Computational methods for Medical Physics

Lecture 4: The convolution operation and its use in proton dose calculation

Dr. George Dedes

WS 2016-2017

Contents

- Lectures 1-3:
  - MC integration technique
  - Random sampling
  - Sampling from distributions
  - MC particle transport examples

- This Lecture:
  - Definition of Convolution
  - Convolution and its role in image processing and restoration
  - Basic properties and theorems
  - The pencil beam dose calculation algorithm in proton therapy

The convolution operation

- Let f and g be both Lebesgue integrable functions*

- We define the convolution operation $f * g$ as:

$$h(t) = (f * g)(t) = \int_{-\infty}^{\infty} f(\tau) g(t - \tau) d\tau$$

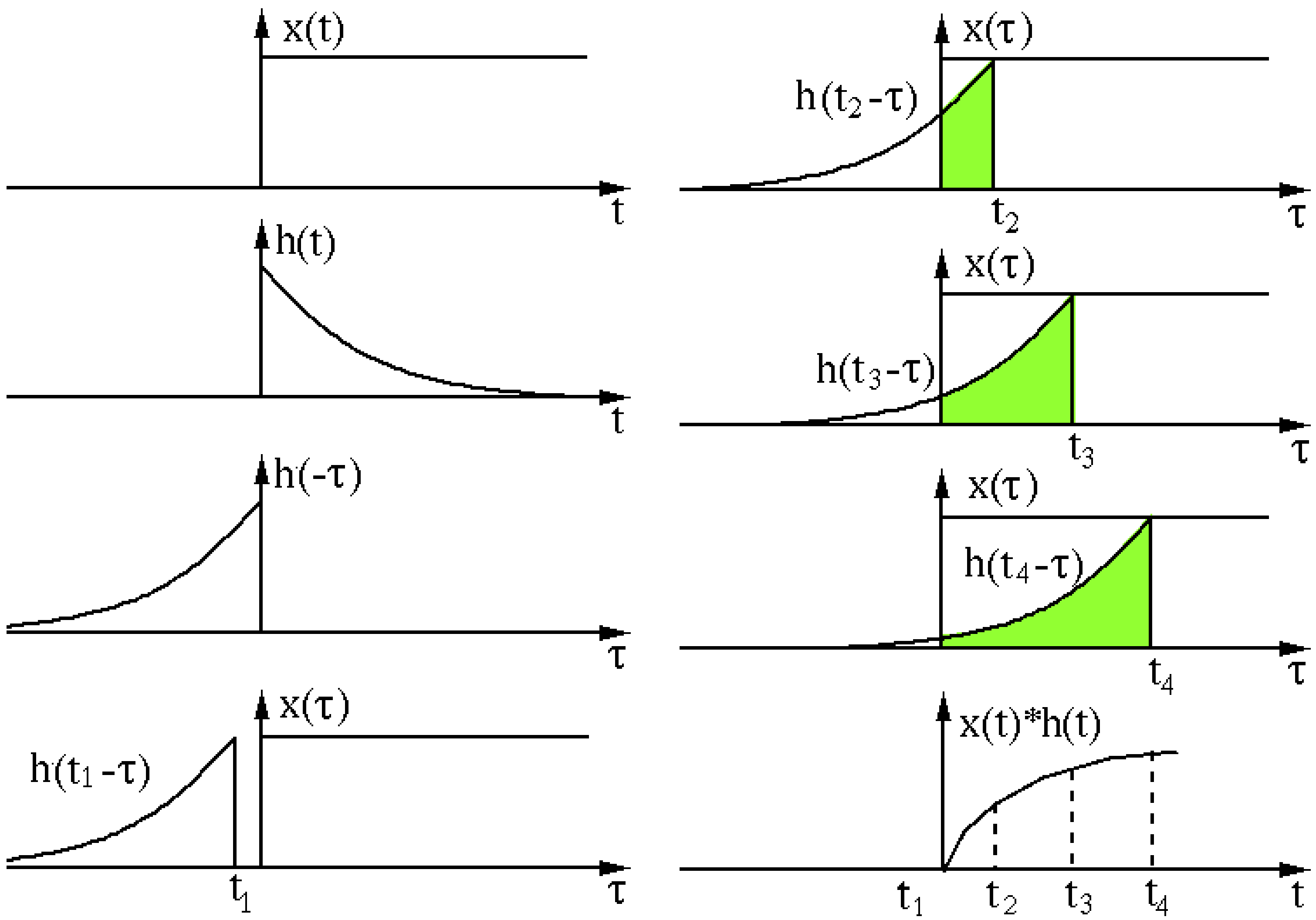
and we say that the function ***h*** *is the* ***convolution*** *of the functions* ***f*** *and* ***g***.

- The operation is similarly defined in higher dimensional spaces

*The condition of Lebesgue integrability is sufficient but not necessary

The convolution operation

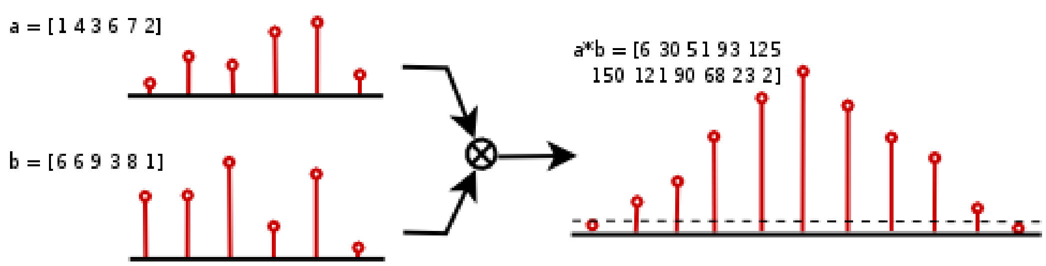
- Schematic representation of convolution x∗h



The convolution operation

- Convolution can also be defined for discrete signals
- In the majority of the time this is the type of signals we deal with

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$



- The length of the final vector is: length (f) + length (g) - 1

The convolution operation

Example:

$$\boxed{h[n]=(f*g)[n]=\sum_{m=-\infty}^{\infty} f[m]\cdot g[n-m]}$$

f= [1, 4, 7]
g=[3, 5, 2]

Index 0 Index 1 Index 2

$$(f*g)[0]=\sum_{m=-\infty}^{\infty} f[m]\cdot g[-m]=f[0]g[0]=3$$

$$(f*g)[1]=\sum_{m=-\infty}^{\infty} f[m]\cdot g[1-m]=f[0]g[1]+f[1]g[0]=1*5+4*3=17$$

$$(f*g)[2]=\sum_{m=-\infty}^{\infty} f[m]\cdot g[2-m]=f[0]g[2]+f[1]g[1]+f[2]g[0]=1*2+4*5+7*3=43$$

$$(f*g)[3]=\sum_{m=-\infty}^{\infty} f[m]\cdot g[3-m]=f[0]g[3]+f[1]g[2]+f[2]g[1]+f[3]g[0]=0+4*2+5*7+0=43$$

$$(f*g)[4]=\sum_{m=-\infty}^{\infty} f[m]\cdot g[4-m]=f[2]g[2]=2*7=14$$

$$(f*g)[5]=\sum_{m=-\infty}^{\infty} f[m]\cdot g[5-m]=0$$

$$\boxed{h=(f*g)=[3,17,43,43,14]}$$

The convolution operation

- The convolution operation appears in many fields and applications:
  - The potential of a field caused by an extended body (Physics)
  - The distribution of the sum of two random variables (Probability)
  - Filtering (Image processing)
  - Image deblurring (Image processing)
  - Spectra unfolding (Radiation detection)
  - Dose calculation in radiation therapy (Medical Physics)

The convolution operation

- The convolution operation appears in many fields and applications:
  - The potential of a field caused by an extended body (Physics)
  - The distribution of the sum of two random variables (Probability)
  - Filtering (Image processing)
  - Image deblurring (Image processing)
  - Spectra unfolding (Radiation detection)
  - Dose calculation in radiation therapy (Medical Physics)
- Why the convolution operation appears in so many, seemingly completely different problems?

