Applied Stochastic Analysis HW Report (Ising model)

Zixian Zhou
1500010607

January 4, 2019

1 Metropolis for Ising Model

In order to generate the Gibbs distribution and therefore study related variables, we implemented the classical Metropolis algorithm with C++.

Algorithm 1 Metropolis (original)

Require: β, J, h
Initialize spins s_i with 1 or -1 equally randomly.
Define $H(\sigma) = -J\sum_{<i,j>}\sigma_i\sigma_j - h\sum_i \sigma_i$.
repeat
1. Generate the proposal sate σ'.
2. Compute $\Delta H = H(\sigma') - H(\sigma_n)$, $A = \min\{1, \exp(-\beta\Delta H)\}$
3. Generate R.V. $r \sim \mathcal{U}[0,1]$.
4. If $r \leq A$, then $\sigma_{n+1} = \sigma'$; else $\sigma_{n+1} = \sigma_n$.

until convergence

Here the generating stage is chosen as uniformly randomly flipping one spin. (In order to save memory and speeding up, we keep each spin as a boolean instead of $+1$ and -1.) The convergence is judged mainly according to whether the macro-variables such as U, C, etc. get stable and smooth with regard to T.

We only studied the 2D case. The grid's size is $N \times N$, where N is set to 16 or 32. The program uses uniformly random states as the initial stage. To ensure convergence and thus getting the correct average energy U and heat capacity C, we always perform some warm-up transformations before estimating U and C for any new β. In detail, the first warm-up performs 10^7 transformations, short warm-ups 5×10^6 and sampling are done with 10^7 transformations in each situation.

In this way we generate a series of stages Σ_t, which will be used to study the following problems.

1.1 (a). Plot u and c

$$u = \frac{U}{N^2}, U = \langle H \rangle$$

$$c = \frac{C}{N^2}, C = k_B \beta^2 \mathrm{Var}(H) = k_B \beta^2 (\langle H^2 \rangle - \langle H \rangle^2)$$

The 2D situation has been solved exactly by Onsager (1944). The critical temperature should be

$$T^* = \frac{2J}{k \ln(1+\sqrt{2})}$$

. In our case $J = 1, k = 1$, we should have $T* = \frac{2}{\ln(1+\sqrt{2})} \approx 2.27$.

In the numerical experiment, we estimate u, c by

$$u \approx \frac{1}{N^2 T} \sum_{t=1}^{T} H(\Sigma_t), c \approx \frac{k_B \beta^2}{N^2} [\frac{1}{T} \sum_{t=1}^{T} H(\Sigma_t)^2 - (\frac{1}{T} \sum_{t=1}^{T} H(\Sigma_t))^2]$$

It is too slow to directly calculate H every time we get a new stage. Instead, we update it according to the local change around the flipped spin. In detail, when flipping a spin s, we have

$$\Delta H = 2J\sigma_s(2n_{nb} - 4) + 2h\sigma_s$$

, where n_{nb} means the number of s's neighboring spins equaling to 1. Here σ_s is the state of s before being flipped, same as in the following formulas.

1.2 (b). Plot the Magnetization

$$m = \frac{M}{N^2}, M = \langle \sum_i \sigma_i \rangle$$

In the numerical experiment, we estimate m by

$$m \approx \frac{1}{N^2 T} \sum_{t=1}^{T} M(\Sigma_t)$$

Similarly as previous H, we update M when a spin s is flipped:

$$\Delta M = -2\sigma_s$$

1.3 (c). Spatial Correlation Function

$$\Gamma(r) = \langle \rho(0)\rho(r) \rangle$$

(Note that $\langle \rho(0) \rangle \langle \rho(r) \rangle = M^2 = 0$ when $h = 0$.) Here we approximate $\Gamma(r)$ by considering the spin correlations in the horizontal direction, and the starting point $\rho(0)$ is chosen by taking average with respect to all possible sites.

Denote

$$S(\Sigma, r) = \sum_i \sigma_i \sigma_{i[r]}$$

, where $s[r]$ is the rth spin away from s on the horizontal direction. Then

$$\Gamma(r) = \frac{\langle S(r) \rangle}{N^2}, \Gamma(r) \approx \frac{1}{N^2 T} \sum_{t=1}^{T} S(\Sigma_t, r)$$

Similarly as previous H, M, we update $\Gamma(r)$ when a spin s is flipped:

$$\Delta\Gamma(r) = -2\sigma_s(\sigma_{s[-r]} + \sigma_{s[r]})$$

(Wikipedia) At high temperatures exponentially-decaying correlations are observed with increasing distance, with the correlation function being given asymptotically by

$$\Gamma(r) \approx \frac{1}{r^{d-2+\eta}} \exp\left(\frac{-r}{\xi}\right)$$

Here our $d = 2$. Note that scaling of r (multiplying it with a constant) would change the value, so we should have

$$\ln \Gamma(r) \approx -\eta \ln r - \frac{r}{\xi} + C$$

, where C is a constant depending on the temperature and the scale.

To study ξ numerically, we run a linear regression on $\ln \Gamma(r)$ with regard to $\ln r$ and r with intercept. Then the coefficient of r is an approximation of ξ.

1.4 (d). Find the Scaling Exponents

We assume the limiting behavior

$$m \sim m_0 \epsilon^{\alpha}, c \sim c_0 \epsilon^{-\gamma}, \xi \sim \xi_0 \epsilon^{-\delta}$$

, where $\epsilon = |1 - T/T_*|$. We calculate the scaling exponents α, γ, δ using the data obtained in the previous 3 problems using the same method as mentioned above in problem (c).

2 Advanced Methods

We applied the kinetic Monte Carlo method (KMC), Swendsen-Wang algorithm (SW) and Wolff's algorithm (Wolff) on these problems. Unfortunately, SW and Wolff requires $h = 0$. So these two are not applied to problem (b) which studies M under different hs.

2.1 Kinetic Monte Carlo method (KMC)

Algorithm 2 Kinetic Monte Carlo method (KMC)

Require: β, J, h

Initialize spins s_i with 1 or -1 equally randomly.

Define $H(\sigma) = -J \sum_{<i,j>} \sigma_i \sigma_j - h \sum_i \sigma_i$.

Calculate $p_j = \min\{1, \exp(-\beta \Delta H_j)\}, j = 1, 2, \ldots, 10$

Count each spin's number of neighbors equaling to 1 and classify the spins into 10 classes, and saving the size of the classes as $n_j, j = 1, \ldots, 10$.

repeat

1. Calculate $Q_i = \sum_{j=1}^{i} n_j P_j, i = 1, \ldots, 10$.
2. Generate R.V. $R \sim \mathcal{U}[0, Q_{10})$.
3. Identify $Q_{i-1} \leq R < Q_i, (Q_0 = 0)$.
4. Generate an integer I uniformly in $\{1, \ldots, n_i\}$.
5. Search the spins in the regular order and flip the Ith spin of class i.
6. Update n_j, and number of neighbors.
7. Generate R.V. $R' \sim \mathcal{U}[0, 1]$.
8. Set time increment as $-\frac{1}{Q_{10}} \ln R'$.

until convergence

The time increment can be multiplied by an arbitary constant since it only serves as a weight of the previous state. For controling running time, here it is multiplied by N^2, since it's easy to observe that $Q_{10} \sim N^2$ and the searching step (step 5) also takes $O(N^2)$ time.

2.2 Swendsen-Wang algorithm (SW)

Algorithm 3 Swendsen-Wang algorithm (SW)

Require: $\beta, J, h = 0$

Initialize spins s_i with 1 or -1 equally randomly.

repeat

1. For a given configuration of the spins, form the bond variable by giving every edge of the lattice $< i, j >$, between two like spins ($s_i = s_j$) a bond value of 1 with probability $1 - \exp(2\beta J)$, and a bond value of 0 otherwise.

2. Flip each cluster with probability 0.5.

until convergence

However, this is a rough decription and not very concrete. We implement the two steps in the loop in the following way.

Algorithm 4 SW - Detailed Steps in the Loop

Form and save all the bond variables. Given every edge of the lattice $< i, j >$, if $s_i = s_j$, then set the bond value as 1 with probability $1 - \exp(2\beta J)$, and as 0 otherwise.

Set all the spins as unchecked.

for each unchecked spin s in the lattice **do**

Init a set $C_1 = \{s\}$ and denote $C_0 = \emptyset$.

Set s as checked.

repeat

Init $C_{i+1} = C_i$

for each unchecked neighboring spin s_{nb} of $s \in C_i \backslash C_{i-1}$ **do**

If the bond value between s_{nb} and s equals 1, then add s_{nb} to C_{i+1} and set s_{nb} as checked.

end for

until $C_{i+1} \backslash C_i = \emptyset$

Flip all the spins corresponding to the sites in the final set C_n with probability 0.5.

end for

2.3 Wolff's algorithm (Wolff)

Algorithm 5 Wolff's algorithm (Wolff)

Require: $\beta, J, h = 0$
  Initialize spins s_i with 1 or -1 equally randomly.
  repeat
    Randomly picks a spin s. Init a set $C_1 = \{s\}$ and denote $C_0 = \emptyset$.
    Set all the spins as unchecked except s.
    repeat
      Init $C_{i+1} = C_i$
      for each unchecked neighboring spin s_{nb} of $s \in C_i \backslash C_{i-1}$ **do**
        Form the bond between s_{nb} and s a bond value of 1 with probability $1 - \exp(2\beta J)$ if $\sigma_{s_{nb}} = \sigma_s$, and a bond value of 0 otherwise.
        If the bond value equals 1, then add s_{nb} to C_{i+1} and set s_{nb} as checked.
      end for
    until $C_{i+1} \backslash C_i = \emptyset$
    Flip all the spins corresponding to the sites in the final set C_n.
  until convergence

Since SW and Wolff both involves flipping more than one spins at one time, if the number of spins to be flipped is small, we flip them one by one and update the variables (that is, H, M, or $S(r)$) concerned at each step, else flip them all and calculate the new variables by definition. Here we take $N^2/4$ as the threshold.

