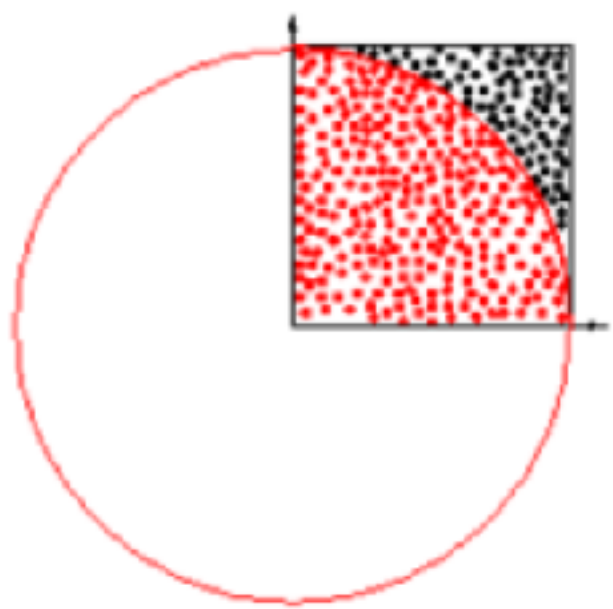
993SM - Laboratory of Computational Physics week 7 November 8, 2024

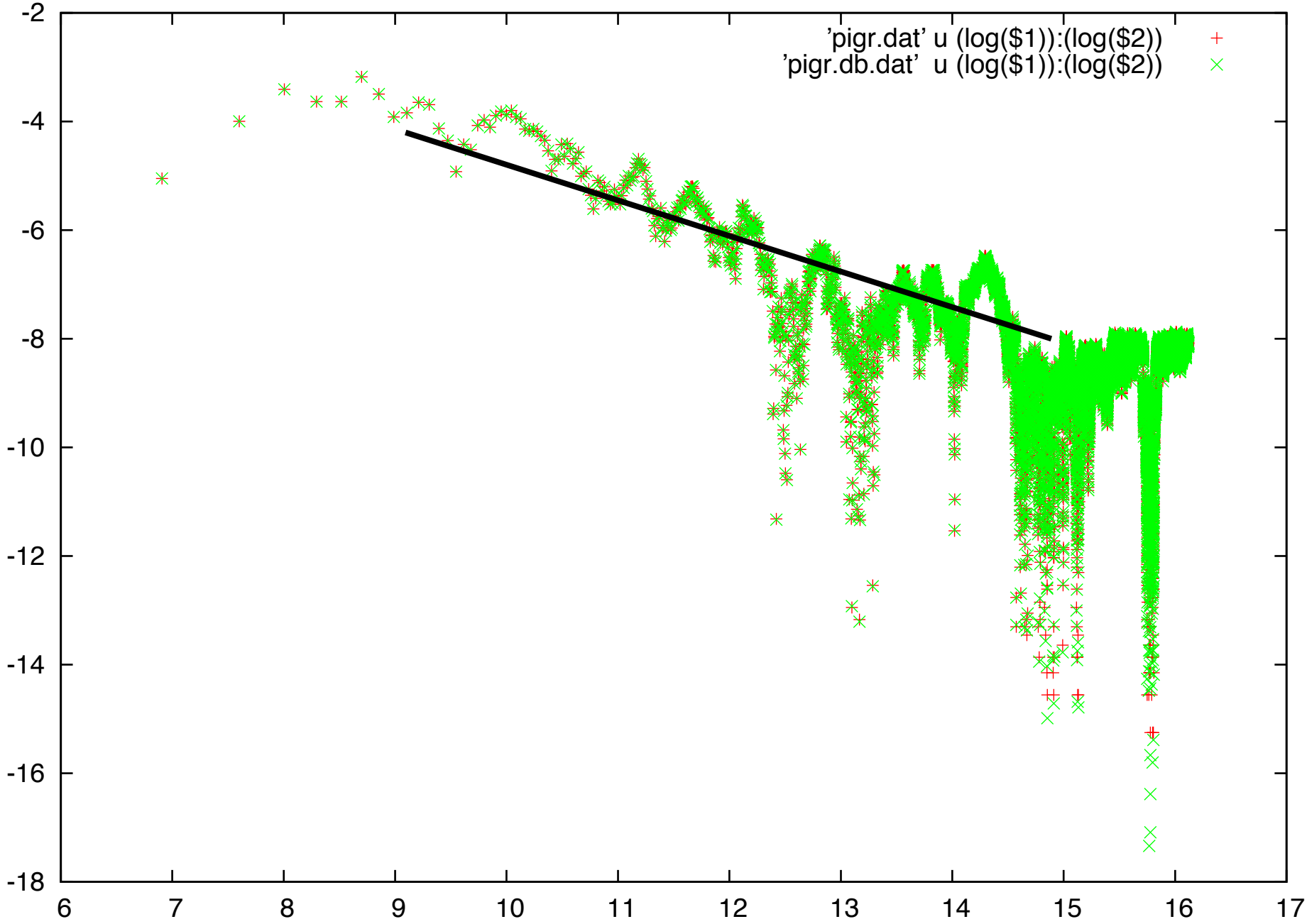
Maria Peressi
Università degli Studi di Trieste - Dipartimento di Fisica
Sede di Miramare (Strada Costiera 11, Trieste)
e-mail: peressi@units.it
tel.: +39 040 2240242

