FRIEDRICH-ALEXANDER-UNIVERSITÄT ERLANGEN-NÜRNBERG
INSTITUT FÜR INFORMATIK (MATHEMATISCHE MASCHINEN UND DATENVERARBEITUNG)

Lehrstuhl für Informatik 10 (Systemsimulation)

A parallel K-SVD implementation for CT image denoising

D. Bartuschat, A. Borsdorf, H. Köstler, R. Rubinstein, M. Stürmer

Lehrstuhlbericht 09-01

A parallel K-SVD implementation for CT image denoising

D. Bartuschat, A. Borsdorf, H. Köstler, R. Rubinstein, M. Stürmer

Abstract

In this work we present a new patch-based approach for the reduction of quantum noise in CT images. It utilizes two data sets gathered with information from the odd and even projections respectively that exhibit uncorrelated noise for estimating the local noise variance and performs edge-preserving noise reduction by means of the K-SVD algorithm. It is an efficient way for designing overcomplete dictionaries and finding sparse representations of signals from these dictionaries. For image denoising, the K-SVD algorithm is used for training an overcomplete dictionary that describes the image content effectively. K-SVD has been adapted to the non-gaussian noise in CT images. In order to achieve close to real-time performance we parallelized parts of the K-SVD algorithm and implemented them on the Cell Broadband Engine Architecture (CBEA), a heterogenous, multicore, distributed memory processor. We show denoising results on synthetic and real medical data sets.

1 Introduction

A major field in CT research has been the investigation of approaches for image noise reduction. Reducing noise corresponds to an increase of the signal-to-noise ratio (SNR). Consequently, the patient's x-ray exposure can be reduced, since a smaller radiation dose sufficies for acquiring the x-ray projection data, from which CT images are then reconstructed.

The main source of noise in CT is quantum noise. It results from statistical fluctuations of x-ray quanta reaching the dectector. Noise in projection data is known to be poisson-distributed. However, during the reconstruction process, by means of the (most common) filtered backprojection method, noise distribution is changed. Due to complicated dependencies of noise on scan parameters and on spatial position [15], noise distribution in the final CT image is usually unknown and noise variance in CT images is spatially changing. Additionally, strong directed noise in terms of streak artifacts may be present [13].

Several different algorithmic approaches for CT noise reduction exist, like methods that suppress noise in projection data before image reconstruction. Other algorithms reduce noise during CT reconstruction by optimizing statistical objective functions [17, 16]. Another common area of research includes the development of algorithms for noise reduction in reconstructed CT images. These methods are required to reduce noise while preserving edges, as well as small structures, that might be important for diagnosis. Standard edge-preserving methods in the spatial domain are partial differential equation (PDE) based methods [20, 22]. In the frequency domain, wavelet-based methods for denoising are prevalently investigated [9, 6].

We proposed CT denoising methods that deal with the complicated noise properties by utilizing two spatially identical images containing uncorrelated noise. In order to differentiate between structure and noise in a certain region, correlation can be computed and used as a measure for similarity, what has been done for wavelet based methods in [3, 5]. Additionally, position dependent noise estimation can be performed by computing the noise variance, as in [4]. These wavelet based methods are compared to PDE based denoising methods with nonlinear isotropic and anisotropic diffusion[18, 19].

In this work, we present a new patch-based approach for the reduction of quantum noise in CT images. It utilizes again the idea of using two data sets with uncorrelated noise for estimating the local noise variance and performs edge-preserving noise reduction by means of the K-SVD algorithm that is an efficient algorithm for designing overcomplete dictionaries and then finding sparse representations of signals from these dictionaries [11]. For image denoising, the K-SVD algorithm is used for training an overcomplete dictionary that describes the image content effectively. This method performs very well for removing additive gaussian noise from images and has been adapted to the non-gaussian noise in CT images.

In order to achieve close to real-time performance, the algorithm is – in addition to a C++ implementation – also parallelized on the Cell Broadband Engine Architecture (CBEA), a heterogenous, multicore, distributed memory processor. Its special architecture explicitly parallelizes computations and transfer of data and instructions. For efficient code, this kind of parallelism has to be exploited, together with data-level parallelism in the form of vector processing and thread-level parallelism by means of software threads.

We introduce the idea of sparse representations and the K-SVD algorithm in section 2 and show how it can be used for CT image denoising. In section 3 we discuss the Cell implementation and in section 4 we present qualitative and quantitative denoising and performance results both for synthetic and real medical data sets. Our work is summarized in section 5.

2 Methods

In the following, we describe sparse representations of signals, and methods for finding these, given an overcomplete dictionary, which contains a set of elementary signals.