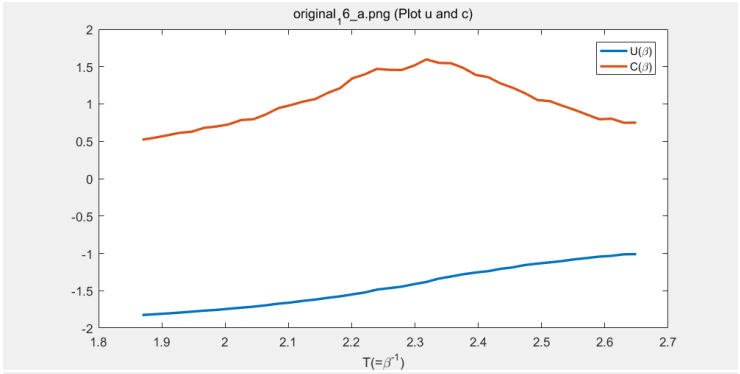
In order to make the four algorithms (original, KMC, SW, Wolff) run for roughly the same time, the time gap between Σ_t and Σ_{t+1} is

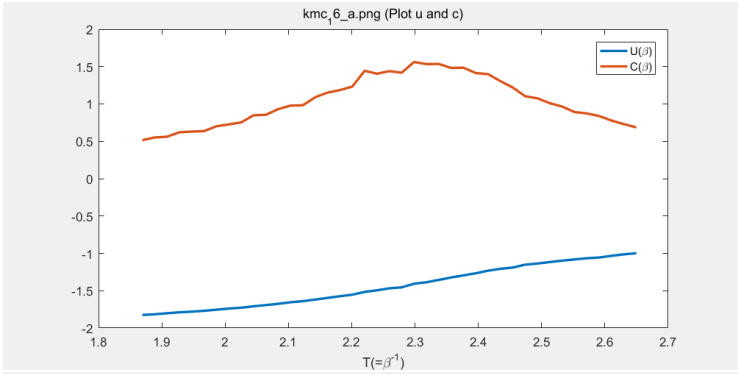
algorithm	original	KMC	SW	Wolff
time gap	1.0	$-\frac{N^2}{Q_{10}} \ln R'$	$4N^2$	$\frac{4N^2}{1+\exp(-30\beta+13)}$

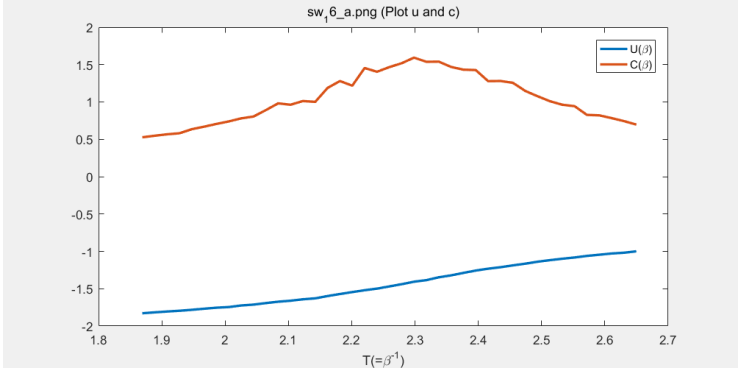
The time gap for Wolff's algorithm depends on β because the number of spins flipped at each stage depends largely on β and we use a sigmoid function to automatically adjust its total stage number (T). Since the number of spins flipped at a time ranges from 1 to N^2, it seems a reasonable choice.

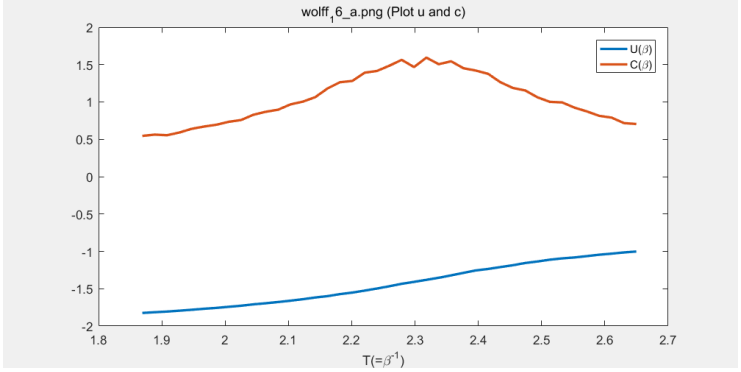
3 Numerial Results

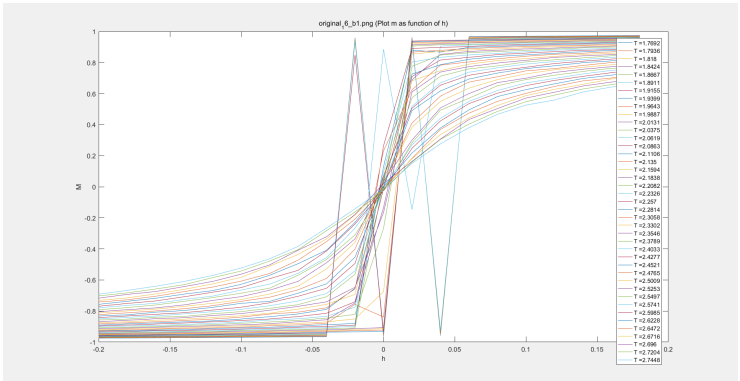
3.1 $N = 16$

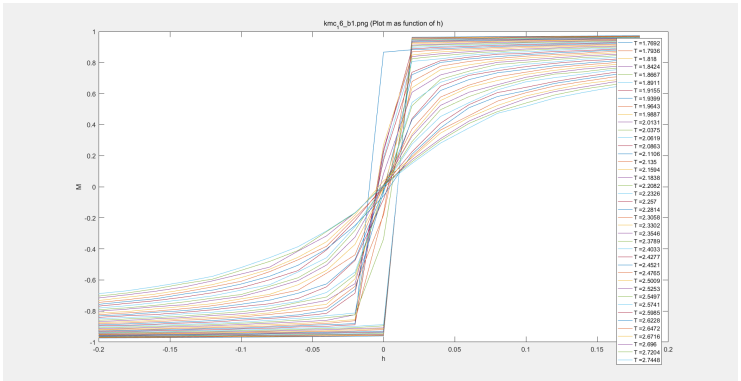


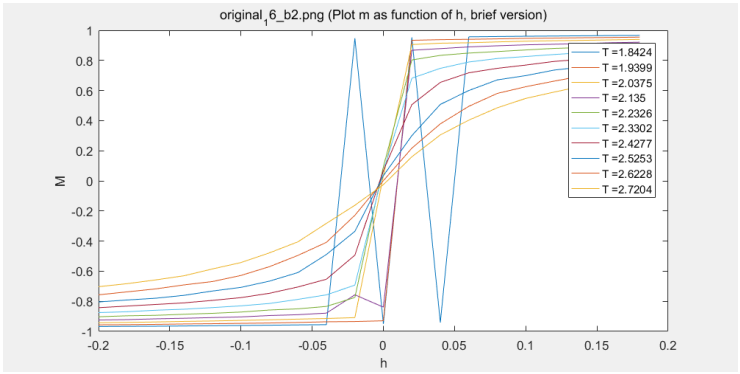


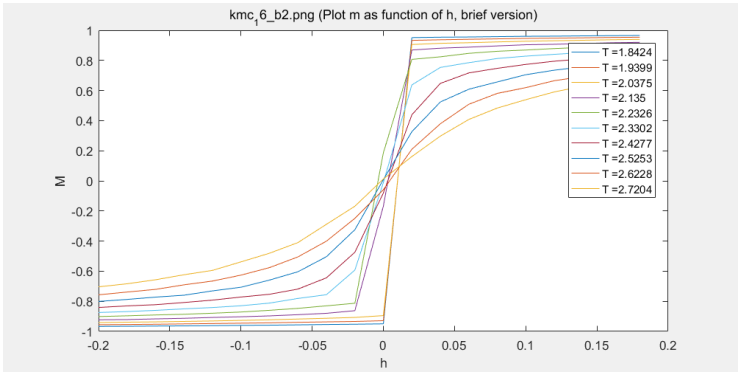


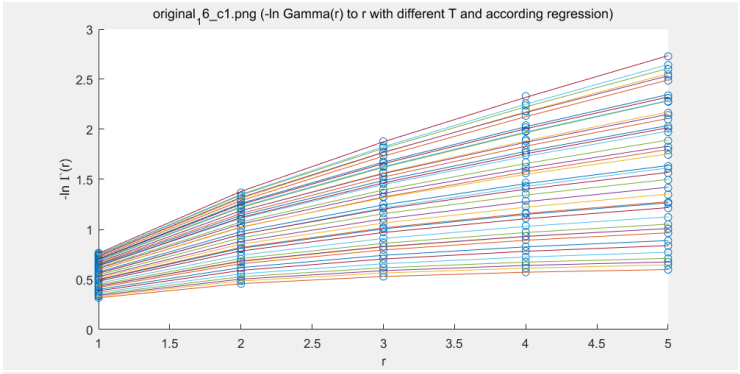


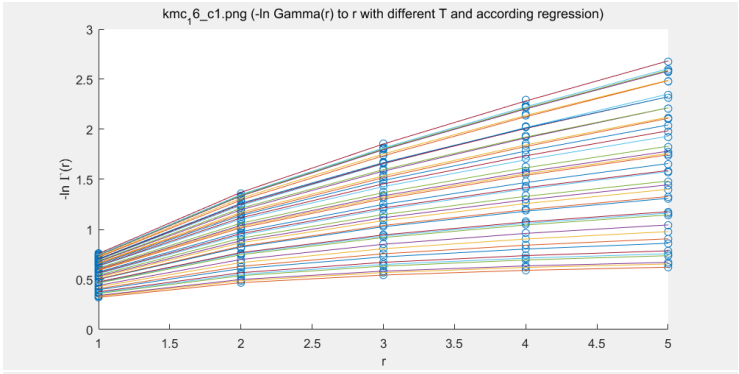


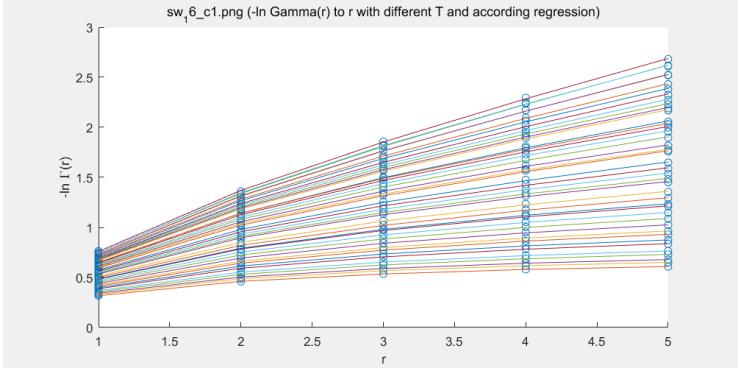


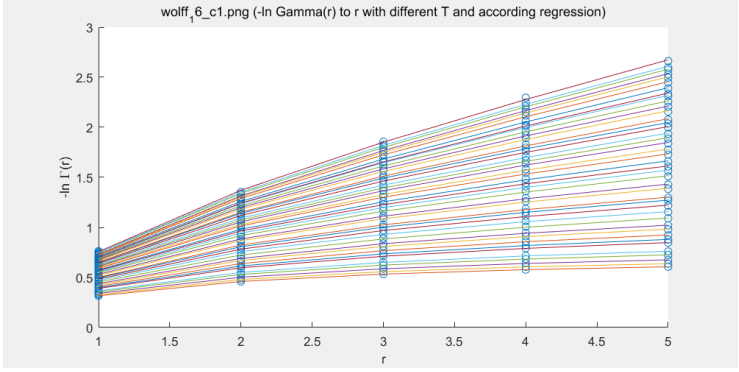

original_16_b2.png (Plot m as function of h, brief version)
M
h
T =1.8424
T =1.9399
T =2.0375
T =2.135
T =2.2326
T =2.3302
T =2.4277
T =2.5253
T =2.6228
T =2.7204

