atom: H5. Renamed residue: PE.    New residue name: PE5
bond    mol.1.C2    mol.5.Ca

#####  Bond info GRAFT: 5####
#Removed Atom: H4. Renamed residue: PEB.    New residue name: PE4
bond    mol.2.Cb    mol.6.Cs

#####  Bond info GRAFT: 6####
#Removed Atom: H5. Renamed residue: PEB.    New residue name: PE5
bond    mol.3.Cb    mol.7.Cs

#####  Bond info GRAFT: 7####
#Removed Atom: H4. Renamed residue: PEB.    New residue name: PE4
bond    mol.5.Cb    mol.8.Cs
```

```
#####  Bond info GRAFT: 8####
#Removed Atom: H5. Renamed residue: PEB.    New residue name: PE5
bond    mol.5.Cb    mol.9.Cs

#####  Bond info GRAFT: 9####
#Removed Atom: H4. Renamed residue: PEB.    New residue name: PE4
bond    mol.4.Cb    mol.10.Cs

#####  Bond info GRAFT: 10####
#Removed Atom: H5. Renamed residue: PEB.    New residue name: PE5
bond    mol.4.Cb    mol.11.Cs

#####  Bond info GRAFT: 11####
#Removed Atom: H4. Renamed residue: PEB.    New residue name: PE4
bond    mol.3.Cb    mol.12.Cs

