Aalto University
School of Science

PHYS-E0412 Computational Physics spring 2020

Emppu Salonen
Ville Havu
Filippo Zonta

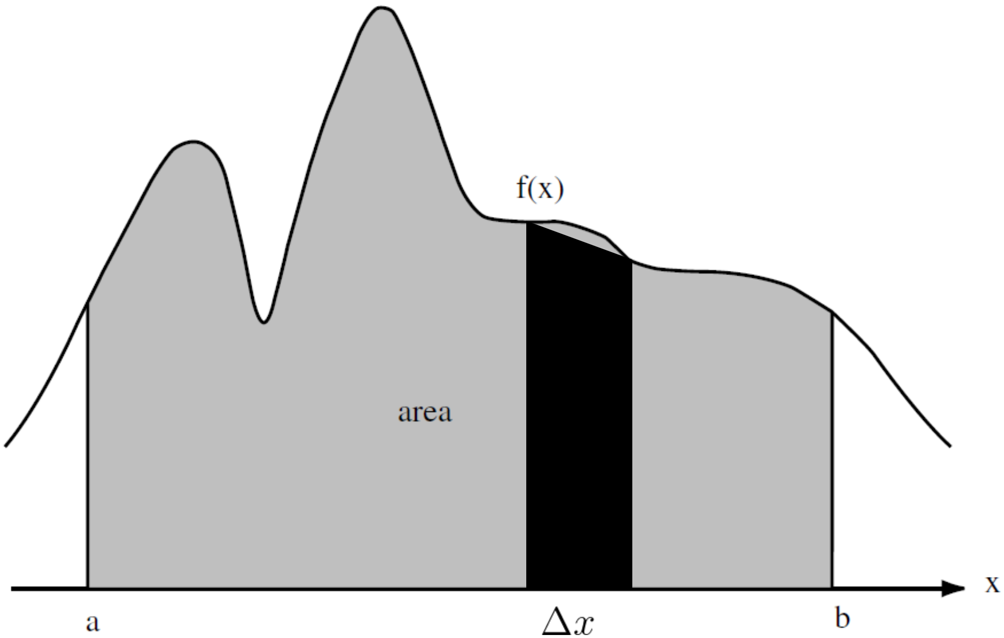
Lecture 3: Monte Carlo integration and error analysis

Tuesday 21.1.2020

Aalto University
School of Science

NUMERICAL INTEGRATION

$$F = \int_a^b f(x)dx$$



N intervals of width

$$\Delta x = \frac{b-a}{N}$$

Various ways of estimating the integral: rectangular and trapezoidal approximations, parabolic approximation (Simpson's rule) etc.

For example, trapezoidal approximation:

$$F \approx \left[\frac{1}{2}f(x_0) + \sum_i^{N-1} f(x_i) + \frac{1}{2}f(x_N)\right]\Delta x$$

NUMERICAL INTEGRATION: *d > 1* DIMENSIONS

$$F = V^{(d+1)} = \frac{(b_1 - a_1)(b_2 - a_2)...(b_d - a_d)}{N_1 N_2 ... N_d} \sum_{i_1}^{N_1} \sum_{i_2}^{N_2} ... \sum_{i_d}^{N_d} f(\vec{x}_i)$$

This can be written as ($N = N_1 N_2 \ldots N_d$ being the total number of points)

$$F = V^{(d+1)} = \frac{V^d}{N} \sum_{i_1}^{N} \sum_{i_2}^{N} ... \sum_{i_d}^{N} f(\vec{x}_i) = V^d \frac{\sum_{i_1}^{N} \sum_{i_2}^{N} ... \sum_{i_d}^{N} f(\vec{x}_i)}{N}$$

$$\Leftrightarrow \boxed{F = V^d \langle f \rangle}$$

This is basically an approximate form of the *mean value theorem for integrals.*

NUMERICAL INTEGRATION: ERROR

Error in 1D rectangular ~ N^{-1}
trapezoidal ~ N^{-2}
parabolic ~ N^{-4}

The error depends on the bin width Δx. To which extent (power), depends on the method. See ***Gould, Tobochnik & Christian, Appendix 11A.***

In general, if the error of the method in 1D scales as

$$N^{-a}$$

in *d* dimensions it scales as

$$N^{-a/d}$$

for *N* points used for evaluating the integral.

