Coarsening in Adaptive Finite Element Methods

Peter Binev

University of South Carolina

WPI workshop "Adaptive numerical methods for PDE's"

Vienna January 23, 2008

Adaptive methods for approximation

- Given $f \in X$ define a sequence of approximations $f_j \in S_j \subset X$ for $j = 0, 1, 2, \ldots$.
- The space S_{j+1} is chosen from a (fixed) variety of finite dimensional spaces after f_j is found.
- We require $n_j < n_{j+1}$ where n_j is the dimension of S_j.
- In most of the cases $S_j \subset S_{j+1}$.
- The decision about S_{j+1} is made based upon the information how f_j approximates f.

Adaptive methods for approximation II

- Usually, $S_j = span\{\psi_{j,k} : k = 1, ..., n_j\}$.
- Certain functionals $\Phi_{j,m}(f, f_j)$ are used to indicate to what extent adding some set $\Psi_{j,m}$ of new basic functions might contribute to the improvement of the approximation.
- In many cases $\Phi_{j,m}$ is a *local error indicator*, and $\sum_m \Phi_{j,m}(f, f_j)$ relates to the error $\|f - f_j\|$.
- In the PDE settings “local” means only that it is a local quantity but not necessarily that $\Phi_{j,m}$ gives a bound for the local error.

Goal

Find methods that realize *optimal* balance between accuracy and resources.

- adaptive methods – *make decisions during the computations*
- sparse representations
- complexity $\leftrightsquigarrow$ information content

Can we derive theoretical estimates about the performance?

Example: A Toy Problem

Target function: $f \in L_\infty[0,1]$

Piecewise constant approximation on N intervals: $\tilde{f}_N$

Error measured in $L_\infty[0,1]$

- Linear Approximation

$$f' \in L_\infty[0,1] \quad \Rightarrow \quad \|f - \tilde{f}_N\|_\infty \leq N^{-1}\,\|f'\|_\infty$$

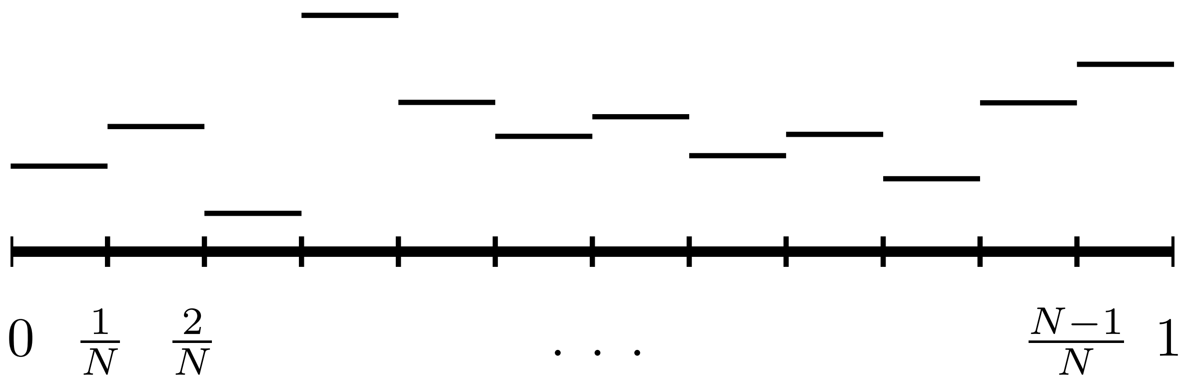
- Nonlinear Approximation

$$f' \in L_1[0,1] \quad \Rightarrow \quad \|f - \tilde{f}_N\|_\infty \leq N^{-1}\,\|f'\|_1$$

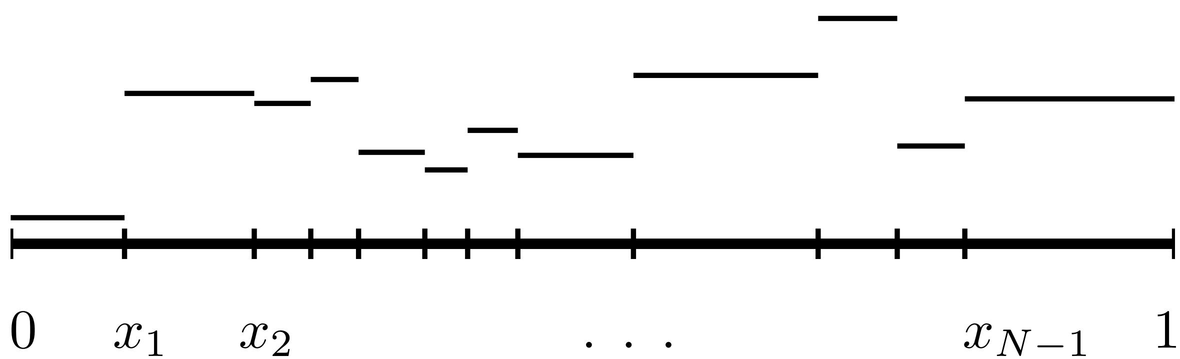
$$\|u\|_1 := \int_0^1 |u(x)|dx \qquad \|u\|_\infty := \sup_{x\in[0,1]} |u(x)|$$

Linear *vs* Nonlinear Approximation

Linear Approximation $S_N := \left\{ \tilde{f} : \begin{array}{l} \tilde{f} \text{ - piecewise constant, } N \text{ pieces,} \\ \textit{fixed} \text{ breakpoints } \left\{ \frac{k}{N} \right\}_{k=0}^{N} \end{array} \right\}$



Nonlinear Approximation $\Sigma_N := \left\{ \tilde{f} : \begin{array}{l} \tilde{f} \text{ - piecewise constant, } N \text{ pieces,} \\ \textit{arbitrary} \text{ breakpoints } \{x_k\}_{k=1}^{N-1} \end{array} \right\}$



Approximation of Functions

Universal set $\Psi = \{\psi_\lambda\}_{\lambda \in \Lambda^\star}$ with index set $\Lambda^\star$

Linear Approximation: $\tilde{f} = \sum_{\lambda \in \Lambda} c_\lambda \psi_\lambda$ Λ is fixed.