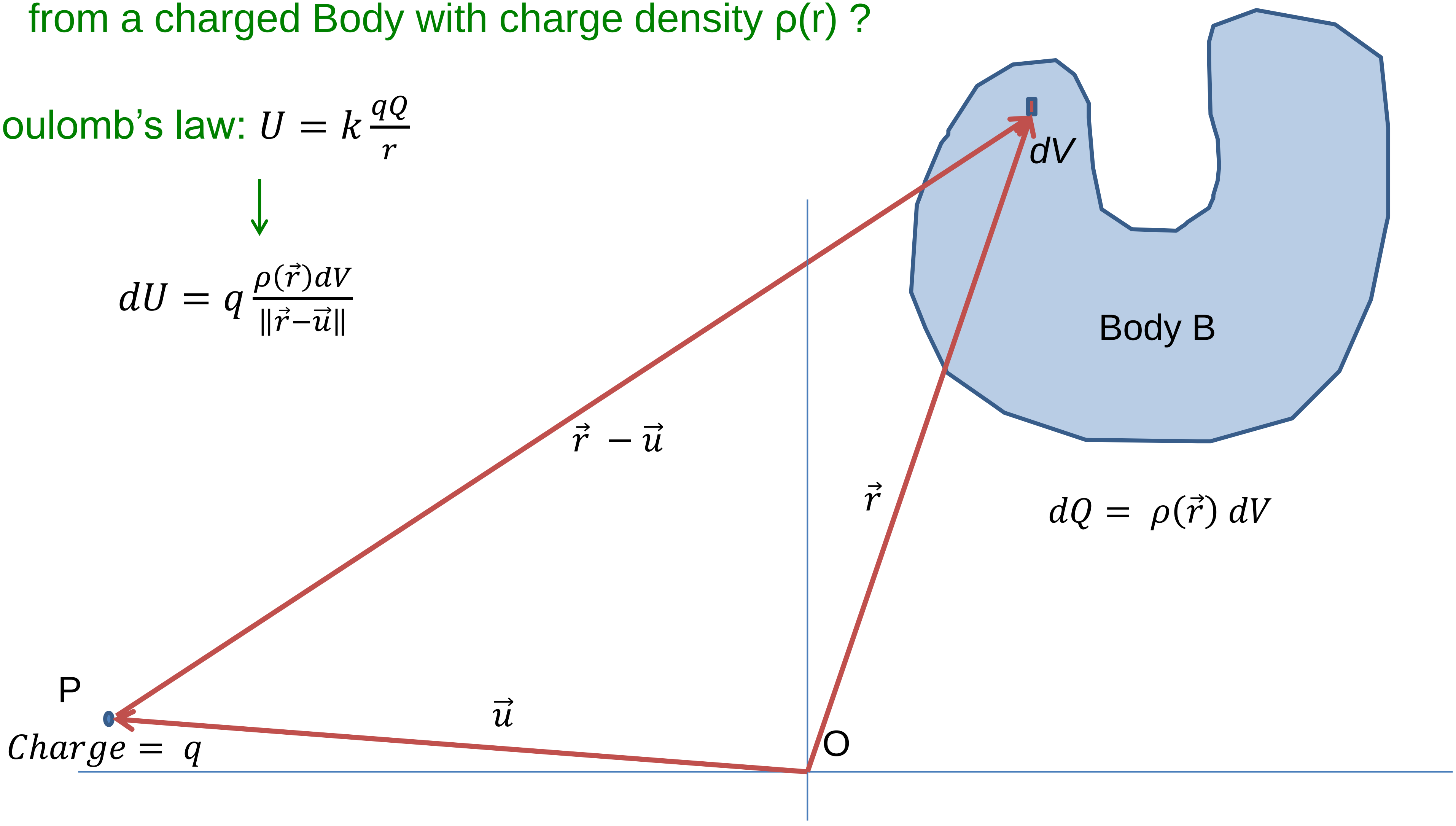
The convolution operation
General example

- The electrostatic problem:
  What is the electrostatic potential of experienced by a point charge P from a charged Body with charge density ρ(r) ?

Coulomb's law: $U = k\frac{qQ}{r}$

$\downarrow$

$$dU = q\frac{\rho(\vec{r})dV}{\|\vec{r}-\vec{u}\|}$$



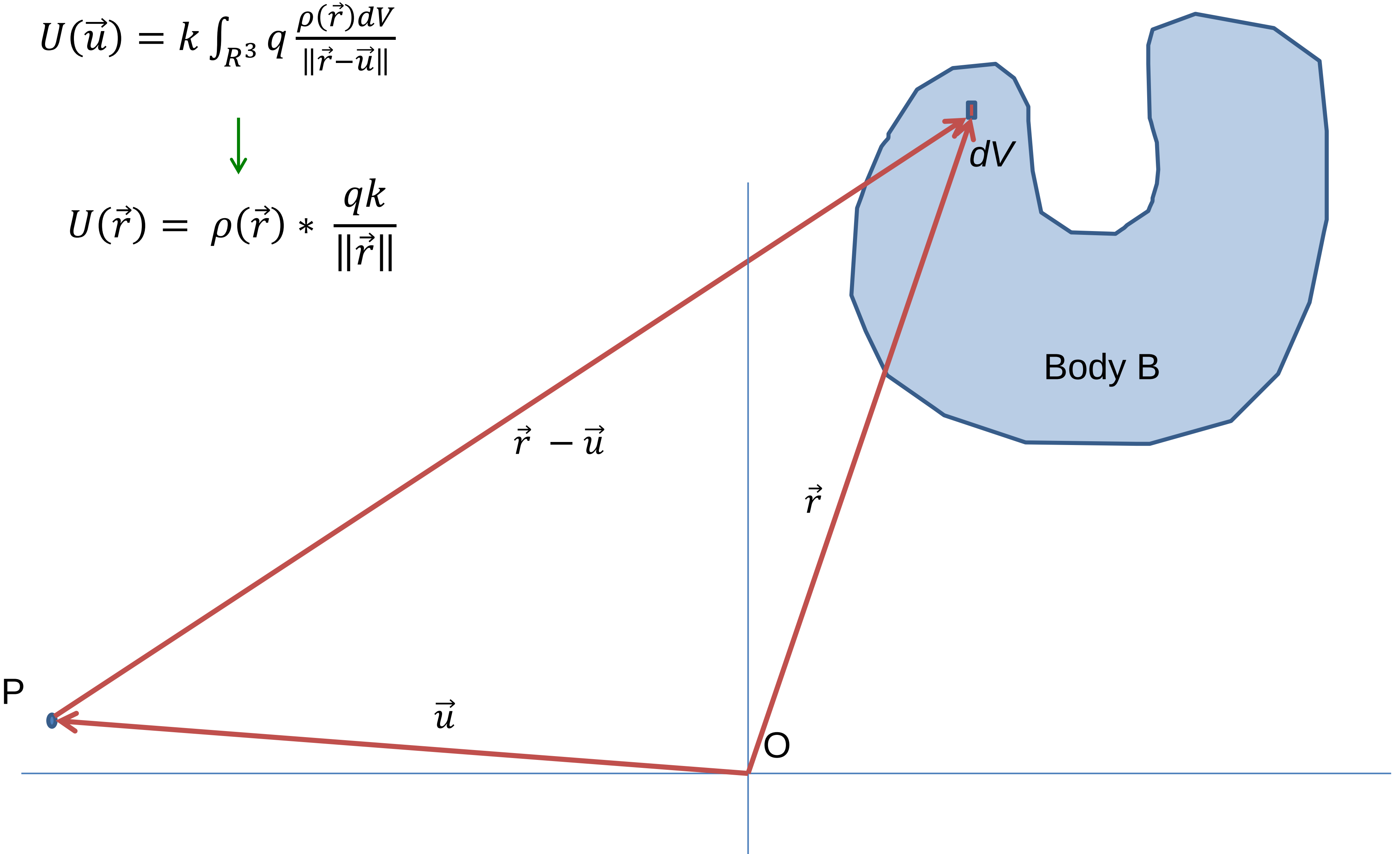
The convolution operation
General example

- The electrostatic problem:

$$U(\vec{u}) = k \int_{R^3} q \frac{\rho(\vec{r})dV}{\|\vec{r}-\vec{u}\|}$$

$$\downarrow$$

$$U(\vec{r}) = \rho(\vec{r}) * \frac{qk}{\|\vec{r}\|}$$



The convolution operation
General example

- The electrostatic problem:
  What is the electrostatic potential of experienced by a point charge P from a charged Body with charge density ρ(r) ?

$$\boldsymbol{U(\vec{r}) = \rho(\vec{r}) * \frac{qk}{||\vec{r}||}}$$

This is an **arbitrary**
charge distribution
(it may be different in
every problem)

This is Coulomb's law
(**a Universal Law)**

Properties of convolution

Commutative: $$(f * g) = (g * f)$$

Associative: $$(f * g) * h = f * (g * h)$$

Distributive: $$f * (g + h) = (f * g) + (f * h)$$

With delta function: $$f * \delta = f$$

With derivation: $$\partial_x (f * g) = (\partial_x f * g)$$

Properties of convolution

- Complexity of the convolution operation:

$$(f*g)[n]=\sum_{m=-\infty}^{\infty} f[m]\cdot g[n-m]$$

- The complexity of the above operation is $O(n^2)$
- It roughly means that the CPU/memory/time/storage scales with n^2

The convolution operation 2D

- Just as in the continuous case, convolution can also be defined for discrete signals of higher dimensions:

$$(f * g)[x, y] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m, n] \cdot g[x-m, y-n]$$

- Example:

1	2	3
4	5	6
7	8	9

Input f

$*$

-1	-2	-1
0	0	0
1	2	1

Kernel g

$=$

?