1. Monte Carlo method: acceptance-rejection

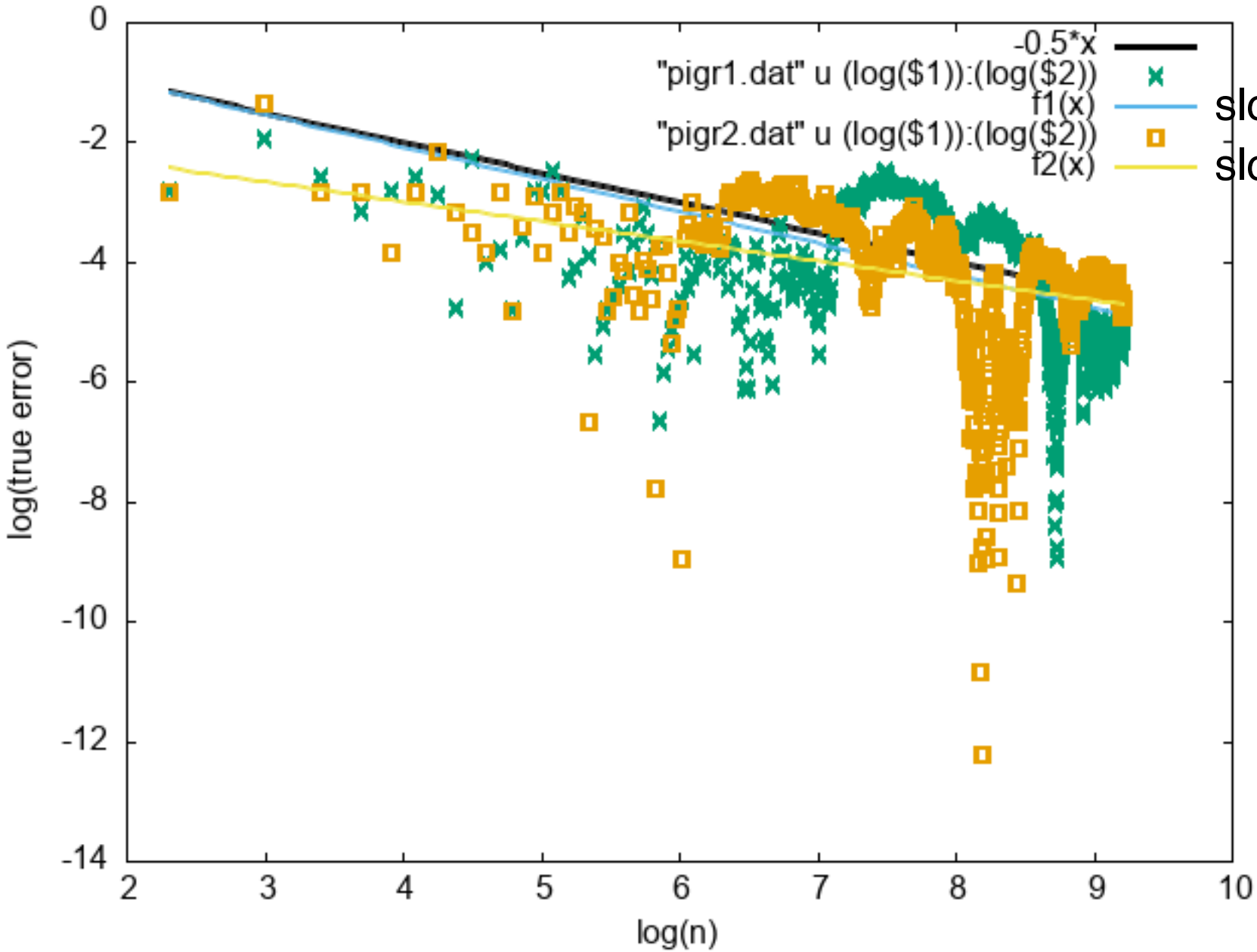
Using the acceptance-rejection method, calculate $I = \int_0^1 \sqrt{1-x^2}dx$ (notice that $\pi = 4I$). The numerical estimate of the integral is $F_n = \frac{n_s}{n}$ where n_s is the number of points under the curve $f(x) = \sqrt{1-x^2}$, and n the total number of points generated. An example is given in `pi.f90`. Estimate the error associated, i.e. the difference between F_n and the true value. Discuss the dependence of the error on n.
(*Notice that many points are needed to see the* $n^{-1/2}$ *behavior, which can be hidden by stochastic fluctuations; it is easier to see it by averaging over many results (obtained from random numbers sequences with different seeds)*)



error(MC)~1/√N => see log(error) vs. log(N)