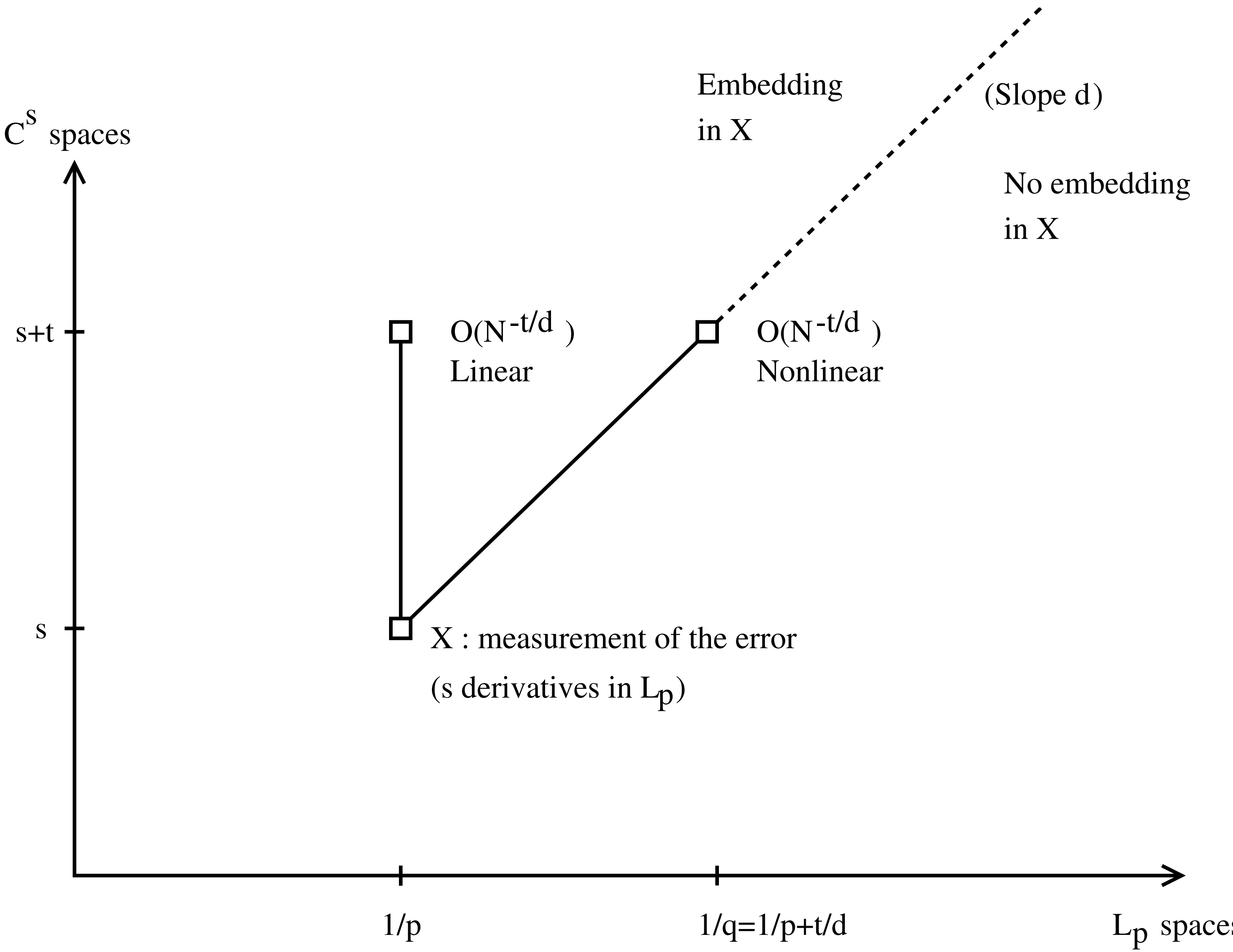
for a given set Λ *with* $\#\Lambda = N$ *find constants* c_λ
so that $\|f - \tilde{f}\|$ *is as small as possible*

Nonlinear Approximation: $\tilde{f} = \sum_{\lambda \in \Lambda} c_\lambda \psi_\lambda$ Λ can vary.

find a set $\Lambda \subset \Lambda^\star$ *with* $\#\Lambda \leq N$ *and constants* c_λ
so that $\|f - \tilde{f}\|$ *is as small as possible*

Data organization $\rightsquigarrow$ structure of $\Lambda \subset \Lambda^\star$

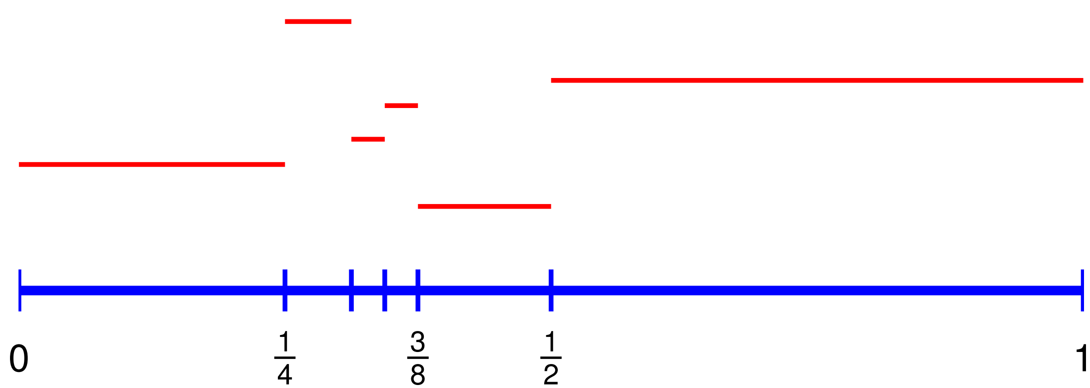
Topography of smoothness spaces

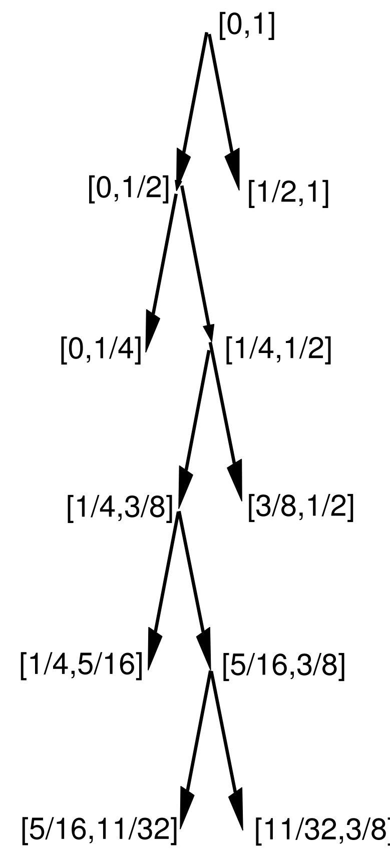


Nonlinear Adaptive Approximation

Dyadic Intervals : $\lambda = \left[\frac{k}{2^j}, \frac{k+1}{2^j}\right]$

Associated Tree





Wavelet Representations

Featured Applications: Image Compression (JPEG 2000) and Numerical PDE

wavelet representation – the infinite index set is a tree $\Lambda^\star = T^\star$

$$f = \sum_{\lambda \in \Lambda^\star} f_\lambda \psi_\lambda$$

best N-term approximation

$$\sigma_N(f)_X := \inf_{\Lambda \subset \Lambda^\star,\ \#\Lambda \le N} \left\| f - \sum_{\lambda \in \Lambda} f_\lambda \psi_\lambda \right\|_X$$

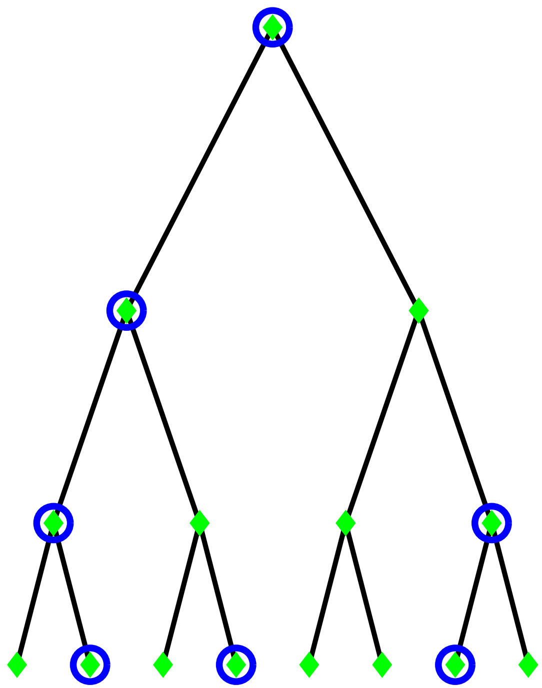
best N-term tree approximation

$$\sigma_N^\tau(f)_X := \inf_{T \text{ is a subtree of } T^\star,\ \#T \le N} \left\| f - \sum_{\lambda \in T} f_\lambda \psi_\lambda \right\|_X$$