Result f$*$g

The convolution operation 2D

- Example:

$$\begin{array}{c|ccc} x\backslash y & 0 & 1 & 2 \\ \hline 0 & 1 & 2 & 3 \\ 1 & 4 & 5 & 6 \\ 2 & 7 & 8 & 9 \end{array} \; * \; \begin{array}{c|ccc} x\backslash y & 0 & 1 & 2 \\ \hline 0 & -1 & -2 & -1 \\ 1 & 0 & 0 & 0 \\ 2 & 1 & 2 & 1 \end{array}$$

Input f   Kernel g

$$(f*g)[x,y]=\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} f[m,n]\cdot g[x-m,y-n]$$

$$(f*g)[0,0]=\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} f[m,n]\cdot g[-m,-n]=\sum_{m=-\infty}^{\infty}\left(f[m,0]\cdot g[-m,0]\right)=f[0,0]\cdot g[0,0]=-1$$

$$(f*g)[0,1]=\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} f[m,n]\cdot g[-m,1-n]=\sum_{m=-\infty}^{\infty}\left(f[m,0]\cdot g[-m,1]+f[m,1]\cdot g[-m,0]\right)$$

$$=f[0,0]\cdot g[0,1]+f[0,1]\cdot g[0,0]=-4$$

Continue like that until we fill a matrix with dimensions $x' = x_f + x_g - 1$ and $y' = y_f + y_g - 1$

The convolution operation 2D

- Example:

$$(f*g)[x,y]=\sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} f[m,n]\cdot g[x-m,y-n]$$

Input f

1	2	3
4	5	6
7	8	9

*

Kernel g

-1	-2	-1
0	0	0
1	2	1

=

-1	-4	-8	-8	-3
-4	-13	-20	-17	-6
-6	-18	-24	-18	-6
4	13	20	17	6
7	22	32	26	9

The convolution operation 2D

- Example (with a trick): $(f * g)[x,y] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m,n] \cdot g[x-m, y-n]$

- Take kernel g

-1	-2	-1
0	0	0
1	2	1

Kernel g

The convolution operation 2D

- Example (with a trick): $$(f * g)[x, y] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m, n] \cdot g[x-m, y-n]$$

- Take kernel g, flip it in both dimensions

1	2	1
0	0	0
-1	-2	-1

Kernel g

The convolution operation 2D

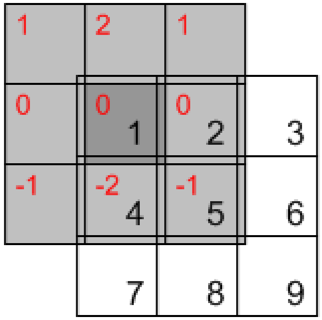
- Example (with a trick): $$(f * g)[x,y] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m,n] \cdot g[x-m, y-n]$$

- Take kernel g, flip it in both dimensions

1	2	1
0	0	0
-1	-2	-1

Kernel g

- Apply it on the input (element by element multiplication and addition)



= -13

The convolution operation 2D

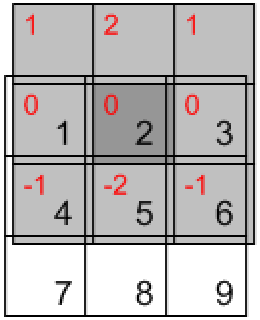
- Example (with a trick): $$(f * g)[x,y] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m,n] \cdot g[x-m, y-n]$$

- Take kernel g, flip it in both dimensions

1	2	1
0	0	0
-1	-2	-1

Kernel g

- Apply it on the input (element by element multiplication and addition)



$= -20$

The convolution operation 2D

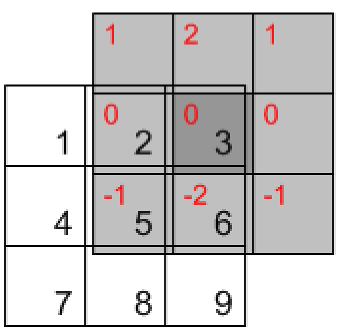
- Example (with a trick): $(f * g)[x,y] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m,n] \cdot g[x-m, y-n]$

- Take kernel g, flip it in both dimensions

1	2	1
0	0	0
-1	-2	-1

Kernel g

- Apply it on the input (element by element multiplication and addition)



= -17

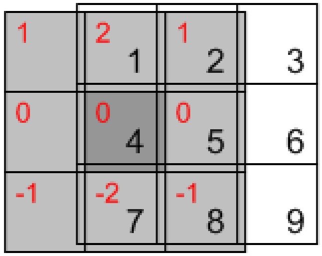
The convolution operation 2D

- Example (with a trick): $$(f*g)[x,y] = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} f[m,n]\cdot g[x-m,y-n]$$
- Take kernel g, flip it in both dimensions

1	2	1
0	0	0
-1	-2	-1

Kernel g

- Apply it on the input (element by element multiplication and addition)



= -18

The convolution operation 2D

- Example (with a trick): $$(f * g)[x, y] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m, n] \cdot g[x-m, y-n]$$

- Take kernel g, flip it in both dimensions

1	2	1
0	0	0
-1	-2	-1

Kernel g

- Apply it on the input (element by element multiplication and addition)

1	2	3
4	5	6
7	8	9

Input f

$*$

-1	-2	-1
0	0	0
1	2	1

Kernel g

$=$?

The convolution operation 2D

- Example (with a trick): $(f * g)[x,y] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m,n] \cdot g[x-m, y-n]$

- Take kernel g, flip it in both dimensions

1	2	1
0	0	0
-1	-2	-1

Kernel g

- Apply it on the input (element by element multiplication and addition)

Input f

1	2	3
4	5	6
7	8	9

*

Kernel g

-1	-2	-1
0	0	0
1	2	1

=

-1	-4	-8	-8	-3
-4	-13	-20	-17	-6
-6	-18	-24	-18	-6
4	13	20	17	6
7	22	32	26	9

The convolution operation 2D

- Example (with a trick): $(f * g)[x,y] = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f[m,n] \cdot g[x-m, y-n]$

- Take kernel g, flip it in both dimensions

1	2	1
0	0	0
-1	-2	-1

Kernel g

- Apply it on the input (element by element multiplication and addition)

Input f

1	2	3
4	5	6
7	8	9

*

Kernel g

-1	-2	-1
0	0	0
1	2	1

=

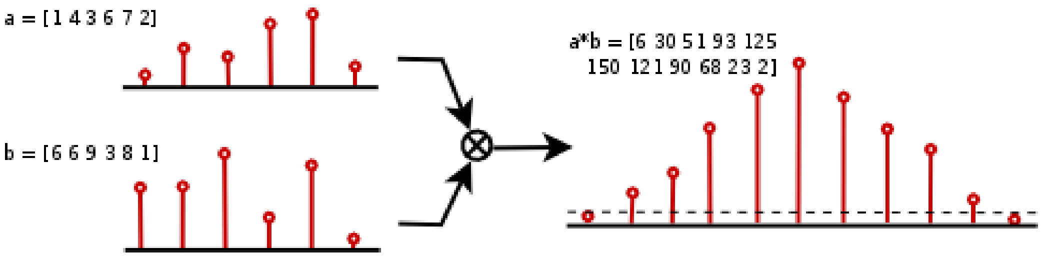
-1	-4	-8	-8	-3
-4	-13	-20	-17	-6
-6	-18	-24	-18	-6
4	13	20	17	6
7	22	32	26	9

(Image filtering)

Filtering

- Filtering: a signal (or filter, or kernel) g is used to modify another signal f

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$



- Some examples from image filtering

- Filtering:

Original image

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

What is the result of this filter?

Original image

Filtering

- Filtering:

Original image

$$\frac{1}{5}\cdot\begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

What is the result of this filter?

Blurring

Filtering

- Filtering:

$$\frac{1}{13}\cdot\begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Original image

What is the result of this filter?

(More) Blurring

Filtering

- Filtering:

$$\frac{1}{9}\cdot\begin{pmatrix}1&0&0&0&0&0&0&0&0\\0&1&0&0&0&0&0&0&0\\0&0&1&0&0&0&0&0&0\\0&0&0&1&0&0&0&0&0\\0&0&0&0&1&0&0&0&0\\0&0&0&0&0&1&0&0&0\\0&0&0&0&0&0&1&0&0\\0&0&0&0&0&0&0&1&0\\0&0&0&0&0&0&0&0&1\end{pmatrix}$$

Original image

What is the result of this filter?

