

PRZYGOTOWANIE PLIKÓW DO DYNAMIKI MOLEKULARNEJ

ŁUKASZ CHARZEWSKI

22.12.2015



KLASYCZNA DYNAMIKA MOLEKULARNA

KORZYSTAMY Z RÓWNAŃ RUCHU NEWTONA:

$$\vec{F}_i = m_i \frac{d^2 \vec{r}_i}{dt^2} \quad \vec{F}_i = -\vec{\nabla}_{\vec{r}_i} V(\vec{r}_1, \dots, \vec{r}_N)$$

I WYZNACZAMY TRAJEKTORIE UKŁADU CZĄSTEK W POLU SIŁOWYM

(U NAS – CHARMM27)

KLASYCZNA DYNAMIKA MOLEKULARNA

SYMULACJĘ RUCHU CZĄSTECZEK BIOLOGICZNYCH PROWADZIMY
W WARUNKACH JAK NAJBARDZIEJ ZBLIŻONYCH DO RZECZYWISTYCH:

- W WODZIE
- PRZY ODPOWIEDNIM STĘŻENIU JONÓW
- W RAZIE POTRZEBY - W BŁONIE BIOLOGICZNEJ

Jak taki układ przygotować?

TOPOLOGIA UKŁADU

- W SYMULACJACH PROWADZONYCH METODAMI KLASYCZNEJ DYNAMIKI MOLEKULARNEJ TOPOLOGIA UKŁADU JEST USTALONA I NIEZMIENNA
- PLIKI PDB SIĘ PRZECHOWUJĄ TEJ INFORMACJI
- PO OPRACOWANIU UKŁADU NALEŻY PRZYGOTOWAĆ PLIK ZAWIERAJĄCY OPIS TOPOLOGICZNY (PSF – PROTEIN STRUCTURE FILE)

TOPOLOGIA UKŁADU – PLIK PSF

PSF CMAP

29 !NTITLE

REMARKS original generated structure x-plor psf file

REMARKS 25 patches were applied to the molecule.

REMARKS topology top_all127_prot_na.inp

REMARKS segment MMP3 { first NTER; last CNEU; auto angles dihedrals }

REMARKS defaultpatch NTER MMP3:83

REMARKS patch CNEU MMP3:250

REMARKS patch ASPP MMP3:183

2640 !NATOM

1	MMP3	83	PHE	N	NH3	-0.300000	14.0070	0
2	MMP3	83	PHE	HT1	HC	0.330000	1.0080	0
3	MMP3	83	PHE	HT2	HC	0.330000	1.0080	0
4	MMP3	83	PHE	HT3	HC	0.330000	1.0080	0
5	MMP3	83	PHE	CA	CT1	0.210000	12.0110	0

2684 !NBOND: bonds

1	5	2	1	3	1	4	1
5	6	7	5	7	8	7	9

4799 !NTHETA: angles

1	5	6	1	5	21	2	1	5
2	1	4	2	1	3	3	1	5

7104 !NPHI: dihedrals

1	5	7	10	1	5	7	8
1	5	7	9	1	5	21	23

433 !NIMPHI: impropers

21	5	23	22	23	21	25	24
38	39	42	36	45	25	47	46

TOPOLOGIA UKŁADU – PLIK PSF

- Podział na segmenty – każdy segment należy zapisać jako oddzielny plik PDB:
 - Każde białko osobno
 - Każdy ligand osobno
 - Jony strukturalne – mogą być razem
- Program psfgen – dostarczany razem z NAMDem:
 - `$ module load namd/2.9`
 - `$ psfgen < psf.in > psf.out`
- Powstają 3 pliki:
 - PDB – koordynaty wszystkich atomów
 - PSF – topologia układu
 - OUT – log z przebiegu programu:
 - Atomy, których brakuje w układzie (lub które nie zostały rozpoznane) mogą zostać odgadnięte przez program na podstawie wpisów „internal coordinates” znajdujących się w jednym z plików wejściowych
 - Zgadza się tylko na odgadywanie położenia wodorów.