N-term *vs* Tree Approximation

N-term Approximation

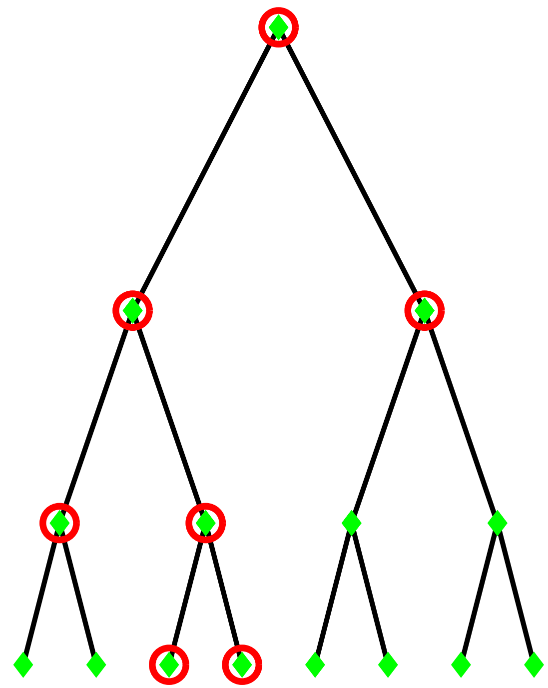
$$\tilde{f} = \sum_{\lambda \in \Lambda} c_\lambda \psi_\lambda \qquad \#\Lambda \leq N$$



Λ *can be* any *set*

Tree Approximation

$$\tilde{f} = \sum_{\lambda \in \tau} c_\lambda \psi_\lambda \qquad \#\tau \leq N$$



τ *must be a* tree

Best Tree Approximation

Approximation class

$$\mathcal{A}^s_X := \left\{ f \in X \ : \ \|f\|_{\mathcal{A}^s_X} := \sup_N \ N^s \sigma^\tau_N(f)_X \ < \infty \right\}$$

Theorem **[Cohen, Daubechies, Dahmen, DeVore]**
compact embedding $\rightsquigarrow$ *tree approximation is optimal*

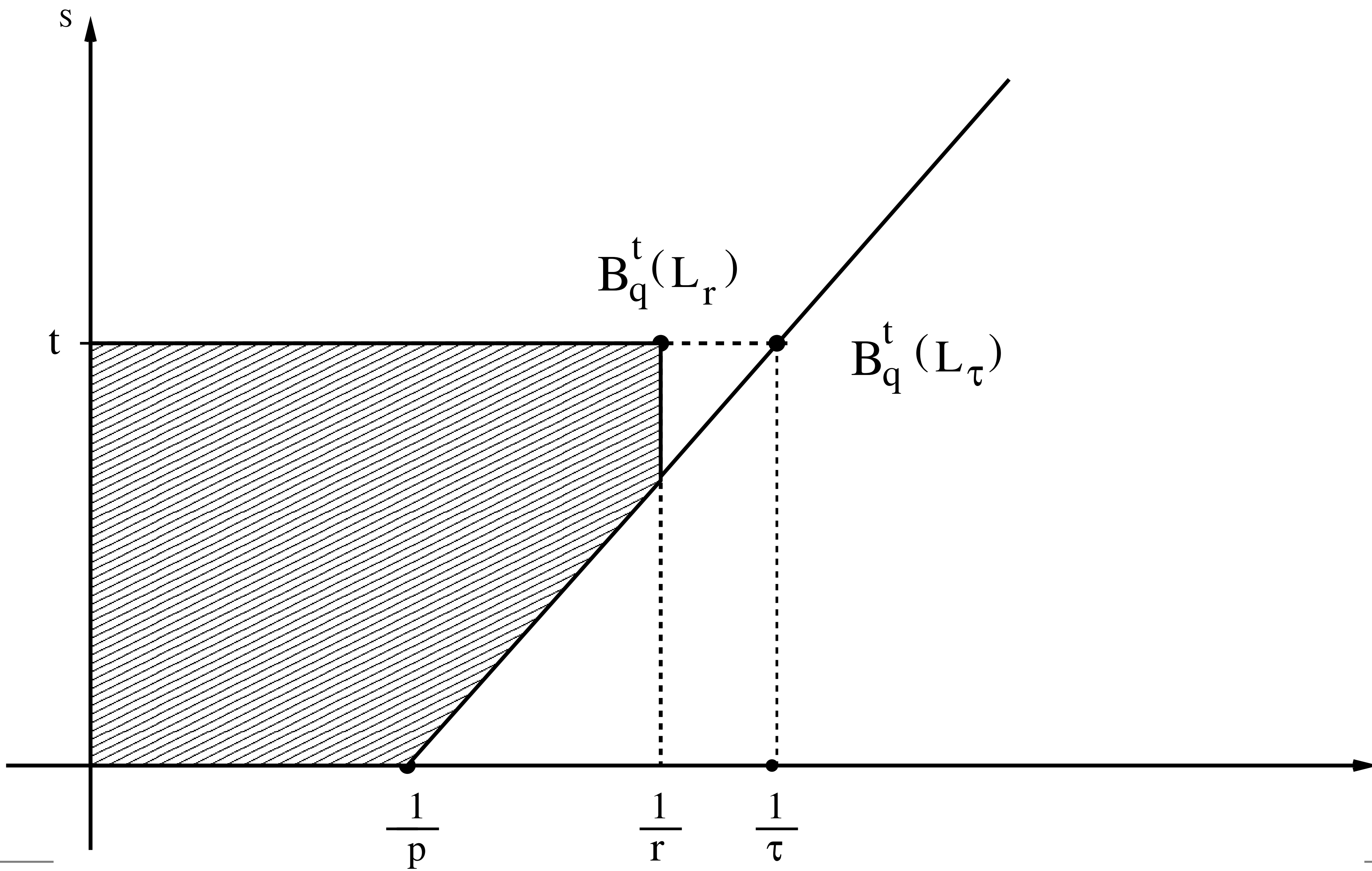
e.g. $X = W^t(L_q)$ – functions in $\mathbb{R}^d$ with t derivatives in L_p

$$W^{t+sd}(L_q) \subset \mathcal{A}^s_{W^t(L_p)} \quad \text{if} \quad \frac{1}{q} < s + \frac{1}{p}$$

$\mathcal{U}$ – unit ball in $W^{t+sd}(L_q)$

$$\sup_{f \in \mathcal{U}} \sigma^\tau_N(f)_X \leq C \ \sup_{f \in \mathcal{U}} \sigma_N(f)_X$$

Smoothness spaces for tree approximation



Advantages of Tree Approximation

- an efficient encoding of indices λ
- in image compression $\rightsquigarrow$ embedded zero-trees
- locally refined meshes $\leftrightsquigarrow$ trees
- allows fast access to the potential additions
- still essentially optimal convergence rates and optimal complexity rates

What could be improved?

- for $f \in \mathcal{U} \subset \mathcal{A}^s_X$ the estimate is only for the whole class $\mathcal{U}$ (i.e. worst case scenario)

- the goal is to achieve instance optimality

$$\|f - \tilde{f}\|_X \leq const \;\; \sigma^{\tau}_N(f)_X$$

- *?* choose the largest coefficients and complete to a tree $\rightsquigarrow$ *not good – too greedy*

- *?* examine all cases $\rightsquigarrow$ *complexity exponential in* N

- the complexity of the algorithm should be linear in N

Adaptive Finite Element Methods

Convergence of AFEM:

◇ **I. Babuška and M. Vogelius** [1984]

- *rates of convergence in the univariate case.*

◇ **W. Dörfler** [1996]

◇ **P. Morin, R. Nochetto, and K. Siebert** [2000]

- *convergence – complexity in terms of the number of iterations*

◇ **P. B., W. Dahmen, and R. DeVore** [2004]

- *sparsity check* $\leftrightsquigarrow$ *coarsening step*
- *optimal convergence rates and complexity of the algorithm*

◇ **R. Stevenson** [2006]

- *shows optimal convergence rates without the coarsening step*

Estimate – Mark – Refine – Solve

The usual adaptive strategy for solving PDEs

- **(1)** start with an initial partition
- **(2)** estimate the global error via local error indicators
- **(3)** if the error is smaller than ϵ, then STOP
- **(4)** mark triangles for refinement
- **(5)** refine and complete the triangulation
- **(6)** solve the discrete problem and go to **(2)**

• the marking strategy is often based on bulk chasing which guarantees that the sum of the local errors for the marked cells is an essential part of the global error.