Bottom line*: with high-dimensional integrals we both need a huge number of points for evaluating the integral and the error of the calculation decreases very weakly with N.*

LEARNING OBJECTIVES FOR WEEK 3

Case study: Coulomb interaction energy between Gaussian charges in 3D using the Metropolis importance sampling

1. Inverse transform sampling (1D)
2. Simple Monte Carlo integration and error analysis
3. Importance sampling Monte Carlo
4. The Metropolis method

Inverse transform sampling (1D)

This topic was covered on the blackboard at the lecture. See this week's lecture notes.

Also: Gould, Tobochnik & Christian 11.5

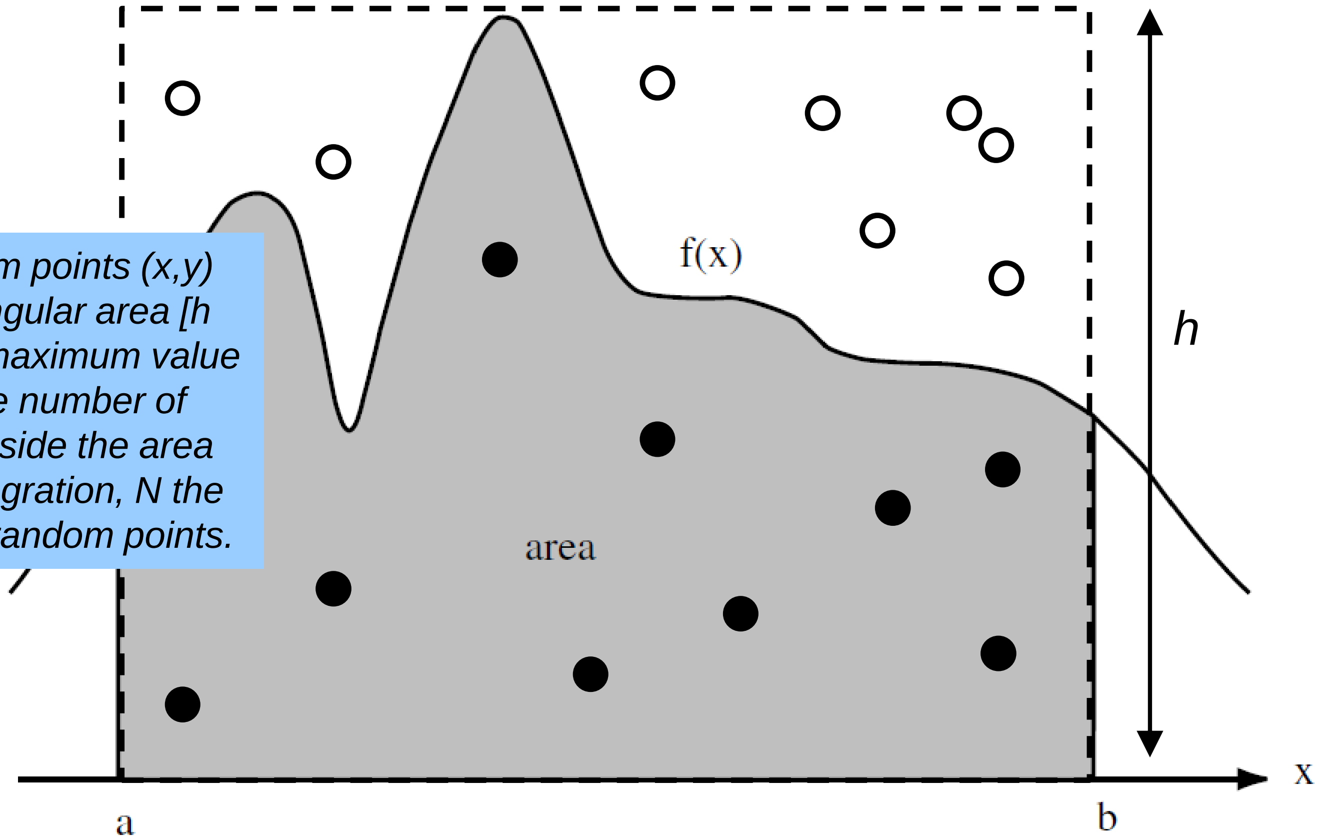
Aalto University
School of Science

Basic Monte Carlo integration

