METRO
MEtallurgical TRaining On-line

Adaptive numerical methods

Arkadiusz Nagórka
CzUT

Education and Culture

Introduction

Common steps of finite element computations consists of

- *preprocessing* - definition of geometry, boundary conditions, initial conditions, material properties, meshing
- *processing* - assembly and solution of a system of equation (possibly in a time and/or non-linear loop)
- *postprocessing* - evaluation of derived quantities, visualisation of the results

Cost of computations

Some parts are more time-consuming than another

- time of creation of the geometric model and setting up the problem depends on its complexity and user's experience
- meshing of complicated models can take a long time
- assembly time is proportional to number of elements and polynomial degree
- imposition of boundary condition is as above
- **solution of linear systems** is usually the most time-consuming, $O(N^3)$ for LU, $O(N^2)$ for LU and banded matrices, $O(N^{3/2})$ for for conjugate gradients

In transient problems cost per step may depend on step size

Error in the results

Numerical methods give approximate solutions.

The error at a point is $e(\mathbf{x}) = u(\mathbf{x}) - u_h(\mathbf{x})$

Global measures over the entire domain Ω

- infinity norm ("maximal absolute value of the difference")

$$E = \|e\|_{L_\infty(\Omega)} = \operatorname*{ess\,sup}_{\mathbf{x}\in\Omega} |e(\mathbf{x})| \qquad \text{(essential supremum)}$$

- L1, L2 norms, etc.

$$E = \|e\|_{L_p(\Omega)} = \sqrt[p]{\int_\Omega |e(\mathbf{x})|^p \, d\mathbf{x}}$$

- problem-dependent energy norm, e.g. for the Poisson eq.

$$E = \|e\|_{E(\Omega)} = \sqrt{\int_\Omega k \nabla e(\mathbf{x}) \cdot \nabla e(\mathbf{x}) \, d\mathbf{x}}$$

Convergence

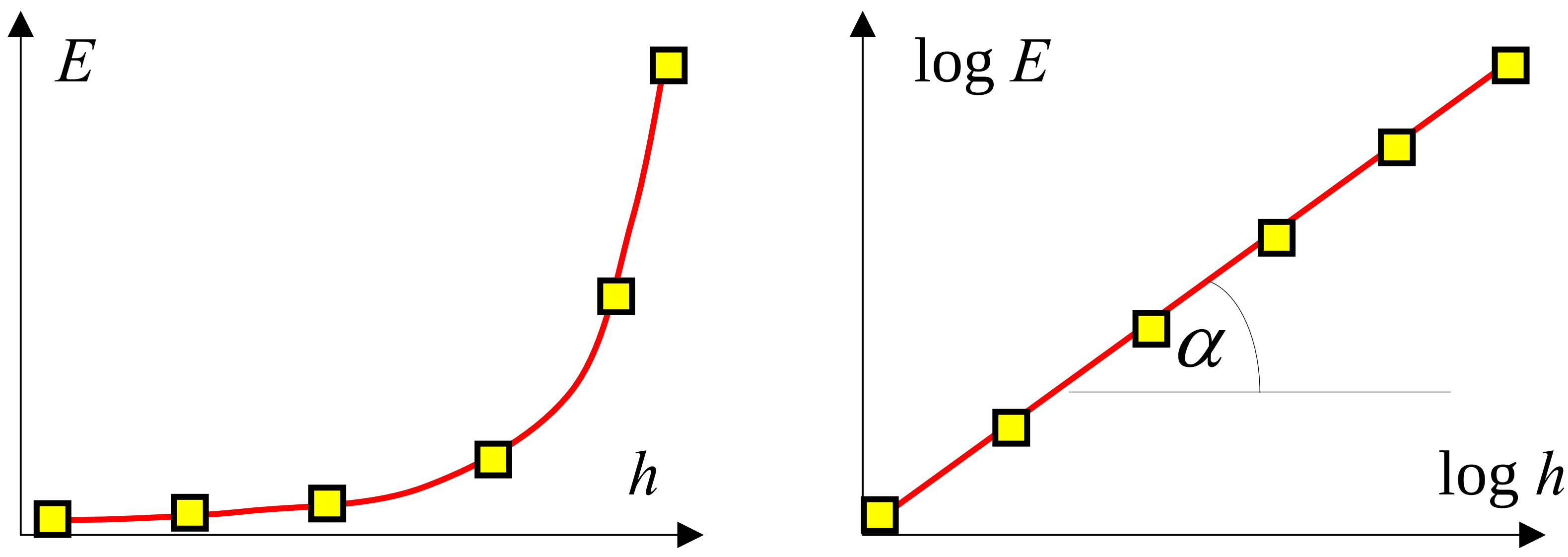
The smaller element size in FEM (and also FDM, FVM etc.), the smaller the error. The approximate solution is converging to the exact one.

Convergence rate describes how quickly the convergence occurs, i.e.

- $h = h_0 / 2 \rightarrow E = E_0 / 2, \quad E = Ch$ rate = 1
- $h = h_0 / 2 \rightarrow E = E_0 / 4, \quad E = Ch^2$ rate = 2
- $h = h_0 / 2 \rightarrow E = E_0 / 8, \quad E = Ch^3$ rate = 3

Convergence plot

Solution on a mesh with element of size h has the error E.
It is *a* point per solution in a size-error coordinate system.



$$E = Ch^r \qquad \log E = \log C + r \log h$$

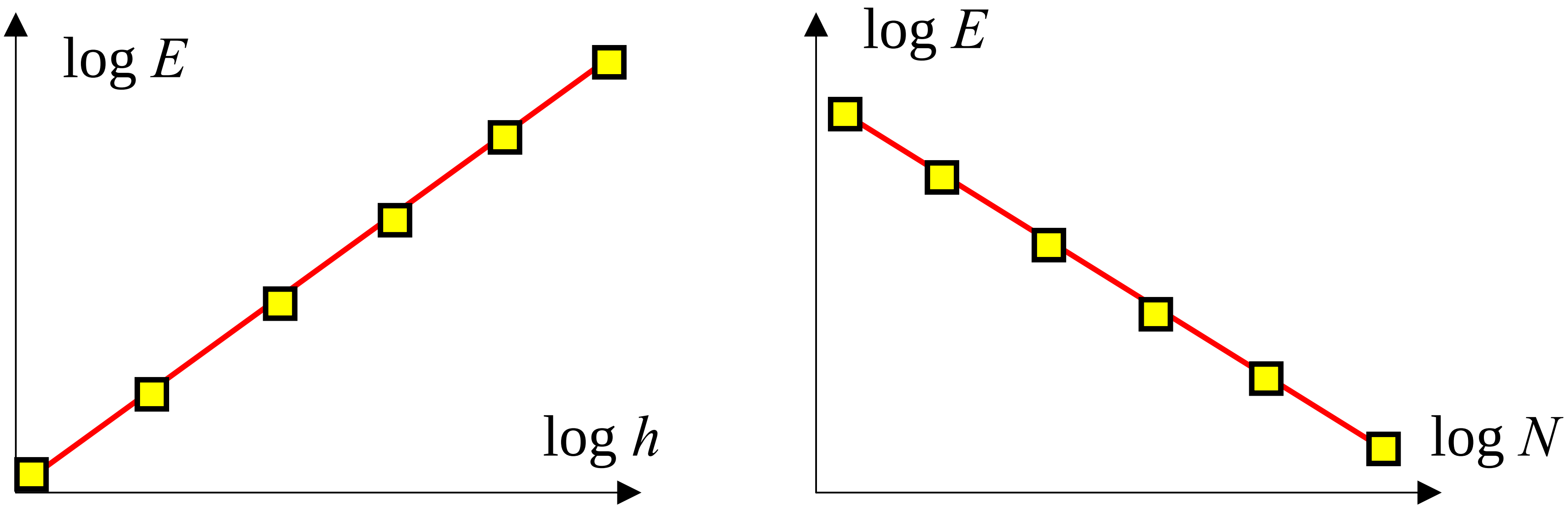
$r = \mathrm{tg}\,\alpha$ is convergence rate, *h* measures problem size

Convergence plot, cont.