Basic Properties of Standard AFEM

- guarantied error reduction after each loop
- the rate of convergence cannot be optimal, if the approximate solution $\tilde{u}$ is not sparse at *every* step
- optimal *asymptotic* rates can be maintained, if the bulk constant θ is controlled in such a way that $\tilde{u}$ remains asymptotically sparse
- the solution $u \in \mathcal{A}_X^s$ has to show its asymptotic behavior even for small N
- no instance optimality feasible

Approximation class $\mathcal{A}_X^s := \left\{ u \in X \;:\; \|u\|_{\mathcal{A}_X^s} := \sup_N \; N^s \sigma_N^\tau(u)_X \;\; < \infty \right\}$

AFEM with Coarsening

- **(1)** start with an initial partition
- **(2)** calculate error indicators and mark triangles for refinement
- **(3)** refine and complete the triangulation
- **(4)** calculate error indicators and check the global error
- **(5)** if the error is larger than ϵ, then coarsen the triangulation and continue with **(2)**
- **(6)** stop

Remarks:
• the refinement strategy guarantees an error reduction by a constant factor α
• the coarsening increases the error at most $\alpha/2$ times.

Coarsening

- Calculate error indicators η_E to receive a partition P with global error indicator $\Phi = \Phi(P)$ which gives a global error estimate

$$\|u - \widetilde{u}_P\| \leq \Phi$$

- Use TREE ALGORITHM on the already known $\widetilde{u}_P$ to find $\overline{u}_N$ which is defined on a coarser a partition P', approximates $\widetilde{u}_P$ with an error $\varepsilon \leq \Phi/2$, and is near best (near optimal), i.e.

$$\|u - \overline{u}_N\| \leq C_2 \inf_{\#P' \leq N/c_1} \inf_{v \in \mathcal{S}_{P'}} \|u - v\|$$

AFEM Theorem

Theorem 1 *For any function* $f \in H^{-1}(\Omega)$ *and any* $\epsilon > 0$, **the AFEM algorithm** *produces a partition* P *for which*

$$\Phi(P) \leq \epsilon.$$

If $s > 0$ *and* $u \in \dot{\mathcal{A}}^s$, *and* $f \in \bar{\mathcal{A}}^s$, *then*

$$\#(P) \leq \#(P_0) + C(s)(\|f\|_{\bar{\mathcal{A}}^s}^{1/s} + \|u\|_{\dot{\mathcal{A}}^s}^{1/s})\epsilon^{-1/s}$$

with $C(s) > 0$ *a constant depending only on* s *and the initial partition* P_0. *Moreover, The number of computations used in producing* P *does not exceed*

$$C(s)(\#(P_0) + \|f\|_{\bar{\mathcal{A}}^s}^{1/s} + \|u\|_{\dot{\mathcal{A}}^s}^{1/s})\epsilon^{-1/s}.$$

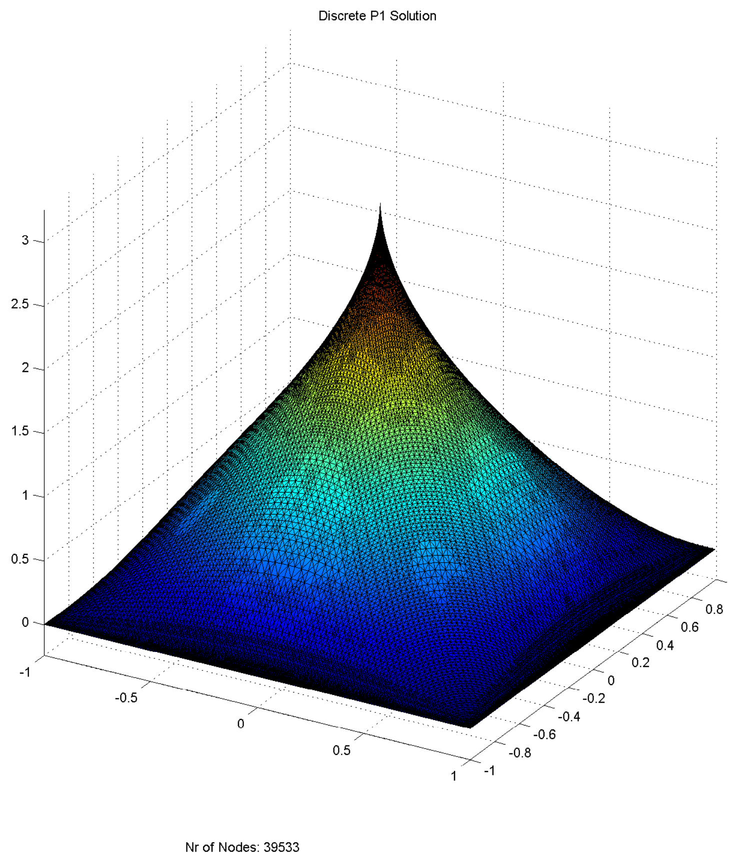
Comments

The theorem is formulated in the framework of the result of **Cohen, Dahmen, and DeVore** [2001,2002] about rates of convergence for wavelet based adaptive methods:

If the solution u can be approximated **(using complete knowledge of u)** *in the energy norm by an n-term wavelet expansion to accuracy $\mathcal{O}(n^{-s})$, $n \to \infty$, then the adaptive method will do the same using* **only** *knowledge of u gained through the adaptive iteration.*

- the coarsening step gives optimal individual performance
- the actual coarsening of the mesh should be performed only if $\widetilde{u}$ is not sparse.
- sparsity is a necessary condition for optimal performance
- the complexity of the sparsity check is approximately the same as the calculation of error indicators

