Outline

- Introduction, principles
- Method n°1 : Demons
- Evaluation
- **Method n°2 : B-Splines**
- Method n°3 : Deep-learning registration
- Method n°4 : TPS (Thin Plate Spline)
- The « sliding » problem
- Spatio-temporal deformable registration
- Conclusion

Outline

- Introduction, principles
- Method n°1 : Demons
- Evaluation
- **Method n°2 : B-Splines**
- Method n°3 : Deep-learning registration
- Method n°4 : TPS (Thin Plate Spline)
- The « sliding » problem
- Spatio-temporal deformable registration
- Conclusion

Example 2 : « B-Splines »

Method proposed by **[Rueckert1999] [Kybic2003]**

Daniel Rueckert, LI Sonoda, C Hayes, DLG Hill, and MO Leach. Nonrigid registration using free-form deformations: application to breast MR images.
IEEE Transactions on Medical Imaging, 18(8):712-721, 1999

- Popular
- Parametric approach
- Very flexible, “adaptive”
- Numerous developments

Generic model: optimisation

$$T_{opt} = \arg_T \max \left[\alpha E_{sim}(A,B,T) + (1-\alpha) E_{reg}(T) \right]$$

A, B = the two images to register (reference & moving)

T_{opt} = the sought optimal transformation

E_{sim} = similarity measure

E_{reg} = regularization measure of T

α = tradeoff parameter

$\arg_T \max$ = optimization algorithm

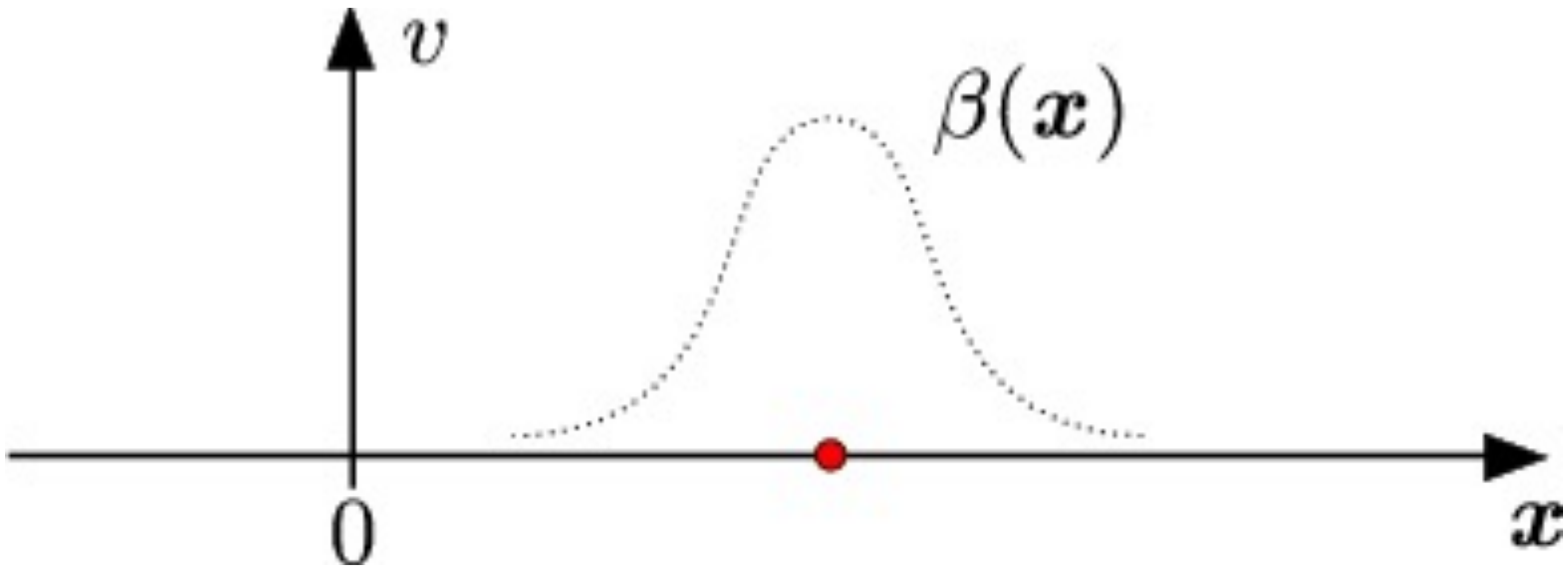
B-Splines ?

- Deformation field : a pixel by vector
- Alternative representation: parametric
- Deformation at a given point = computation
- Linear combination of piecewise polynomials
- Polynomials of degree r (often cubic r=3)

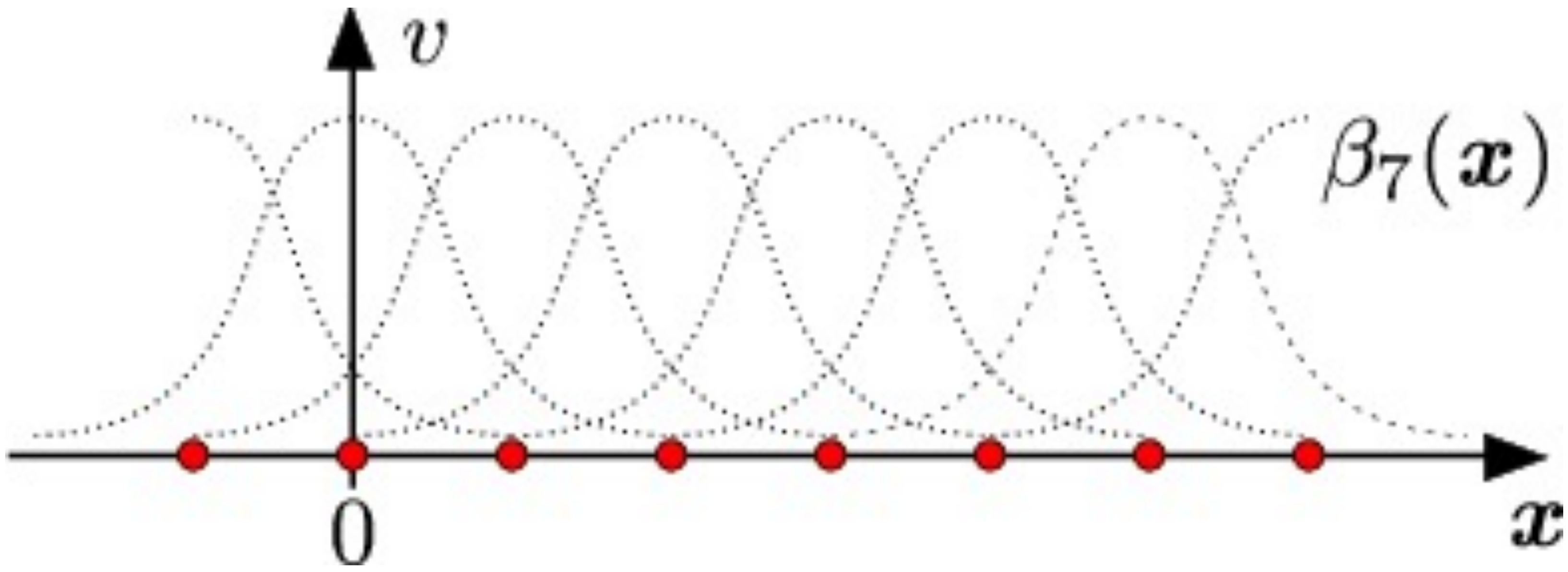
$$T(x) = x + \sum_i^{(r+1)^d} c_i \beta_i^r(x)$$