If a discretization is non-uniform, total number of nodes *N* is a better measure of problem size. For unit line, square etc.

$$h = 1/N = O(N^{-1}) \text{ 1D} \qquad h = 1/\sqrt{N} = O(N^{-1/2}) \text{ 2D}$$

$$h = 1/\sqrt[3]{N} = O(N^{-1/3}) \text{ 3D} \qquad p = O(N)$$



Convergence of FEM with h-refinement

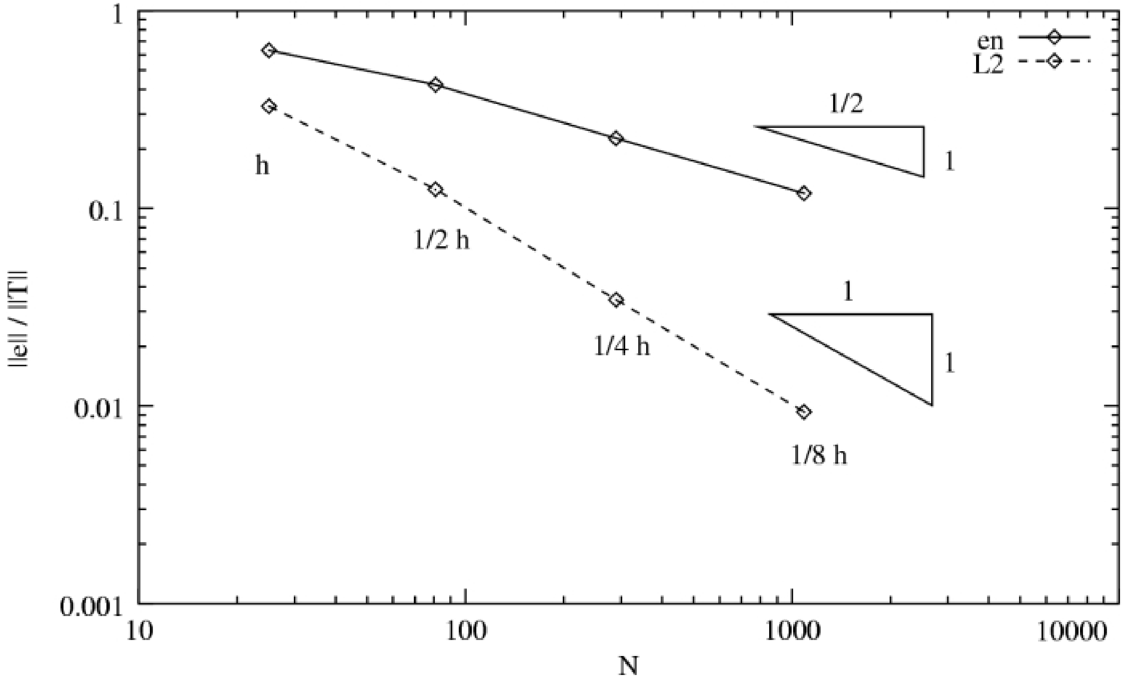
$$\|e\|_{L_2(\Omega)} = C_1 h^{p+1}$$

$$\|e\|_{E(\Omega)} = C_2 h^{p}$$

(constants depend on the solution)

$$\|e\| = C N^{-q}$$

algebraic convergence



Convergence of FEM with p-refinement

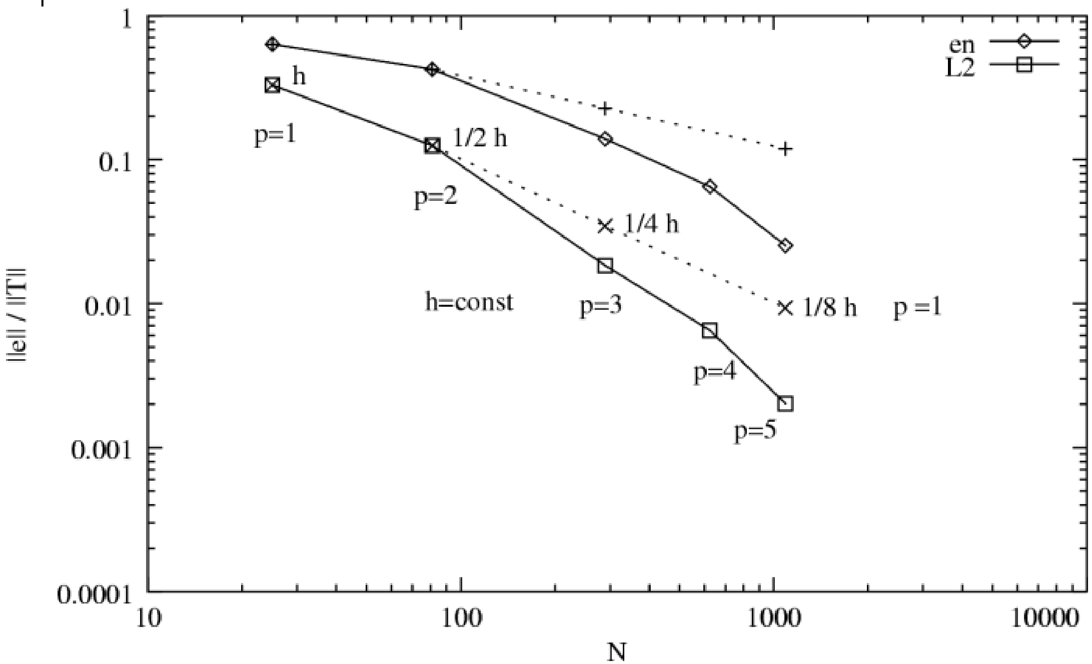
$$\|e\|_{L_2(\Omega)} = C_1 \frac{h^{p+1}}{p^r}$$

$$\|e\|_{E(\Omega)} = C_2 \frac{h^p}{p^r}$$

(constants depend on the solution)

$$\|e\| = C \exp\left(-q_1 N^{q_2}\right)$$

exponential convergence



Adaptivity

- Smaller element size gives more accurate results but at higher cost. Shorter step size means more time steps to do. A trade-off exists between accuracy and solution time.
- Adaptivity allows to reduce cost while controlling accuracy.
- Cost reduction is possible thanks to refining where necessary. Accuracy is assessed by error estimates.
- According to user needs it is possible to
  - obtain the solution with required accuracy at the lowest cost
  - obtain the most accurate solution possible

Adaptive loop for a steady problem

Mesh is adapted to the solution iteratively

Assume a coarse initial mesh

repeat

  Solve the problem on current mesh

  Estimate the error of the solution

  if the solution is not accurate enough **then**

    Select where to refine or coarsen the mesh

    Modify the mesh

until required accuracy is achieved

Mesh adaptation for a transient problem

Mesh adaptation can be embedded in a transient loop

while $t < T$

Assume a coarse initial mesh or use the previous mesh

repeat

Transfer the solution from the previous to the current mesh

Solve the time step on the current mesh

Estimate the error of the solution

if the solution is not accurate enough **then**

Select where to refine or coarsen the mesh

Modify the mesh

until required accuracy is achieved

$t \leftarrow t + \Delta t$

Error estimation

- Error estimation is a key part of adaptive computations

  quantitative information on accuracy, local estimates (called error indicators) point out places to refine, global estimates can be used in stopping condition

- Error estimators: estimate the error of a solution in terms of this solution and problem data

- Estimators of error due to space discretization

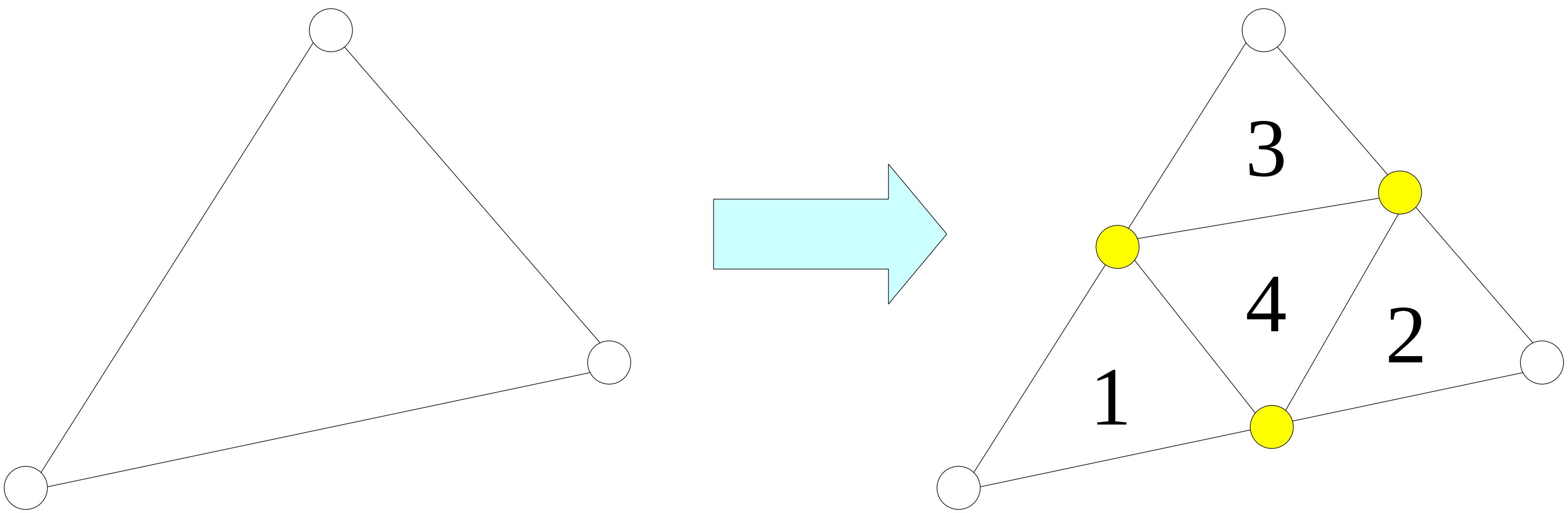
  residual (explicit and implicit), averaging based (postprocessing of solution derivatives)

- Estimators of error due to time discretization

  second temporal derivative, helper solution with half step

Refinement by subdivision

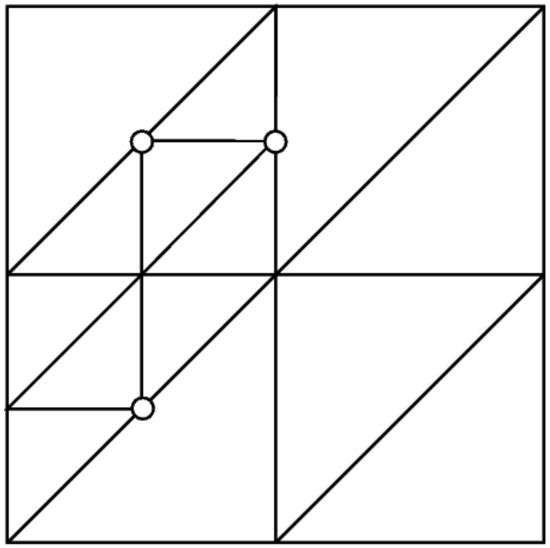
Subdivide a triangle into four smaller triangles