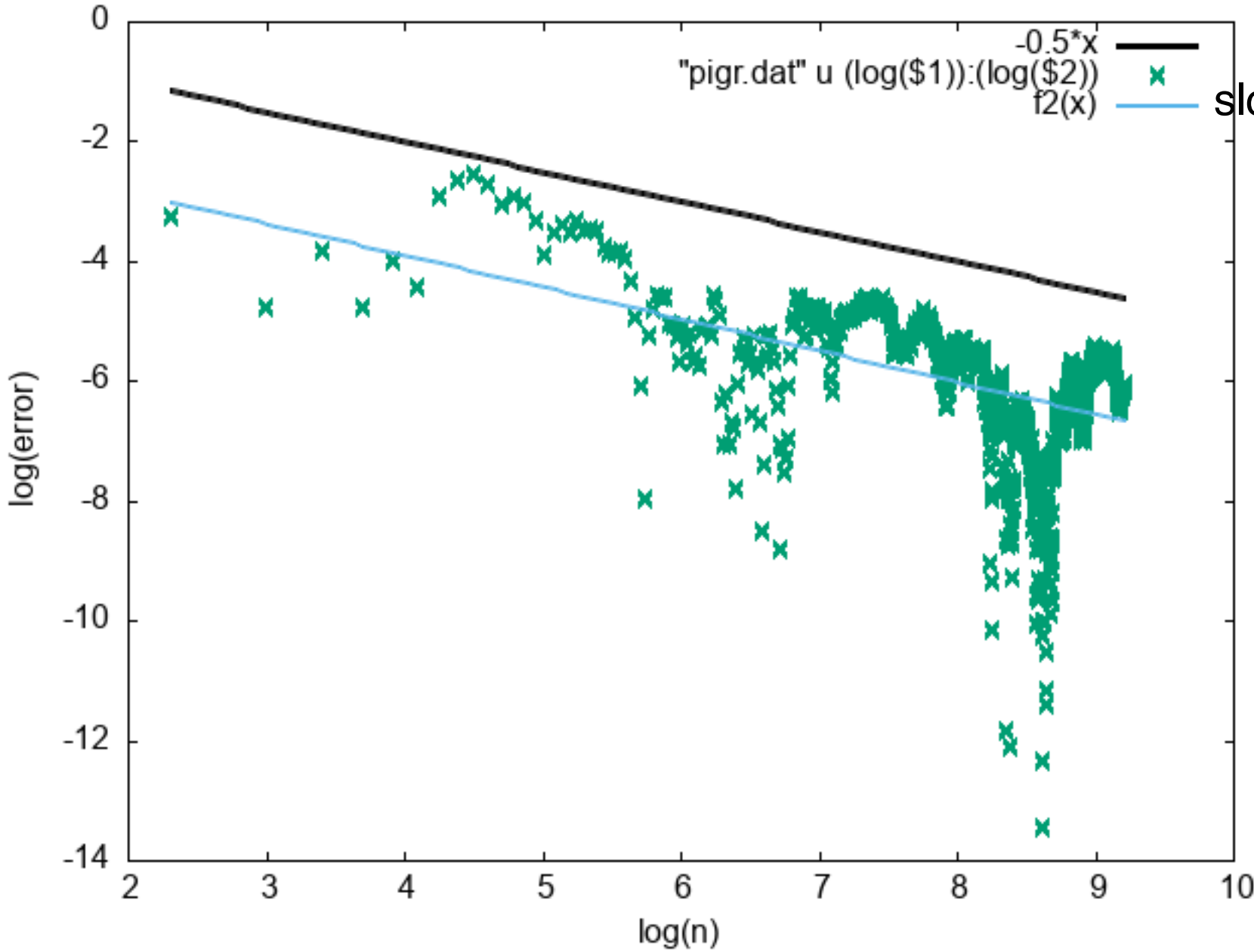
different seeds



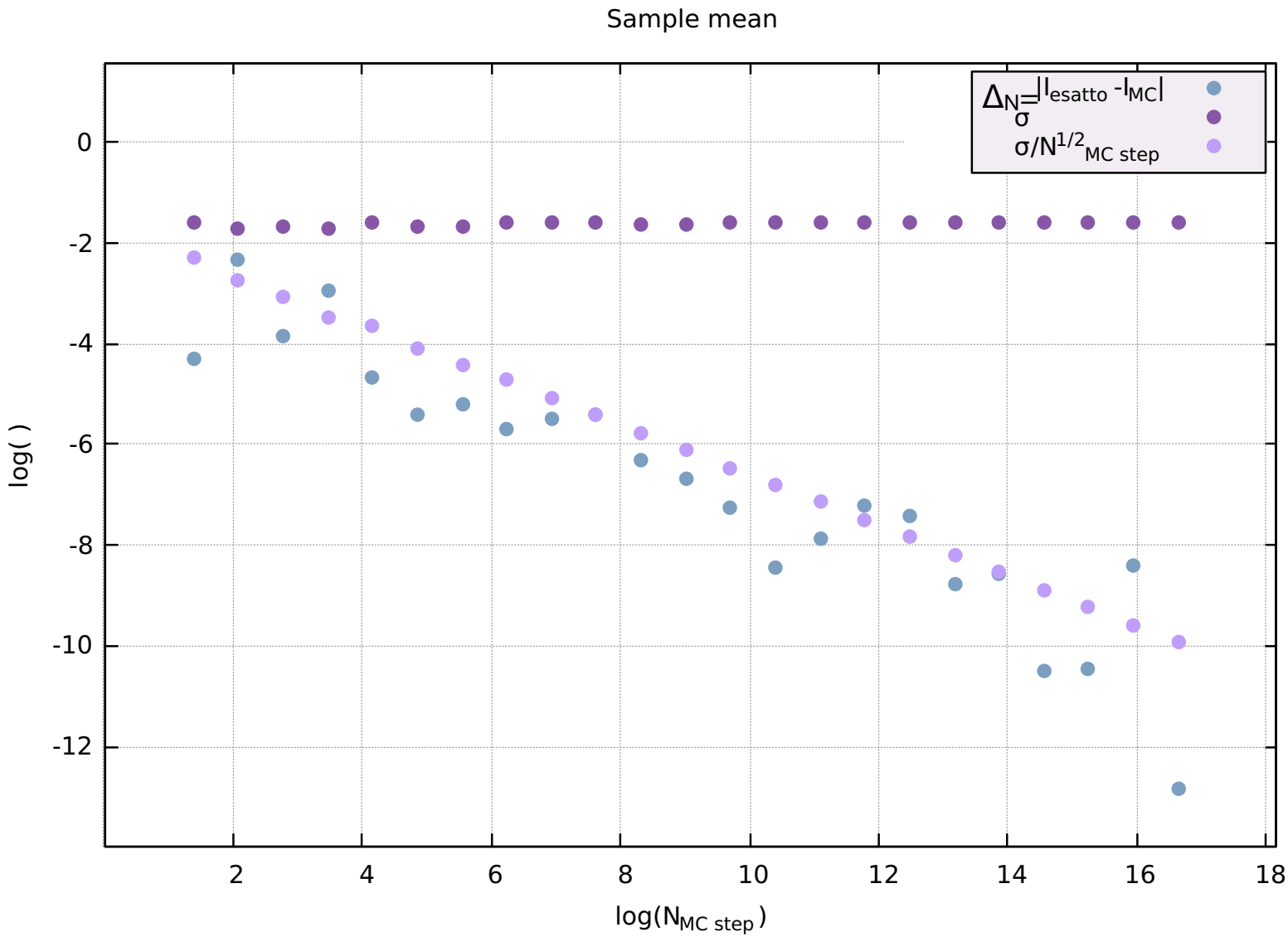
slope -0.54

slope -0.33

average over 20 different seeds



error(MC)~1/√N :"true" error and statistical error



(credits: G. Lautizi, a.y. 2019-20)

