Lic. César L. Araujo
cesar.araujo@oulu.fi
Research Center for Molecular Endocrinology
University of Oulu, Finland

Important: Please mention explicitly the use of any version of this script program in your publications. This will encourage me to continue with the improvements and maintenance of the code.

NAME

dock41.pl - Perl script program to perform MD experiments in different flavours

SYNOPSIS

dock41.pl -config <file.cfg>
-mode <"solvated"|"ionsolvated"|"invacuo">
[-iontype <"p"|"n">]
[-ionname "string"]
[-ioncount #_of_ions]
[-continue <"RUN_POSITION_RESTRAINED"|"RUN_DYNAMICS">]
[-continue_from_crash]
[-docking]
[-complex]
[-growing_ligand]
[-x]

DESCRIPTION

Dock is a Perl script that automates MD studies using **Gromacs**[1].
With Dock you can perform dynamic simulations and docking studies in three different flavors: solvated, solvated with counter ions and in vacuo.

dock41.pl Options:

Option	Description
-config <file.cfg>	**dock41.pl** is parameterized through a set of variables contained in a config file (.cfg). Most of them just define relative location of input and output files. It lets **dock41.pl** to organize your files in a standard directory structure and assign output filenames in a uniform way. You can use the sample templates and customize them to fit your needs. The config file is a Perl file that **dock41.pl** reads at runtime, so you can embed Perl code in it for

	flexibility.
-mode <"solvated"| "ionsolvated" | "invacuo">	The mode determines the way in which the simulation will be carried out. The mode solvated" creates a periodic box and fill it with water molecules. After that, the script performs an energy minimization (EM) of the system followed by a position restraint (PR) and then the molecular dynamics (MD). The "ionsolvated" mode it's the same but you can add counter ions to the system. The "invacuo" mode just perform the EM step followed by the MD step.
-continue <"RUN_POSITION_RESTRAINED" | "RUN_DYNAMICS">	This option lets you restart the script to avoid the repetition of previous calculations. If the EM step was completed, you can skip this using the option "RUN_POSITION_RESTRAINED" and the script will continue from that point. If the script stop after PR step completion you can restart it and continue from MD step with the option "RUN_DYNAMICS".
-continue_from_crash	This option let you restart your calculations after a shutdown in the system. After finalization you must concatenate the files as usual (see Gromacs documentation). The script will attempt to discard the last frame before the crash. Example: after a crash you can do: ../../**dock41.pl** –x –config <file.cfg> -mode "your mode" -docking -continue_from_crash **NOTE**: The current work directory should be the project one. You must supply the original options too. So, you should just add the option "-continue_from_crash" to your script.
-extend <time>	This option extends and existing simulation by the time in picoseconds provided as an option.
-iontype <"p"|"n">	This option is used in "ionsolvated" mode for genion program (see Gromacs documentation). Use "p" to add positive ions and "n" for negative ones.

Lic. César L. Araujo
cesar.araujo@oulu.fi
Research Center for Molecular Endocrinology
University of Oulu, Finland

-ionname <"string">	Name of the ions to add, e.g. "Na" for sodium.
-ioncount <#_of ions>	Number of ions to add.
-docking	This option indicates to the program that you are going to perform a docking experiment. In this case the protein and the ligand are in different files. This option is optimized for **PRODRG** server, so there are some restrictions. For example, the file names must be: **DRGGMX.ITP** ligand ITP file as is returned by **PRODRG** server By default, the name of the ligand returned by **PRODRG** is **DRG**. You can use the configuration variable **$LIGAND_NAME** to change it to another one. **NOTE**: you can change all of this but be sure about the consistency of these file's content.
-complex	This option is the same as "docking" but the complex is represented just by one file. In this case you must be sure that the topology of the ligand can be handled by Gromacs by pdb2gmx utility.
-growing_ligand	This is an experimental option.This option is based in a previous idea described in the literature[2] and let you grow the ligand in the active site. The trick consists in perform cycles EM+PR changing the ligand topology in each iteration. You start with a topology with most of your ligand's atoms turned-OFF (force-field parameters, mass and charge set to zero) and then continue with a set of topologies, each one with more atoms turned-ON until the last one in which all the atoms are present for calculation of MD step. The effect is to grow the ligand in a smooth way avoiding LINCS warnings problems during PR step that usually occur when the ligand is clashing with the protein residues. See the section "HOW-TO" for more information.

