A practical guide to computer simulation II

Alexander K. Hartmann, University of Göttingen

June 18, 2003

7 Data analysis

7.1 Data plotting

Two programs discussed here: *gnuplot* and *xmgrace*.

	gnuplot	*xmgrace*
user interface	terminal, output to extra window	windows based
speed of use	fast	medium
scripts	good	not so good
data fitting	very good	medium
output quality	medium	good
flexibility	medium	very good

7.1.1 Using *gnuplot*

Invoke: Call `gnuplot` in a shell. A prompt `gnuplot >` appear, commands can be entered.
Plotting a x-y-dy file, e.g. `sg_e0_L.dat` (Ground state energy of a spin glass as a function of system size). File format: 1 point per line, comments (starting with `#`) are ignored, e.g.:

```
# ground state energy of +-J spin glasses
# L    e_0   error
  3 -1.6710 0.0037
  4 -1.7341 0.0019
  ...
 14 -1.7866 0.0007
```

```
gnuplot> plot "sg_e0_L.dat" with yerrorbars
```

(short:) `p "sg_e0_L.dat" w e`

→ Window with plot pops up.

Other styles possible, e.g. `with lines`. Plotting multicolumn files, e.g.

```
gnuplot> plot "test.dat" using 1:4:5 w e
```

More information: enter `help plot`.

3d plots: `splot`.

Setting labels:

```
gnuplot> set xlabel "L"
```

and issue plot command again, or use `replot`.
To write to postscript file:

```
gnuplot> set terminal postscript
gnuplot> set output "test.eps"
```

and plot again.

Scripts = ASCII files (e.g. `commands.gp`) with gnuplot commands. Can e.g. be generated automatically by other scripts. Invoke `gnuplot command.gp` from a shell.

7.1.2 Using *xmgrace*

Call `xmgrace` and a window will pop up. Commands: via menus.

Loading data to display: use menu **Data** — **Import** — **ASCII ...**. Choose *single set* or *block data* (where some columns among many are selected).

Styles, lines, symbols, sizes, error bars, colors, legend strings etc can be adjusted all and independently of each others. Use menu **Plot**— **Set appearance**.

For strings like legends, axis labels etc there are some special formatting sequences, e.g.:

- `\x` greek font
- `\1` italic font
- `\4` sans serif font
- `\S` change to superscript
- `\N` change back to normal script
- `\s` subscript

To change position, style, size etc of legend: menu **Plot**— **Graph appearance**

To change axis labels, tick marks, sizes, etc: use menu **Plot**— **Axis properties**.

To combine several plots into one (e.g. insets): menu **Edit**— **arrange graphs**.

To print: Menu **File** — **Print setup**.

7.2 Fitting data

Fitting = match parametrized function to data points via changing parameters.

Here: Using gnuplot: Example: fit function $f(L) = e_\infty + aL^b$ to spin-glass energy data. Define function + starting values:

```
gnuplot> f(x)=e+a*x**b
gnuplot> e=-1.8
gnuplot> a=1
gnuplot> b=-1
```

Fit used Marquardt-Levenberg algorithm [2] algorithm. Command `fit`. One has to provide function, data set and state the parameters with are variable. E.g.:

```
gnuplot> fit f(x) "sg_e0_L.dat" via e,a,b
```

Gnuplot writes log informations (not shown) and result:

```
After 17 iterations the fit converged.
final sum of squares of residuals : 7.55104e-06
rel. change during last iteration : -2.42172e-09

degrees of freedom (ndf) : 5
rms of residuals      (stdfit) = sqrt(WSSR/ndf)      : 0.00122891
variance of residuals (reduced chisquare) = WSSR/ndf : 1.51021e-06

Final set of parameters            Asymptotic Standard Error
=======================            ==========================


e               = -1.78786         +/- 0.0008548    (0.04781%)
a               = 2.5425           +/- 0.2282       (8.976%)
b               = -2.80103         +/- 0.08265      (2.951%)


correlation matrix of the fit parameters:

               e      a      b
e              1.000
a              0.708  1.000
b             -0.766 -0.991  1.000
```

Estimate quality of fit: 3 lines after “`degree of freedom`”.