In combination with a dictionary representing the noise-free part of a noisy signal, sparse representations can be used for signal denoising. A dictionary that has this property, can be obtained by performing dictionary training with the K-SVD algorithm. The advantage of this algorithm is, that it can be used for adapting the dictionary to the image to be denoised. This trained dictionary can then be used for denoising the image, on which it was trained. This is possible, since stationary Gaussian white noise is not learned by that dictionary.

2.1 Sparse Representations and the OMP Algorithm

By extending a set of elementary vectors beyond a basis of the signal vector space, signals can be compactly represented by a linear combination of only few of these vectors. The problem of finding the sparsest representation of a signal $\mathbf{y} \in \mathbb{R}^n$ among infinitely many solutions of this underdetermined problem reads as

$$\min_{\mathbf{a}} ||\mathbf{a}||_0 \text{ subject to } \mathbf{Da} = \mathbf{y}, \tag{1}$$

where $||\cdot||_0$ denotes the ℓ_0 - seminorm that counts the nonzero entires of a vector $||\mathbf{a}||_0 = \sum_{j=0}^{K} |\mathbf{a}_j|^0$. The given full rank matrix $\mathbf{D} \in \mathbb{R}^{n \times K}$ represents the overcomplete dictionary, which has a higher number of atoms K than the dimension of the atoms n. For known error tolerance ϵ, eq.(1) can be reformulated as

$$\min_{\mathbf{a}} ||\mathbf{a}||_0 \text{ subject to } ||\mathbf{Da} - \mathbf{y}||_2 \leq \epsilon. \tag{2}$$

This sparsest approximation allows a more compact representation of signals. It can be deployed for removing noise from signals, if ϵ fits the present noise.

OMP Algorithm: Exactly determining the sparsest representation of signals is an NP-hard combinatorial problem [8]. A simple greedy algorithm, the orthogonal matching pursuit (OMP) algorithm [8] depicted in algorithm 1, is used for performing sparse-coding approximately by sequentially selecting dictionary atoms.

Algorithm 1 OMP algorithm: $\mathbf{a} = \text{OMP}(\mathbf{D}, \mathbf{x}, \epsilon \text{ or } L)$

```
1 Init: Set $\mathbf{r}_0 = \mathbf{x}$, $j = 0$, $I = \emptyset$
2 while $j < L$ && $\|\mathbf{r}\|_2^2 > \epsilon$ do
3     $\mathbf{p} = \mathbf{D}^T\mathbf{r}_{j-1}$
4     $\hat{k} = \underset{k}{\text{argmax}}\,|\mathbf{p}|$
5     $I = I \cup \hat{k}$
6     $\mathbf{a}_I = \mathbf{D}_I^{+}\mathbf{x}$
7     $\mathbf{r}_j = \mathbf{x} - \mathbf{D}_I\mathbf{a}_I$
8     $j = j + 1$
9 end while
```

Here, I is a data structure for storing a sequence of chosen atoms' indices. $\mathbf{D}_I$ denotes a matrix that comprises only those atoms of the dictionary, which have been selected in the previous steps. In line 3, the projection of the residual on the dictionary is computed. After that, the greedy selection step is performed in line 4. It selects the index $\hat{k}$ of the largest element of $\mathbf{p}$, which corresponds to the the atom with maximal correlation to the residual. Having added the new atom index to the set I, the orthogonalization step in line 6 follows. It ensures that all selected atoms are linearly independent [8]. Here, $\mathbf{D}_I^{+}$ denotes the pseudo-inverse of $\mathbf{D}_I$. In step 7, the new residual is computed, which is orthogonal to all previously selected atoms. When the stopping criterion is fulfilled, the coefficients have been found and the algorithm terminates. The stopping criterion is fulfilled, if either the sparse representation error is smaller than the error tolerance, or the maximum number of atoms L has been found.

OMP is guaranteed to converge in finite-dimensional spaces in a finite number of iterations [8]. It consecutively removes those signal components, that are highly correlated to few dictionary atoms. At the first iterations, the representation error decays quickly, as long as highly correlated atoms have not yet been chosen. Later, when all similar atoms have been selected, the residual decays only slowly and the residuals behave like realizations of white noise [8]. Therefore, signal approximation is truncated, depending on the error tolerance ϵ of (2).

Batch-OMP Algorithm: The Batch-OMP algorithm [21], summarized in algorithm 2, accelerates the OMP algorithm for large sets of signals by two techniques: It replaces the computation of the pseudoinverse in the orthogonalization step, which is done in OMP by a singular value decomposition (SVD), by a progressive Cholesky update. And it performs pre-computation of the gram matrix $\mathbf{G} = \mathbf{D}^{\mathbf{T}}\mathbf{D}$, which results in lower-cost computations at each iteration.