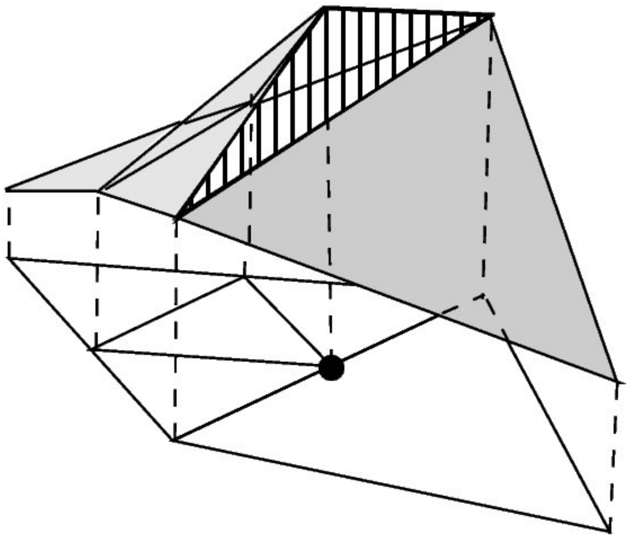
Uniform and isotropic $h = \frac{1}{2} h_0$

Hanging nodes

- A mesh is *irregular* if it contains *hanging nodes*
- Hanging nodes cause discontinuity. *Constraints* should be imposed to prevent it



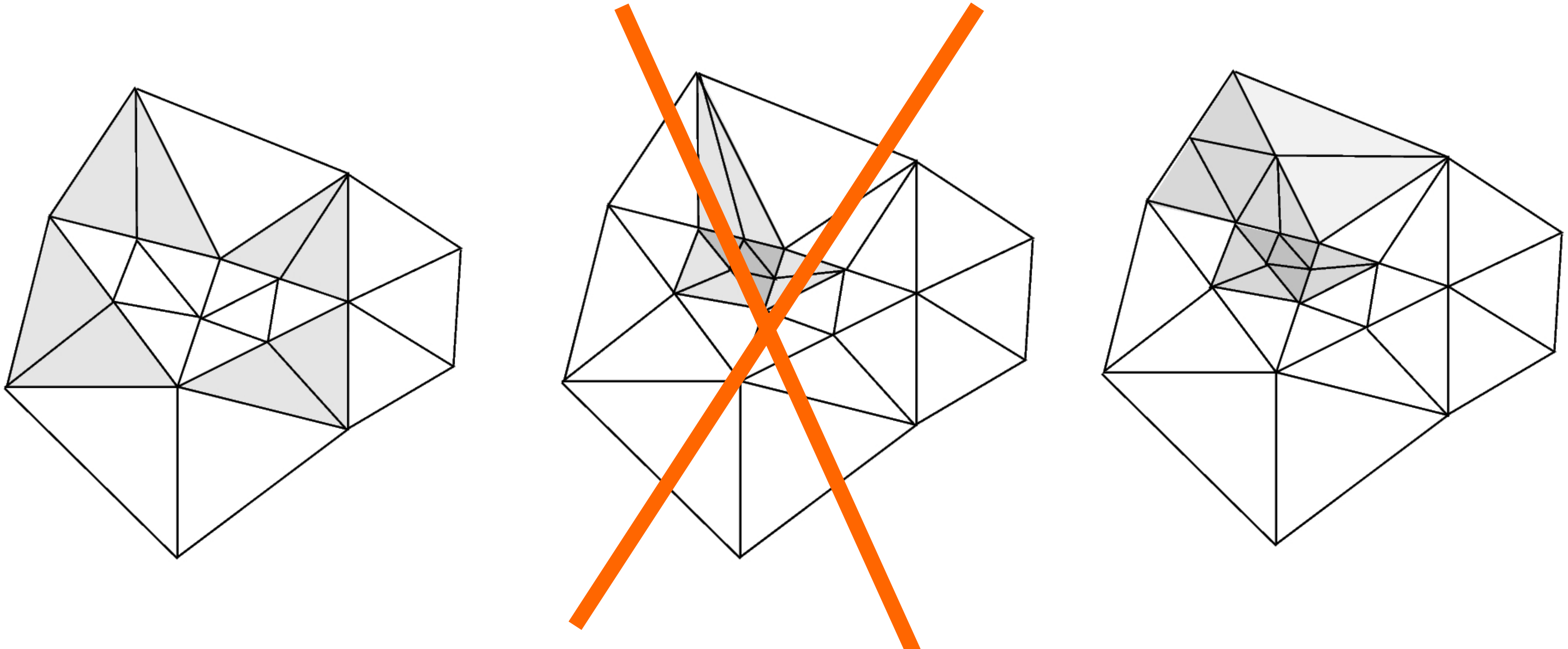
A one-irregular mesh



Discontinuity of approximation

Elimination of hanging nodes

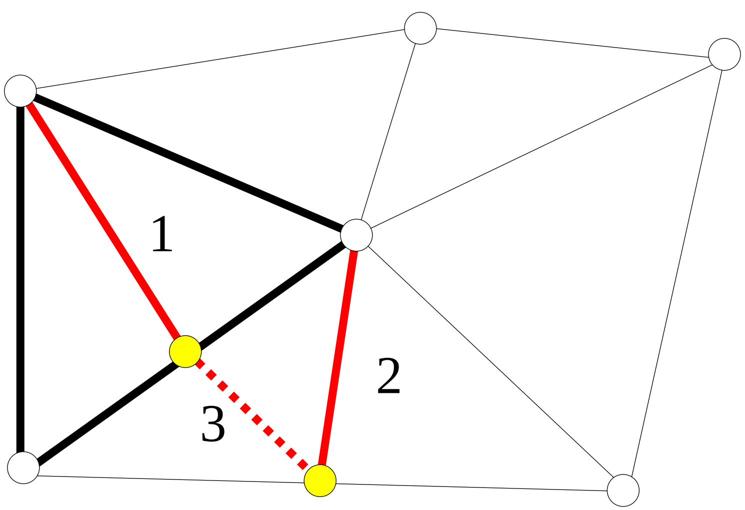
- Triangle with hanging node(s) is a *green triangle*
- A green triangle can be bisected into 2 *transient elements*



Bad shape should be avoided - one-irregular rule, three-neighbour rule

Alternative refinement by subdivision

Subdivision into two triangles by the longest edge



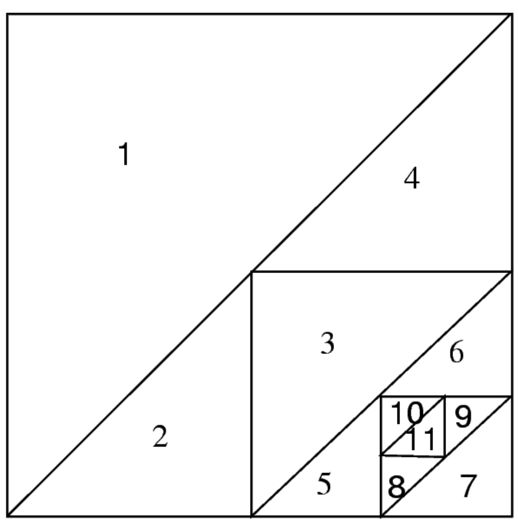
Subdivision of the bold triangle may cause subdivision of neighbour(s)

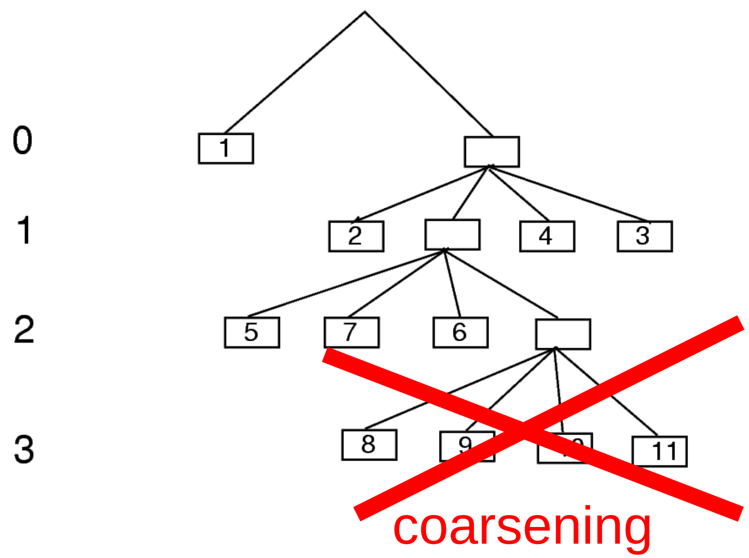
Rivara algorithm of refinement of a set of elements

- Bisect all elements in the set by the longest edge (1)
- Place all irregular elements in the set (the bottom triangle)
- Bisect all elements from the set (2).
- If the midpoint point is a new hanging node, connect it with the other (3)
- Repeat until the set is empty