- *Degrees of freedom* = number of data points minus the number of parameters in the fit.
- `WSSR` means the deviation of the fit function $f(x)$ from the data points $(x_i, y_i \pm \sigma_i)$ $(i = 1, \ldots, N)$ is given by $\chi^2 = \sum_{i=1}^N \left[\frac{y_i - f(x_i)}{\sigma_i}\right]^2$.
- Calculate *quality* Q = probability that the value of χ^2 is worse than in the current fit, given the assumption that the data points y_i are Gaussian distributed with mean $f(x_i)$ and variance one [2]. The larger the value of Q, the better is the quality of the fit. To calculate Q you can use the little program `Q.c`

```
#include <stdio.h>
#include "nr.h"
int main(int argc, char **argv)
{
  float ndf, chi2_per_df;
  sscanf(argv[1], "%f", &ndf);
  sscanf(argv[2], "%f", &chi2_per_df);
  printf("Q=%e\n", gammq(0.5*ndf, 0.5*ndf*chi2_per_df));
  return(0);
}
```

uses `gammaq` from *Numerical Recipes* [2]. Program is called in the form `Q <ndf> <WSSR/ndf>`.

Note 1: Convergence depends on choice of initial parameters, algorithm may be trapped in local minimum (e.g. try e=1,a=-3,b=1)
Note 2: In the example above all values entered with the same weight. To include also error bars: `fit f(x) "sg_e0_L.dat" using 1:2:3 via a,b,c`

xmgrace: Menu **Data** — **Transformations** — **Non-linear curve fitting**. Enter formula `y = a0+a1*x**a2`, choose parameters = 3 and enter starting values `a0=-1.8`, `a1=1`, `a2=-1`. Select a set (to which function is fitted) and hit on **Apply**. Result: in pop-up window + fitting curve as a new set. Note: *NO* error bars for resulting parameters.

7.3 Finite-size scaling

Statistical physics: thermodynamic limit ($N \to \infty$). In simulations: only small finite systems. Solution: *finite-size scaling* = study *several* small system sizes.

Example: spin glass $H = -\sum_{\langle i,j\rangle} J_{ij} S_i S_j$ with fraction p of antiferromagnetic $J_{ij} = -1$ and fraction $(1-p)$ of ferromagnetic $J_{ij} = 1$ bonds. Calculation of ground states + evaluation of magnetisation $m = |\sum_i S_i/N|$ as a function of p. We look for transition ferromagnet ($m > 0$, $p < p_c$ small) to spin glass ($m = 0$, $p > p_c$ "large".). Results for different system sizes:
Theory of finite-size scaling [3]:

$$m(p,L) = L^{-\beta/\nu}\tilde{m}(L^{1/\nu}(p-p_{\mathrm{c}})) \tag{1}$$

$\tilde{m}$ is a universal, i.e. non size-dependent, function. β, ν = *critical exponents* describing the algebraic behavior of the magnetisation and correlation length near p_c.

Obtain p_c, ν, β from plotting $L^{\beta/\nu} m(p,L)$ against $L^{1/\nu}(p-p_{\mathrm{c}})$. If parameters are correct → data collapse.

Using gnuplot: input file `m_scale.dat` with three columns $L, p, m(p,L)$, then:

```
gnuplot> b=0.27
gnuplot> n=1.1
gnuplot> pc=0.222
gnuplot> plot [-1:1] "m_scale.dat" u (($2-pc)*$1**(1/n)):($3*$1**(b/n))
```

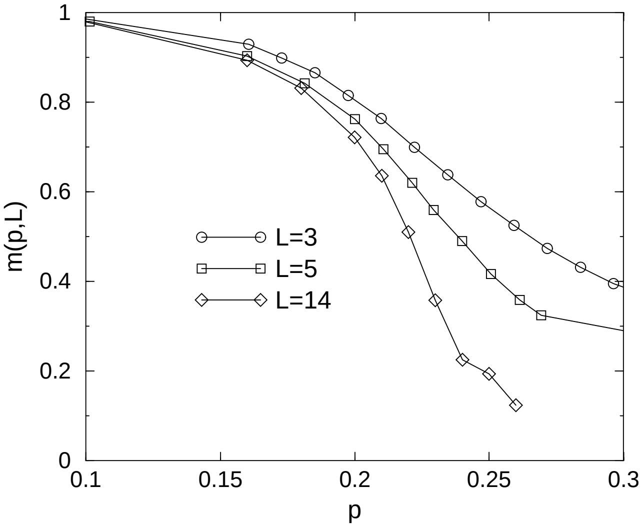

Figure 1: Average ground-state magnetisation m of a three-dimensional ±J spin glass with fractions p of antiferromagnetic bonds. Lines are guides to the eyes only.

Result:
Normally: Parameters are not know. Start with reasonable assumptions and change iteratively until good data collapse obtained.
Hint: more convenient is *fsscale* [4]. One can change the scaling exponents interactively via keys.