SKRYPT PSF.IN

```
topology /home/charzewski/wzory/top27rodlip.inp
```

plik zawierający opis topologii każdego residuum w układzie

```
alias residue his hsd  
segment mmp3 {  
  pdb /home/charzewski/uklad/MMP3.pdb }  
alias atom ile cd1 cd  
coordpdb /home/charzewski/uklad/MMP3.pdb mmp3
```

konstrukcja pojedynczego segmentu

```
alias residue his hsd  
segment tim1 {  
  pdb /home/charzewski/uklad/TIMP1.pdb }  
alias atom ile cd1 cd  
patch disu tim1:1 tim1:70  
patch glup tim1:122  
coordpdb /home/charzewski/uklad/TIMP1.pdb tim1
```

zmiana nazwy atomu w całym segmencie

zmiana nazwy residuum w całym segmencie

```
segment jony { auto none  
  pdb /home/charzewski/uklad/jony.pdb }  
coordpdb /home/charzewski/uklad/jony.pdb jony
```

blokuje automatyczne dodawanie C- i N-końca, kątów itd.

modyfikacje pojedynczych aminokwasów

```
guesscoord  
writepdb /home/charzewski/uklad/poPSF.pdb  
writepsf /home/charzewski/uklad/poPSF.psf
```

PLIK INP

- Krótki opis każdego typu atomu
- Masa atomu
- Budowa residuum:

```
RESI ALA          0.00
GROUP
ATOM N    NH1    -0.47  !      |
ATOM HN   H      0.31  !  HN-N
ATOM CA   CT1    0.07  !      |      HB1
ATOM HA   HB     0.09  !      |      /
GROUP      !  HA-CA--CB-HB2
ATOM CB   CT3    -0.27  !      |      \
ATOM HB1  HA     0.09  !      |      HB3
ATOM HB2  HA     0.09  !  O=C
ATOM HB3  HA     0.09  !      |
GROUP      !
ATOM C    C       0.51
ATOM O    O     -0.51
BOND CB CA  N  HN  N  CA
BOND C  CA  C  +N  CA HA  CB HB1  CB HB2  CB HB3
DOUBLE O  C
IMPR N  -C CA HN  C CA +N O
CMAP -C  N  CA  C  N  CA  C  +N
DONOR HN N
ACCEPTOR O C

IC -C  CA  *N  HN    1.3551 126.4900 180.0000 115.4200 0.9996
IC -C  N   CA  C     1.3551 126.4900 180.0000 114.4400 1.5390
```


TOPOLOGIA UKŁADU TOPO/ASF

Skrypt psf.in dla układu ASF/TOPO:

```
topology (ścieżka do pliku topologicznego top_all27_prot.inp)
alias residue his hse
segment (4 literowa nazwa) { pdb (ścieżka do pliku z topoizomerazą) }
alias atom ile cd1 cd
coordpdb (ścieżka do pliku z topoizomerazą) (4 literowa nazwa)
alias residue his hse
segment (4 literowa nazwa) { pdb (ścieżka do pliku z asf) }
alias atom ile cd1 cd
coordpdb (ścieżka do pliku z asf) (4 literowa nazwa)
guesscoord
writepdb (ścieżka do wyjściowego pliku pdb)
writepsf (ścieżka do wyjściowego pliku psf)
```

umownie
nazwijmy te pliki
poPSF.psf i poPSF.pdb

Wyjściowe pliki PDB i PSF należy obejrzyć w VMD

DODAWANIE WODY W VMD

- Znajdując się w odpowiednim katalogu roboczym uruchamiamy VMD należy wpisać w TK Console:

```
package require solvate  
solvate poPSF.psf poPDB.pdb -t 10
```

- Powstają 3 nowe pliki: solvated.psf, solvated.pdb i solvated.log
- Nazwę plików wyjściowych można zmienić stosując parametr:
-o mojaNazwa
- Można także podać dokładne wymiary pudełka z wodą w postaci macierzy min_max lub podać po ile Å od białka chcemy tej wody dodać w każdym wymiarze.

DODAWANIE JONÓW W VMD

- Znajdując się w katalogu z plikami solvated.psf i solvated.pdb należy wpisać w TK Console:

```
package require autoionize
```

```
autoionize -psf solvated.psf -pdb solvated.pdb -is 0.05
```

- Powstają dwa nowe pliki: ionized.psf i ionized.pdb
- Program autoionize wstawia tyle jonów ile trzeba do zneutralizowania ładunku w układzie, a następnie **dodatkowo** wstawia tyle ile wynika z zadanego parametru