The orthogonalization step and the following residual update in the OMP algorithm of the previous section, can be written as

$$\mathbf{r} = \mathbf{x} - \mathbf{D}_I(\mathbf{D}_I^T\mathbf{D}_I)^{-1}\mathbf{D}_I^T\mathbf{x}. \tag{3}$$

Due to the orthogonalization step, the matrix $(\mathbf{D}_I^T\mathbf{D}_I)$ stays non-singular. This matrix is symmetric positive definite (SPD) and can therefore be decomposed by means of a Cholesky decomposition. To this matrix, a new row and column are added at each iteration, since a

Algorithm 2 $\mathbf{a}$ = Batch-OMP $\left(\mathbf{p}^0 = \mathbf{D}^T\mathbf{x},\, \epsilon^0 = \mathbf{x}^T\mathbf{x},\, \mathbf{G} = \mathbf{D}^T\mathbf{D}\right)$

```
1  Init: Set $I = \emptyset$, $\mathbf{L} = [1]$, $\mathbf{a} = 0$, $\delta^0 = 0$, $\mathbf{p} = \mathbf{p}^0$, $n = 1$
2  while $\epsilon^{n-1} > \epsilon$ do
3    $\hat{k} = \underset{k}{\text{argmax}}\, |\mathbf{p}_{n-1}|$
4    $I^n = I^{n-1} \cup \hat{k}$
5    if $n > 1$ then
6      $\mathbf{w}$ = Solve for $\mathbf{w}\left\{\mathbf{L}^n\mathbf{w} = \mathbf{G}_{I^n,\hat{k}}\right\}$
7      where $\mathbf{L}^n = \begin{bmatrix} \mathbf{L}^{n-1} & 0 \\ \mathbf{w}^T & \sqrt{1-\mathbf{w}^T\mathbf{w}} \end{bmatrix}$
8    end if
9    $\mathbf{a}_{I^n}$ = Solve for $\left\{\mathbf{L}^n\left(\mathbf{L}^n\right)^T\mathbf{a}_{I^n} = \mathbf{p}^0_{I^n}\right\}$
10   $\beta = \mathbf{G}_{I^n}\mathbf{a}_{I^n}$
11   $\mathbf{p}_n = \mathbf{p}^0 - \beta$
12   $\delta^n = \mathbf{a}^T_{I^n}\beta_{I^n}$
13   $\epsilon^n = \epsilon^{n-1} - \delta^n + \delta^{n-1}$
14   $n = n + 1$
15 end while
```

new atom is there added to $\mathbf{D}_I$. Thus, a new row is added at each iteration to its decomposed lower triangular Cholesky matrix $\mathbf{L}$.

The residual in the OMP algorithm does not need to be computed explicitly at each iteration. Instead, only $\mathbf{D}^T\mathbf{r}$, the projection of the residual on the dictionary, is required. This can be exploited by directly computing $\mathbf{D}^T\mathbf{r}$ instead of the residual, yielding

$$\begin{aligned} \mathbf{p} = \mathbf{D}^T\mathbf{r} &= \mathbf{p}^0 - \mathbf{G}_I\left(\mathbf{D}_I\right)^{+}\mathbf{x} && (4) \\ &= \mathbf{p}^0 - \mathbf{G}_I(\mathbf{G}_{I,I})^{-1}\mathbf{p}^0_I. && (5) \end{aligned}$$

with $\mathbf{p}^0 = \mathbf{D}^T\mathbf{x}$. This is done in line 11 of algorithm 2.
One index I at a matrix indicates that only a sub-matrix is considered containing the columns of the matrix which are indexed by I. For a vector this means that only those elements are selected, which are indexed by I. Two indices I, I denote that the sub-matrix consists of indexed rows and columns.
From this, the projection can be computed without explicitly computing $\mathbf{r}$. The update step requires now only multiplication by the $\mathbf{G}_I$, which can be selected from the pre-computed matrix $\mathbf{G}$. The matrix $\mathbf{G}_{I,I}$ is inverted using the progressive Cholesky update yielding $\mathbf{L}^n\left(\mathbf{L}^n\right)^T$. This progressive Cholesky update is performed in lines 5 – 8. The new row of the decomposed lower triangular Cholesky matrix is computed from the previous lower triangular Cholesky matrix, and a vector that contains gram matrix entries from the $\hat{k}$-th column of the gram matrix and rows corresponding to the previously selected atoms I, by means of forward substitution. For more details, see [21].
The nonzero element coefficient vector $\mathbf{a}_{I^n}$ can then be computed from the formula

$$\mathbf{L}^n\left(\mathbf{L}^n\right)^T\mathbf{a}_{I^n} = \mathbf{p}^0_{I^n}, \quad (6)$$

by means of a forward- and backward substitution, as done in line 9. Here, n denotes the iteration counter.

However, when the OMP is used for solving an error-constrained sparse approximation problem, the residual is required to check the termination criterion. Therefore, an incremental formula for the ℓ^2 norm of the residual has been derived in [21]. This formula is used in line