Tree data structure

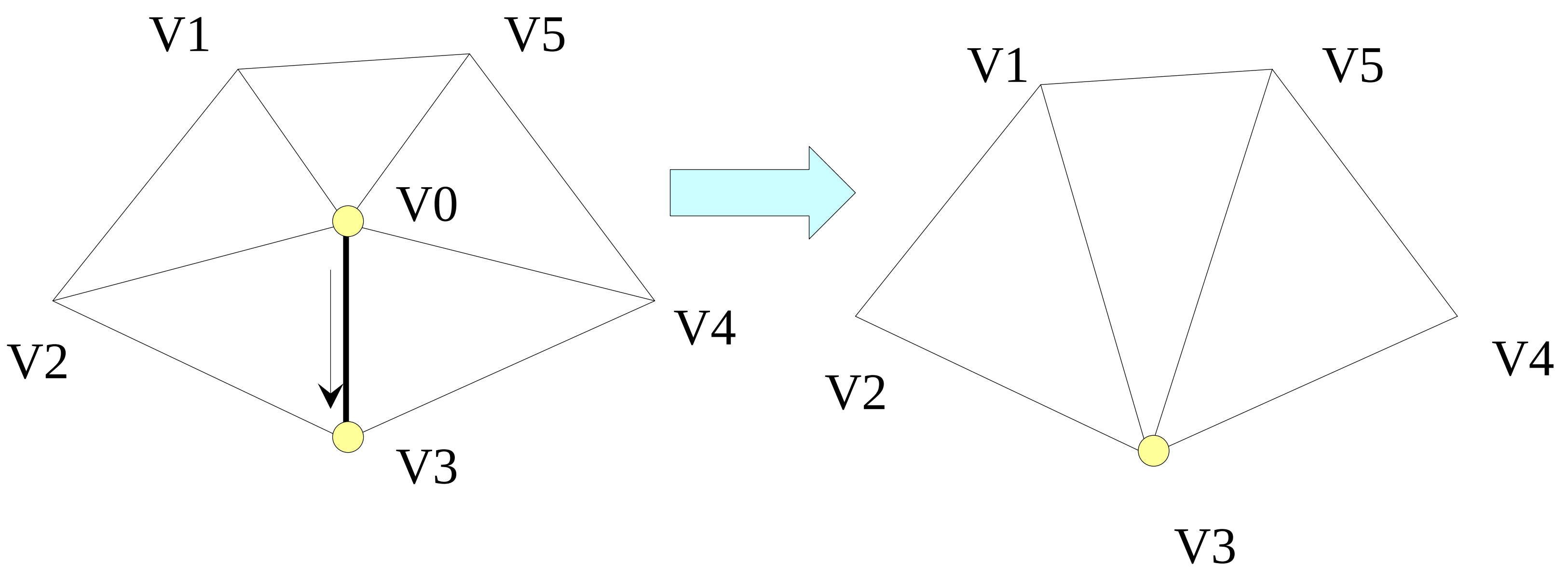
- Element being subdivided is the parent. It has four or two child elements
- Refinement: the parent becomes a tree node, children become leafs
- Coarsening: pruning leaf elements and restoring the parent





Coarsening by edge collapsing

- Coarsening does not require earlier refinement
- Edge collapsing: make the length of an edge equal to zero by shifting the first vertex to the second one
- Erase all internal edges, then retriangulate
- *Collapsed vertex* V0 is shifted to the *target vertex* V3



p-Refinement and hierarchic approximation

- Mesh is adapted to the solution by varying polynomial degree of elements, whereas mesh geometry is fixed
- A family of finite elements with increasing order of approximation is needed
- Hierarchic basis functions are well suited for refinement. In a higher order element a correction is added to the solution in the original element

linear

$$u^{(1)}(x) = N_0^{(1)}(x)\,u_0 + N_1^{(1)}(x)\,u_1$$

quadratic

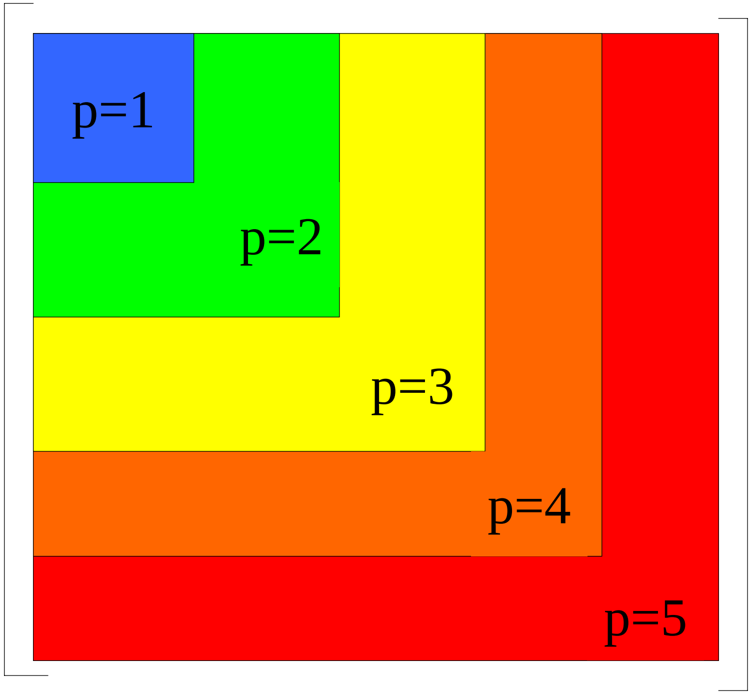
$$u^{(2)}(x) = N_0^{(1)}(x)\,u_0 + N_1^{(1)}(x)\,u_1 \boxed{+ N_2^{(2)}(x)\,a_2}$$

cubic

$$u^{(3)}(x) = N_0^{(1)}(x)\,u_0 + N_1^{(1)}(x)\,u_1 + N_2^{(2)}(x)\,a_2 \boxed{+ N_3^{(3)}(x)\,a_3}$$

Mass and stiffness matrices of hierarchical elements

- Mass and stiffness matrices are the integrals of product of shape functions or of their derivatives
- Mass or stiffness matrix of element of the order *p+1* includes the matrix of the element of the order *p*



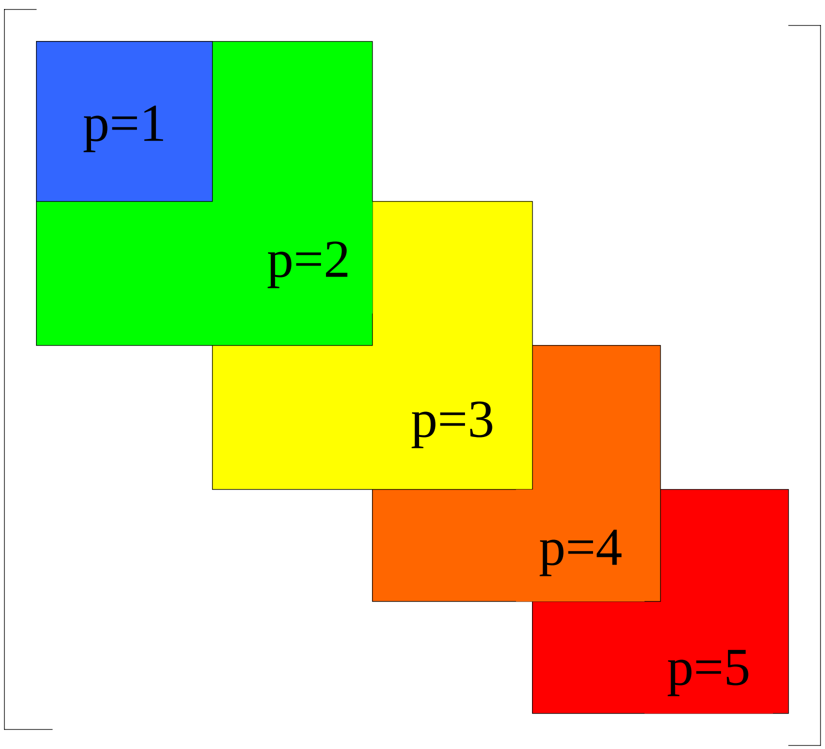
The matrix inherited from the element of order *p-1* is *augmented* with terms involving the new *p*-th order functions

Hierarchical degrees of freedom are **not the values of the solution** at nodes or edges or in the centre

Orthogonality

Two shape functions or their gradients are orthogonal if the integral of the appropriate product is zero.

$$\int_\Omega N_i N_j dx = 0 \qquad \int_\Omega \nabla N_i \cdot \nabla N_j dx = 0, \quad i \neq j$$



If integrals of some products of shape functions are zero, the mass matrix will have zero elements. Similarly for gradients and the stiffness matrix.