- Interesting properties
  - Compact support
  - Continue
  - Separable
  - Derivable (r-1) times

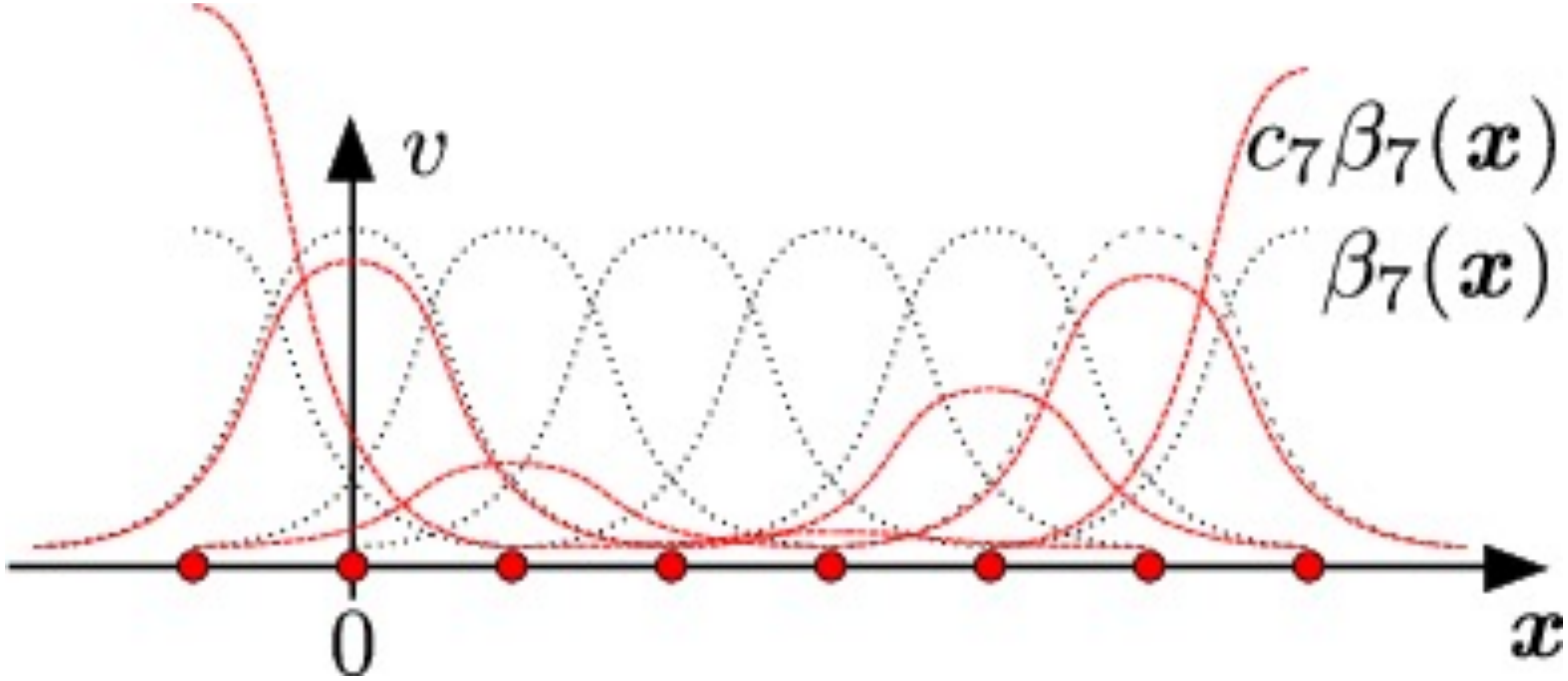
B-Splines



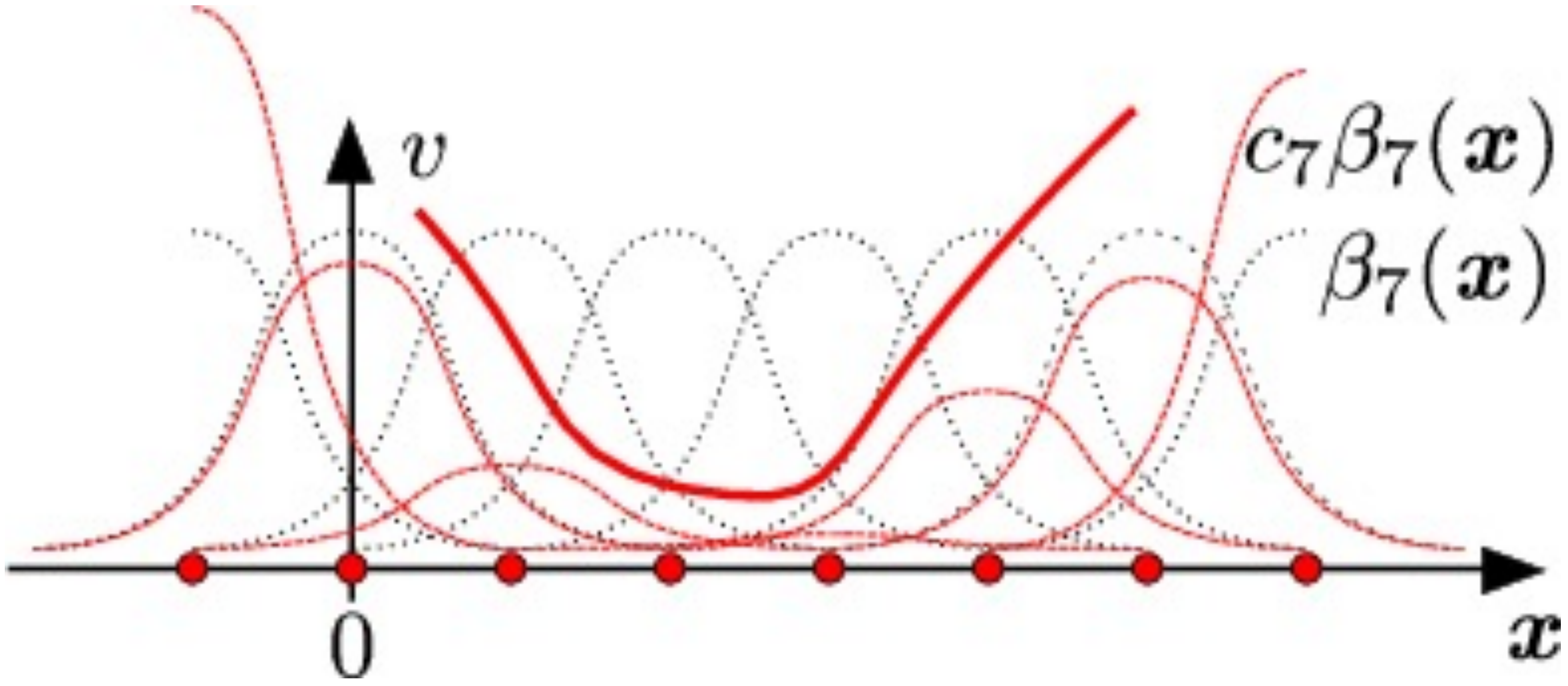
B-Splines



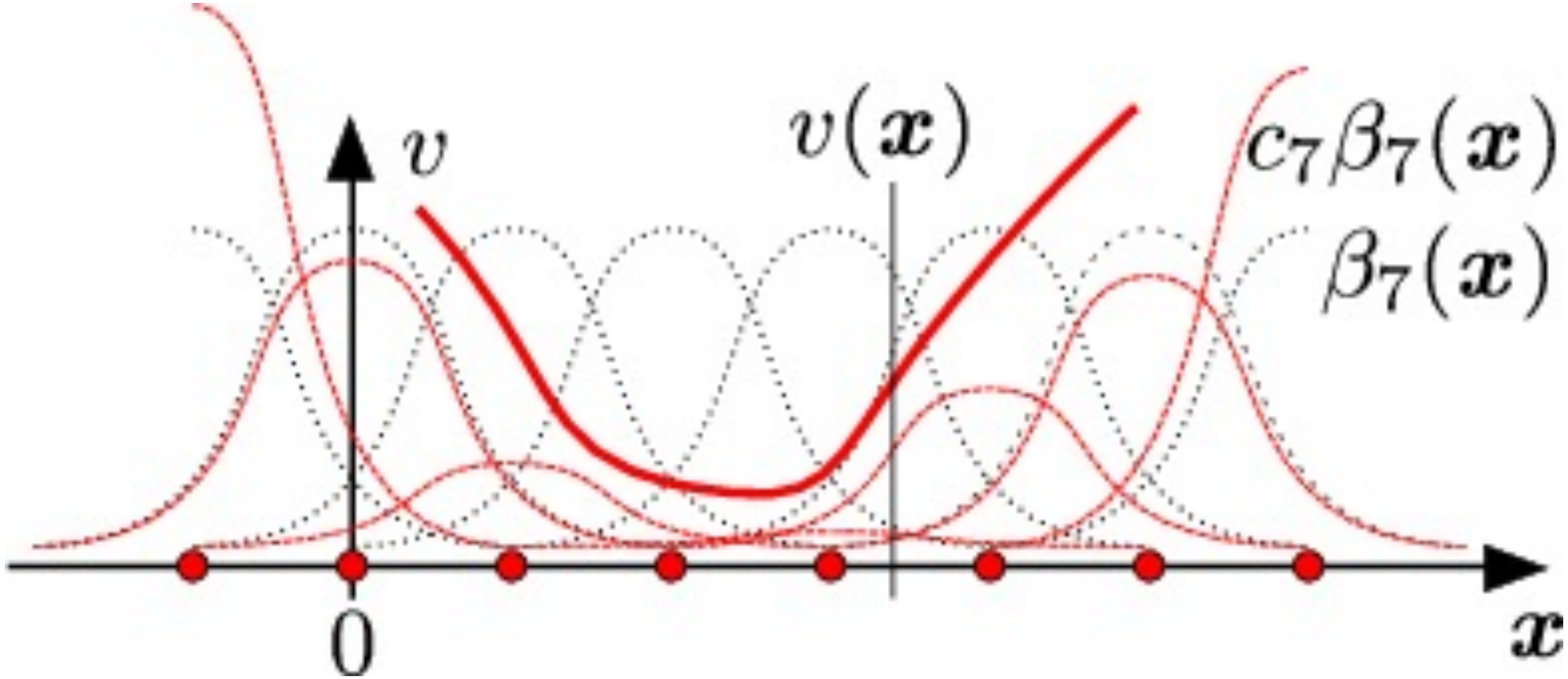