Extension to the hypersphere

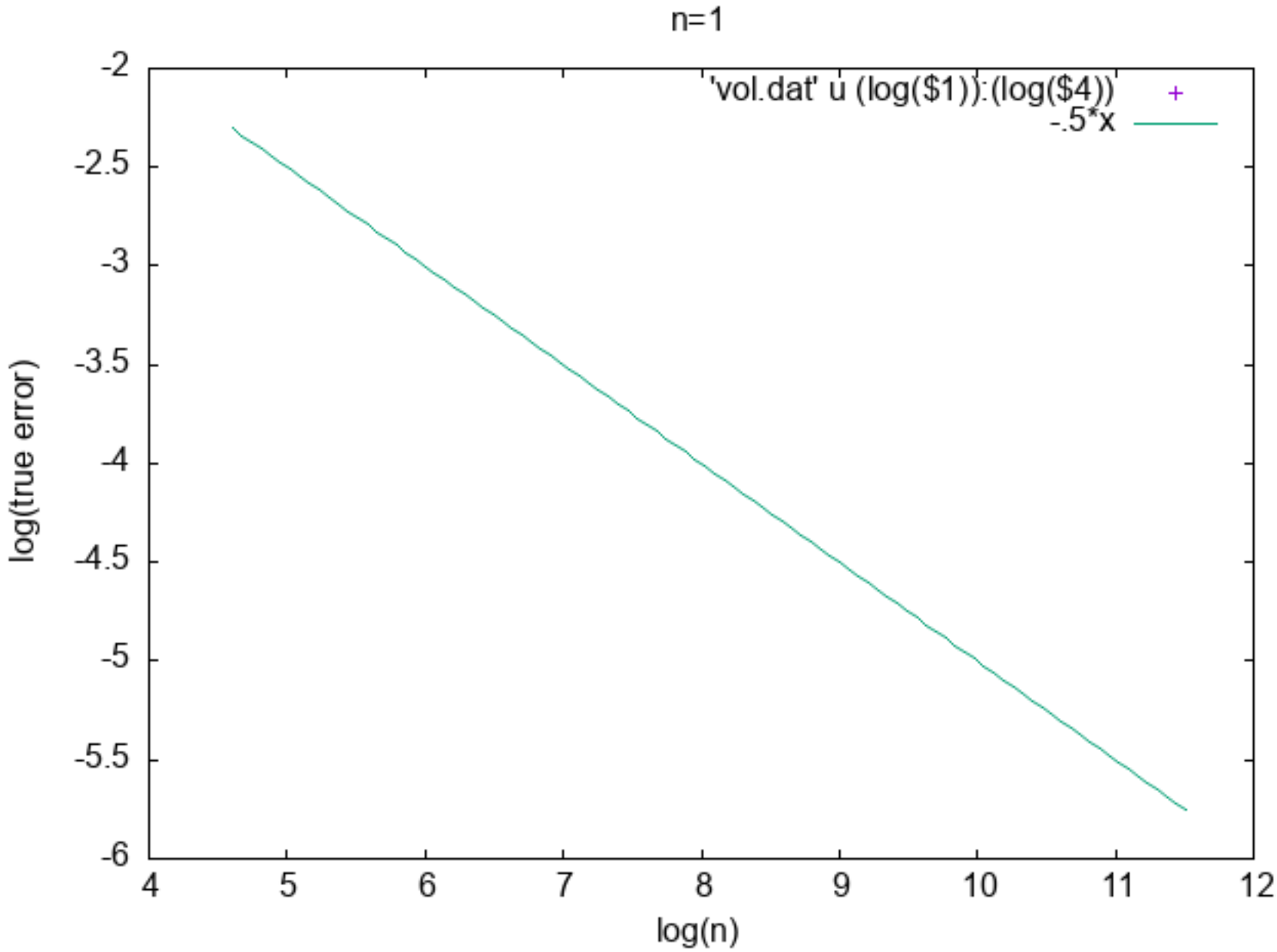
VOLUME OF HYPERSPHERE

$$V_n = \frac{\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2}+1)}$$

$$\Gamma(1/2) = \sqrt{\pi}; \Gamma(1) = 1; \Gamma(x+1) = x\Gamma(x)$$

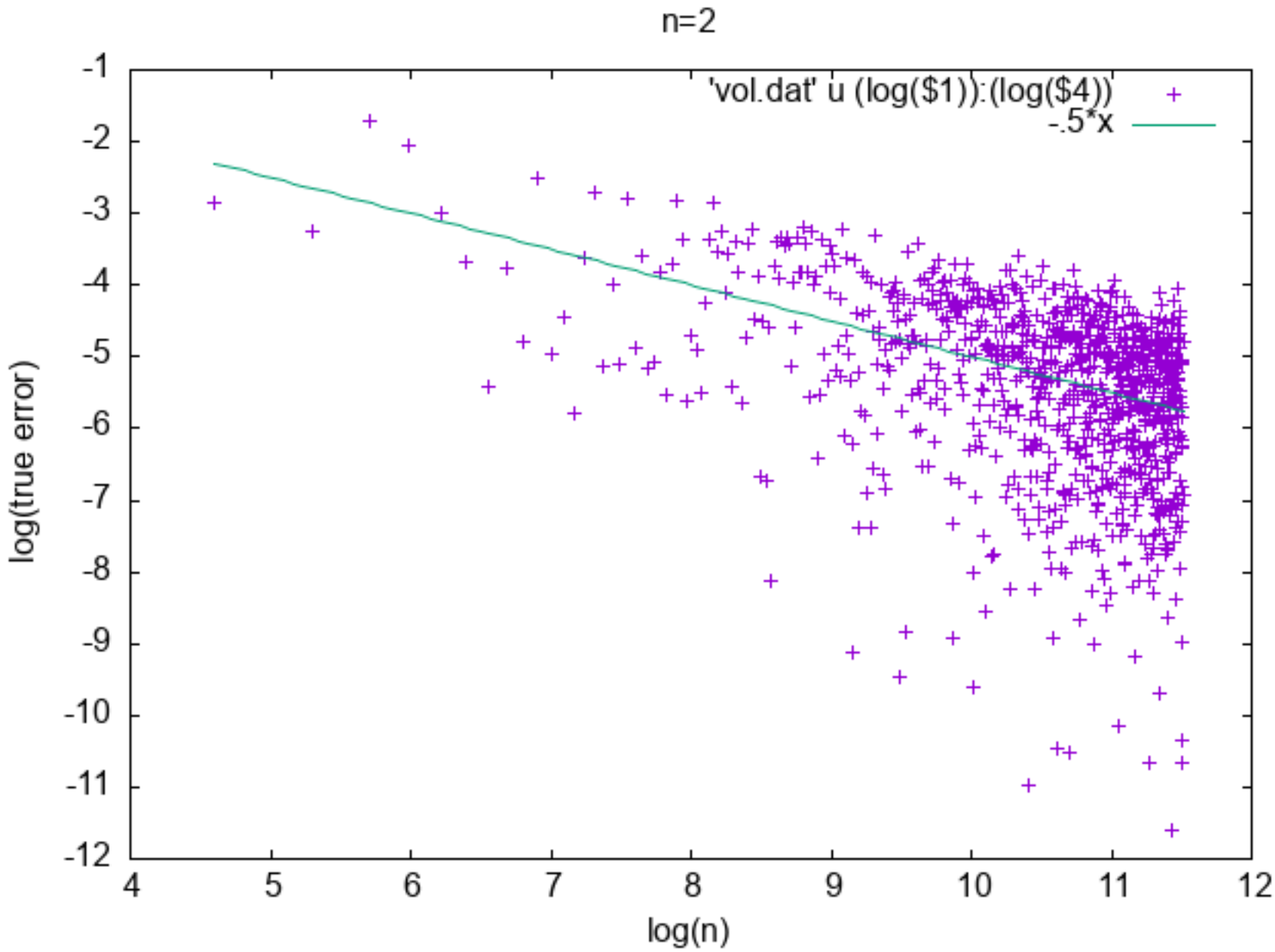
$$V_{n+2} = 2\pi V_n/(n+2)$$

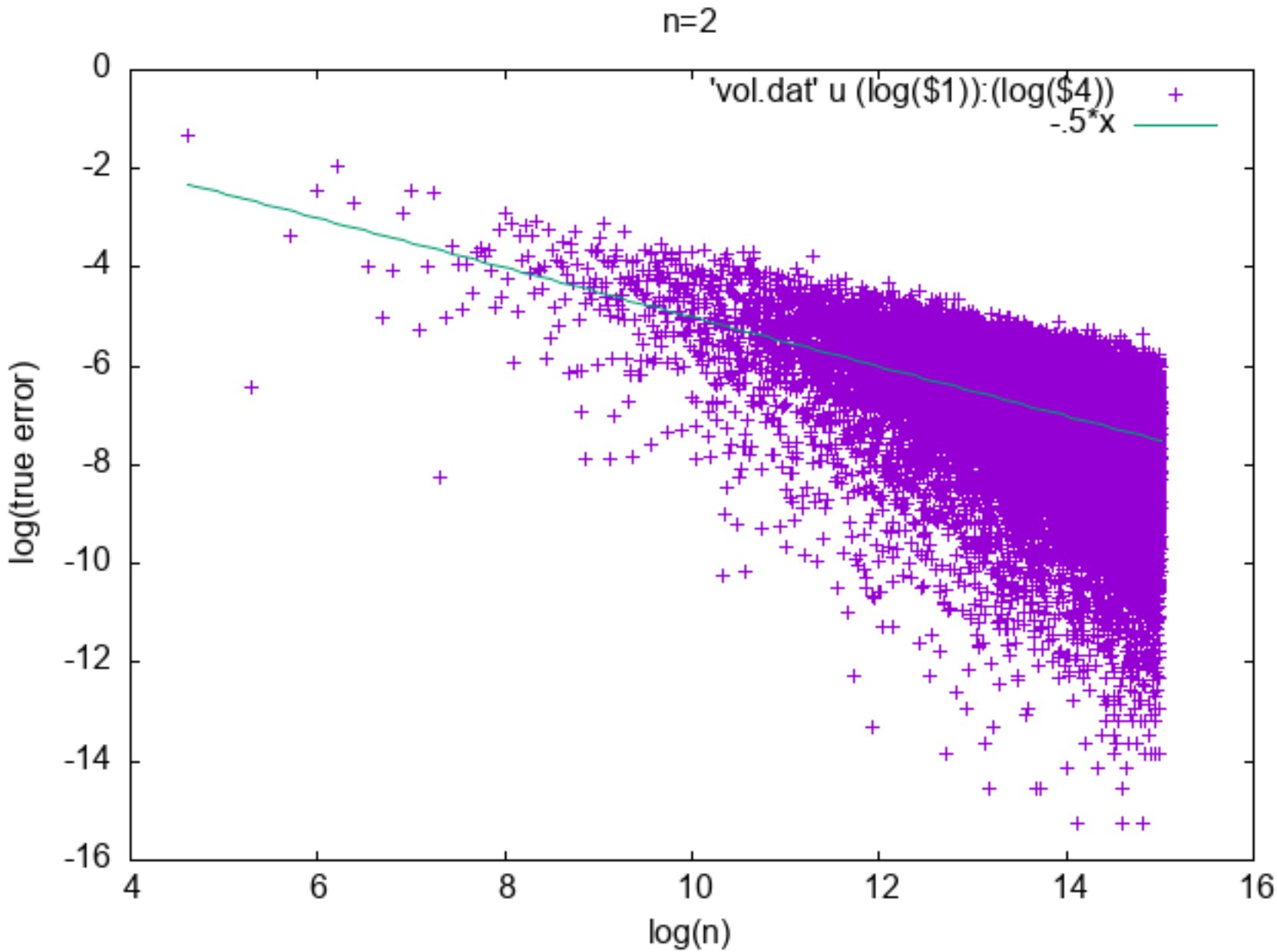
Numero di dimensioni *n*	Ipervolume $V_n(r)$	Misura ipersuperficiale $S_n(r)$	Valore numerico $V_n(1)$	Valore numerico $S_n(1)$
1	$2r$	2	2,000.000.000	2,000.000.000
2	πr^2	$2\pi r$	3,141.592.654	6,283.185.307
3	$\frac{4}{3}\pi r^3$	$4\pi r^2$	4,188.790.205	12,566.370.614
4	$\frac{1}{2}\pi^2 r^4$	$2\pi^2 r^3$	4,934.802.201	19,739.208.802
5	$\frac{8}{15}\pi^2 r^5$	$\frac{8}{3}\pi^2 r^4$	5,263.789.014	26,318.945.070
19	$\frac{1.024}{654.729.075}\pi^9 r^{19}$	$\frac{1.024}{34.459.425}\pi^9 r^{18}$	0,046.621.601	0,885.810.420
20	$\frac{1}{3.628.800}\pi^{10} r^{20}$	$\frac{1}{181.440}\pi^{10} r^{19}$	0,025.806.891	0,516.137.828


n=1
'vol.dat' u (log($1)):(log($4))
-.5*x
log(true error)
log(n)
-2
-2.5
-3
-3.5
-4
-4.5
-5
-5.5
-6
4
5
6
7
8
9
10
11
12