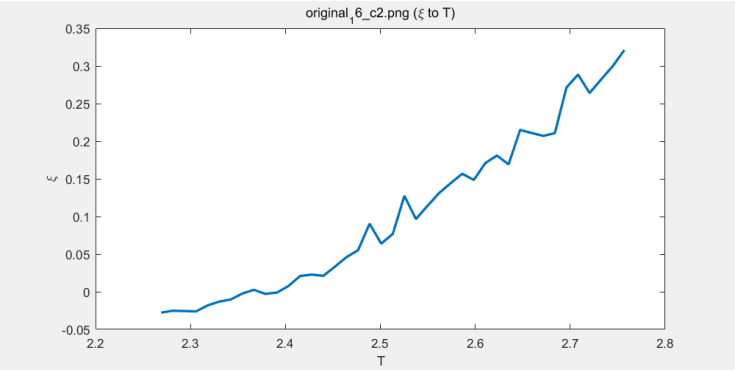

kmc_16_b2.png (Plot m as function of h, brief version)
M
h
T =1.8424
T =1.9399
T =2.0375
T =2.135
T =2.2326
T =2.3302
T =2.4277
T =2.5253
T =2.6228
T =2.7204

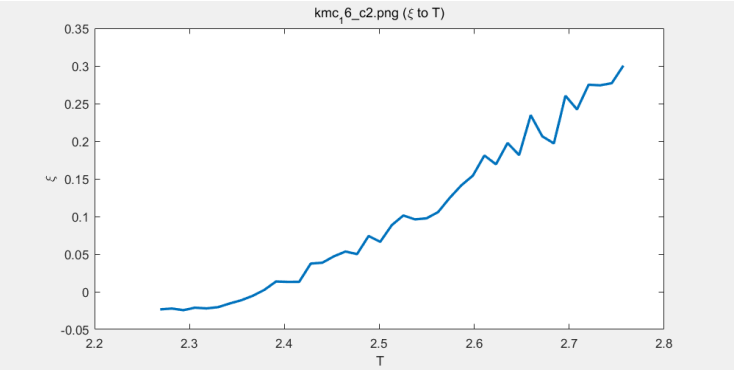

original_16_c1.png (-ln Gamma(r) to r with different T and according regression)
-ln Γ(r)
r

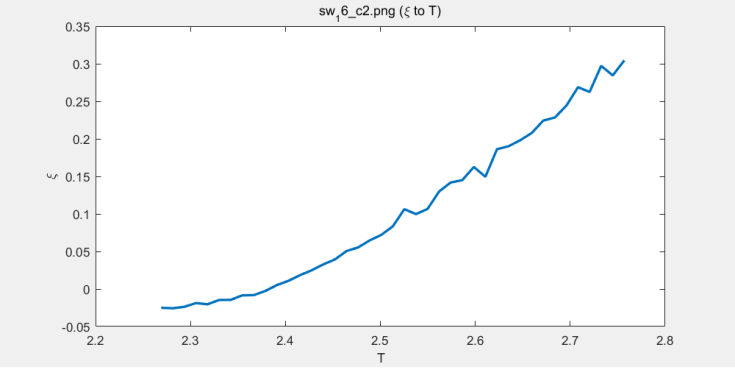

kmc_16_c1.png (-ln Gamma(r) to r with different T and according regression)
-ln Γ(r)
r

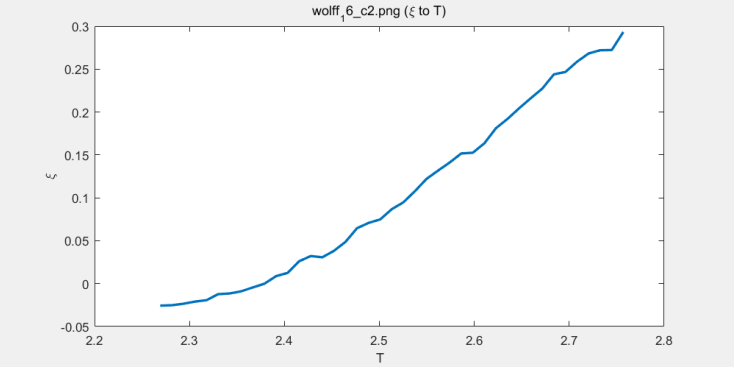

sw_16_c1.png (-ln Gamma(r) to r with different T and according regression)
-ln Γ(r)
r

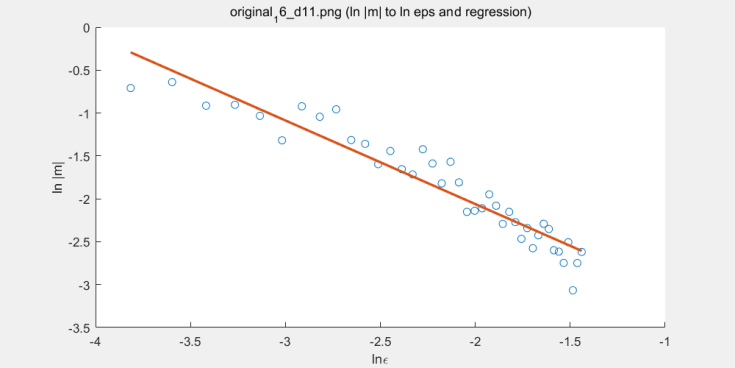

wolff_16_c1.png (-ln Gamma(r) to r with different T and according regression)
-ln Γ(r)
r