Motion Blurring

Filtering

- Sobel operators or Sobel filters

$$Gx = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \qquad Gy = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \qquad G = \sqrt{Gx^2 + Gy^2}$$

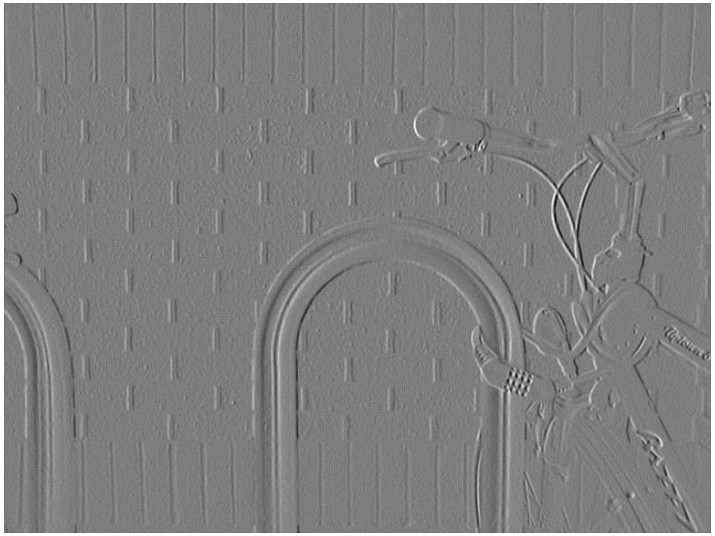
$\xrightarrow{G}$

Filtering

- Sobel operators or Sobel filters

$$Gx = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \qquad Gy = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \qquad G = \sqrt{Gx^2 + Gy^2}$$

Gx

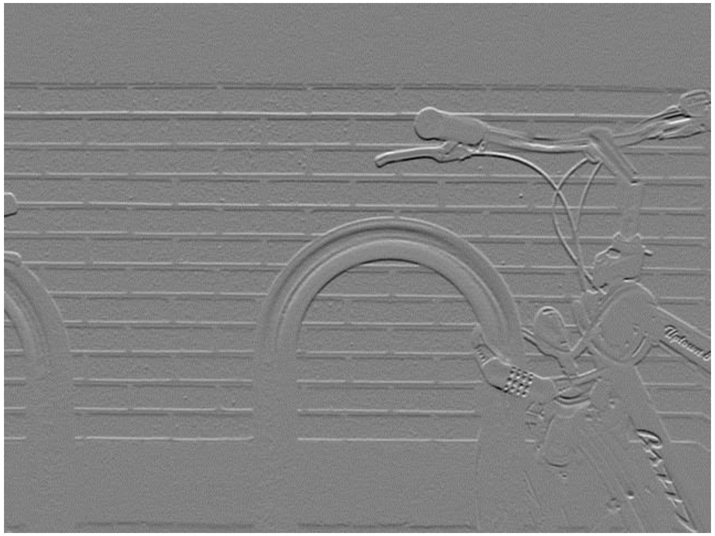


Filtering

- Sobel operators or Sobel filters

$$Gx = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \qquad Gy = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \qquad G = \sqrt{Gx^2 + Gy^2}$$

Gy



Convolution based photon dose calculation

- Treatment planning of external beam radiation therapy changed drastically in the last two decades of the 20^{th} century
  - Computed tomography (CT) scanners began being actively used in clinics world wide (3D anatomy)
  - Computer manufacturers begun providing solutions that were attractive to the clinical environment (increased computational power)
- Physicists realized that the combination of CT and computer technology there could improve the accuracy of the dose calculations in clinical practice
- In the last 30 years several dose calculation algorithms were proposed in order to achieve the goal of “accurate” treatment planning

Convolution based photon dose calculation

- The 90’s and early 00’s was the Pencil Beam (PB) algorithm was developed
- Of course there is also Monte Carlo. Highly accurate but too slow to be used
- The idea of the convolution methods:
  - Instead of doing a full Monte Carlo dose calculation
  - Precalculate via Monte Carlo the relevant physical information common to all particular cases
- The problem of dose calculation uses two types of info:
  - “Universal” physical information (dose deposition patterns)
  - Specific physical information (fluences, materials, geometries)

Convolution based photon dose calculation

- Treatment planning of external beam radiation therapy changed drastically in the last two decades of the 20th century
  - Computed tomography (CT) scanners began being actively used in clinics world wide (3D anatomy)
  - Computer manufacturers begun providing solutions that were attractive to the clinical environment (increased computational power)
- Physicists realized that the combination of CT and computer technology there could improve the accuracy of the dose calculations in clinical practice
- In the last 30 years several dose calculation algorithms were proposed in order to achieve the goal of “accurate” treatment planning

Convolution based photon dose calculation

- In the 80's the center of development was what is now called the Convolution/superposition method

- The 90's and early 00's was the time of the Pencil Beam Algorithm

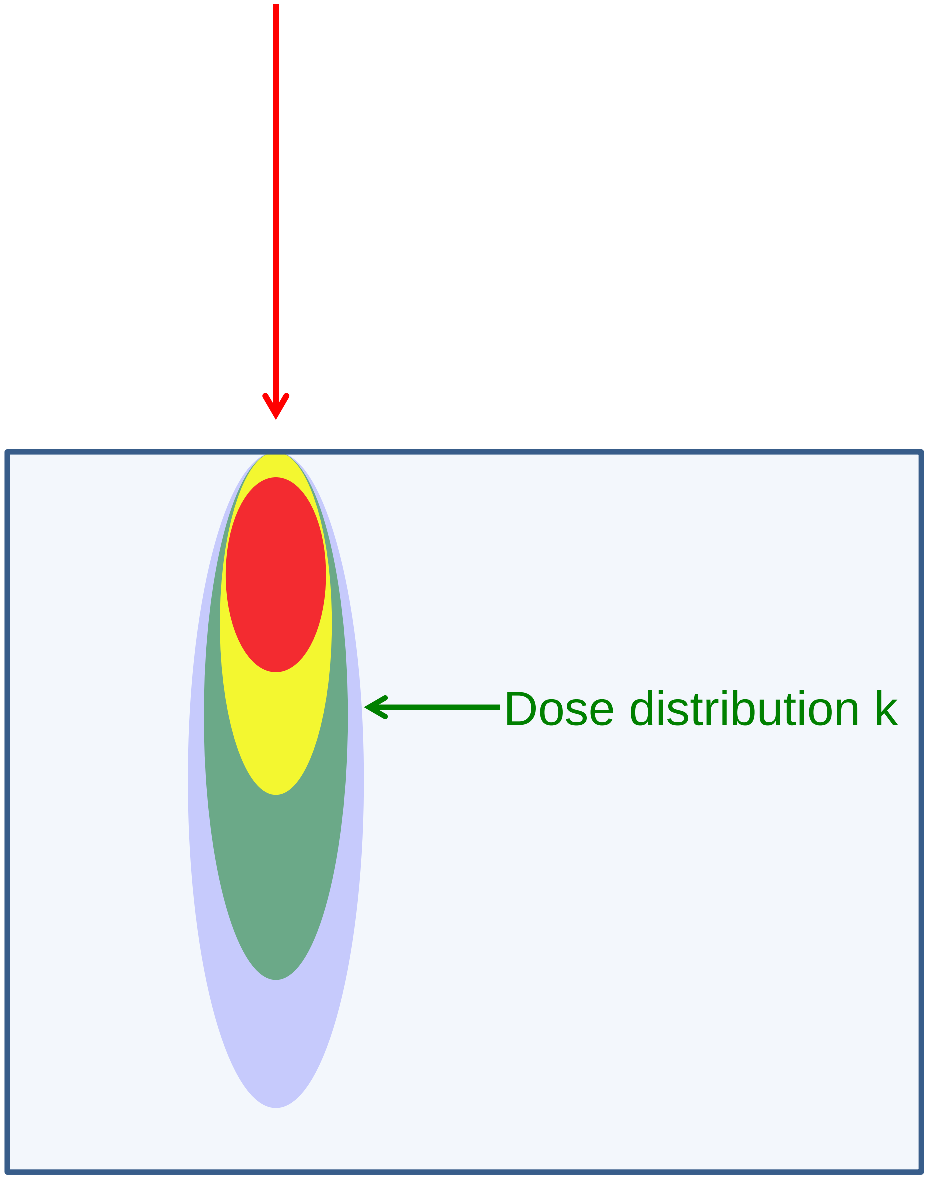
- Of course there is also Monte Carlo. Highly accurate but too slow to be used

- The idea of the convolution methods:
  - Instead of doing a full Monte Carlo dose calculation
  - Precalculate via Monte Carlo the relevant physical information common to all particular cases

- The problem of dose calculation uses two types of info:
  - "Universal" physical information (dose deposition patterns)
  - Specific physical information (fluences, materials, geometries)

The Pencil Beam Algorithm (PB)

- Consider the case of an infinitesimal pencil beam on a water target



The Pencil Beam Algorithm (PB)

- A bit more realistic:
  - An extended beam
  - The dose contribution in point P
  - If each PB has a different intensity/fluence