Aalto University
School of Science

MONTE CARLO: HIT AND MISS

$$F_N = h(b-a)\frac{N_{\text{hit}}}{N}$$

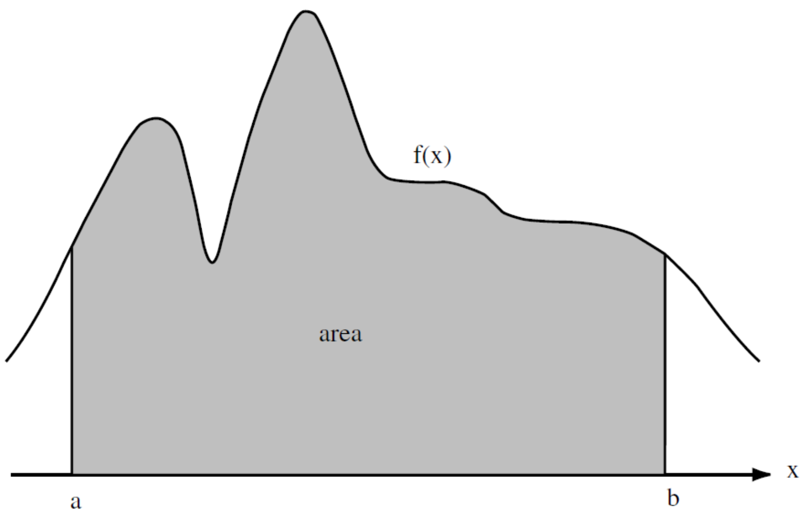


Generate random points (x,y) inside the rectangular area [h larger than the maximum value of f(x)]. N_{hit} is the number of points that fall inside the area given by the integration, N the total number of random points.

MONTE CARLO INTEGRATION

According to the mean value theorem

$$F = \int_a^b f(x)dx = (b-a)\langle f \rangle$$



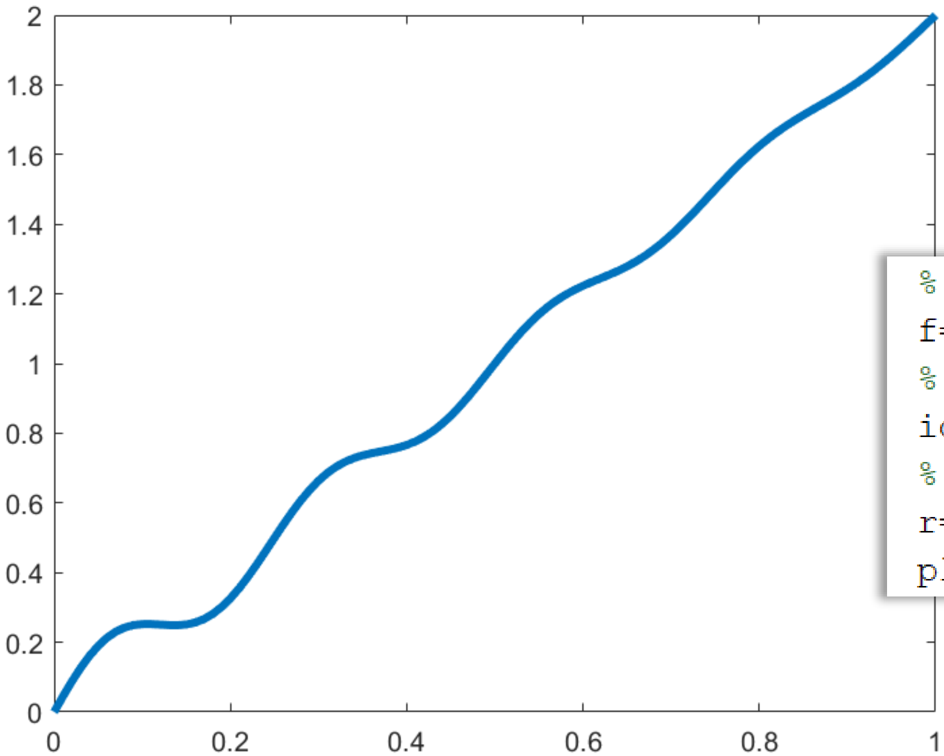
Plain Monte Carlo integration:

$$F_N = (b-a)\frac{1}{N}\sum_i^N f(x_i) \approx (b-a)\langle f \rangle$$

With *N* data points, the error of the mean is $$\frac{\sigma}{\sqrt{N}} \sim N^{-1/2}$$