100000 punti - runs non correlati con diversi n

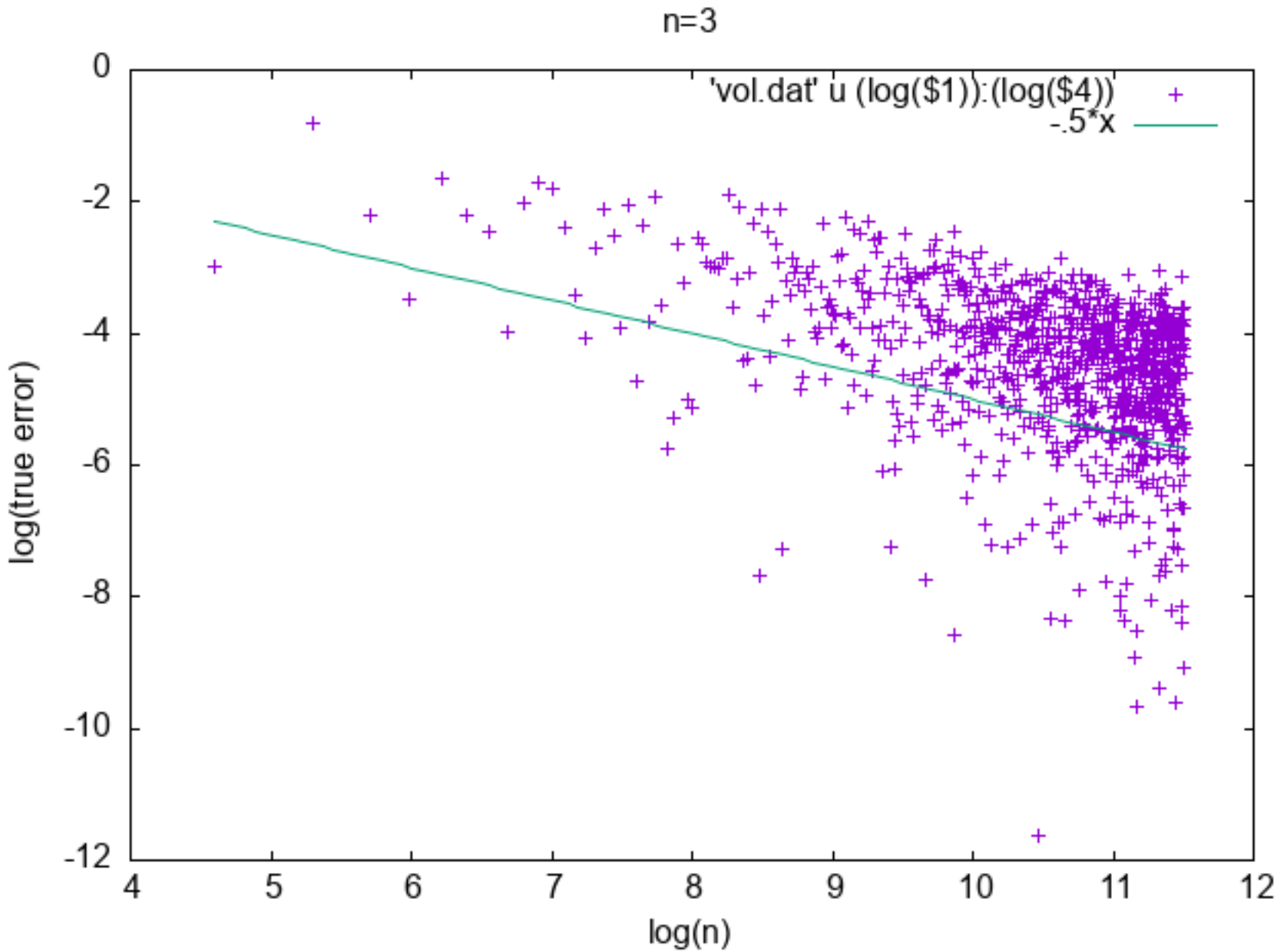
volume sfera numerico e esatto: 3.14256001 3.14159274




n=2
'vol.dat' u (log($1)):(log($4))
-.5*x
log(true error)
log(n)
0
-2
-4
-6
-8
-10
-12
-14
-16
4
6
8
10
12
14
16