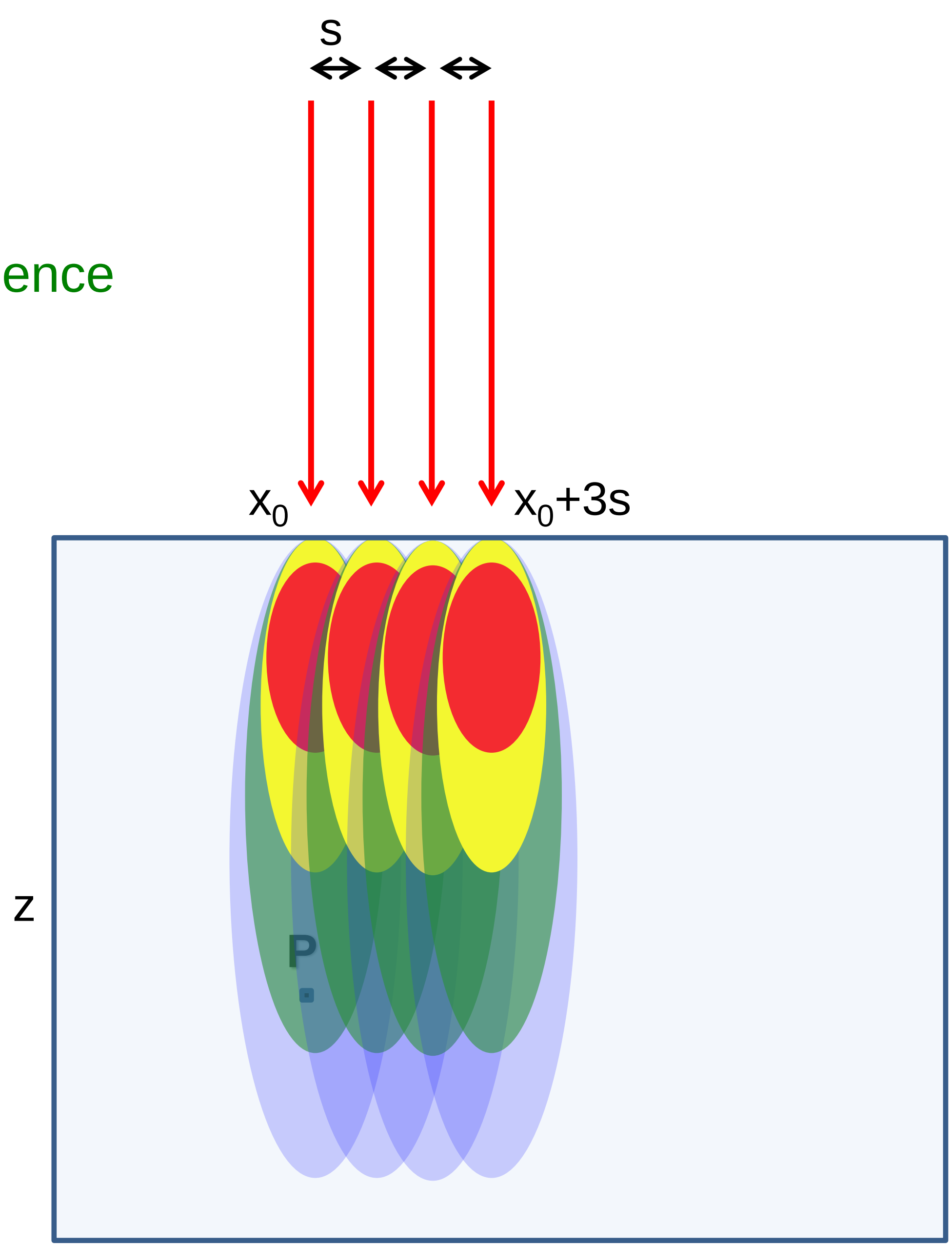
$$d(x,z) = \sum_{i=0}^{3} \phi(i) k(x - i \cdot s, z)$$

  - Then the dose in position P is:

$$\boxed{d(x,z) = \phi(x) * k(x,z)}$$

  - Similarly to:

$$h[n] = (f * g)[n] = \sum_{m=-\infty}^{\infty} f[m] \cdot g[n-m]$$

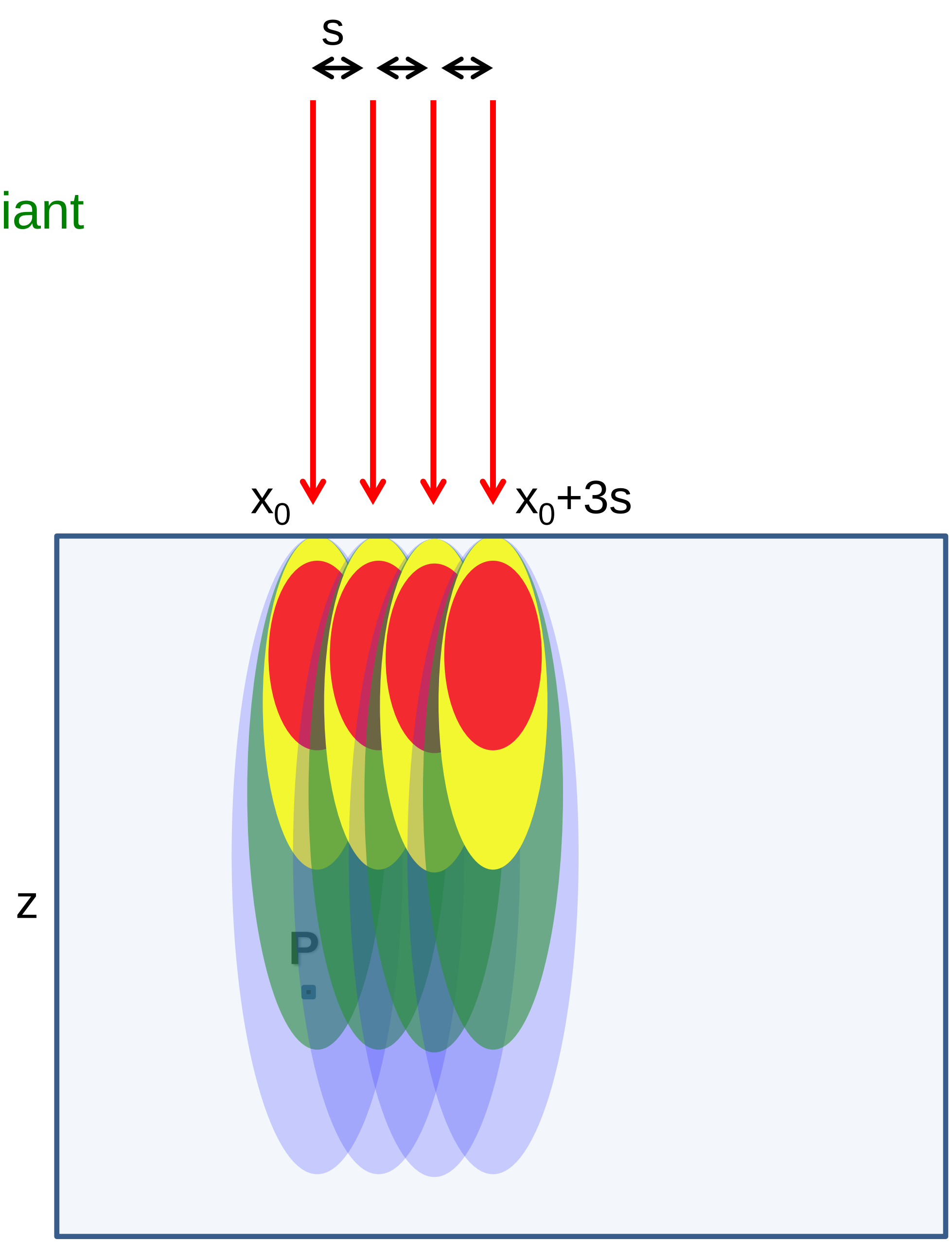


The Pencil Beam Algorithm (PB)

- It shouldn't be surprising that convolution has appeared:

  1. The pencil beam kernel is shift invariant

  2. The absorbed dose obeys linearity



- Convolution is a superposition operation with a constant shift invariant kernel

- The kernel is a constant
(for the same energy the dose kernel remains the same)

- The intensity/fluence of each PB is the specific information
(could be different in every case)

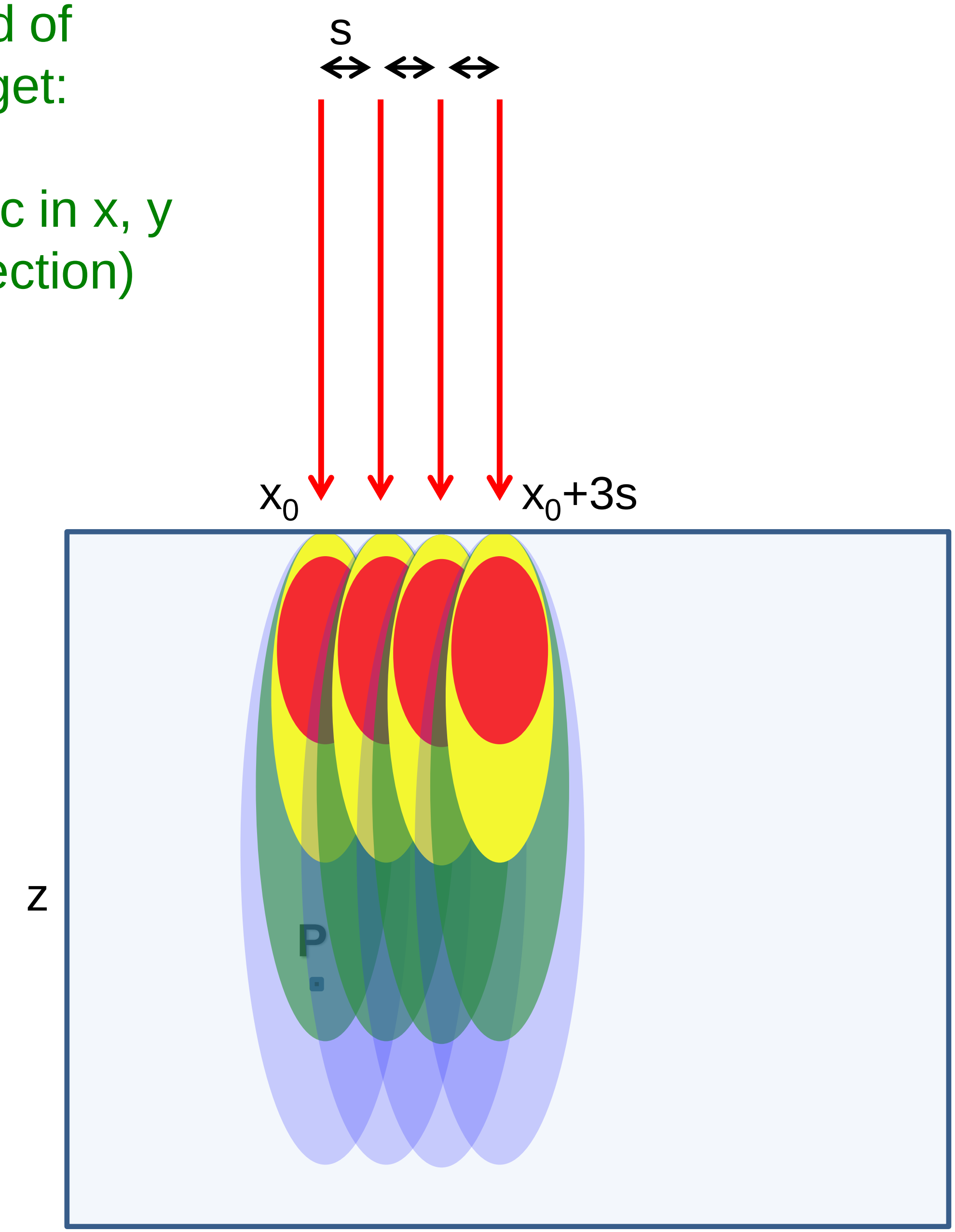
The Pencil Beam Algorithm (PB)