TRADITIONAL INTEGRATION IN MATLAB

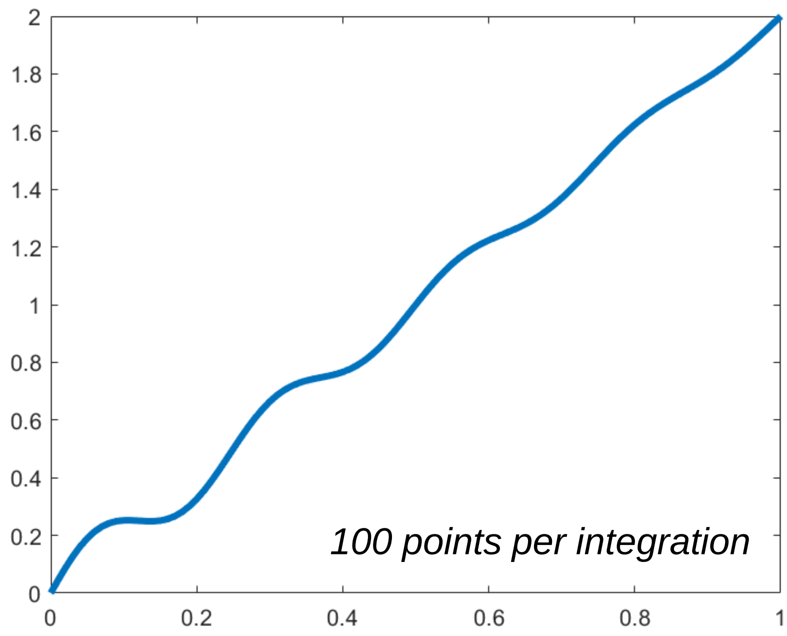
$$f(x) = 2x + \sin(8\pi x)\mathrm{erfc}(x)/10$$



There are Matlab routines for 1D, 2D, and 3D integrals but nothing beyond that.

```
% Function handle:
f=@(x) 2*x+.1*sin(8*pi*x).*erfc(x);
% Integration from zero to one:
iq=integral(f,0,1,'AbsTol',1e-10);
% Plot:
r=0:.01:1;
plot(r,feval(f,r),'linewidth',3)
```

SIMPLE MONTE CARLO



Matlab	Monte Carlo	Difference	Error estimate
1.0034	1.1483	0.14498	0.054058
1.0034	1.0698	0.066466	0.057181
1.0034	1.1045	0.10112	0.05802
1.0034	0.97395	0.029412	0.057082
1.0034	0.98416	0.019199	0.057364
1.0034	0.95521	0.048145	0.048703
1.0034	0.9706	0.032755	0.053725
1.0034	1.0588	0.055481	0.059431
1.0034	1.0219	0.018549	0.057678
1.0034	1.0542	0.050853	0.055569

```
mean_error =

    0.0567
```

```
mean_error=0;
Ns=10;
for i=1:Ns,
    f_values=feval(f,rand(N,1));
    im=mean(f_values);
    em=std(f_values)./sqrt(N);
    disp(num2str([iq, im, abs(im-iq), em]))
    mean_error=mean_error+abs(im-iq);
end

mean_error=mean_error/Ns
```

Consistent with the error estimate $\sigma/\sqrt{N}$

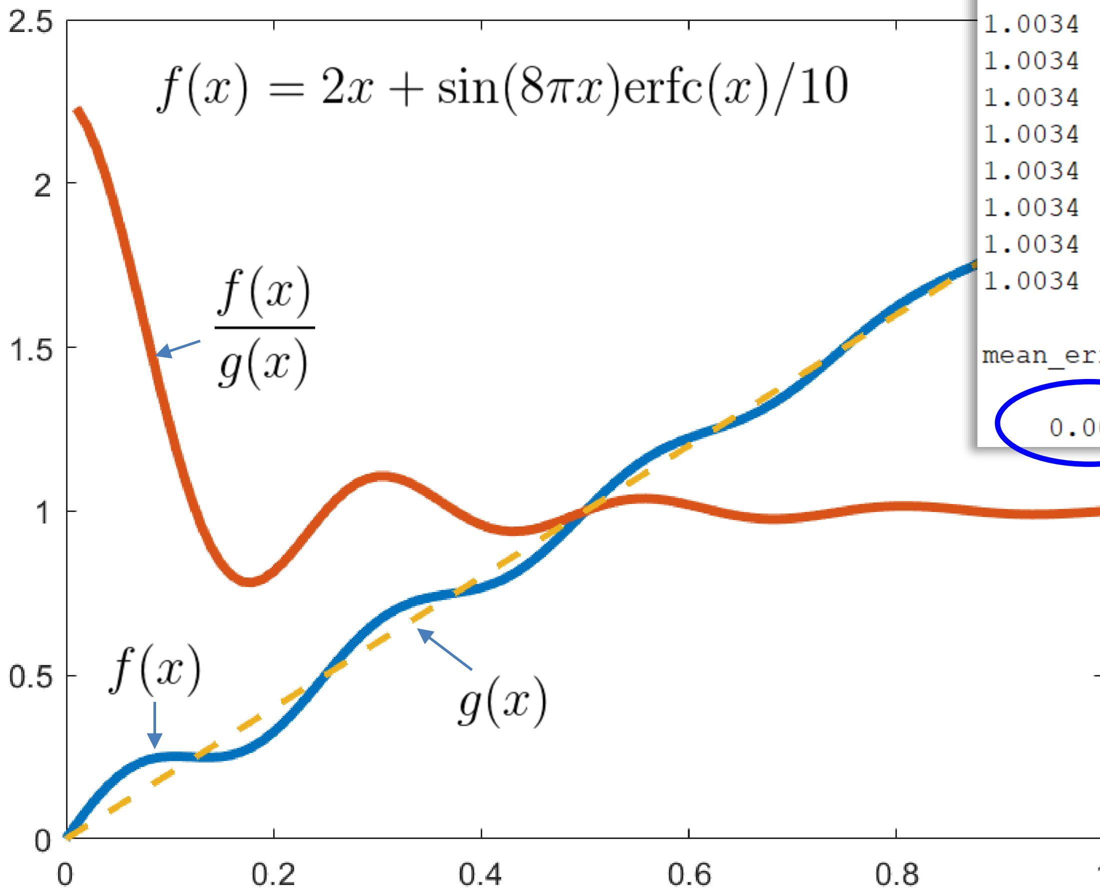
Can we make the error smaller?