B-Splines



B-Splines

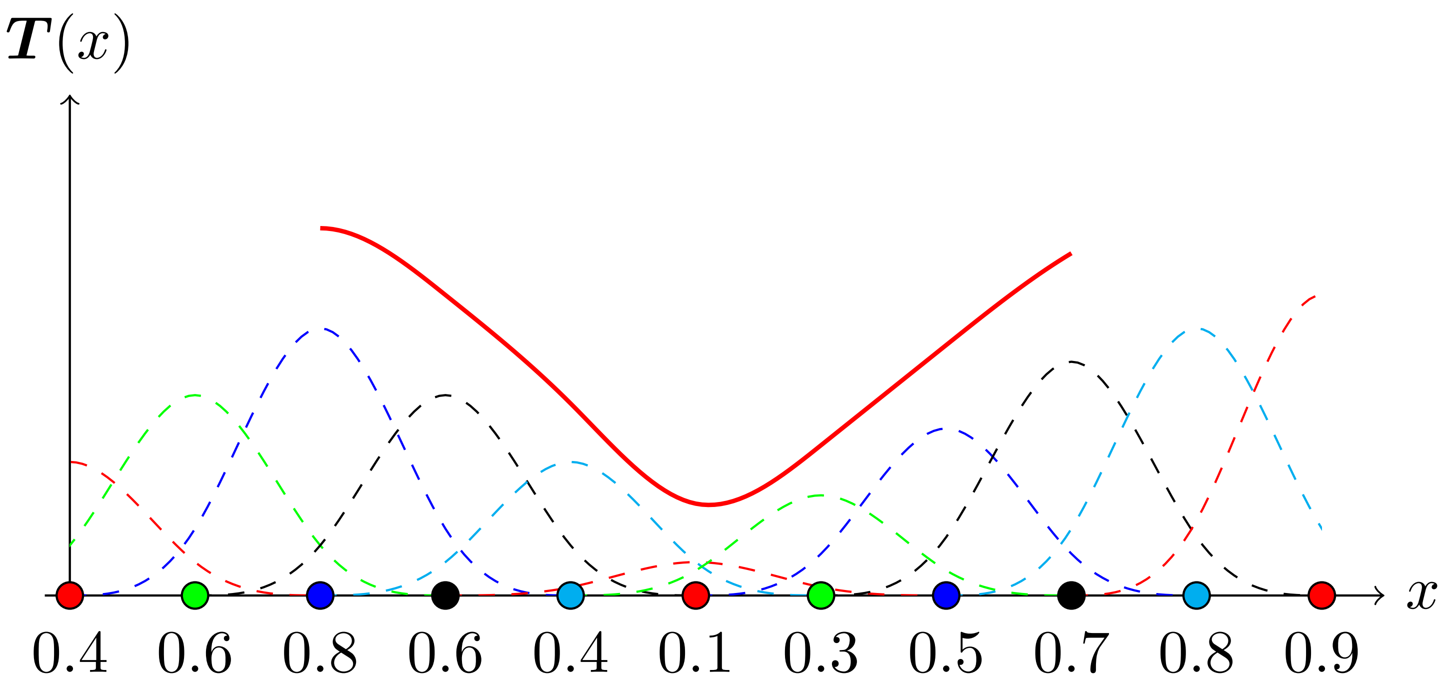


B-Splines



v
$v(\boldsymbol{x})$
$c_7\beta_7(\boldsymbol{x})$
$\beta_7(\boldsymbol{x})$
0
$\boldsymbol{x}$

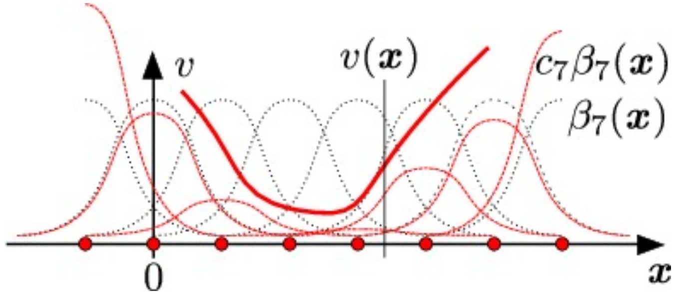