Lic. César L. Araujo
cesar.araujo@oulu.fi
Research Center for Molecular Endocrinology
University of Oulu, Finland

-x	If this option is present all the Gromacs' programs are executed. If not, **dock41.pl** just print out on the screen the programs invocation with their arguments for debug purposes. I encourage you to use this option to test the consistency of your config file settings and verify the consistency of data files. The **dock.log** file contains all the script activity, so you can see what goes wrong and fix it before run the script for production. **NOTE**: if you run the script without the "x" option, and you see Gromacs errors, you can ignore them at the moment, because they could be consequence that the programs are not being executing at all, so input files for some of them could be missing.

MISCELANEOUS

dock41.pl is able to run in **LAM/MPI** environment adjusting their settings for a multiprocessor environment.

To do this, you must provide the same file used to boot the LAM environment with the same name that the starting node. For example, if you are going to use the nodes **brutus27** and **brutus28** in cluster **Brutus**, you can provide the file named "**brutus27**" in the directory of the project and use it to boot the LAM environment:

For example, if the **brutus27** file's content is:

```
brutus27 cpu=2
brutus28 cpu=2
```

and then:

user@**brutus27**> **lamboot** -v brutus27

When you run the script, **dock41.pl** will look for a file with the same name as the current machine and then open it and counts the number of cpu's used for -np option in mdrun calls.

Lic. César L. Araujo
cesar.araujo@oulu.fi
Research Center for Molecular Endocrinology
University of Oulu, Finland

All the mdrun calls are made through **nohup** utility in background so you can detach your ssh session safely.

HOW-TO

How to setup the system to use the **–growing_ligand** option.

If you have the ITP file for your ligand in the data directory, make many copies of it as the number of steps needed to grow the ligand. For example, consider you have the file "ligand.itp" for the cholesterol molecule. Then you want to grow this ligand in the active site of your protein through 3 steps. So, first you have to make 2 copies of your ligand.itp file called: ligand1.itp and ligand2.itp (the last step will be your original ligand.itp file). Obviously, you can use as many steps as you want.

Then, modify the files ligand1.itp and ligand2.itp changing charge and masses to zero and atom type to some dummy particle type with force-field parameters set to zero (for example for Gromacs force-file you can use the atom type IW) for those atoms that should be "invisible" during EM+PR step. For example, you can leave "visible" atoms from rings A and B in ligand1.itp and in ligand2.itp you can leave "visible" atoms from rings A, B, C y D. Then ligand.itp is your original ITP file with all the visible atoms. See file examples below.

ligand.itp:

```
[ atoms ]
;   nr       type  resnr resid  atom  cgnr    charge      mass
     1        CH3      1  CLR     C27     1   -0.045  15.0350
     2        CH1      1  CLR     C13     1    0.029  12.0110
     3        CH2      1  CLR     C14     1   -0.001  14.0270
     4        CH2      1  CLR     C15     1   -0.002  14.0270
     5        CH1      1  CLR     C16     1    0.019  13.0190
     6        CH1      1  CLR     C21     2    0.037  12.0110
     7        CH3      1  CLR     C26     2   -0.036  15.0350
     8        CH2      1  CLR     C22     2    0.000  14.0270
     9        CH2      1  CLR     C23     2   -0.001  14.0270
    10        CH1      1  CLR     C24     3    0.078  13.0190
    11         OA      1  CLR      O1     3   -0.073  15.9994
    12         HO      1  CLR     HAA     3   -0.005   1.0080
    13        CH2      1  CLR     C25     3    0.000  14.0270
    14         CB      1  CLR     C20     4   -0.045  12.0110
    15       CR61      1  CLR     C19     4   -0.038  13.0190
    16        CH2      1  CLR     C18     4   -0.001  14.0270
    17        CH1      1  CLR     C17     4    0.042  13.0190
    18        CH1      1  CLR     C12     4    0.042  13.0190
    19        CH2      1  CLR     C11     5   -0.003  14.0270
    20        CH2      1  CLR     C10     5   -0.003  14.0270
    21        CH1      1  CLR      C9     5    0.006  13.0190
    22        CH1      1  CLR      C7     6    0.030  13.0190
    23        CH3      1  CLR      C8     6   -0.028  15.0350
    24        CH2      1  CLR      C6     6   -0.001  14.0270
    25        CH2      1  CLR      C5     6    0.000  14.0270
    26        CH2      1  CLR      C4     6   -0.001  14.0270
    27        CH1      1  CLR      C2     7    0.041  13.0190
```