Example stiffness matrix for a hierarchical triangle

Integrated Legendre polynomials

- Legendre polynomials are orthogonal in $[0,1]$

$$P_p(\xi) = \frac{1}{(p-1)!}\frac{1}{2^{p-1}}\frac{d^p}{d\xi^p}\left[\left(\xi^2-1\right)^p\right]$$

- The mass matrix is nearly diagonal if basis functions are Legendre polynomials
- The stiffness matrix nearly diagonal if basis function derivatives are Legendre polynomials. Then

$$N^{(p)} = \int P_{p-1} d\xi, \quad \xi(0) = \xi(1) = 0, \quad p \geq 2$$

$$N^{(2)} = \xi^2 - 1, \quad N^{(3)} = 2\left(\xi^3 - \xi\right), \quad N^{(4)} = \frac{1}{4}\left(15\xi^4 - 18\xi^2 + 3\right) \text{ etc.}$$

Hierarchical basis functions on the triangle

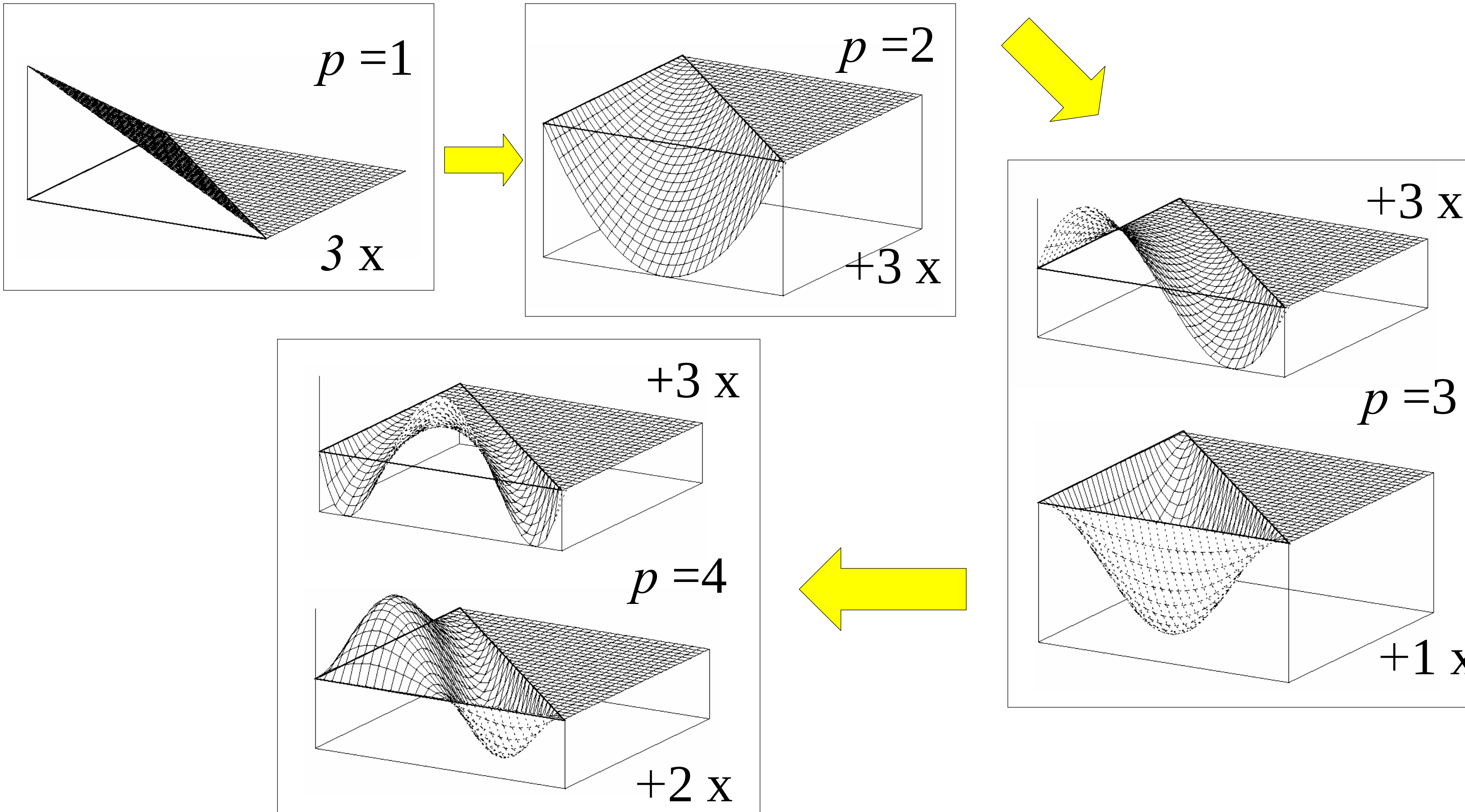
- A hierarchical basis function associated with an edge should reduce to one dimensional-function on the edge. Such an *edge function* should be zero on other edges and at vertices

- Three edge functions of the order *p* can be introduced

- Starting from *p*=3 extra functions are needed to represent complete polynomial. These are the *bubble functions* associated with element interior*.* They are zero on element boundary

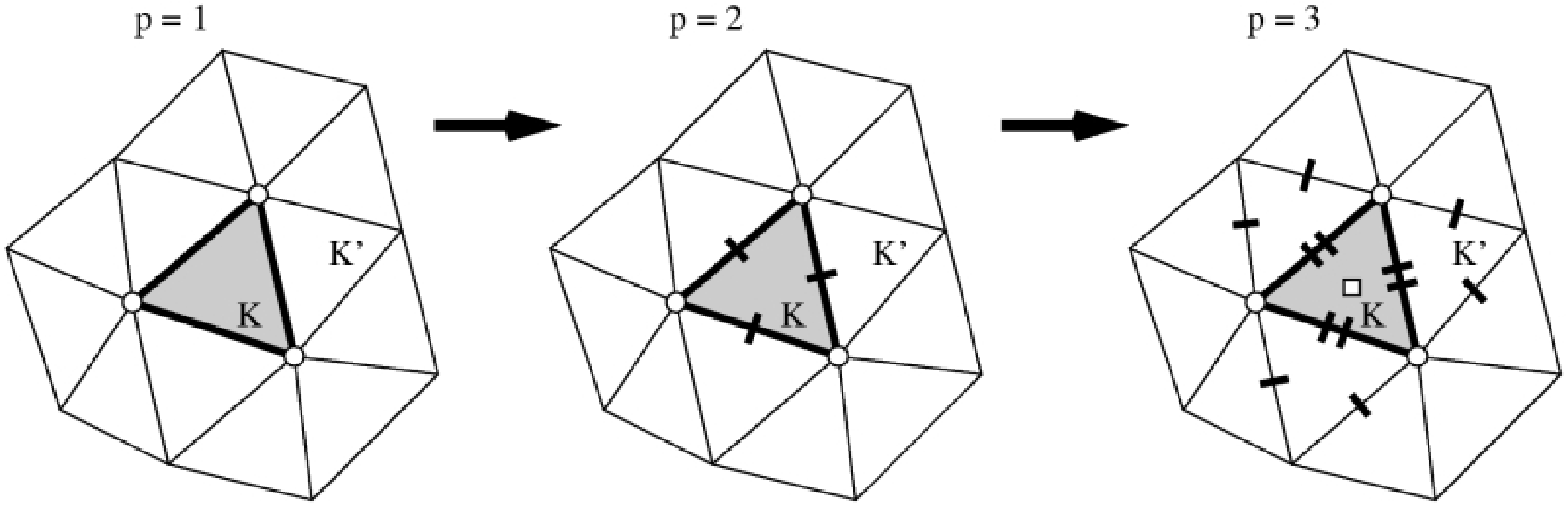
- Bubble functions can be defined as products of edge and vertex functions

Hierarchical basis functions on the triangle, cont.



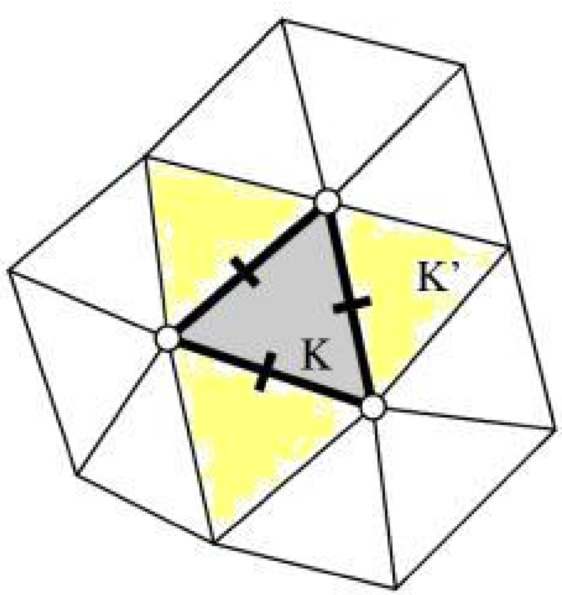
p-Refinement

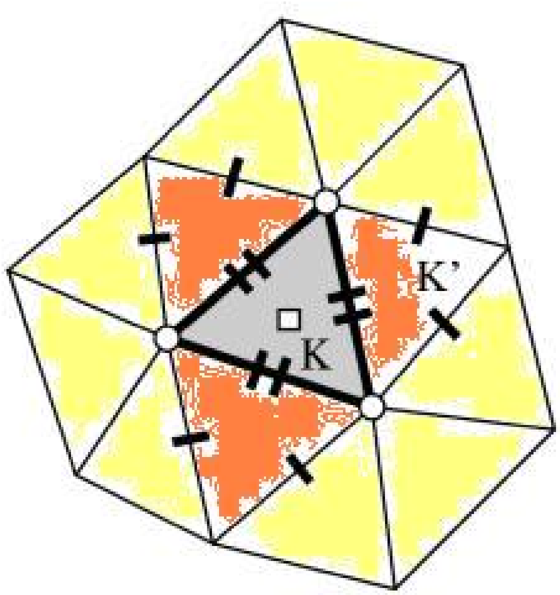
- Refinement: introduce a higher order correction (extra basis functions and degrees of freedom) without altering element shape
- The one-level difference rule should be obeyed
- Coarsening: removal of some hierarchical deg. of freedom



p-Refinement, cont.