- For an extended photon beam composed of infinitesimal PBs on a homogeneous target:
  - Assuming that the PBs are symmetric in x, y (plane perpendicular to the beam direction)

$$\boxed{d(x,y,z) = \boxed{\phi(x,y)} * \boxed{k(x,y,z)}}$$

Fluence map PB kernel

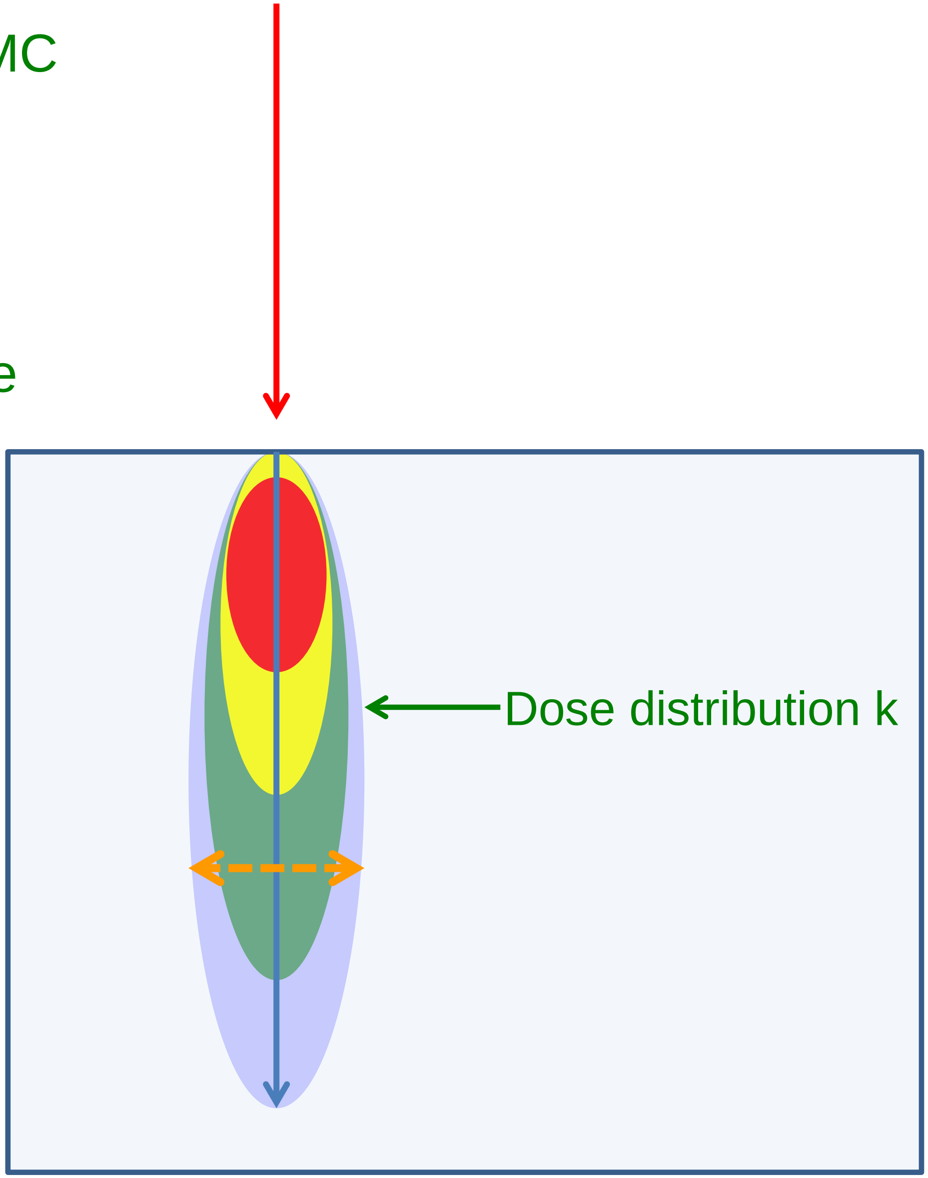
$$\boxed{d(r,z) = \boxed{\phi(r)} * \boxed{k(r,z)}}$$



The Pencil Beam Algorithm (PB)

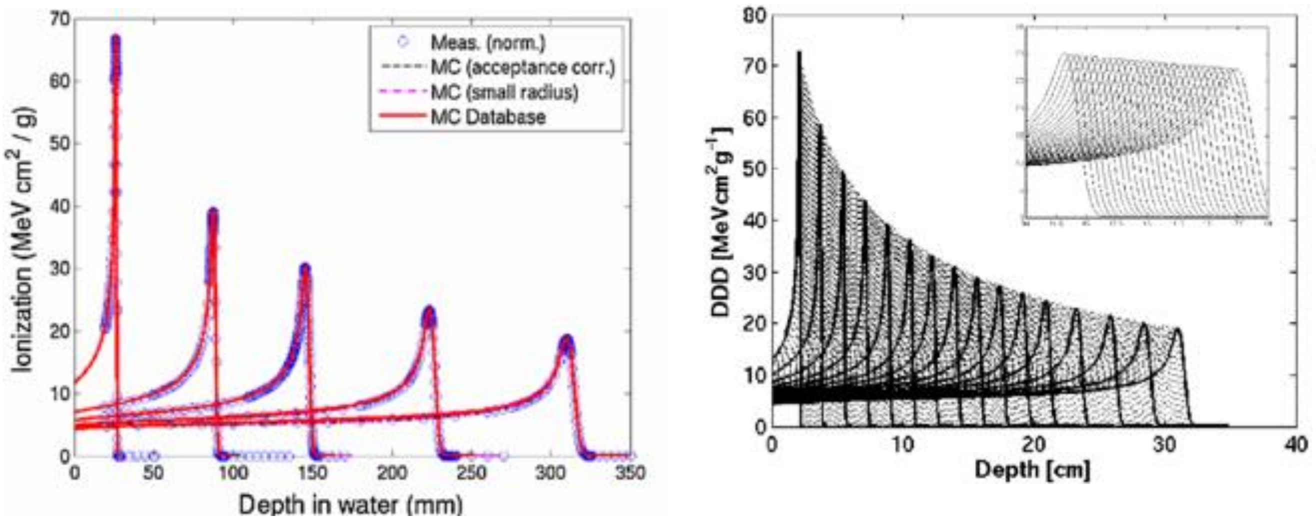
- Description of the dose kernel k:
  - The dose kernel is precalculated with MC and consists of dose in water from PBs of different E (basic input data to the TPS)
  - Based on those data, the TPS splits the dose kernel into two terms: a central-axis term C and an off-axis term O, so that:

$$k(x,y,z) = \boxed{C(z)} \cdot \boxed{O(x,y,z)}$$



The Pencil Beam Algorithm (PB)

- The central-axis term is taken from the laterally integrated PB input data



Parodi et al, Phys Med Biol, Volume 57, Number 12

- Scaling of the Bragg peaks as calculated into water for the cases of different materials is based on water equivalent path length