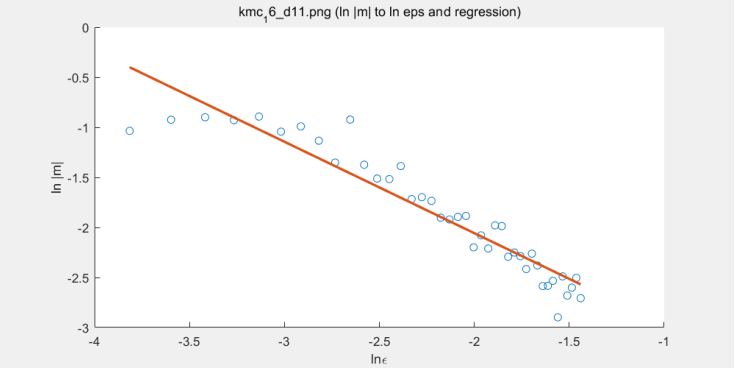

original_16_c2.png (ξ to T)
ξ
T

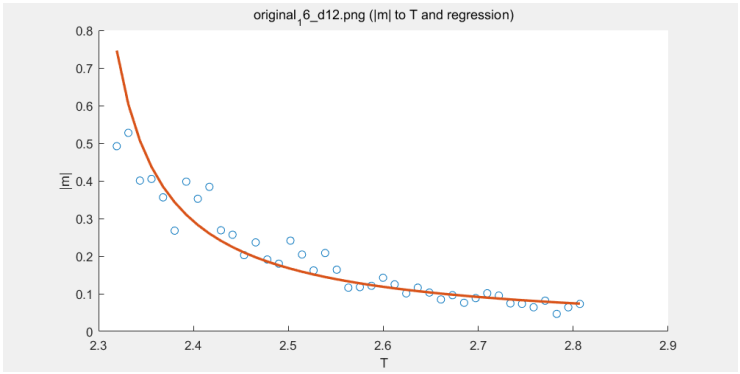

kmc_16_c2.png (ξ to T)
ξ
T

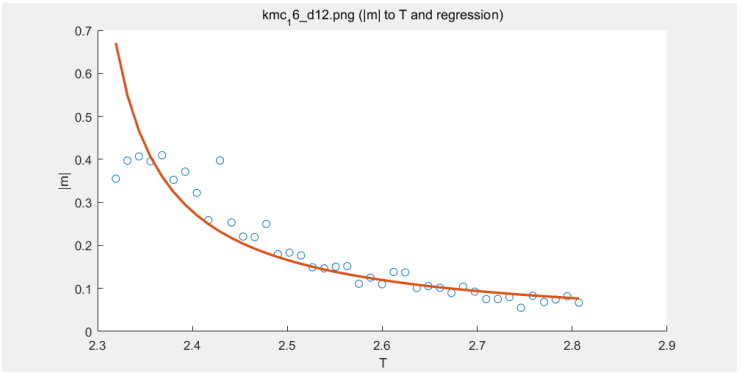

sw_16_c2.png (ξ to T)
ξ
T

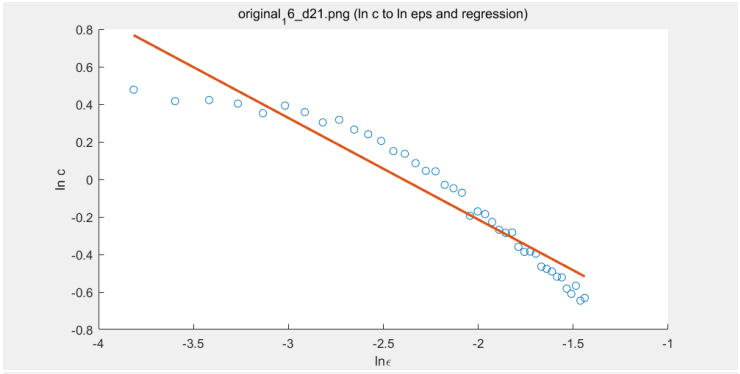

wolff_16_c2.png (ξ to T)
ξ
T

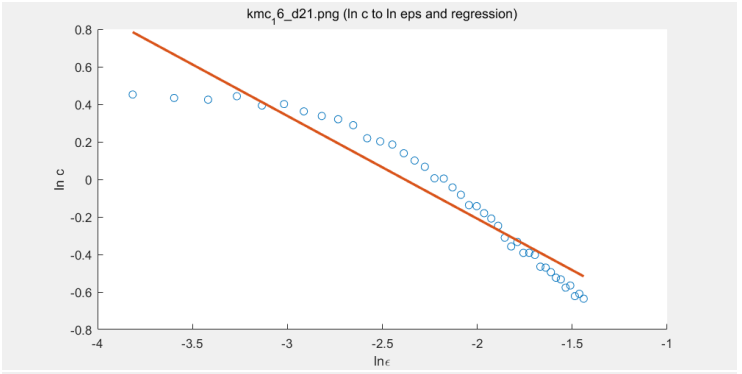

original_16_d11.png (ln |m| to ln eps and regression)
ln |m|
lnϵ

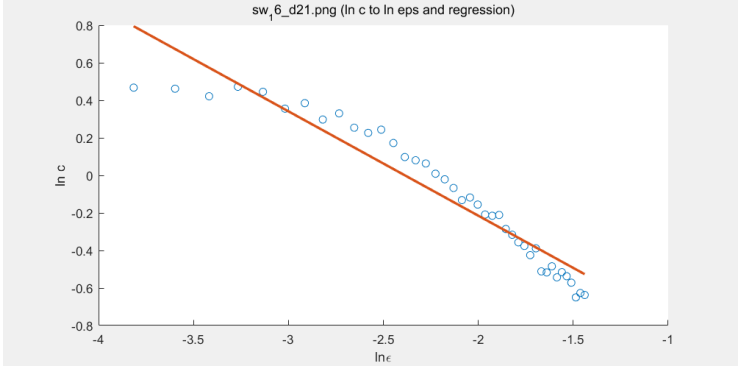

kmc_16_d11.png (ln |m| to ln eps and regression)
ln |m|
lnϵ

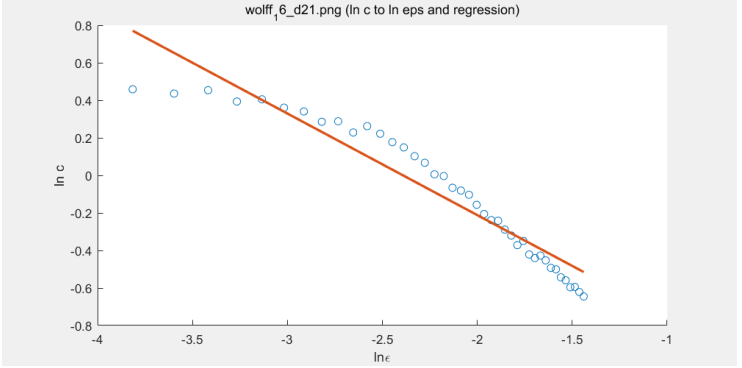

original_16_d12.png (|m| to T and regression)
|m|
T

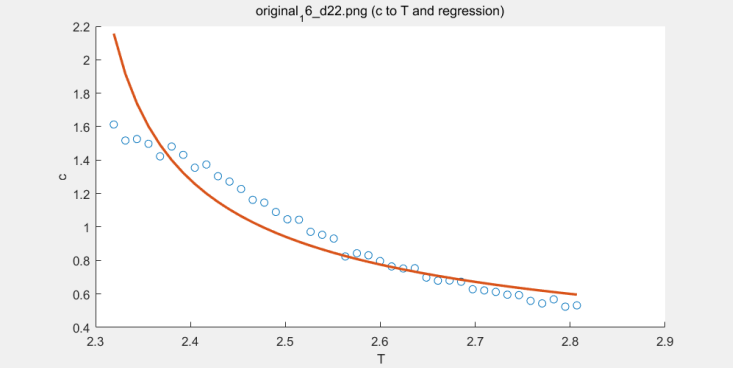

kmc_16_d12.png (|m| to T and regression)
|m|
T

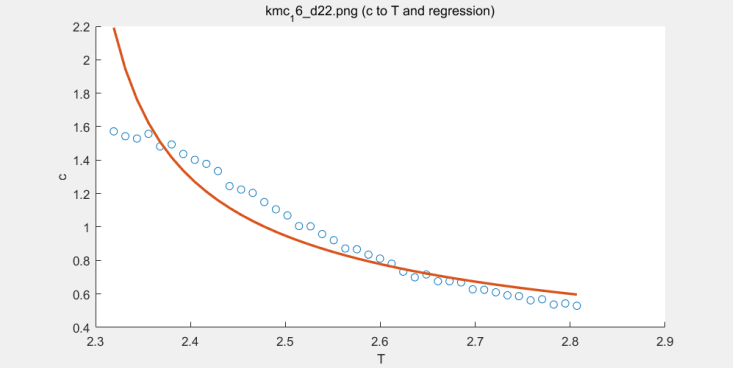

original_16_d21.png (ln c to ln eps and regression)
ln c
lnϵ

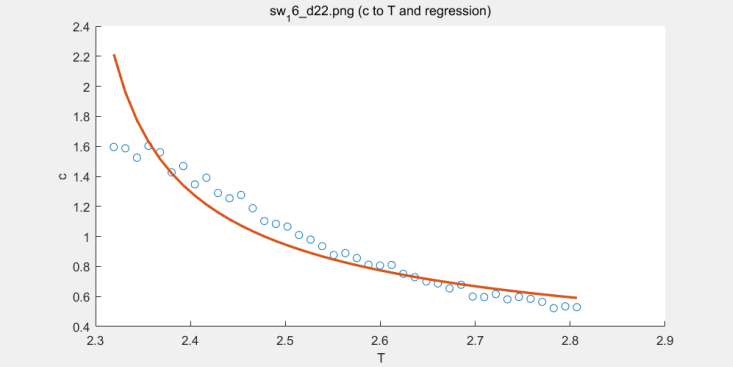

kmc_16_d21.png (ln c to ln eps and regression)
ln c
lnϵ

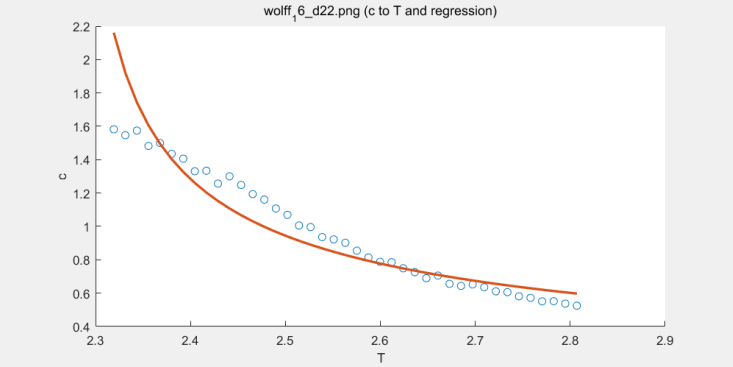

sw_16_d21.png (ln c to ln eps and regression)
ln c
lnϵ

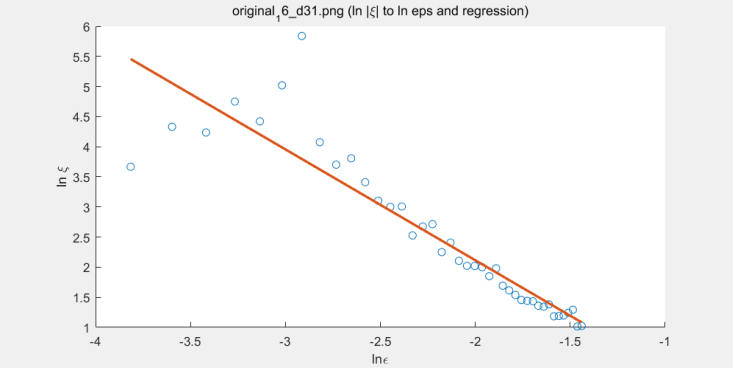

wolff_16_d21.png (ln c to ln eps and regression)
ln c
lnϵ

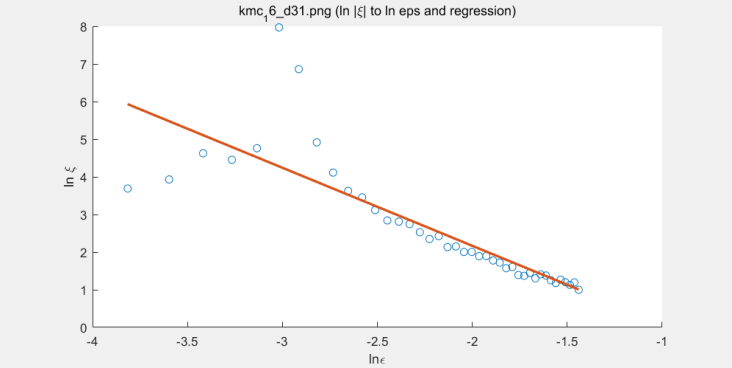

original_16_d22.png (c to T and regression)
c
T

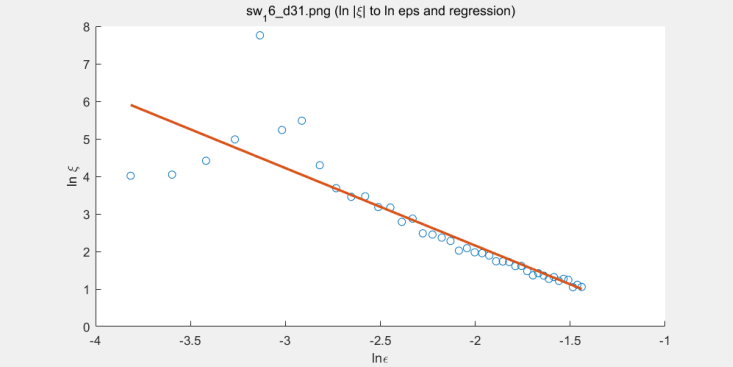

kmc_16_d22.png (c to T and regression)
c
T