Parametrised curve



Cubic B-splines (1D)

$$\beta_i^r(x) = \prod_j^d \beta^r(p_{ij} - x_j)$$

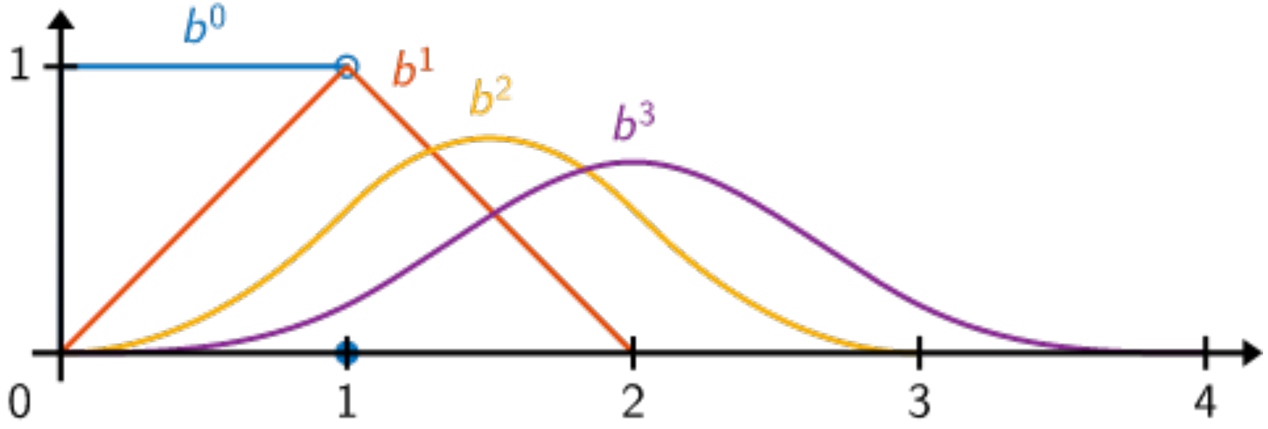
- p_i = coordinates of the control point I on the support (0,1,2,3)
- $u = p_i - x$