Importance sampling Monte Carlo

This topic was covered on the blackboard at the lecture. See this week's lecture notes.

Also: Gould, Tobochnik & Christian 11.6

EXAMPLE, *g(x)* = 2*x*



Matlab	Monte Carlo	Difference	Error estimate
1.0034	0.99801	0.0053482	0.0041507
1.0034	1.0004	0.0029705	0.0065426
1.0034	1.0199	0.016512	0.012941
1.0034	1.0207	0.017298	0.01104
1.0034	1.0036	0.00022261	0.0092136
1.0034	1.0118	0.0084005	0.010771
1.0034	1.0063	0.0029477	0.0096535
1.0034	1.0138	0.010466	0.012228
1.0034	1.0007	0.002661	0.0061599
1.0034	0.99382	0.0095336	0.0037211

```
mean_error =

   0.0076
```

The error is an order of magnitude smaller than in the case of standard MC integration

Aalto University
School of Science

The Metropolis method

This topic was covered on the blackboard at the lecture. See this week's lecture notes.

Also: Gould, Tobochnik & Christian 11.7

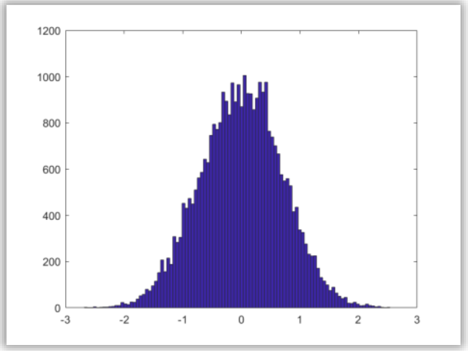
SIMPLE EXAMPLE

$$I = \frac{\int_{-\infty}^{\infty} x^p \exp(-x^2) dx}{\int_{-\infty}^{\infty} \exp(-x^2) dx}$$

"Importance sampling" both with $g(x) = C \exp(-x^2)$

We get the integral ratio $I \approx \frac{\langle x^p \rangle}{\langle 1 \rangle}$

```
g=@(x) exp(-x.^2);
x=0;
delta=5;
g_now=feval(g,x);
mean_fs=zeros(1,10);
N=2^15;
xx=zeros(N,1);
accepted=0;
for i=1:N,
    xt=x+(rand-.5)*delta;
    g_trial=feval(g,xt);
    if g_trial/g_now>rand,
        x=xt;
        g_now=g_trial;
        accepted=accepted+1;
    else
    % rejected, nothing done
    end
    xx(i)=x; % we store coordinates to show correct sampling
    mean_fs=mean_fs+(x.^(1:10)-mean_fs)./i;
end
disp(strcat('acceptance=',num2str(accepted/N)))
hist(xx,100)
```



p	Monte Carlo	Exact
1	0.0016	0
2	0.4915	0.5
3	-0.0090	0
4	0.7044	0.75
5	-0.0198	0
6	1.6418	1.8750
7	-0.0534	0