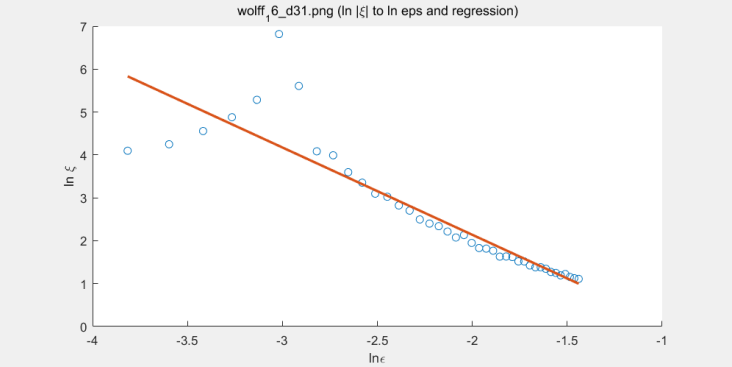

sw_16_d22.png (c to T and regression)
c
T

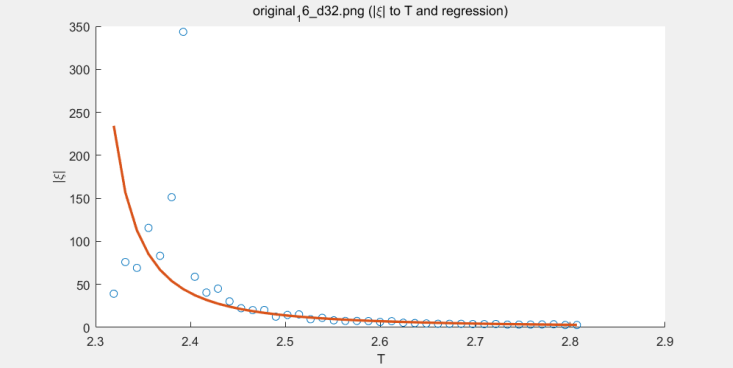

wolff_16_d22.png (c to T and regression)
c
T

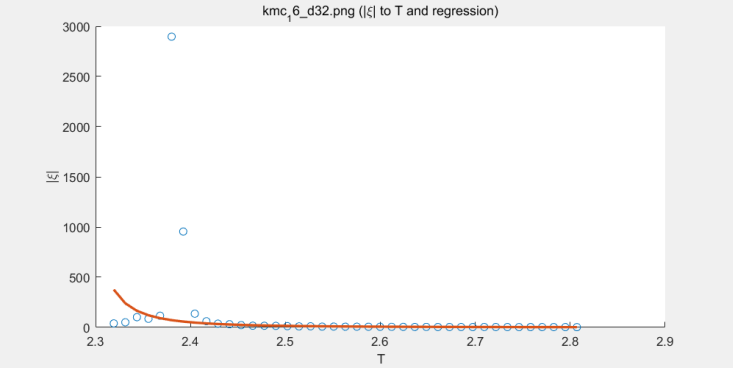

original_16_d31.png (ln |ξ| to ln eps and regression)
ln ξ
lnϵ

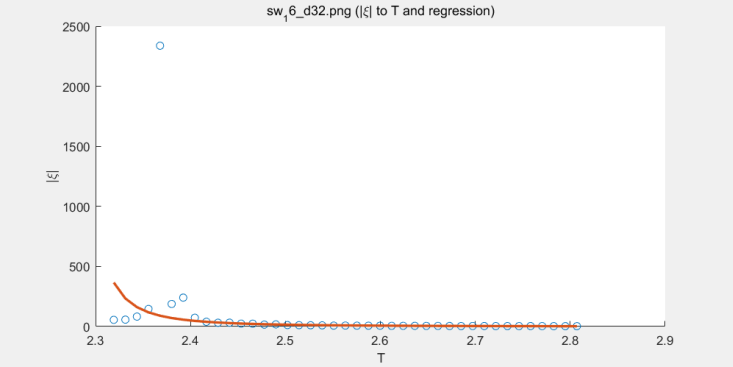

kmc_16_d31.png (ln |ξ| to ln eps and regression)
ln ξ
lnϵ

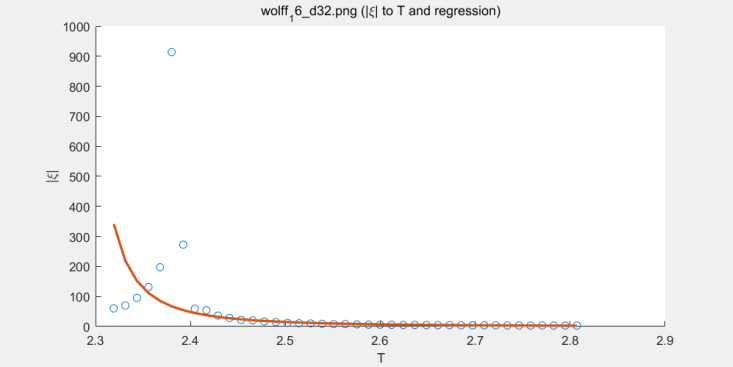

sw_16_d31.png (ln |ξ| to ln eps and regression)
ln ξ
lnϵ


wolff_16_d31.png (ln |ξ| to ln eps and regression)
ln ξ
lnϵ

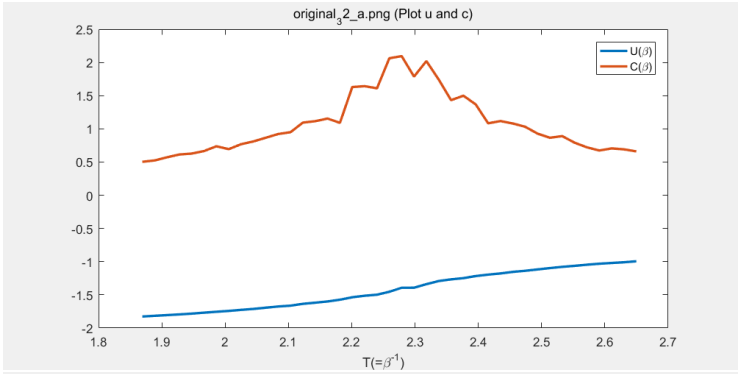

original_16_d32.png (|ξ| to T and regression)
|ξ|
T
0
50
100
150
200
250
300
350
2.3
2.4
2.5
2.6
2.7
2.8
2.9

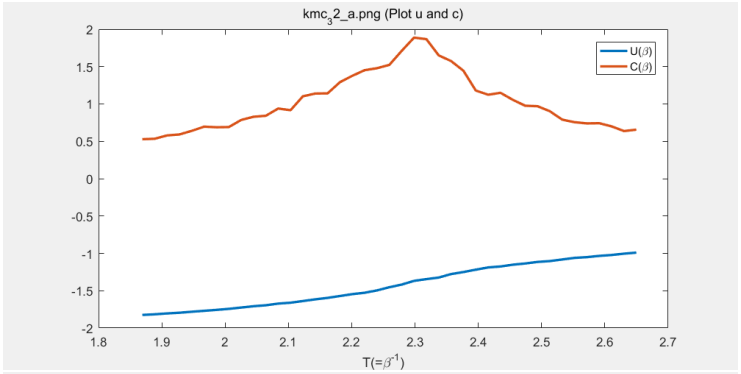

kmc_16_d32.png (|ξ| to T and regression)
|ξ|
T
0
500
1000
1500
2000
2500
3000
2.3
2.4
2.5
2.6
2.7
2.8
2.9

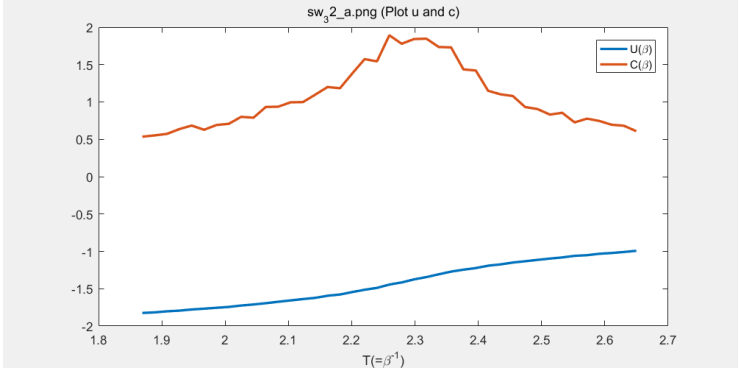

sw_16_d32.png (|ξ| to T and regression)
|ξ|
T
0
500
1000
1500
2000
2500
2.3
2.4
2.5
2.6
2.7
2.8
2.9

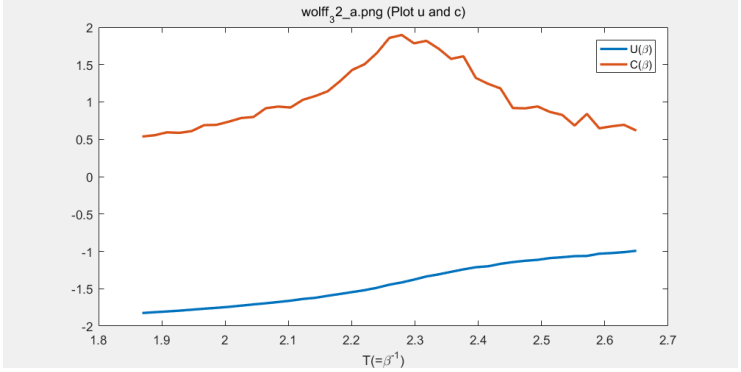

wolff_16_d32.png (|ξ| to T and regression)
|ξ|
T
0
100
200
300
400
500
600
700
800
900
1000
2.3
2.4
2.5
2.6
2.7
2.8
2.9

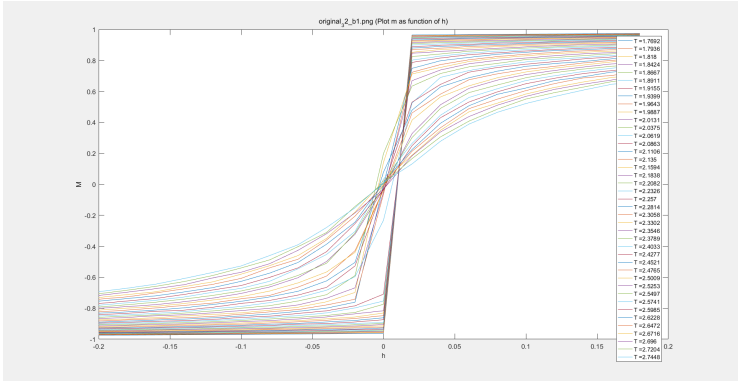
3.2 $N = 32$

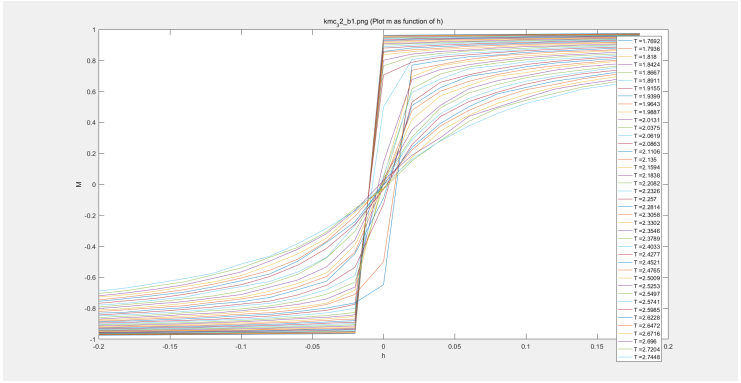

original$_3$2_a.png (Plot u and c)
U(β)
C(β)
T(=β^{-1})