- Transient elements need to be introduced to preserve continuity of approximation
- A transient element has extra unknowns on the edge shared with a higher-order element





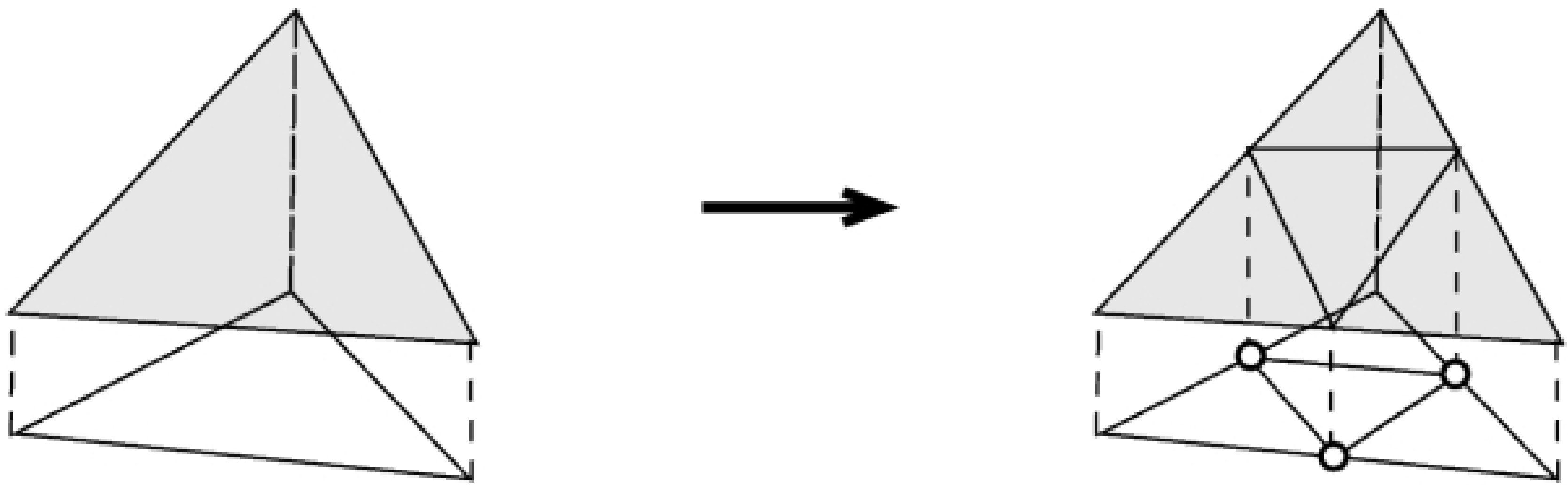
Solution transfer

After refinement, a solution from the original mesh often needs to be *transferred (projected)* to the new adapted mesh

- new starting vector for an iterative solution method (conjugate gradients etc.)
- previous step solution in time stepping schemes
- solution from the previous non-linear iteration

Solution transfer during h-refinement by subdivision

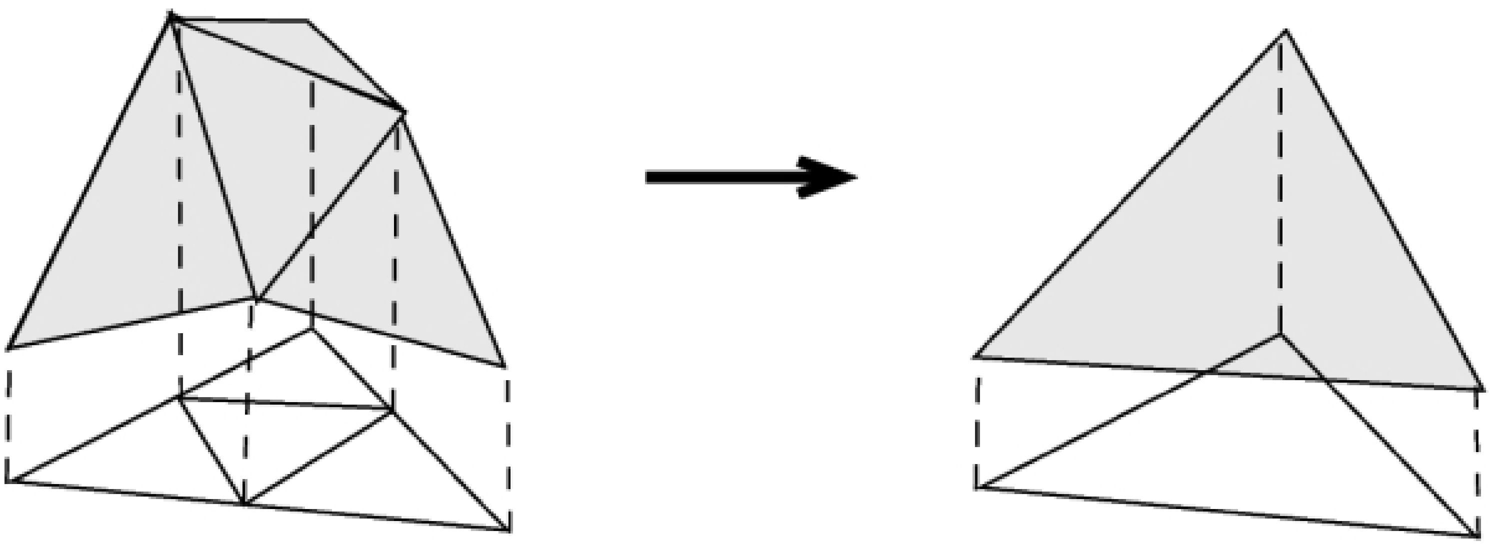
- Linear interpolation at new midpoint nodes



- Interpolation involving more surrounding nodes is more accurate (e.g moving least-squares)

Solution transfer during coarsening of h-refined mesh

- Linear interpolation in parent element instead of piecewise-linear interpolation in deleted children



- Loss of accuracy

Solution transfer during p-refimenent and coarsening

- Higher-order unknowns introduced can be initially 0
- Trimming highest-order unknowns during coarsening

Before refinement

$$u_h^{(p)} = \sum_{i=1}^{M} N_i a_i$$

After refinement

$$u_h^{(p+1)} = \sum_{i=1}^{M} N_i a_i + \sum_{j=1}^{N} N_{M+j}^{(p+1)} a_{M+j}$$

higher-order correction

$$u_h^{(p+1)} = u_h^{(p)} \quad \text{if} \quad a_{M+1} = 0,\, a_{M+2} = 0, \ldots$$

higher-order unknowns (degrees of freeedom)

Adaptive strategy

Assume error indicators and global estimate is available

- How many elements of the mesh are to refine, how many to coarsen (if any)? What is the preferred element size?
- h-refinement, p-refinement or both? Adapt the time step?
- Should the mesh be adapted in this time step?

More refined elements per iteration means greater error reduction. Overrefinement is possible.

Fewer refined elements - more iterations of the adaptive loop

Fixed adaptive strategies

- *Fixed threshold strategy*
  - Refine elements with the error greater than $E_r = \alpha_r E_{\max}$
  - Coarsen elements with the error less than $E_c = \alpha_c E_{\max}$
- *Fixed fraction strategy*
  - Refine $\alpha_r \cdot 100\%$ elements with the highest error
  - Coarsen $\alpha_c \cdot 100\%$ elements with the lowest error
- *Fixed contribution strategy*
  - Refine elements with the highest error where

$$\sum_{elems} E_{elem} = \alpha_r E_{total}$$

Coarsen elements with the lowest error contributing $\alpha_c E_{total}$

Error equidistribution strategy

Error equidistribution (Zienkiewicz-Zhu) strategy

Each element should contribute the same local error in energy to the total error

$$E_{elem}^2 = E_{avg}^2 = \frac{E_{total}^2}{N_{elems}} = \frac{\left(\eta \left\| u_h \right\|_E\right)^2}{N_{elems}}$$

$$E_{elem} = C h_{elem}^p$$ (a priori estimates of convergence)

Suggested local element size is

$$\boxed{h_{elem}^{new} = h_{elem} \left(\frac{\eta \left\| u_h \right\|_E}{E_{elem} \sqrt{N_{elems}}} \right)^{1/p}}$$

Subdivide if $h^{new} < 0.5h$

Unrefine if $h^{new} > 2h$

Automatic selection of time step size

Adaptation of time step size in the *k*-th time increment

- Solve the step at $t_{k-1} + \Delta t_k$, estimate the error E
- Calculate the suggested step length