siła jonowa
roztworu

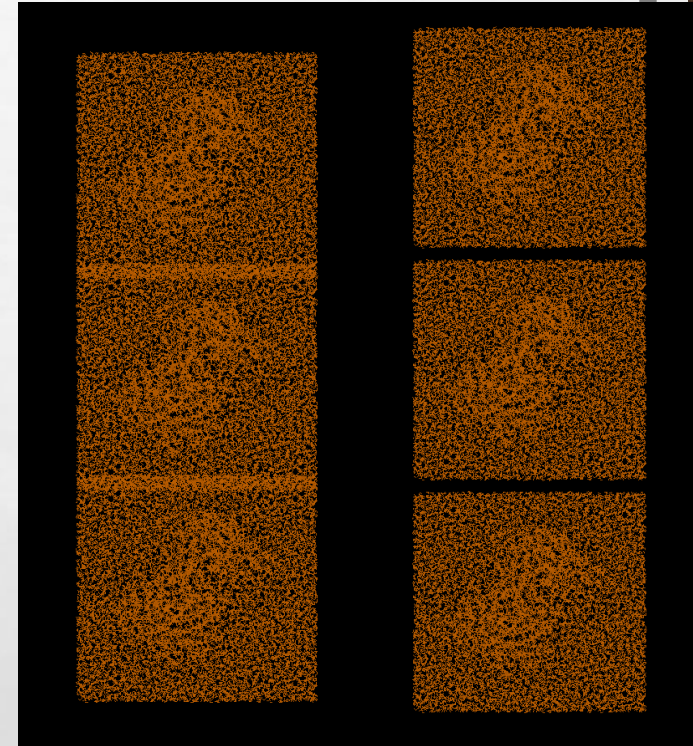
USTALANIE WYMIARÓW KOMÓRKI ELEMENTARNEJ

- Wczytać do VMD pliki ionized.pdb i ionized.psf
- W TK Console należy wpisać:

```
molinfo top get {a b c}
```
- Dostajemy wymiary boxu proponowane przez program –
zwykle są za duże o 1-2 Å. np.:

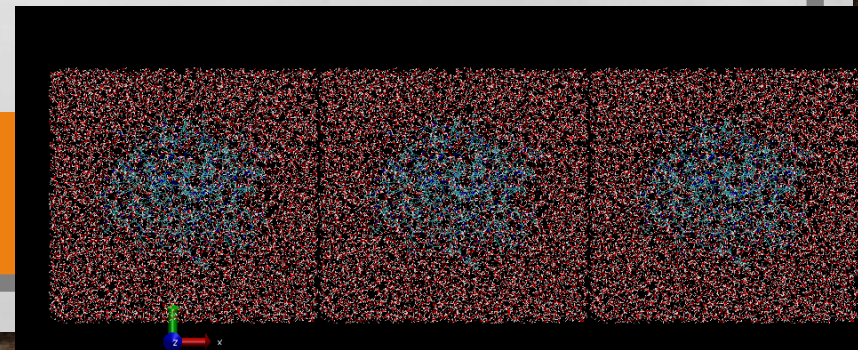
63.672001	60.186001	65.139999
(wymiar X)	(wymiar Y)	(wymiar Z)
- W menu **Graphics/Representations** wybieramy zakładkę **Periodic** i wyświetlamy dodatkowe kopie układu (na razie – te na osi X)
- Sprawdzamy czy wymiary są prawidłowe (powinna być widoczna cienka granica pomiędzy obrazami boksu) – dzięki temu wiemy, że cząsteczki nie nakładają się na siebie, oraz że nie ma między nimi próżni.
- W razie potrzeby podajemy nowy wymiar komórki, np.:

```
molinfo top set a 62.96
```



Przydatne polecenie TK Console:

```
rotate x by 90
```



USTALANIE WYMIARÓW KOMÓRKI ELEMENTARNEJ

- Potrzebne są jeszcze współrzędne środka układu:

```
% molinfo top measure center
```

- Zapisujemy wszystkie informacje do pliku xsc:

```
# NAMD extended system configuration output file
```

```
#$LABELS step a_x a_y a_z b_x b_y b_z c_x c_y c_z o_x o_y o_z
```

```
0000 64.85 0 0 0 75.18 0 0 0 74.000 86.687 29.248 -0.395
```

↑	↑	↑	↑	↑
krok				
symulacji	wymiar x	wymiar y	Wymiar z	Współrzędne
	komórki	komórki	komórki	środką komórki

ETAPY SYMULACJI

Symulację można podzielić na trzy etapy (każdy składa się z minimalizacji i dynamiki molekularnej):

1. Rusza się tylko woda z jonami
2. Rusza się woda, jony i reszty boczne aminokwasów (poza tymi, które tworzą modelowane oddziaływania)
3. Rusza się wszystko

Potrzebne są dwa pliki REF, które zawierają informację co podczas symulacji ma się ruszać, a co nie.