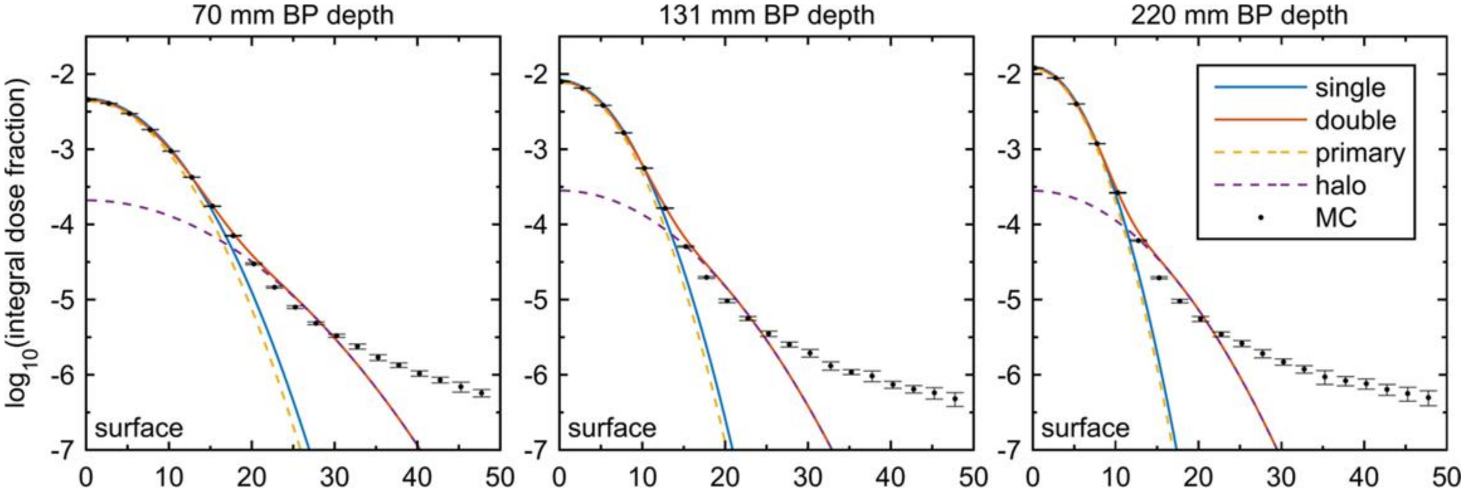
The Pencil Beam Algorithm (PB)

- The off-axis term was initially parameterized as a Gaussian shape of the lateral profile as a function of depth, for each PB

$$O(x,y,z)=\frac{1}{2\pi\left[\sigma_{tot}(z)\right]^2}\exp\left(-\frac{x^2+y^2}{2\left[\sigma_{tot}(z)\right]^2}\right)$$

  - Where σ_{tot} is the quadratic sum of all contributions broadening the beam

- The single Gaussian parameterization is not adequate. Therefore, double Gaussian is used:



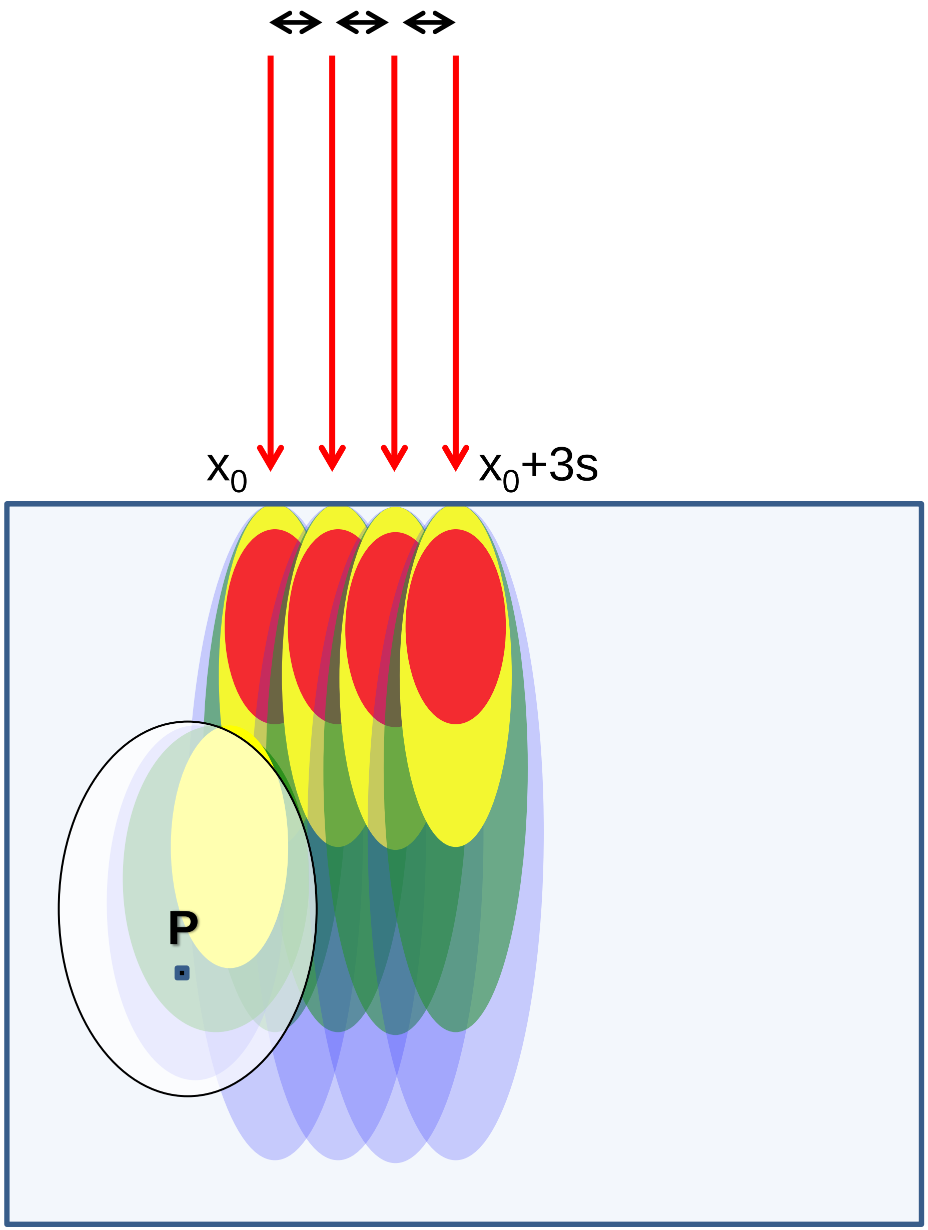
The Pencil Beam Algorithm (PB)

- A patient is clearly not a water tank
- What happens with heterogeneites?

$$\boxed{d(x,y,z) = \boxed{\phi(x,y)} * \boxed{k(x,y,z)}}$$

Fluence map　　PB kernel

$$\boxed{d(r,z) = \boxed{\phi(r)} * \boxed{k(r,z)}}$$



- The kernel is not shift invariant anymore (takes a position dependent shape)

The Pencil Beam Algorithm (PB)

- What happens in presence of heterogeneities?:
- Tabulate the dose kernel k(r,z) in water (as for the homogeneous case)
- using CT data, apply a correction factor $A(z_{rad})$ (1D)

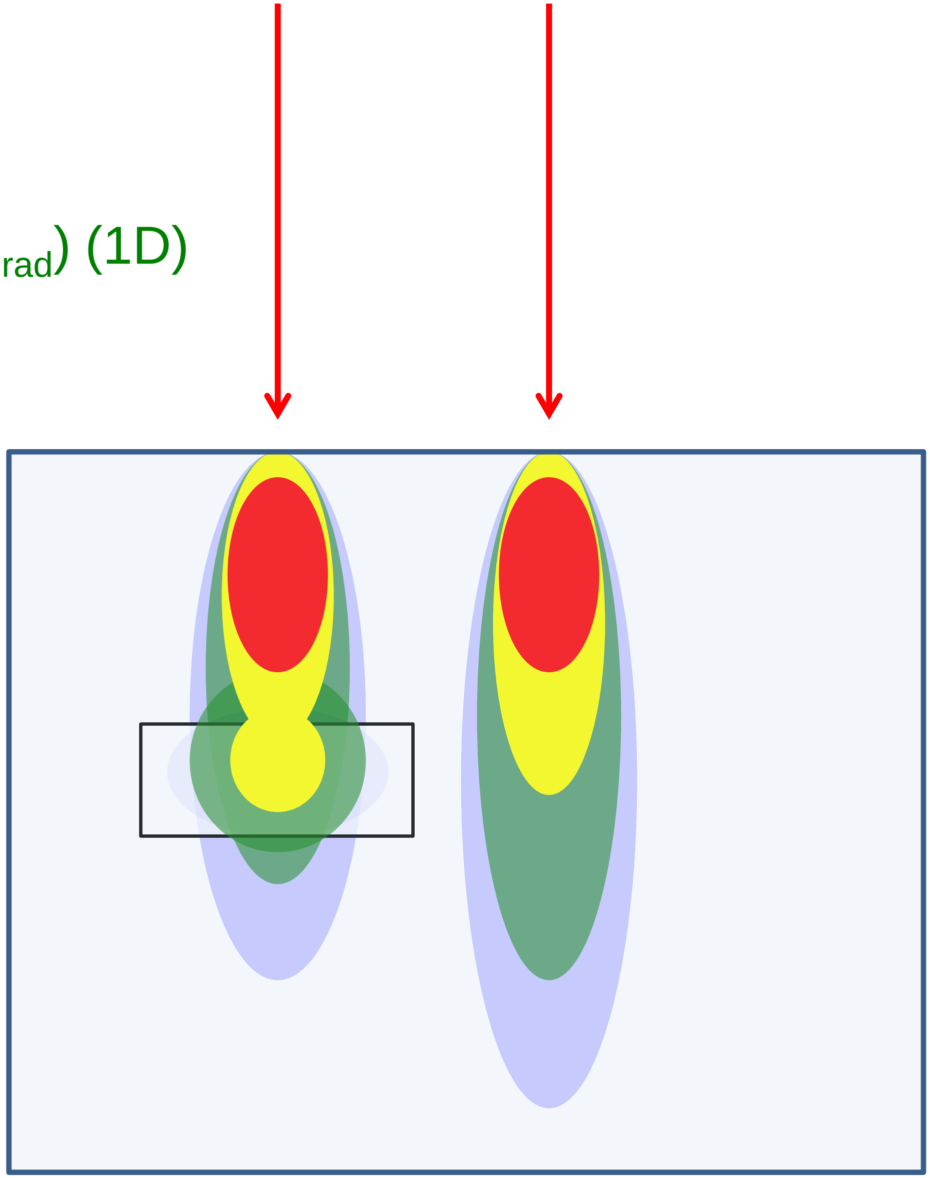
$$k'(x,y,z) = A(z_{rad})k(x,y,z)$$

- or $A(x,y,z_{rad})$ (3D)