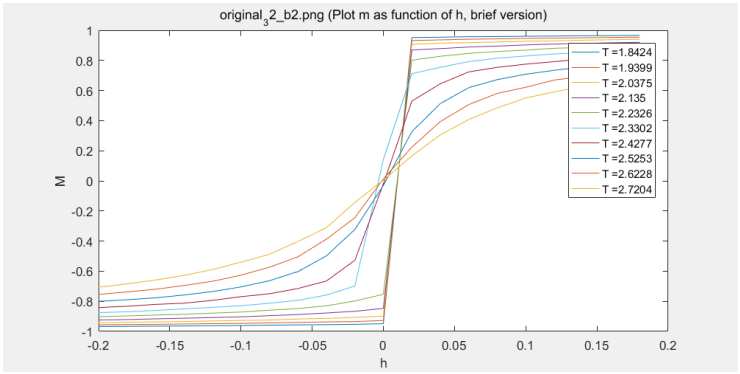

kmc$_3$2_a.png (Plot u and c)
U(β)
C(β)
T(=β^{-1})

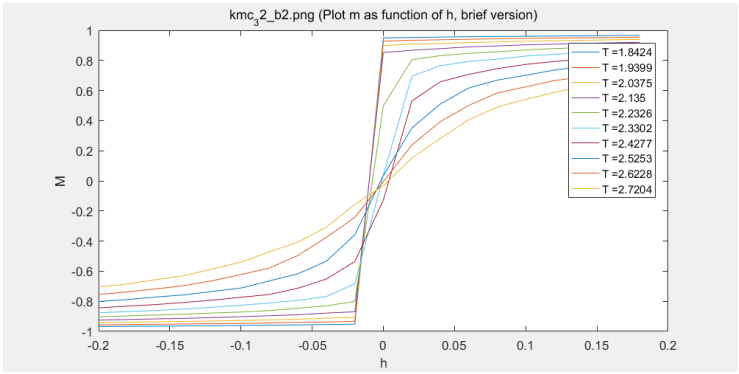

sw$_3$2_a.png (Plot u and c)
U(β)
C(β)
T(=β^{-1})

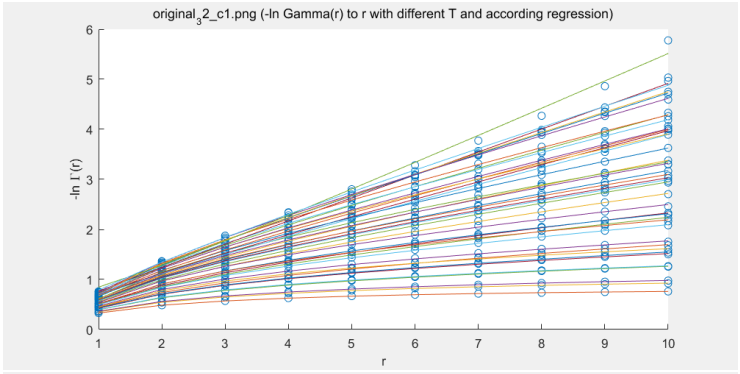

wolff$_3$2_a.png (Plot u and c)
U(β)
C(β)
T(=β^{-1})

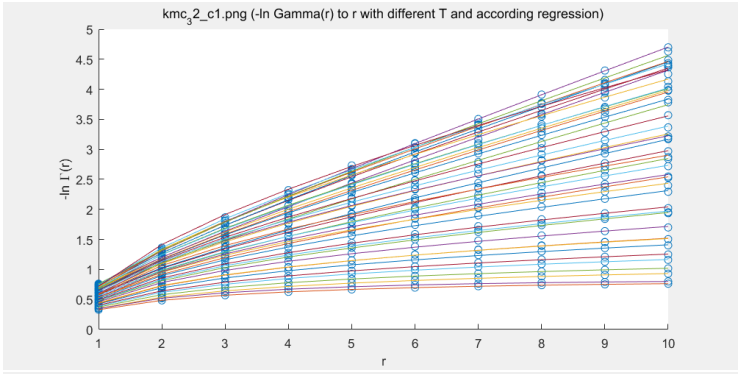

original$_3$2_b1.png (Plot m as function of h)
M
h

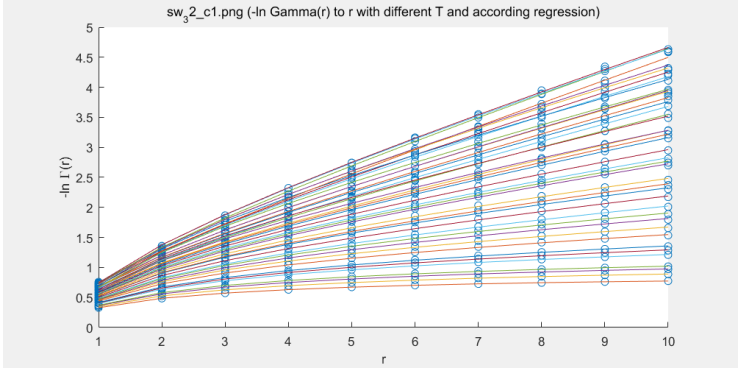

kmc$_3$2_b1.png (Plot m as function of h)
M
h

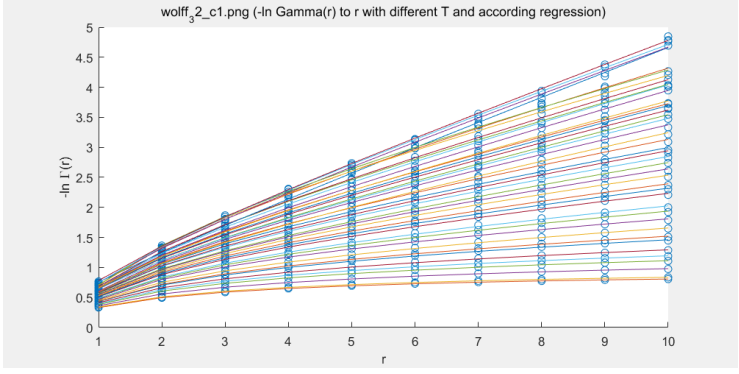

original$_3$2_b2.png (Plot m as function of h, brief version)
M
h
T =1.8424
T =1.9399
T =2.0375
T =2.135
T =2.2326
T =2.3302
T =2.4277
T =2.5253
T =2.6228
T =2.7204

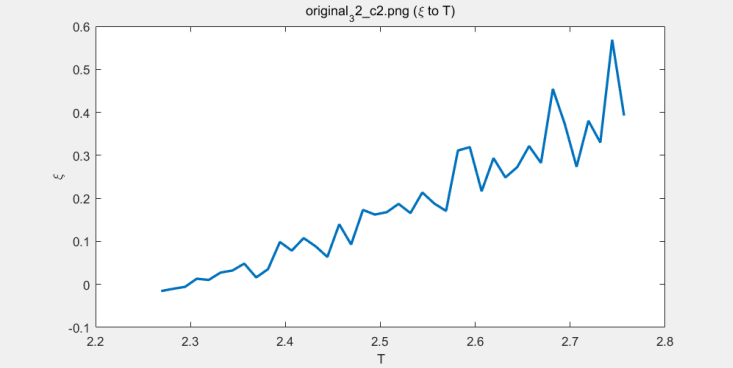

kmc$_3$2_b2.png (Plot m as function of h, brief version)
M
h
T =1.8424
T =1.9399
T =2.0375
T =2.135
T =2.2326
T =2.3302
T =2.4277
T =2.5253
T =2.6228
T =2.7204

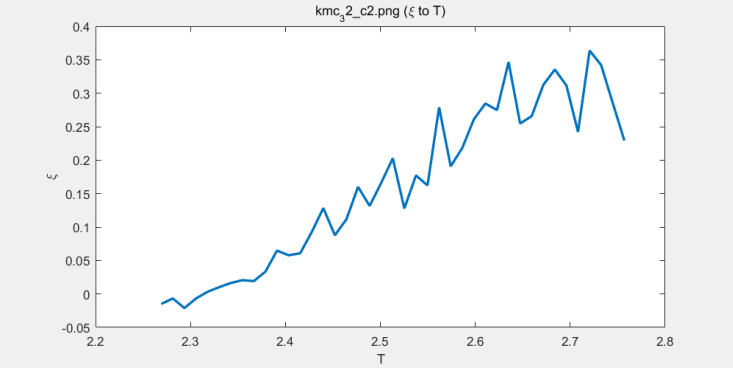

original$_3$2_c1.png (-ln Gamma(r) to r with different T and according regression)
-ln Γ(r)
r

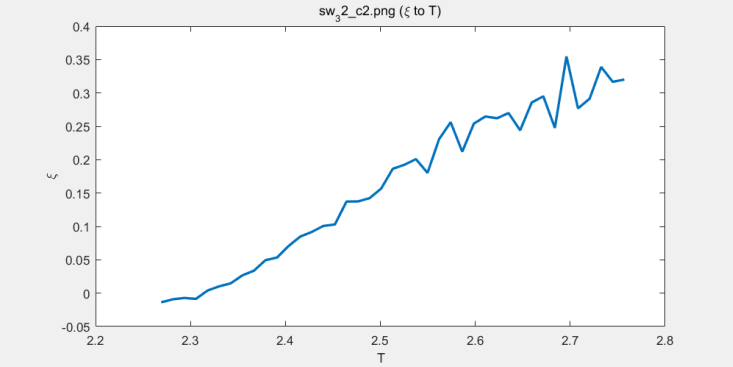

kmc$_3$2_c1.png (-ln Gamma(r) to r with different T and according regression)
-ln Γ(r)
r

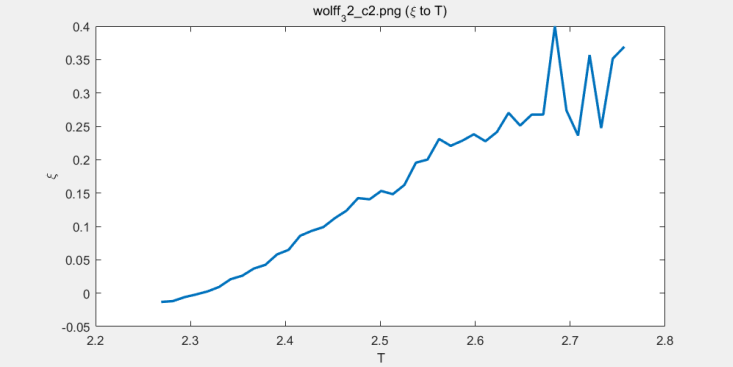

sw$_3$2_c1.png (-ln Gamma(r) to r with different T and according regression)
-ln Γ(r)
r

