Dimension-adaptive Sparse Grids

Jörg Blank

April 14, 2008
1. Data Mining
   - Function Reconstruction

2. Sparse Grids
   - Motivation
   - Introduction

3. Combination technique
   - Motivation
   - Dimension-adaptive

4. Examples

5. Outlook
Data Mining: An use case for sparse grids
- Deduce knowledge from a (large) database
- Recover a function from test results
- Cope with measurement errors
Function Reconstruction
Function Reconstruction
Datasets

- Higher dimensions are common.

\[ S = \{(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}\}_{i=1}^M \]

- \( d \)-dimensional dataset with \( M \) entries
- \( y_i \) - function value
- Restricting \( y_i \) to an arbitrary number of classes is possible
We assume that the data points are evaluations of an unknown function \( f \)

**Definition**

Wanted: a function

\[
y = f(x_1, x_2, \ldots, x_d)
\]

\( f \in V \)

where \( V \) is a function space over \( \mathbb{R}^d \)
Regularisation

Recover this function as good as possible!

\[
\min_{f \in V} R(f)
\]

\[
R(f) = \frac{1}{M} \sum_{i=1}^{M} \Psi(f(x_i), y_i) + \lambda \Phi(f)
\]

- \(\Psi(x, y) = (x - y)^2\)
- \(\Phi(f) = \|\nabla f\|^2\)
Discretization

- We confine $V$ to a discrete space $V_N$
- A function $f_N \in V_N$ can now be written as:
  \[
  f_N = \sum_{j=1}^{N} \alpha_j \phi_j(x)
  \]
  
  weights: $\{\alpha_i\}_{i=1}^{N}$
  
  a base: $\Phi_N = \{\varphi_i\}_{i=1}^{N}$

The choice of basis functions has a major impact on viability and accuracy of this approach.
Differentiation of $R(f_N)$ now yields for $k = 1 \ldots N$:

$$
\sum_{j=1}^{N} \alpha_j \left[ M \lambda (\nabla \varphi_j, \nabla \varphi_k)_{L_2} + \sum_{i=1}^{M} \varphi_j(x_i) \cdot \varphi_k(x_i) \right] = \sum_{i=1}^{M} y_i \varphi_k(x_i)
$$
Which is a system of linear equations with N unknowns and N equations and can be written in matrix form:

$$(\lambda C + B \cdot B^T)\alpha = By$$

This system is symmetric and positiv definite and can be solved using a standard solver like Conjugated Gradients method.
Choice of base functions

- Nodal basis yields: $O(n^d)$
- Not viable even for medium dimension counts
- Solution: Use less grid points!
Hat functions

\[ \phi(x) = \begin{cases} 
1 - |x| & \text{if } x \in [-1, 1] \\
0 & \text{otherwise} 
\end{cases} \]

\[ \phi_{l,i}(x) = \phi\left(\frac{x - i \cdot h_l}{h_l}\right) = \phi\left(\frac{x - i \cdot 2^{-l}}{2^{-l}}\right) = \phi(x \cdot 2^l - i) \]
An hierarchical basis

Figure: Datamining mit Dünnen Gittern, Pflüger
An hierarchical basis

Figure: Datamining mit Dünnen Gittern, Pflüger
This function can be enhanced to a d-linear function

**Definition**

\[ \phi_{l,i}(\mathbf{x}) := \prod_{j=1}^{d} \phi_{l,j}(x_j) \]

**Figure: AWR2, Bungartz**

**Pagoda**

Data Mining
Sparse Grids
Combination technique
Examples
Outlook
Extending the hierarchical pattern yields a subgrid scheme (in 2D case):

Figure: AWR2, Bungartz
Use grids with large contribution to the solution and few gridpoints.

Regular Sparse Grids have a far better behaviour: \( O(n \times \log(n)^{d-1}) \)
This leads to the well known pattern:

Figure: AWR2, Bungartz
Working with Sparse Grids involves a lot of overhead:

Figure: AWR, Bungartz
It is possible to create a similar structure by combining multiple, much coarser full grids.

\[ \Omega_3^{(1)} = \Omega_{(3,1)} + \Omega_{(2,2)} + \Omega_{(1,3)} - \Omega_{(2,1)} - \Omega_{(1,2)} \]

**Figure:** AWR2, Bungartz
For the 2D case:

\[
 f_n^{(c)}(\mathbf{x}) := \sum_{|\mathbf{l}|_1=n+1} f_{\mathbf{l}}(\mathbf{x}) - \sum_{|\mathbf{l}|_1=n} f_{\mathbf{l}}(\mathbf{x})
\]
Or more general:

**Definition**

\[
f_n^{(c)}(x) := \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{|\mathbf{l}|_1 = n-q} f_l(x)
\]

This formula is derived from the combinatorial 'inclusion-exclusion' principle!
Characteristics

- Existing codes for full grids can be used
- Embarrassingly parallel: Each subgrid can be computed without communication
- Still less points than regular nodal grids
- Only for regular sparse grids!
Generalisation
Generalisation

Allowing all subspace combinations would be a bad idea!

**Definition**

Admissibility

\( \mathcal{I} \) - set of selected indices

\[ k \in \mathcal{I} \text{ and } j \leq k \implies j \in \mathcal{I} \]
Combination

Example: Combining a (2, 3) and a (3, 1) grid
Adaptivity

Start with the smallest grid: \( l = (1, \ldots, 1) \)

Successively add new grid indexes:

- new index set must remain admissible
- new index must provide a significant contribution to the general solution
How to measure the contribution of an index?

- Calculate $\varepsilon = R(f)$
- The regularisation term may be omitted
- A large $\varepsilon$ indicates a bad fitting $\rightarrow$ further refinement needed

How to measure the contribution of an index?
Algorithm

- Initialize index set $I = \{1\}$
- Initialize old index set $O = \{\}$
- Solve problem on $1$
- while global $\varepsilon >$ bound
  - Choose $i \in I$ with largest $\varepsilon_i$
  - Refine in all dimensions, if admissible in $O$
  - Move $i$ to $O$
  - Calculate problems and $\varepsilon$ on new indexes
  - Update global $\varepsilon$
Why dimension-adaptivity?

Consider:

\[ f(\mathbf{x}) = f_1(x_1) + f_2(x_2) + \ldots + f_d(x_d) \]

- All dimensions are totally independent
- It is possible to reconstruct the function with very little grid points
- The introduced algorithm can produce a near optimal result
Additive functions

\[ f(x_1, x_2) = e^{-x_1^2} + e^{-x_2^2} \]
Multiplicative functions

\[ f(x_1, x_2) = e^{-(x_1^2 + x_2^2)} \]
Mixed functions

\[ f(x_1, x_2) = e^{-x_1^2} + e^{-(x_1^2 + x_2^2)} \]
Outlook

- Up to 15 dimensions possible in real world applications
- Or more if not too many dimensions hold information...
- It is possible to use other coefficients for combination
  - One possibility: minimize difference to ’normal’ sparse grids
  - This is called *opticom* technique by Garcke
  - Additional computational complexity, but better stability
Other application areas:

- Integration
  An alternative for Monte-Carlo-Integration for high dimensional integrals

- Solving PDEs
  Possibility to solve PDEs in high dimensions or using a space-time-discretization

- Of course there is still the ’real’ sparse grid technique
Thank you for your attention.