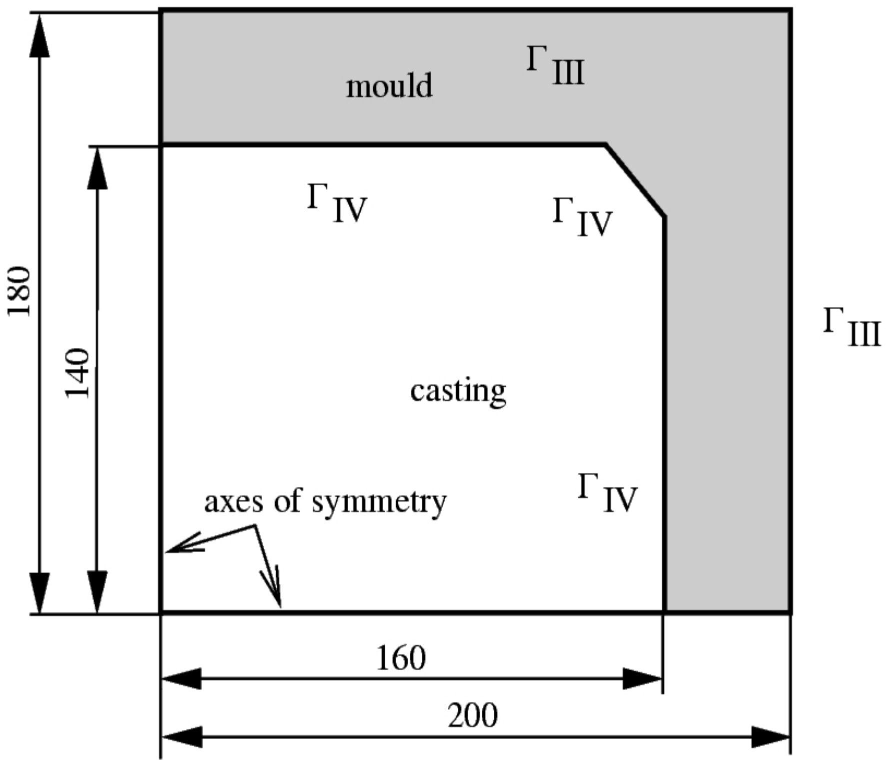
$$\Delta t_k^{new} = \Delta t_k \left(\frac{\eta_t \left\| \Delta u \right\|}{E} \right)^{\frac{1}{n+1}}$$

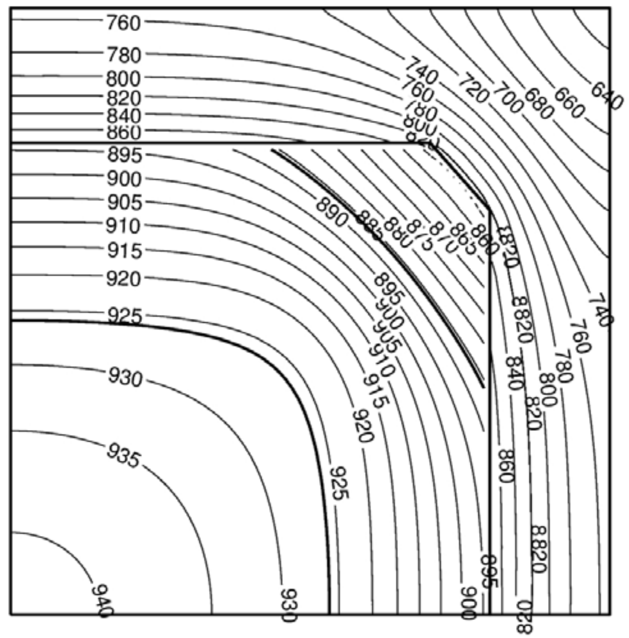
n - order of the method

- If $E < E_{crit} = \eta \left\| \Delta u \right\|$ (the solution is accurate enough)
  - use this step length in the next time increment *k+1*
- Otherwise repeat the step (find the solution at $t_{k-1} + \Delta t_k^{new}$)

Adaptive solution of a solidification problem

Solidification of a casting made of Al-Cu alloy

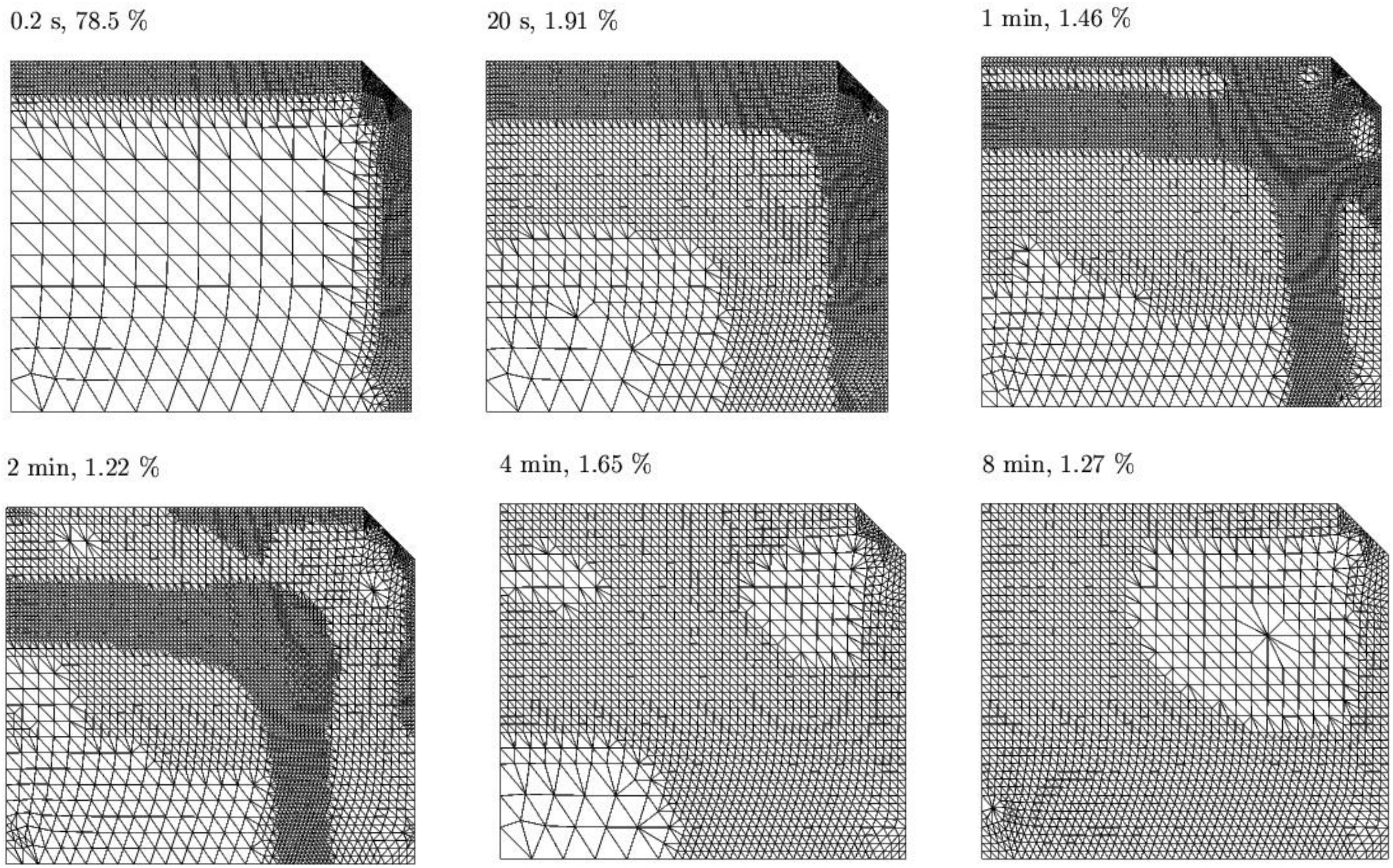




T = 886 K - solidus isotherm, T = 926 K - liquidus isotherm

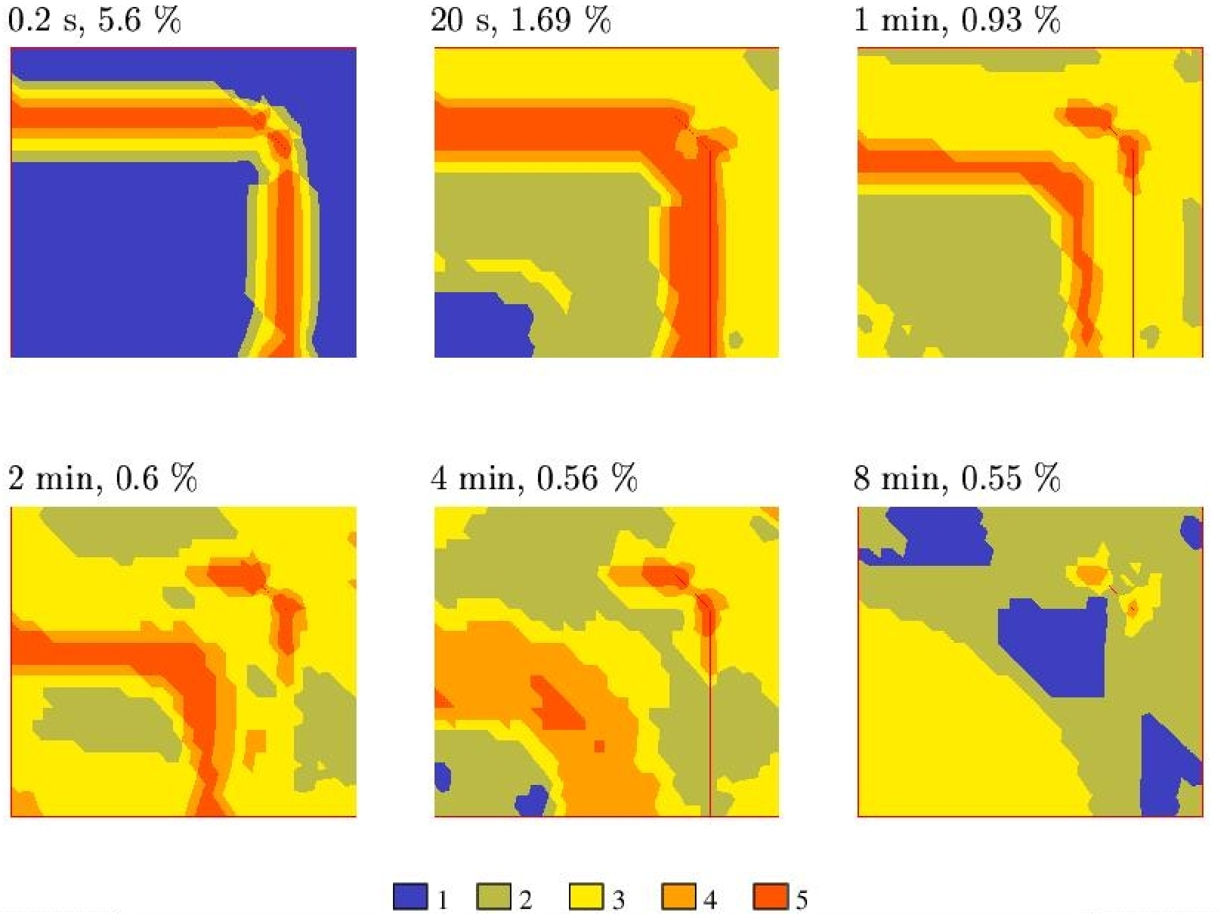
Solution of the solidification problem with h-refinement