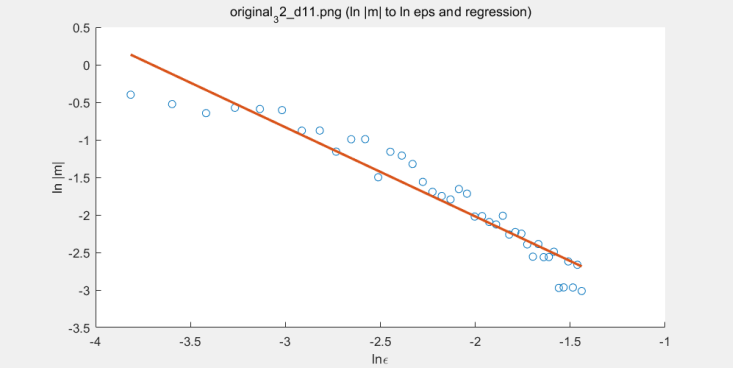

wolff$_3$2_c1.png (-ln Gamma(r) to r with different T and according regression)
-ln Γ(r)
r

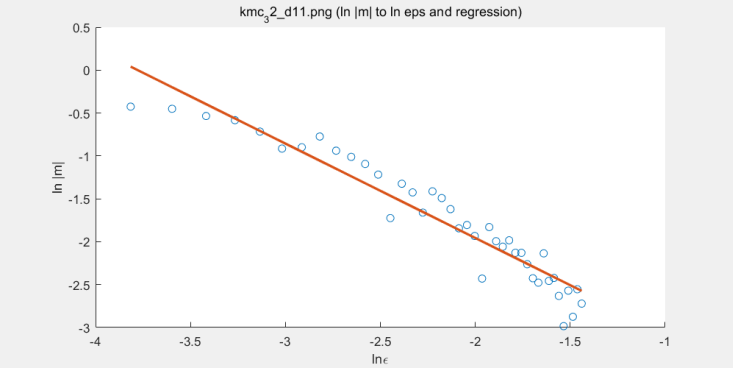

original$_3$2_c2.png (ξ to T)
ξ
T

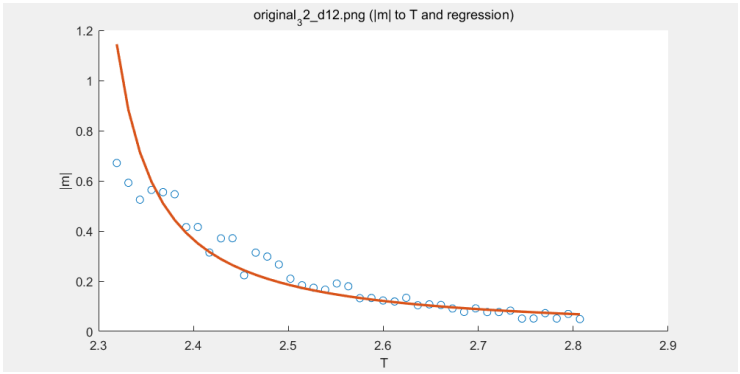

kmc$_3$2_c2.png (ξ to T)
ξ
T

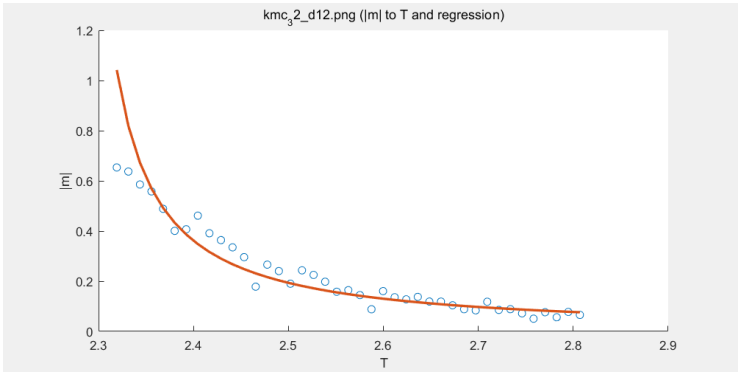

sw$_3$2_c2.png (ξ to T)
ξ
T

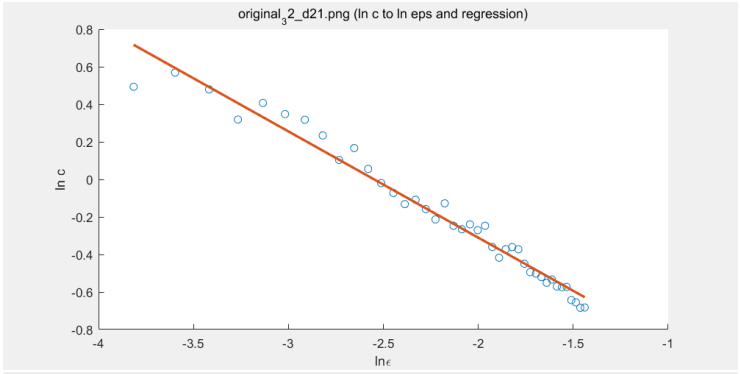

wolff$_3$2_c2.png (ξ to T)
ξ
T

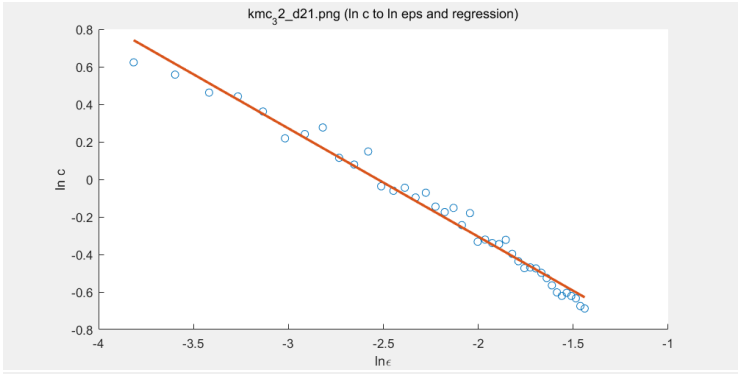

original$_3$2_d11.png (ln |m| to ln eps and regression)
ln |m|
lnϵ

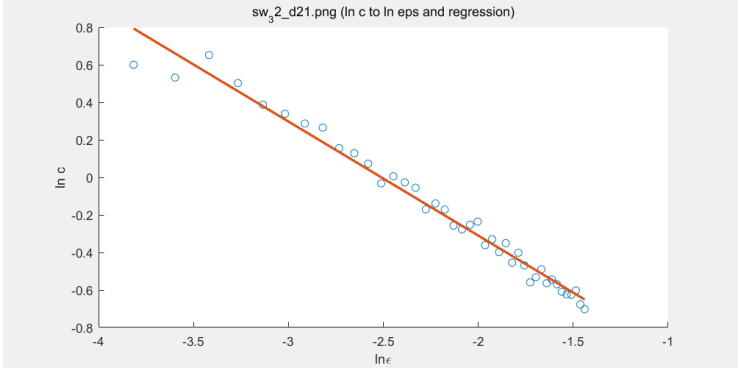

kmc$_3$2_d11.png (ln |m| to ln eps and regression)
ln |m|
lnϵ

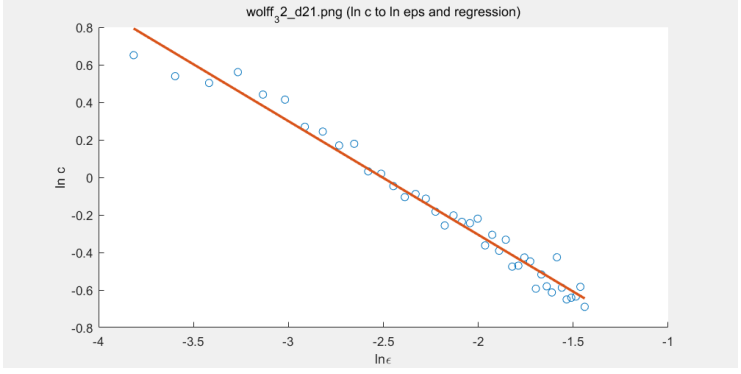

original_3_2_d12.png (|m| to T and regression)
|m|
T

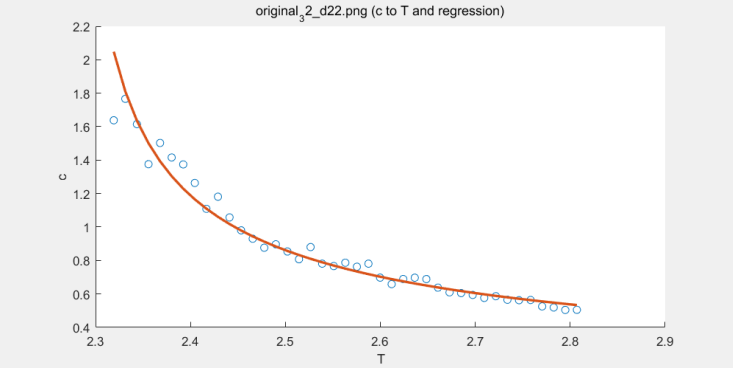

kmc_3_2_d12.png (|m| to T and regression)
|m|
T

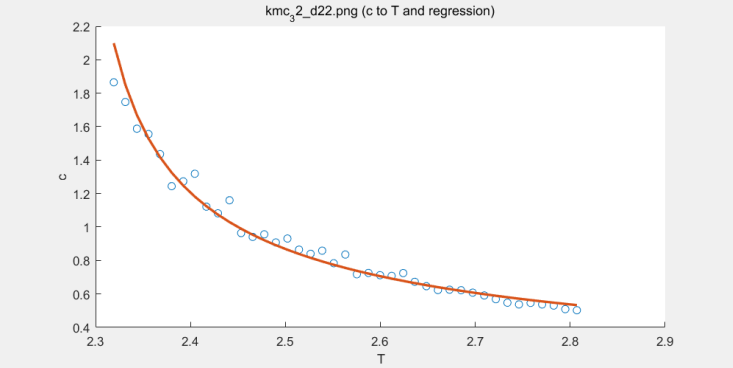

original_3_2_d21.png (ln c to ln eps and regression)
ln c
lnϵ

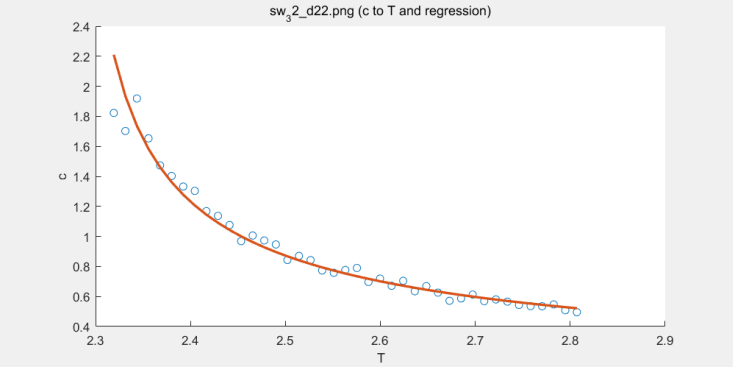

kmc_3_2_d21.png (ln c to ln eps and regression)
ln c
lnϵ

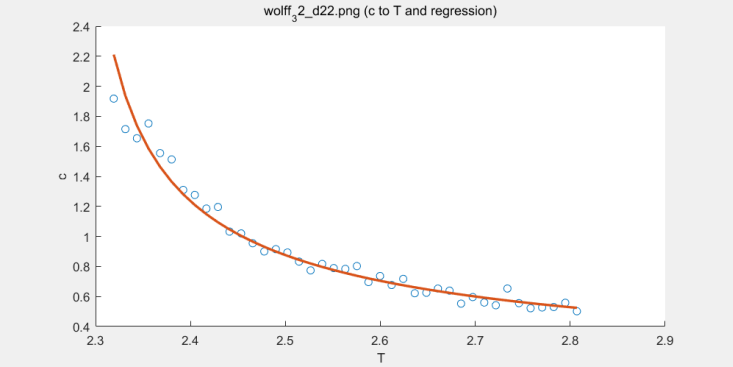

sw_3_2_d21.png (ln c to ln eps and regression)
ln c
lnϵ

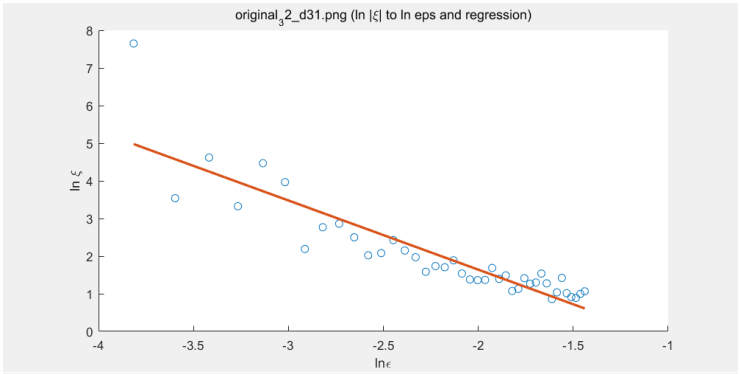

wolff_3_2_d21.png (ln c to ln eps and regression)
ln c
lnϵ

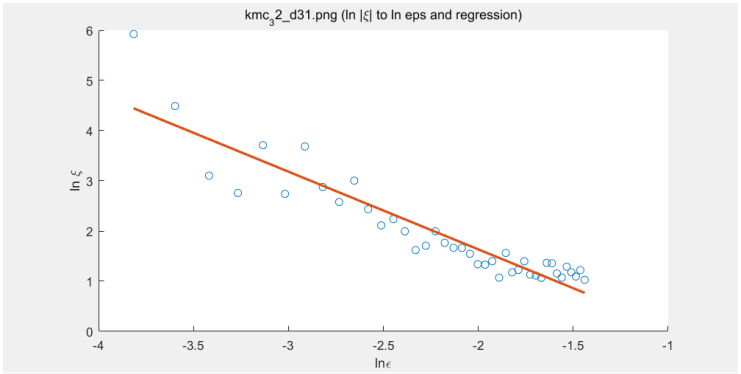

original_3_2_d22.png (c to T and regression)
c
T
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