#####  Bond info GRAFT: 12####
#Removed Atom: H5. Renamed residue: PEB.    New residue name: PE5
bond    mol.2.Cb    mol.13.Cs
```

The resulting pdb file can be visualized in any molecular visualization program (Fig. 15) and used as a starting configuration within LEaP for molecular dynamics simulations.

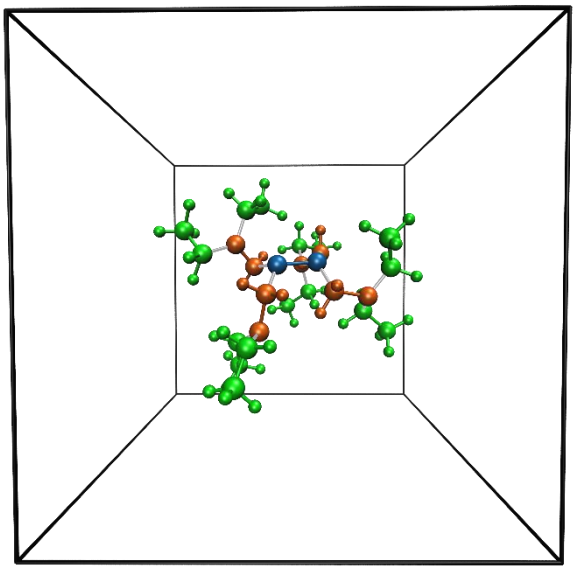

Fig. 15. Simulation box generated with SuSi. PE residue in the first level is represented in blue, the second level in orange, and the third level in green.

5. Error addressing

If the process does not execute correctly, the program will not generate the PDB file. Instead, it will display a message indicating the error that prevented the generation of the final system coordinates.

This section serves as a guide to help identify and resolve issues encountered during the execution of the program. Users are encouraged to refer to it when troubleshooting problems. Below is a list of errors that may be detected by the program, along with possible solutions to address them.

Error	Addressing
`The input file can't have less than 8 lines.`	Users must specify in the input file at least 8 lines with the parameters: • system_name, • box, • periodical_coordinates, • retries, • minimum_distance, • maximum_deviation, • path_prepi, • molecules_number,
`Collisions could not be avoided for this molecule.`	The chains cannot be placed in the box within the specified parameters. Depending on the system, different approaches can be done to address this error. If the box is too small for the number of molecules, the user could increase the size of the box or the number of retries. Users are encouraged to estimate the system's size and the chains they attempt to place within a box. Increasing the maximum deviation should be done cautiously since large values could derive un highly deformed residues. Decreasing the minimum distance should also be done cautiously since shorter values would ease the process of molecule construction, but close contacts between atoms could be identified later. Users are encouraged to check the structure of the PREPI files before running SuSi. Special attention should be paid to the values of the internal

	coordinates of the atoms connected to DUMMY atoms since these atoms (DUMMY) are used as a reference to place residues with less than 3 M-type atoms.
system_name not found in the input file.	Users must specify the keyword system_name followed by the resulting file's name within the input file parameters.
box dimension not found in the input file.	Users must specify the keyword box followed by dimensions in Å along each axis within the input file parameters.
periodical_coordinates not found in the input file.	Users must specify the keyword periodical_coordinates followed by Y or N whether periodical box rearrangement will be used to build the system, respectively.
minimum_distance not found in the input file.	Users must specify the keyword minimum_distance followed by the minimum distance value in Å within the input file parameters.
retries not found in the input file.	Users must specify the keyword retries followed by the number of retries within the input file parameters.
maximum_deviation not found in the file.	Users must specify the keyword maximum_deviation followed by the number of retries within the input file parameters.
molecules_number not found in the input file.	Users must specify the keyword molecules_number followed by the number of molecules within the input file parameters.
PATH of prepi files not found in the input file.	Users must specify the keyword path_PREPI followed by the path of the PREPI files within the input file parameters.
Error in X dimension unit.	

	Users must specify the units of the box along the x-axis in Å. The letter A should follow the number that sets the dimension.
`Error in Y dimension unit.`	Users must specify the units of the box along the y-axis in Å. The letter A should follow the number that sets the dimension.
`Error in Z dimension unit.`	Users must specify the units of the box along the z-axis in Å. The letter A should follow the number that sets the dimension.
`Error in minimum distance unit.`	Users must specify the units of the minimum distance in Å. The letter A should follow the number that sets the minimum distance.
`Error in value of periodical coordinates. Possible values are Y or N.`	Users must set either Y or N to use periodical rearrangement or not, respectively.
`Error in minimum distance.`	Users must specify the value of the minimum distance followed by the letter A.
`molecules number not found in the input file.`	Users must specify the keyword `molecules_number` followed by the number of molecules.
`PATH of prepi files not found in the input file.`	Users must specify the path of the PREPI files within the input file parameters. This path is relative to the path where SuSi runs.
`Molecules number in the input file is different than molecule number found.`	Users must check that the number of lines of molecule definition matches the number of molecules set in the input file.
`Error in box.`	Users must check the format of the box dimensions.
`Could not read file:`	

	Users must check the name of the input file they are attempting to run with SuSi.
`No errors found in the molecules building`	No errors have been detected during the process.
`Could not create file:`	Users must check the name of the system they are attempting to create with SuSi. If the file already exists, SuSi will not overwrite it.
`Large real number to fit a PDB format in atom`	The number of atoms in the system exceeds the conventional value of 99,999 for PDB files.
`Wrong number of starting and closing atoms for crosslinking`	Users must set only one starting and one closing atom for every crosslinking molecule.
`Starting or closing atom does not exist in Prepi files. Wrong name for atom:`	Users must check the name of the starting and closing atom is the name of an actual atom in the PREPI files used to create the system.
`Error in distortion angle. Wrong or not specified value.`	Users must specify the keyword `distortion_angle` followed by the value of the distortion angle in degrees (not unit specification is necessary).
`crosslinks_number not found in the input file.`	Users must specify the keyword `crosslinks_number` followed by the number of crosslinker molecules.
`Crosslinks number in the input file is different than crosslinks number found.`	Users must check that the definition of the number of lines of crosslinker molecules matches the number of crosslinks set in the input file.
`Could not create cross links. The linear chains have been written in the restart file.`	The crosslinks molecules could not be created because collisions could not be avoided. For a more detailed explanation, users are encouraged to check the error "`Collisions could not be avoided for this molecule.`" in this table.

Error in crosslink starting bond length. Wrong or not specified value.	Users must specify the keyword bond_length_CL_start followed by the crosslink starting bond length value in Å within the input file parameters.
Error in the units. Only Angstrom (A) are supported.	Users must specify the value of the starting or closing bond length followed by the letter A.
Minimum distance is greater than crosslink starting bond lenght.	The value of the crosslink starting bond length cannot be less than the minimum distance because it would always be detected as a collision.
Wrong number of removable atoms for starting and/or closing atoms for crosslinking. If no atoms need to be removed, specify as null.	User must specify the removable atoms using one of the three ways: name of a single atom, branch atoms, or null for no atoms.
Removable atoms for the starting or closing atom does not exist in Prepi files. Wrong name for atom:	When the name of the removable atoms is specified, users must check this name is the name of an actual atom in the PREPI files used to create the system.
Removable atoms not bonded to the starting or closing atom. Wrong name for atom:	When the name of the removable atoms is specified, users must check these atoms are bonded to the starting or closing atom, which is defined by the connection column in the PREPI file used to create the system.
Error in crosslink closing bond lenght. Wrong or not specified value.	Users must specify the keyword bond_length_CL_close followed by the crosslink closing bond length value in Å within the input file parameters.
Minimum distance is greater than crosslink closing bond lenght.	The value of the crosslink closing bond length cannot be less than the minimum distance because it would always be detected as a collision.
No closing atoms were found within the distance defined by the crosslink size.	Users are encouraged to check the dimensions of the systems. Usually, dense systems are required for short crosslinker molecules. On the other hand,

	bigger boxes are required for systems with long crosslinker molecules.