7.4 Resampling

Given N uncorrelated random number x_i.
Repetition: Estimator for average $\langle x \rangle$:

$$\overline{x} = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{2}$$

Estimator for error bar (65%):

$$\sigma_{\overline{x}} = \sqrt{[\overline{x^2} - \overline{x}^2]/(N-1)} \tag{3}$$

Problem: combined measures $g(x_1, \ldots, x_N)$ like Binder parameters

$$g = 0.5(3 - \frac{\overline{x^4}}{\overline{x^2}^{\,2}}) \tag{4}$$

How to calculate error bars?
Solution: Jackknife method [1]. *Define*

$$g_i^J = g(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N) \tag{5}$$

i.e. the average omitting data point i.
One can show that

$$[g]_J \equiv \frac{1}{N} \sum_i g_i^J \rightarrow \langle g \rangle \tag{6}$$

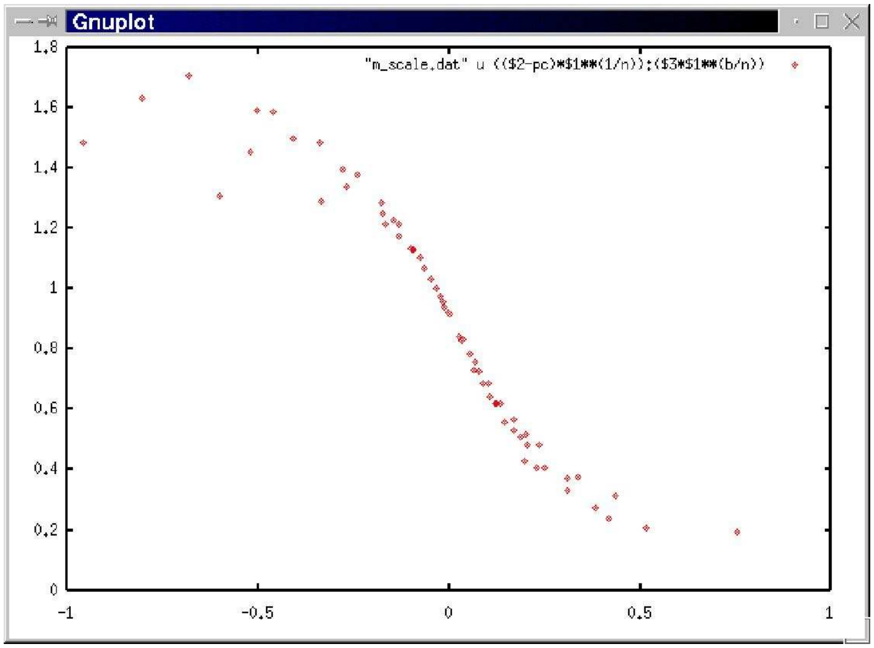

Figure 2: *Gnuplot* output of a finite-size scaling plot. The ground-state magnetisation of a three-dimensional $\pm J$ spin glass as a function of the concentration p of the antiferromagnetic bonds is shown. For the fit, the parameters $p_{\rm c} = 0.222, \beta = 0.27$ and $\nu = 1.1$ have been used.

Estimator for error bar is:

$$\sigma_{g^J} = \sqrt{(N-1)\left([g^2]_J - [g]_J^2\right)} \tag{7}$$

Remark: It is usually sufficient to divide the data into K blocks and omit each time one of the blocks.

8 Exercise

Use the program for ballistic deposition to record the average (1000 realizations) roughness $W(L,t)$ for different system sizes ($L = 10, 20, 40, 100$) as a function of time (up to 100 sweeps). One expects:
$W \sim t^\beta$ for small times $t < t_X(L)$ and $W \sim L^\alpha$ for large times $t < t_x(L)$. This combines to $W(L,t) \sim L^\alpha f(t/L^z)$ $(\star)$ with $z = \alpha/\beta$.
First plot $W(t)$ for different L.
Then use gnuplot to obtain β from fitting $W(t)$ for different sizes and make a scaling plot for $(\star)$.
Hint: `fit [10:100] f(x) "data.dat"` ... fits the data only in the range $[10:100]$.
[Result: from fit β around 0.25 (true: $\beta = 1/3$), good collapse for $z = 1.2$, $\alpha = 0.37 \to \beta = \alpha/z = 0.31$]

References

[1] B. Efron, *The Jackknife, the Bootstrap and Other Resampling Plans*, SIAM 1982

[2] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, *Numerical Recipes in C* (Cambridge University Press, Cambridge 1995)

[3] K. Binder and D.W. Heermann, *Monte Carlo Simulations in Statistical Physics*, (Springer, Heidelberg 1988)

[4] The program *fsscale* is written by A. Hucht, see http://www.thp.uni-duisburg.de/fsscale/