2.3
2.4
2.5
2.6
2.7
2.8
2.9

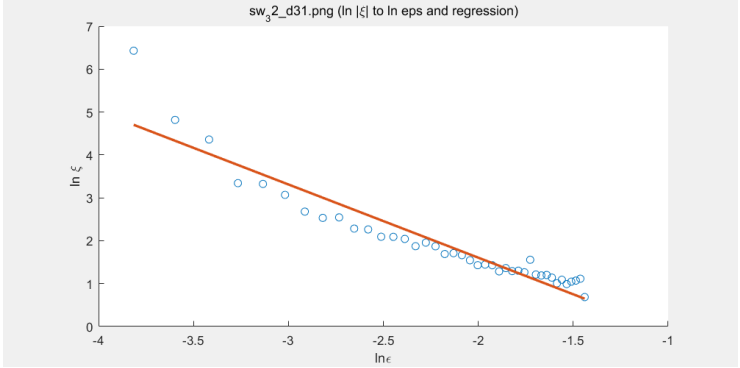

kmc_3_2_d22.png (c to T and regression)
c
T
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
2.3
2.4
2.5
2.6
2.7
2.8
2.9

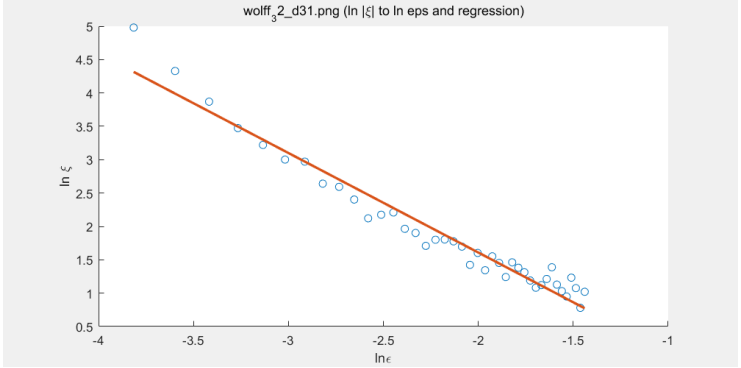

sw_3_2_d22.png (c to T and regression)
c
T
2.4
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
2.3
2.4
2.5
2.6
2.7
2.8
2.9

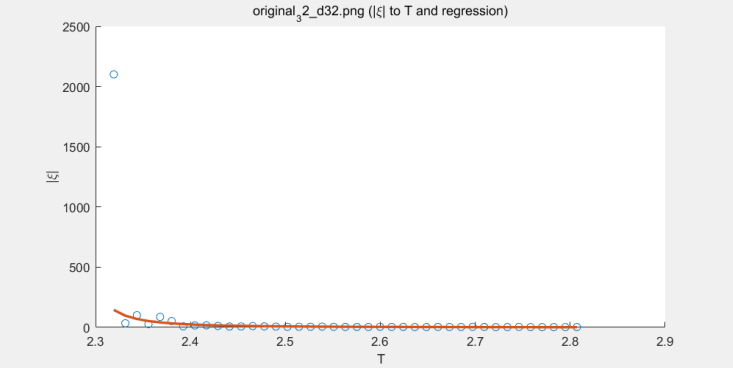

wolff_3_2_d22.png (c to T and regression)
c
T
2.4
2.2
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
2.3
2.4
2.5
2.6
2.7
2.8
2.9

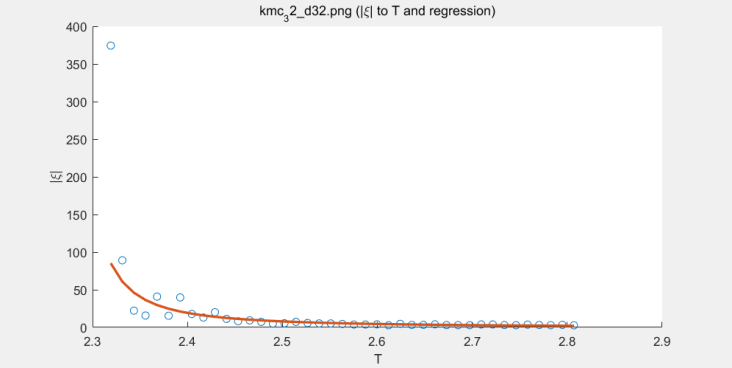

original$_3$2_d31.png (ln |ξ| to ln eps and regression)
ln ξ
lnϵ

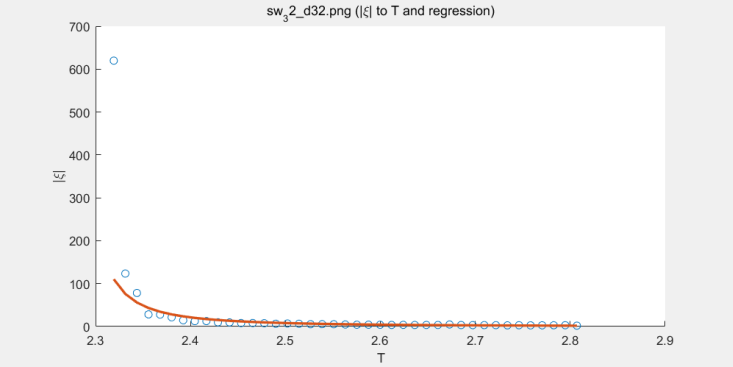

kmc$_3$2_d31.png (ln |ξ| to ln eps and regression)
ln ξ
lnϵ

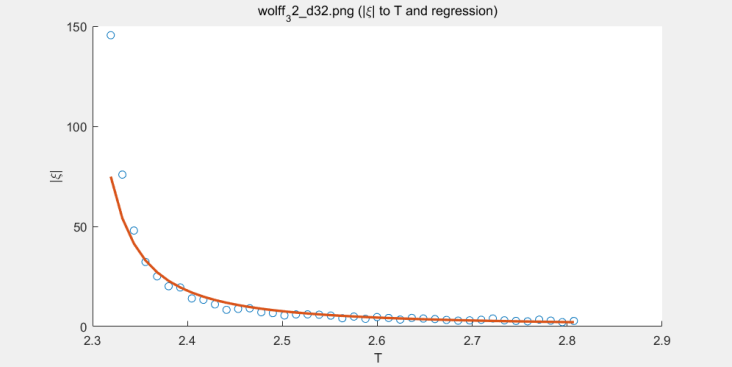

sw$_3$2_d31.png (ln |ξ| to ln eps and regression)
ln ξ
lnϵ


wolff$_3$2_d31.png (ln |ξ| to ln eps and regression)
ln ξ
lnϵ


original$_3$2_d32.png (|ξ| to T and regression)
|ξ|
T
2500
2000
1500
1000
500
0
2.3
2.4
2.5
2.6
2.7
2.8
2.9


kmc$_3$2_d32.png (|ξ| to T and regression)
|ξ|
T
400
350
300
250
200
150
100
50
0
2.3
2.4
2.5
2.6
2.7
2.8
2.9


sw$_3$2_d32.png (|ξ| to T and regression)
|ξ|
T
700
600
500
400
300
200
100
0
2.3
2.4
2.5
2.6
2.7
2.8
2.9


wolff$_3$2_d32.png (|ξ| to T and regression)
|ξ|
T
150
100
50
0
2.3
2.4
2.5
2.6
2.7
2.8
2.9