100000 punti - runs non correlati con diversi n

volume sfera numerico e esatto: 4.19895983 4.18879080

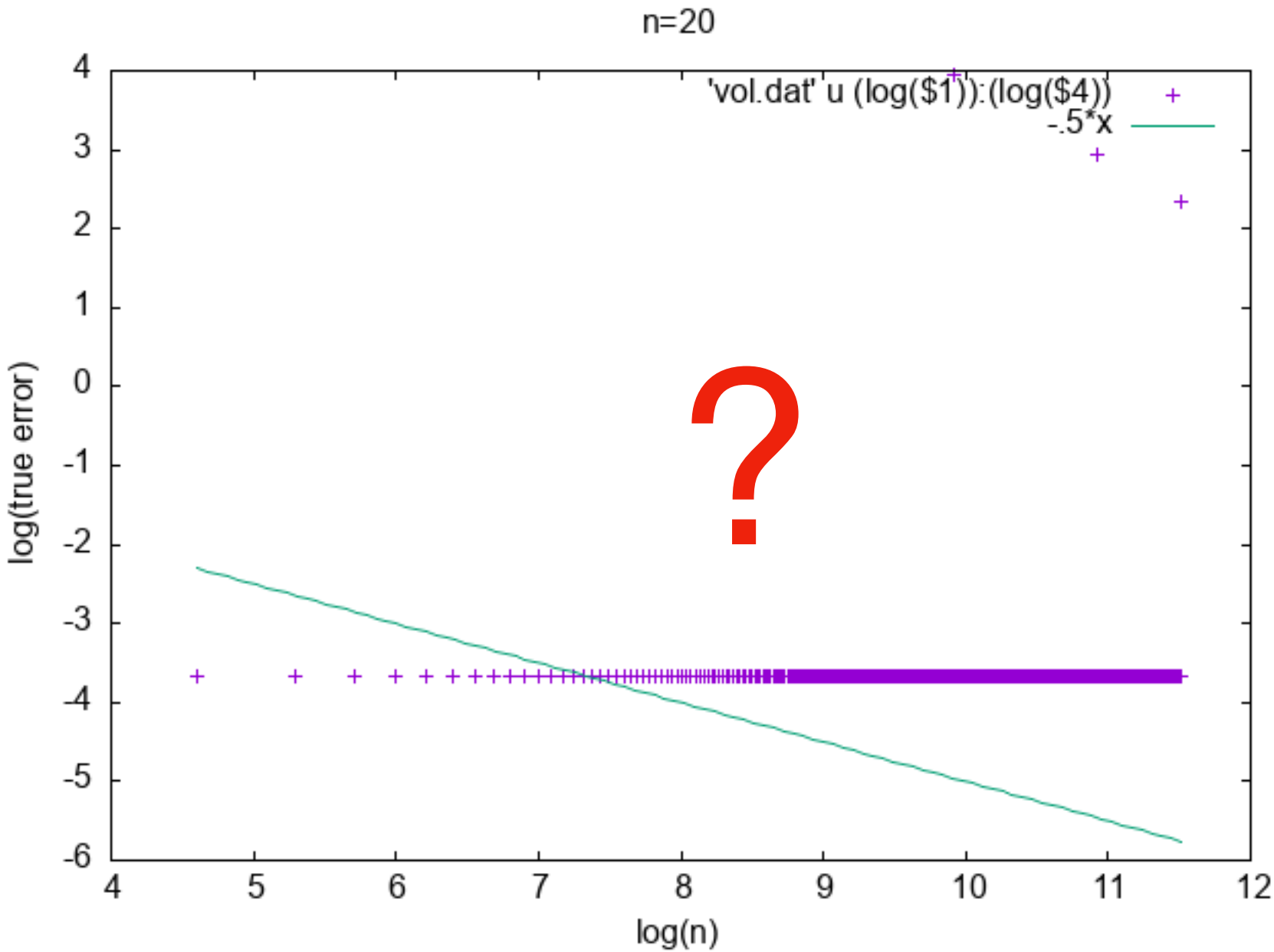


100000 punti - runs non correlati con diversi n
volume sfera numerico e esatto per diverse dimensioni

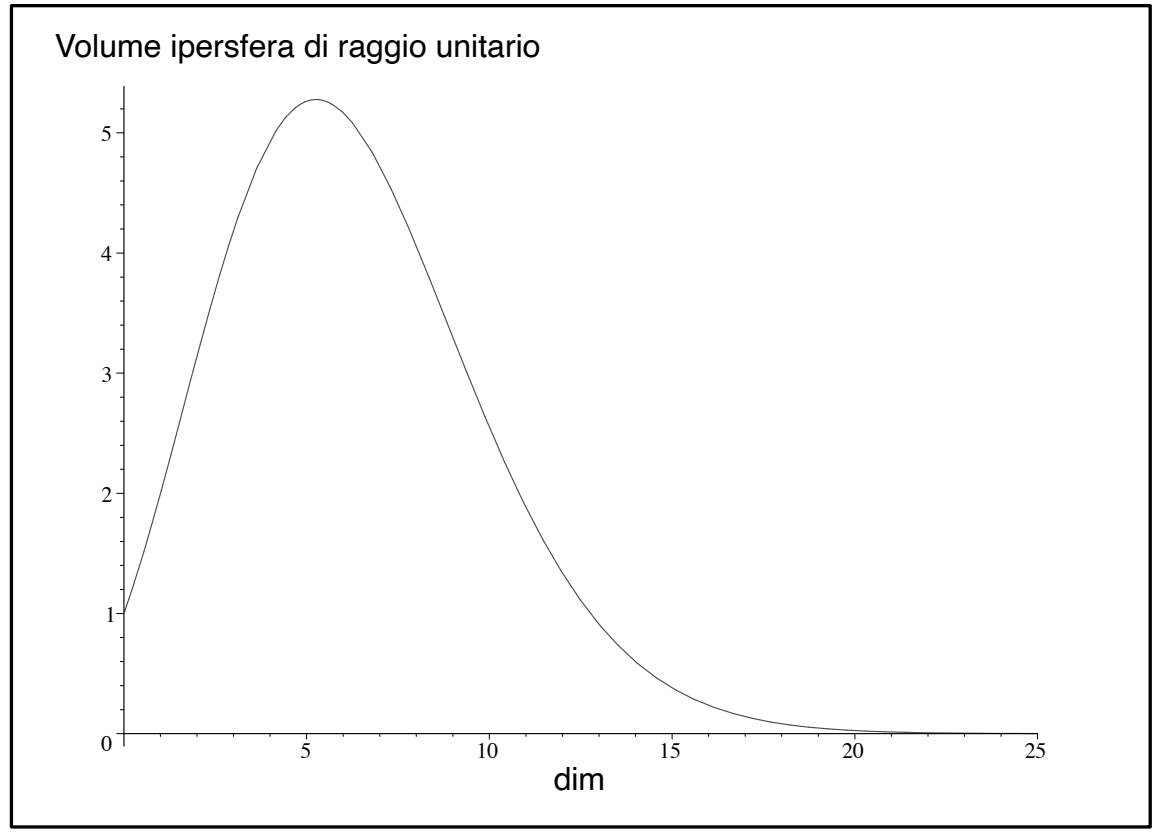
dim	V calc	V vero
10	2.82624006	2.55016422
12	1.43359995	1.33526301
15	0.327679992	0.381443381
16	0.655359983	0.235330686

perché il risultato peggiora con il crescere della dimensione?

100000 punti - runs non correlati con diversi n
volume sfera numerico e esatto: 0.00000000 2.58068983E-02



il volume dell'ipersfera ha un andamento non monotonous con la dimensione



crescendo con la dimensione, gli "spigoli vuoti" dell'ipercubo dov'è iscritta l'ipersfera pesano molto!
quindi di fatto si rifiutano molti dei punti generati
e il metodo accettazione-rifiuto diventa molto inefficiente!

2. **Monte Carlo method:**
generic sample mean and importance sampling

(a) Write a code to compute the numerical estimate F_n of $I = \int_0^1 e^{-x^2} dx = \frac{\sqrt{\pi}}{2} erf(1) \approx 0.746824$ with the MC *sample mean* method using a set $\{x_i\}$ of n random points uniformly distributed in [0,1]:

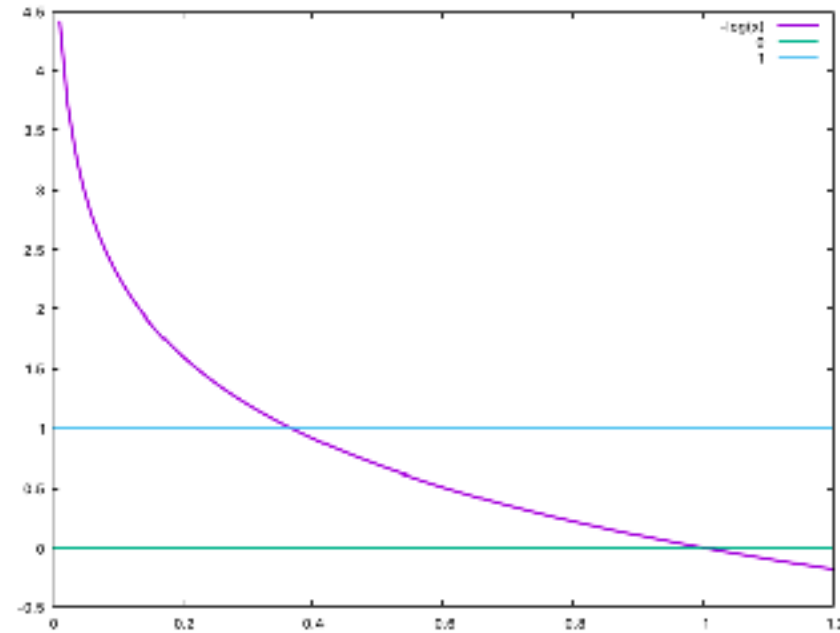
$$F_n = \frac{1}{n} \sum_{i=1}^{n} f(x_i)$$

(b) Write a code (a different one, or, better, a unique code with an option) to compute F_n using the *importance sampling* with a set $\{x_i\}$ of points generated according to the distribution $p(x) = Ae^{-x}$ (*Notice that* erf *is an intrinsic fortran function; useful to compare the numerical result with the true value*). Remind that in the *importance sampling* approach:

$$\int_a^b f(x)dx = \left\langle \frac{f(x)}{p(x)} \right\rangle \int_a^b p(x)dx \approx \frac{1}{n} \sum_{i=1}^{n} \frac{f(x_i)}{p(x_i)} \int_a^b p(x)dx = F_n$$

with $p(x)$ which approximates the behaviour of $f(x)$, and the average is calculated over the random points $\{x_i\}$ with distribution $p(x)$.

