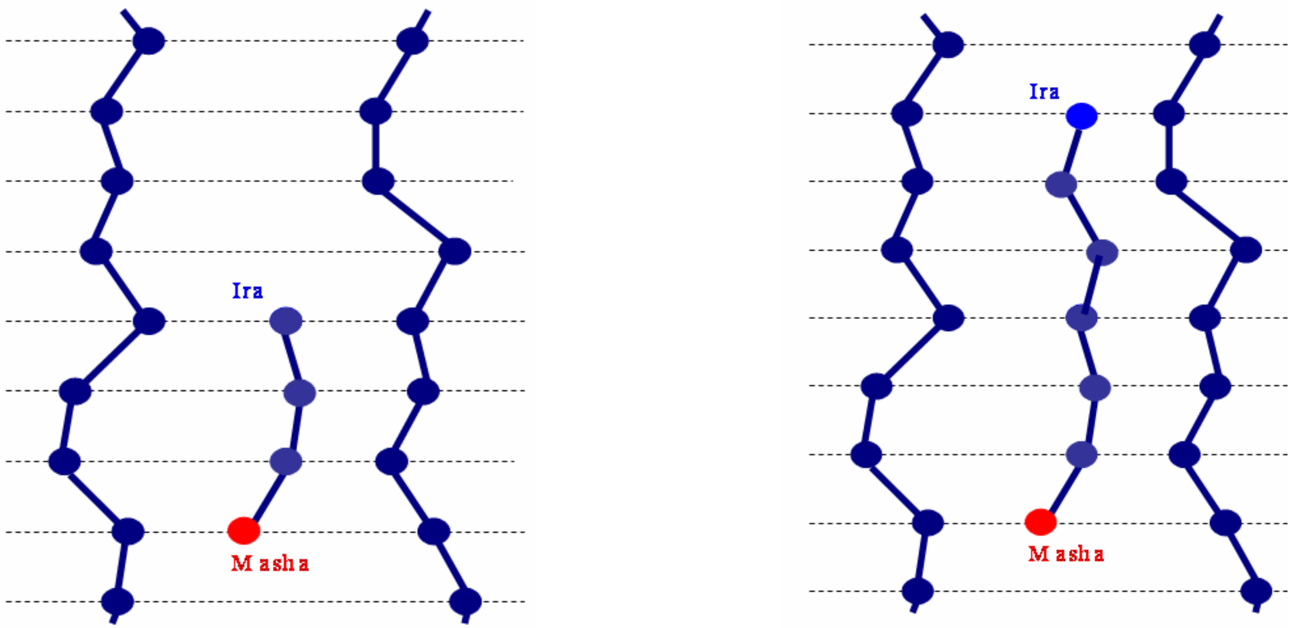
DIAGRAMMATIC MONTE CARLO LAB:

In Diagrammatic Quantum MC the number of variables is fluctuating:

$$Z(\vec{y}) = \sum_{n=0}^{\infty} \sum_{\xi} \iiint \underbrace{d\vec{x}_1 d\vec{x}_2 \ldots d\vec{x}_n}_{\text{Integration variables}} \; W_n(\xi; \vec{x}_1, \vec{x}_2, \ldots \vec{x}_n, \vec{y})$$

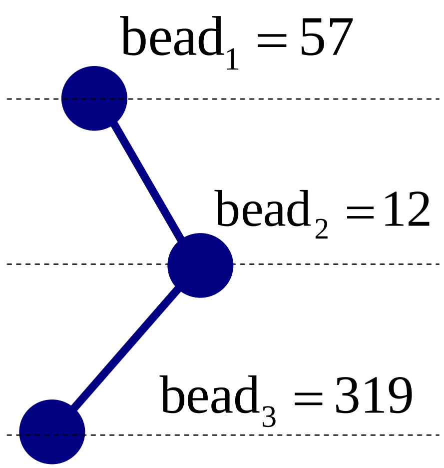
term order

For example, in Worm Algorithm PIMC one has to change the number of beads



(advance/recede update)

Data structure: linked arrays (common to all Diag.MC schemes)



bead3=prev(bead2)
bead2=next(bead3)

β **- periodicity is automatic**

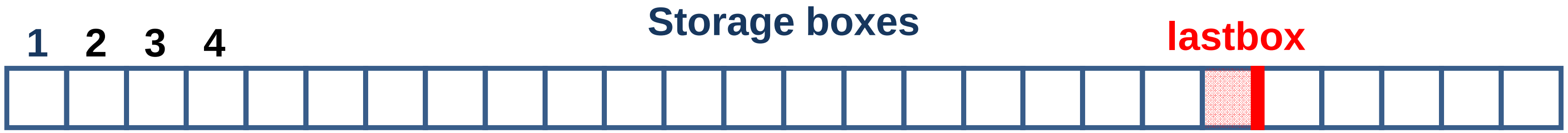
beads have the usual attributes:

R(bead)
tau(bead)
type(bead)

- spatial coordinates
- time slice
- particle type
etc.

Beads as objects must be assigned a unique ID number.

To ensure this in the simulation where beads are constantly created and deleted introduce two additional arrays.