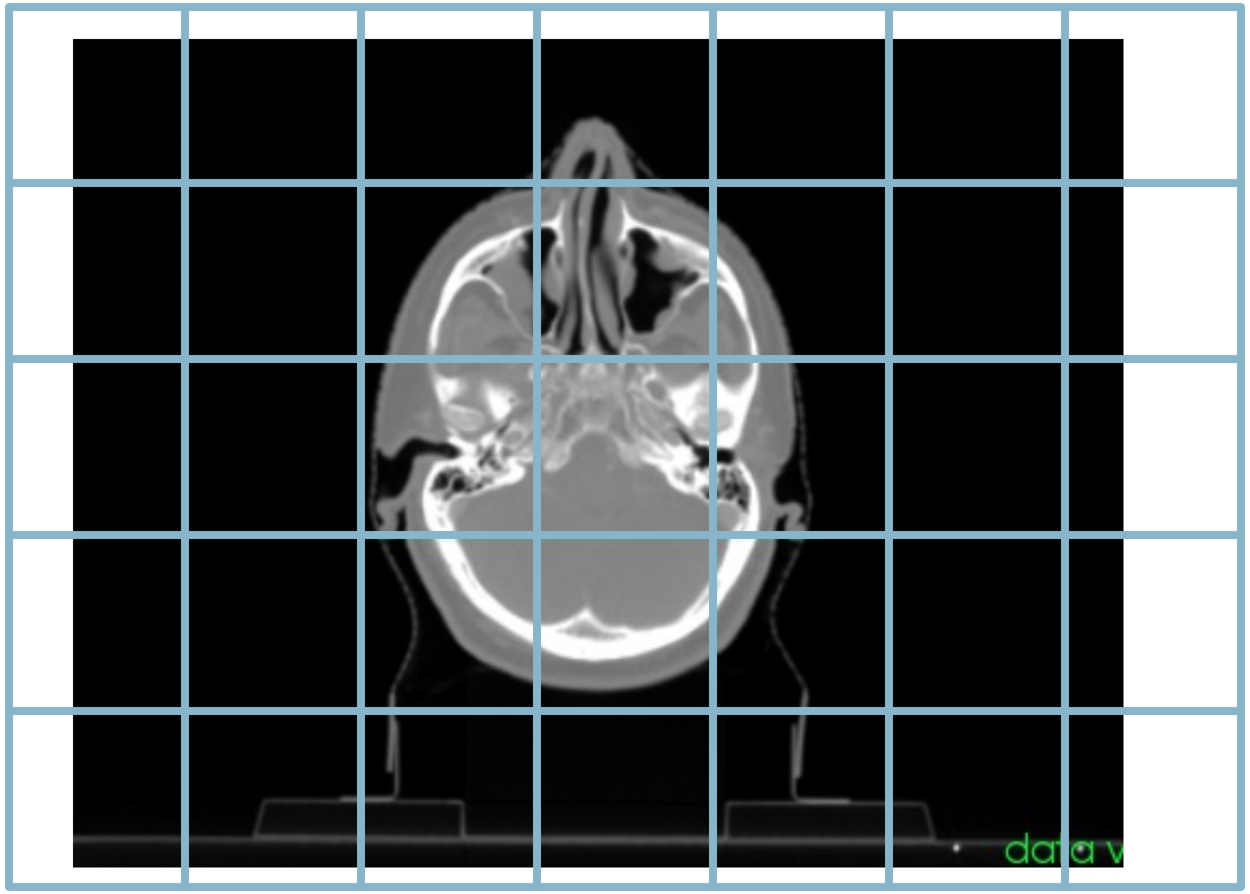
$$B_0(u) = (1-u)^3/6$$
$$B_1(u) = (3u^3 - 6u^2 + 4)/6$$
$$B_2(u) = (-3u^3 + 3u^2 + 3u + 1)/6$$
$$B_3(u) = u^3/6.$$