Punto di attenzione:
servono punti con distribuzione esponenziale tra 0 e 1 :
è possibile limitare il numero random
con distribuzione uniiforme in ingresso
alla subroutine expdev(rnd)



exponential variate $0 \le rnd < 1 \Rightarrow x = -\ln(rnd)$ results in $x > 0$

hence $0 < x < 1 \Rightarrow \frac{1}{e} < rnd < 1$

but since $0 \le rnd < 1 => rnd' = \frac{1}{e} + \left(1 - \frac{1}{e}\right) * rnd$ results in $\frac{1}{e} \le rnd' < 1$

4. **Random numbers with gaussian distribution: Metropolis algorithm**

Here we use the Metropolis algorithm to generate points with the distribution $P(x) = e^{-x^2/(2\sigma^2)}$. The algorithm is implemented for instance in the code `gauss_metropolis.f90`. We consider $\sigma = 1$, but the suggestion is to write the code for a generic σ.

The Metropolis algorithm

$p(x)$ is given.

If the "walker" is at position x_i and we wish to generate x_{i+1}, we can implement this choice of $T(x_i \rightarrow x_j)$ by the following steps:

1. Choose a trial position $x_{\text{trial}} = x_i + \delta_i$, where δ_i is a random number in the interval $[-\delta, \delta]$.
2. Calculate $w = p(x_{\text{trial}})/p(x_i)$.
3. If $w \geq 1$, accept the change and let $x_{i+1} = x_{\text{trial}}$.

else

4. If $w < 1$, generate a random number r.
5. If $r \leq w$, accept the change and let $x_{i+1} = x_{\text{trial}}$.
6. If the trial change is not accepted, then let $x_{i+1} = x_i$.

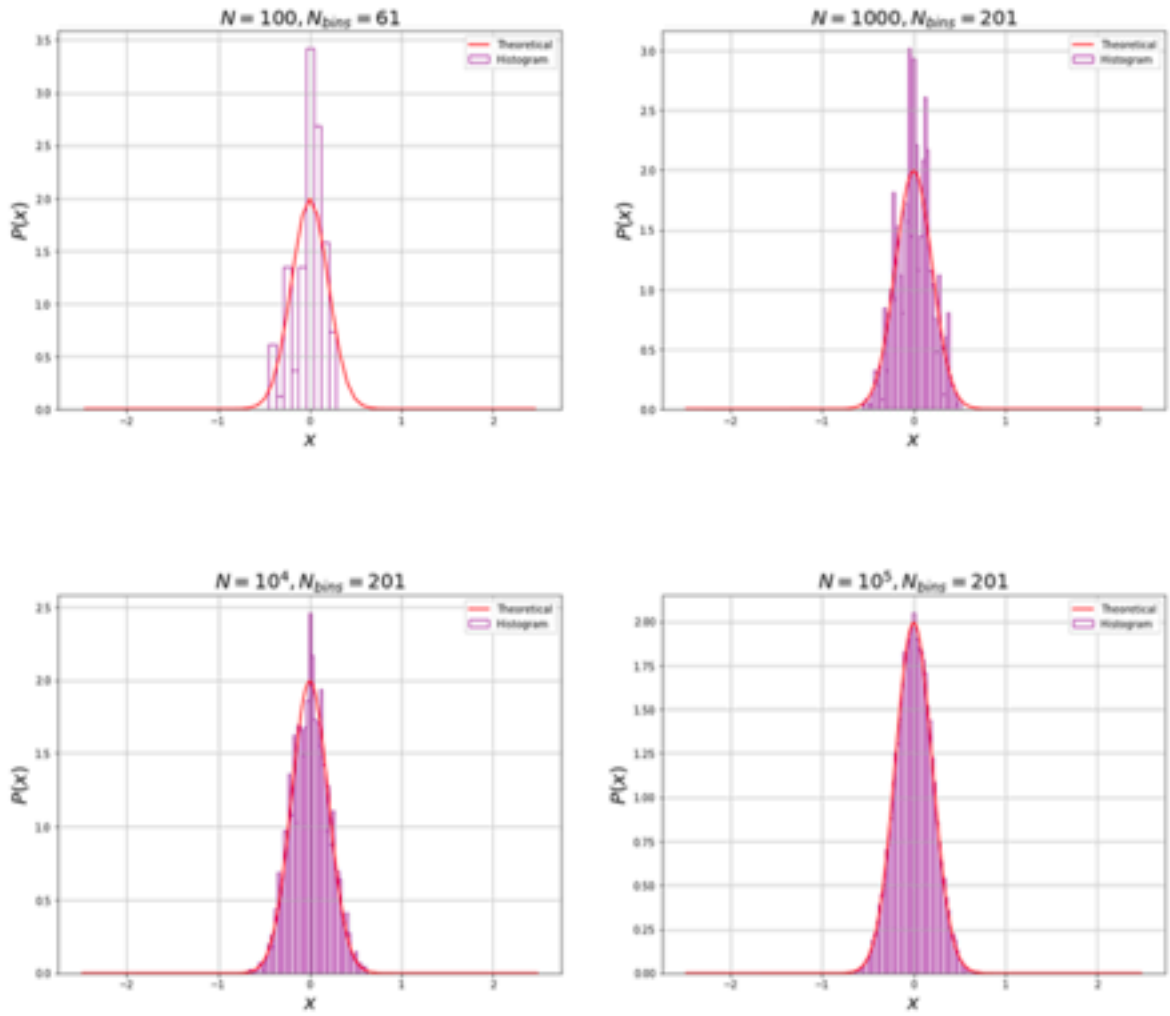
The algorithm from 1) to 6) has to be repeated until the distribution $p(x)$ of the points $\{x_i\}$ is reached.