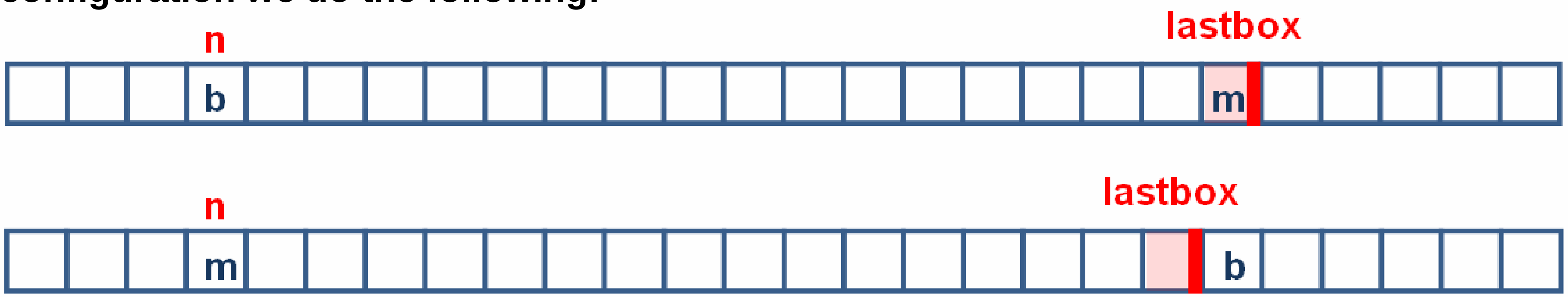
- **each box has an ID number**
- **IDs in boxes 1,2,3, …, lastbox are all used for existing beads**
- **IDs in boxes lastbox+1, lastbox+2, … are all free to use for new beads**

The actual arrays are:

box(bead) — **storage box number where the bead ID is "kept"**

ID(box) — **ID kept in the "storage box"**

Now, if a bead is eliminated from the configuration we do the following:



n=box(bead)
m=ID(lastbox)

collect information

ID(lastbox)=bead
ID(n)=m

exchange IDs places

lastbox=lastbox-1

Done! Eliminated Bead’s ID is now available for new beads

If a new bead is created we assign it ID from the lastbox+1

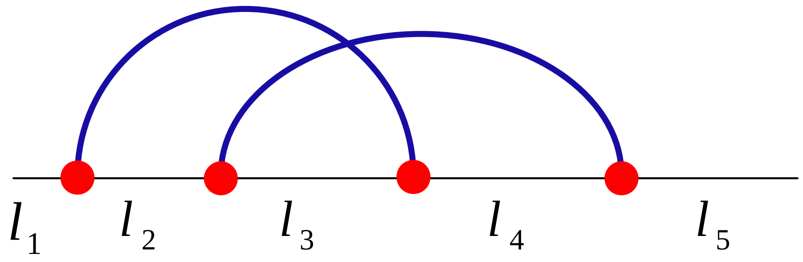
lastbox=lastbox+1
newbead=ID(lastbox)

Done!

In the config. space of Feynman diagrams one uses more than one linking array (topology is more complex).

In this example IDs are given to line elements:

l3=prev(l4)
l4=next(l3)
etc.

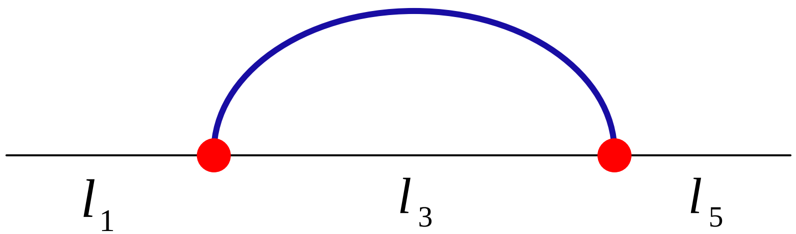


But also:

l2=link(l4)
l4=link(l2)

“left end” convention:

What is required is the minimal information to draw the graph. When updates are performed and graph elements are eliminated one has to update links



l3=next(l1)
l1=prev(l3)
l5=next(l3)
l3=prev(l5)

+ update line attrributes (mometum, duration in time, etc.)

Summation of divergent/asymptotic series.

What do you think of the following series?

$$A=\sum_{n=0}^{\infty} c_n = \sum_{n=0}^{\infty}(-1)^n = 1-1+1-1+1 \;\; \ldots \text{ (Grandi series)}$$

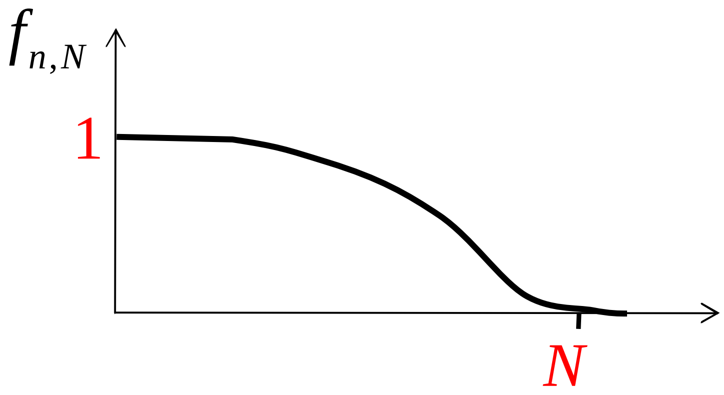
In Diag.MC you can get something of this kind (with c_n being the result of the simulation) but may divergent and oscillating more strongly, e.g. $A=1-5+25-125+625 \;\; \ldots$
Does the simulation make sense?