Lic. César L. Araujo
cesar.araujo@oulu.fi
Research Center for Molecular Endocrinology
University of Oulu, Finland

```
    28       CH3     1  CLR      C3     7   -0.021  15.0350
    29       CH3     1  CLR      C1     7   -0.020  15.0350
```

ligand1.itp:

```
[ atoms ]
;   nr       type  resnr resid  atom  cgnr    charge      mass
     1         IW     1  CLR     C27     1   -0.000  00.0000
     2         IW     1  CLR     C13     1    0.000  00.0000
     3         IW     1  CLR     C14     1   -0.000  00.0000
     4         IW     1  CLR     C15     1   -0.000  00.0000
     5        CH1     1  CLR     C16     1    0.019  13.0190
     6        CH1     1  CLR     C21     2    0.037  12.0110
     7         IW     1  CLR     C26     2   -0.000  00.0000
     8        CH2     1  CLR     C22     2    0.000  14.0270
     9        CH2     1  CLR     C23     2   -0.001  14.0270
    10        CH1     1  CLR     C24     3    0.078  13.0190
    11         IW     1  CLR      O1     3   -0.000  00.0000
    12         IW     1  CLR     HAA     3   -0.000   0.0000
    13        CH2     1  CLR     C25     3    0.000  14.0270
    14         CB     1  CLR     C20     4   -0.045  12.0110
    15       CR61     1  CLR     C19     4   -0.038  13.0190
    16        CH2     1  CLR     C18     4   -0.001  14.0270
    17        CH1     1  CLR     C17     4    0.042  13.0190
    18         IW     1  CLR     C12     4    0.000  00.0000
    19         IW     1  CLR     C11     5   -0.000  00.0000
    20         IW     1  CLR     C10     5   -0.000  00.0000
    21         IW     1  CLR      C9     5    0.000  00.0000
    22         IW     1  CLR      C7     6    0.000  00.0000
    23         IW     1  CLR      C8     6   -0.000  00.0000
    24         IW     1  CLR      C6     6   -0.000  00.0000
    25         IW     1  CLR      C5     6    0.000  00.0000
    26         IW     1  CLR      C4     6   -0.000  00.0000
    27         IW     1  CLR      C2     7    0.000  00.0000
    28         IW     1  CLR      C3     7   -0.000  00.0000
    29         IW     1  CLR      C1     7   -0.000  00.0000
```

ligand2.itp:

```
[ atoms ]
;   nr       type  resnr resid  atom  cgnr    charge      mass
     1        CH3     1  CLR     C27     1   -0.045  15.0350
     2        CH1     1  CLR     C13     1    0.029  12.0110
     3        CH2     1  CLR     C14     1   -0.001  14.0270
     4        CH2     1  CLR     C15     1   -0.002  14.0270
     5        CH1     1  CLR     C16     1    0.019  13.0190
     6        CH1     1  CLR     C21     2    0.037  12.0110
     7        CH3     1  CLR     C26     2   -0.036  15.0350
     8        CH2     1  CLR     C22     2    0.000  14.0270
     9        CH2     1  CLR     C23     2   -0.001  14.0270
    10        CH1     1  CLR     C24     3    0.078  13.0190
    11         IW     1  CLR      O1     3   -0.000  00.0000
    12         IW     1  CLR     HAA     3   -0.000   0.0000
    13        CH2     1  CLR     C25     3    0.000  14.0270
    14         CB     1  CLR     C20     4   -0.045  12.0110
    15       CR61     1  CLR     C19     4   -0.038  13.0190
    16        CH2     1  CLR     C18     4   -0.001  14.0270
    17        CH1     1  CLR     C17     4    0.042  13.0190
    18        CH1     1  CLR     C12     4    0.042  13.0190
    19        CH2     1  CLR     C11     5   -0.003  14.0270
    20        CH2     1  CLR     C10     5   -0.003  14.0270
    21        CH1     1  CLR      C9     5    0.006  13.0190
    22         IW     1  CLR      C7     6    0.000  00.0000
    23         IW     1  CLR      C8     6   -0.000  00.0000
    24         IW     1  CLR      C6     6   -0.000  00.0000
    25         IW     1  CLR      C5     6    0.000  00.0000
```