Cubic B-splines

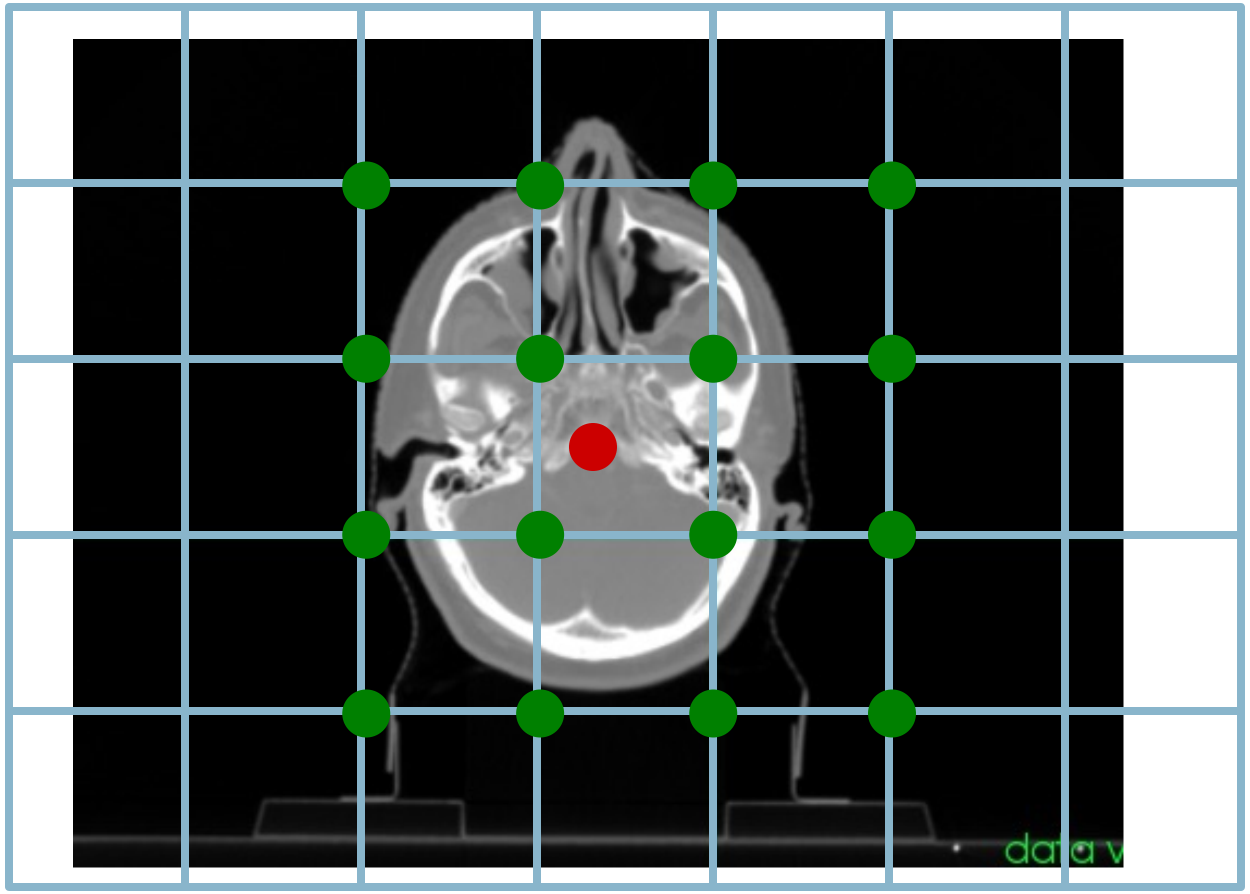
Cardinal Bsplines degrees



Grid of control points



Grid of control points



B-Splines

- Grid of uniformly spaced control points (e.g. 30 mm)
- Separable

$$T(x) = x + \sum_i^{(r+1)^d} c_i \beta_i^r(x)$$

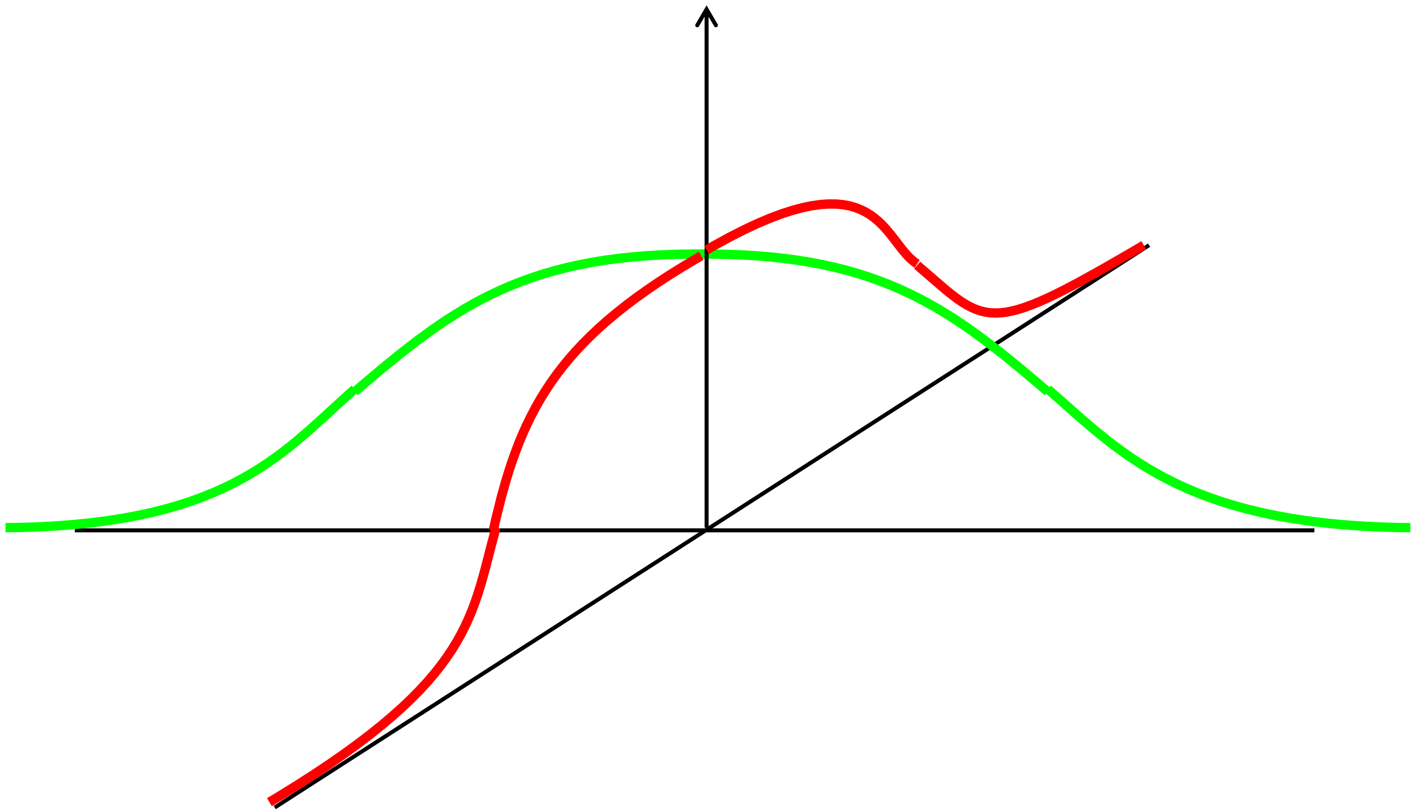
$$\beta_i^r(x) = \prod_j^d \beta^r(p_{ij} - x_j)$$

d	the image dimension
r	is the B-splines degree
x	3D point coordinate
x_j	is the jth coordinate of x
c_i	the B-splines coefficients (a vector)
i	the index of the control points with coordinate pij
$B_i^r(x)$	is the tensor product of B^r