$$k'(x,y,z) = A(x,y,z_{rad})k(x,y,z)$$

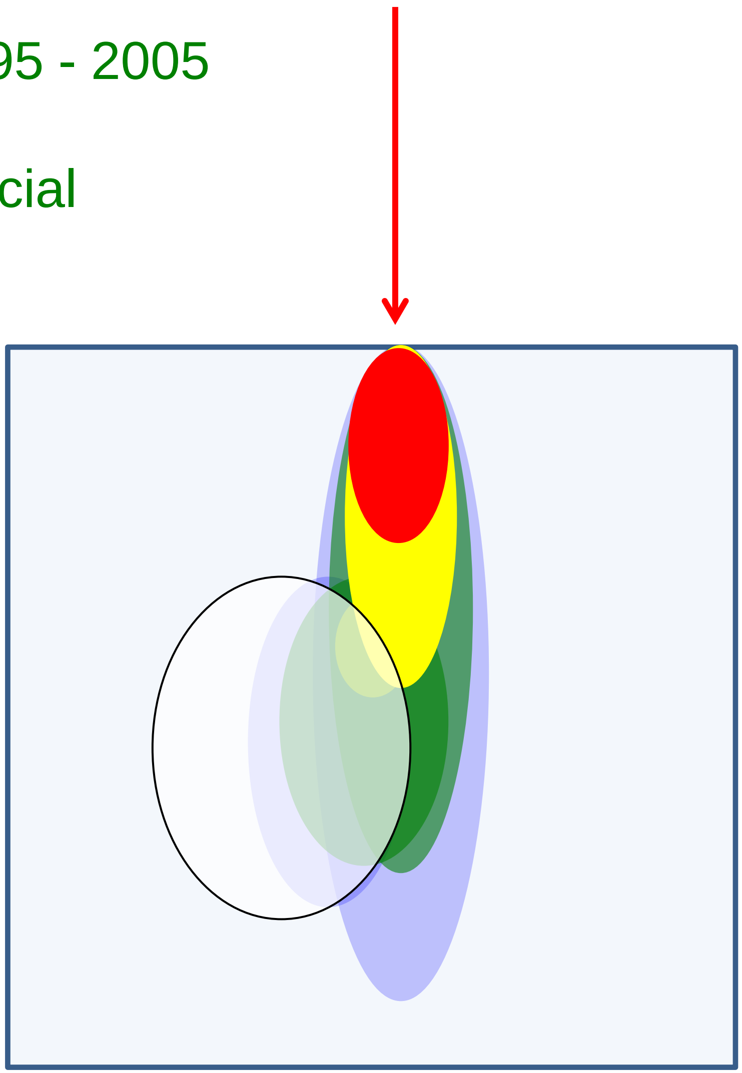
- This function is not shift invariant:
- Convolution does not describe the problem
- It becomes a more general superposition

$$d(x,y,z) = \sum_{i=0}^{N} \phi_i k_i'(x,y,z)$$



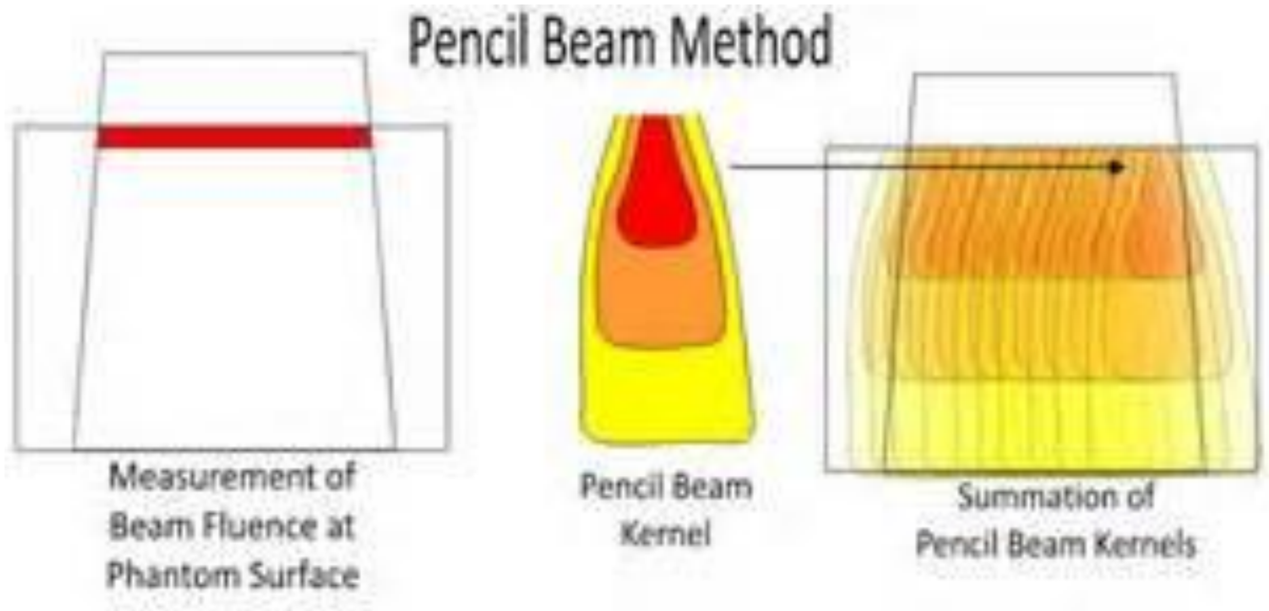
The Pencil Beam Algorithm (PB)

- The golden years of the pencil beam algorithm: 1995 - 2005
- The PB algorithm for photons appeared in commercial implementations of several manufacturers
- The main reason of the popularity of PB algorithms was the need for a fast dose calculation for IMRT
- Nevertheless, the algorithm fails to give a good description in the presence of strong heterogeneities
- Especially 1D scaling extremely fast but not accurate enough

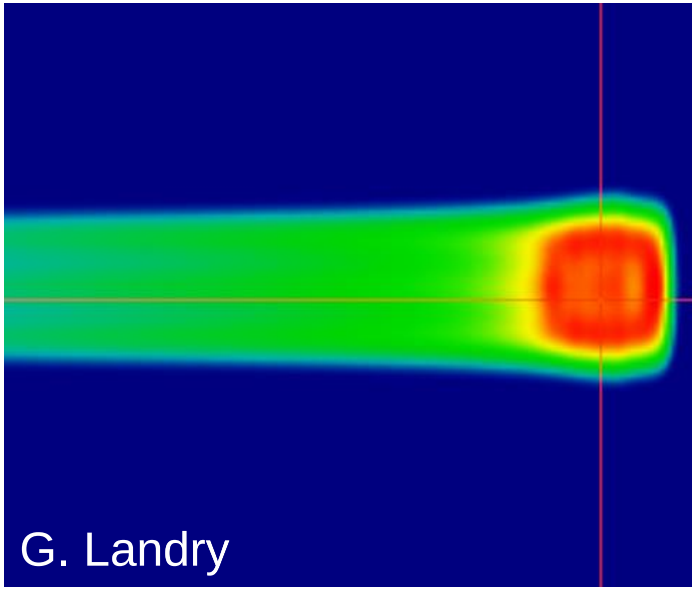


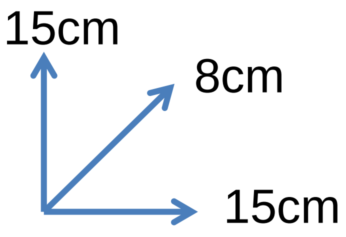
The Pencil Beam Algorithm (PB)

- A summarizing sketch of the PB algorithm:
-



- Programming Exercise 5:
  - Calculate the dose distribution using the PB algorithm according to the basic data (dose kernel) available on the lecture’s website
  - Assume homogeneous water geometry (15cmx15cmx8cm – voxel size 1mmx1mmx1mm) and a single Gaussian lateral spread parameterization
  - For PB energies in between the nominal ones, use linear interpolation for both the laterally integrated dose and the lateral spread σ
  - The dose should look like:





Pencil Beam Algorithm (PB): Programming Exercise

- Programming Exercise 5:
  - Report on the calculation time and plot a longitudinal and a lateral dose profile