The answer is YES, all of this makes perfect sense, keep reducing error bars!

Define a finction $f_{n,N}$ which has the following shape, i.e.

$$f_{n,N} \to 1 \;\; \text{for} \;\; n << N$$

$$f_{n,N} \to 0 \;\; \text{for} \;\; n > N$$



Construct sums $A_N = \sum_{n=0}^{\infty} c_n f_{n,N}$ **and extrapolate** $\lim_{N\to\infty} A_N$ **to get** A

Let's try the Grandi series.

Introduce Cesaro function:

$$f_{n,N} = \frac{N-n}{N}$$

Calculate:

$$A_N = \sum_{n=0}^{N-1} c_n \; f_{n,N}$$

$$N = 1, 2, 3, \ldots$$

To get:

$$A_N = \frac{1}{2} + \begin{pmatrix} 0 & N = even \\ 1/2N & N = odd \end{pmatrix}$$

With the limit:

$$\lim_{N \to \infty} A_N = 1/2$$

Now, write a simple code doing the same job for:

$$f_{n,N} = \left(\frac{N-n}{N} \right)^p$$

Riesz-function for p=2,3…

Is the limit the same?

For p=2 the data should look like this and when plotted as a function of 1/N allow a perfect linear extrapolation.

1	1.000000
2	0.7500000
3	0.6666667
4	0.6250000
5	0.6000001
6	0.5833333
7	0.5714285
8	0.5625000
9	0.5555555
10	0.5500000
11	0.5454546
12	0.5416666
13	0.5384616
14	0.5357143
15	0.5333334
16	0.5312500
17	0.5294117
18	0.5277778
19	0.5263158
20	0.5250000

Generated by:

```
integer, parameter :: M=100, p=2
real :: s, a(1:M), f(0:M,1:M)

DO n=1,M
  DO k=0,n
    f(k,n)=((n-k)*1./n)**p
  ENDDO
ENDDO

DO n=1,M
a(n)=0
   s=1.
   DO k=0,n-1
      a(n)=a(n)+f(k,n)*s
   s=-s
   ENDDO
ENDDO

DO n=1,M
PRINT*,  n, a(n)
ENDDO

END
```

Try p=3,4,…

Well, now feel the power by trying

$$f_{n,N} = e^{-n^2/N} \text{ for } (n < N)$$

Main lesson:

When the re-summation method works the final answer is the same and method independent !
Re-summation determines an analytic function behind the series outside the radius of convergence.

In our case it was

$$A(x) = \frac{1}{1+x} = \sum_{n=0}^{\infty} (-1)^n x^n \quad \text{for} \;\; x = 1$$

and the final answer is 1/2 (for x>1 the "step-type" function f has to suppress exponentially growing high-order terms; otherwise f is arbitrary).

Thus series divergence is NOT a problem preventing one from using Diag.MC

Solving equations using Diag.MC

$$f = \frac{a}{1+u}$$

$$f = a - uf$$

use Diag.MC

MC scheme
each diagram → f

$$f_{n+1} = a - u\langle f \rangle_{MC,n}$$

$$\langle f \rangle_{MC,n} = \sum_{i=1}^{n} \frac{f_i}{n}$$

$$f_{n+1} = a - u\sum_{i=1}^{n} \frac{f_i}{n}$$

solve it iteratively

$f_0=a$
$f_{n+1}=a-uf_n$

$f_0=a$
$f_1=a-uf_0=a-ua$
$f_2=a-uf_1=a-ua+u^2a$
....

$f=a-ua+u^2a-u^3a+u_4a- \cdots$

|u|>1, divergent series:
resummation techniques

Write a simple program which mimicks a Monte Carlo calculation

$$a = 1 \qquad u = 2.5$$

$$f_1 = a$$

do loop

$$f_{n+1} = a - u \sum_{i=1}^{n} \frac{f_i}{n}$$

end do loop

1	1.
2	-1.5
3	1.625
4	0.0625
5	0.2578125
6	0.27734375
7	0.282226562
8	0.283970424
9	0.284733364
10	0.285114833
11	0.285324642
12	0.285448619
13	0.285526105
14	0.285576769
15	0.285611148
16	0.285635214
17	0.285652511
18	0.285665229
19	0.285674768
20	0.285682048

$$f = 0.285714286$$

```
double precision:: a=1.0,  u=2.5
double precision:: f_result, f_now, f_average
integer :: n=20,i

f_average = a
     f_now = f_average

do I = 1, n
     print*, i, f_now
   f_now = a – u * f_average/i
   f_average = f_average + f_now
enddo

f_result = f_now
print*, f_result
end
```