Lic. César L. Araujo
cesar.araujo@oulu.fi
Research Center for Molecular Endocrinology
University of Oulu, Finland

```
    26        IW     1  CLR      C4     6   -0.000  00.0000
    27        IW     1  CLR      C2     7    0.000  00.0000
    28        IW     1  CLR      C3     7   -0.000  00.0000
    29        IW     1  CLR      C1     7   -0.000  00.0000
```

Then, setup the variables in the corresponding configuration file (e.g. growing.cfg):

….

```
$LIGAND_ITP_INCLUDE        = "../data/DRGGMX.ITP";
$LIGAND_ITP_FILE           = "$PROJECT_DIRECTORY/data/ligand.itp";

@LIGAND_GROWING_TOPOLOGIES =("$PROJECT_DIRECTORY/data/ligand1.itp",
"$PROJECT_DIRECTORY/data/ligand2.itp", $LIGAND_ITP_FILE);

$LIGAND_GRO_FILE           = "$PROJECT_DIRECTORY/data/ligand.gro";
$LIGAND_NAME               = "CLR";
```

Observe that @LIGAND_GROWING_TOPOLOGIES is an array and note how the files are referenced.

Finally, run the script adding the –growing_ligand option, for example:

>./dock41.pl -config "growing.cfg" -mode "solvated" -docking -growing_ligand

And the magic starts. The program will perform EM+PR with each topology using the results from one cycle as the input for the next one. After those cycles it will continue with the regular MD step.

Obviously, this method is not perfect. If you still receive LINCS warnings, you should consider break your ligand in smaller fragments. This will increase the time of the simulation because you are adding more EM+PR cycles.

Lic. César L. Araujo
cesar.araujo@oulu.fi
Research Center for Molecular Endocrinology
University of Oulu, Finland

FAQS

« Where do reside the output files? »

Each provided configuration script file in project's templates creates the corresponding output directory according to the selected mode (**solvated**, **ionsolvated** and **invacuo**). So it is safe to delete those subdirectories because all required files to start the simulations (mdp files, pdb files, script config files, etc.) are in project directory and in the data subdirectory. All the output filenames begin with the value of $FILE_PREFIX configuration variable. For example, if the value of $FILE_PREFIX is "dock" after a docking experiment these will be the output files:

Description	Files
EM step	dock_em.tpr, dock_em.trr, em.log, em.edr
PR step	dock_pr.tpr, dock_pr.trr, pr.edr, pr.log
MD step	dock_md.tpr, dock_md.trr, md.edr, md.log
Coordinate file	dock.gro
Topology file	dock.top
Coordinate file with solvent molecules (output file of **genbox**).	dock_b4ion.gro
Coordinate input file for EM step	dock_b4em.gro
Coordinate input file for PR step	dock_b4pr.gro
Coordinate input file for MD step	dock_b4md.gro
Coordinate result file of MD step	**dock_b4pmd.gro**

« I made a mistake in the mdp file and I want to restart the simulation from scratch. Is it safe to delete the output directory?»

Yes. As it was said before, each output directory is created if it doesn't exist at runtime by the config script file.

« I want to start a new project. How I would proceed? »

1) Copy the appropriate template subdirectory and give it a new name, e.g. "**MyNewProject**".

2) If it is a "**dynamics**" project type you should replace the following files with your owns:

Lic. César L. Araujo
cesar.araujo@oulu.fi
Research Center for Molecular Endocrinology
University of Oulu, Finland

All **MDP** files in data subdirectory:

File	Mode	Description
em.mdp	All	Energy minimization step
pr.mdp	solvated	Position restraints step
pr_ion.mdp	ionsolvated	Position restraints step
md.mdp	solvated	Molecular dynamics step
md_ion.mdp	ionsolvated	Molecular dynamics step
md_vac.mdp	invacuo	Molecular dynamics step

Replace the **protein.pdb** file in data subdirectory with your own PDB file.

If it is a "**docking**" project type you should replace in addition to the previous files the **DRGGMX.ITP** file with the corresponding **ITP** for your ligand.

3) Edit the file dynamics.cfg. In general you will only need to modify the lines after the comment "**# Modify the following variables values according to your needs**".

« Which are the differences between "dynamics", "relaxing" and "docking" templates?»

The **dynamics** template is prepared to run an MD simulation step after EM and PR steps of a system defined just by a PDB file.

The **relaxing** template is similar to **dynamics** template but for arbitrary molecules using topologies returned by PRODRG server.

The **docking** template is prepared to run an MD simulation step after EM and PR steps but it's optimized to work with arbitrary ligands (for topologies not recognized by pdb2gmx program) and topologies generated by PRODRG server.