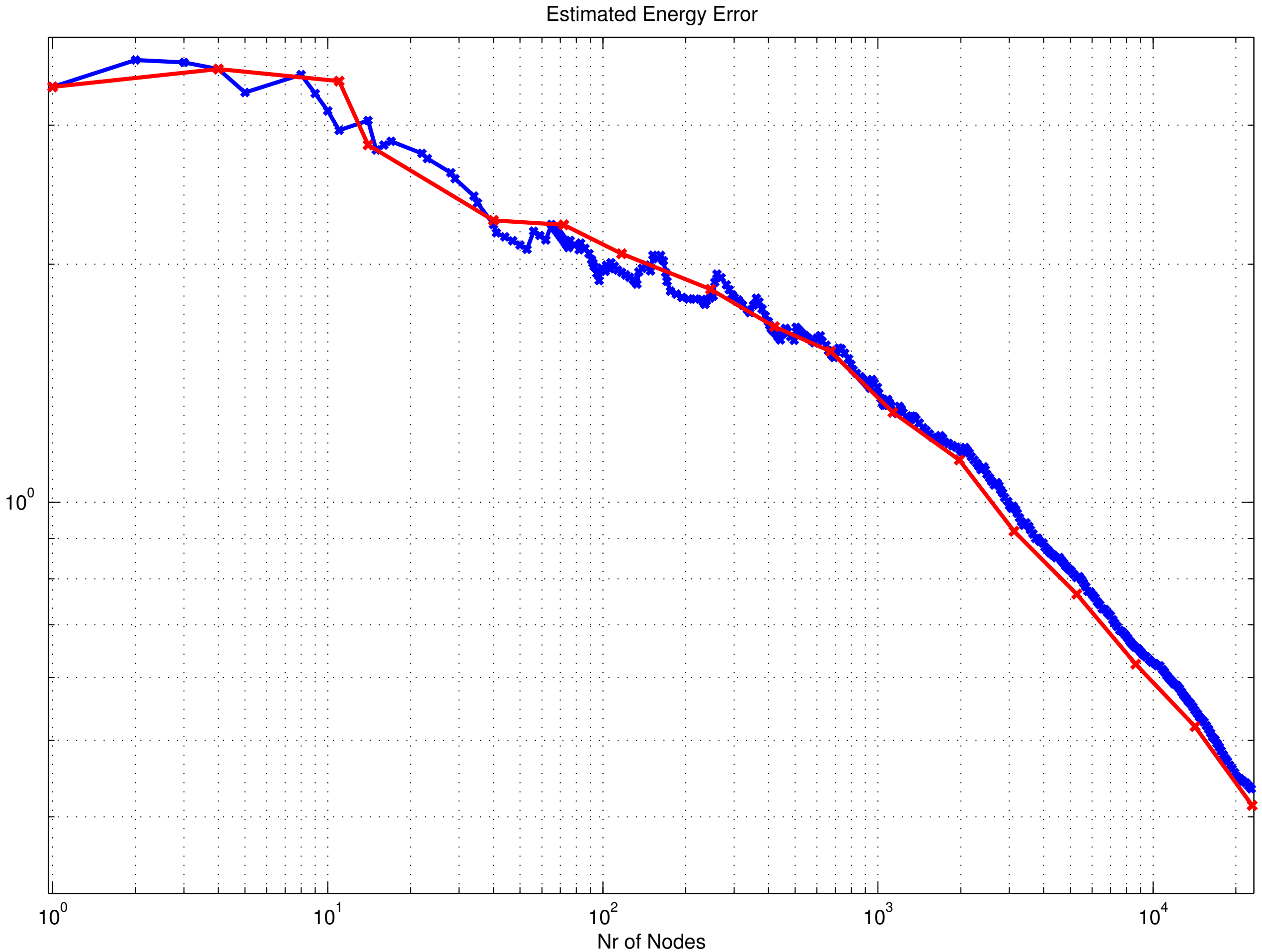
Example - solution



two types of singularities: in the center and near the boundary

- the theoretical value for θ is small

Example - performance of the algorithms



bulk chasing with $\theta = 10^{-2}$ > 200 iterations

AFEM with sparsity check 16 iterations

Near Best Approximation

Best Approximation $\sigma_N(f) := \inf_{g \in \Sigma_N} \|f - g\|$

- There exist absolute constants $c_1, C_2 > 0$ such that
  $\|f - \tilde{f}_N\| \leq C_2\, \sigma_{c_1 N}(f)$ for any N and f.

- For a given smoothness space B^s and for any $f \in B^s$ we have $\sigma_N(f) = \mathcal{O}(N^{-s})$ and $\|f - \tilde{f}_N\| = \mathcal{O}(N^{-s})$.
  There might be functions $f \notin B^s$ for which $\sigma_N(f) = \mathcal{O}(N^{-s})$.

- Approximation Class $\mathcal{A}^s := \{f : \exists C \quad \sigma_N(f) \leq C N^{-s}\}$
  For any $f \in \mathcal{A}^s$ we have $\|f - \tilde{f}_N\| = \mathcal{O}(N^{-s})$.
  The constant C might depend on f. Usually, the estimates are proven for $N > N_0$ where N_0 can be large.

Optimal Rate vs. Near Best

There is an important difference between an approximation that has optimal rate and a near best approximation. In general, "rate" means a number r for which $f \in \mathcal{A}^r$ and therefore the complexity should be $N = \mathcal{O}(\varepsilon^{-\frac{1}{r}})$ with $\varepsilon = \sigma_N(f)$. For example, if

$$N \asymp \varepsilon^{-\frac{1}{r}} |\log \varepsilon|^{\beta} \quad ,$$

then $f \in \mathcal{A}^r$ for any $r \in (0, k)$ regardless of the value of $\beta > 0$. However, it should be clear that the constants in this asymptotic rates differ a lot in cases $\beta = 1$ and $\beta = 1000$. Thus, aiming for the rates only would force us to anticipate the possibility to have very large constants in the second case. At the same time, it would not make a difference for the near best approximation.

Near Best Approximation - comments

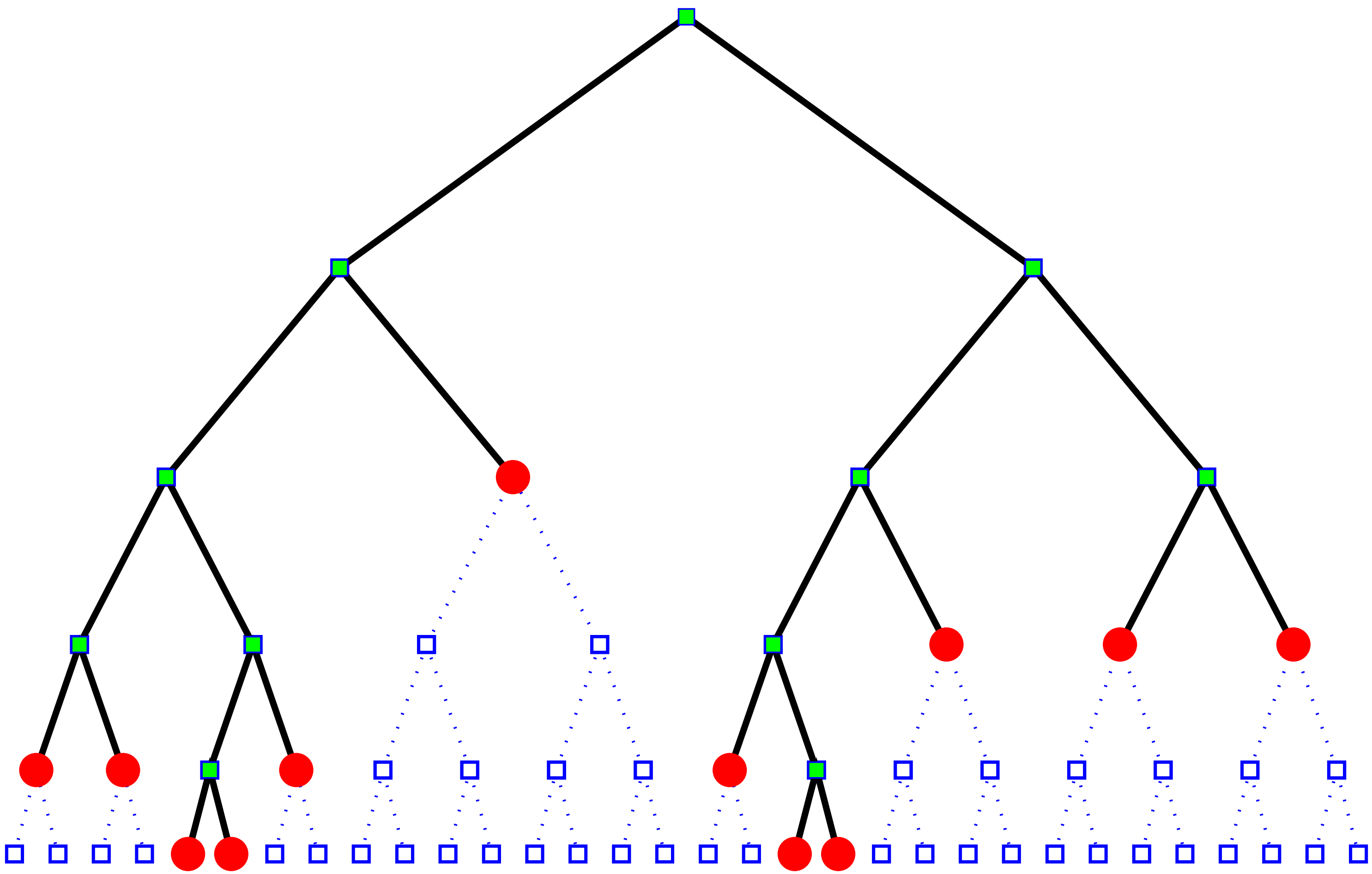
- It is not only the question of computing $\tilde{f}_N$. One have to get the information about f, as well.
- It is not realistic to expect a practical solution to handle infinite possibilities. Therefore, we want to examine adaptive procedures that have finitely many varieties for any fixed N. In Tree Approximation the number of possible N-term bases is exponential on N.
- We want to have an algorithm that finds the near best tree approximation for $\mathcal{O}(N)$ steps, i.e. checking only this number of bases.
- We want the *complexity* constant c_1 and the *approximation* constant C_2 to be close to 1.