N-Dimension BSplines

$$\beta_i^r(x) = \prod_j^d \beta^r(p_{ij} - x_j)$$

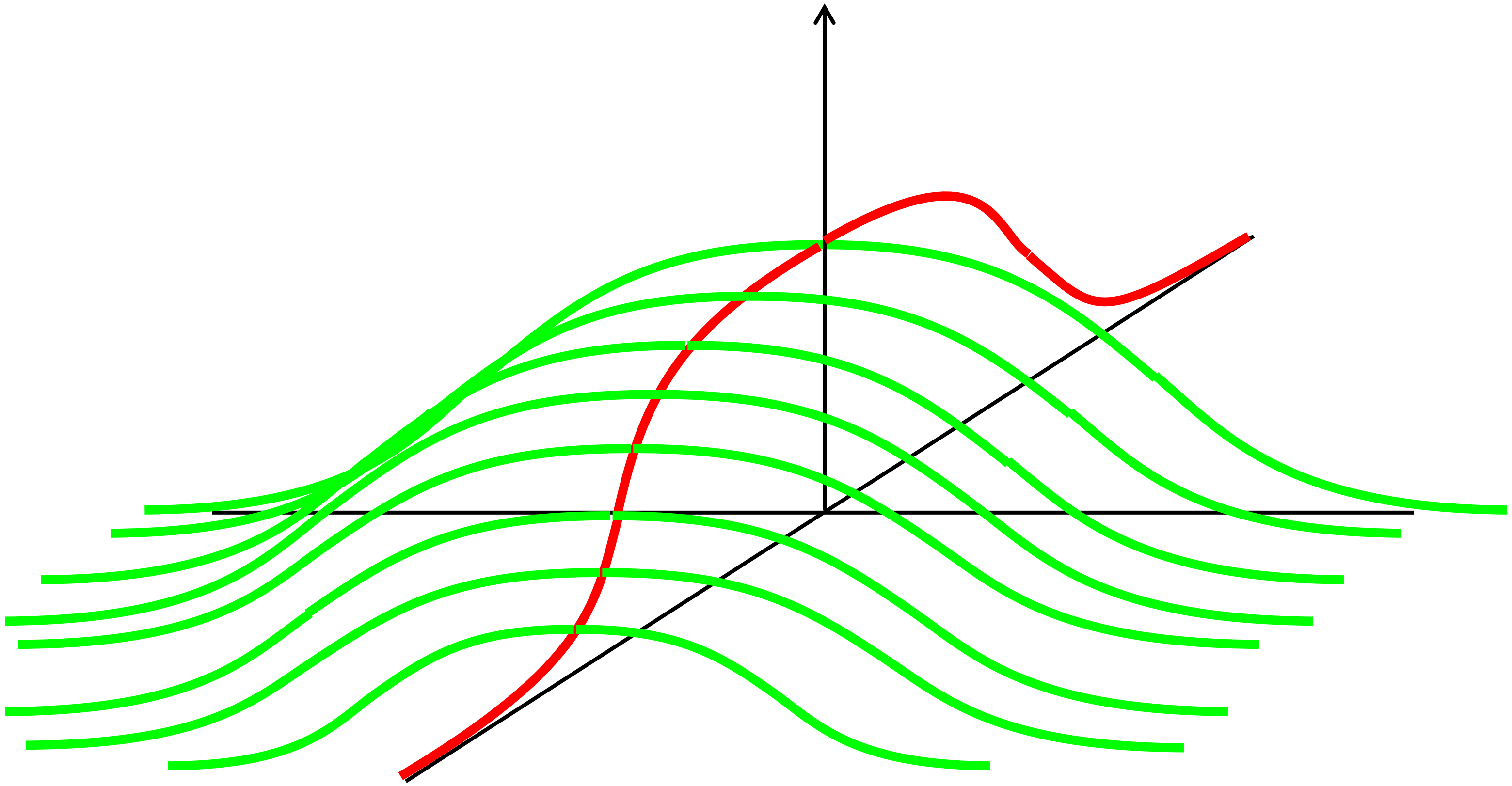
- In 3D : 3D B-Splines.
- Obtained from tensor product of 1D B-Splines



N-Dimension BSplines

$$\beta_i^r(x) = \prod_j^d \beta^r(p_{ij} - x_j)$$

- In 3D : 3D B-Splines.
- Obtained from tensor product of 1D B-Splines



Recursive Cox-de-Boor formula

$$\beta^r(e) = \boldsymbol{u}\boldsymbol{M} \text{ with } \begin{cases} \boldsymbol{u} = [e^r\, e^{r-1}\ \ldots\ e\, 1] \\ \boldsymbol{M} \text{ matrix of size} k \times k \text{ with } k = r+1 \end{cases}$$

$$\boldsymbol{M} = [\mathrm{M}_{ij}] = \left[\frac{1}{(k-1)!} C_{k-1,i} \sum_{m=j}^{k-1} (k-(m+1))^i\, (-1)^{m-j}\, C_{k,m-j} \right]$$

$$C_{i,j} = \frac{i!}{j!(i-j)!} = \text{binomial coefficient}$$

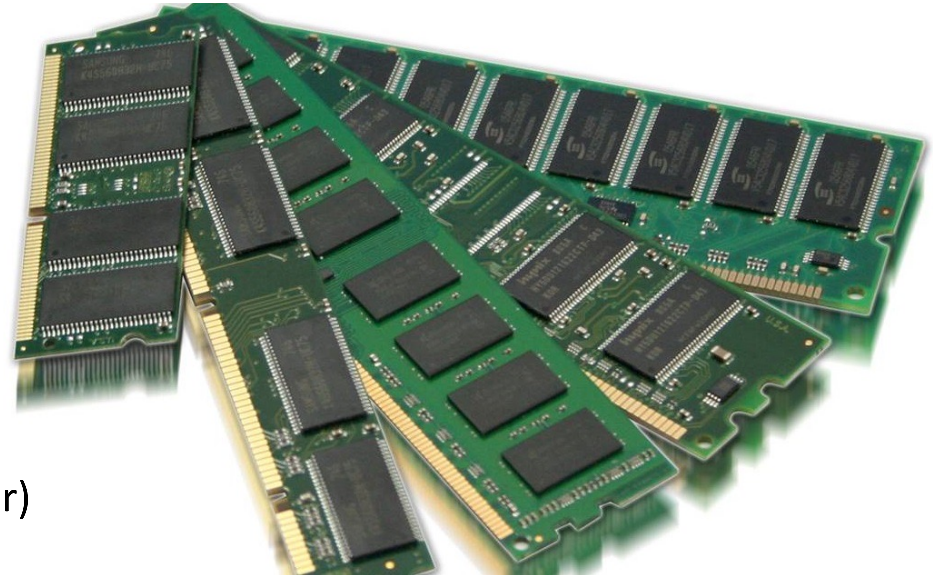
The B-Splines function of order n is obtained by n times convolution of zero-th order B-spline function (is 1 between [-0.5;+0.5], 0 elsewhere)