Final meshes at selected times



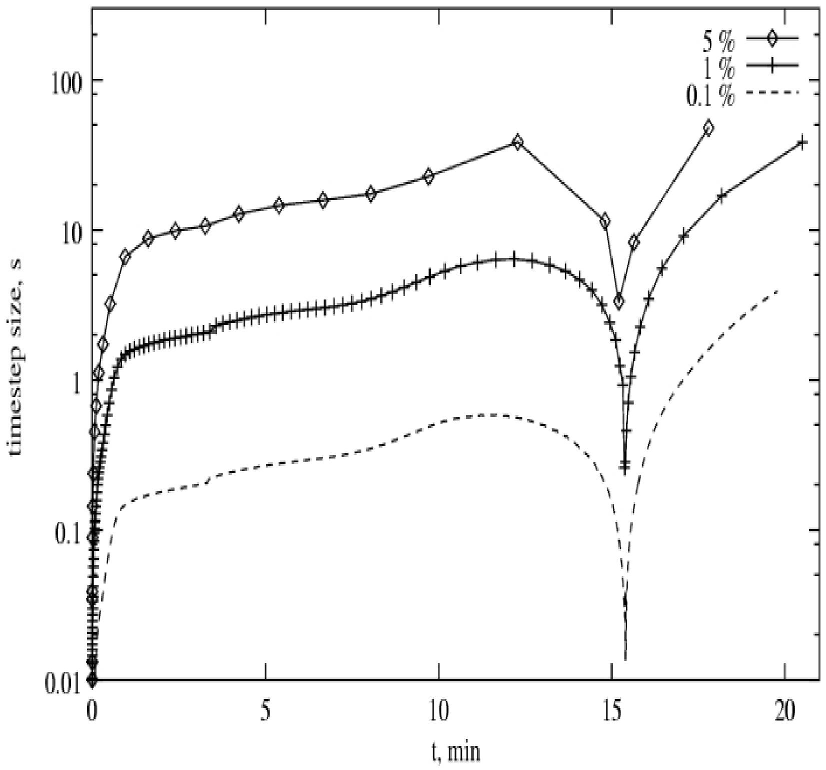
Solution of the solidification problem with p-refinement

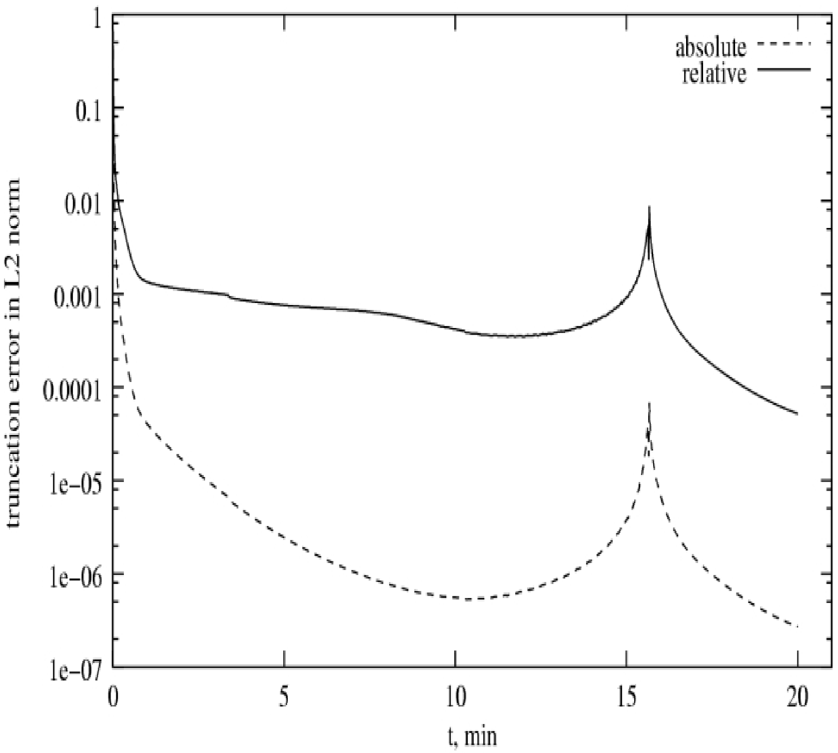
Final meshes at selected times



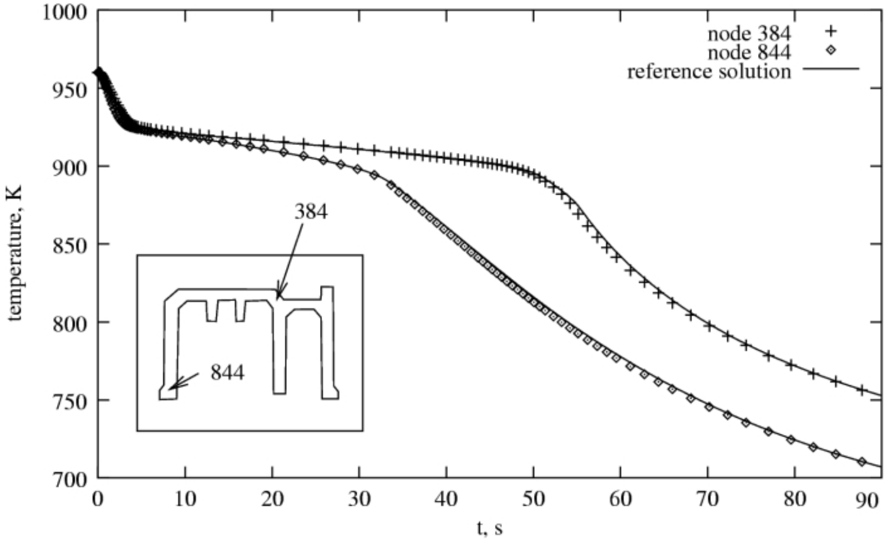
Solidification with adaptive time stepping

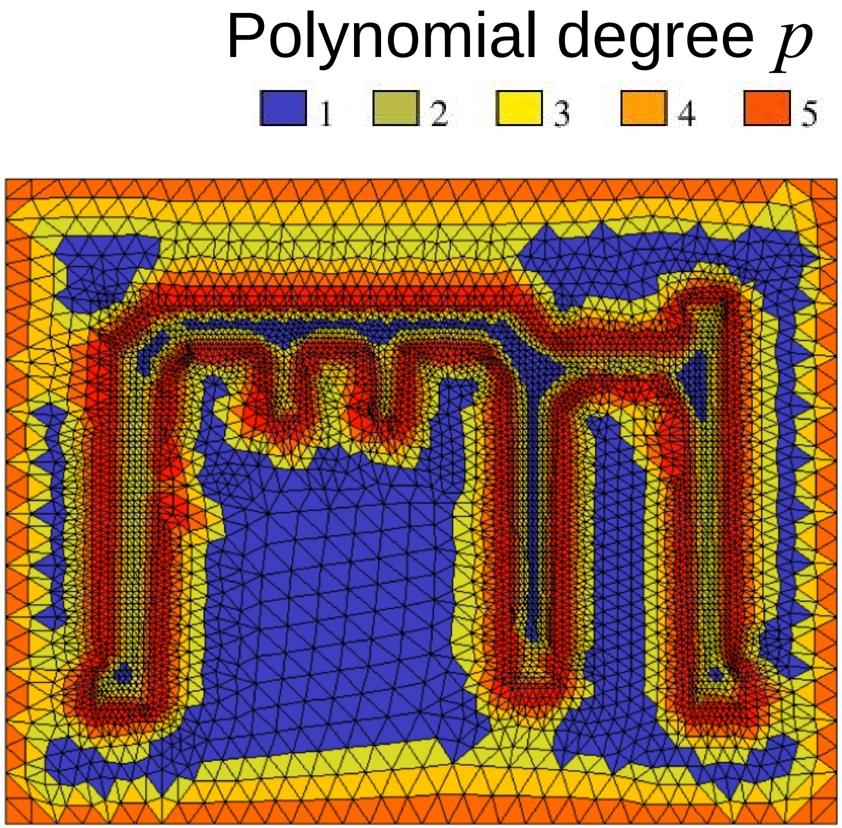
Adapted time step reflects the error of time stepping





Solidification with hp-refinement and adaptive time stepping





Time step is the distance between the symbols. It is smaller when solidification starts and finishes.

