CGSB Cheat Sheet

Basic syntax

CGSB can be used in a **Python script**:

```
CGSB.CGSB(argument = "subarg1:val subarg2:val")
```

or in the **terminal command line**:

```
python -m CGSB -argument subarg1:val subarg2:val
```

Box commands

Box types:

`box_type = "rectangular"`	min 3 dimensions (default).
`box_type = "hexagonal"`	min 2 dimensions.
`box_type = "skewed_hexagonal"`	min 1 dimension.
`box_type = "dodecahedron"`	min 1 dimension.

Box dimensions:

`box = [10, 10, 10]`	Box dimensions `x, y, z` (in nm).
`x = 10, y = 10, z = 10`	Box dimensions `x, y, z` (in nm).

Pdb/gro box format:

`pdb_unitcell = [x, y, z, α, β, γ]`

`gro_unitcell = [`$v_1(x)$, $v_2(y)$, $v_3(z)$, $v_1(y)$, $v_1(z)$, $v_2(x)$, $v_2(z)$, $v_3(x)$, $v_3(y)$`]`

Environment variables

Output:

`out_sys`	(str) Output .pdb and .gro file.
`out_sys_pdb`	(str) Output .pdb file.
`out_sys_gro`	(str) Output .gro file.
`out_top`	(str) Output .top file.
`out_log`	(str) Output .log file.

Topology:

`itp_input`	(str) Input .top file.

Parameters:

`sys_params`	(str) Parameter library for system.
`lipid_params`	(str) Parameter library for lipids.
`solv_params`	(str) Parameter library for solvents.
`prot_params`	(str) Parameter library for protein.

Miscellaneous:

`sn`	System name under `[ system ]`.
`backup`	(bool) Backup overwritten files (default: `True`).
`randseed`	(int) Random seed.
`verbose`	(int) Verboseness (default: 1)

Molecule import

1) If molecule topology exists:

```
molecule_import = "file:pdbfile moleculetype:MOLNAME"
```

2) If molecule topology does not exist:

```
molecule_import = "file:pdbfile name:NAME charge:[int]"
```

Lipid import with orientation:

```
molecule_import = "file:pdbfile moleculetype:MOLNAME
upbead:0:res:0 downbead:11:res:0"
```

Charge handling:

No `charge` command	charge:0.
`charge:[int]`	Charge divided across all beads.
`charge:[int]:res:[int]: bead:[int]`	Charge for a bead in a residue.

Membrane building

Membrane types:

`type:bilayer`	default.
`type:mono_upper`	upwards-facing monolayer.
`type:mono_lower`	downwards-facing monolayer.
`type:mono`	upwards-facing monolayer.
`type:upper`	upper leaflet (in isolation `mono_upper`)
`type:lower`	upper leaflet (in isolation `mono_lower`)

Lipid composition:

`lipid:POPC`	lipid_type:ratio=1
`lipid:POPE:5`	lipid_type:ratio
`lipid:CHOL:3:params:dev18`	lipid_type:ratio:params:LIBRARY

Area per lipid:

`apl:0.6`	(float) default: 0.6

Leaflets:

`leaflet:upper`	Upper leaflet subcommands follow.
`leaflet:lower`	Lower leaflet subcommands follow.
`leaflet:both`	Following subcommands apply to both.

Placement and size:

`xlength, ylength`	(float) x/y dimensions of the membrane
`cx, xy, xz`	(float) membrane position in x, y, or z

Membrane patches and holes:

`hole:circle:radius:[r]`	circular hole
`hole:ellipse:xradius:[x]:yradius:[y]`	elliptical hole
`hole:square:length:[l]`	square hole
`hole:rectangle:xlength:[x]:ylength:[y]`	rectangular hole
`hole:polygon:p:[x]:[y]:p:[x]:[y]...`	polygonal hole

For patches, replace `hole` with `patch`.

Modification of holes/patches (`hole:..:modif:val`):

`rotate`	(float) rotation (in degrees).
`cx, cy`	(float) x and y center of the hole/patch (in nm).
`xscaling`	(float) shape scaling in x dimension.
`yscaling`	(float) shape scaling in y dimension.

Examples

(1) POPC:CHOL bilayer in 5:1 ratio:

```
membrane = "lipid:POPC:5 lipid:CHOL:1"
```

(2) Asymmetric bilayer:

```
membrane = "apl:0.5 leaflet:upper lipid:POPC:5
lipid:CHOL:1 leaflet:lower lipid:POPC:3 lipid:CHOL:2
leaflet:both params:Dev18"
```

(3) Phase-separated membrane:

```
membrane = ["lipid:POPC:5 lipid:CHOL:1 xlength:5
cx:2.5", "lipid:POPC:4 lipid:CHOL:2 xlength:5 cx:-2.5"]
```

(4) Membrane with a rotated rectangle hole:

```
membrane = "lipid:POPC:3 lipid:POPE:2 apl:0.5
hole:rectangle:xlength:2:ylength:3:rotate:45"
```

Protein insertion

Syntax:

```
protein = "file:pdbfile subcommand:[val]"
```

Add topology:

`moleculetype:NAME`	`[ moleculetype ]` name
`moleculetype:NAME1:NAME2`	Multiple `[ moleculetype ]`.

Placement:

`cx, cy, cz` (float) where to place center in system (in nm)

`rx, ry, rz` (float) rotations around given axis (in degrees)

Centering method:

`cen_method:cog`	(default) Center of geometry.
`cen_method:axis`	Axial mean coordinate.
`cen_method:res:5-7:10-20`	Residues 5-7 and 10-20.
`cen_method:point:0:0:0`	a coordinate point (0,0,0).

Solvation

Solvent composition:

`solv:W`	`solvent_type`.
`solv:W:2`	`solvent_type:ratio`.
`solv:W:2:params:DevWater5`	`solvent_type:ratio:params:LIBRARY`.
`pos:NA`	Positive ions.
`neg:CL`	Negative ions.

Modification:

`solv_molarity`	(float) default: 55.56 mol/L.
`salt_molarity`	(float) default: 0.15 mol/L.
`solv_per_lipid`	(int) $N_{\text{solvent}} = d_{\text{solv per lipid}} \cdot N_{\text{lipid}}$

Examples:

```
solvation = "solv:W pos:NA neg:CL"
solvation = "default" # same as above
```

Flooding commands

Syntax:

```
flooding = "solute:NAME1:COUNT1 solute:NAME2:COUNT2"
```

Example:

```
flooding = "solute:RHO:30"
```

Example (if topology exists):

```
flooding = "solute:SUCR:50",
molecule_import = "file:sucrose.pdb moleculetype:SUCR"
```

Example (if topology does not exist):

```
flooding = "solute:SUCR:50",
molecule_import = "file:sucrose.pdb name:SUCR charge:0"
```

References

http://github.com/MikkelDA/CGSB