B-Splines

- Grid of control points
- A each CP : d coefficients

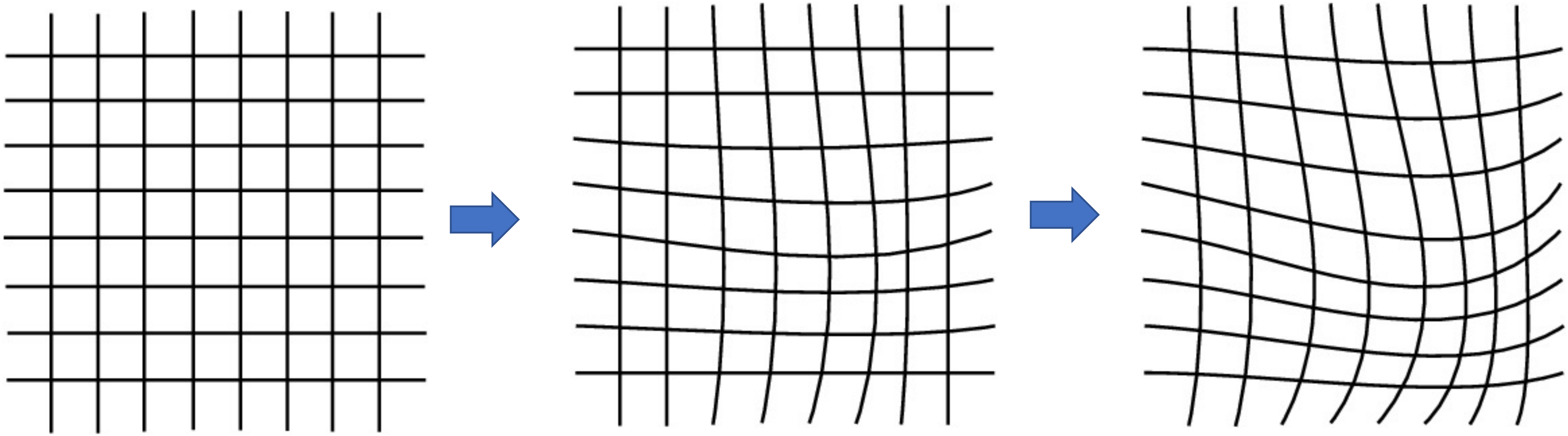
- Example
  - Image
    - Spacing 1 x 1 x 2 mm
    - Size 512 x 512 x 100 = 26 M pixels
    - Memory (Float pixels) 104 Mb (or 312Mb if 3D vector)

  - Grid of control points
    - Spacing 30 x 30 x 30 mm
    - Size X,Y : 512/30+4 = 22 (sup)
Z : 100/30+4 = 8
22 x 22 x 8 = 3872
    - Memory 47 Kb



B-Splines FFD

- Other term : FFD Free Form Deformation



Generic model: optimisation

$$T_{opt} = \arg_T \max \left[\alpha E_{sim}(A, B, T) + (1-\alpha) E_{reg}(T) \right]$$

A, B = the two images to register (reference & moving)

T_{opt} = the sought optimal transformation

E_{sim} = similarity measure

E_{reg} = regularization measure of T

α = tradeoff parameter

$\arg_T \max$ = optimization algorithm

B-splines parameter derivation

- Optimisation, gradient-based :
  - update parameters (c_i) at each iteration
  - partial derivatives

- Derivative of the cost function according to the parameters c_i
- Cost function : (weighted) sum of similarity measure and smoothness function.
  - Similarity measure: SSD, Mutual Information, etc
  - Smoothness function: based on transformation derivative, etc

- Mathematically derived or numerically estimated
  (Taylor expansion = fct represented as a sum of its derivatives)

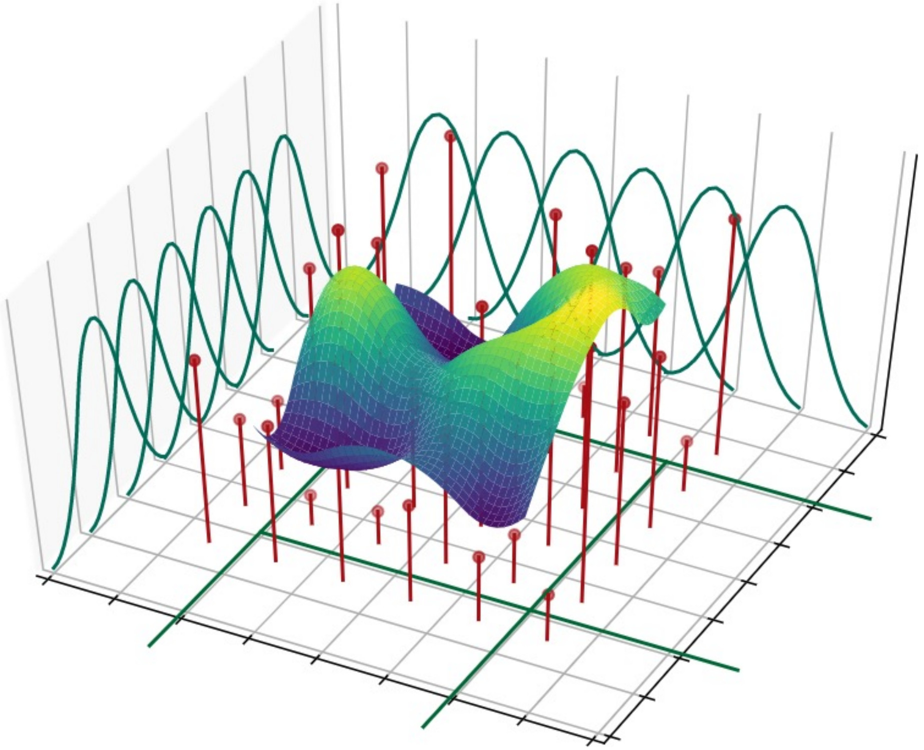
Stochastic Sampling

- **Adaptive stochastic gradient descent** [Klein et al 2009]

- Gradient approximated by stochastic subsampling.
  Voxels are randomly selected at every iteration.
- Adaptive descent step size :
  - Determined with a decaying function of the iteration number k
  - Based on the inner product of the current and previous gradient. Intuitively, if the gradients in two consecutive iterations point in (almost) the same direction, it is expected that larger steps can be taken.

- Results : fast convergence using much less voxels at each iterations.

B-Splines



- B-Splines : store the **deformation**
- To be included in the optimisation
  - Similarity measure (e.g. Mutual Information)
  - Parameters to optimise : the “c_i” coefficients

- Advantages
  - Customizable (grid of control, degree)
  - Compact (less parameters to optimise)
  - Intrinsic regularization
    (but additional regularization could be used)
  - Can be used with several similarity measures, with several optimiser
  - Developments: cyclicality, etc …

- Drawbacks
  - Is the regularization physically plausible ?
  - Computation time could be long