« I have my own topology file defined for my ligand. Can I use it? »

Yes. But you must to take care about the needed file modifications to fit the "**docking**" template model.

« I need to use –asp option in pdb2gmx program for my protein, but I don't want to specify the protonation state every time I run the simulation with different parameters. What can I do? »

Lic. César L. Araujo
cesar.araujo@oulu.fi
Research Center for Molecular Endocrinology
University of Oulu, Finland

Just run the **pdb2gmx** program with your protein using the following options (the same that **dock41.pl** would use):

user@machine> **pdb2gmx -ignh -his -asp -ff gmx -f protein.pdb -o dock.gro -p dock.top -i posre.itp**

Then copy the **dock.top**, **.ITP** file(s) and your **dock.gro** file to the data directory of your project. The next time you run the simulation, the configuration script will look for a data file called "**dock.top**" (e.g. see **docking.cfg**) and if it exists, it will be copied to the corresponding output subdirectory before the starting of the calculations.

In addition, you should specify an empty string for **$PDB_FILE** variable in your config script file. So your modified config script file should look as:

…

$PDB_FILE = "";

…

In this way, **dock41.pl** will skip the **pdb2gmx** program call every time you run the script.

« Can I run the script in Sepeli cluster? »

Yes. You just take into account the extra steps needed to interact with the cluster queue system. In this version of the script, the executable name for **mdrun** program is parameterized in the configuration variable **$MDRUN**. By default, this variable has the value "**mdrun**" so you can still use the script in **Brutus** cluster. For **Sepeli** cluster, you must supply this variable with the "**mdrun_mpi**" value in the configuration file. In addition, the value of **$NOHUP** and **$BACKGROUND** variables should be set to empty string ("").
Example:

1) In your configuration file (.CFG) setup the following variables with the following values:

```
$MDRUN = "mdrun_mpi";   # default value is "mdrun"
$NOHUP = "";            # default value is "nohup"
$BACKGROUND = "";       # default value is "&"
```

2) To run your project you can use a script like this:

Lic. César L. Araujo
cesar.araujo@oulu.fi
Research Center for Molecular Endocrinology
University of Oulu, Finland

solvated.sh:

```
#!/bin/bash

#$ -S /bin/bash
#$ -cwd
#$ -e ./stderr.log
#$ -o ./stdout.log
#$ -V
#$ -j y
#$ -R y
#$ -N <your job name>
#$ -M <your e-mail address>
#$ -pe lam-path64 4
#$ -l h_rt=24:00:00,h_vem=500M

source /v/linux24_x8664/appl/chem/gmx-321-lam-path_64/GMXRC.bash

echo "Got $NSLOTS processors"

<path to script>/dock41.pl –x –config "your config file" –mode "solvated"
```

Note: you should provide suitable values for the number of processors to use in "-pe" option (in this example is 4), and for the running time (24:00:00) and the amount of memory (500M) (see –l option).

3) To submit the job just type:

```
>qsub solvated.sh
```

For further information about Sepeli cluster, please read CSC documentation.

« During grompp processing I got warnings referring that some atoms names in topology do not match those in coordinate file. What could be wrong?»

If you are using ligand coordinates from PRODRG you have to execute first the following command in the directory where the GRO file (e.g. ligand.gro) for the ligand resides:

```
>editconf –f ligand.gro –o ligand.gro
```

Lic. César L. Araujo
cesar.araujo@oulu.fi
Research Center for Molecular Endocrinology
University of Oulu, Finland

This seems to be silly but it is necessary to avoid those warnings. It seems to be that PRODRG generates GRO files making some mistakes with the atom name field length.

LICENSE

WARRANTY

BUGS

Please feel free to e-mail me to cesar.araujo@oulu.fi with a detailed description of the problem and the version of the script (it is the number in the filename of the script (e.g. **dock41.pl** is intended for version 4.1). In addition, you can fell free to fix them and send me the feedback ☺.

References

1. Berendsen, H. J. C., van der Spoel, D., and van Drunen, R.: GROMACS: A message-passing parallel molecular dynamics implementation. Comput Phys Commun, 91**:** 43, 1995.

2. Sharma, S., Pirilä, P., Kaija, H., Porvari, K., Vihko, P., and Juffer, A. H.: Theoretical Investigations of Prostatic Acid Phosphatase. PROTEINS, 58**:** 295, 2005.