Trees

- $T^\star$ – an infinite rooted tree called the master tree
- a node Δ of $T^\star$ may have different number of children
- $\mathcal{L}(T)$ the set of leaf nodes of a subtree T of $T^\star$
- proper subtree T $\leftrightsquigarrow$ partition
  - T contains the root node of $T^\star$
  - if a node $\Delta \in T$ has a child $\Delta' \in T$, then all of the children (from $T^\star$) of Δ are also in T

In some cases we may require more properties for the proper trees

Trees – example



Error Functionals

- a functional e : node $\Delta \in T^\star \;\longrightarrow\;$ error $e(\Delta) \geq 0$
- total error $E(T) := \sum_{\Delta \in \mathcal{L}(T)} e(\Delta)$.

Subadditivity

For any node $\Delta \in T^\star$ if $\mathcal{C}(\Delta)$ is the set of its children, then

$$\sum_{\Delta' \in \mathcal{C}(\Delta)} e(\Delta') \leq e(\Delta) \tag{1}$$

Weak Subadditivity

$\exists\; C_0 \geq 1$ such that for any $\Delta \in T^\star$ and
for any finite subtree $T_\Delta \subset T^\star$ with root node Δ

$$\sum_{\Delta' \in \mathcal{L}(T_\Delta)} e(\Delta') \leq C_0\, e(\Delta) \tag{2}$$

The Idea of Tree Approximation

Initially, for all of the root nodes of $T^{\star}$ we define $\tilde{e}(\Delta) = e(\Delta)$.
Then, for each child Δ_j, $j = 1, \ldots, m(\Delta)$ of Δ

$$\tilde{e}(\Delta_j) := q(\Delta) := \frac{\sum_{j=1}^{m(\Delta)} e(\Delta_j)}{e(\Delta) + \tilde{e}(\Delta)}\, \tilde{e}(\Delta). \qquad (3)$$

Note that $\tilde{e}$ is constant on the children of Δ.

It is useful to define the penalty terms $p(\Delta_j) := \frac{e(\Delta_j)}{\tilde{e}(\Delta_j)}$

The main property we shall need for $\tilde{e}$ is

$$\sum_{j=1}^{m(\Delta)} p(\Delta_j) = p(\Delta) + 1\,. \qquad (4)$$

Adaptive Tree Algorithm

Modified Error $\tilde{e}$:

- initial partition $\rightsquigarrow$ subtree $T_0 \subset T^\star$, $\quad \Delta \in T_0$: $\tilde{e}(\Delta) := e(\Delta)$
- for each child Δ_j of Δ : $\quad \tilde{e}(\Delta_j) := \left(\frac{1}{e(\Delta_j)} + \frac{1}{\tilde{e}(\Delta)} \right)^{-1}$

Adaptive Tree Algorithm

(creates a sequence of trees T_j, $j = 1, 2, \ldots$):

- start with T_0
- subdivide leaves $\Delta \in \mathcal{L}(T_{j-1})$ with largest $\tilde{e}(\Delta)$
  to produce T_j

Remark 1 *To eliminate sorting, we can consider all* $\tilde{e}(\Delta)$ *with* $2^\ell \le \tilde{e}(\Delta) < 2^{\ell+1}$ *to be equally large.*

Adaptive Tree Algorithm - Theorems

[Binev, DeVore, 2004] *Fast Computation in adaptive tree approximation*
Numerische Mathematik (2004) 97: 193–217

Theorem 2 *Let* $\sigma_n(\mathcal{T}) := \min_{\#\mathcal{L}(T)\leq n} E(T)$ *be the best* n*-term tree approximation. Then, there is an absolute constant* $C_1 > 0$ *such that at each step the output tree* T *of the* ADAPTIVE TREE ALGORITHM *satisfies*

$$E(T) \leq C_1 \sigma_n(\mathcal{T}) \qquad \text{whenever } n \leq \#\mathcal{L}(T)/(2K+2).$$

Theorem 3 [2007] *Let the local errors* $e(\Delta)$ *in* $\mathcal{T}$ *satisfy the subadditivity condition. Then at each step of the* NEW TREE ALGORITHM *the output tree* T *satisfies*

$$E(T) \leq \left(1 + \frac{\min\{n, 2(n-N_0)\}}{\#\mathcal{L}(T) - n + 1}\right) \sigma_n(\mathcal{T})$$

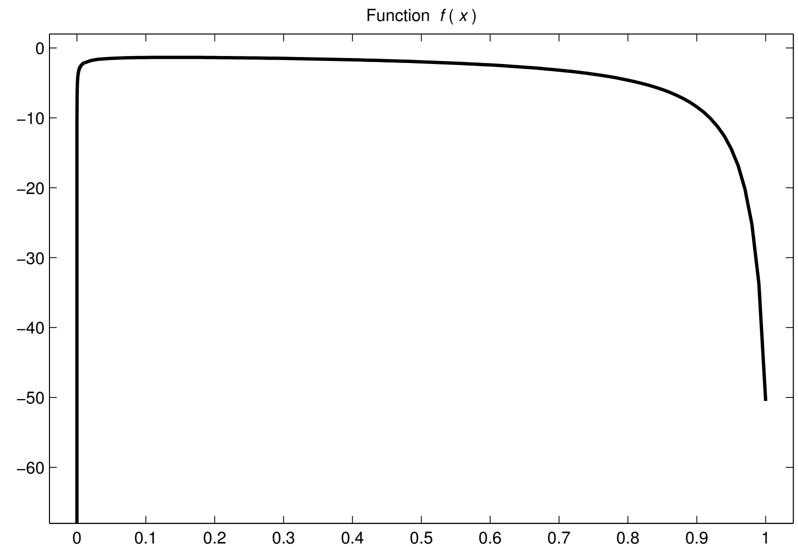
whenever $n \leq \#\mathcal{L}(T)$ *and* N_0 *is the size of the initial partition.*

Numerical Examples

Problem:

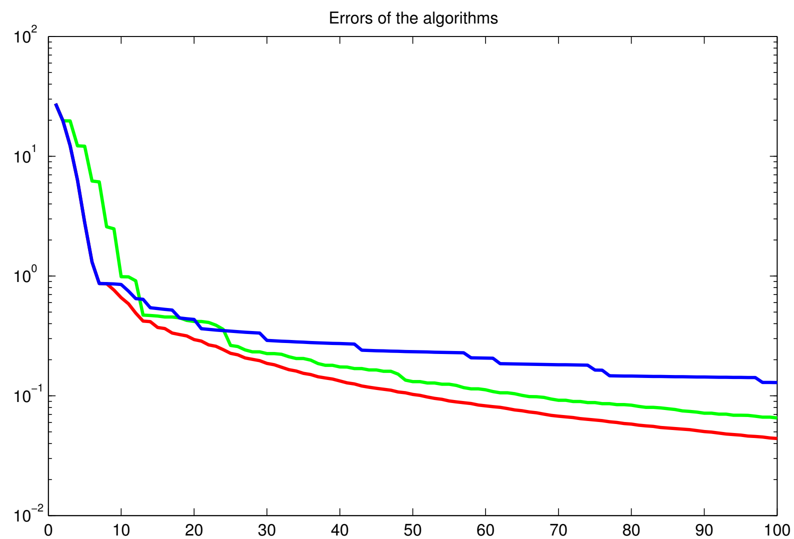
Find an adaptive L_2-approximation of the function

$$f(x) = \frac{x^{-1/2}}{\ln(x/1.02)}$$

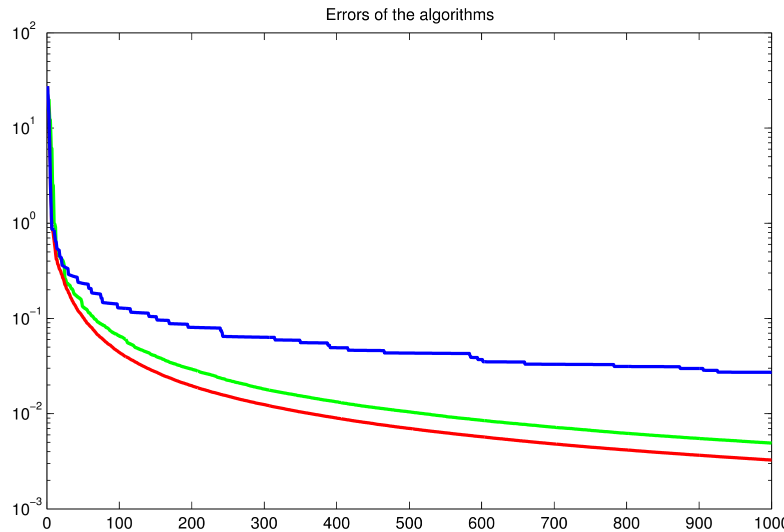


Algorithms:

- greedy
- adaptive - old
- adaptive



Errors for the first 1000 iterations

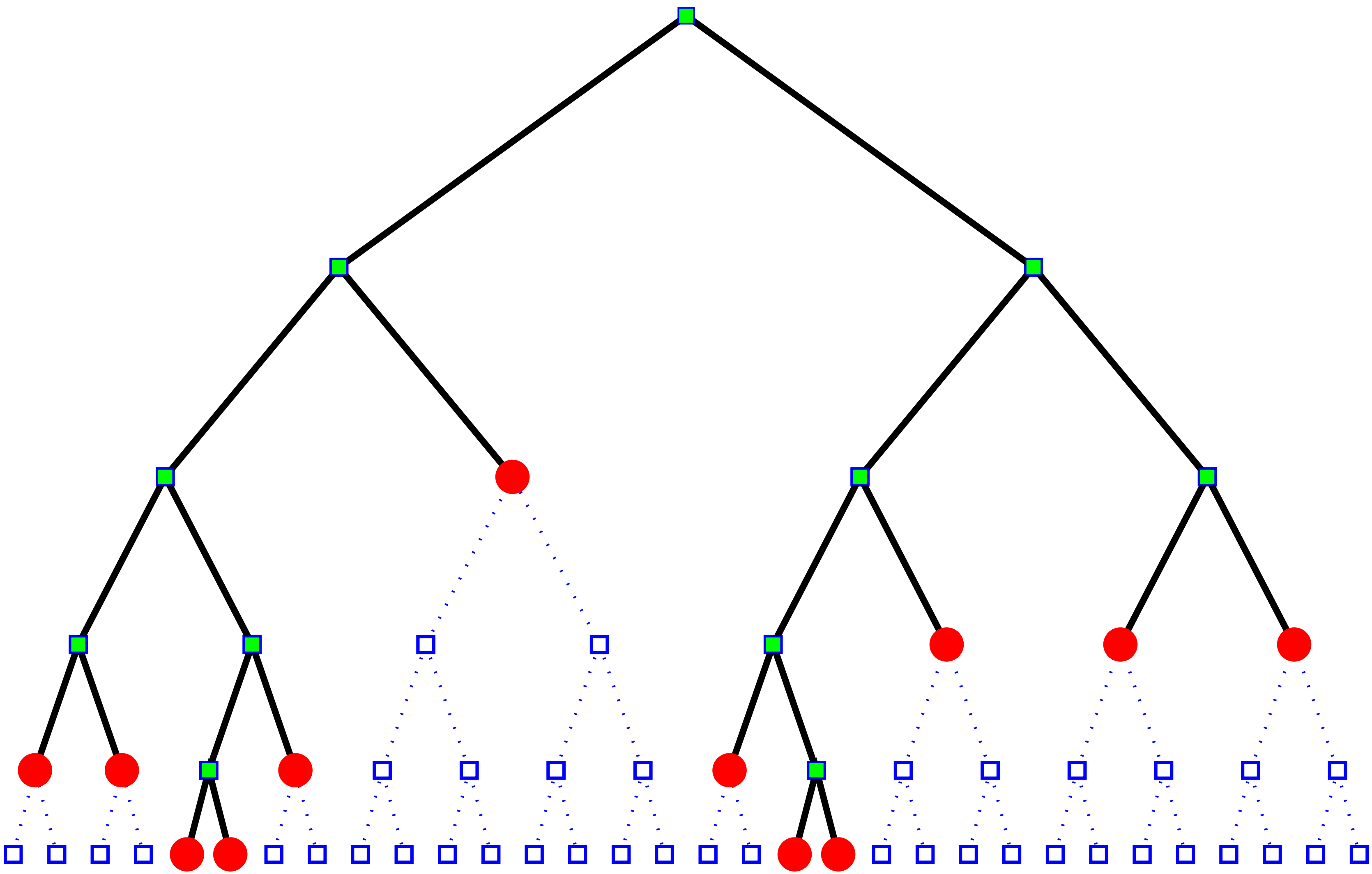


- greedy
- adaptive - old
- adaptive

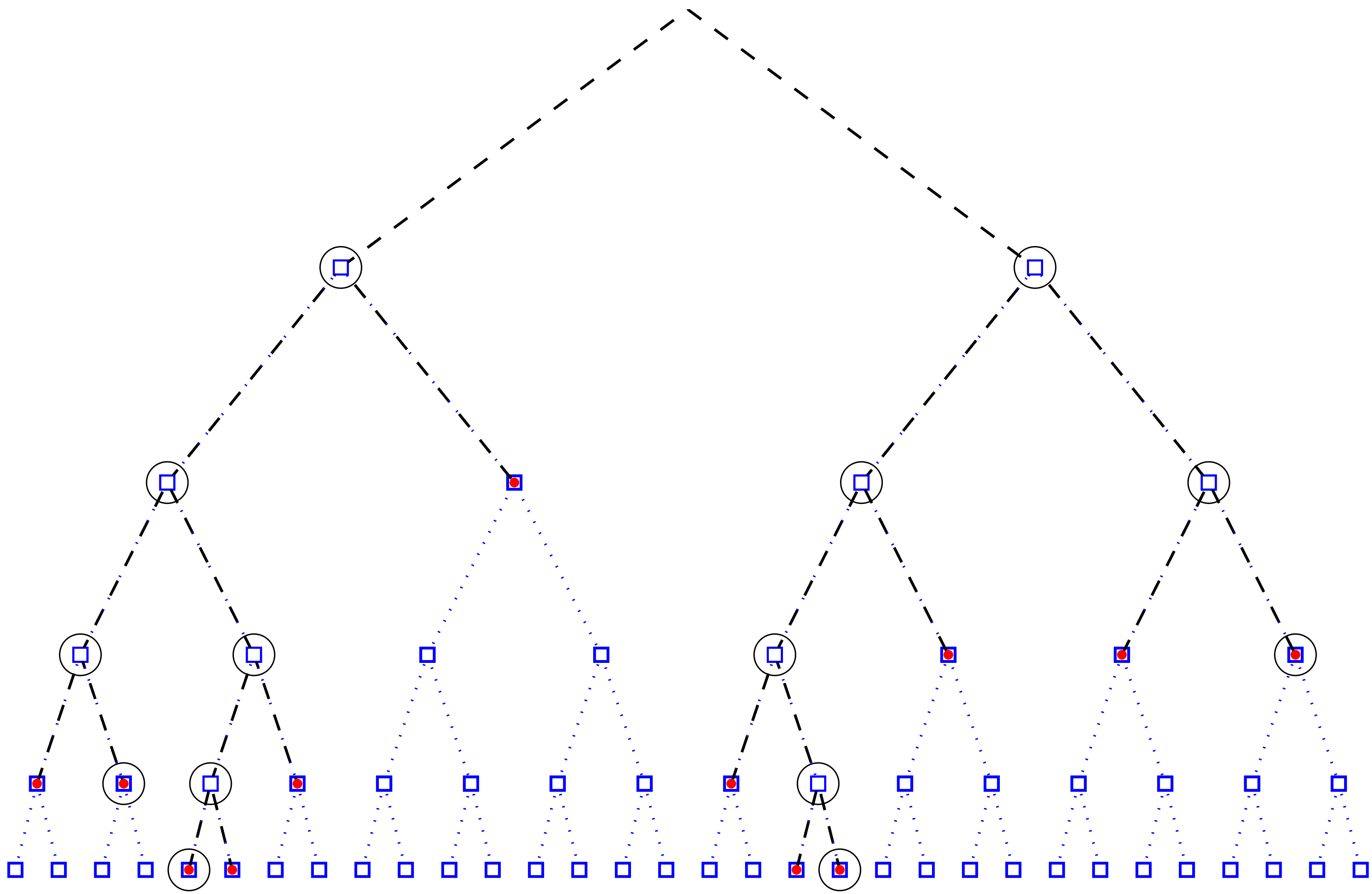
Recent and Future Developments

- improved constants for the case of week subadditivity
- tree based framework for AFEM : combine the tree algorithm with a marking strategy
  only theoretical significance – the constants would be large
- high dimensions : sparse occupancy trees

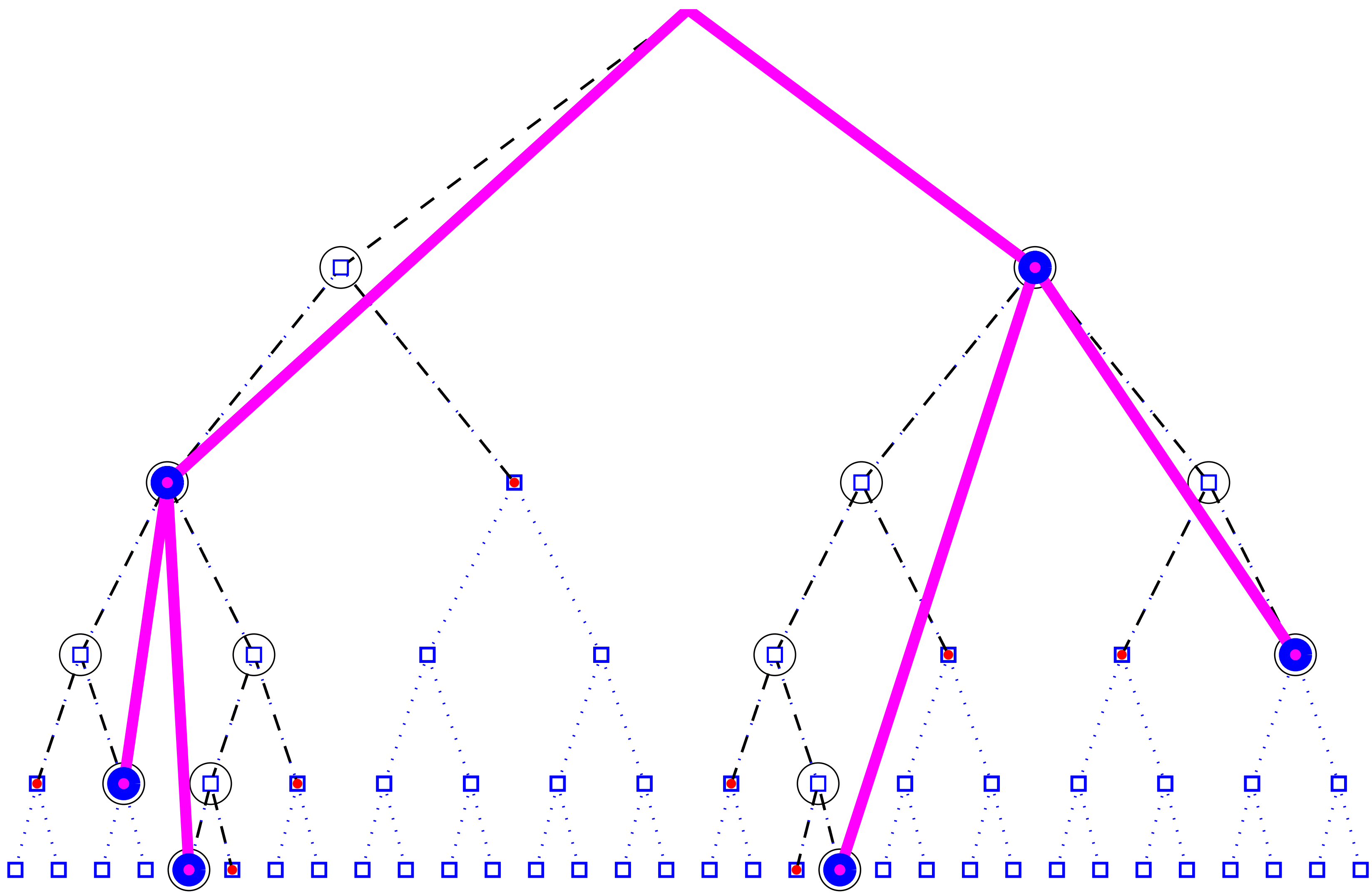
Tree for a Partition



Occupancy Tree



Sparse Occupancy Tree



Sparse Occupancy Trees

Special indexing of the objects that allows fast access and cross level communications in multiresolutional settings (applications include very high dimensional problems > 100)

- adaptive space partition that keeps the information at the level of detail (just) necessary for the problem to be solved;
- key-words: one letter for each level to show the position inside of the parent cell on that level;
- complexity is limited by the number of points;
- the memory is limited to the number of occupied cells at the finest level;
- if required, only a small part of the point cloud can be kept in the fast memory.