Wykorzystujemy do tego kolumnę beta w pliku typu PDB:

- 0 – ruch
- 1 – zamrożenie

FIXOWANIE ATOMÓW W VMD

- Wczytać ionized.psf i ionized.pdb do VMD
- W TK Console należy wpisać:

```
% set all [atomselect top all]
% $all set beta 0
% set prot [atomselect top „segname TOPO ASF”]
% $prot set beta 1
% $all writepdb fixprot.ref
```

zamrożone
całe białko

```
% $all set beta 0
% set bb [atomselect top „name CA C O N HN”]
% $bb set beta 1
% set oddtopo [atomselect top „segname TOPO and resid 345 346”]
% $oddtopo set beta 1
% set oddasf [atomselect top „segname ASF and resid 134 135”]
% $oddasf set beta 1
% $all writepdb fixbb.ref
```

zamrożony
łańcuch główny
oraz wszystkie
wymodelowane
oddziaływania

0 – ruch
1 – zamrożenie

PLIKI WEJŚCIOWE DO DYNAMIKI


- Struktura i topologia układu – ionized.pdb, ionized.psf
- Pliki konfiguracyjne:
 - białkofix_min.conf
 - białkofix_dyn.conf
 - bbfix_min1.conf
 - bbfix_dyn1.conf
 - bialkowolne_min2.conf
 - bialkowolne_dyn2.conf
- Pliki blokujące ruch wybranych atomów:
 - fixprot.ref do symulacji białkofix_min.conf i białkofix_dyn.conf
 - fixbb.ref do symulacji bbfix_min1.conf i bbfix_dyn1.conf
- Plik z wymiarami komórki elementarnej – nazwa.xsc
- Pliki uruchamiające symulację:
 - runM runM1 runM2 runD runD1 - muszą być wykonywalne! (`chmod a+x run*`)
 - wsadMPI wsadCUDA - wsadowe do systemu kolejkowego SLURM klastra

PLIKI KONFIGURACYJNE

```
numsteps          20000
coordinates        min.coor
extendedSystem    xyz.xsc
outputname        dyn
structure         ionized.psf

PME               on
PMEGridSizeX      96
PMEGridSizeY      160
PMEGridSizeZ      90

if {1} {
fixedAtoms        on
fixedAtomsFile    fixbb.ref
fixedAtomsCol     B
}
```



Liczby dla każdego z wymiarów komórki elementarnej. Najmniejsza możliwa, która jest większa od wymiaru boku i podzielna tylko przez 2, 3, 5 i 7

PRZEBIEG SYMULACJI

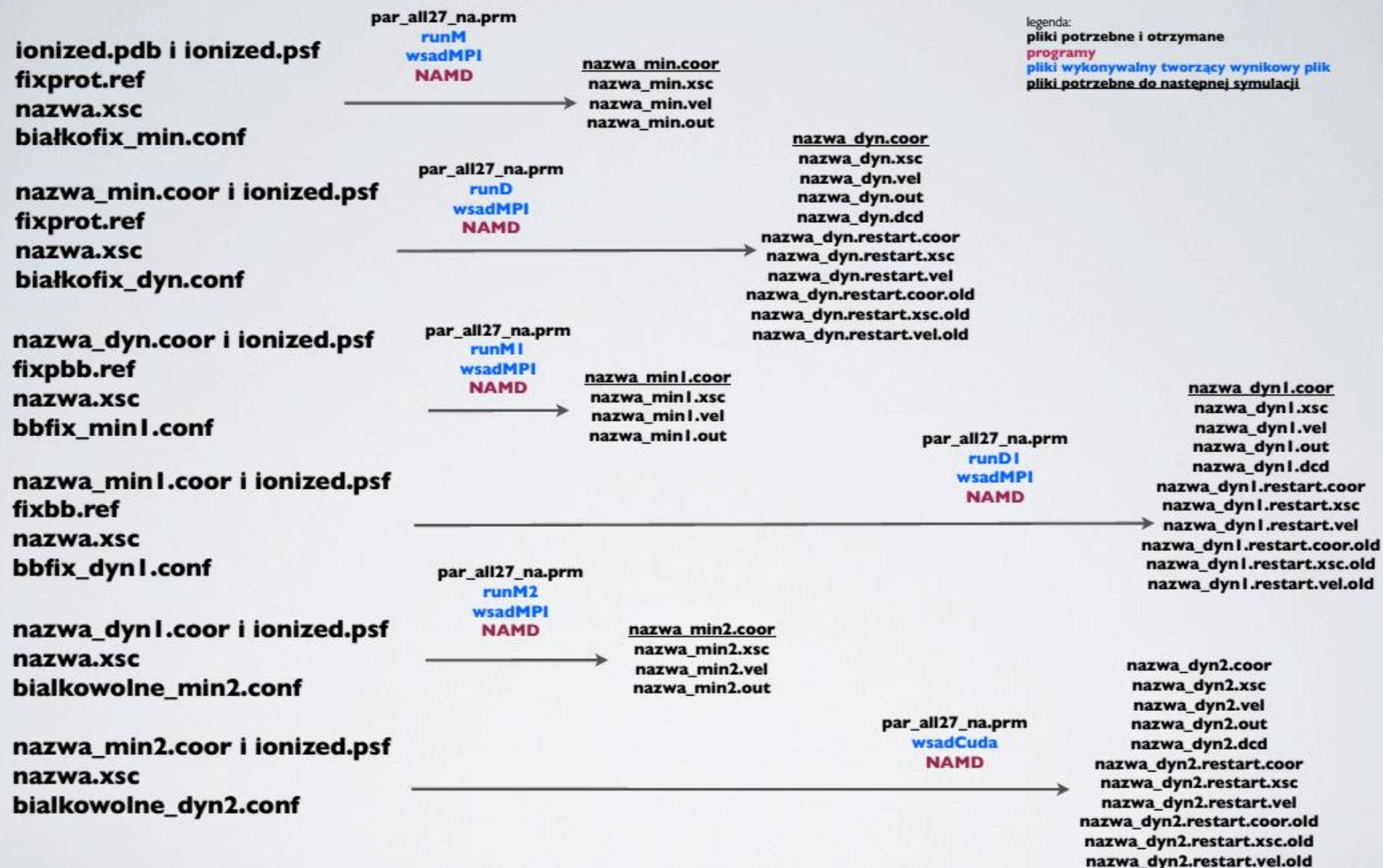
legenda:

pliki potrzebne i otrzymane

programy

pliki wykonywalny tworzący wynikowy plik

pliki potrzebne do następnej symulacji



SYSTEM KOLEJKOWY SLURM

- Polecenia są wykonywane na maszynach msys1-msys28. Można się na nie dostać przez SSH, ale wybór węzła pozostawiamy SLURMowi
- Logowanie na klaster: `ssh login@bioexploratorium.pl`
- Podstawowe polecenia SLURMa:
 - **squeue** – wyświetla kolejkę zadań
 - **scancel *JOBID*** – przerywa wykonywanie wskazanego zadania
 - **srun *polecenie*** – uruchamia polecenie interaktywnie
 - **sbatch *plik_wsadowy*** – uruchamia skrypt z pliku
- Standardowe wyjście i standardowe wyjścia błędów są przekierowywane do wskazanych plików (tu: stdout i stderr)
- Wszystkie etapy uruchamiamy korzystając z polecenia sbatch:
`$ sbatch wsadMPI`
- Plik wsadMPI przeprowadzi kolejno wszystkie wstępne etapy symulacji. W razie potrzeby odpowiednie linijki można usunąć lub zakomentować znakiem #.
- Plik wsadCUDA uruchamia ostatni etap dynamiki molekularnej korzystając ze wsparcia GPU.

USTALANIE LICZBY KROKÓW MINIMALIZACJI

- Liczba kroków we wszystkich dynamikach zostaje bez zmian, natomiast należy poprawić liczbę kroków w min1 i min2. W tym celu korzystamy z plików out z poprzednich etapów minimalizacji (odpowiednio min i min1)
- Po skończonej minimalizacji sprawdzamy jaki jest gradient energii w pliku min.out, chcielibyśmy aby był 0,01 (reszta miejsc po przecinku nie jest ważna)
- Jeśli gradient bardzo odbiega od tej wartości, należy zwiększyć liczbę kroków w następnej minimalizacji (min1) i po skończonej symulacji znowu sprawdzamy gradient (min1.out)
- Jeśli gradient bardzo odbiega od tej wartości, należy zwiększyć liczbę kroków w następnej minimalizacji (min2) i po skończonej symulacji znowu sprawdzamy gradient (min2.out)
- Jeśli gradient odbiega od tej wartości, należy minimalizować układ (uruchamiać kolejne symulacje minimalizacji – min2a itd.), aż po skończonej symulacji uzyskamy gradient na poziomie 0,01.

LINE MINIMIZER REDUCING GRADIENT FROM 88.0673 TO 0.0880673

DZIĘKUJĘ ZA UWAGĘ!