Some issues:

- how to choose x_0 ?
  Convenient to start from a maximum
- how to choose δ ?
  (if too small, most trial steps accepted, but the walker moves too slowly; if too large, only a few trial steps are accepted...)
  A good compromise is a choice accepting from ~ 1/3 to ~1/2 of the trial steps
- equilibration is necessary (how many steps?)
  A possible criterion based on error estimate

(a)

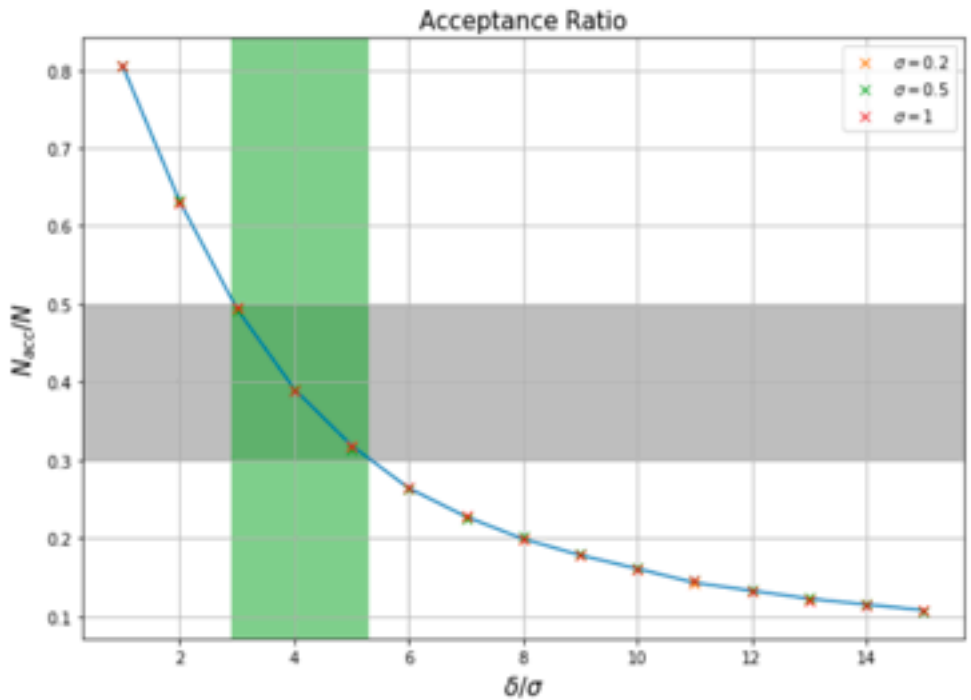
Si considera $x_0 = 0, \sigma = 0.2$ e $\delta = 5\sigma$



Per $N \sim 10^4, 10^5$ si ha un un buon accordo fra la distribuzione teorica e quella ottenuta numericamente con l'algoritmo Metropolis

(b)

Si fissa $N = 10^5$ e si studia l'acceptance ratio al variare di δ, fissato σ, ovvero in funzione del rapporto δ/σ

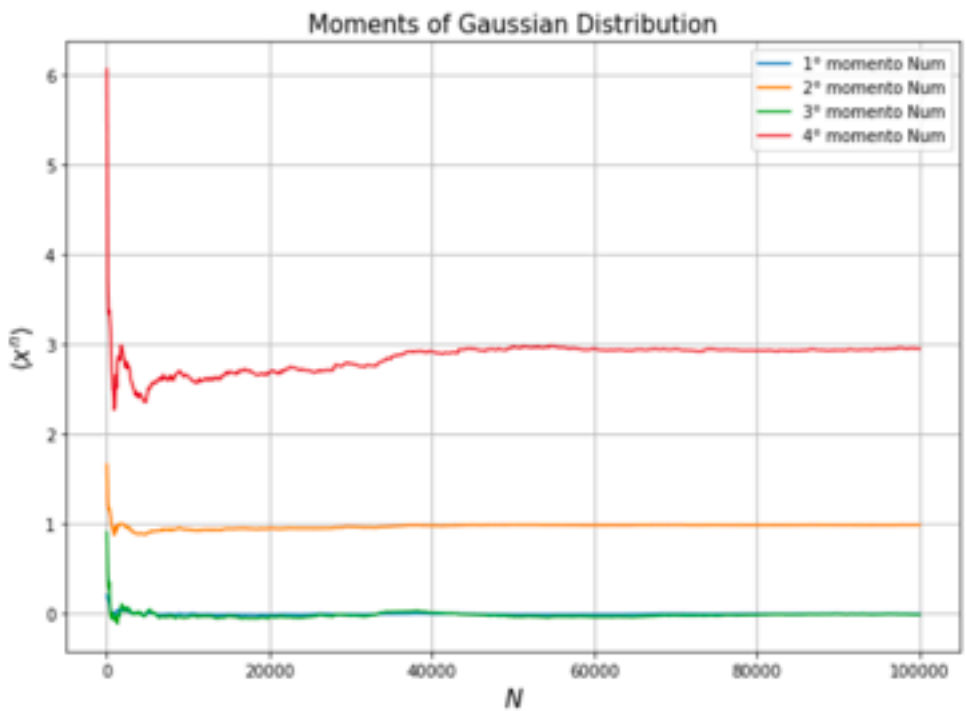


Per una distribuzione gaussiana con media μ e varianza σ si ha la seguente relazione per i momenti

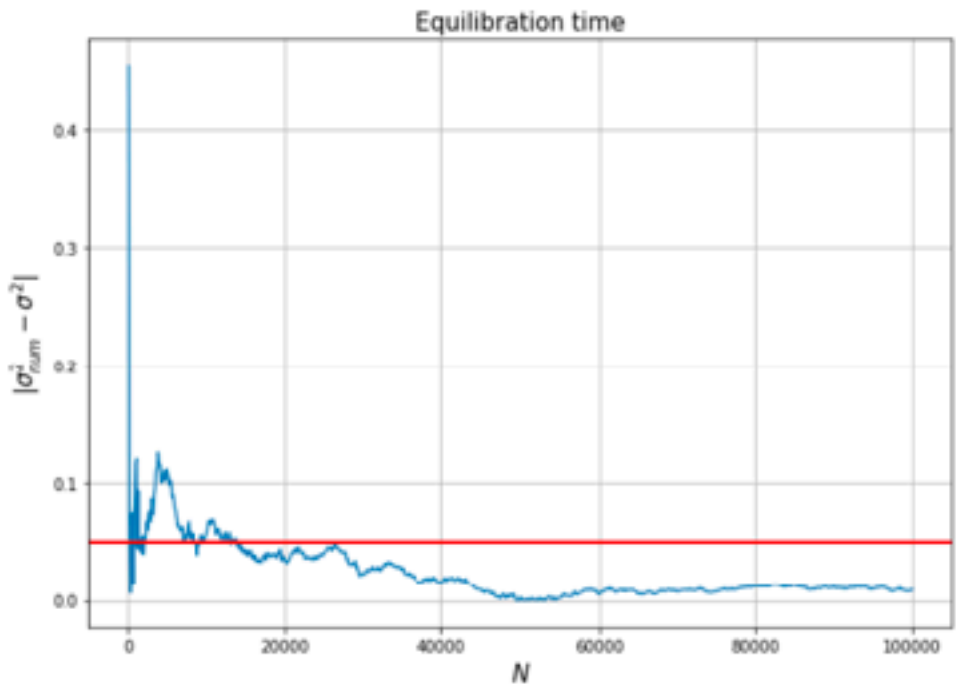
$$\langle x^n \rangle = \begin{cases} (n-1)!! \cdot \sigma^n, & n \text{ even} \\ 0, & n \text{ odd} \end{cases} \tag{2}$$

Si fissa $\mu = 0, \sigma = 1$ e $\delta = 5\sigma$ e si studiano i primi quattro momenti della distribuzione, dalla formula precedente si ottiene

$$\langle x \rangle = 0 \qquad \langle x^2 \rangle = 1 \qquad \langle x^3 \rangle = 0 \qquad \langle x^4 \rangle = 3 \tag{3}$$



In particolare si studia l'andamento della varianza stimata numericamente rispetto al valore teorico che può essere utilizzato per stimare l' equilibration time



Dal grafico di osserva che a partire da $N^* = 3 \cdot 10^4$ l'errore fra varianza teorica e numerica risulta minore del 5%, si può stimare dunque con N^* l'equilibration time